UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

PhD Thesis

New Results on Hash Functions and Hashing-Based Algorithms

Jakob Bæk Tejs Houen

Advisor: Mikkel Thorup

Submitted: May 26, 2023

This thesis has been submitted to the PhD School of The Faculty of Science, University of Copenhagen

Abstract

With the increasing size of datasets in the era of machine learning and AI, exact computations on these datasets have become increasingly infeasible. Nevertheless, in many applications, approximate answers are sufficient. This motivates the question of whether efficient algorithms and data structures can be designed to provide reliable approximate answers on huge datasets.

Randomization, particularly through *hash functions*, is a powerful tool for simplifying these algorithms. However, existing analyses of such algorithms regularly assume *fully random* or *highly independent* hash functions, which ignores the issue of efficiency in practice. While there exist efficient *theoretical* constructions of highly independent hash functions none of them are efficient in *practice*. This thesis addresses these issues by providing new analyses of practical *tabulation-based* hashing schemes. First in [HT22], we obtain a near-optimal understanding of concentration guarantees for *simple tabulation* hashing of hash-based sums by bounding the moments. This analysis allows us to show that *mixed tabulation* hashing has strong concentration guarantees for hash-based sums that closely match those of fully random hashing. Furthermore in [HT23], we demonstrate how these insights can be used to implement a *sparse Johnson-Lindenstrauss transform*, a widely used technique in high-dimensional data analysis.

Additionally in [AK20], we study the problem of *approximate set similarity search*, where the goal is to build a data structure that can identify similar sets in a database of known sets or report that the database contains no similar set. We show that our algorithm is optimal for all hashing-based data structures for random sets, providing a comprehensive solution to this problem.

Finally in [AKT21], we consider the problem of *dynamic load balancing* in an environment where both balls and bins can be added and removed. We assume that each bin has a capacity of C, that is, each bin can contain at most C balls. We construct a data structure that in expectation moves $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin. Where f is the fraction of non-full bins in the following simpler probabilistic problem: Place the balls into bins with capacity C, one ball at the time, where each ball picks a uniformly random non-full bin. We also solve this simpler problem and provide a tight bound for f. In order to prove these results, we needed a new result in probability theory which was proven in [AAHT22].

Dansk resumé

Med den stadig stigende størrelse af datasæt i tidsalderen for machine learning og AI er præcise beregninger på disse datasæt blevet stadig mere udfordrende. Ikke desto mindre er approksimative svar tilstrækkelige i mange anvendelser. Dette motiverer spørgsmålet om, hvorvidt der kan designes effektive algoritmer og datastrukturer til at levere pålidelige approksimative svar på enorme datasæt.

Randomisering, især gennem *hashfunktioner*, er et kraftfuldt værktøj til at forenkle disse algoritmer. Dog antager eksisterende analyser af sådanne algoritmer ofte *fuldstændigt tilfældige* eller *meget uafhængige* hashfunktioner, hvilket ignorerer spørgsmålet om effektivitet i praksis. Selvom der eksisterer effektive *teoretiske* konstruktioner af meget uafhængige hashfunktioner, er ingen af dem effektive i *praksis*. Denne afhandling angriber disse problemer ved at give nye analyser af praktiske *tabulation*-baserede hashingsmetoder. I [HT22] opnår vi en næsten optimal forståelse af koncentrationsgarantier for *simpel tabulation hashing* af hashbaserede summer ved at begrænse momenterne. Denne analyse gør det muligt for os at vise, at *mixed tabulation hashing* har stærke koncentrationsgarantier for hashbaserede summer, der er tæt på at matche dem fra fuldstændigt tilfældig hashing. Derudover viser vi i [HT23], hvordan disse nye ideer kan anvendes til at implementere en *spare Johnson-Lindenstrauss transformation*, en bredt anvendt teknik inden for højdimensionel dataanalyse.

I [AK20] studerer vi problemet med *approksimativ similaritetssøgning i mængder*, hvor målet er at opbygge en datastruktur, der kan identificere lignende mængder i en database med kendte mængder eller retunerer, at databasen ikke indeholder nogen lignende mængder. Vi viser, at vores algoritme er optimal for alle hashbaserede datastrukturer for tilfældige mængder.

Til sidst i [AKT21] betragter vi problemet med *dynamisk load balancing* i et dynamisk system, hvor både bolde og spande kan tilføjes og fjernes. Vi antager, at hver spand har en kapacitet på C, dvs. at hver spand kan indeholde højst C bolde. Vi konstruerer en datastruktur, der i forventning flytter $O(1/f)$ bolde, når en bolde indsættes eller fjernes, og $O(C/f)$ bolde, når en spand indsættes eller fjernes. Hvor f er brøkdelen af ikke-fulde spande i det følgende simplere sandsynlighedsproblem: Placer boldene i spandene med kapacitet C, en bold ad gangen, hvor hver bold vælger en uniformt tilfældig ikke-fuld spand. Vi løser også dette simplere problem og giver en tæt grænse for f. For at bevise disse resultater har vi brug for et nyt resultat inden for sandsynlighedsteori, der blev bevist i [AAHT22].

Preface

The *General rules and guidelines for the PhD programme*[1] at Faculty of Science, University of Copenhagen, adopted in March 2023 states that *"A thesis may either be written as a monograph, or as a synopsis with manuscripts of papers or already published papers attached."* The present thesis has adopted the latter format, that is, the thesis is written as a synopsis with manuscripts of papers attached in the appendix.

In order to limit the scope of the thesis, I have chosen to only present a selected subset of my work. I will present four papers which focus on different aspects of hashing which has been the main area of focus for my research during my PhD.

For completeness, this preface contains a brief introduction to all the papers authored by me, even those that are not included in this thesis. Over the course of my PhD, I have had the pleasure of being co-author of 13 papers, 9 of which are published or are accepted at peer-reviewed conferences or journals, while the remaining 4 are currently in the submission process. The complete list of these papers is presented below for reference.

List of Papers

[AAKR21] Anders Aamand, Mikkel Abrahamsen, Jakob Bæk Tejs Knudsen, and Peter Michael Reichstein Rasmussen. "Classifying Convex Bodies by Their Contact and Intersection Graphs". In: *37th International Symposium on Computational Geometry, SoCG 2021, June 7-11, 2021, Buffalo, NY, USA (Virtual Conference)*. Ed. by Kevin Buchin and Éric Colin de Verdière. Vol. 189. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 3:1–3:16.

[AAHT22] Anders Aamand, Noga Alon, Jakob Bæk Tejs Houen, and Mikkel Thorup. "On sums of monotone random integer variables". In: *Electronic Communications in Probability* 27 (2022), pp. 1–8.

[ADKK+22] Anders Aamand, Debarati Das, Evangelos Kipouridis, Jakob Bæk Tejs Knudsen, Peter M. R. Rasmussen, and Mikkel Thorup. "No Repetition: Fast and Reliable Sampling with Highly Concentrated Hashing". In: *Proc. VLDB Endow.* 15.13 (2022), pp. 3989–4001.

[AKKR+20] Anders Aamand, Jakob Bæk Tejs Knudsen, Mathias Bæk Tejs Knudsen, Peter Michael Reichstein Rasmussen, and Mikkel Thorup. "Fast hashing with strong concentration bounds". In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. Ed. by Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy. ACM, 2020, pp. 1265–1278.

[1] https://science.ku.dk/phd/filer/regelsaet/general_rules_and_guidelines_phd_programme/ Accessed 8. May 2023

[AKT21] Anders Aamand, Jakob Bæk Tejs Knudsen, and Mikkel Thorup. "Load balancing with dynamic set of balls and bins". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 1262–1275.

[AKKP+20] Thomas D. Ahle, Michael Kapralov, Jakob Bæk Tejs Knudsen, Rasmus Pagh, Ameya Velingker, David P. Woodruff, and Amir Zandieh. "Oblivious Sketching of High-Degree Polynomial Kernels". In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. Ed. by Shuchi Chawla. SIAM, 2020, pp. 141–160.

[AK20] Thomas D. Ahle and Jakob Bæk Tejs Knudsen. "Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search". In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. Ed. by Sandy Irani. IEEE, 2020, pp. 728–739.

[BBHK+23] Ioana O. Bercea, Lorenzo Beretta, Jakob Bæk Tejs Houen, Jonas Klausen, and Mikkel Thorup. *Locally Uniform Hashing*. 2023. In submission.

[BHP22] Ioana O. Bercea, Jakob Bæk Tejs Houen, and Rasmus Pagh. "Daisy Bloom Filters". In: *CoRR* abs/2205.14894 (2022). arXiv: 2205.14894. In submission.

[EHNR+22] Talya Eden, Jakob Bæk Tejs Houen, Shyam Narayanan, Will Rosenbaum, and Jakub Tetek. "Bias Reduction for Sum Estimation". In: *CoRR* abs/2208.01197 (2022). arXiv: 2208.01197. In submission.

[HPW23] Jakob Bæk Tejs Houen, Rasmus Pagh, and Stefan Walzer. "Simple Set Sketching". In: *2023 Symposium on Simplicity in Algorithms, SOSA 2023, Florence, Italy, January 23-25, 2023*. Ed. by Telikepalli Kavitha and Kurt Mehlhorn. SIAM, 2023, pp. 228–241.

[HT22] Jakob Bæk Tejs Houen and Mikkel Thorup. "Understanding the Moments of Tabulation Hashing via Chaoses". In: *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*. Ed. by Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff. Vol. 229. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 74:1–74:19.

[HT23] Jakob Bæk Tejs Houen and Mikkel Thorup. *A Sparse Johnson-Lindenstrauss Transform using Fast Hashing*. 2023. To appear in ICALP 2023.

In broad terms, the papers can be categorized into four topics. The first two topics, namely hash functions [ADKK+22; AKKR+20; BBHK+23; HT22; HT23] and hashing-based algorithms [AK20; AKKP+20; AKT21; BHP22; EHNR+22; HPW23], form the

main focus of this thesis. The remaining two topics covered in the papers are probability theory [AAHT22] and computational geometry [AAKR21].

Hash functions The focus of my research into hash functions has been on the design of fast and practical hash functions with provable theoretical guarantees, comparable to those of fully random hashing. In collaboration with my advisor Mikkel Thorup and other researchers, I have explored various families of tabulation-based hash functions in several papers [ADKK+22; AKKR+20; BBHK+23; HT22; HT23].

In [AKKR+20], we establish that simple tabulation hashing satisfies Chernoff-style concentration bounds under severe restrictions, and propose a new hashing scheme called tabulation-permutation that overcomes these restrictions and achieves Chernoff-style concentration bounds without sacrificing performance. This hashing scheme is easy to implement and has been demonstrated to be highly efficient in practice.

In [HT22], we take a different approach by bounding the moments of hashing-based sums of simple tabulation using techniques from functional analysis, providing a near-optimal understanding of the moments and leveraging these results to achieve strong concentration guarantees for mixed tabulation hashing. Building on this research, [HT23] shows that the strong concentration results of mixed tabulation can be applied to prove that a sparse Johnson-Lindenstrauss transform implemented using mixed tabulation hashing performs almost as well as when implemented using fully random hashing.

In a slightly different direction, [BBHK+23] introduces tornado tabulation hashing, which exhibits local randomness that provably enables diverse algorithms, such as HyperLogLog for counting distinct elements and one-permutation hashing for large-scale machine learning, to perform almost as if fully-random hashing was used. This paper also provides an efficient solution to obtaining fully-random hashing on a fixed set of n keys with $O(n)$ space complexity. Finally, in [ADKK+22], we demonstrate how access to hash functions with concentration guarantees similar to fully random hashing can significantly speed up various streaming algorithms.

Hashing-based algorithms I have further been involved in several projects on hashing-based algorithms [AK20; AKKP+20; AKT21; BHP22; EHNR+22; HPW23]. Here, the focus has not been on designing or proving guarantees for hash functions, but instead on designing algorithms that use hash functions as a subroutine.

In [AKT21], we consider the problem of dynamic load balancing, where we wish to distribute balls into bins in an environment where both balls and bins can be added and removed. Each bin has a capacity of C, i.e., each bin can contain at most C balls. We want to respect the capacities of the bins while minimizing the number of balls and bins that are affected when adding or removing a ball or a bin. We construct a data structure that, in expectation, moves $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin. Here, f is the fraction of non-full bins in the following simpler probabilistic problem: Place the balls into bins with capacity C, one ball at the time, where each ball picks a uniformly random non-full bin. We also solve this simpler problem and provide a tight bound for f.

In [AK20], we formulate and optimally solve a new generalized Set Similarity Search problem, which assumes the size of the database and query sets are known in advance. Our algorithm differs from the previous approaches by exploiting the information both present in the sets as well as their complements, and doing so asymmetrically between queries and stored sets. Turning the geometric concept, based on Boolean supermajority functions, into a practical algorithm requires ideas from branching random walks on $\mathbb{Z}^2$, for which we give the first non-asymptotic near tight analysis.

In [AKKP+20], we study the problem of sketching the polynomial kernel. The previous results on the problem depended exponentially on the degree of the polynomial kernel, while our algorithms only has a polynomial dependence on the degree.

In [HPW23], we consider a variant of the Invertible Bloom Filter of Eppstein and Goodrich. While Invertible Bloom Filters have an explicit checksum per bucket to determine whether the bucket stores a single key, we instead exploit the idea of quotienting, namely that some bits of the key are implicit in the location where it is stored, and we use these bits as an implicit checksum. The main technical challenge is that the implicit checksum is not enough to not ensure that no errors occur during decoding, so we have to show that the decoding algorithm can recover from those errors.

In [BHP22], we introduce a parameterization of the weighted Bloom filter called a Daisy Bloom filter. A weighted Bloom filter is a Bloom filter that adapt the number of hash functions according to the query element. We determine a near-optimal parameterization in the model where n element are inserted independently from a probability distribution, $\mathcal{P}$, and query elements are chosen from a probability distribution, $\mathcal{Q}$, under an upper bound on the false positive probability F.

In [EHNR+22], we investigate the well-studied problem in statistics of estimating the sum of a multiset of N real values by sampling from a distribution $\mathcal{P}$. Instead of sampling from $\mathcal{P}$, we assume that we can only sample from a distribution $\mathcal{Q}$ which is pointwise close to $\mathcal{P}$. We provide an algorithm to this problem that reduces the bias of sampling from the noisy distribution and show that it is essentially optimal.

Probability theory In [AAHT22], we introduce the notion of monotone random variables which are random integer variables X where the modulus of the characteristic function of X is decreasing on $[0, \pi]$. This class of random variables include many common distributions, e.g., the Bernoulli distribution, the Poisson distribution, and the geometric distribution. We provide estimates for the probability that the sum of independent monotone integer variables attains precisely a specific value without assuming that the variables are identically distributed. Our estimates show that the point probabilities are close to the density function of the normal distribution when the point is close to the mean.

Geometry Finally in a completely different direction, I have been involved in a project within geometry. In [AAKR21], we classify convex bodies by their contact, union, and intersection graphs. We show that two symmetric convex bodies, A, B, can construct the same contact, union, and intersection graphs if and only if there exist a linear transformation, T, such that $A = T(B)$.

Papers included in the thesis The thesis commences with a broad introduction, which encourages the exploration of practical hash functions and hashing-based algorithms. It then presents the papers included in the thesis, namely [HT22; HT23; AKT21; AK20; AAHT22]. The first two papers concentrate on hash functions, whereas the subsequent two papers deal with hashing-based algorithms. The final paper is a mathematics paper that discusses a basic probability theory problem.

Acknowledgements

First, I wish to thank my advisor, Mikkel Thorup, for his invaluable support. Throughout my PhD, he has consistently provided guidance and assistance, making himself readily available for both professional and personal discussions. I also wish to thank Mikkel for facilitating the friendly and collaborative environment at BARC which have been an absolute pleasure to have been a part of.

I would like to express my deep appreciation for the amazing individuals at BARC. Their presence has made going to the office a delight, and their support has been invaluable, particularly during the challenging times of the Covid pandemic.

I have thoroughly enjoyed engaging in extensive and stimulating discussions with them, delving into various interesting and complex problems. Some of these discussions were research-oriented, while others explored more important matters, such as the distinction between precision and accuracy. I am truly grateful for the social gatherings that have taken place at BARC, including the delightful Christmas and Easter lunches, the enjoyable pizza dinners, the festive gatherings at Warpigs, and the memorable retreats to Sweden, Hanstholm, and Romania. These occasions have not only strengthened our professional bonds but have also created cherished memories that I will forever treasure.

I want to extend a big thanks to my office mates throughout the years: Anders Aamand, Thomas Ahle, and Ioana Bercea. Each of you has contributed in unique ways, infusing our workspace with happiness and vitality, and today, I consider you all to be dear friends. Our camaraderie has not only enriched my personal life but has also played a significant role in my professional success. It is evident from the fact that 9 out of the 13 papers I have authored had one of you as a co-author. This collaborative synergy has been instrumental in our remarkable productivity and achievements within our field.

I also want to all my co-authors Anders Aamand, Mikkel Abrahamsen, Thomas Ahle, Noga Alon, Ioana Bercea, Lorenzo Beretta, Debarati Das, Talya Eden, Mathias Bæk Tejs Langhede, Michael Kapralov, Evangelos Kipouridis, Jonas Klausen, Shyam Narayanan, Rasmus Pagh, Peter Michael Reichstein Rasmussen, Will Rosenbaum, Mikkel Thorup, Ameya Velingker, Stefan Walzer, David Woodruff, and Amir Zandieh. It has been a pleasure to work together with you and I am very proud of our results.

Finally, I want to thank my family, whose support has been instrumental in my journey. I am especially indebted to my beloved wife, Maiken, whose unwavering love and encouragement have played a pivotal role in my achievements. Your presence has been a constant source of inspiration, and I am eternally thankful for your unwavering support.

I also want to extend my appreciation to our newborn son, Valdemar. Although his arrival coincided with the stressful process of thesis-writing, his presence has brought joy and a much-needed distraction. I dedicate any typographical errors in this thesis to him, for he reminds me of the beauty and unpredictability of life.

Contents

Part I

Synopsis

Chapter 1

Introduction

In the ever-evolving landscape of machine learning and artificial intelligence, the rapid advancements and widespread adoption of these technologies have generated a large demand for algorithms that can efficiently process massive amounts of data. The availability of vast datasets, coupled with the complexity of modern computational problems, has made the exact computation of solutions an intractable task. The magnitude of the data at hand often surpass the capabilities of traditional exact computation methods.

Moreover, the challenges are further complicated by the presence of conditional lower bounds, which indicate that achieving exact results within reasonable time frames is simply unfeasible for certain problems. As such, the focus has shifted towards deriving approximate solutions, which can be obtained in a more feasible and efficient manner. However, rather than resorting to simplistic heuristics that provide pragmatic yet unreliable results, the aspiration is to devise algorithms and data structures that can deliver dependable statistical analyses, even in large-scale data scenarios. This pursuit is fueled by the desire to strike a balance between computational efficiency and the accuracy of results.

Numerous problems and influential algorithms have been explored within this area of research. Randomization is a widely used and valuable approach in algorithm design for addressing such problems, with hash functions playing a particularly crucial role. When analyzing the performance of these algorithms, it is often convenient to assume that the hash functions are fully random, meaning that the hash values of keys are mutually independent and uniformly distributed. This assumption simplifies the analysis considerably. However, in practice, achieving fully random hashing is unfeasible. Conversely, if a weak hash function is used, we lose the strong theoretical guarantees, and the algorithms may fail completely when dealing with certain structured data sets. Therefore, the goal is to discover practical and implementable hash functions that can offer some of the same desirable theoretical guarantees as fully random hashing.

This thesis will adopt a dual focus. The first part centers around analyzing and establishing theoretical guarantees for practical families of tabulation-based hash functions, specifically simple tabulation hashing and mixed tabulation hashing. The latter part, on the other hand, concentrates on devising hashing-based algorithms under the assumption of fully random hashing.

Structure of this thesis The thesis consists of eight chapters including this introductory chapter. It first consists of two chapters with a general introduction to hash functions and moments of hash-based sums. Afterwards, it contains five chapters that introduces each of the papers included in the thesis.

Chapter 4 This chapter is dedicated to presenting the results of the research paper "Understanding the Moments of Tabulation Hashing via Chaoses" [HT22] of Appendix A. This paper is published in the proceedings of the 49th EATCS International Colloquium on Automata, Languages and Programming (ICALP) 2022.

Chapter 5 This chapter is dedicated to presenting the results of the research paper "A Sparse Johnson-Lindenstrauss Transform using Fast Hashing" [HT23] of Appendix B. This paper is to be published in the proceedings of the 50th EATCS International Colloquium on Automata, Languages and Programming (ICALP) 2023.

Chapter 6 This chapter is dedicated to presenting the results of the research paper "Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search" [AK20] of Appendix C. This paper is published in the proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS) 2020.

Chapter 7 This chapter is dedicated to presenting the results of the research paper "Load Balancing with Dynamic Set of Balls and Bins" [AKT21] of Appendix D. This paper is published in the proceedings of the 53rd Annual ACM Symposium on Theory of Computing (STOC) 2021.

Chapter 8 This chapter is dedicated to presenting the results of the research paper "On Sums of Monotone Random Integer Variables" [AAHT22] of Appendix E. This paper is published in the journal Electronic Communications in Probability.

Chapter 2

Hash Functions

In this chapter, we will provide a brief introduction to the study hash functions and hashing-based algorithms. It is in parts based on the introduction from [HT22].

The concept of hash functions dates all the way back to the 1950s [Dum56] and has since become an ubiquitous tool in the design of randomized algorithms. A *hash function* is a random function $h\colon U \to R$ from a large universe of keys U to a range R chosen with respect to some probability distribution $\mathcal{D}$. Usually, U and R are bounded integer ranges, $U = [u] = \{0, \dots, u-1\}$, and $R = [m] = \{0, \dots, m-1\}$. Most often, and this will be the case in this thesis, $\mathcal{D}$ will be the uniform distribution restricted to a subset of function of $\mathcal{H} \subseteq R^U$. We will then refer to $\mathcal{H}$ as a *family* of hash functions. In this thesis, we will not explicitly state $\mathcal{H}$ but it will instead be described implicitly. It is common to refer to the keys of U as *balls* and the elements of R as *bins*. One can then think of h as throwing the balls of U into the bins of R according to the distribution $\mathcal{D}$. A particularly interesting case is when $\mathcal{D}$ is the uniform distribution on R^U, i.e., $\mathcal{H} = R^U$. In that case, h is said to be a *fully random* or *uniformly random* hash function. A fully random hash function h thus assigns each key $x \in U$ a uniform random random hash value $h(x) \in R$ and the hash values of the keys $(h(x))_{x \in U}$ are mutually independent.

Fully random hash functions are incredibly powerful and the mutual independence of the hash values often allows for simple probabilistic arguments showing strong theoretical guarantees. For this reason a lot progress of analysis of hashing-based algorithms start by assuming access to fully random hash functions before trying to weaken that assumption. Unfortunately, the assumption of access to fully random hash function is wholly unrealistic in practice. Since the size of the hash family $\mathcal{H}$ is $|R|^{|U|}$ then we would need to use $|U| \log_2 |R|$ to represent h. In most application, U will be for too large and often the idea of hashing is to map down to a smaller domain where we can represent the elements. Instead, we want a *practical* hash function that requires less space and be evaluated quickly while still be random enough that it has theoretical guarantees similar to those of fully random hash functions.

Let us consider a simple of example of an algorithm employing a hash function. Perhaps, the simplest example of this is *hashing with chaining* which was also the original motivation for studying hash function [Dum56]. In hashing with chaining, we distribute

a subset of keys $S \subseteq U$ from a large universe U into a table of size $m = |R|$ with a hash function $h\colon U \to R$, storing a key $x \in S$ at position $h(x) \in R$. We will handle collisions by making a linked list of keys hashing to the same entry. We can then determine if a key $y \in U$ is a member of S by hashing y and checking if y is present in the present in the linked list stored in position $h(y)$. Clearly, we then get that the query time is proportional with the number of keys colliding with y. Now consider the simple case where $|S| = |R| = n$, i.e., we distribute n balls into n bins. Let us furthermore assume that h is fully random. The expected number of collisions with y is then[1]

$$\sum_{x \in S \setminus \{y\}} \Pr[h(x) = h(y)] = 1 - \frac{[y \in S]}{n}$$

This holds true as long as the hash values are pairwise independent which is obviously true for fully random hash functions. However, often we are not just interested in the expected query time but also want to bound the query time with high probability. The probability that a bin receives more than k balls are at most

$$n\binom{n}{k}n^{-k} \leq n\left(\frac{e}{k}\right)^k$$

by employing a simple union bound. If we chose $k = O(\log n / \log\log n)$ large enough then standard calculations give us that no bin receives more than k keys with probability $1 - n^{-\gamma}$, where γ depends on how large we choose k. For this analysis to work, we need that any k balls are hashing mutually independent, again this clearly the case for fully random hash function.

When evaluating the performance of a hash function, three main parameters are typically considered. Firstly, the time required to evaluate the hash function, secondly, the space required to represent it, and finally, theoretical guarantees. Theoretical guarantees involve distilling the necessary properties of a fully random hash function to determine whether a practical hash function with comparable theoretical guarantees exists. This is typically accomplished by aiming to replicate the properties of fully random hashing.

In the context of this thesis, examples of theoretical guarantees include concentration results on hash-based sums and the number of (weighted) collisions. An additional example can be found in [HT23], where an analysis of a Sparse Johnson-Lindenstrauss Transform [KN14] is provided, which identifies a set of necessary properties from fully random hashing and demonstrates that a practical hash function satisfies these properties.

k-independence Wegman and Carter [WC81] introduced the key concept of k-independent hash functions. A hash function, $h\colon U \to R$, drawn from a distribution, $\mathcal{D}$, is said to be k-independent if $(h(x_0), \ldots, h(x_{k-1}))$ is uniformly distributed in R^k for any k distinct keys $x_0, \ldots, x_{k-1} \in U$. An important observation is that this implies that for any k distinct keys $x_0, \ldots, x_{k-1} \in U$ their hash values $h(x_0), \ldots, h(x_{k-1})$ are mutually independent. It is commonly the case when analysing hashing-based algorithms, that the

[1] For a statement P we let $[P]$ be 1 if P is true and 0 otherwise.

crucial property of the hash function is some form of limited independence, hence we can substitute the fully random hash function with a k-independent hash function for a large enough k. For example, consider the above analysis of hashing with chaining, here we used 2-independence to calculate the expected query time and used $O(\log n / \log \log n)$-independence to give a high probability bound on query time. Thus, we get the same result if we just assume that the hash function is k-independent for $k = O(\log n / \log \log n)$ large enough.

We are often interested in how concentrated a hash-based sum, X, is around its mean, μ. If the hash function k-independent for an even k then we can employ the k'th moment bound to get that

$$\Pr[|X - \mu| \geq t] \leq \frac{\mathrm{E}\left[|X - \mu|^k\right]}{t^k} .$$

Now since the hash function is k-independent then the k'th central moment, $\mathrm{E}\left[|X - \mu|^k\right]$, is exactly the same as if the hash function were a fully random hash function. The study of moments are incredibly powerful and will be discussed more in depth in Chapter 3.

Wegman and Carter [WC81] gave a simple way of constructing a k-independent hash function. Let p be a prime then $h \colon [p] \to [p]$ is constructed by choosing $a_0, \ldots, a_{k-1} \in [p]$ independently and uniformly at random and defining

$$h(x) = \sum_{i=0}^{k-1} a_i x^i \pmod{p} . \tag{2.1}$$

This gives a k-independent hash function that maps to $[p]$, to obtain a hash function that maps to $R = [m]$ we define $h'(x) = h(x) \mod m$. This construction only approximately satisfy the definition of k-independence but if we choose $p \gg m$ large enough then it becomes a non-issue for most applications.

The main drawback of the construction in (2.1) is that it takes $\Omega(k)$ time to evaluate. This can be quite problematic since a lot application will have $k = \Omega(\log n)$ and then the evaluation of the hash function adds a significant overhead. A natural question is then whether it is possible to construct a k-independent hash function that can be evaluated in $o(k)$ time. This was studied by Siegel [Sie04] who showed this is indeed possible but then you need to much more space. More precisely, Siegel showed that if you want to construct a k-independent hash function that can be evaluated using $t < k$ memory probes then you need to use $\Omega(k(u/k)^{1/t})$ words of space. The lower bound shows that there is an inherent connection between the independence of the hash function, the evaluation of the hash function, and the space usage of the hash function. If we want to construct a hash function that is k-independent and can be evaluated in time $c = O(1)$ then we need to use $u^{1/c}$ words of space. In the same paper, Siegel also constructs a hash function that uses $O(u^{1/c})$ words of space, can be evaluated in $O(c)^c$ time, and which is $u^{\Omega(1/c^2)}$ independent. Unfortunately, as Siegel notes in his paper, the construction is "*[...] are far too slow for any practical application*".

2.1 Tabulation-Based Hashing

The construction of Siegel is part of the class of hash function which we call *tabulation-based* hash functions. Tabulation-based hash functions are hash functions that are centered around a table with random entries which are used when evaluating the hash functions. More precisely, tabulation-based hash functions will have access to a random table, $T\colon \Gamma \to R$, which we will think of as a function, which takes values from some set Γ and returns values in R. Usually, the size of Γ is a root of the size of the universe or proportional to the number of elements, that is, either $|\Gamma| = O(u^\varepsilon)$ for some small constant $\varepsilon < 1$ or $|\Gamma| = O(n)$ where n is the number of elements. Let us briefly compare the tabulation-based hash functions to the polynomial hashing scheme, (2.1). Tabulation-based hash functions uses significantly more space, $O(u^\varepsilon)$ compared to $O(k)$, but in contrast with the polynomial hashing scheme, they do not need to read all the randomness when evaluating the functions. In fact, the constructions we consider in this thesis only need to access a constant number of words when evaluating the functions.

It has been an active research area to improve on Siegel's construction. Several papers have focused on constructing better k-independent hash functions with the best construction currently having an evaluation time of $O(c \log c)$ while using the same space and having the same independence as Siegel's construction [Tho13; CPT15]. While they are asymptotically better than Siegel's construction they have so far not been proven to work in practice. In a slightly different direction, a series of papers have studied constructing hash functions that are highly independent on a fixed (unknown) set $S \subseteq U$ but not on the entire universe [DW03; ÖP03; PP08; DR09]. Currently, the best construction for constructing a function that is fully independent on a fixed (unknown) set $S \subseteq U$ is by Dietzfelbinger and Rink [DR09] and it uses $(1+\varepsilon)\,|S|$ words of space and can be evaluated in $O(\log(1/\varepsilon))$ time. Dahlgaard et al. [DKRT15] considered the even more general problem where the set S is chosen by a random process that uses parts of the hash values. They gave a construction that uses $O(|S|)$ space and can be evaluated in constant time.

The construction discussed above are all concerned with getting a highly independent hash function. In this thesis, we present several new results on tabulation-based hashing but in contrast with the above construction they do not exploit independence.

Simple tabulation hashing *Simple tabulation* hashing dates back to 1970 and was first introduced by Zobrist for optimizing chess computers [Zob70]. In simple tabulation hashing, we view the universe, U, to be of the form $U = \Sigma^c$ for some alphabet, Σ, and a positive integer c. Let $T\colon [c] \times \Sigma \to [2^l]$ be a uniformly random table, i.e., each value is chosen independently and uniformly at random from the set $[2^l]$. A simple tabulation hash function, $h\colon \Sigma^c \to [2^l]$, is then defined by

$$h(\alpha_0, \ldots, \alpha_{c-1}) = \bigoplus_{i=0}^{c-1} T(i, \alpha_i)\ ,$$

where $\oplus$ is the bitwise XOR-operation, i.e., addition when $[2^l]$ is identified with the Abelian group $(\mathbb{Z}/2\mathbb{Z})^l$. We say that h is a simple tabulation hash function with c characters. With

8- or 16-bit characters, the random table T fits in cache, and then simple tabulation is very fast, e.g., in experiments, [PT12] found it to be as fast as two to three multiplications.

Let us reflect a bit about how simple tabulation hashing fits in the framework of tabulation-hashing introduced above. In simple tabulation hashing, we have $\Gamma = [c] \times \Sigma$, the space usage is $O(c|\Sigma|) = O(cu^{1/c})$, and in each evaluation of a simple tabulation hash function we access precisely c memory locations. We see that there is an interesting trade-off between evaluation time with c lookups and the space usage of $O(cU^{1/c})$. This motivates analysing how c affects the theoretical guarantees of simple tabulation hashing.

The constructions discussed above were all concerned with getting high independence either on a subset of the universe or on the entire universe. An obvious question is therefore how much independence simple tabulation hashing possess. As it turns out, simple tabulation hashing is only 3-independent. The simplest case showing that simple tabulation hashing is not 4-independent is as follows: Let $c = 2$ and choose $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Sigma$ with $\alpha_1 \neq \alpha_2$ and $\beta_1 \neq \beta_2$. We now consider the keys $x = (\alpha_1, \beta_1)$, $y = (\alpha_1, \beta_2)$, $z = (\alpha_2, \beta_1)$, and $w = (\alpha_2, \beta_2)$. It now follows from the definition of simple tabulation hashing that $h(x) \oplus h(y) \oplus h(z) \oplus h(w) = 0$, irrespective of the randomness of h, thus the keys x, y, w, z are not mutually independent.[2] The low independence of simple tabulation hashing might suggest that it does not have strong theoretical guarantees but surprisingly it does. This was first studied by Pătraşcu and Thorup [PT12] who showed that simple tabulation hashing in certain situations has concentration guarantees like fully and studied its performance in linear probing, cuckoo hashing, and min-wise hashing. The concentration guarantees of simple tabulation hashing was later refined [AKKR+20] and it was shown that the concentration guarantees tightly linked to size of the output range $|R|$. A series of papers [DKRT16; AKT18; AT19] have studied simple tabulation in the context of 2- and d-choice balanced allocation schemes and the number of non-empty bins.

One of the reasons, that simple tabulation has these strong guarantees even though that it is only 3-independent, is that the hash values of *most* subsets are independent. There exists a quite simple description of which subsets are independent under simple tabulation hashing. First, we note that we can consider a key $x = (x_0, \ldots, x_{c-1}) \in \Sigma^c$ as a set of c *position characters*, $\bar{x} = \{(0, x_0), \ldots, (c-1, x_{c-1})\} \subseteq [c] \times \Sigma$. Now a subset of keys $A \subseteq \Sigma^c$ have independent hash values, that is, $(h(x))_{x \in A}$ are mutually independent, if and only if there does not exist a non-empty subset $\emptyset \neq B = \{x_0, \ldots, x_{t-1}\} \subseteq A$ such that $\overline{x_0} \triangle \cdots \triangle \overline{x_{t-1}} = \emptyset$ where $\triangle$ denotes the symmetric difference operator. A counting argument in [DKRT15] shows that there cannot be that many subsets, B, with that property.

Mixed tabulation hashing Let us consider the following way to construct a hash function: Let $g \colon \Sigma^r \to R$ be a simple tabulation hash function and $f \colon U \to \Sigma^r$ be a function. We then define $h \colon U \to R$ by $h(x) = g(f(x))$. Now if we can construct f such that for all subsets $B = \{x_0, \ldots, x_{t-1}\} \subseteq U$ up to some size k, we have that $\overline{f(x_0)} \triangle \cdots \triangle \overline{f(x_{t-1})} \neq \emptyset$, then h will be k-independent. This is roughly speaking the

[2] We can easily extend the example to $c > 2$, simply choose $\gamma \in \Sigma^{c-2}$ and consider the keys $x = (\alpha_1, \beta_1, \gamma)$, $y = (\alpha_1, \beta_2, \gamma)$, $z = (\alpha_2, \beta_1, \gamma)$, and $w = (\alpha_2, \beta_2, \gamma)$.

idea behind most of the construction of highly independent tabulation-based hash functions. Unfortunately, it is not known how to construct f deterministically. Instead, the constructions choose f to be a random hash function and show that with high probability it has the desired property.

One such construction is *mixed tabulation hashing* which was introduced in [DKRT15]. Mixed tabulation hashing can be constructed as follows: Let $g\colon \Sigma^{c+d} \to [2^l]$ and $h_2\colon \Sigma^c \to \Sigma^d$ be simple tabulation hash function and define $f\colon \Sigma^c \to \Sigma^{c+d}$ by $f(x) = (x, h_2(x))$. A mixed tabulation hash function, $h\colon \Sigma^c \to [2^l]$, is then defined by

$$h(x) = g(f(x)) \ .$$

We call h a mixed tabulation hash function with c characters and d derived characters. It can be beneficial to view g as two simple tabulation hash functions, $h_1\colon \Sigma^c \to [2^l]$ and $h_3\colon \Sigma^d \to [2^l]$, then $g(x, y) = h_1(x) \oplus h_3(y)$ for $x \in \Sigma^c, y \in \Sigma^d$. With this view we then get that $h(x) = h_1(x) \oplus h_3(h_2(x))$. When implementing a mixed tabulation hash function it is possible to combine h_1 and h_2 into a single simple tabulation hash function $\Sigma^c \to [2^l] \times \Sigma^d$, and then h is implemented with only $c + d$ lookups. Mixed tabulation hashing was originally introduced to obtain high independence on a set S but in this thesis we will study properties of mixed tabulation hashing unrelated to its independence.

Chapter 3

Moments

In this chapter, we will introduce moments of hash-based sums. It is in parts based on the introduction from [HT22].

Before we go into discussing moments of hash-based sums, it will be instructive to first consider some classical concentration results for random variables. The Chernoff's bounds [Che52] go all the way back to the 1950s and were originally introduced in the study of statistics. In fact, the study of such bounds can be traced event further back to Bernstein [Ber24] in the 1920s. Today Chernoff's bounds are an essential part of analysing randomized algorithms.

Consider the random variable $X = \sum_{i\in[n]} X_i$ where $(X_i)_{i\in[n]}$ are independent Bernoulli random variables, i.e., $X_i \in \{0,1\}$ and $\Pr[X_i = 1] = 1 - \Pr[X_i = 0] = \mathrm{E}[X_i]$ for all $i \in [n]$. Writing $\mu = \mathrm{E}[X]$ Chernoff's bounds show that for every $\varepsilon > 0$ it holds that

$$\Pr[X \geq (1+\varepsilon)\mu] \leq \left(\frac{e^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}}\right)^{\mu} = \exp(-\mu\mathcal{C}(\varepsilon)) \ ,$$
$$\Pr[X \leq (1-\varepsilon)\mu] \leq \left(\frac{e^{-\varepsilon}}{(1-\varepsilon)^{1-\varepsilon}}\right)^{\mu} = \exp(-\mu\mathcal{C}(-\varepsilon)) \ .$$

Here $\mathcal{C}(x) : (-1,\infty) \to \mathbb{R}_+$ is defined by $\mathcal{C}(x) = (1+x)\ln(1+x) - x$.

The proof of Chernoff's bounds follow by Markov's inequality and by upper bounding the moment generating function. Here we will focus on the upper tail but the proof of lower tail is analogous.

$$\Pr[X \geq (1+\varepsilon)\mu] \leq \inf_{\lambda>0} \left(\mathrm{E}\left[e^{\lambda X}\right] \exp(-(1+\varepsilon)\mu)\right) \ .$$

The last step of the proof is to upper bound the moment generating function, $\mathrm{E}\left[e^{\lambda X}\right] \leq \exp\big(\mu(e^{\lambda}-1)\big)$. An interesting observation is that $\exp\big(\mu(e^{\lambda}-1)\big) = \mathrm{E}\left[e^{\lambda Y}\right]$ where Y is random variable that have a Poisson distribution with parameter μ. So we upper bounded the moment generating function of X by the moment generating function of the Poisson distribution with parameter μ.

An alternative method for proving concentration result is by using moment bounds. Let $p \geq 1$ and consider a random variable X for which $\mathrm{E}[X] < \infty$ exists then for all $t > 0$

it holds that

$$\Pr[X > t] \leq \mathrm{E}\big[(X^+)^p\big]\, t^{-p}\,.$$

Here $X^+ = \max\{X, 0\}$. There are several advantages to using moment bounds instead of using the moment generating function. It has been shown that moment bounds are always tighter [PN95], but even more crucial is the fact that the moment generating function might not exist even random variables where all the moments exist. This motivates studying the moments of random variables.

It often more convenient to work with p-norms instead of working the moments of random variables directly. The p-norm of a random variable is the p'th root of the p'th moment of the random variable and is formally defined as follows:

Definition 3.1 (p-norm)**.** Let $p \geq 1$ and X be a random variable with $\mathrm{E}[|X|^p] < \infty$. We then define the p-norm of X by $\|X\|_p = \mathrm{E}[|X|^p]^{1/p}$.

A nice and simple feature of working with p-norms is the following tail bound which follows by a standard application of Markov's inequality.

$$\Pr\Big[|X| \geq e\,\|X\|_p\Big] \leq e^{-p}\,.$$

This is useful a bound since a lot of applications are interested in bounding the smallest deviation that happens with probability at most δ. Thus, if we bound $\|X\|_p$ for $p = \log(1/\delta)$ then we immediately obtain such a bound.

As discussed above, Chernoff's bounds are proven by upper bounding the moment generating function by the moment generating function of the Poisson distribution. This inspires us to similarly bound the p-norms with the p-norms of the Poisson distribution. In order to do this, we first need to understand the p-norms of the Poisson distribution. In [HT23], they introduced the function $\Psi_p(M, \sigma^2)$ which does exactly that. THe definition of $\Psi_p(M, \sigma^2)$ is quite technical but [HT22] proved that $\Psi_p(1, \lambda)$ is equal up to a constant factor to the central p-norm of a Poisson distributed variable with mean λ.

Definition 3.2. For $p \geq 2$ we define the function $\Psi_p \colon \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ as follows

$$\Psi_p(M, \sigma^2) = \begin{cases} \left(\frac{\sigma^2}{pM^2}\right)^{1/p} M & \text{if } p < \log \frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{p}\sigma & \text{if } p < e^2 \frac{\sigma^2}{M^2} \\ \frac{p}{e \log \frac{pM^2}{\sigma^2}} M & \text{if } \max\left\{\log \frac{pM^2}{\sigma^2}, e^2 \frac{\sigma^2}{M^2}\right\} \leq p \end{cases}.$$

Remark 3.3. When p is *small* then case 1 and 2 apply while for *large* p case 3 applies. If $2 < e^2 \frac{\sigma^2}{M^2}$ then we always have that $p > \log \frac{pM^2}{\sigma^2}$ for $2 \leq p$, hence only case 2 and 3 apply. Similarly, if $e^2 \frac{\sigma^2}{M^2} \leq 2$ then $p \geq e^2 \frac{\sigma^2}{M^2}$ for all $2 \leq p$, hence only case 1 and 3 apply. This shows that the cases disjoint and cover all parameter configurations.

The definition $\Psi_p(M, \sigma^2)$ is quite technical but the important property of it is not its exact definition but the fact that it captures the central p-norms of Poisson distributed random variables. This was proven in [HT22] and is stated formally in the following lemma.

Lemma 3.4 ([HT22]). *There exist universal constants K_1 and K_2 satisfying that for a Poisson distributed random variable, X, with $\lambda = \mathrm{E}[X]$*

$$K_2\Psi_p(1,\lambda) \leq \|X - \lambda\|_p \leq K_1\Psi_p(1,\lambda) \ ,$$

for all $p \geq 2$.

3.1 Moments of Hash-Based Sums

Let us formalize the notion of a hashed-based sum. For a hash function $h\colon U \to R$ and a fixed *value function*, $v\colon U \times R \to \mathbb{R}$, we define the random variable $X_x = v(x, h(x))$ for every key $x \in U$. We are then interested in proving concentration bounds for the sum $X = \sum_{x\in U} X_x = \sum_{x\in U} v(x, h(x))$. It should be noted that the randomness of X is derived from the hash function h, thus the results will depend on the strength of h.

This is quite a general problem, and at first glance, it might not be obvious why this is a natural construction to consider, but it does generalize a variety of well-studied constructions:

1. Let $S \subseteq U$ be a set of balls and assign a weight, $w_x \in \mathbb{R}$, for every ball, $x \in S$. The goal is to distribute the balls, S, into a set of bins $R = [m]$. For a bin, $y \in [m]$, we define the value function $v_y\colon U \times [m] \to \mathbb{R}$ by $v_y(x, j) = w_x\,[j = y]\,[x \in S]$, then $X = \sum_{x\in U} v_y(x, h(x)) = \sum_{x\in S} w_x\,[h(x) = y]$ will be the weight of the balls hashing to bin y.

2. Instead of concentrating on a single bin, we might be interested in the total weight of the balls hashing below some threshold l. This is useful for sampling, because if $h(x)$ is uniform in $[m]$, then $\Pr[h(x) < l] = l/m$. We then define the value function $v\colon U\times[m] \to \mathbb{R}$ by $v(x, j) = w_x\,[j < l]\,[x \in S]$, then $X = \sum_{x\in U} v(x, h(x)) = \sum_{x\in S} w_x\,[h(x) < l]$ will be precisely the total weight of the balls hashing below l.

We already saw the first case appear when we discussed hashing with chaining in the chapter and it generally appears when one tries to allocate resources. The second case arises in streaming algorithms. In [ADKK+22], it was shown that if the hash function provides strong concentration guarantees then the running time of certain streaming algorithms can be improved. Finally, in [HT23], the full generality of hashed-based sums are exploited to obtain a new analysis of a Sparse Johnson-Lindenstrauss transform. The goal is generally to prove that X is concentrated around the mean $\mu = \mathrm{E}[X]$. If h is a uniformly random hash function then this will be the case under mild assumptions about v but it cannot otherwise be assumed a priori to be the case.

Now we fix a value function $v\colon U \times R \to \mathbb{R}$ where $\sum_{j\in R} v(x, j) = 0$ for all keys $x \in U$, and we assume that the hash function, $h\colon U \to R$, is uniformly distributed for all keys

$x \in U$, i.e., the has value $h(x)$ is uniformly distributed in R. These assumptions imply that the random variable $X_x = v(x, h(x))$ has mean 0 for every key $x \in U$. We then define some notation which will be useful.

$$M_v = \max_{x \in U, j \in [m]} |v(x,j)| \ , \tag{3.1}$$

$$\sigma_v^2 = \frac{\sum_{x \in U, j \in [m]} v(x,j)^2}{m} \ . \tag{3.2}$$

Here M_v is the smallest upper bound that always holds $|X_x|$ for all keys $x \in U$, and $\sigma_2 = \mathrm{Var}[X]$ if the hash function h is pairwise independent. With this notation, we can obtain a stronger tail bound than Chernoff's bound for general value functions by employing Bennett's inequality [Ben62]. For a fully random hash function, h, Bennett's inequality give us that.

$$\begin{aligned} \Pr\left[\left|\sum_{x \in U} v(x, h(x))\right| \geq t\right] &\leq 2\exp\left(-\tfrac{\sigma_v^2}{M_v^2}\mathcal{C}\left(\tfrac{tM_v}{\sigma_v^2}\right)\right) \\ &\leq \begin{cases} 2\exp\left(-\frac{t^2}{3\sigma_v^2}\right) & \text{if } t \leq \frac{\sigma_v^2}{M_v} \\ 2\exp\left(-\frac{t}{2M_v}\log\left(1+\frac{tM_v}{\sigma_v^2}\right)\right) & \text{if } t > \frac{\sigma_v^2}{M_v} \end{cases} , \end{aligned} \tag{3.3}$$

The proof Bennett's inequality follows the same structure as the proof of Chernoff's bounds. The crucial difference between them is how they upper bound the moment generating function. In the proof of Bennett's inequality, the estimate $\mathrm{E}\left[e^{\lambda X}\right] \leq \exp\left(\frac{\sigma_v^2}{M_v^2}(e^{\lambda M_v} - 1)\right)$ is used. If Y is a Poisson distributed random variable with parameter $\frac{\sigma^2}{M^2}$ then $\exp\left(\frac{\sigma_v^2}{M_v^2}(e^{\lambda M_v} - 1)\right) \leq \mathrm{E}\left[e^{\lambda(M_v Y)}\right]$. Thus, the moment generating function of X is upper bounded by the moment generating function of $M_v Y$. Combining this insight with Lemma 3.4 which shows that $\Psi_p(M, \sigma^2)$ captures the central p-norms of the Poisson distribution, it should not be too surprising that the p-norms of $\sum_{x \in U} v(x, h(x))$ can be controlled using $\Psi_p(M, \sigma^2)$. This is exactly what was shown in [HT22].

Theorem 3.5 ([HT22])**.** *Let $h \colon U \to [m]$ be a uniformly random function, let $v \colon U \times [m] \to \mathbb{R}$ be a fixed value function, and assume that $\sum_{j \in [m]} v(x,j) = 0$ for all keys $x \in U$. Define the random variable $X_v = \sum_{x \in U} v(x, h(x))$. Then for all $p \geq 2$*

$$\|X_v\|_p \leq L\Psi_p\left(M_v, \sigma_v^2\right) \ ,$$

where $L \leq 16e$ is a universal constant.

To get a further intuition for $\Psi_p(M, \sigma^2)$, it is instructive to apply Markov's inequality and compare the tail bound to Bennett's inequality. More precisely, assume that $\|Z - \mathrm{E}[Z]\|_p \leq L\Psi_p(M, \sigma^2)$ for a constant L and for all $p \geq 2$. Then we can use Markov's

inequality to get the following tail bound for all $t > 0$.

$$\begin{aligned}\Pr\Big[\Big|Z - \mathrm{E}[Z]\Big| \geq t\Big] &\leq \left(\frac{\|Y - \mathrm{E}[Y]\|_p}{t}\right)^p \\ &\leq \begin{cases} \frac{L^2\sigma^2}{2t^2} & \text{if } t \leq L\max\Big\{M, \frac{e\sigma}{\sqrt{2}}\Big\} \\ \exp\Big(-\frac{4t^2}{e^2L^2\sigma^2}\Big) & \text{if } L\frac{e\sigma}{\sqrt{2}} \leq t \leq L\frac{e^2\sigma^2}{2M} \\ \exp\big(-\frac{t}{LM}\log\big(\frac{2tM}{L\sigma^2}\big)\big) & \text{if } L\max\Big\{\frac{e^2\sigma^2}{2M}, M\Big\} \leq t \end{cases}.\end{aligned} \tag{3.4}$$

In order to obtain these bounds, p is chosen as follows: If $t \leq \max\Big\{M, \frac{e\sigma}{\sqrt{2}}\Big\}$ then $p = 2$ and otherwise p is chosen such that $\|Z - \mathrm{E}[Z]\|_p \leq e^{-1}t$. More precisely, we have that

$$p = \begin{cases} 2 & \text{if } t \leq L\max\Big\{M, \frac{e\sigma}{\sqrt{2}}\Big\} \\ \frac{4t^2}{e^2L^2\sigma^2} & \text{if } L\frac{e\sigma}{\sqrt{2}} \leq t \leq L\frac{e^2\sigma^2}{2M} \\ \frac{t}{LM}\log\left(\frac{2tM}{L\sigma^2}\right) & \text{if } L\max\Big\{\frac{e^2\sigma^2}{2M}, M\Big\} \leq t \end{cases}.$$

We see that eq. (3.4) gives the same tail bound as Bennett's inequality, eq. (3.3), up to a constant in the exponent.

An interesting question is whether we can do better, that is, is the $\Psi_p(M, \sigma^2)$ the best we can do. In [HT22], we showed that if all we know about our value function v is M_v and σ_v^2 then the answer is no. More precisely, in [HT22], we show the following lower bound.

Theorem 3.6 ([HT22])**.** *Let $h\colon U \to [m]$ be a uniformly random function, then there exists a value function, $v\colon U \times [m] \to \mathbb{R}$, where $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in U$, such that the random variable $X_v = \sum_{x\in U} v(x, h(x))$ satisfies that for all $p \leq L_1\,|U|\log(m)$*

$$\left\|\sum_{x\in U} v(x,h(x))\right\|_p \geq L_2\Psi_p\left(M_v, \sigma_v^2\right),$$

where L_1 and L_2 are a universal constant.

Chapter 4

Understanding the Moments of Tabulation Hashing via Chaoses

This chapter is dedicated to presenting the results of our research paper "Understanding the Moments of Tabulation Hashing via Chaoses" [HT22] of Appendix A, and includes a slightly modified subset of its introduction. The work presented in the paper overlaps with the master thesis submitted by the author in May 2021 [Hou23]. The core results were present in the master thesis [Hou23] but several proofs have been rewritten and significantly simplified.

4.1 Introduction

In [HT22], we will focus on analyzing hash-based sums. More precisely, we consider a fixed *value function*, $v\colon U \times R \to \mathbb{R}$, and define the random variable $X_x = v(x, h(x))$ for every key $x \in U$. We are then interested in proving concentration bounds for the sum $X = \sum_{x\in U} X_x = \sum_{x\in U} v(x, h(x))$. It should be noted that the randomness of X derives from the hash function h, thus the results will depend on the strength of h. We will focus on the case where h is either a simple tabulation hash function or a mixed tabulation hash function.

Pătraşcu and Thorup [PT12] studied the theoretical guarantees of simple tabulation hashing. They showed that simple tabulation hashing provides Chernoff-style tail bounds for distributing n balls into m bins as long as $m = n^{1-1/(2c)}$. This is fine for some applications but a lot applications need the number of bins m to be much smaller. In a later paper [PT13], the same authors introduced a variant of simple tabulation hashing called twisted tabulation hashing. They showed that twisted tabulation hashing provides Chernoff-style tail bounds for any hash-based sum (as defined in Section 3.1) as long as the expectation μ is not too large. More precisely, they need $\mu \leq |\Sigma|^{1-\varepsilon}$ for a constant $0 < \varepsilon$. Again, there are a lot of applications where this assumption will be violated.

In a recent paper [AKKR+20], the analysis of simple tabulation hashing was strengthened. They showed that simple tabulation hashing provides Chernoff-style tail bounds for distributing n balls into m without any restriction on n and m, but instead they

needed an additive term of $m^{-\gamma}$ where γ is a constant. They also showed that such a term is necessary. Thus, if $m = O(1)$ then simple tabulation hashing cannot provide nice Chernoff-style tail bounds. In the same paper [AKKR+20], they introduced tabulation-permutation which they showed have Chernoff-style tail bounds for any hash-based sum with an additive term of $|\Sigma|^{-\gamma}$ where γ is a constant.

The big issue with the prior results is that they all contain an extra additive term. If we try to use the tail bounds to bound the central moments of X then we will only obtain something useful for $p = O(1)$ since the additive term will be prohibitive for higher moments. In contrast, if we prove strong bounds for the central moments of X for $p = O(\log n)$ then we can use Markov's inequality to prove a bound the tail that is exponentially decreasing but with an additive term of the form $n^{-\gamma}$ where $\gamma = O(1)$. Thus in some sense, it is more robust to bound the moments compared to bounding the tail.

While most of the study of simple tabulation hashing have focussed on proving tail bounds, there have also been a couple of paper studying the moments of simple tabulation hashing. Braverman et al. [BCLM+10] showed that for a fixed bin the 4th central moment is close to that achieved by truly random hashing. Dahlgaard et al. [DKT17] generalized this to any constant moment p. Their proof works for any p but with a doubly exponential dependence on p, so their bound is only useful for $p = O(1)$. In [HT22], we obtain bounds for all the moments of hash-based sums for simple tabulation hashing which are tight up to constants depending only on c.

4.2 Moments of Tabulation Hashing

In [HT22], we analyze the p-norms of hash-based sums for simple tabulation hashing, and our analysis is the first that provides useful bounds for non-constant moments. Furthermore, it is also the first analysis of simple tabulation hashing that does not assume that c is constant. We obtain an essentially tight understanding of this problem and show that simple tabulation hashing only works well when the range is large. This was also noted by Aamand et al. [AKKR+20] and they solve this deficiency of simple tabulation hashing by introducing a new hashing scheme, tabulation-permutation hashing. We show that it is also possible to break the bad instances of simple tabulation hashing by using mixed tabulation hashing.

We introduce a bit of notation to make the theorems cleaner. We will view a value function $v\colon \Sigma^c \times [m] \to \mathbb{R}$ as a vector, more precisely, we let

$$\|v\|_q = \left(\sum_{x \in \Sigma^c} \sum_{j \in [m]} |v(x,j)|^q \right)^{1/q}$$

for all $q \in [1, \infty]$. For every key $x \in \Sigma^c$ we define $v[x]$ to be the sub-vector v restricted to x, more precisely, we let

$$\|v[x]\|_q = \left(\sum_{j \in [m]} |v(x,j)|^q \right)^{1/q}$$

for all $q \in [1, \infty]$.

4.2.1 Moments of Simple Tabulation Hashing

Tha main result of [HT22] for simple tabulation hashing is a version of Theorem 3.5.

Theorem 4.1 ([HT22]). *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{simple}} = \sum_{x\in\Sigma^c} v(x, h(x))$. Then for all $p \geq 2$*

$$\left\| V_v^{\text{simple}} \right\|_p \leq L_1 \Psi_p \left(K_c \gamma_p^{c-1} M_v, K_c \gamma_p^{c-1} \sigma_v^2 \right) ,$$

where $K_c = (L_2 c)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{ \log(m) + \log\left(\frac{\sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\max_{x\in\Sigma^c} \|v[x]\|_2^2} \right) / c, p \right\}}{\log\left(e^2 m \left(\max_{x\in\Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} \right)^{-1} \right)}$$

It is instructive to compare this result to Theorem 3.5 for fully random hashing. Ignoring the constant K_c, the result for simple tabulation hashing corresponds to the result for fully random hashing if we group keys into groups of size γ_p^{c-1}.

The definition of γ_p is somewhat complicated because of the generality of the theorem, but we will try to explain the intuition behind it. The expression $\max_{x\in\Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}$ measures how spread out the mass of the value function is. It was also noted in the previous analysis by Aamand et al. [AKKR+20] that this measure is naturally occurring. In fact, their result needs that $\max_{x\in\Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} \leq m^{1/4}$. If we consider the example of hashing below a threshold $l \leq m$ where each key, $x \in \Sigma^c$, has weight w_x, then the value function, v, will be $v(x,j) = w_x \left([j < l] - \frac{l}{m} \right)$ for $x \in \Sigma^c, j \in [m]$, and we then get that

$$\max_{x\in\Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} = 4l \left(1 - \frac{l}{m} \right) \leq 4l .$$

This correctly measures that the mass of the value function is mostly concentrated to the l positions of $[m]$.

The expression $\frac{\sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\max_{x\in\Sigma^c} \|v[x]\|_2^2}$ is a measure for how many keys that have significant weight. This also showed up in the previous analyses of simple tabulation hashing [AKKR+20; PT12]. If we again consider the example from before, we get that

$$\frac{\sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\max_{x\in\Sigma^c} \|v[x]\|_2^2} = \frac{\sum_{x\in\Sigma^c} w_x^2}{\max_{x\in\Sigma^c} w_x^2} .$$

We can summarize the example in the following corollary.

Corollary 4.2 ([HT22]). *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, assign a weight, $w_x \in \mathbb{R}$, to every key, $x \in \Sigma^c$, and consider a threshold $l \leq m$. Define the random variable $V_v^{\text{simple}} = \sum_{x\in\Sigma^c} w_x \left([h(x) < l] - \frac{l}{m}\right)$. Then for all $p \geq 2$*

$$\left\|V_v^{\text{simple}}\right\|_p \leq \Psi_p\left(K_c\gamma_p^{c-1}\max_{x\in\Sigma^c}|w_x|, K_c\gamma_p^{c-1}\left(\sum_{x\in\Sigma^c} w_x^2\right)\frac{l}{m}\left(1-\frac{l}{m}\right)\right),$$

where $K_c = L_1\left(L_2 c\right)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c} w_x^2}{\max_{x\in\Sigma^c} w_x^2}\right)/c, p\right\}}{\log\left(\frac{e^2 m}{4l}\right)}$$

A natural question is how close Theorem A.7 is to being tight. We show that if $\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}\right)/c = O\left(\log\left(1 + m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)\right)$ then the result is tight up to a universal constant depending only c. Formally in [HT22], we prove the following lemma.

Theorem 4.3 ([HT22]). *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, and $2 \leq p \leq L_1|\Sigma|\log(m)$, then there exists a value function, $v\colon U \times [m] \to \mathbb{R}$, where $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$, and for which*

$$\left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \geq K_c'\Psi_p\left(\gamma_p^{c-1}M_v, \gamma_p^{c-1}\sigma_v^2\right),$$

where $K_c' = L_1^c$ and L_1 is a universal constant, and

$$\gamma_p = \max\left\{1, \frac{p}{\log\left(e^2 m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)}\right\}$$

4.2.2 Moment of Mixed Tabulation Hashing

The results of simple tabulation hashing work well when the range is large and when the mass of the value function is on few coordinates. In [HT22], we show that mixed tabulation hashing works well even if the range is small.

Theorem 4.4 ([HT22]). *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with $d \geq 1$ derived characters, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{mixed}} = \sum_{x\in\Sigma^c} v(x,h(x))$. For all $p \geq 2$ then*

$$\left\|V_v^{\text{mixed}}\right\|_p \leq \Psi_p\left(K_c\gamma_p^c M_v, K_c\gamma_p^c\sigma_v^2\right)$$

where $K_c = L_1 (L_2 c)^c$, L_1 *and* L_2 *are universal constants, and*

$$\gamma_p = \max\left\{1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)}\right\} .$$

Usually, in hashing contexts, we do not map to a much larger domain, i.e., we will usually have that $m \leq |U|^\gamma$ for some constant $\gamma \geq 1$. If this is the case then we can obtain the following nice tail bound for mixed tabulation hashing by using Markov's inequality.

Corollary 4.5 ([HT22])**.** *Let* $h\colon \Sigma^c \to [m]$ *be a mixed tabulation function with* $d \geq 1$ *derived characters,* $v\colon \Sigma^c \times [m] \to \mathbb{R}$ *a value function, and assume that* $\sum_{j\in[m]} v(x,j) = 0$ *for all keys* $x \in \Sigma^c$*. Define the random variable* $V_v^{\text{mixed}} = \sum_{x\in\Sigma^c} v(x, h(x))$*. If* $m \leq |U|^\gamma$ *for a value* $\gamma \geq 1$ *then for all* $t \geq 0$

$$\Pr\left[\left|V_v^{\text{mixed}}\right| \geq t\right] \leq \exp\left(-\frac{\sigma_v^2}{M_v^2}\mathcal{C}\left(\frac{tM_v}{\sigma_v^2}\right)/K_{c,\gamma}\right) + |U|^{-\gamma} ,$$

where $\mathcal{C}(x) = (x+1)\log(x+1) - x$, $K_{c,\gamma} = L_1\left(L_2 c^2 \gamma\right)^c$, *and* L_1 *and* L_2 *are universal constants.*

4.3 Conclusion

In [HT22], we have studied the moments of hash-based sums of both simple tabulation hashing and mixed tabulation hashing. We have shown that the moments of hash-based sums of simple tabulation hashing are close to the moments of hash-based sums of fully random hashing as long as the number of bins are large. But when the number of bins are small then simple tabulation hashing does not provide good concentration. We show that mixed tabulation hashing does not suffer from this restriction and does in general provide good concentration.

Chapter 5

A Sparse Johnson-Lindenstrauss Transform using Fast Hashing

This chapter is dedicated to presenting the results of our research paper "A Sparse Johnson-Lindenstrauss Transform using Fast Hashing" [HT23] from Appendix B, and includes a slightly modified subset of its introduction.

5.1 Introduction

Dimensionality reduction is an often applied technique to obtain a speedup when working with high dimensional data. The basic idea is to map a set of points $X \subseteq \mathbb{R}^u$ to a lower dimension while approximately preserving the geometry. The Johnson-Lindenstrauss lemma [JL84] is a foundational result in that regard.

Lemma 5.1 ([JL84])**.** *For any $0 < \varepsilon < 1$, integers n, u, and $X \subseteq \mathbb{R}^u$ with $|X| = n$, there exists a map $f \colon X \to \mathbb{R}^m$ with $m = O(\varepsilon^{-2} \log n)$ such that*

$$\forall w, w' \in X,\ \left| \left\| f(w) - f(w') \right\|_2 - \left\| w - w' \right\|_2 \right| \leq \varepsilon \left\| w - w' \right\|_2 .$$

It has been shown in [AK17; LN17] that the target dimension m is optimal for nearly the entire range of n, u, ε. More precisely, for any n, u, ε there exists a set of points $X \subseteq \mathbb{R}^u$ with $|X| = n$ such that for any map $f \colon X \to \mathbb{R}^m$ where the Euclidean norm is distorted by at most $(1 \pm \varepsilon)$ must have $m = \Omega(\min\left\{u, n, \varepsilon^{-2} \log(\varepsilon^2 n)\right\})$.

All known proofs of the Johnson-Lindenstrauss lemma constructs a linear map f. The original proof of Johnson and Lindenstrauss [JL84] chose $f(x) = \Pi x$ where $\Pi \in \mathbb{R}^{m \times u}$ is an appropriately scaled orthogonal projection into a random m-dimensional subspace. Another simple construction is to set $f(x) = \frac{1}{\sqrt{m}} A x$ where $A \in \mathbb{R}^{m \times u}$ and each entry is an independent Rademacher variable. In both cases, it can be shown that as long as $m = \Omega(\varepsilon^{-2} \log 1/\delta)$ then

$$\forall w \in \mathbb{R}^u,\ \Pr\left[\left| \|f(w)\|_2^2 - \|w\|_2^2 \right| \geq \varepsilon \|w\|_2^2 \right] \leq \delta. \tag{5.1}$$

The Johnson-Lindenstrauss lemma follows by setting $\delta < 1/\binom{n}{2}$ and taking $w = z - z'$ for all pairs $z, z' \in X$ together with a union bound. (B.1) is also known as the distributional Johnson-Lindenstrauss lemma and it has been shown that the target dimension m is tight, more precisely, m must be at least $\Omega(\min\{u, \varepsilon^{-2}\log 1/\delta\})$ [JW13; KMN11].

5.1.1 Sparse Johnson-Lindenstrauss Transform

One way to speed up the embedding time is replacing the dense A of the above construction by a sparse matrix. The first progress in that regard came by Achlioptas in [Ach03] who showed that A can be chosen with i.i.d. entries where $A_{ij} = 0$ with probability $2/3$ and otherwise A_{ij} is chosen uniformly in $\pm\sqrt{\frac{3}{m}}$. He showed that this construction can achieve the same m as the best analyses of the Johnson-Lindenstrauss lemma. Hence this achieves essentially a 3x speedup, but the asymptotic embedding time is still $O(m\|x\|_0)$ where $\|x\|_0$ is number of non-zeros of x.

Motivated by improving the asymptotic embedding time, Kane and Nelson in [KN14], following the work in [DKS10; KN10; BOR10], introduced the Sparse Johnson-Lindenstrauss Transform which maps down to essentially optimal dimension $m = O(\varepsilon^{-2}\log n)$ and only has $s = O(\varepsilon^{-1}\log n)$ non-zeros entries per column. This speeds up the embedding time to $O(\varepsilon^{-1}\log n\|x\|_0) = O(\varepsilon m\|x\|_0)$ thus improving the embedding time by a factor of ε^{-1}. It nearly matches a sparsity lower bound by Nelson and Nguyen [NN13] who showed that any sparse matrix needs at least $s = \Omega(\varepsilon^{-1}\log(n)/\log(1/\varepsilon))$ non-zeros per column. Kane and Nelson [KN14] provided two different constructions with the same sparsity. Later a simpler analysis was given in [CJN18] which also generalized the result to a more general class of constructions. In [HT23], we will only focus on one of the constructions which is described below.

We will first consider the related CountSketch which was introduced in [CCF04] and was analyzed for dimensionality reduction in [TZ12]. In CountSketch, we construct the matrix A as follows: We pick a pairwise independent hash function, $h\colon [u] \to [m]$, and a 4-wise independent sign function $\sigma\colon [u] \to \{-1, 1\}$. For each $x \in [u]$, we set $A_{h(x),x} = \sigma(x)$ and the rest of the x'th column to 0. Clearly, this construction has exactly 1 non-zero entry per column. It was shown in [TZ12] that if $m = \Omega(\varepsilon^{-2}\delta^{-1})$ then it satisfies the distributional Johnson-Lindenstrauss lemma, eq. (5.1). The result follows by bounding the second moment of $\|Ax\|_2^2 - \|x\|_2^2$ for any $x \in \mathbb{R}^d$ and then apply Chebyshev's inequality.

The construction of the Sparse Johnson-Lindenstrauss Transform is s CountSketch matrices concatenated and scale the resulting matrix by $\frac{1}{\sqrt{s}}$. This clearly gives a construction that has s non-zero entries per column and as it has been shown in [KN14; CJN18] if $s = \Omega(\varepsilon^{-1}\log(1/\delta))$ then we can obtain the optimal target dimension $m = O(\varepsilon^{-2}\log(1/\delta))$. More formally, we construct the matrix A as follows:

1. We pick a hash function, $h\colon [s] \times [u] \to [m/s]$ and a sign function $\sigma\colon [s] \times [u] \to \{-1, 1\}$.

2. For each $x \in [u]$, we set $A_{i\cdot m/s + h(i,x),x} = \frac{\sigma(i,x)}{\sqrt{s}}$ for every $i \in [s]$ and the rest of the x'th column to 0.

In the previous analyses [KN14; CJN18], it was shown that if h and σ are $\Omega(\log 1/\delta)$-wise independent then the construction works. Unfortunately, it is not practical to use a $\Omega(\log 1/\delta)$-wise independent hash function so the goal of [HT23] is to obtain an analysis of a Sparse Johnson-Lindenstrauss Transform with fewer assumptions about the hash function. In particular, the analysis of [HT23] relax the assumptions of the hash function, h, and the sign function, σ, to just satisfying a decoupling-decomposition and a strong concentration property.

In [HT23], it also shown that Mixed Tabulation satisfies these properties and thus that the Sparse Johnson-Lindenstrauss Transform can be implemented using Mixed Tabulation. Let us describe more formally, what we mean by saying that Mixed Tabulation can implement the Sparse Johnson-Lindenstrauss Transform. We let $h_1 \colon \Sigma^c = [u] \to [m/s]$, $h_2 \colon \Sigma^c \to \Sigma^d$, and $h_3 \colon \Sigma^d \to [m/s]$ be the independent Simple Tabulation hash functions that implement the Mixed Tabulation hash function, $h_1(x) \oplus h_3(h_2(x))$. We then extend it to the domain $[s] \times [u]$ as follows:

1. Let $h_2' \colon [s] \times \Sigma^c \to \Sigma^d$ be defined by $h_2'(i, x) = h_2(x) \oplus \underbrace{(i, \ldots, i)}_{d \text{ times}}$, i.e., each derived character gets xor'ed by i.

2. We then define $h \colon [s] \times [u] \to [m/s]$ and $\sigma \colon [s] \times [u] \to \{-1, 1\}$ by $h(i, x) = h_1(x) \oplus h_3(h_2'(i, x))$ and $\sigma(i, x) = \sigma_1(x) \cdot \sigma_3(h_2'(i, x))$, where h_1 and h_3 are the Simple Tabulation hash functions described above, and $\sigma_1 \colon \Sigma^c \to \{-1, 1\}$ and $\sigma_3 \colon \Sigma^d \to \{-1, 1\}$ are independent Simple Tabulation functions.

5.2 Overview of the New Analysis

The main the technical contribution of [HT23] is a new analysis of the Sparse Johnson-Lindenstrauss Transform that relaxes the assumptions on the hash function, h. In [HT23], we show that if h satisfies a decoupling decomposition property and a strong concentration property then we obtain the same bounds for the Sparse Johnson-Lindenstrauss Transform. Both of these properties are satisfied by h if h is $\Omega(\log 1/\delta)$-wise independent so our assumptions are weaker than those of the previous analyses.

In order to describe the approach of [HT23], we look at the random variable

$$Z = \|Aw\|_2^2 - 1 = \frac{1}{s} \sum_{i \in [s]} \sum_{x \neq y \in [u]} \sigma(i, x)\sigma(i, y) \left[h(i, x) = h(i, y)\right] w_x w_y. \tag{5.2}$$

Here $w \in \mathbb{R}^u$ is a unit vector. With this notation the goal becomes to bound $\Pr[|Z| \geq \varepsilon]$.

The first step in the analysis is to decouple eq. (5.2). Decoupling was also used in one of the proofs in [CJN18], but since we want to prove the result for more general hash functions, we cannot directly use the standard decoupling inequalities. We will instead assume that our hash function allows a *decoupling-decomposition*. The definition of a decoupling-decomposition is a bit technical and we postpone the formal definition to Appendix B. Intuitively speaking the idea of the decoupling-decomposition is to decompose the universe

U into pieces $(U_\alpha)_\alpha$ where each of the pieces can be decoupled. Thus, for simplicity, we will assume that our hash function allows for the standard decoupling inequality. If we apply Markov's inequality and a standard decoupling inequality for fully random hashing we obtain the expression.

$$\begin{aligned}\Pr[|Z| \geq \varepsilon] &\leq \varepsilon^{-p} \operatorname{E}[|Z|^p] \\ &\leq \left(\varepsilon^{-1}\frac{4}{s}\right)^p \operatorname{E}\left[\left|\sum_{i\in[s]}\sum_{x,y\in[u]} \sigma(i,x)\sigma'(i,y)\left[h(i,x)=h'(i,y)\right] w_x w_y\right|^p\right] \end{aligned} \tag{5.3}$$

where (h', σ') are independent copies of (h, σ) and $p \geq 2$. The power of decoupling stems from the fact that it breaks up some of the dependencies and allows for a simpler analysis.

The goal is now to analyze $\left\|\sum_{i\in[s]}\sum_{x,y\in[u]} \sigma(i,x)\sigma'(i,y)\left[h(i,x)=h'(i,y)\right] w_x w_y\right\|_p$. First we fix (h', σ') and define the value function $v\colon [s]\times[u]\times[m/s]\to\mathbb{R}$ by $v(i,x,j) = w_x \sum_{y\in[u]} \sigma'(i,y)\left[h(i,y)=j\right] w_y$. We then use the randomness of (h,σ) to bound the hash-based sum $\sum_{i\in[s]}\sum_{x\in[u]}\sigma(i,x)v(i,x,h(i,x))$. In order to do this, we will assume that (h,σ) is *strongly concentrated*. The formal definition of strongly concentrated is deferred to Appendix B, but informally speaking, it requires the pair (h,σ) to have similar concentration as fully random hashing, that is, it satisfies a lemma akin to Theorem 3.5.

Now we note that $v(i,x,j) = w_x a_{i,j}$ where $a_{i,j} = \sum_{y\in[u]}\sigma'(i,y)\left[h(i,y)=j\right] w_y$. We then take the view that $|a_{ij}|$ is the load of the bin $(i,j)\in[s]\times[m/s]$. We can then split $[s]\times[m/s]$ into heavy and light bins and handle each separately.

In [HT23], we show that the contribution from the light bins is as if the collisions are independent. This should be somewhat intuitive since if we only have few collisions in each bin then the collisions behave as if they were independent. In contrast, we show that the contribution from the heavy bins is dominated by the heaviest bin. This turns out to be exactly what we need to finish the analysis. The following is an informal statement of the main technical lemma of [HT23].

Lemma 5.2 ([HT23], informal)**.** *Let $h, \overline{h}\colon [s]\times U\to[m/s]$ be hash functions and $\sigma, \overline{\sigma}\colon [s]\times U\to\{-1,1\}$ be sign functions. Assume that (h,σ) and $(\overline{h},\overline{(\sigma)})$ are strongly concentrated then for all vectors $w\in\mathbb{R}^U$,*

$$\begin{aligned}&\left\|\sum_{i\in[s]}\sum_{x,y\in U} \sigma(i,x)\overline{\sigma}(i,y)\left[h(i,x)=\overline{h}(i,y)\right] w_x w_y\right\|_p \\ &\qquad\leq \Psi_p\left(L\left\|w\right\|_2^2, L\frac{s^2}{m}\left\|w\right\|_2^4\right) + L\frac{p}{\log m/s}\left\|w\right\|_2^2.\end{aligned}$$

Here L is a constant only depending on (h,σ) and $(\overline{h},\overline{(\sigma)})$.

If we combine the technical lemma with a decoupling-decomposition then we obtain the main result of [HT23] which stated informally is the following.

Theorem 5.3 ([HT23], informal)**.** *Let* $h\colon [s]\times[u]\to[m/s]$ *be a hash function and* $\sigma\colon [s]\times[u]\to\{-1,1\}$ *be a sign function. Furthermore, let* $0<\varepsilon<1$ *and* $0<\delta<1$ *be given.*

Assume that (h,σ) *allows for a decoupling-decomposition and that* (h,σ) *is strongly concentrated then the following is true*

$$\Pr[|Z|\geq\varepsilon]\leq\delta.$$

5.3 Conclusion

In [HT23], we have provided a new analysis of a sparse Johnson-Lindenstrauss transform with fewer assumptions on the hash function. With this new analysis we have shown that it is possible to implement a sparse Johnson-Lindenstrauss transform using a practical hash functions, namely, mixed tabulation hashing.

Chapter 6

Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search

This chapter is dedicated to presenting the results of our research paper "Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search" [AK20] from Appendix C, and includes a slightly modified subset of its introduction.

6.1 Introduction

Set Similarity Search (SSS) is the problem of indexing sets (or sparse boolean data) to allow fast retrieval of sets, similar under a given similarity measure. The sets may represent one-hot encodings of categorical data, "bag of words" representations of documents, or "visual/neural bag of words" models, such as the Scale-invariant feature transform (SIFT), that have been discretized. The applications are ubiquitous across Computer Science, touching everything from recommendation systems to gene sequences comparison. See [CCT10; JZYY+18] for recent surveys of methods and applications.

Set similarity measures are any function, s that takes two sets and return a value in $[0, 1]$. Unfortunately, most variants of Set Similarity Search, such as Partial Match, are hard to solve assuming popular conjectures around the Orthogonal Vectors Problem [Wil05; APRS16; ARW17; CW19], which roughly implies that the best possible algorithm is to not build an index, and "just brute force" scan through all the data, on every query. A way to get around this is to study Approximate SSS: Given a query, q, for which the most similar set y has similarity$(q, y) \geq s_1$, we are allowed to return any set y' with similarity$(q, y') > s_2$, where $s_2 < s_1$. In practice, even the best *exact* algorithms for similarity search use such an (s_1, s_2)-approximate solution as a subroutine [CPT18].

The question is made harder by the fact that previous algorithms study the problem under different similarity measures, such as Jaccard, Cosine, or Braun-Blanquet similarity. The only thing those measures have in common is that they can be defined as a function f of the sets sizes, the universe size, and the intersection size. In other words,

similarity$(q, y) = f(|q|, |y|, |q \cap y|, |U|)$ where $|U|$ is the size of the universe from which the sets are taken. In fact, any symmetric measure of similarity for sets must be defined by those four quantities.

Hence, to fully solve Set Similarity Search, we avoid specifying a particular similarity measure, and instead define the problem solely from those four parameters. This generalized problem is what we solve optimally in [AK20], for all values of the four parameters:

Definition 6.1 (The (w_q, w_u, w_1, w_2)-GapSS problem)**.** Given some universe U and a collection $Y \subseteq \binom{U}{w_u|U|}$ of $|Y| = n$ sets of size $w_u|U|$, build a data structure that for any query set $q \in \binom{U}{w_q|U|}$: either returns $y' \in Y$ with $|y' \cap q| > w_2|U|$; or determines that there is no $y \in Y$ with $|y \cap q| \geq w_1|U|$.

For the problem to make sense, we assume that $w_q|U|$ and $w_u|U|$ are integers, that $w_q, w_u \in [0, 1]$, and that $0 < w_2 < w_1 \leq \min\{w_q, w_u\}$. Note that $|U|$ may be very large, and as a consequence the values w_q, w_u, w_1, w_2 may all be very small.

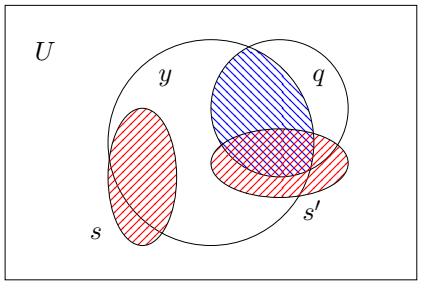
At first sight, the problem may seem easier than the version where the sizes of sets may vary. However, the point is that making polylog(n) data-structures for sets and queries of progressively bigger sizes, immediately yields data structures for the original problem. Similarly, any algorithm assuming a specific set similarity measure also yields an algorithm for (w_q, w_u, w_1, w_2)-GapSS, so the lower bounds of [AK20] also hold for all previously studied SSS problems.

Example 1 As an example, assume we want to solve the Subset Search Problem, in which we, given a query q, want to find a set y in the database, such that $y \subseteq q$. If we allow a two-approximate solution, GapSS includes this problem by setting $w_1 = w_u$ and $w_2 = w_1/2$: The overlap between the sets must equal the size of the stored sets; and we are guaranteed to return a y' such that at least $|q \cap y'| \geq |y|/2$.

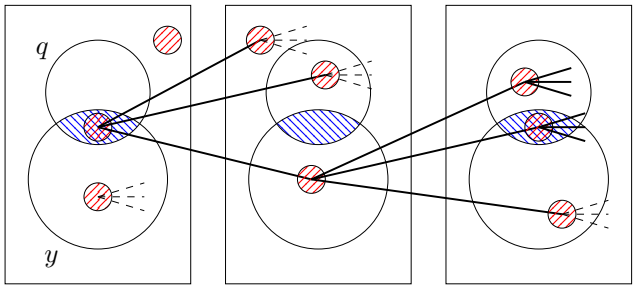
Example 2 In the (j_1, j_2)-Jaccard Similarity Search Problem, given a query, q, we must find y such that the Jaccard Similarity $|q \cap y|/|q \cup y| > j_2$ given that a y' exists with similarity at least j_1. After partitioning the sets by size, we can solve the problem using GapSS by setting $w_1 = \frac{j_1(w_q+w_u)}{1+j_1}$ and $w_2 = \frac{j_2(w_q+w_u)}{1+j_2}$. The same reduction works for any other similarity measure with polylog(n) overhead.

The version of this problem where $w_2 = w_q w_u$ is similar to what is in the literature called *"the random instance"* [Pan06; Laa15; ALRW17]. To see why, consider generating $n-1$ sets independently at random with size $w_u|U|$, and a "planted" pair, (q, y), with size respectively $w_q|U|$ and $w_u|U|$ and with intersection $|q \cap y| = w_1|U|$. Insert the size $w_u|U|$ sets into the database and query with q. Since q is independent from the $n-1$ original sets, its intersection with those is strongly concentrated around the expectation $w_q w_u|U|$. Thus, if we parametrize GapSS with $w_2 = w_q w_u + o(1)$, the query for q is guaranteed to return the planted set y.

There is a tradition in the Similarity Search literature for studying such this independent case, in part because *it is expected that one can always reduce to the random instance*, for example using the techniques of "data-dependent hashing" [AINR14; AR15]. However,



(a) Two cohorts, y and q with a large intersection (blue). The first representative set, s, favours y, while the second, s', favours both y and q.



(b) Branching random walk run on two cohorts q and y. The bold lines illustrate paths considered by sets, while the dashed lines adorn paths only considered by only one of x or y. Here q has a higher threshold ($t_q = 2/3$) than y ($t_u = 1/2$), so q only considers paths starting with two favourable representatives.

Figure 6.1: The representative sets, coloured in red, are scattered in the universe to provide an efficient space partition for the data.

for such a reduction to make sense, we would first need an optimal "data-independent" algorithm for the $w_2 = w_q w_u$ case, which is what we provide in [AK20].

6.2 Supermajorities

In Social Choice Theory a supermajority is when a fraction strictly greater than $1/2$ of people agree about something.In the analysis of Boolean functions a t-supermajority function $f : \{0,1\}^n \to \{0,1\}$ can be defined as 1, if a fraction $\geq t$ of its arguments are 1, and 0 otherwise. We will sometimes use the same word for the requirement that a fraction $\leq t$ of the arguments are 1.

The main conceptual point of our algorithm is the realization that an optimal algorithm for Set Similarity Search must take advantage of the information present in the given sets, as well as that present in their complement. A similar idea was leveraged by Cohen et al. [CK09] for Set Similarity *Estimation*. In [AK20], we show that there is a better way of combining this information and that doing so results in an optimal hashing based data structure for the entire parameter space of random instance GapSS.

The algorithm (idealized) While our data structure is technically a tree with a carefully designed pruning rule, the basic concept is very simple.

We start by sampling a large number of "representative sets" $R \subseteq \binom{U}{k}$. Here roughly $|R| \approx n^{\log n}$ and $k \approx \log n$. Given family $Y \subseteq \binom{U}{w_u|U|}$ of sets to store, which we call "cohorts", we say that $r \in R$ "t-favours" the cohort y if $|y \cap r|/|r| \geq t$. Representing sets as vectors in $\{0,1\}^d$, this is equivalent to saying $f_t(r \cap y) = 1$, where f_t is the t-supermajority function. (If t is less than w_u, the expected size of the overlap, we instead require $|y \cap r|/|r| \leq t$.)

Given the parameters $t_q, t_u \in [0,1]$, the data-structure is a map from elements of R to the cohorts they t_u-favour. When given a query $q \in \binom{U}{w_q|U|}$, (a $w_q|U|$ sized cohort), we compare it against all cohorts y favoured by representatives $r \in R$ which t_q-favour q (that is $|q \cap r|/|r| \geq t_q$). This set $R_{t_q}(q)$ is much smaller than $|R|$ (we will have $|R_{t_q}(q)| \approx n^{\varepsilon}$ and $E[|R_{t_u}(y) \cap R_{t_q}(q)|] \approx n^{\varepsilon - 1}$), so the filtering procedure greatly reduces the number of cohorts we need to compare to the query from n to n^{ε} (where $\varepsilon = \rho_q < 1$ is defined later.)

The intuition is that while it is quite unlikely for a representative to favour a given cohort, and it is *very* unlikely for it to favour two given cohorts (q and y). So if it does, the two cohorts probably have a substantial overlap. Figure 6.1a has a simple illustration of this principle.

In order to fully understand supermajorities, we want to understand the probability that a representative set is simultaneously in favour of two distinct cohorts given their overlap and representative sizes. This paragraph is a bit technical and may be skipped at first read. Chernoff bounds in $\mathbb{R}$ are a common tool in the community, and for iid. $X_i \sim \text{Bernoulli}(p) \in \{0,1\}$ the sharpest form (with a matching lower bound) is $\Pr[\sum X_i \geq tn] \leq \exp(-n\,\mathrm{d}(t \parallel p))$, which uses the binary KL-Divergence $\mathrm{d}(t \parallel p) = t \log \frac{t}{p} + (1-t) \log \frac{1-t}{1-p}$. The Chernoff bound for $\mathbb{R}^2$ is less common, but likewise has a tight description in terms of the KL-Divergence between two discrete distributions: $\mathrm{D}(P \parallel Q) = \sum_{\omega \in \Omega} P(\omega) \log \frac{P(\omega)}{Q(\omega)}$ (summing over the possible events). In our case, we represent the four events that can happen as we sample an element of U as a vector $X_i \in \{0,1\}^2$. Here $X_i = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ means the ith element hit both cohorts, $X_i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ means it hit only the first and so on. We represent the distribution of each X_i as a matrix $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$, and say $X_i \sim \text{Bernoulli}(P)$ iid. such that $\Pr[X_i = \left[\begin{smallmatrix}1-j\\1-k\end{smallmatrix}\right]] = P_{j,k}$. Then $\Pr[\sum X_i \geq \left[\begin{smallmatrix}t_q\\t_u\end{smallmatrix}\right] n] \approx \exp(-n\,\mathrm{D}(T \parallel P))$ where $T = \begin{bmatrix} t_1 & t_q - t_1 \\ t_u - t_1 & 1 - t_q - t_u + t_1 \end{bmatrix}$ and $t_1 \in [0, \min\{t_u, t_q\}]$ minimizes $\mathrm{D}(T \parallel P)$. (Here the notation $\left[\begin{smallmatrix}x\\y\end{smallmatrix}\right] \geq \left[\begin{smallmatrix}t_u\\t_q\end{smallmatrix}\right]$ means $x \geq t_u \wedge y \geq t_q$.)

These bounds above would immediately allow a cell probe version of our upper bound Theorem 6.2, e.g. a query would require $n^{\frac{\mathrm{D}(T_1 \| P_1) - \mathrm{d}(t_q \| w_q)}{\mathrm{D}(T_2 \| P_2) - \mathrm{d}(t_q \| w_q)}}$ probes, where $P_i = \begin{bmatrix} w_i & w_q - w_1 \\ w_u - w_i & 1 - w_q - w_u + w_i \end{bmatrix}$ and T_i defined accordingly. The algorithmic challenge is that for optimal performance, $|R|$ must be in the order of $\Omega(n^{\log n})$, and so checking which representatives favour a given cohort takes super polynomial time!

We augment the above representative sampling procedure as follows: Instead of independent sampling sets, we (implicitly) sample a large, random height k tree, with nodes being elements from the universe. The representative sets are taken to be each path from the root to a leaf. Hence, some sets in R share a common prefix, but mostly they are still independent. We then add the extra constraint that *each of the prefixes of a representative has to be in favour of a cohort*, rather than only having this requirement on the final set. This is the key to making the tree useful: Now given a cohort, we walk down the tree, pruning any branches that do not consistently favour a supermajority of the cohort. Figure 6.1b has a simple illustration of this algorithm. This pruning procedure can be shown to imply that we only spend time on representative sets that end up being in favour of our cohort, while only weakening the geometric properties of the idealized algorithm

negligibly.

While conceptually simple and easy to implement (modulo a few tricks to prevent dependency on the size of the universe, $|U|$), the pruning rule introduces dependencies that are quite tricky to analyze sufficiently tight. The way to handle this will be to consider the tree as a "branching random walk" over $\mathbb{Z}_+^2$ where the value represents the size of the representative's intersection with the query and a given set respectively. The paths in the random walk at step i must be in the quadrant $[t_q i, i] \times [t_u i, i]$ while only increasing with a bias of $\left[\begin{smallmatrix} w_q \\ w_u \end{smallmatrix}\right]$ per step. The branching factor is carefully tuned to just the right number of paths survive to the end.

6.3 Main Results

Results on approximate similarity search are usually phrased in terms of two quantities: (1) The "query exponent" $\rho_q \in [0, 1]$ which determines the query time by bounding it by $O(n^{\rho_q})$; (2) The "update exponent" $\rho_u \in [0, 1]$ which determines the time required to update the data structure when a point is inserted or deleted in Y and is given by $O(n^{\rho_u})$. The update exponent also bounds the space usage as $O(n^{1+\rho_u})$. Given parameters (w_q, w_u, w_1, w_2), the important question is for which pairs of (ρ_q, ρ_u) there exists data structures. E.g. given a space budget imposed by ρ_u, we ask how small can one make ρ_q?

6.3.1 Upper Bound

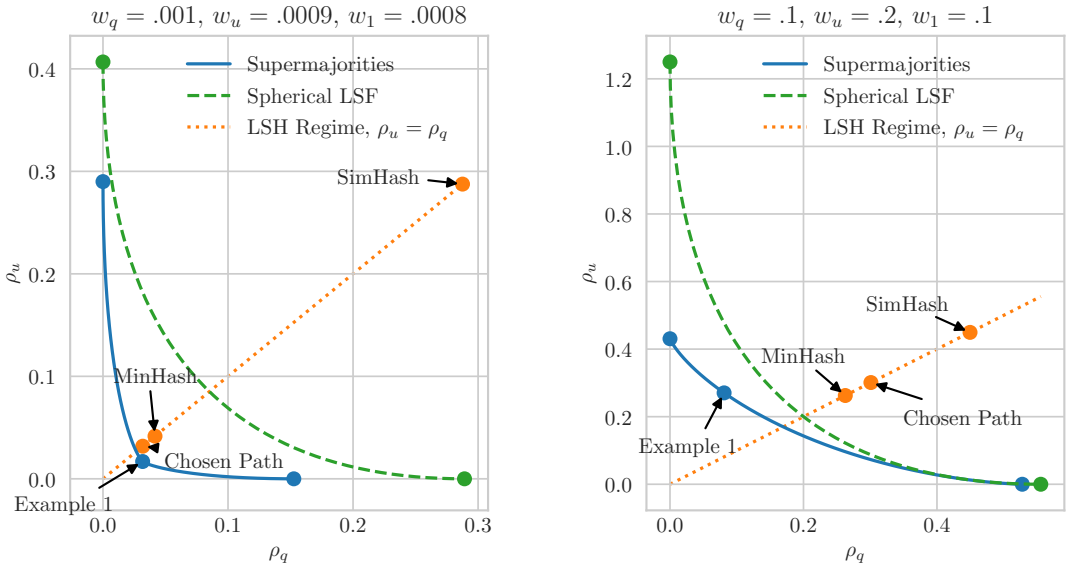
As discussed, the performance of our algorithm is described in terms of KL-divergences. To ease understanding, we give a number of special cases, in which the general bound simplifies. The bounds in this section assume w_q, w_u, w_1, w_2 are constants. See Appendix C for a version without this assumption.

Theorem 6.2 ([AK20])**.** *For any choice of constants $w_q, w_u \geq w_1 \geq w_2 \geq 0$ and $1 \geq t_q, t_u \geq 0$ we can solve the (w_q, w_u, w_1, w_2)-GapSS problem over universe U with query time $\tilde{O}(n^{\rho_q} + w_q|U|) + n^{o(1)}$ and auxiliary space usage $\tilde{O}(n^{1+\rho_u})$, where*

$$\rho_q = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)}, \quad \rho_u = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)}. \tag{6.1}$$

and T_1, T_2 are distributions with expectation $\left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]$ minimizing respectively $\mathrm{D}(T_1 \parallel P_1)$ and $\mathrm{D}(T_2 \parallel P_2)$, as described in Section 6.2.

The two bounds differ only in the $\mathrm{d}(t_q \parallel w_q)$ and $\mathrm{d}(t_u \parallel w_u)$ terms in the numerator. The thresholds t_q and t_u can be chosen freely in $[0, 1]^2$. Varying them compared to each other allows a full space/time trade-off with $\rho_q = 0$ in one end and $\rho_u = 0$ (and $\rho_q < 1$) in the other. Note that for a given GapSS instance, there are many (t_q, t_u) which are not optimal anywhere on the space/time trade-off. Using Lagrange's condition $\nabla\rho_q = \lambda\nabla\rho_u$ one gets a simple equation that all optimal (t_q, t_u) trade-offs must satisfy. Figure 6.2 provides some additional intuition for how the ρ values behave for different settings of GapSS.



(a) Example of search with very small sets. (b) Example with larger sets of different sizes.

Figure 6.2: Comparison to Spherical LSF: Plots of the achievable ρ_q (time exponent) and ρ_u (space exponent) achievable with Theorem 6.2. The plots are drawn in the "random setting", $w_2 = w_q w_u$ where Spherical LSF and Data-Dependent LSH coincide.

Example 1: Near balanced ρ values. As noted, many pairs (t_q, t_u) are not optimal on the trade-off, in that one can reduce one or both of ρ_q, ρ_u by changing them. The pairs that are optimal are not always simple to express, so it is interesting to study those that are. One such particularly simple choice on the Lagrangian is $t_q = 1 - w_u$ and $t_u = 1 - w_q$. This point is special because the values of t_q and t_u depend only on w_u and w_q, while in general they will also depend on w_1 and w_2. In this setting we have $T_i = \begin{bmatrix} 1-w_q-w_u+w_i & w_u-w_i \\ w_u-w_i & w_i \end{bmatrix}$, which can be plugged into Theorem 6.2.

In the case $w_q = w_u = w$ we get the balanced ρ values $\rho_q = \rho_u = \log(\frac{w_1}{w}\frac{1-w}{1-2w+w_1})/\log(\frac{w_2}{w}\frac{1-w}{1-2w+w_2})$ in which case it is simple to compare with Chosen Path's ρ value of $\log(\frac{w_1}{w})/\log(\frac{w_2}{w})$. Chosen Path on balanced sets was shown in [CP17] to be optimal for w, w_1, w_2 small enough, and we see that Supermajorities do indeed recover this value for that range.

Example 2: Linear space/constant time. Setting t_1 in $T_1 = \begin{bmatrix} t_1 & t_q-t_1 \\ t_u-t_1 & 1-t_q-t_u+t_1 \end{bmatrix}$ such that either $\frac{t_1}{w_1} = \frac{t_q-t_1}{w_q-w_1}$ or $\frac{t_1}{w_1} = \frac{t_u-t_1}{w_u-w_1}$ we get respectively $\mathrm{D}(T_1 \parallel P_1) = \mathrm{d}(t_q \parallel w_q)$ or $\mathrm{D}(T_1 \parallel P_1) = \mathrm{d}(t_u \parallel w_u)$. Theorem 6.2 then yields algorithms with either $\rho_q = 0$ or $\rho_u = 0$ corresponding to either a data structure with $\approx e^{\tilde{O}(\sqrt{\log n})}$ query time, or with $\tilde{O}(n)$ auxiliary space. Like [ALRW17] we have $\rho_q < 1$ for any parameter choice, even when $\rho_u = 0$.

6.3.2 Lower Bound

Since the first lower bounds on Locality Sensitive Hashing [MNP06], lower bounds for approximate near neighbours have split into two kinds: (1) Cell probe lower bounds [PTW08; PTW10; ALRW17] and (2) Lower bounds in restricted models [ODo14; AR16; ALRW17; CP17]. The most general such model for data-independent algorithms was formulated by [ALRW17] and defines a type of data structure called "list of points":

Definition 6.3 (List-of-points)**.** Given some universes, Q, U, a similarity measure $S : Q \times U \to [0,1]$ and two thresholds $1 \geq s_1 > s_2 \geq 0$,

1. We fix (possibly random) sets $A_i \subseteq \{-1,1\}^d$, for $1 \leq i \leq m$; and with each possible query point $q \in \{-1,1\}^d$, we associate a (random) set of indices $I(q) \subseteq [m]$;
2. For a given dataset P, we maintain m lists of points $L_1, L_2, \ldots, L_m$, where $L_i = P \cap A_i$.
3. On query q, we scan through each list L_i for $i \in I(q)$ and check whether there exists some $p \in L_i$ with $S(q,p) \geq s_2$. If it exists, return p.

The data structure succeeds, for a given $q \in Q, p \in P$ with $S(q,p) \geq s_1$, if there exists $i \in I(q)$ such that $p \in L_i$. The total space is defined by $S = m + \sum_{i \in [m]} |L_i|$ and the query time by $T = |I(q)| + \sum_{i \in I(q)} |L_i|$.

The List-of-points model contains all known Similarity Search data structures, except for the so-called "data-dependent algorithms". It is however conjectured [ALRW17] that data-dependency does not help on random instances (recall this corresponds to $w_2 = w_q w_u$), which is the setting of Theorem 6.4.

In [AK20], we present two main lower bounds: (1) That requires $w_q = w_u$ and $\rho_q = \rho_u$ and (2) That requires $w_2 = w_q w_u$. In this introduction, we will only focus on the latter and refer the reader to Appendix C for a discussion of the former.

The lower bound are concerned with random instances. This is powerful since every known data-dependent algorithm reduces the general case to a random instance. It is even conjectured that data-dependence does not help on random instances [ALRW17]. This lower bound is tight for any $0 < w_q w_u < w_1 < \min\{w_q, w_u\}$ in the list-of-points model.

Theorem 6.4 ([AK20])**.** *Consider any list-of-point data structure for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem over a universe of size d of n points with $w_q w_u d = \omega(\log n)$, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $\alpha \in [0,1]$ we have that*

$$\alpha\rho_q + (1-\alpha)\rho_u \geq \inf_{\substack{t_q, t_u \in [0,1] \\ t_u \neq w_u}} \left(\alpha \frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{d}(t_u \parallel w_u)} + (1-\alpha) \frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{d}(t_u \parallel w_u)} \right),$$

where $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ *and* $T = \underset{T \ll P, \underset{X \sim T}{E}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]}{\arg\inf} \mathrm{D}(T \parallel P).$

Note that for $w_2 = w_q w_u$, the term $\mathrm{D}(T_2 \parallel P_2)$, in Theorem 6.2, splits into $\mathrm{d}(t_q \parallel w_q) + \mathrm{d}(t_u \parallel w_u)$, and so the upper and lower bounds perfectly match. This shows that for any linear combination of ρ_q and ρ_u our algorithm obtains the minimal value. By continuity of the terms, this equivalently states as saying that no list-of-points algorithm can get a better query time than our Theorem 6.2, given a space budget imposed by ρ_u.

Example 1: Choices for t_q and t_u. As in the upper bounds, it is not easy to prove that a particular choice of t_q and t_u minimizes the lower bound. One might hope that having corresponding lower and upper bounds would help in this endeavour, but alas both results have a minimization. E.g. setting $t_q = 1 - w_u$ and $t_u = 1 - w_q$ the expression in Theorem 6.4 we obtain the same value as in Theorem 6.2, however it could be (though we strongly conjecture not) that another set of values would reduce both the upper and lower bound.

The good news is that the hypercontractive inequality by Oleszkiewicz [Ole03], can be used to prove certain optimal choices on the space/time trade-off. In particular, in [AK20], we show that for $w_q = w_u = w$ the choice $t_q = t_u = 1 - w$ is optimal in the lower bound, and matches exactly the value $\rho = \log\left(\frac{w_1(1-w)}{w(1-2w+w_1)}\right) / \log(\frac{w_2(1-w)}{w(1-2w+w^2)})$ from Example 1 in the Upper Bounds section.

6.4 Conclusion

In [AK20], we have studied the problem of approximate set similarity search. We provide a data structure that solves the problem optimally in the random setting. Our data structure exploits information both in the sets and their complements by using Boolean supermajority functions. Ideas and concepts from branching random walks in $\mathbb{Z}^2$ are employed in order to turn the abstract ideas into an actual algorithm.

Chapter 7

Load Balancing with Dynamic Set of Balls and Bins

This chapter is dedicated to presenting the results of our research paper "Load Balancing with Dynamic Set of Balls and Bins" [AKT21] from Appendix D, and includes a slightly modified subset of its introduction.

7.1 Introduction

Load balancing in dynamic environments is a central problem in designing several networking systems and web services [SMLK+03; KLLP+97]. We wish to allocate *clients* (also referred to as *balls*) to *servers* (also referred to as *bins*) in such a way that none of the servers gets overloaded. Here, the *load* of a server is the number of clients allocated to it. We want a hashing-style solution where we given the ID of a client can efficiently find its server. Both clients and servers may be added or removed in any order, and with such changes, we do not want to move too many clients. Thus, while the dynamic allocation algorithm has to always ensure a proper load balancing, it should aim to minimize the number of clients moved after each change to the system. For every update in the system, we need to change the allocation of clients to servers. For simplicity, we assume that the updates (ball and bin insertions and removals) do not happen simultaneously and will be operated one at a time, so that we have time to finish changing the allocation before we get another update. Such allocation problems become even more challenging when we face hard constraints in the capacity of each server, that is, each server has a *capacity* and the load may not exceed this capacity. Typically, we want capacities close to the average loads.

There is a vast literature on solutions in the much simpler case where the set of servers is fixed and only the client set is updated. For now, we focus on solutions that are known to work in our fully-dynamic case where both clients and servers can be added and removed in an arbitrary order. This rules out solutions where only the last added server may be removed. The above problem formulation is very general, and does not assume anything

about the ratio between the number of clients n, and the number of servers m. However, it is also conceivable having a system with many clients or a balanced system with $n \approx m$.

The classic solution to the scenario where both clients and servers can be added and removed is *consistent hashing* [SMLK+03; KLLP+97] where the current clients are assigned in a random way to the current servers. While consistent hashing schemes minimize the expected number of movements, they may result in hugely overloaded servers, and they do not allow for explicit capacity constraints on the servers. More recently, Mirrokni et al. [MTZ18] presented an algorithm that works with arbitrary capacity constraints on the servers. For the purpose of load balancing, the system designer can specify a balancing parameter $c = 1 + \varepsilon$, guaranteeing that the maximum load is at most $\lceil cn/m \rceil$. While maintaining this hard balancing constraint, they limit the expected number of clients to be moved when clients or servers are inserted or removed. From a more practical perspective, we think of the load balancing parameter $c = 1 + \varepsilon$ as a simple knob which captures the tradeoff between load balancing and stability upon changes in the system. This gives a more direct control to the system designer in meeting explicit balancing constraints.

With the algorithm from [MTZ18], while guaranteeing a balancing parameter $c = 1 + \varepsilon \leq 2$, when a client is added or removed, the expected number of clients moved is $O(\frac{1}{\varepsilon^2})$. When a server is added or removed, the expected number of clients moved is $O(\frac{n}{\varepsilon^2 m})$. These numbers are only a factor $O(\frac{1}{\varepsilon^2})$ worse than the general lower bounds without capacity constrains.

Before going into the new scheme from [AKT21], it instructive to first consider a simpler probabilistic problem, in order to understand the results of the new scheme. The probabilistic problem is as follows: Consider placing n balls in m bins, each of capacity $C = (1+\varepsilon)n/m$, one ball at the time, where each ball picks a uniformly random non-full bin. We are interested in the number of non-full bins both in expectation and with concentration bounds. This relatively simple problem had not been analyzed before [AKT21]. To state the bounds from [AKT21], we define

$$f = \begin{cases} \varepsilon C & \text{if } C \leq \log 1/\varepsilon \\ \varepsilon\sqrt{C} \cdot \sqrt{\log(1/(\varepsilon\sqrt{C}))} & \text{if } \log 1/\varepsilon \leq C < \frac{1}{2\varepsilon^2} \\ 1 & \text{if } C \geq \frac{1}{2\varepsilon^2} \end{cases}, \tag{7.1}$$

whenever $0 < \varepsilon \leq 1$ and $C \geq 1$ is integral. The problem was solved in [AKT21] and we proved the following result.

Theorem 7.1 ([AKT21]). *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$ be such that $C = (1 + \varepsilon)n/m$ is integral. Moreover assume that that $1/\varepsilon = m^{o(1)}$. Suppose we distribute n balls sequentially into m bins each of capacity C, for each ball choosing a uniformly random non-full bin. The expected fraction of non-full bins is $\Theta(f)$.*

How does this result relate to our dynamic load allocation problem? We can think of the distribution scheme in the theorem as the algorithmically *weakest* way to assign the balls to the capacitated bins. Here, by algorithmically weak, we mean that it cannot be implemented in the dynamic setting where balls and bins can come and go. However, it is

still helpful to think of it as the mathematically *ideal* way of solving dynamic load allocation with bounded loads in the following sense. Imagine that an insertion of a ball is carried out by repeatedly choosing a random bin until we find a non-full one where we place the ball. Then we avoid all the unpleasant dependencies between the loads of the bins visited during the insertion that arise in algorithmically stronger schemes. For example, one can compare to a scheme like linear probing where the cascading effect of balls causes heavy dependencies between the loads of bins visited during a search or an insertion. It follows from Theorem 7.1 that in the simple scheme above, the expected number of bins visited when making an insertion is $O(1/f)$. The main contribution of [AKT21] is to present a much stronger scheme which supports general insertions and deletions of both balls and bins, and which, nonetheless, achieves complexity bounds that are analogous to those in the mathematically ideal scheme above. To be precise, with the scheme of [AKT21], we expect to move $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin and this is tight. Similar bounds holds on the number of bins visited when performing any of these updates.

7.2 Consistent Hashing

The standard solution to our fully-dynamic allocation problem is variations of consistent hashing [SMLK+03; KLLP+97]. The new scheme of [AKT21] combines ideas from these variations, so we start by reviewing those.

7.2.1 Simple Consistent Hashing

In the simplest version of consistent hashing, we hash the active balls and bins onto a unit circle, that is, we hash to the unit interval, using the hash values to create a circular order of balls and bins. Assuming no collisions, a ball is placed in the bin succeeding it in the clockwise order around the circle. One of the nice features of consistent hashing is that it is history-independent, that is, we only need to know the IDs of the balls and the bins and the hash functions, to compute the distribution of balls in bins. If a bin is closed, we just move its balls to the succeeding bin. Similarly, when we open a new bin, we only have to consider the balls from the succeeding bin to see which ones belong in the new bin.

With n balls, m bins, and a fully random hash function h, each bin is expected to have n/m balls. This is also the number of balls we expect to move when a bin is opened or closed.

The problem with simple consistent hashing is that the maximum load on a bin is much larger than the average load, approximately $\Theta(\log m)$ times bigger. This is due to the significant variation in the coverage of bins. Some bins cover intervals of size $\Theta(\frac{\log m}{m})$ and are expected to receive $\Theta(\frac{n \log m}{m})$ balls, resulting in a higher maximum load.

Additionally, there is an issue where the expected number of balls landing in the same bin as a given ball is nearly twice the average. This is because the expected distance between neighboring bins for a ball is $1/(m+1)$, leading to an interval size of $2/(m+1)$. Consequently, approximately $2n/m$ other balls are expected to land in the same bin as

the given ball, causing each bin's load to be almost twice the average. This negatively affects the server's performance, as the load determines its efficiency in serving clients.

7.2.2 Consistent Hashing with Virtual Bins

To get a more uniform bin cover, [KLLP+97] suggests the use of *virtual bins*. The virtual bin trick is that the ball contents of $k = O(\log m)$ virtual bins is united in a single super bin. The super bins are the m bins seen by the user of the system. Internally it is the km virtual bins we place on the cycle together with the n balls. Each virtual bin has a pointer to its super bin. To place a ball, we go along the cycle to the first virtual bin, and then we follow the pointer to its super bin.

A super bin covers the union of the intervals covered by its k virtual bins. The point is that for any constant $\varepsilon > 0$, if we pick a large enough $k = O((\log m)/\varepsilon^2)$, then with high probability, each super bin covers a fraction $(1 \pm \varepsilon)/m$ of the unit cycle.

We note that many other methods have been proposed to maintain such a uniform bin cover as bins are added and removed (see, e.g., [BSS00; GH05; Man04; KM05; KR06; TR98]), and in our algorithms, we shall also employ such virtual bins.

With a uniform bin cover, balls distribute uniformly between bins. On the positive side, in the heavily loaded case when n/m is large, e.g., $n/m = \omega(\log m)$, all loads are $(1 \pm o(1))n/m$, w.h.p. However, with $n = m$, we still expect many bins with $\Theta((\log m)/(\log \log m))$ balls even though the average is 1.

7.2.3 Simple Consistent Hashing with Bounded Loads.

As we mentioned earlier, Mirrokni et al. [MTZ18] presented an algorithm that works with arbitrary capacity constraints on the bins. For the purpose of load balancing, the system designer can specify a balancing parameter $c = 1 + \varepsilon$, guaranteeing that the maximum load is at most $C = \lceil cn/m \rceil$.

Their idea is very simple. As in simple consistent hashing, we place balls and bins randomly on a cycle, but instead of placing balls in the first bin along the cycle, we place them in the first non-full bin. Thus we can think of the distribution as first placing all the bins on the cycle, and then placing the balls one-by-one, putting each in the first non-full bin found by going in clockwise around the cycle. If we have hash functions for placing arbitrary balls and bins along the cycle, and if we have a priority order on all balls, telling us the order in which we insert balls, then this completely determines the placement of any set of the balls in any set of capacitated bins. This means that the distribution is history independent as in [BG07]. It also means that we know exactly which balls to move if balls or bins are added or removed.

As terminology, we say a ball *hash to* the first bin following it in the clockwise order. However, the ball may be *placed* in a later bin if the bin it hashed to was full.

Note that the priority order makes the insertion of a new ball a bit more complicated since it may have higher priority than balls already in the system. To place it, we first place it in the bin it hashes to directly (that is, the one just after its hash location on the cycle). If the bin becomes overfull, we pop the lowest priority ball and place it in the next

bin, and repeat. It is, however, important to notice that the bins we end up considering are exactly the bins from the one the ball hashes to, and to the first non-full bin.

Mirrokni et al. [MTZ18] provided an analysis of their system. With $\varepsilon \leq 1$, they showed that starting from the hash location of any ball, the expected number of full bins passed on the way to the first non-full bin is $O(1/\varepsilon^2)$. From this they get that the expected number of balls that has to be moved when a ball is inserted or deleted is $O(1/\varepsilon^2)$. Likewise, the expected number of balls that has to be moved when a bin is inserted or deleted is $O(C/\varepsilon^2)$. These bounds are all tight for simple consistent hashing with bounded loads.

7.3 Consistent Hashing with Virtual Bins and Bounded Loads

The new algorithm of [AKT21] basically just combines the bounded loads with virtual bins. When a ball is placed in a virtual bin, it is also placed in its super bin which has a limited capacity. We fix some natural number k, which is the number of virtual bins for each super bin. In the following, we describe a mathematically simple version of the scheme in [AKT21]. In [AKT21], they also consider a variation of this scheme which is more suitable to be implemented in practice. The description of this scheme can be found in Appendix D.

We hash each super bin to k different cycles or levels using independent hash functions.[1] The k hash values on the k cycles will be the associated virtual bins of the given super bin. We also hash the balls to the cycles, but contrary to the bins, each ball gets just a single random hash value on a single random cycle.

The static placement of the balls can be described as follows: We start by placing all balls which hash to the first cycle using standard consistent hashing with bounded loads as described in Section 7.2.3. We assume that we have priorities on the balls and we will simulate that they are inserted in priority order. After the first level, the balls hashing to this level have thus been distributed into the virtual bins and we put them in the corresponding super bins. Initially, each super bin had capacity C. If the virtual bin of such a super bin received a balls at the first level, its new capacity is then reduced accordingly to $C - a$. We continue this process on level $i = 2, \ldots, k$. At level i, each super bin has a certain remaining capacity and we use standard consistent hashing with bounded loads (with these capacities) to place the balls at level i into the virtual bins and thus, into the corresponding super bins. If a super bin had capacity C_0 before the hashing to level i, and it received a balls at level i, its remaining capacity for the next levels is $C_0 - a$. Traversing the levels one at a time like described, corresponds to enforcing that regardless of the initial priorities of the balls, if two balls hash to different levels, the ball hashing to the lower level will have the highest priority of the two. With these modified priorities, the static image at a given point can be obtained by simply inserting the balls one by one in priority order, placing each ball in the first virtual bin whose super bin is not full. This

[1] For simplicity, we advice the reader to think of all our hash functions as fully random. However, our results hold even when the hashing is implemented with the practical mixed tabulation from [DKRT15].

completely describes the placement of balls in bins if we know the hash functions and the priority order, so the system is history-independent as described in [BG07].

Searching for a ball x is almost the same as for normal consistent hashing. We calculate the hash value of x and visit the virtual bins starting from that hash value in cyclic order until we either find x in a corresponding super bin or we meet a ball of lower priority hashing to the same level.

Insertions are a bit more complicated. For inserting a ball x we calculate $h(x)$ which in particular indicates the level, i, that x hashes to. We traverse level i starting at $h(x)$ until we meet a bin, b, which either (a) is not full or (b) contains a ball of lower priority than x (all balls hashing to levels $j > i$ have lower priority than x by convention). We insert x in b. In case (a), the insertion is complete, but in case (b) we pop y from b and recurse the insertion starting with y (which happens at some level $j \geq i$).

Ball deletions are symmetric to ball insertions in the sense that the hash functions tells us exactly the placement of all balls in bins, both before and after the ball which we are to insert or delete is inserted or deleted. Deleting a bin is the same as re-inserting all balls in it, and inserting a bin is symmetric to deleting a bin. Therefore we get that the number of balls to be moved is essentially determined by the number that has to be moved in connection with an insertion.

Main Result In classic consistent hashing without virtual bins, we obtain no advantage when the number of balls n are much larger than the number of bins m, or in other words, when the capacity of a bin, C, is large. The basic issue is that most of the uncertainty in the system without virtual bins stems from the uncertainty in the distance between a bin and its predecessor, which determines the expected number of balls hashing directly to the bin.

However, the use of virtual bins improves the concentration of the number of balls hashing directly to a super bin, and we do obtain an advantage of this improved concentration. This was in fact the whole point of introducing virtual bins in classic consistent hashing without load bounds [SMLK+03]. To be precise, fix $k = A(\log n)/\varepsilon^2$ for some appropriately large constant A. Then standard Chernoff bounds show that each bin cover a fraction $(1 \pm \lambda\varepsilon)/m$ of the combined hash range, where λ can be made arbitrarily small (by increasing A). If further the average load m/n is above k, then with high probability, no bin gets load above $C = (1 + \varepsilon)m/n$ by balls hashing directly to them. In particular, all load bounds are satisfied without the having to forward a single ball. The result below (which is the main result of [AKT21]) asymptotically settles the expected insertion time for general C, in particular for any $C \leq (\log n)/\varepsilon^2$. Before stating the theorem, we encourage the reader to recall the definition of f in eq. (7.1)

Theorem 7.2 ([AKT21])**.** *Let $0 < \varepsilon < 1$ and suppose that we distribute n balls into m bins each of capacity $C = (1 + \varepsilon)n/m$ using consistent hashing with bounded loads and $k = c/\varepsilon^2$ uniform levels for a sufficiently large constant c. Assume that $1/\varepsilon = n^{o(1)}$. In expectation we move $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin. Finally, when searching a ball, we expect to visit $O(1)$ bins when $C \geq \log 1/\varepsilon$ and $O(\frac{\log 1/\varepsilon}{C})$ bins when $C < \log 1/\varepsilon$.*

Our bounds in Theorem 7.2 show that we do get an advantage from bigger capacities even when C is smaller than $k = \Theta((\log n)/\varepsilon^2)$. In fact, already for $C = 1/\varepsilon^2$, the expected insertion time drops to $O(1)$.

7.4 Conclusion

In [AKT21], we have given a new solution to the problem of load balancing in a dynamic system where both balls and bins can be added and removed. Our new data structure combines the ideas of the previous work and its performance is comparable to the natural probabilistic problem of throwing balls into capacitated bins. We have also given a solution to this probabilistic problem.

Chapter 8

On Sums of Monotone Random Integer Variables

This chapter is dedicated to presenting the results of our research paper "On Sums of Monotone Random Integer Variables" [AAHT22] from Appendix E, and includes a slightly modified subset of its introduction.

8.1 Introduction

We study the problem of estimating probability that the sum of independent (not necessarily identically distributed) integer-valued random variables attains precisely a specific value. In [AKT21], we provide sharp estimates and our estimates hold under a fairly general assumption on the properties of the random variables, which for example is satisfied for Bernoulli, Poisson and geometric random variables. The bounds on the point probabilities derived in this paper have been used to understand the distribution of balls in capacitated bins [AKT21]. In the cleanest combinatorial variant of the problem, where the balls arrive sequentially and each ball picks a uniformly random non-full bin, they just needed the point probabilities of sums of i.i.d. Bernoulli variables. However, for a more dynamic distribution system, they had to apply the bounds for sums of a mix of Bernoulli and geometrically distributed variables.

Recall that for a real random variable X, the *characteristic function* of X is the map $f_X : \mathbb{R} \to \mathbb{C}$ given by $f_X(\lambda) = \mathrm{E}[e^{i\lambda X}]$. We say that a real random variable X is *monotone* if $|f_X|$ is decreasing on $[0, \pi]$. In [AKT21], we provide estimates for the point probabilities of a sum, $X = \sum_{j\in[k]} X_j$, of independent monotone random integer variables. To be precise, for any given $t \in \mathbb{Z}$, we estimate the probability $\Pr[X = t]$. The estimates of [AKT21] are sharp whenever t is close to the mean $\mathrm{E}[X]$, but they are not useful further out in the tail. To handle point probabilities in the tail, we require a slightly stronger assumption on the random variables which we now describe.

For a random integer variable X we define $I_X = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X}] < \infty\}$, to consist of those $\theta \in \mathbb{R}$ for which the moment generating function of X is defined. We note that I_X is an interval with $0 \in I_X$. For $\theta \in I_X$, we may define the *exponentially tilted* random

variable X_θ by $\Pr[X_\theta = t] = \frac{\Pr[X=t]e^{\theta t}}{\mathrm{E}[e^{\theta X}]}$ for $t \in \mathbb{Z}$. We say that X is *strongly monotone* if (1) $I_X \neq \{0\}$ and (2) X_θ is monotone for each $\theta \in I_X$. In [AKT21], we use the trick of exponential tilting to provide estimates for the point probabilities of a sum of independent strongly monotone random integer variables, $X = \sum_{j\in[k]} X_j$, which are also sharp in the tail.

It follows by direct computation that Bernoulli, Poisson, and geometric random variables are monotone, and moreover, that exponentially tilting these variables again yields Bernoulli, Poisson and geometric variables. In particular, these variables are all strongly monotone, so the results of [AKT21] give sharp estimates for the point probabilities of the sum of (a mix of) such variables.

We will consider the following setting. Let k be an integer and $(X_j)_{j\in[k]}$ independent integer-valued random variables with $\mathrm{E}[X_j] = \mu_j$ and $\mathrm{Var}[X_j] = \sigma_j^2$ for $j \in [k]$. Let $X = \sum_{i\in[k]} X_i$, and further $\mu = \sum_{j\in[k]} \mu_j$ and $\sigma^2 = \sum_{j\in[k]} \sigma_j^2$ be respectively the expectation and variance of X. The main result of [AKT21] is the following theorem.

Theorem 8.1 ([AKT21])**.** *There exists a universal constant c, such that if X is monotone, then for every t for which $\mu + t\sigma$ is an integer, the probability that X is precisely $\mu + t\sigma$ satisfies,*

$$\left|\Pr[X = \mu + t\sigma] - \frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2}\right| \leq c \left(\frac{\sum_{j\in[k]} \mathrm{E}\left[|X_j - \mu_j|^3\right]}{\sigma^3}\right)^2. \tag{8.1}$$

Remark 8.2. We note that if each X_j is monotone, then X is as well. Indeed, the characteristic function of X can be factorized as $f_X(\lambda) = \prod_{j\in[k]} f_{X_j}(\lambda)$. In particular, Theorem 8.1 holds when each of the variables $(X_j)_{j\in[k]}$ is monotone.

The result is reminiscent of the Berry-Esseen theorem, but instead of bounding the distance between the cumulative function of X and the cumulative function of the normal distribution as the Berry-Esseen theorem does, our result bounds the distance between the density function of X and the density function of the normal distribution. This setting has been studied before in the context of large deviation theory, e.g., by Blackwell and Hodges [BJ59] and by Iltis [Ilt95] in the d-dimensional case. They do not require X to be monotone but they only consider the case where $(X_j)_{j\in[k]}$ are identically distributed and are interested in the asymptotical behavior when $k \to \infty$. In particular the distribution of the variables $(X_j)_{j\in[k]}$ cannot depend on k. McDonald [McD79] considers variables that are not necessarily identically distributed but again in the limit $k \to \infty$ and with certain extra assumptions on the distribution of the variables. In this work we are not interested in such asymptotic bounds and our result is a uniform bound for monotone variables.

Point probabilites in the tail As is, Theorem 8.1 is only useful when $|t\sigma|$ is not too large. Indeed, for large $|t|$, the term $\frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2}$ will typically be much smaller than the error term on the right hand side of (8.1). In [AAHT22], we now show that if our variables satisfy the stronger property of being strongly monotone, we may also obtain

precise estimates for the point probabilities in the tail by combining with the trick of exponential tilting.

Now suppose $X = \sum_{j\in[k]} X_j$ is a sum of independent random integer variables and moreover that X is not almost surely equal to a constant. We are interested in estimates for the probability $\Pr[X = t]$ for some $t \in \mathbb{Z}$. Let $I_j = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X_j}] < \infty\}$ and $I = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X}] < \infty\} = \cap_{j\in[k]} I_j$. We note each I_j and I are intervals containing 0. We define[1] $A = \operatorname{ess\,inf} X$ and $B = \operatorname{ess\,sup} X$. Let further $\psi_X : I \to \mathbb{R}$ be the cumulant generating function defined by $\psi_X : \theta \mapsto \log(\mathrm{E}[e^{\theta X}])$. It is well known that ψ_X is strictly convex and infinitely often differentiable for θ lying in the interior of I with $\psi'_X(\theta) = \frac{\mathrm{E}[Xe^{\theta X}]}{\mathrm{E}[e^{\theta X}]}$. For $t \in \mathbb{R}$, we define $g(t) = \sup_{\theta\in I}(\theta t - \psi_X(\theta))$. Now it is a standard fact about the cumulant generating function that if I contains a non-empty open interval (i.e., consists of more than a single point), then $\inf_{\theta\in I} \psi'_X(\theta) = A$ and $\sup_{\theta\in I} \psi'_X(\theta) = B$. If in particular $A < t < B$, there exists a θ_0 in the interior of I with $\psi'_X(\theta_0) = t$. Moreover, this θ_0 is unique since ψ_X is strictly convex.

Now let $(Y_j)_{j\in[k]}$ be independent random variables obtained by tilting each X_j by θ_0 as above. Let further $Y = \sum_{j\in[k]} Y_j$. For $s \in \mathbb{Z}$, we define $A_s = \{z \in \mathbb{Z}^k : z_1 + \cdots + z_k = s\}$. Then for any $t \in \mathbb{Z}$,

$$\Pr[X = t] = \sum_{z\in A_t} \prod_{j\in[k]} \Pr[X_j = z_j] = \frac{\mathrm{E}[e^{\theta_0 X}]}{e^{\theta_0 t}} \sum_{z\in A_t} \prod_{j\in[k]} \Pr[Y_j = z_j] = \frac{\mathrm{E}[e^{\theta_0 X}]}{e^{\theta_0 t}} \Pr[Y = t],$$

so Y is simply the variable obtained by tilting X by θ_0. Moreover, by the choice of θ_0,

$$\mathrm{E}[Y] = \sum_{z\in\mathbb{Z}} \frac{\Pr[X = z]e^{\theta_0 z} z}{\mathrm{E}[e^{\theta_0 X}]} = \frac{\mathrm{E}[Xe^{\theta_0 X}]}{\mathrm{E}[e^{\theta_0 X}]} = \psi'_X(\theta_0) = t.$$

Now the fact that $\mathrm{E}[Y] = t$, suggests using Theorem 8.1 to estimate the probability that $\Pr[Y = t]$. Doing so, we immediately obtain the following result of [AAHT22].

Theorem 8.3 ([AAHT22])**.** *Assume that X is strongly monotone and not almost surely equal to a constant. Moreover assume that $I \neq \{0\}$. Let t be an integer with with $A < t < B$ and θ be the unique real in the interior of I having $\psi'_X(\theta) = t$. Let Y be the exponential tilt of X by θ. Then $\mathrm{E}[Y] = t$ and*

$$\Pr[X = t] = \frac{\mathrm{E}[e^{\theta X}]}{e^{\theta t}} \left(\frac{1}{\sqrt{2\pi}\sigma_Y} \pm O\left(\frac{\eta_Y^2}{\sigma_Y^6} \right) \right), \tag{8.2}$$

where $\sigma_Y^2 = \mathrm{Var}[Y]$ and $\eta_Y = \sum_{j\in[k]} \mathrm{E}[|Y_j - \mathrm{E}[Y_j]|^3]$.

Remark 8.4. We note that if either $A = \operatorname{ess\,inf} X \neq -\infty$ or $B = \operatorname{ess\,sup} X \neq \infty$, then $[0, \infty) \subset I$ or $(-\infty, 0] \subset I$, respectively, and we can therefore always apply the exponential

[1] Recall that the essential infimum and supremum of a random variable X are defined by $\operatorname{ess\,inf} X = \sup\{t : \Pr[X < t] = 0\}$ and $\operatorname{ess\,sup} X = \inf\{t : \Pr[X > t] = 0\}$ which are values in $\mathbb{R} \cup \{-\infty, \infty\}$.

tilt in the lemma. We moreover note that for $t < A$ and $t > B$, it trivially holds that $\Pr[X = t] = 0$ and it is an easy exercise to show that

$$\Pr[X = A] = \prod_{j\in[k]} \Pr[X_j = \operatorname{ess\,inf} X_j], \quad \text{and} \quad \Pr[X = B] = \prod_{j\in[k]} \Pr[X_j = \operatorname{ess\,sup} X_j],$$

whenever $A \neq -\infty$ and $B \neq \infty$. Even though the lemma does not provide estimates for these probabilities, they are therefore usually easy to determine for concrete families of random variables.

To apply Theorem 8.3, for $X = \sum_{j\in[k]} X_j$ a concrete sum of strongly monotone random variables, say geometric variables, we would calculate ψ_X and find the unique θ with $\psi'_X(\theta) = t$. We would then determine the tilted random variables $(Y_j)_{j\in[k]}$. Typically Y_j comes from the same family of random variables as X_j, e.g., an exponential tilt of respectively a Bernoulli, geometric, and Poisson variable is again Bernoulli, geometric and Poisson. We would then determine the quantities η_Y and σ^2_Y and plug into (8.2).

8.2 Conclusion

Bibliography

[AAHT22] Anders Aamand, Noga Alon, Jakob Bæk Tejs Houen, and Mikkel Thorup. "On sums of monotone random integer variables". In: *Electronic Communications in Probability* 27 (2022), pp. 1–8.

[ADKK+22] Anders Aamand, Debarati Das, Evangelos Kipouridis, Jakob Bæk Tejs Knudsen, Peter M. R. Rasmussen, and Mikkel Thorup. "No Repetition: Fast and Reliable Sampling with Highly Concentrated Hashing". In: *Proc. VLDB Endow.* 15.13 (2022), pp. 3989–4001.

[AKKR+20] Anders Aamand, Jakob Bæk Tejs Knudsen, Mathias Bæk Tejs Knudsen, Peter Michael Reichstein Rasmussen, and Mikkel Thorup. "Fast hashing with strong concentration bounds". In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. Ed. by Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy. ACM, 2020, pp. 1265–1278.

[AKT21] Anders Aamand, Jakob Bæk Tejs Knudsen, and Mikkel Thorup. "Load balancing with dynamic set of balls and bins". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 1262–1275.

[AKT18] Anders Aamand, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. "Power of d Choices with Simple Tabulation". In: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. Ed. by Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella. Vol. 107. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 5:1–5:14.

[AT19] Anders Aamand and Mikkel Thorup. "Non-empty Bins with Simple Tabulation Hashing". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*. Ed. by Timothy M. Chan. SIAM, 2019, pp. 2498–2512.

[ARW17] Amir Abboud, Aviad Rubinstein, and Ryan Williams. "Distributed PCP theorems for hardness of approximation in P". In: *2017 IEEE 58th An-*

nual Symposium on Foundations of Computer Science (FOCS). IEEE. 2017, pp. 25–36.

[Ach03] Dimitris Achlioptas. "Database-friendly random projections: Johnson-Lindenstrauss with binary coins". In: *Journal of Computer and System Sciences* 66.4 (2003). Special Issue on PODS 2001, pp. 671–687. ISSN: 0022-0000.

[AK20] Thomas D. Ahle and Jakob Bæk Tejs Knudsen. "Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search". In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. Ed. by Sandy Irani. IEEE, 2020, pp. 728–739.

[APRS16] Thomas Dybdahl Ahle, Rasmus Pagh, Ilya Razenshteyn, and Francesco Silvestri. "On the complexity of inner product similarity join". In: *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. ACM. 2016, pp. 151–164.

[AK17] Noga Alon and Bo'az Klartag. "Optimal Compression of Approximate Inner Products and Dimension Reduction". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. Oct. 2017, pp. 639–650.

[AINR14] Alexandr Andoni, Piotr Indyk, Huy L Nguyen, and Ilya Razenshteyn. "Beyond locality-sensitive hashing". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2014, pp. 1018–1028.

[ALRW17] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. "Optimal hashing-based time-space trade-offs for approximate near neighbors". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2017, pp. 47–66.

[AR15] Alexandr Andoni and Ilya Razenshteyn. "Optimal data-dependent hashing for approximate near neighbors". In: *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*. ACM. 2015, pp. 793–801.

[AR16] Alexandr Andoni and Ilya Razensteyn. "Tight Lower Bounds for Data-Dependent Locality-Sensitive Hashing". In: *32nd International Symposium on Computational Geometry (SoCG 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016.

[Ben62] George Bennett. "Probability Inequalities for the Sum of Independent Random Variables". In: *Journal of the American Statistical Association* 57.297 (1962), pp. 33–45. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1962.10482149.

[Ber24] Sergei Natanovich Bernstein. "On a modification of Chebyshev's inequality and of the error formula of Laplace". In: *Ann. Sci. Inst. Sav. Ukraine, Sect. Math.* 1 (1924), pp. 38–49.

[BJ59] David Blackwell and J. L. Hodges Jr. "The Probability in the Extreme Tail of a Convolution". In: *The Annals of Mathematical Statistics* 30.4 (1959), pp. 1113–1120.

[BG07] G. E. Blelloch and D. Golovin. "Strongly History-Independent Hashing with Applications". In: *Proc. 48th IEEE Symposium on Foundations of Computer Science (FOCS)*. 2007, pp. 272–282.

[BCLM+10] Vladimir Braverman, Kai-Min Chung, Zhenming Liu, Michael Mitzenmacher, and Rafail Ostrovsky. "AMS Without 4-Wise Independence on Product Domains". In: *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France*. Ed. by Jean-Yves Marion and Thomas Schwentick. Vol. 5. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010, pp. 119–130.

[BOR10] Vladimir Braverman, Rafail Ostrovsky, and Yuval Rabani. "Rademacher Chaos, Random Eulerian Graphs and The Sparse Johnson-Lindenstrauss Transform". In: *CoRR* abs/1011.2590 (2010). arXiv: 1011.2590.

[BSS00] André Brinkmann, Kay Salzwedel, and Christian Scheideler. "Efficient, distributed data placement strategies for storage area networks". In: *Proceedings of the Twelfth annual ACM Symposium on Parallel Algorithms and Architectures, SPAA*. 2000, pp. 119–128.

[CCF04] Moses Charikar, Kevin Chen, and Martin Farach-Colton. "Finding frequent items in data streams". In: *Theoretical Computer Science* 312.1 (2004). Automata, Languages and Programming, pp. 3–15. ISSN: 0304-3975.

[CW19] Lijie Chen and Ryan Williams. "An equivalence class for orthogonal vectors". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 21–40.

[Che52] Herman Chernoff. "A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the sum of Observations". In: *Annals of Mathematical Statistics* 23.4 (1952), pp. 493–507.

[CCT10] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. "A survey of binary similarity and distance measures". In: *Journal of Systemics, Cybernetics and Informatics* 8.1 (2010), pp. 43–48.

[CP17] Tobias Christiani and Rasmus Pagh. "Set similarity search beyond MinHash". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*. 2017, pp. 1094–1107.

[CPT15] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. "From Independence to Expansion and Back Again". In: *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*. Ed. by Rocco A. Servedio and Ronitt Rubinfeld. ACM, 2015, pp. 813–820.

[CPT18] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. "Confirmation Sampling for Exact Nearest Neighbor Search". In: *arXiv preprint arXiv:1812.02603* (2018).

[CK09] Edith Cohen and Haim Kaplan. "Leveraging discarded samples for tighter estimation of multiple-set aggregates". In: *ACM SIGMETRICS Performance Evaluation Review* 37.1 (2009), pp. 251–262.

[CJN18] Michael B. Cohen, T. S. Jayram, and Jelani Nelson. "Simple Analyses of the Sparse Johnson-Lindenstrauss Transform". In: *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*. Ed. by Raimund Seidel. Vol. 61. OASIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 15:1–15:9.

[DKRT15] S. Dahlgaard, M. B. T. Knudsen, E. Rotenberg, and M. Thorup. "Hashing for Statistics over K-Partitions". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. 2015, pp. 1292–1310.

[DKRT16] Søren Dahlgaard, Mathias Bæk Tejs Knudsen, Eva Rotenberg, and Mikkel Thorup. "The Power of Two Choices with Simple Tabulation". In: *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '16. Arlington, Virginia: Society for Industrial and Applied Mathematics, 2016, pp. 1631–1642. ISBN: 9781611974331.

[DKT17] Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. "Practical Hash Functions for Similarity Estimation and Dimensionality Reduction". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6618–6628. ISBN: 978-1-5108-6096-4.

[DKS10] Anirban Dasgupta, Ravi Kumar, and Tamás Sarlos. "A Sparse Johnson: Lindenstrauss Transform". In: STOC '10. Cambridge, Massachusetts, USA: Association for Computing Machinery, 2010, pp. 341–350. ISBN: 9781450300506.

[DR09] Martin Dietzfelbinger and Michael Rink. "Applications of a Splitting Trick". In: *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*. Ed. by Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas. Vol. 5555. Lecture Notes in Computer Science. Springer, 2009, pp. 354–365.

[DW03] Martin Dietzfelbinger and Philipp Woelfel. "Almost random graphs with simple hash functions". In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*. Ed. by Lawrence L. Larmore and Michel X. Goemans. ACM, 2003, pp. 629–638.

[Dum56] A. I. Dumey. "Indexing for rapid random access memory systems". In: *Computers and Automation* 5.12 (1956), pp. 6–9.

[GH05] George Giakkoupis and Vassos Hadzilacos. "A scheme for load balancing in heterogenous distributed hash tables". In: *Proceedings of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing, PODC*. 2005, pp. 302–311.

[Hou23] Jakob Bæk Tejs Houen. *Dimensionality Reduction using Practical Hash Functions*. 2023. Master's thesis.

[HT22] Jakob Bæk Tejs Houen and Mikkel Thorup. "Understanding the Moments of Tabulation Hashing via Chaoses". In: *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*. Ed. by Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff. Vol. 229. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 74:1–74:19.

[HT23] Jakob Bæk Tejs Houen and Mikkel Thorup. *A Sparse Johnson-Lindenstrauss Transform using Fast Hashing*. 2023. To appear in ICALP 2023.

[Ilt95] Michael Iltis. "Sharp asymptotics of large deviations in $\mathbb{R}^d$". In: *Journal of Theoretical Probability* 8.3 (1995), pp. 501–522.

[JW13] T. S. Jayram and David P. Woodruff. "Optimal Bounds for Johnson-Lindenstrauss Transforms and Streaming Problems with Subconstant Error". In: *ACM Trans. Algorithms* 9.3 (June 2013). ISSN: 1549-6325.

[JZYY+18] Lianyin Jia, Lulu Zhang, Guoxian Yu, Jinguo You, Jiaman Ding, and Mengjuan Li. "A Survey on Set Similarity Search and Join." In: *International Journal of Performability Engineering* 14.2 (2018).

[JL84] William Johnson and Joram Lindenstrauss. "Extensions of Lipschitz maps into a Hilbert space". In: *Contemporary Mathematics* 26 (Jan. 1984), pp. 189–206.

[KMN11] Daniel Kane, Raghu Meka, and Jelani Nelson. "Almost Optimal Explicit Johnson-Lindenstrauss Families". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Ed. by Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 628–639. ISBN: 978-3-642-22935-0.

[KN10] Daniel M. Kane and Jelani Nelson. *A Derandomized Sparse Johnson-Lindenstrauss Transform*. 2010.

[KN14] Daniel M. Kane and Jelani Nelson. "Sparser Johnson-Lindenstrauss Transforms". In: *J. ACM* 61.1 (Jan. 2014). ISSN: 0004-5411.

[KLLP+97] David R. Karger, Eric Lehman, Frank Thomson Leighton, Rina Panigrahy, Matthew S. Levine, and Daniel Lewin. "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web". In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, STOC*. 1997, pp. 654–663.

[KR06] David R. Karger and Matthias Ruhl. "Simple Efficient Load-Balancing Algorithms for Peer-to-Peer Systems". In: *Theory Comput. Syst.* 39.6 (2006). Announced at SPAA'05, pp. 787–804.

[KM05] Krishnaram Kenthapadi and Gurmeet Singh Manku. "Decentralized algorithms using both local and random probes for P2P load balancing". In: *SPAA 2005: Proceedings of the 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures*. 2005, pp. 135–144.

[Laa15] Thijs Laarhoven. "Tradeoffs for nearest neighbors on the sphere". In: *arXiv preprint arXiv:1511.07527* (2015).

[LN17] Kasper Green Larsen and Jelani Nelson. "Optimality of the Johnson-Lindenstrauss Lemma". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017, pp. 633–638.

[Man04] Gurmeet Singh Manku. "Balanced binary trees for ID management and load balance in distributed hash tables". In: *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing, PODC*. 2004, pp. 197–205.

[McD79] David McDonald. "A Local Limit Theorem for Large Deviations of Sums of Independent, Nonidentically Distributed Random Variables". In: *The Annals of Probability* 7.3 (1979), pp. 526–531. ISSN: 00911798.

[MTZ18] Vahab S. Mirrokni, Mikkel Thorup, and Morteza Zadimoghaddam. "Consistent Hashing with Bounded Loads". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*. Ed. by Artur Czumaj. SIAM, 2018, pp. 587–604.

[MNP06] Rajeev Motwani, Assaf Naor, and Rina Panigrahi. "Lower bounds on locality sensitive hashing". In: *Proceedings of the twenty-second annual symposium on Computational geometry*. ACM. 2006, pp. 154–157.

[NN13] Jelani Nelson and Huy L. NguyÅn. "Sparsity Lower Bounds for Dimensionality Reducing Maps". In: *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*. STOC '13. Palo Alto, California, USA: Association for Computing Machinery, 2013, pp. 101–110. ISBN: 9781450320290.

[ODo14] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[Ole03] Krzysztof Oleszkiewicz. "On a nonsymmetric version of the Khinchine-Kahane inequality". In: *Stochastic inequalities and applications*. Springer, 2003, pp. 157–168.

[ÖP03] Anna Östlin and Rasmus Pagh. "Uniform hashing in constant time and linear space". In: *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*. Ed. by Lawrence L. Larmore and Michel X. Goemans. ACM, 2003, pp. 622–628.

[PP08] Anna Pagh and Rasmus Pagh. "Uniform Hashing in Constant Time and Optimal Space". In: *SIAM J. Comput.* 38.1 (2008), pp. 85–96.

[Pan06] Rina Panigrahy. "Entropy based nearest neighbor search in high dimensions". In: *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics. 2006, pp. 1186–1195.

[PTW08] Rina Panigrahy, Kunal Talwar, and Udi Wieder. "A geometric approach to lower bounds for approximate near-neighbor search and partial match". In: *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2008, pp. 414–423.

[PTW10] Rina Panigrahy, Kunal Talwar, and Udi Wieder. "Lower bounds on near neighbor search via metric expansion". In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 805–814.

[PT13] Mihai Patrascu and Mikkel Thorup. "Twisted Tabulation Hashing". In: *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*. Ed. by Sanjeev Khanna. SIAM, 2013, pp. 209–228.

[PT12] Mihai Pătraşcu and Mikkel Thorup. "The Power of Simple Tabulation Hashing". In: *J. ACM* 59.3 (June 2012). ISSN: 0004-5411.

[PN95] Thomas K. Philips and Randolph Nelson. "The Moment Bound Is Tighter Than Chernoff's Bound for Positive Tail Probabilities". In: *The American Statistician* 49.2 (1995), pp. 175–178. ISSN: 00031305.

[Sie04] Alan Siegel. "On Universal Classes of Extremely Random Constant-Time Hash Functions". In: 33.3 (2004). Announced at FOCS'89, pp. 505–543.

[SMLK+03] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. "Chord: a scalable peer-to-peer lookup protocol for internet applications". In: *IEEE/ACM Trans. Netw.* 11.1 (2003), pp. 17–32.

[TR98] David Thaler and Chinya V. Ravishankar. "Using name-based mappings to increase hit rates". In: *IEEE/ACM Trans. Netw.* 6.1 (1998), pp. 1–14.

[Tho13] Mikkel Thorup. "Simple Tabulation, Fast Expanders, Double Tabulation, and High Independence". In: *54th Annual Symposium on Foundations of Computer Science (FOCS)*. 2013, pp. 90–99.

[TZ12] Mikkel Thorup and Yin Zhang. "Tabulation-Based 5-Independent Hashing with Applications to Linear Probing and Second Moment Estimation". In: *SIAM Journal on Computing* 41.2 (2012), pp. 293–331. eprint: https://doi.org/10.1137/100800774.

[WC81] Mark N. Wegman and Larry Carter. "New Classes and Applications of Hash Functions". In: *jcss* 22.3 (1981). Announced at FOCS'79, pp. 265–279.

[Wil05] Ryan Williams. "A new algorithm for optimal 2-constraint satisfaction and its implications". In: *Theoretical Computer Science* 348.2 (2005), pp. 357–365.

[Zob70] Albert Lindsey Zobrist. *A New Hashing Method with Application for Game Playing*. Tech. rep. 88. Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 1970.

Part II

Appendix

Appendix A

Understanding the Moments of Tabulation Hashing via Chaoses

Understanding the Moments of Tabulation Hashing via Chaoses

Jakob Bæk Tejs Houen*
University of Copenhagen
jakn@di.ku.dk

Mikkel Thorup*
University of Copenhagen
mthorup@di.ku.dk

Dansk resumé

Simple tabulation hashing dates back to Zobrist in 1970 and is defined as follows: Each key is viewed as c characters from some alphabet Σ, we have c fully random hash functions $h_0, \ldots, h_{c-1} \colon \Sigma \to \{0, \ldots, 2^l - 1\}$, and a key $x = (x_0, \ldots, x_{c-1})$ is hashed to $h(x) = h_0(x_0) \oplus \ldots \oplus h_{c-1}(x_{c-1})$ where $\oplus$ is the bitwise XOR operation. The previous results on tabulation hashing by Pătraşcu and Thorup [J.ACM'11] and by Aamand et al. [STOC'20] focused on proving Chernoff-style tail bounds on hash-based sums, e.g., the number keys hashing to a given value, for simple tabulation hashing, but their bounds do not cover the entire tail. Thus their results cannot bound moments. The paper Dahlgaard et al. [FOCS'15] provides a bound on the moments of certain hash-based sums, but their bound only holds for constant moments, and we need logarithmic moments.

Chaoses are random variables of the form $\sum a_{i_0,\ldots,i_{c-1}} X_{i_0} \cdot \ldots \cdot X_{i_{c-1}}$ where X_i are independent random variables. Chaoses are a well-studied concept from probability theory, and tight analysis has been proven in several instances, e.g., when the independent random variables are standard Gaussian variables and when the independent random variables have logarithmically convex tails. We notice that hash-based sums of simple tabulation hashing can be seen as a sum of chaoses that are not independent. This motivates us to use techniques from the theory of chaoses to analyze hash-based sums of simple tabulation hashing.

In this paper, we obtain bounds for all the moments of hash-based sums for simple tabulation hashing which are tight up to constants depending only on c. In contrast with the previous attempts, our approach will mostly be analytical and does not employ intricate combinatorial arguments. The improved analysis of simple tabulation hashing allows us to obtain bounds for the moments of hash-based sums for the mixed tabulation hashing introduced by Dahlgaard et al. [FOCS'15]. With simple tabulation hashing, there are certain inputs for which the concentration is much worse than with fully random hashing. However, with mixed tabulation, we get logarithmic moment bounds that are only a constant factor worse than those with fully random hashing for any possible input. This is a strong addition to other powerful probabilistic properties of mixed tabulation hashing proved by Dahlgaard et al.

*Research supported by Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

Contents

A.1 Introduction

Hashing is a ubiquitous tool of randomized algorithms which dates all the way back to the 1950s [Dum56]. A hash function is a random function, $h\colon U \to R$, that assigns a random hash value, $h(x) \in R$, to every key, $x \in U$. When designing algorithms and data structures, it is often assumed that one has access to a uniformly random hash function that can be evaluated in constant time. Even though this assumption is very useful and convenient, it is unfortunately also unrealistic. It is thus a natural goal to find practical and efficient constructions of hash functions that provably have guarantees akin to those of uniformly random hashing.

If we want implementable algorithms with provable performance similar to that proven assuming uniformly random hashing, then we have to find practical and efficient constructions of hash functions with guarantees akin to those of uniformly random hashing. An example of this is simple tabulation hashing introduced by Zobrist in 1970 [Zob70]. The scheme is efficient and easy to implement, and Pătraşcu and Thorup [PT12] proved that it

could replace uniformly random hashing in many algorithmic contexts. The versatility of simple tabulation does not stem from a single probabilistic power like k-independence (it is only 3-independent), but from an array of powers that have different usages in different applications. Having one hash function with multiple powers has many advantages. One is that we can use the same hash function implementation for many purposes. Another is that hash functions are often an inner-loop bottleneck, and then it is an advantage if the same hash value can be used for multiple purposes. Also, if we have proved that a simple hash function has some very different probabilistic properties, then, morally, we would expect it to possess many other properties to be uncovered as it has happened over the years for simple tabulation (see, e.g., [AKT18; AT19]). Finally, when we hash a key, we may not even know what property is needed, e.g., with weighted keys, we may need one property to deal with a few heavy keys, and another property to deal with the many light keys, but when we hash the key, we may not know if it is heavy or light.

One of the central powers proved for simple tabulation in [PT12] is that it has strong concentration bounds for hash-based sums (will be defined shortly in Section A.1.1). The concentration holds only for quite limited expected values, yet this suffices for important applications in classic hash tables. Recently, Aamand et al. [AKKR+20] introduced tabulation-permutation, which is only about twice as slow as simple tabulation, and which offers general concentration bounds that hold for all hash-based sums regardless of the expected size. An issue with tabulation-permutation is that it is not clear if it possesses the other strong powers of simple tabulation.

A different way to go is to construct increasingly strong schemes, each inheriting all the nice properties of its predecessors. In this direction, [PT13] introduced twisted tabulation strengtening simple tabulation, and [DKRT15] introduced mixed tabulation strengthening twisted tabulation. Each new scheme was introduced to get some powers not available with the predecessor. In particular, mixed tabulation has some selective full-randomness that is needed for aggregating statistics over hash-based k-partitions. These applications also needed concentration bounds for hash-based sums, but [DKRT15] only provided some specialized suboptimal concentration bounds.

In this paper, we do provide strong concentration bounds for mixed tabulation hashing which can then be used in tandem with all the other strong properties of simple, twisted, and mixed tabulation. In fact our bounds are more general than the strong concentration bounds proved in [AKKR+20] for tabulation-permutation. More precisely, the concentration bounds in [AKKR+20] are Chernoff-style tail bounds that hold with high probability, while what we do is to show moment bounds that imply such tail bounds as special cases. Indeed the key to our results for mixed tabulation is a much stronger understanding of the moments of simple tabulation.

Below we proceed to describe our new mathematical understanding, including the relevance of chaoses. We will contextualize this with other work later in Section A.1.6.

A.1.1 Moment bounds for hash-based sums

In this paper, we will focus on analyzing hash-based sums. More precisely, we consider a fixed *value function*, $v\colon U \times R \to \mathbb{R}$, and define the random variable $X_x = v(x, h(x))$

for every key $x \in U$. We are then interested in proving concentration bounds for the sum $X = \sum_{x \in U} X_x = \sum_{x \in U} v(x, h(x))$. It should be noted that the randomness of X derives from the hash function h, thus the results will depend on the strength of h.

This is quite a general problem, and at first glance, it might not be obvious why this is a natural construction to consider, but it does generalize a variety of well-studied constructions:

1. Let $S \subseteq U$ be a set of balls and assign a weight, $w_x \in \mathbb{R}$, for every ball, $x \in S$. The goal is to distribute the balls, S, into a set of bins $R = [m]$.[1] For a bin, $y \in [m]$, we define the value function $v_y \colon U \times [m] \to \mathbb{R}$ by $v_y(x, j) = w_x [j = y] [x \in S]$, then $X = \sum_{x \in U} v_y(x, h(x)) = \sum_{x \in S} w_x [h(x) = y]$ will be the weight of the balls hashing to bin y.[2]

2. Instead of concentrating on a single bin, we might be interested in the total weight of the balls hashing below some threshold l. This is useful for sampling, for if $h(x)$ is uniform in $[m]$, then $\Pr[h(x) < l] = l/m$. We then define the value function $v \colon U \times [m] \to \mathbb{R}$ by $v(x, j) = w_x [j < l] [x \in S]$, then $X = \sum_{x \in U} v(x, h(x)) = \sum_{x \in S} w_x [h(x) < l]$ will be precisely the total weight of the balls hashing below l.

The first case appears when one tries to allocate resources, and the second case arises in streaming algorithms, see, e.g., [ADKK+20]. In any case, X ought to be concentrated around the mean $\mu = \mathrm{E}[X]$. If h is a uniformly random hash function then this will be the case under mild assumptions about v but it cannot otherwise be assumed a priori to be the case.

There are two natural ways to quantify the concentration of X, either we bound the tail of X, i.e., we bound $\Pr[|X - \mu| \geq t]$ for all $t \geq 0$, or we bound the central moments of X, i.e., we bound the p-th moment $\mathrm{E}[|X - \mu|^p]$ for all $p \geq 2$. If we have a bound on the tail that is exponentially decreasing, we can bound the central moments of X for all $p \geq 2$. Unfortunately, some of the prior works [AKKR+20; DR09; Tho13] prove bounds on the tail that are exponentially decreasing but also has an additive term of the form $n^{-\gamma}$ where $\gamma = O(1)$. It will then only be possible to give strong bounds for the central moments of X for $p = O(1)$. This is not necessarily a fault of the hash function but a defect of the analysis. In contrast, if we prove strong bounds for the central moments of X for $p = O(\log n)$ then we can use Markov's inequality to prove a bound the tail that is exponentially decreasing but with an additive term of the form $n^{-\gamma}$ where $\gamma = O(1)$. Thus in some sense, it is more robust to bound the moments compared to bounding the tail.

We can use the classic k-independent hashing framework of Wegman and Carter [WC81] as an easy way to obtain a hash function that has bounds on the central moments as a uniformly random hash function. A random hash function, $h \colon U \to R$, is k-independent if $(h(x_0), \ldots, h(x_{k-1}))$ is uniformly distributed in R^k for any k distinct keys $x_0, \ldots, x_{k-1} \in U$. The p-th central moment $\mathrm{E}[(X - \mu)^p]$ of X for a k-independent

[1] For a positive integer $m \in \mathbb{N}$ we define $[m] = \{0, \ldots, m - 1\}$.
[2] For a statement P we let $[P]$ be 1 if P is true and 0 otherwise.

hash function h is the same as the p-th central moment of X for a fully random hash function when p is an even integer less than k.

A.1.2 Tabulation Hashing

Simple tabulation hashing dates back to 1970 and was first introduced by Zobrist for optimizing chess computers [Zob70]. In simple tabulation hashing, we view the universe, U, to be of the form $U = \Sigma^c$ for some alphabet, Σ, and a positive integer c. Let $T\colon \{0,\dots,c-1\} \times \Sigma \to [2^l]$ be a uniformly random table, i.e., each value is chosen independently and uniformly at random from the set $[2^l]$. A simple tabulation hash function, $h\colon \Sigma^c \to [2^l]$, is then defined by

$$h(\alpha_0,\dots,\alpha_{c-1}) = \bigoplus_{i=0}^{c-1} T(i,\alpha_i) \ ,$$

where $\oplus$ is the bitwise XOR-operation, i.e., addition when $[2^l]$ is identified with the Abelian group $(\mathbb{Z}/2\mathbb{Z})^l$. We say that h is a simple tabulation hash function with c characters. With 8- or 16-bit characters, the random table T fits in cache, and then simple tabulation is very fast, e.g., in experiments, [PT12] found it to be as fast as two to three multiplications.

The moments of simple tabulation hashing have been studied in multiple papers. Braverman et al. [BCLM+10] showed that for a fixed bin the 4th central moment is close to that achieved by truly random hashing. Dahlgaard et al. [DKT17] generalized this to any constant moment p. Their proof works for any p but with a doubly exponential dependence on p, so their bound is only useful for $p = O(1)$. In this paper, we obtain bounds for all the moments of hash-based sums for simple tabulation hashing which are tight up to constants depending only on c.

Previous work has just treated c as a constant, hidden in O-notation. However, c does provide a fundamental trade-off between evaluation time with c lookups and the space $cU^{1/c}$. We therefore find it relevant to elucidate how our moment bounds depend on c even though we typically choose $c = 4$.

Mixed tabulation hashing was introduced by Dahlgaard et al. [DKRT15]. As in simple tabulation hashing, we view the universe, U, to be of the form $U = \Sigma^c$ for some alphabet, Σ, and a positive integer c. We further assume that the alphabet, Σ, has the form $\Sigma = [2^k]$. Let $h_1\colon \Sigma^c \to [2^l]$, $h_2\colon \Sigma^c \to \Sigma^d$, and $h_3\colon \Sigma^d \to [2^l]$ be independent simple tabulation hash functions. A mixed tabulation hash function, $h\colon \Sigma^c \to [2^l]$, is then defined by

$$h(x) = h_1(x) \oplus h_3(h_2(x)) \ .$$

As in simple tabulation hashing, $\oplus$ is the bitwise XOR-operation. We call h a mixed tabulation hash function with c characters and d derived characters. We note that h_1 and h_2 can be combined in a single simple tabulation hash function $\Sigma^c \to [2^l] \times \Sigma^d$, and then h is implemented with only $c + d$ lookups.

With simple tabulation hashing, there are certain inputs for which the concentration is much worse than with fully random hashing. However, with mixed tabulation, even if we have just $d = 1$ derived character, we get logarithmic moment bounds that, for

$c = O(1)$, are only a constant factor worse than those with fully-random hashing for any input assuming that hash range at most polynomial in the key universe.

Getting within a constant factor is very convenient within algorithm analysis, where we typically only aim for O-bounds that are tight within a constant factor.

A.1.3 Relation between Simple Tabulation and Chaoses

A *chaos* of order c is a random variable of the form

$$\sum_{0\leq i_0<\ldots<i_{c-1}<n} a_{i_0,\ldots,i_{c-1}} \prod_{j\in[c]} X_{i_j} \ ,$$

where $(X_i)_{i\in[n]}$ are independent random variables and $(a_{i_0,\ldots,i_{c-1}})_{0\leq i_0<\ldots<i_{c-1}<n}$ is a multiindexed array of real numbers. And a *decoupled chaos* of order c is a random variable of the form

$$\sum_{i_0,\ldots,i_{c-1}\in[n]} a_{i_0,\ldots,i_{c-1}} \prod_{j\in[c]} X_{i_j}^{(j)} \ ,$$

where $(X_i^{(j)})_{i\in[n],j\in[c]}$ are independent random variables and $(a_{i_0,\ldots,i_{c-1}})_{i_0,\ldots,i_{c-1}\in[n]}$ is a multiindexed array of real numbers. Chaoses have been studied in different settings, e.g., when the variables are standard Gaussian variables [Lat06; Leh11], when the variables have logarithmically concave tails [AL12], and when the variables have logarithmically convex tails [KL15].

From the definition of a chaos and simple tabulation hashing it might not be immediately clear that there is connection between the two. But we can rewrite the expression for hash-based sums of simple tabulation hashing as follows

$$\begin{aligned}\sum_{x\in\Sigma^c} v(x,h(x)) &= \sum_{\alpha_0,\ldots,\alpha_{c-1}\in\Sigma} v((\alpha_0,\ldots,\alpha_{c-1}),h(\alpha_0,\ldots,\alpha_{c-1})) \\ &= \sum_{j_0,\ldots,j_{c-1}\in[m]} \sum_{\alpha_0,\ldots,\alpha_{c-1}\in\Sigma} v\left((\alpha_0,\ldots,\alpha_{c-1}),\bigoplus_{i\in[c]} j_i\right) \prod_{i\in[c]} [T(i,\alpha_i)=j_i] \ .\end{aligned}$$

We then notice that $\sum_{\alpha_0,\ldots,\alpha_{c-1}\in\Sigma} v\left((\alpha_0,\ldots,\alpha_{c-1}),\bigoplus_{i\in[c]} j_i\right)\prod_{i\in[c]}[T(i,\alpha_i)=j_i]$ is a decoupled chaos of order c for any $(j_i)_{i\in[c]}$, thus hash-based sums of simple tabulation hashing can be seen as a sum of chaoses. Now since the random variables, $([T(i,\alpha_i)=j])_{j\in[m]}$, are not independent then the chaoses are not independent either which complicates the analysis. Nonetheless, this realization inspires us to use techniques from the study of chaoses to analyze the moments of tabulation hashing, in particular, our approach will be analytical in contrast with the combinatorial approach of the previous papers. We will expand further on the techniques in Appendix A.1.5.

A.1.4 Our Results

When proving and stating bounds for the p-th moment of a random variable it is often more convenient and more instructive to do it in terms of the p-norm of the random

variable. The p-norm of a random variable is the p-th root of the p-th moment of the random variable and is formally defined as follows:

Definition A.1 (p-norm)**.** Let $p \geq 1$ and X be a random variable with $\mathrm{E}[|X|^p] < \infty$. We then define the p-norm of X by $\|X\|_p = \mathrm{E}[|X|^p]^{1/p}$.

Our main contributions of this paper are analyses of the moments of hash-based sums of simple tabulation hashing and mixed tabulation hashing. To do this we first had to analyze the moments of hash-based sums of fully random hashing which as far as we are aware have not been analyzed tightly before.

The Moments of Fully Random Hashing

Previously, the focus has been on proving Chernoff-like bounds by using the moment generating function but a natural, different approach would be to use moments instead. Both the Chernoff bounds [Che52] and the more general Bennett's inequality [Ben62] bound the tail using the Poisson distribution. More precisely, let $v\colon U\times[m] \to \mathbb{R}$ be a value function that satisfies that $\sum_{j\in[m]} v(x,j) = 0$ and define the following two parameters M_v and σ_v^2 which will be important throughout the paper as follows:

$$M_v = \max_{x\in U, j\in[m]} |v(x,j)| \ , \tag{A.1}$$

$$\sigma_v^2 = \frac{\sum_{x\in U, j\in[m]} v(x,j)^2}{m} \ . \tag{A.2}$$

Bennett's inequality specialized to our setting then says that for a fully random hash function h

$$\begin{aligned} \Pr\left[\left|\sum_{x\in U} v(x,h(x))\right| \geq t\right] &\leq 2\exp\left(-\tfrac{\sigma_v^2}{M_v^2}\mathcal{C}\left(\tfrac{tM_v}{\sigma_v^2}\right)\right) \\ &\leq \begin{cases} 2\exp\left(-\frac{t^2}{3\sigma_v^2}\right) & \text{if } t \leq \frac{\sigma_v^2}{M_v} \\ 2\exp\left(-\frac{t}{2M_v}\log\left(1+\frac{tM_v}{\sigma_v^2}\right)\right) & \text{if } t > \frac{\sigma_v^2}{M_v} \end{cases} , \end{aligned} \tag{A.3}$$

where $\mathcal{C}(x) = (x+1)\log(x+1) - x$.[3]

This inspires us to try to bound the p-norms of X_v with the p-norms of the Poisson distribution. To do this we will introduce the function $\Psi_p(M,\sigma^2)$ which is quite technical but we will prove that $\Psi_p(1,\lambda)$ is equal up to a constant factor to the central p-norm of a Poisson distributed variable with mean λ. One should think of $\Psi_p(M,\sigma^2)$ as a p-norm version of $\frac{\sigma_v^2}{M_v^2}\mathcal{C}\left(\frac{tM_v}{\sigma_v^2}\right)$ which appears in Bennett's inequality.

[3]Here and throughout the paper $\log(x)$ will refer to the natural logarithm.

Definition A.2. For $p \geq 2$ we define the function $\Psi_p \colon \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ as follows

$$\Psi_p(M, \sigma^2) = \begin{cases} \left(\frac{\sigma^2}{pM^2}\right)^{1/p} M & \text{if } p < \log \frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{p}\sigma & \text{if } p < e^2 \frac{\sigma^2}{M^2} \\ \frac{p}{e \log \frac{pM^2}{\sigma^2}} M & \text{if } \max\left\{\log \frac{pM^2}{\sigma^2}, e^2 \frac{\sigma^2}{M^2}\right\} \leq p \end{cases} .$$

Remark A.3. When p is *small* then case 1 and 2 apply while for *large* p case 3 applies. If $2 < e^2 \frac{\sigma^2}{M^2}$ then we always have that $p > \log \frac{pM^2}{\sigma^2}$ for $2 \leq p$, hence only case 2 and 3 apply. Similarly, if $e^2 \frac{\sigma^2}{M^2} \leq 2$ then $p \geq e^2 \frac{\sigma^2}{M^2}$ for all $2 \leq p$, hence only case 1 and 3 apply. This shows that the cases disjoint and cover all parameter configurations.

The definition $\Psi_p(M, \sigma^2)$ might appear strange but it does in fact capture the central p-norms of Poisson distributed random variables. This is stated more formally in the following lemma.

Lemma A.4. *There exist universal constants K_1 and K_2 satisfying that for a Poisson distributed random variable, X, with $\lambda = \mathrm{E}[X]$*

$$K_2 \Psi_p(1, \lambda) \leq \|X - \lambda\|_p \leq K_1 \Psi_p(1, \lambda) ,$$

for all $p \geq 2$.

Bennett's inequality shows that we can bound the tail of $\sum_{x \in U} v(x, h(x))$ and Lemma A.4 shows that $\Psi_p(M, \sigma^2)$ captures the central p-norms of the Poisson distribution. It is therefore not so surprising that we are to bound the p-norms of $\sum_{x \in U} v(x, h(x))$ using $\Psi_p(M, \sigma^2)$.

Theorem A.5. *Let $h \colon U \to [m]$ be a uniformly random function, let $v \colon U \times [m] \to \mathbb{R}$ be a fixed value function, and assume that $\sum_{j \in [m]} v(x, j) = 0$ for all keys $x \in U$. Define the random variable $X_v = \sum_{x \in U} v(x, h(x))$. Then for all $p \geq 2$*

$$\|X_v\|_p \leq L \Psi_p\left(M_v, \sigma_v^2\right) ,$$

where $L \leq 16e$ is a universal constant.

To get a further intuition for $\Psi_p(M, \sigma^2)$ is is instructive to apply Markov's inequality and compare the tail bound to Bennett's inequality. More precisely, assume that $\|Y - \mathrm{E}[Y]\|_p \leq L \Psi_p(M, \sigma^2)$ for a constant L and for all $p \geq 2$. Then we can use Markov's inequality to get the following tail bound for all $t > 0$

$$\begin{aligned} \Pr\Big[\Big|Y - \mathrm{E}[Y]\Big| \geq t\Big] &\leq \left(\frac{\|Y - \mathrm{E}[Y]\|_p}{t}\right)^p \\ &\leq \begin{cases} \frac{L^2\sigma^2}{2t^2} & \text{if } t \leq L \max\left\{M, \frac{e\sigma}{\sqrt{2}}\right\} \\ \exp\left(-\frac{4t^2}{e^2 L^2 \sigma^2}\right) & \text{if } L\frac{e\sigma}{\sqrt{2}} \leq t \leq L\frac{e^2\sigma^2}{2M} \\ \exp\left(-\frac{t}{LM} \log\left(\frac{2tM}{L\sigma^2}\right)\right) & \text{if } L \max\left\{\frac{e^2\sigma^2}{2M}, M\right\} \leq t \end{cases} . \end{aligned} \tag{A.4}$$

In order to obtain these bounds p is chosen as follows: If $t \leq \max\left\{M, \frac{e\sigma}{\sqrt{2}}\right\}$ then $p = 2$ and otherwise p is chosen such that $\|Y - \mathrm{E}[Y]\|_p \leq e^{-1}t$. More precisely, we have that

$$p = \begin{cases} 2 & \text{if } t \leq L\max\left\{M, \frac{e\sigma}{\sqrt{2}}\right\} \\ \frac{4t^2}{e^2L^2\sigma^2} & \text{if } L\frac{e\sigma}{\sqrt{2}} \leq t \leq L\frac{e^2\sigma^2}{2M} \\ \frac{t}{LM}\log\left(\frac{2tM}{L\sigma^2}\right) & \text{if } L\max\left\{\frac{e^2\sigma^2}{2M}, M\right\} \leq t \end{cases}.$$

We see that eq. (A.4) gives the same tail bound as Bennett's inequality, eq. (A.3), up to a constant in the exponent.

We also prove a matching lower bound to Theorem A.5 which shows that $\Psi_p(M, \sigma^2)$ is the correct function to consider.

Theorem A.6. *Let $h\colon U \to [m]$ be a uniformly random function, then there exists a value function, $v\colon U \times [m] \to \mathbb{R}$, where $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in U$, such that the random variable $X_v = \sum_{x\in U} v(x, h(x))$ satisfies that for all $p \leq L_1 |U| \log(m)$*

$$\left\|\sum_{x\in U} v(x,h(x))\right\|_p \geq L_2\Psi_p\left(M_v, \sigma_v^2\right),$$

where L_1 and L_2 are a universal constant.

The Moments of Tabulation Hashing

We analyze the p-norms of hash-based sums for simple tabulation hashing, and our analysis is the first that provides useful bounds for non-constant moments. Furthermore, it is also the first analysis of simple tabulation hashing that does not assume that c is constant. We obtain an essentially tight understanding of this problem and show that simple tabulation hashing only works well when the range is large. This was also noted by Aamand et al. [AKKR+20] and they solve this deficiency of simple tabulation hashing by introducing a new hashing scheme, tabulation-permutation hashing. We show that it is also possible to break the bad instances of simple tabulation hashing by using mixed tabulation hashing.

We introduce a bit of notation to make the theorems cleaner. We will view a value function $v\colon \Sigma^c \times [m] \to \mathbb{R}$ as a vector, more precisely, we let

$$\|v\|_q = \left(\sum_{x\in\Sigma^c}\sum_{j\in[m]} |v(x,j)|^q\right)^{1/q}$$

for all $q \in [1,\infty]$. For every key $x \in \Sigma^c$ we define $v[x]$ to be the sub-vector v restricted to x, more precisely, we let

$$\|v[x]\|_q = \left(\sum_{j\in[m]} |v(x,j)|^q\right)^{1/q}$$

for all $q \in [1,\infty]$.

Simple Tabulation Hashing. Our main result for simple tabulation hashing is a version of Theorem A.5.

Theorem A.7. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{simple}} = \sum_{x\in\Sigma^c} v(x,h(x))$. Then for all $p \geq 2$*

$$\left\|V_v^{\text{simple}}\right\|_p \leq L_1 \Psi_p\left(K_c \gamma_p^{c-1} M_v, K_c \gamma_p^{c-1} \sigma_v^2\right) ,$$

where $K_c = (L_2 c)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}\right)/c, p\right\}}{\log\left(e^2 m \left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)}$$

It is instructive to compare this result to Theorem A.5 for fully random hashing. Ignoring the constant K_c, the result for simple tabulation hashing corresponds to the result for fully random hashing if we group keys into groups of size γ_p^{c-1}.

The definition of γ_p is somewhat complicated because of the generality of the theorem, but we will try to explain the intuition behind it. The expression $\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}$ measures how spread out the mass of the value function is. It was also noted in the previous analysis by Aamand et al. [AKKR+20] that this measure is naturally occurring. In fact, their result needs that $\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} \leq m^{1/4}$. If we consider the example from the introduction of hashing below a threshold $l \leq m$ where each key, $x \in \Sigma^c$, has weight w_x, then the value function, v, will be $v(x,j) = w_x\left([j<l] - \frac{l}{m}\right)$ for $x \in \Sigma^c, j \in [m]$, and we then get that

$$\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} = 4l\left(1-\frac{l}{m}\right) \leq 4l .$$

This correctly measures that the mass of the value function is mostly concentrated to the l positions of $[m]$.

The expression $\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}$ is a measure for how many keys that have significant weight. This also showed up in the previous analyses of simple tabulation hashing [AKKR+20; PT12]. If we again consider the example from before, we get that

$$\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2} = \frac{\sum_{x\in\Sigma^c} w_x^2}{\max_{x\in\Sigma^c} w_x^2} .$$

We can summarize the example in the following corollary.

Corollary A.8. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, assign a weight, $w_x \in \mathbb{R}$, to every key, $x \in \Sigma^c$, and consider a threshold $l \leq m$. Define the random variable*

$V_v^{\text{simple}} = \sum_{x\in\Sigma^c} w_x \left([h(x) < l] - \frac{l}{m}\right)$. *Then for all* $p \geq 2$

$$\left\|V_v^{\text{simple}}\right\|_p \leq \Psi_p\left(K_c\gamma_p^{c-1}\max_{x\in\Sigma^c}|w_x|, K_c\gamma_p^{c-1}\left(\sum_{x\in\Sigma^c} w_x^2\right)\frac{l}{m}\left(1-\frac{l}{m}\right)\right) ,$$

where $K_c = L_1\left(L_2 c\right)^{c-1}$, L_1 *and* L_2 *are universal constants, and*

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c} w_x^2}{\max_{x\in\Sigma^c} w_x^2}\right)/c, p\right\}}{\log\left(\frac{e^2 m}{4l}\right)}$$

A natural question is how close Theorem A.7 is to being tight. We show that if $\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}\right)/c = O\left(\log\left(1 + m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)\right)$ then the result is tight up to a universal constant depending only c. Formally, we prove the following lemma.

Theorem A.9. *Let* $h\colon \Sigma^c \to [m]$ *be a simple tabulation hash function, and* $2 \leq p \leq L_1|\Sigma|\log(m)$*, then there exists a value function,* $v\colon U\times[m]\to\mathbb{R}$*, where* $\sum_{j\in[m]} v(x,j) = 0$ *for all keys* $x \in \Sigma^c$*, and for which*

$$\left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \geq K_c'\Psi_p\left(\gamma_p^{c-1}M_v, \gamma_p^{c-1}\sigma_v^2\right) ,$$

where $K_c' = L_1^c$ *and* L_1 *is a universal constant, and*

$$\gamma_p = \max\left\{1, \frac{p}{\log\left(e^2 m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)}\right\}$$

Mixed Tabulation Hashing. The results of simple tabulation hashing work well when the range is large and when the mass of the value function is on few coordinates. We show that mixed tabulation hashing works well even if the range is small.

Theorem A.10. *Let* $h\colon \Sigma^c \to [m]$ *be a mixed tabulation function with* $d \geq 1$ *derived characters,* $v\colon \Sigma^c\times[m]\to\mathbb{R}$ *a value function, and assume that* $\sum_{j\in[m]} v(x,j) = 0$ *for all keys* $x \in \Sigma^c$*. Define the random variable* $V_v^{\text{mixed}} = \sum_{x\in\Sigma^c} v(x,h(x))$*. For all* $p \geq 2$ *then*

$$\left\|V_v^{\text{mixed}}\right\|_p \leq \Psi_p\left(K_c\gamma_p^c M_v, K_c\gamma_p^c\sigma_v^2\right) \tag{A.5}$$

where $K_c = L_1\left(L_2 c\right)^c$, L_1 *and* L_2 *are universal constants, and*

$$\gamma_p = \max\left\{1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)}\right\} .$$

Usually, in hashing contexts, we do not map to a much larger domain, i.e., we will usually have that $m \leq |U|^\gamma$ for some constant $\gamma \geq 1$. If this is the case then we can obtain the following nice tail bound for mixed tabulation hashing by using Markov's inequality.

Corollary A.11. *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with $d \geq 1$ derived characters, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{mixed}} = \sum_{x\in\Sigma^c} v(x,h(x))$. If $m \leq |U|^\gamma$ for a value $\gamma \geq 1$ then for all $t \geq 0$*

$$\Pr\left[\left|V_v^{\text{mixed}}\right| \geq t\right] \leq \exp\left(-\frac{\sigma_v^2}{M_v^2}\mathcal{C}\left(\frac{tM_v}{\sigma_v^2}\right)/K_{c,\gamma}\right) + |U|^{-\gamma} \ ,$$

where $\mathcal{C}(x) = (x+1)\log(x+1) - x$, $K_{c,\gamma} = L_1\left(L_2c^2\gamma\right)^c$, and L_1 and L_2 are universal constants.

Proof. The idea is to combine Theorem A.10 and Markov's inequality. We use Theorem A.10 for $2 \leq p \leq \gamma\log|U|$ to get that

$$\left\|V_v^{\text{mixed}}\right\|_p \leq \Psi_p\left(K_c\gamma_p^c M_v, K_c\gamma_p^c\sigma_v^2\right) \ ,$$

where we can bound γ_p by

$$\gamma_p = \max\left\{1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)}\right\} \leq c\gamma \ .$$

So we have that

$$\left\|V_v^{\text{mixed}}\right\|_p \leq \Psi_p\left(\left(L_2c^2\gamma\right)^c M_v, \left(L_2c^2\gamma\right)^c\sigma_v^2\right) \ .$$

Now by the same method as in eq. (A.4), we get the result. □

Adding a query element In many cases, we would like to prove that these properties continue to hold even when conditioning on a query element. An example would be the case where we are interested in the weight of the elements in the bin for which the query element, q, hashes to, i.e., we would like that $\sum_{x\in S} w_x\,[h(x) = h(q)]$ is concentrated when conditioning on q. Formally, this corresponds to having the value function $v\colon \Sigma^c \times [m] \times [m]$ defined by $v(x,j,k) = w_x\,[x \in S]\,[j = k]$ and then proving concentration on $\sum_{x\in\Sigma^c\setminus\{q\}} v(x,h(x),h(q))$ when conditioning on q. We show that this holds both for simple tabulation and mixed tabulation.

Theorem A.12. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function and let $q \in \Sigma^c$ be a designated query element. Let $v\colon \Sigma^c \times [m] \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j,k) = 0$ for all keys $x \in U$ and all $k \in [m]$. Define the random variable $V_{v,q}^{\text{simple}} = \sum_{x\in\Sigma^c\setminus\{q\}} v(x,h(x),h(q))$ and the random variables*

$$M_{v,q} = \max_{x\in\Sigma^c\setminus\{q\}, j\in[m]} |v(x,j,h(q))| \ ,$$

$$\sigma_{v,q}^2 = \frac{1}{m}\sum_{x\in\Sigma^c\setminus\{q\}}\sum_{j\in[m]} v(x,j,h(q))^2 \ ,$$

which only depend on the randomness of $h(q)$. Then for all $p \geq 2$

$$\mathrm{E}\Big[\Big(V_{v,q}^{\text{simple}}\Big)^p \,\Big|\, h(q)\Big]^{1/p} \leq \Psi_p\left(K_c\gamma_p^{c-1}M_{v,q}, K_c\gamma_p^{c-1}\sigma_{v,q}^2\right) ,$$

where $K_c = L_1\left(L_2 c\right)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}\right)/c, p\right\}}{\log\left(e^2 m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)} .$$

Theorem A.13. *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation hash function and let $q \in \Sigma^c$ be a designated query element. Let $v\colon \Sigma^c \times [m] \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j,k) = 0$ for all keys $x \in U$ and all $k \in [m]$. Define the random variable $V_{v,q}^{\text{simple}} = \sum_{x\in\Sigma^c\setminus\{q\}} v(x,h(x),h(q))$ and the random variables*

$$M_{v,q} = \max_{x\in\Sigma^c\setminus\{q\}, j\in[m]} |v(x,j,h(q))| ,$$
$$\sigma_{v,q}^2 = \frac{1}{m}\sum_{x\in\Sigma^c\setminus\{q\}}\sum_{j\in[m]} v(x,j,h(q))^2 ,$$

which only depend on the randomness of $h(q)$. For all $p \geq 2$ then

$$\mathrm{E}\Big[\Big(V_{v,q}^{\text{simple}}\Big)^p \,\Big|\, h(q)\Big]^{1/p} \leq \Psi_p\left(K_c\gamma_p^{c}M_{v,q}, K_c\gamma_p^{c}\sigma_{v,q}^2\right) \tag{A.6}$$

where $K_c = L_1\left(L_2 c\right)^{c}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \max\left\{1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)}\right\} .$$

A.1.5 Technical Overview

Fully Random Hashing

Sub-Gaussian bounds A random variable X is said to be sub-Gaussian with parameter σ if $\|X\|_p \leq \sqrt{p}\sigma$ for all $p \geq 2$. It is a well-known fact that the sum of independent bounded random variables are sub-Gaussian. In the context of fully random hashing, we have that

$$\left\|\sum_{x\in U} v(x,h(x))\right\|_p \leq \sqrt{4p}\sqrt{\sum_{x\in U}\|v[x]\|_\infty^2} . \tag{A.7}$$

A natural question is whether this is the best sub-Gaussian bound we can get. If we are just interested in the contribution to a single bin, i.e., $v(x,j) = w_x([j=0] - \frac{1}{m})$, then we

can obtain a better sub-Gaussian bound. By using the result of Oleszkiewicz [Ole03], we get that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_p \leq L \sqrt{\frac{p}{\log m}} \sqrt{\sum_{x \in U} w_x^2} \,, \tag{A.8}$$

where L is a universal constant. This shows that eq. (A.7) can be improved in certain situations. We improve on this by proving a generalization of eq. (A.8). We show that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_p \leq L \sqrt{\frac{p}{\log \left(\frac{e^2 m \sum_{x \in U} \|v[x]\|_\infty^2}{\sum_{x \in U} \|v[x]\|_2^2} \right)}} \sqrt{\sum_{x \in U} \|v[x]\|_\infty^2} \,, \tag{A.9}$$

where L is a universal constant. It is easy to check that if $v(x, j) = w_x([j = 0] - \frac{1}{m})$ then it reduces to eq. (A.8) and that it is stronger than eq. (A.7).

Moments for general random variables As part of our analysis we develop a couple of lemmas for general random variables which might be of independent interest. We prove a lemma that provides a simple bound for weighted sums of independent and identically distributed random variables.

Lemma A.14. *Let $(X_i)_{i \in [n]}$ and X be independent and identically distributed symmetric random variables, and let $(a_i)_{i \in [n]}$ be a sequence of reals.*[4] *If $p \geq 2$ is an even integer then*

$$\left\| \sum_{i \in [n]} a_i X_i \right\|_p \leq K \sup \left\{ \frac{p}{s} \left(\frac{\sum_{i \in [n]} a_i^s}{p} \right)^{1/s} \|X\|_s \,\middle|\, 2 \leq s \leq p \right\} \,,$$

where $K \leq 4e$ is a universal constant.

If we consider Laplace distributed random variables then it is possible to show that Lemma A.14 is tight up to a universal constant. Thus a natural question to ask is whether Lemma A.14 is tight, i.e., can we prove a matching lower bound. But unfortunately, if you consider Gaussian distributed variables then we see that Lemma A.14 is not tight. It would be nice if there existed a simple modification of Lemma A.14 which had a matching lower bound.

Moments of functions of random variables As part of the analysis of tabulation hashing, we will need to analyze random variables of the form $\Psi_p(X, Y)$ where X and Y are random variables. More precisely, we have to bound $\|\Psi_p(X, Y)\|_p$. It is not immediately clear how one would do this but we prove a general lemma that helps us in this regard.

[4] A symmetric random variable, X, is a random variable that is symmetric around zero, i.e., $\Pr[X \geq t] = \Pr[-X \geq t]$ for all $t \geq 0$.

Lemma A.15. *Let $f\colon \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a non-negative function which is monotonically increasing in every argument, and assume that there exist positive reals $(\alpha_i)_{i\in[n]}$ and $(t_i)_{i\in[n]}$ such that for all $\lambda \geq 0$*

$$f(\lambda^{\alpha_0} t_0, \ldots, \lambda^{\alpha_{n-1}} t_{n-1}) \leq \lambda f(t_0, \ldots, t_{n-1}) \,.$$

Let $(X_i)_{i\in[n]}$ be non-negative random variables. Then for all $p \geq 1$ we have that

$$\|f(X_0, \ldots, X_{n-1})\|_p \leq n^{1/p} \max_{i\in[n]} \left(\frac{\|X_i\|_{p/\alpha_i}}{t_i} \right)^{1/\alpha_i} f(t_0, \ldots, t_{n-1}) \,.$$

If we can choose $t_i = \|X_i\|_{p/\alpha_i}$ for all $i \in [n]$, then we get the nice expression

$$\|f(X_0, \ldots, X_{n-1})\|_p \leq n^{1/p} f(\|X_0\|_{p/\alpha_0}, \ldots, \|X_{n-1}\|_{p/\alpha_{n-1}}) \,.$$

Now the result is natural to compare to the triangle inequality that says that $\|X + Y\|_p \leq \|X\|_p + \|Y\|_p$, which corresponds to considering $f(x, y) = x + y$, and to Cauchy-Schwartz that says that $\|XY\|_p \leq \|X\|_{2p} \|Y\|_{2p}$, which corresponds to $f(x, y) = xy$. These two examples might point to that the $n^{1/p}$ is superfluous, but by considering $f(x_0, \ldots, x_{n-1}) = \max\{x_0, \ldots, x_{n-1}\}$ and Gaussian distributed variables, it can be shown that Lemma A.15 is tight up to a constant factor.

Tabulation Hashing

Symmetrization The analyses of chaoses have mainly focused on two types of chaoses: Chaoses generated by non-negative random variables and chaoses generated by symmetric random variables. It might appear strange that focus has not been on chaoses generated by mean zero random variables. The reason is that a symmetrization argument reduces the analysis of chaoses generated by mean zero random variables to the analysis of chaoses generated by symmetric random variables. More precisely, a standard symmetrization shows that

$$2^{-c} \left\| \sum_{i_0,\ldots,i_{c-1}\in[n]} a_{i_0,\ldots,i_{c-1}} \prod_{j\in[c]} \varepsilon_{i_j}^{(j)} X_{i_j}^{(j)} \right\|_p \leq \left\| \sum_{i_0,\ldots,i_{c-1}\in[n]} a_{i_0,\ldots,i_{c-1}} \prod_{j\in[c]} X_{i_j}^{(j)} \right\|_p \leq 2^c \left\| \sum_{i_0,\ldots,i_{c-1}\in[n]} a_{i_0,\ldots,i_{c-1}} \prod_{j\in[c]} \varepsilon_{i_j}^{(j)} X_{i_j}^{(j)} \right\|_p ,$$

where $(\varepsilon_i^{(j)})_{i\in[n],j\in[n]}$ are independent Rademacher variables.[5]

In our case, we can assume that $v(x, h(x))$ is a mean zero random variable but is not necessarily symmetric. We can remedy this by using the same idea of symmetrization. We

[5] A Rademacher variable, ε, is a random variable chosen uniformly from the set $\{-1, 1\}$, i.e., $\Pr[\varepsilon = -1] = \Pr[\varepsilon = 1] = \frac{1}{2}$.

define $\varepsilon\colon \Sigma^c \to \{-1,1\}$ to be a simple tabulation sign function, more precisely, we have a fully random table, $T_\varepsilon\colon [c]\times\Sigma\to\{-1,1\}$, and ε is then defined by $\varepsilon(\alpha_0,\ldots,\alpha_{c-1}) = \prod_{i\in[c]} T(i,\alpha_i)$. We then prove that for all $p\geq 2$

$$2^{-c}\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p \leq \left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \leq 2^c\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p . \tag{A.10}$$

The power of symmetrization lies in the fact that we get to assume that v is symmetric in the analysis without actually changing the value functions.

Somewhat surprisingly, we are able to improve the moment bound of Dahlgaard et al. [DKRT15] just by using symmetrization. Their result has a doubly exponential dependence on the size of the moment, p, which stems from a technical counting argument where they bound the number of terms which does not have an independent factor when expanding the expression $\left(\sum_{x\in\Sigma^c} v(x,h(x))\right)^p$. It appears difficult to directly improve their counting argument but by using eq. (A.10) we are able to circumvent this. Thus, just by using symmetrization and the insights of Dahlgaard et al. [DKRT15] we obtain the following result.

Lemma A.16. *Let $h\colon\Sigma^c\to[m]$ be a simple tabulation function, $\varepsilon\colon\Sigma^c\to\{-1,1\}$ be a simple tabulation sign function, and $v\colon\Sigma^c\times[m]\to\mathbb{R}$ be a value function. Then for every real number $p\geq 2$*

$$\left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \leq 2^c\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p \leq \sqrt{4p}^c\sqrt{\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}\,.$$

General value functions For most applications of hashing, we are either interested in the number of balls landing in a bin or in the number of elements hashing below a threshold. But we are studying the more general setting where we have a value function. A natural question is whether it is possible to obtain a simpler proof for the simpler settings. We do not believe this to be the case since the general setting of value functions will naturally show up when proving results by induction on c. More precisely, let us consider the case where we are interested in the number of elements from a set, $S\subseteq\Sigma^c$, that hash to 0. We then want to bound $\sum_{x\in S}\left([h(x)=0]-\frac{1}{m}\right) = \sum_{x\in\Sigma^c}[x\in S]\left([h(x)=0]-\frac{1}{m}\right)$. This can be rewritten as[6]

$$\sum_{x\in\Sigma^c}[x\in S]\left([h(x)=0]-\tfrac{1}{m}\right) = \sum_{\alpha\in\Sigma}\sum_{y\in\Sigma^{c-1}}[(y,\alpha)\in S]\left([h(y)\oplus T(c-1,\alpha)=0]-\tfrac{1}{m}\right)\,.$$

So if we define the value function $v'\colon\Sigma\times[m]\to\mathbb{R}$ by

$$v'(\alpha,j) = \sum_{y\in\Sigma^{c-1}}[(y,\alpha)\in S]\left([h\oplus j=0]-\tfrac{1}{m}\right)\,,$$

[6]For a partial key $y=(\beta_0,\ldots,\beta_{c-2})\in\Sigma^{c-1}$, we let $h(y)=\bigoplus_{i\in[c-1]}T(i,\beta_i)$.

then we get that $\sum_{x\in S}\left([h(x)=0]-\frac{1}{m}\right)=\sum_{\alpha\in\Sigma}v'(\alpha,T(c-1,\alpha))$. Thus, we see that general value functions are natural to consider in the context of tabulation hashing.

Instead of shying away from general value functions, we embrace them. This force us look at the problem differently and guides us in the correct direction. Using this insight naturally leads us to use eq. (A.9) and we prove the following moment bound, which is strictly stronger than Lemma A.16.

Lemma A.17. *Let $h\colon \Sigma^c\to[m]$ be a simple tabulation function, $\varepsilon\colon\Sigma^c\to\{-1,1\}$ be a simple tabulation sign function, and $v\colon\Sigma^c\times[m]\to\mathbb{R}$ be value function. Then for every real number $p\geq 2$*

$$\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p\leq\sqrt{K_c\frac{p\left(\max\{p,\log(m)\}\right)^{c-1}}{\log\left(1+\frac{m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)^c}}\sqrt{\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}\,,$$

where $K_c=(Lc)^c$ for a universal constant L.

This statement is often weaker than Theorem A.7 but perhaps a bit surprisingly, we will use Lemma A.17 as an important step in the proof of Theorem A.7.

Sum of squares of simple tabulation hashing A key element when proving Theorem A.7 is bounding the sums of squares

$$\sum_{j\in[m]}\left(\sum_{x\in\Sigma^c}v(x,h(x)\oplus j)\right)^2. \tag{A.11}$$

This was also one of the main technical challenges for the analysis of Aamand et al. [AKKR+20]. Instead of analyzing eq. (A.11), we will analyze a more general problem: Let $v_i\colon\Sigma^c\times[m]\to\mathbb{R}$ be a value function $i\in[k]$, we then want to understand the random variable.

$$\sum_{\substack{j_0,\dots,j_{k-1}\in[m]\\ \bigoplus_{i\in[k]}j_i=0}}\sum_{x_0,\dots,x_{k-1}\in\Sigma^c}\prod_{i\in[k]}v_i(x_i,j_i\oplus h(x_i)) \tag{A.12}$$

If we have $k=2$ and $v_0=v_1$ then this corresponds to eq. (A.11). By using a decoupling argument, it is possible to reduce the analysis of eq. (A.12) to the analysis of hash-based sums for simple tabulation hashing. We can then use Lemma A.17 to obtain the following lemma.

Lemma A.18. *Let $h\colon\Sigma^c\to[m]$ be a simple tabulation function, $\varepsilon\colon\Sigma^c\to\{-1,1\}$ be a simple tabulation sign function, and $v_i\colon\Sigma^c\times[m]\to\mathbb{R}$ be a value function for $i\in[k]$. For every real number $p\geq 2$*

$$\left\|\sum_{j\in[m]}\left(\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right)^2\right\|_p\leq\left(\frac{Lc\max\{p,\log(m)\}}{\log\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\sum_{x\in\Sigma^c}\|v[x]\|_1^2}\right)}\right)^c\sum_{x\in\Sigma^c}\|v[x]\|_2^2\,,$$

where L is a universal constant.

Proving the main result The proof of Theorem A.7 is by induction on c. We will use Theorem A.5 on one of the characters while fixing the other characters. This will give us an expression of the form

$$\left\| \Psi_p \left(\max_{\alpha\in\Sigma, j\in[m]} \left| \sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right|, \frac{\sum_{\alpha\in\Sigma, j\in[m]} \left(\sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right)^2}{m} \right) \right\|_p .$$

By applying Lemma A.15, we bound this by

$$\Psi_p \left(\left\| \max_{\alpha\in\Sigma, j\in[m]} \left| \sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right| \right\|_p, \left\| \frac{\sum_{\alpha\in\Sigma, j\in[m]} \left(\sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right)^2}{m} \right\|_p \right) .$$

We will bound $\left\| \max_{\alpha\in\Sigma, j\in[m]} \left| \sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right| \right\|_p$ by using the induction hypothesis, and we bound $\left\| \frac{\sum_{\alpha\in\Sigma, j\in[m]} \left(\sum_{y\in\Sigma^{c-1}} v((y,\alpha), h(y)\oplus j) \right)^2}{m} \right\|_p$ by using Lemma A.18. While this sketch is simple, the actual proof is quite involved and technical since one has to be very careful with the estimates.

A.1.6 Mixed Tabulation Hashing in Context

Our concentration bounds for mixed tabulation hashing are similar to those Aamand et al. [AKKR+20] for their tabulation-permutation hashing scheme and the schemes also have very similar efficiency, roughly a factor 2 slower than simple tabulation and orders of magnitude faster than any alternative with similar known concentration bounds. We shall make a more detailed comparison with tabulation-permutation in Section A.1.6.

As mentioned in the beginning of the introduction, the big advantage of proving concentration bounds for mixed tabulation hashing rather than for tabulation-permutation is that mixed tabulation hashing has many other strong probabilistic properties that can now be used in tandem with strong concentration. This makes mixed tabulation an even stronger candidate to replace abstract uniform hashing in real implementations of algorithms preserving many of the asymptotic performance guarantees.

Mixed tabulation inherits all the nice probabilistic properties known for simple and twisted tabulation[7]. Dahlgaard et al. [DKRT15] introduced mixed tabulation hashing to further get good statistics over k-partitions as used in classic streaming algorithms for counting of distinct elements by Flajolet et al. [FM85; FFGa07; HNH13], and for fast set similarity in large-scale machine learning by Li et al. [LOZ12; SL14a; SL14b].

Selective full randomness with mixed tabulation The main result of Dahlgaard et al. [DKRT15] for mixed tabulation is that it has a certain kind of selective full randomness (they did not have a word for it). An ℓ-bit mask M with don't cares is of the form $\{0,1,?\}^\ell$. An ℓ-bit string $B \in \{0,1\}^\ell$ matches M if it is obtained from M by replacing each ? with a 0 or a 1. Given a hash function returning ℓ-bit hash values, we can use M to select the set Y of keys that match M. Consider a mixed tabulation hash function $h : \Sigma^c \to \{0,1\}^\ell$ using d derived characters. The main result of Dahlgaard et al. [DKRT15, Theorem 4] is that if the expected number of selected keys is less than $|\Sigma|/2$, then, w.h.p., the free (don't care) bits of the hash values of Y are fully random and independent. More formally,

Theorem A.19 (Dahlgaard et al. [DKRT15, Theorem 4])**.** *Let $h : \Sigma^c \to \{0,1\}^\ell$ be a mixed tabulation hash function using d derived characters. Let M be an ℓ-bit mask with don't cares. For a given key set $X \subseteq \Sigma^c$, let Y be the set of keys from X with hash values matching M. If $\mathrm{E}[|Y|] \leq |\Sigma|/(1+\Omega(1))$, then the free bits of the hash values in Y are fully random with probability $1 - O(|\Sigma|^{1-\lfloor d/2 \rfloor})$.*

The above result is best possible in that since we only have $O(|\Sigma|)$ randomness in the tables, we cannot hope for full randomness of an asymptotically larger set Y.

In the applications from [DKRT15], we also want the size of the set Y to be concentrated around its mean and by Corollary A.11, the concentration is essentially as strong as with fully random hashing and it holds for any $d \geq 1$.

In [DKRT15] they only proved weaker concentration bounds for the set Y selected in Theorem A.19. Based on the concentration bounds for simple tabulation by Pătraşcu and Thorup [PT12], they proved that if the set Y from A.19 had $\mathrm{E}[Y] \in [|\Sigma|/8, 3|\Sigma|/4]$, then within the same probability of $1 - O(|\Sigma|^{1-\lfloor d/2 \rfloor})$, it has

$$|Y| = \mathrm{E}[Y]\left(1 \pm O\left(\sqrt{\frac{\log|\Sigma|(\log\log|\Sigma|)^2}{|\Sigma|}}\right)\right). \tag{A.13}$$

With Corollary A.11, for $\mathrm{E}[Y] = \Theta(|\Sigma|)$, we immediately tighten (A.13) to the cleaner

$$|Y| = \mathrm{E}[Y]\left(1 \pm O\left(\sqrt{\frac{\log|\Sigma|}{|\Sigma|}}\right)\right). \tag{A.14}$$

[7]This is not a black box reduction, but both twisted and mixed tabulation hashing applies simple tabulation to a some changed keys, so any statement holding for arbitrary sets of input keys is still valid. Moreover, mixed tabulation with one derived character corresponds to mixed tabulation applied to keys with an added 0-character head, and having more derived characters does not give worse results.

While the improvement is "only" a factor $(\log\log|\Sigma|)^2$, the important point here is that (A.14) is the asymptotic bound we would get with fully-random hashing. Also, while Dahlgaard et al. only proved (A.13) for the special case of $\mathrm{E}[Y] \in [|\Sigma|/8, 3|\Sigma|/4]$, our (A.14) is just a special case of Corollary A.11 which holds for arbitrary values of $\mathrm{E}[Y]$ and arbitrary value functions.

Dahlgaard et al. presented some very nice applications of mixed tabulation to problems in counting and machine learning and machine learning. The way they use Theorem A.19 is rather subtle.

Mixed Tabulation Hashing Versus Tabulation-Permutation Hashing

As mentioned earlier, our new concentration bounds are similar to those proved by Aamand et al. [AKKR+20] for their tabulation-permutation hashing scheme. However, now we also have moment bounds covering the tail, and we have the first understanding of what happens when c is not constant. It is not clear if this new understanding applies to tabulation-permutation. As discussed above, the advantage of having the concentration bounds for mixed tabulation hashing is that we can use them in tandem with the independence result from Theorem A.19, which does not hold for tabulation-permutation.

Tabulation-permutation is similar to mixed tabulation hashing in its resource consumption. Consider the mapping $\Sigma^c \to \Sigma^c$. Tabulation-permutation first uses simple tabulation $h : \Sigma^c \to \Sigma^c$. Next it applies a random permutation $\pi_i : \Sigma \overset{1-1}{\to} \Sigma$ to each output character $h(x)_i$, that is, $x \mapsto (\pi_1(h(x)_1), \ldots, \pi_c(h(x)_c)$. Aamand et al. [AKKR+20] also suggest tabulation-1permutation hashing, which only permutes the most significant character. This scheme does not provide concentration for all value functions, but it does work if we select keys from intervals.

Aamand et al. [AKKR+20] already made a thorough experimental and theoretical comparison between tabulation-permutation, mixed tabulation, and many other schemes. In this comparison, mixed tabulation played the role of a similar scheme with not as strong known concentration bounds. In the experiments, mixed tabulation hashing with c derived characters performed similar to tabulation-permutation in speed. Here we proved stronger concentration bounds for mixed tabulation even with a single character, where it should perform similar to tabulation-1permutation (both use $c+1$ lookups). Both mixed tabulation hashing and tabulation-permutation hashing were orders of magnitude faster than any alternative with similar known concentration bounds. We refer to [AKKR+20; DKT17] for more details. In particular, [DKT17] compares mixed tabulation with popular cryptographic hash functions that are both slower and have no guarantees in these algorithmic contexts.

One interesting advantage of mixed tabulation hashing over tabulation-permutation hashing is that mixed tabulation hashing, like simple tabulation hashing, only needs randomly filled character tables. In contrast, tabulation-permutation needs tables that represent permutations. Thus, all we need to run mixed tabulation hashing is a pointer to some random bits. These could be in read-only memory shared across different applications. Read-only memory is much less demanding than standard memory since there can be no write-conflicts, so we could imagine some special large, fast, and cheap read-only memory,

pre-filled with random bits, e.g., generated by a quantum-device. This would open up for larger characters, e.g., 16- or 32-bit characters, and it would free up the cache for other applications.

A.2 Preliminaries

In this section, we will introduce the notation which will be used throughout the paper. We will start by introducing the notation from probability theory that we need and afterwards we will introduce notation that will help in the reasoning about tabulation hashing.

We will use the following basic mathematical notation: We define $\mathbb{N}$ the set of non-negative integers, for $n \in \mathbb{N}$ we shall define $[n] = \{0, \ldots, n-1\}$, in particular $[n] = \emptyset$, and for an event $\mathcal{A}$ we shall define $[\mathcal{A}]$ to be the indicator on $\mathcal{A}$, i.e., $[\mathcal{A}] = 1$ if $\mathcal{A}$ is true and $[\mathcal{A}] = 0$ otherwise.

If $n \in \mathbb{N}$ is non-negative integer, $(X_i)_{i\in[n]}$ are real variables, and $j \in [n+1]$ we shall define $X_{<j} = \sum_{i\in[n],i<j} X_i = \sum_{i\in[j]} X_i$. Similarly, for sets $(A_i)_{i\in[n]}$ and $j \in [n+1]$ we shall define $A_{<j} = \bigcup_{i\in[j]} A_i$.

We will be using the following version of Stirling's approximation [Mar65] which holds for all integers n,

$$\Gamma(n+1) = n! \leq e\sqrt{n}\left(\frac{n}{e}\right)^n . \tag{A.15}$$

A.2.1 Probability Theory

In the following, we introduce the necessary notions of probability theory. We will assume that we are given a probability space $(\Omega, \mathcal{F}, P)$ throughout the paper but we will often not state it explicitly. We will be working with martingales and we shall therefore need notation and concepts from probability theory of a fairly general and abstract character. For an introduction to measure and probability theory, see, for instance, [Sch05].

Definition A.20. Let $(X_i)_{i\in[n]}$ be random variables on the probability space $(\Omega, \mathcal{F}, P)$. We denote by $\mathcal{G} = \sigma((X_i)_{i\in[n]}) \subseteq \mathcal{F}$ the smallest σ-algebra where $(X_i)_{i\in[n]}$ are all $\mathcal{G}$-measurable.

Definition A.21 (Conditional expectation)**.** Let X be a random variable on the probability space $(\Omega, \mathcal{F}, P)$, and let $\mathcal{G} \subseteq \mathcal{F}$ be a sub σ-algebra. If $\mathrm{E}[|X|] < \infty$ we can define the random variable $\mathrm{E}[X \,|\, \mathcal{G}]$ to be the conditional expectation of X given $\mathcal{G}$. It shall be $\mathcal{G}$-measurable and for all $G \in \mathcal{G}$ we have that $\mathrm{E}[\mathbb{1}_G \mathrm{E}[X \,|\, \mathcal{G}]] = \mathrm{E}[\mathbb{1}_G X]$.

We define the conditional expectation of X given a random variable Y by $\mathrm{E}[X \,|\, Y] = \mathrm{E}[X \,|\, \sigma(Y)]$.

Definition A.22 (Filtration and adapted sequence)**.** On a probability space $(\Omega, \mathcal{F}, P)$ a sequence of $(\mathcal{F}_i)_{i\in[n]}$ of sub σ-algebras is called a filtration if $\mathcal{F}_0 \subseteq \ldots \subseteq \mathcal{F}_{n-1} \subseteq \mathcal{F}$.

We say that a sequence of random variables $(X_i)_{i\in[n]}$ is adapted to a filtration $(\mathcal{F}_i)_{i\in[n]}$ if X_i is $\mathcal{F}_i$-measurable for all $i \in [n]$. We call $(X_i, \mathcal{F}_i)_{i\in[n]}$ an adapted sequence.

Definition A.23 (Martingale and martingale difference)**.** We call an adapted sequence $(X_i, \mathcal{F}_i)_{i\in[n]}$ a martingale sequence if $\mathrm{E}[X_i \mid \mathcal{F}_{i-1}] = X_{i-1}$ for all $i \in [n]$. (We define $\mathcal{F}_{-1} = \{\emptyset, \Omega\}$ and $X_{-1} = 0$.)

We call an adapted sequence $(Y_i, \mathcal{F}_i)_{i\in[n]}$ a martingale difference sequence if $\mathrm{E}[Y_i \mid \mathcal{F}_{i-1}] = 0$ for all $i \in [n]$. (We define $\mathcal{F}_{-1} = \{\emptyset, \Omega\}$.)

It should be noted that if $(Y_i, \mathcal{F}_i)_{i\in[n]}$ is a martingale difference sequence then $(Y_{<i+1}, \mathcal{F}_i)_{i\in[n]}$ is a martingale sequence. Similarly, if $(X_i, \mathcal{F}_i)_{i\in[n]}$ is a martingale sequence then $(X_i - X_{i-1}, \mathcal{F}_i)_{i\in[n]}$ is a martingale difference sequence.

Definition A.24 (p-norm)**.** Let $p \geq 1$ and X be a random variable with $\mathrm{E}[|X|^p] < \infty$. We then define the p-norm of X by $\|X\|_p = \mathrm{E}[|X|^p]^{1/p}$.

Let $p \geq 1$, $\mathcal{G}$ be a sub σ-algebra, and X be a random variable with $\mathrm{E}[|X|^p \mid \mathcal{G}] < \infty$. We then define the p-norm of X conditioned on $\mathcal{G}$ by $\|X \mid \mathcal{G}\|_p = \mathrm{E}[|X|^p \mid \mathcal{G}]^{1/p}$.

Similarly to conditional expectation, we will condition on random variables. Let $p \geq 1$ and let X and Y be a random variables. We then define the p-norm of X conditioned on Y by $\|X \mid Y\|_p = \mathrm{E}[|X|^p \mid \sigma(Y)]^{1/p}$

Now an important observation is that the p-norm is a seminorm which follows by the Minkowski inequality.

Lemma A.25 (Triangle inequality(Minkowski inequality))**.** *Let $p \geq 1$ and let X and Y be random variables with $\mathrm{E}[|X|^p] < \infty$ and $\mathrm{E}[|Y|^p] < \infty$. Then $\mathrm{E}[|X+Y|^p] < \infty$ and $\|X+Y\|_p \leq \|X\|_p + \|Y\|_p$.*

A.2.2 Tabulation Hashing

We will need to reason about the individual characters of a key, $x \in \Sigma^c$, and for that, we need some notation. Most of the definitions are taken from the paper by Aamand et al. [AKKR+20].

Definition A.26 (Position characters)**.** Let Σ be an alphabet and $c > 0$ a positive integer. We call an element $(i, y) \in [c] \times \Sigma$ a position character of Σ^c.

We will view a key $x = (y_0, \ldots, y_{c-1}) \in \Sigma^c$ as a set of c position characters, $\{(0, y_0), \ldots (c-1, y_{c-1})\} \subseteq [c] \times \Sigma$. Let $h\colon \Sigma^c \to [2^l]$ be a simple tabulation hash function and let $T\colon [c] \times \Sigma \to [2^l]$ be the random function used to define h. We will then overload the notation and for a set of position characters $y \subseteq [c] \times \Sigma$, define $h(y) = \bigoplus_{\alpha\in y} T(y)$. We note that this definition agrees with our correspondence between keys $x = (y_0, \ldots, y_{c-1}) \in \Sigma^c$, and set of position characters, $\{(0, y_0), \ldots (c-1, y_{c-1})\} \subseteq [c] \times \Sigma$, that is, $h(x) = h(\{(0, y_0), \ldots (c-1, y_{c-1})\})$. We have thus extended the domain of h to $\mathcal{P}([c] \times \Sigma)$. For sets of position characters $x_1, x_2 \in \mathcal{P}([c] \times \Sigma)$ we will write $x_1 \oplus x_2$ for the symmetric difference, i.e., $x_1 \oplus x_2 = (x_1 \cup x_2) \setminus (x_1 \cap x_2)$. We note that with the extended domain for h then $h(x_1 \oplus x_2) = h(x_1) \oplus h(x_2)$.

We will prove several of our statements by induction on the position characters and for this reason, we need the following definition.

Definition A.27 (Group of keys)**.** Let $\{\alpha_0, \dots, \alpha_{r-1}\} = [c] \times \Sigma^c$ be an enumeration of the position characters of Σ^c. For each $i \in [r]$ we denote by $G_i \subseteq \Sigma^c$ the i'th group of keys with respect to the ordering of position characters, and define it by $G_i = \{x \in \Sigma^c \mid \alpha_i \in x, x \subseteq \{\alpha_0, \dots, \alpha_{i-1}\}\}$.

We will need the following lemmas by Aamand et al. [AKKR+20].

Lemma A.28 ([AKKR+20])**.** *Let $w\colon \Sigma^c \to \mathbb{R}_{\geq 0}$ be a weight function, then there exists an ordering $\{\alpha_0, \dots, \alpha_{r-1}\}$ of the position characters of Σ^c, such that for all $i \in [r]$,*

$$\sum_{x \in G_i} w(x) \leq \left(\max_{x \in \Sigma^c} w(x)\right)^{1/c} \left(\sum_{x \in \Sigma^c} w(x)\right)^{1-1/c}.$$

Lemma A.29. *Let $k \in \mathbb{N}$ be a positive integer, and $w_0, \dots, w_{k-1}\colon \Sigma^c \to \mathbb{R}$ be weight functions. Then,*

$$\sum_{\substack{x_0,\dots,x_{k-1} \in \Sigma^c \\ \bigoplus_{i\in[k]} x_i = \emptyset}} \prod_{j \in [k]} w_i(x_i) \leq \sqrt{\tfrac{k}{2}}^{kc} \prod_{i \in [k]} \sqrt{\sum_{x \in \Sigma^c} w_i(x)^2}\,. \tag{A.16}$$

A.3 Moment Inequalities

The goal of this chapter is to establish a series of technical lemmas concerning moments which will be crucial in the later part of the paper. An important tool will be the function $\Psi_p\colon \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ which gives a qualitative way of measuring how close the centered moments of sums of weighted Bernoulli variables resembles the central moments of the Poisson distribution.

Definition A.30. For $p \geq 2$ we define the function $\Psi_p\colon \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ as follows,

$$\Psi_p(M, \sigma^2) = \begin{cases} \left(\frac{\sigma^2}{pM^2}\right)^{1/p} M & \text{if } p < \log \frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{p}\sigma & \text{if } p < e^2 \frac{\sigma^2}{M^2} \\ \frac{p}{e \log \frac{pM^2}{\sigma^2}} M & \text{if } \max\left\{\log \frac{pM^2}{\sigma^2}, e^2 \frac{\sigma^2}{M^2}\right\} \leq p \end{cases}.$$

Remark A.31. From the definition it is not clear that Ψ_p is well-defined, but if $2 \leq p < e^2 \frac{\sigma^2}{M^2}$ then $\log \frac{pM^2}{\sigma^2} < 2$, hence at most one of the first two cases are satisfied at any given time. This shows that Ψ_p is indeed well-defined.

We show in Lemma A.4 that, up to constant, $\Psi_p(1, \lambda)$ is equal to the p-norm of a variable distributed as a centered Poisson variable with parameter λ. This implies that, up to constant, $\Psi_p(M, \sigma^2)$ is equal to M times the p-norm of a variable distributed as a centered Poisson variable with parameter $\frac{\sigma^2}{M^2}$.

When we later prove our concentration results for simple tabulation and mixed tabulation, we will need to bound $\|\Psi_p(X, Y)\|_p$ for random variables X and Y. Now to

handle this we will develop a general lemma that bounds the moments of a function of random variables by the moments of the random variables. This will be the focus of Appendix A.3.3.

A.3.1 Moments of Poisson Distributed Variables

We start this section by proving a number of properties of the Ψ_p-function which we will use extensively. Afterward, we will establish the connection between Ψ_p-function and the p-norm of Poisson distributed variables.

Lemma A.32. *Let $p \geq 2$ then the function Ψ_p satisfies the following properties:*

1. *For all positive reals $M > 0$ and $\sigma > 0$,*

$$\Psi_p(M, \sigma^2) = M \sup \left\{ \frac{p}{s} \left(\frac{\sigma^2}{pM^2} \right)^{1/s} \middle| 2 \leq s \leq p \right\} . \tag{A.17}$$

2. *For all positive reals $M > 0$ and $\sigma > 0$,*

$$\Psi_p(M, \sigma^2) \leq \max\left\{ \tfrac{1}{2}\sqrt{p}\sigma, \tfrac{1}{2e}pM \right\} . \tag{A.18}$$

3. *For all positive reals $M > 0$ and $\sigma > 0$,*

$$\tfrac{1}{2}\sqrt{p}\sigma \leq \Psi_p(M, \sigma^2) . \tag{A.19}$$

4. *For all positive reals $M > 0$ and $\sigma > 0$ with $e^2 \frac{\sigma^2}{M^2} \leq p$,*

$$\Psi_p(M, \sigma^2) \leq \frac{p}{e \log \frac{pM^2}{\sigma^2}} M . \tag{A.20}$$

5. *For all positive reals $M > 0$ and $\sigma > 0$ and all $\lambda \geq 1$,*

$$\Psi_p(\lambda M, \lambda\sigma^2) \leq \lambda \Psi_p(M, \sigma^2) . \tag{A.21}$$

6. *For all positive reals $M > 0$ and $\sigma > 0$ and all $\lambda \geq 1$,*

$$\lambda \Psi_p(M, \sigma^2) \leq \Psi_p(\lambda^2 M, \lambda^2 \sigma^2) . \tag{A.22}$$

7. *If $f : \mathbb{R}_+ \to \mathbb{R}_+$ is an increasing function, where $p \mapsto f(1/p)$ is log-convex and where there exists positive reals $K > 0, M > 0$, and $\sigma > 0$ such that $f(p) \leq K\Psi_p(M, \sigma^2)$ for all even integers $p \geq 2$, then*

$$f(p) \leq 2K\Psi_p(M, \sigma^2) , \tag{A.23}$$

for all reals $p \geq 2$.

Proof.

Proof of eq. (A.17). Let $\alpha = \frac{\sigma^2}{pM^2}$ and define the function $f(s) = \frac{1}{s}\alpha^{1/s}$. Taking the derivative we get that $f'(s) = \frac{1}{s^2}\left(-\frac{\log \alpha}{s}\alpha^{1/s} - \alpha^{1/s}\right)$. From this it is clear that f is maximized at the point $s^* = \min\left\{\max\{2, \log 1/\alpha\}, p\right\}$ on the interval $[2, p]$. This implies that

$$\begin{aligned}
M \sup\left\{\frac{p}{s}\left(\frac{\sigma^2}{pM^2}\right)^{1/s} \,\middle|\, 2 \le s \le p\right\} &= M \sup\left\{pf(s) \,|\, 2 \le s \le p\right\} \\
&= Mpf(s^*) \\
&= Mp\begin{cases} \frac{1}{p}\left(\frac{\sigma^2}{pM^2}\right)^{1/p} & \text{if } p < \log\frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{\frac{\sigma^2}{pM^2}} & \text{if } \log\frac{pM^2}{\sigma^2} < 2 \\ \frac{1}{\log\frac{pM^2}{\sigma^2}}e^{-1} & \text{if } 2 \le \log\frac{pM^2}{\sigma^2} \le p \end{cases} \\
&= \begin{cases} \left(\frac{\sigma^2}{pM^2}\right)^{1/p} M & \text{if } p < \log\frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{p}\sigma & \text{if } p < e^2\frac{\sigma^2}{M^2} \\ \frac{p}{e\log\frac{pM^2}{\sigma^2}}M & \text{if } \max\left\{\log\frac{pM^2}{\sigma^2}, e^2\frac{\sigma^2}{M^2}\right\} \le p \end{cases}
\end{aligned}$$

which proves the claim.

Proof of eq. (A.18). We note that if $p \le \log\frac{pM^2}{\sigma^2}$ then

$$\left(\frac{\sigma^2}{pM^2}\right)^{1/p} M \le \frac{1}{e}M \le \frac{1}{2e}pM \,,$$

and if $p > e^2\frac{\sigma^2}{M^2}$ then,

$$\frac{p}{e\log\frac{pM^2}{\sigma^2}}M \le \frac{1}{2e}pM \,.$$

This shows the upper bound.

Proof of eq. (A.19). The lower bound follows from eq. (A.17) since,

$$\Psi_p(M, \sigma^2) = M \sup\left\{\frac{p}{s}\left(\frac{\sigma^2}{pM^2}\right)^{1/s} \,\middle|\, 2 \le s \le p\right\} \ge M\frac{p}{2}\left(\frac{\sigma^2}{pM^2}\right)^{1/2} = \frac{1}{2}\sqrt{p}\sigma \,.$$

Proof of eq. (A.20). From eq. (A.17) we know that $\Psi_p(M, \sigma^2) = M \sup\left\{\frac{p}{s}\left(\frac{\sigma^2}{pM^2}\right)^{1/s} \,\middle|\, 2 \le s \le p\right\}$. We then get the upper bound,

$$\Psi_p(M, \sigma^2) \le M \sup\left\{\frac{p}{s}\left(\frac{\sigma^2}{pM^2}\right)^{1/s} \,\middle|\, 2 \le s\right\} \,.$$

Now using the same method as in the proof of eq. (A.17), we get that the expression is maximized at $s^* = \frac{\sigma^2}{pM^2} \geq 2$ and we get that

$$\Psi_p(M, \sigma^2) \leq \frac{p}{e \log \frac{pM^2}{\sigma^2}} M \ .$$

Proof of eq. (A.21). We first notice that $\lambda \Psi_p(M, \sigma^2) = \Psi_p(\lambda M, \lambda^2 \sigma^2)$. Since $y \mapsto \Psi_p(\lambda M, y)$ is monotonically increasing then,

$$\Psi_p(\lambda M, \lambda \sigma^2) \leq \Psi_p(\lambda M, \lambda^2 \sigma^2) = \lambda \Psi_p(M, \sigma^2) \ .$$

Proof of eq. (A.22). We will again use that $\lambda \Psi_p(M, \sigma^2) = \Psi_p(\lambda M, \lambda^2 \sigma^2)$. This time we will use that $x \mapsto \Psi_p(x, \lambda^2 \sigma^2)$ is monotonically increasing,

$$\lambda \Psi_p(M, \sigma^2) = \Psi_p(\lambda M, \lambda^2 \sigma^2) \leq \Psi_p(\lambda^2 M, \lambda^2 \sigma^2) \ .$$

Proof of eq. (A.23). Let $p \geq 2$ be a real and define let $2 \leq q$ be the largest even integer with $q \leq p$, that is, q is the unique even integer satisfying that $q \leq p < 2q$. Now let $\theta \geq [0, 1]$ be defined by the equation $\frac{1}{p} = \theta \frac{1}{q} + (1 - \theta) \frac{1}{2q}$. By the log-convexity of $p \mapsto f(1/p)$ we get that $f(p) \leq f(q)^\theta f(2q)^{1-\theta}$. We then have to consider two different cases.

If $\log \frac{pM^2}{\sigma^2} < p$ then it is easy to check that $\Psi_{2p}(M, \sigma^2) \leq 2\Psi_p(M, \sigma^2)$, hence we get that

$$f(p) \leq f(q)^\theta f(2q)^{1-\theta} \leq f(2q) \leq K \Psi_{2q}(M, \sigma^2) \leq K \Psi_{2p}(M, \sigma^2) \leq 2K \Psi_p(M, \sigma^2) \ .$$

If $p \leq \log \frac{pM^2}{\sigma^2}$ then we also know that $q \leq \log \frac{qM^2}{\sigma^2}$ and $\Psi_q(M, \sigma^2) = \left(\frac{\sigma^2}{qM^2}\right)^{1/q} M \leq \sqrt{2} \left(\frac{\sigma^2}{pM^2}\right)^{1/q} M$, where we have used that $p < 2q$ and $2 \leq q$. Let $p < q'$ be defined by $q' = \log \frac{q'}{M^2} \sigma^2$. If $2q \leq q'$ then we have that $\Psi_{2q}(M, \sigma^2) = \left(\frac{\sigma^2}{2qM^2}\right)^{1/2q} M \leq \left(\frac{\sigma^2}{pM^2}\right)^{1/2q} M$ and we get that

$$f(p) \leq f(q)^\theta f(2q)^{1-\theta} \leq \sqrt{2} K \left(\frac{\sigma^2}{qM^2}\right)^{\theta/q} \left(\frac{\sigma^2}{2qM^2}\right)^{(1-\theta)/2q} M = \sqrt{2} K \Psi_q(M, \sigma^2) \ .$$

If $q \leq q' \leq 2q$ then

$$\Psi_{2q}(M, \sigma^2) \leq \Psi_{2q'}(M, \sigma^2) \leq 2\Psi_{q'}(M, \sigma^2) = 2 \left(\frac{\sigma^2}{q'M^2}\right)^{1/q'} M \leq 2 \left(\frac{\sigma^2}{pM^2}\right)^{1/2q} M \ .$$

Combining these two facts give us that

$$f(p) \leq f(q)^\theta f(2q)^{1-\theta} \leq 2K \left(\frac{\sigma^2}{qM^2}\right)^{\theta/q} \left(\frac{\sigma^2}{2qM^2}\right)^{(1-\theta)/2q} M = 2K \Psi_q(M, \sigma^2) \ .$$

This finishes the proof of Lemma A.32. □

We are now ready to establish the connection between the Ψ_p-function and the p-norms of Poisson distributed random variables.

Lemma A.33. *There exist universal constants K_1 and K_2 satisfying that for a Poisson distributed random variable, X, with $\lambda = \mathrm{E}[X]$*

$$K_2\Psi_p(1,\lambda) \leq \|X-\lambda\|_p \leq K_1\Psi_p(1,\lambda)\ ,$$

for all $p \geq 2$.

For the proof, we need the following result by Latała [Lat97] that gives a tight bound for p-norms of sums of independent and identically distributed symmetric variables.

Lemma A.33 (Latała [Lat97])**.** *If $(X_i)_{i\in[n]}$ are independent and identically distributed symmetric variables and $p \geq 2$ then,*

$$\left\|\sum_{i\in[n]} X_i\right\|_p \leq K_1 \sup\left\{\frac{p}{s}\left(\frac{n}{p}\right)^{1/s}\|X_0\|_s \,\middle|\, \max\{2,\tfrac{p}{n}\} \leq s \leq p\right\}\ ,$$

and

$$\left\|\sum_{i\in[n]} X_i\right\|_p \geq K_2 \sup\left\{\frac{p}{s}\left(\frac{n}{p}\right)^{1/s}\|X_0\|_s \,\middle|\, \max\{2,\tfrac{p}{n}\} \leq s \leq p\right\}\ .$$

Here K_1 and K_2 are universal constants.

Proof of Lemma A.4. We will use the standard fact that the Poisson distribution is the limit of a binomial distribution, with fixed mean λ, as the number of trials go to infinity. Let $\left(Y_i^{(n)}\right)_{i\in[n]}$ be independent Bernoulli variables with $\Pr\left[Y_i^{(n)}=1\right] = \frac{\lambda}{n}$ for $n \geq 1$. We then get that

$$\|X-\lambda\|_p = \lim_{n\to\infty}\left\|\sum_{i\in[n]}\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p\ .$$

We let $(\varepsilon_i)_{i\in\mathbb{N}}$ be independent Rademacher variables. We will argue that

$$\tfrac{1}{2}\left\|\sum_{i\in[n]}\varepsilon_i\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p \leq \left\|\sum_{i\in[n]}\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p \leq 2\left\|\sum_{i\in[n]}\varepsilon_i\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p\ . \tag{A.24}$$

We defer the proof of eq. (A.24) to the end. Using eq. (A.24) it is enough to show that $\lim_{n\to\infty}\left\|\sum_{i\in[n]}\varepsilon_i\left(Y_i^{(n)}-\frac{\lambda}{n}\right)\right\|_p$ is at most a constant away from $\Psi_p(1,\lambda)$. We use

Lemma A.33 to get that

$$\begin{aligned}
&\lim_{n\to\infty} \left\| \sum_{i\in[n]} \varepsilon_i \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \right\|_p \\
&\qquad \le \lim_{n\to\infty} K_1 \sup \left\{ \frac{p}{s} \left(\frac{n}{p} \right)^{1/s} \left\| Y_0^{(n)} \right\|_s \,\middle|\, \max\{2, \tfrac{p}{n}\} \le s \le p \right\} \\
&\qquad = \lim_{n\to\infty} K_1 \sup \left\{ \frac{p}{s} \left(\frac{n}{p} \left(\tfrac{\lambda}{n} \left(1 - \tfrac{\lambda}{n}\right)^s + \left(1 - \tfrac{\lambda}{n}\right) \tfrac{\lambda^s}{n^s} \right) \right)^{1/s} \,\middle|\, \max\{2, \tfrac{p}{n}\} \le s \le p \right\} \\
&\qquad = \lim_{n\to\infty} K_1 \sup \left\{ \frac{p}{s} \left(\frac{\lambda}{p} \left(\left(1 - \tfrac{\lambda}{n}\right)^s + \left(1 - \tfrac{\lambda}{n}\right) \tfrac{\lambda^{s-1}}{n^{s-1}} \right) \right)^{1/s} \,\middle|\, \max\{2, \tfrac{p}{n}\} \le s \le p \right\} \\
&\qquad = K_1 \sup \left\{ \frac{p}{s} \left(\frac{\lambda}{p} \right)^{1/s} \,\middle|\, 2 \le s \le p \right\} \\
&\qquad = K_1 \Psi_p(1, \lambda) \,.
\end{aligned}$$

The last equality follows by eq. (A.17). The proof of the lower bound is analogous.

Now we just need to prove eq. (A.24). We first consider the lower bound. Fixing $(\varepsilon_i)_{i\in\mathbb{N}}$ and using the triangle inequality we get that

$$\begin{aligned}
&\left\| \sum_{i\in[n]} \varepsilon_i \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \\
&\qquad = \left\| \sum_{i\in[n]} [\varepsilon_i = 1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) - \sum_{i\in[n]} [\varepsilon_i = -1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \\
&\qquad \le \left\| \sum_{i\in[n]} [\varepsilon_i = 1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p + \left\| \sum_{i\in[n]} [\varepsilon_i = -1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \,.
\end{aligned}$$

Now we use Jensen's inequality on each of the terms.

$$\begin{aligned}
&\left\| \sum_{i\in[n]} [\varepsilon_i = 1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \\
&\qquad = \left\| \sum_{i\in[n]} [\varepsilon_i = 1] \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) + \sum_{i\in[n]} [\varepsilon_i = -1] \, \mathrm{E}\left[Y_i^{(n)} - \tfrac{\lambda}{n} \right] \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \\
&\qquad \le \left\| \sum_{i\in[n]} \left(Y_i^{(n)} - \tfrac{\lambda}{n} \right) \,\middle|\, (\varepsilon_i)_{i\in\mathbb{N}} \right\|_p \,.
\end{aligned}$$

Unfixing $(\varepsilon_i)_{i\in\mathbb{N}}$ we get that

$$\left\|\sum_{i\in[n]}\varepsilon_i\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p \le 2\left\|\sum_{i\in[n]}\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p,$$

which establishes the lower bound of eq. (A.24).

For the upper bound of eq. (A.24) we first define $\left(Z_i^{(n)}\right)_{i\in[n]}$ to be independent copies of $\left(Y_i^{(n)}\right)_{i\in[n]}$. We then use Jensen's inequality to get that

$$\begin{aligned}\left\|\sum_{i\in[n]}\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p &= \left\|\sum_{i\in[n]}\left(\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)-\mathrm{E}\left[Z_i^{(n)}-\tfrac{\lambda}{n}\right]\right)\right\|_p \\ &\le \left\|\sum_{i\in[n]}\left(\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)-\left(Z_i^{(n)}-\tfrac{\lambda}{n}\right)\right)\right\|_p.\end{aligned}$$

We then note that due to independence $\left(Y_i^{(n)}-\frac{\lambda}{n}\right)-\left(Z_i^{(n)}-\frac{\lambda}{n}\right)$ is a symmetric variable, thus it has the same distribution as $\varepsilon_i\left(\left(Y_i^{(n)}-\frac{\lambda}{n}\right)-\left(Z_i^{(n)}-\frac{\lambda}{n}\right)\right)$. We use this and the triangle inequality to finish the upper bound,

$$\begin{aligned}\left\|\sum_{i\in[n]}\left(\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)-\left(Z_i^{(n)}-\tfrac{\lambda}{n}\right)\right)\right\|_p &= \left\|\sum_{i\in[n]}\varepsilon_i\left(\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)-\left(Z_i^{(n)}-\tfrac{\lambda}{n}\right)\right)\right\|_p \\ &\le 2\left\|\sum_{i\in[n]}\varepsilon_i\left(Y_i^{(n)}-\tfrac{\lambda}{n}\right)\right\|_p.\end{aligned}$$

This finishes the proof of Lemma A.4. □

A.3.2 Moments of General Random Variables

We start by proving a lemma that bounds the moments of weighted sums of independent and identically distributed variables. The lemma is similar to Lemma A.33 by Latała [Lat97] but it is not tight for all distributions.

Lemma A.34. *Let $(X_i)_{i\in[n]}$ and X be independent and identically distributed symmetric random variables, and let $(a_i)_{i\in[n]}$ be a sequence of reals. If $p\ge 2$ is an even integer then*

$$\left\|\sum_{i\in[n]}a_iX_i\right\|_p \le K\sup\left\{\frac{p}{s}\left(\frac{\sum_{i\in[n]}a_i^s}{p}\right)^{1/s}\|X\|_s \,\middle|\, 2\le s\le p\right\},$$

where $K\le 4e$ is a universal constant.

In the proof, we will need the following folklore result. We provide a proof of the result for completeness.

Lemma A.34. *Let $n \geq 0$ be a positive integer and let $a_1, \ldots, a_n \in \mathbb{R}$ be real numbers. If $x \in \mathbb{R}$ is a real number satisfying,*

$$x^n \leq \sum_{i=1}^{n} a_i x^{n-i} \ .$$

Then,

$$x \leq 2 \max_{1 \leq i \leq n} |a_i|^{1/i} \ .$$

Proof. The proof follows by noticing that

$$x^n \leq \sum_{i=1}^{n} a_i x^{n-i} \leq \sum_{i=1}^{n} |a_i| \, |x|^{n-i} \ .$$

We will now show that $\sum_{i=1}^{n} |a_i| \, |x|^{n-i} \leq \max_{i=1}^{n} 2^i \, |a_i| \, |x|^{n-i}$ by induction on n. The result is clearly true for $n = 1$. Now assume that the result holds for integers less than n then,

$$\sum_{i=1}^{n} |a_i| \, |x|^{n-i} \leq \max\left\{ 2a_n, 2 \sum_{i=1}^{n-1} |a_i| \, |x|^{n-i} \right\} \leq \max_{i=1}^{n} 2^i \, |a_i| \, |x|^{n-i} \ .$$

We then have that $x^n \leq \max_{i=1}^{n} 2^i \, |a_i| \, |x|^{n-i}$. This is equivalent with the statement that there exists an integer $1 \leq i \leq n$ with $x^n \leq 2^i \, |a_i| \, |x|^{n-i}$. This implies that there exists an integer $1 \leq i \leq n$ with $x \leq 2 \, |a_i|^{1/i}$. This is again equivalent with $x \leq \max_{i=1}^{n} |a_i|^{1/i}$ which is what we wanted to prove. □

We now turn to the proof of Lemma A.14.

Prood of Lemma A.14. Since p is an even integer then,

$$\begin{aligned} \left\| \sum_{i \in [n]} a_i X_i \right\|_p^p &= \sum_{i \in [n]} \sum_{s=1}^{p} \binom{p-1}{s-1} \mathrm{E}\left[(a_i X_i)^s \left(\sum_{j \in [n] \setminus \{i\}} a_j X_j \right)^{p-s} \right] \\ &= \sum_{i \in [n]} \sum_{s=1}^{p} \binom{p-1}{s-1} \mathrm{E}[(a_i X_i)^s] \, \mathrm{E}\left[\left(\sum_{j \in [n] \setminus \{i\}} a_j X_j \right)^{p-s} \right] \ . \end{aligned}$$

The first step follows by noticing that when we expand $(\sum_{i \in [n]} a_i X_i)^p$ then for each term the first factor will give an $i \in [n]$. Now if i has multiplicity $s \geq 1$ in the term then there are $\binom{p-1}{s-1}$ ways to choose the other factors for i.

We note that since the variables are symmetric then $\mathrm{E}[(a_i X_i)^s] = 0$ for s odd. So in the following, we assume that s is even, which implies that $p - s$ is even. Now we use Jensen's inequality to obtain,

$$\mathrm{E}\left[\left(\sum_{j\in[n]\setminus\{i\}} a_j X_j\right)^{p-s}\right] = \mathrm{E}\left[\left(\sum_{j\in[n]\setminus\{i\}} a_j X_j + a_i \mathrm{E}[X_i]\right)^{p-s}\right] \leq \mathrm{E}\left[\left(\sum_{j\in[n]} a_j X_j\right)^{p-s}\right] .$$

Another usage of Jensen's inequality gives us that

$$\mathrm{E}\left[\left(\sum_{j\in[n]} a_j X_j\right)^{p-s}\right] = \left\|\sum_{j\in[n]} a_j X_j\right\|_{p-s}^{p-s} \leq \left\|\sum_{j\in[n]} a_j X_j\right\|_p^{p-s} .$$

Combining these we get that

$$\begin{aligned}\left\|\sum_{i\in[n]} a_i X_i\right\|_p^p &\leq \sum_{i\in[n]}\sum_{s=2}^{p}\binom{p-1}{s-1}\mathrm{E}[(a_i X_i)^s]\left\|\sum_{j\in[n]} a_j X_j\right\|_p^{p-s} \\ &= \sum_{s=2}^{p}\binom{p-1}{s-1}\sum_{i\in[n]} a_i^s \mathrm{E}[X_i^s]\left\|\sum_{j\in[n]} a_j X_j\right\|_p^{p-s} \\ &= \sum_{s=2}^{p}\binom{p-1}{s-1}\mathrm{E}[X^s]\left\|\sum_{j\in[n]} a_j X_j\right\|_p^{p-s}\sum_{i\in[n]} a_i^s .\end{aligned}$$

Now using Lemma A.34 we get that

$$\left\|\sum_{i\in[n]} a_i X_i\right\|_p \leq \sup\left\{2\binom{p-1}{s-1}^{1/s}\left(\sum_{i\in[n]} a_i^s\right)^{1/s}\|X\|_s \,\middle|\, 2\leq s\leq p\right\} .$$

Now using Stirling's approximation, we get that $\binom{p-1}{s-1} = \binom{p}{s}\frac{s}{p} \leq \left(\frac{ep}{s}\right)^s \frac{s}{p}$. Plugging this estimate into our equation gives us that

$$\begin{aligned}\left\|\sum_{i\in[n]} a_i X_i\right\|_p &\leq \sup\left\{2\frac{ep}{s}\frac{s^{1/s}}{p^{1/s}}\left(\sum_{i\in[n]} a_i^s\right)^{1/s}\|X\|_s \,\middle|\, 2\leq s\leq p\right\} \\ &\leq 4e\sup\left\{\frac{p}{s}\left(\frac{\sum_{i\in[n]} a_i^s}{p}\right)^{1/s}\|X\|_s \,\middle|\, 2\leq s\leq p\right\} .\end{aligned}$$

□

We will not be using the result directly instead we will use the following corollary where we further simplify the expression by bounding only in terms of the largest weight and the Euclidean norm.

Corollary A.35. *Let $(X_i)_{i\in[n]}$ be independent and identically distributed symmetric random variables, and let $(a_i)_{i\in[n]}$ be a sequence of reals. If $p \geq 2$ is an even integer then,*

$$\left\| \sum_{i\in[n]} a_i X_i \right\|_p \leq K \max_{i\in[n]} |a_i| \sup \left\{ \frac{p}{s} \left(\frac{\sum_{i\in[n]} a_i^2}{p \max_{i\in[n]} |a_i|^2} \right)^{1/s} \|X\|_s \,\middle|\, 2 \leq s \leq p \right\},$$

where $K = 4e$ is a universal constant.

Proof. This follows from Lemma A.14 and the fact that $a_i^s \leq a_i^2 \max_{i\in[n]} |a_i|^{s-2}$ for all $i \in [n], s \geq 2$.

$$\begin{aligned}
\left\| \sum_{i\in[n]} a_i X_i \right\|_p &\leq K \sup \left\{ \frac{p}{s} \left(\frac{\sum_{i\in[n]} a_i^s}{p} \right)^{1/s} \|X\|_s \,\middle|\, 2 \leq s \leq p \right\} \\
&\leq K \sup \left\{ \frac{p}{s} \left(\frac{\sum_{i\in[n]} a_i^2 \max_{i\in[n]} |a_i|^{s-2}}{p} \right)^{1/s} \|X\|_s \,\middle|\, 2 \leq s \leq p \right\} \\
&\leq K \max_{i\in[n]} |a_i| \sup \left\{ \frac{p}{s} \left(\frac{\sum_{i\in[n]} a_i^2}{p \max_{i\in[n]} |a_i|^2} \right)^{1/s} \|X\|_s \,\middle|\, 2 \leq s \leq p \right\}.
\end{aligned}$$

□

We will now use Corollary A.35 to bound the sum of different types of random variables with Ψ_p-function. We start by looking at Bernoulli-Rademacher variables.

Lemma A.36. *Let $(X_i)_{i\in[n]}$ be independent Bernoulli-Rademacher variables with parameter α, that is, $\Pr[X_i = 1] = \Pr[X_i = -1] = \frac{1}{2} - \Pr[X_i = 0] = \frac{\alpha}{2}$, and let $(a_i)_{i\in[n]}$ be a sequence of reals.*

If $p \geq 2$ is an even integer then,

$$\left\| \sum_{i\in[n]} a_i X_i \right\|_p \leq 4e\Psi_p \left(\max_{i\in[n]} |a_i|, \alpha \sum_{i\in[n]} a_i^2 \right).$$

And if $p \geq 2$ is a real number then,

$$\left\| \sum_{i\in[n]} a_i X_i \right\|_p \leq 8e\Psi_p \left(\max_{i\in[n]} |a_i|, \alpha \sum_{i\in[n]} a_i^2 \right).$$

Proof. We note that $\|X\|_s = \alpha^{1/s}$ for all $s \geq 2$. Let $p \geq 2$ be an even integer then using Corollary A.35 we then get that

$$\begin{aligned}\left\|\sum_{i\in[n]} a_i X_i\right\|_p &\leq 4e \max_{i\in[n]} |a_i| \sup\left\{\frac{p}{s}\left(\frac{\sum_{i\in[n]} a_i^2}{p \max_{i\in[n]} |a_i|^2}\right)^{1/s} \alpha^{1/s} \,\middle|\, 2 \leq s \leq p\right\} \\ &= 4e \max_{i\in[n]} |a_i| \sup\left\{\frac{p}{s}\left(\frac{\alpha \sum_{i\in[n]} a_i^2}{p \max_{i\in[n]} |a_i|^2}\right)^{1/s} \,\middle|\, 2 \leq s \leq p\right\}.\end{aligned}$$

Now eq. (A.17) proves the first claim. By Hölder's inequality we have that $p \mapsto \left\|\sum_{i\in[n]} a_i X_i\right\|_{1/p}$ is log-convex and Jensen's inequality implies that $p \mapsto \left\|\sum_{i\in[n]} a_i X_i\right\|_p$ is increasing, thus eq. (A.23) proves the second claim. □

We are now almost ready to prove Theorem A.5 for fully random hash functions. This will be a principal lemma in the sequel when we prove concentration results for tabulation hashing. But first, we need to prove a symmetrization lemma for fully random functions.

Lemma A.37. *Let $h\colon U \to [m]$ be a uniformly random function, let $v\colon U \times [m] \to \mathbb{R}$ be a fixed value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in U$. Let $\varepsilon\colon U \to \{-1,1\}$ be a uniformly random sign function. Define the random variable $X_v = \sum_{x\in U} v(x,h(x))$. Then for all $p \geq 2$,*

$$2^{-1}\left\|\sum_{x\in U} \varepsilon(x) v(x,h(x))\right\|_p \leq \left\|\sum_{x\in U} v(x,h(x))\right\|_p \leq 2\left\|\sum_{x\in U} \varepsilon(x) v(x,h(x))\right\|_p.$$

Proof. We first consider the lower bound. Fixing ε and using the triangle inequality we get that

$$\begin{aligned}\left\|\sum_{x\in U} \varepsilon(x) v(x,h(x)) \,\middle|\, \varepsilon\right\|_p &= \left\|\sum_{x\in U} [\varepsilon(x)=1] v(x,h(x)) - \sum_{x\in U} [\varepsilon(x)=-1] v(x,h(x)) \,\middle|\, \varepsilon\right\|_p \\ &\leq \left\|\sum_{x\in U} [\varepsilon(x)=1] v(x,h(x)) \,\middle|\, \varepsilon\right\|_p + \left\|\sum_{x\in U} [\varepsilon(x)=-1] v(x,h(x)) \,\middle|\, \varepsilon\right\|_p.\end{aligned}$$

Now we use Jensen's inequality on each of the terms.

$$\begin{aligned}&\left\|\sum_{x\in U} [\varepsilon(x)=1] v(x,h(x)) \,\middle|\, \varepsilon\right\|_p \\ &\qquad = \left\|\sum_{x\in U} [\varepsilon(x)=1] v(x,h(x)) + \sum_{x\in U} [\varepsilon(x)=-1] \operatorname{E}[v(x,h(x))] \,\middle|\, \varepsilon\right\|_p \\ &\qquad \leq \left\|\sum_{x\in U} v(x,h(x)) \,\middle|\, \varepsilon\right\|_p.\end{aligned}$$

Unfixing ε we get that

$$\left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p \leq 2 \left\| \sum_{x \in U} v(x, h(x)) \right\|_p ,$$

which establishes the lower bound.

For the upper bound, we first define $h' \colon U \to [m]$ to be an independent copy of h. We then use Jensen's inequality to get that

$$\begin{aligned} \left\| \sum_{x \in U} v(x, h(x)) \right\|_p &= \left\| \sum_{x \in U} v(x, h(x)) - \sum_{x \in U} \mathrm{E}\big[v(x, h'(x))\big] \right\|_p \\ &\leq \left\| \sum_{x \in U} \big(v(x, h(x)) - v(x, h'(x))\big) \right\|_p . \end{aligned}$$

We then note that due to the independence $v(x, h(x)) - v(x, h'(x))$ is a symmetric variable, thus it has the same distribution as $\varepsilon(x)\left(v(x, h(x)) - v(x, h'(x))\right)$. We use this and the triangle inequality to finish the upper bound,

$$\begin{aligned} \left\| \sum_{x \in U} \big(v(x, h(x)) - v(x, h'(x))\big) \right\|_p &= \left\| \sum_{x \in U} \varepsilon(x) \big(v(x, h(x)) - v(x, h'(x))\big) \right\|_p \\ &\leq 2 \left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p , \end{aligned}$$

which establishes the upper bound. □

Theorem A.5. *Let $h \colon U \to [m]$ be a uniformly random function, let $v \colon U \times [m] \to \mathbb{R}$ be a fixed value function, and assume that $\sum_{j \in [m]} v(x, j) = 0$ for all keys $x \in U$. Define the random variable $X_v = \sum_{x \in U} v(x, h(x))$. Then for all $p \geq 2$*

$$\|X_v\|_p \leq L \Psi_p\left(M_v, \sigma_v^2\right) ,$$

where $L \leq 16e$ is a universal constant.

Proof. We start by using Lemma A.37 to get that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_p \leq 2 \left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p .$$

Let $(Y_x^{(j)})_{x \in U, j \in [m]}$ be independent Bernoulli-Rademacher variables with parameter $\frac{1}{m}$. The idea of the proof is to show that for $p \geq 2$ then,

$$\left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p \leq \left\| \sum_{x \in U, j \in [m]} v(x, j) Y_x^{(j)} \right\|_p .$$

This is nontrivial to do for general p. Instead, we will focus on p being an even integer.

Let $q \geq 2$ be an even integer. Then for all $x \in U$ we have that

$$\|\varepsilon(x)v(x,h(x))\|_q = \left(\frac{\sum_{j\in[m]} v(x,j)^q}{m}\right)^{1/q}.$$

But it is easy to check that

$$\left\|\sum_{j\in[m]} v(x,j)Y_x^{(j)}\right\|_q \geq \left(\frac{\sum_{j\in[m]} v(x,j)^q}{m}\right)^{1/q}.$$

We can now show what we want,

$$\begin{aligned}
\mathrm{E}\left[\left(\sum_{x\in U}\varepsilon(x)v(x,h(x))\right)^p\right] &= \sum_{\sum_{x\in U} q_x = p} \binom{p}{(q_x)_{x\in U}} \prod_{x\in U} \mathrm{E}[(\varepsilon(x)v(x,h(x)))^{q_x}] \\
&= \sum_{\substack{\sum_{x\in U} q_x = p \\ \forall x\in U: q_x \text{ is even}}} \binom{p}{(q_x)_{x\in U}} \prod_{x\in U} \mathrm{E}[(\varepsilon(x)v(x,h(x)))^{q_x}] \\
&\leq \sum_{\substack{\sum_{x\in U} q_x = p \\ \forall x\in U: q_x \text{ is even}}} \binom{p}{(q_x)_{x\in U}} \prod_{x\in U} \mathrm{E}\left[\left(\sum_{j\in[m]} v(x,j)Y_x^{(j)}\right)^{q_x}\right] \\
&= \sum_{\sum_{x\in U} q_x = p} \binom{p}{(q_x)_{x\in U}} \prod_{x\in U} \mathrm{E}\left[\left(\sum_{j\in[m]} v(x,j)Y_x^{(j)}\right)^{q_x}\right] \\
&= \mathrm{E}\left[\left(\sum_{x\in U, j\in[m]} v(x,j)Y_x^{(j)}\right)^p\right]
\end{aligned}$$

Now using Lemma A.36 we get that

$$\begin{aligned}
\left\|\sum_{x\in U}\varepsilon(x)v(x,h(x))\right\|_p &\leq \left\|\sum_{x\in U, j\in[m]} v(x,j)Y_x^{(j)}\right\|_p \\
&\leq 4e\Psi_p\left(\max_{x\in U, j\in[m]} |v(x,j)|, \frac{\sum_{x\in U, j\in[m]} v(x,j)^2}{m}\right).
\end{aligned}$$

for all even integers $p \geq 2$. Now we use eq. (A.23) as in the proof of Lemma A.36 which proves the second claim. □

We also need the standard fact that a sum of independent sub-Gaussian is also sub-Gaussian. We include a proof for completeness.

Lemma A.38. *Let $(X_i)_{i\in[n]}$ be a sequence of independent symmetric random variables. Let $p \geq 2$ and assume that there exists a sequence of real numbers $(a_i)_{i\in[n]}$ such that for all even integers $2 \leq q \leq p$ and all $i \in [n]$ it holds that*

$$\|X_i\|_q \leq \sqrt{q} a_i \, .$$

Then the sum of the random variables satisfies,

$$\left\| \sum_{i\in[n]} X_i \right\|_p \leq \sqrt{p} \sqrt{2e \sum_{i\in[n]} a_i^2} \, .$$

Proof. The main idea of the proof is to compare the random variables $(X_i)_{i\in[n]}$ with a sequence of independent Gaussian $(g_i)_{i\in[n]}$, and then exploit that the sum of Gaussian variables is a Gaussian variable. We will use the standard fact that for all even integers $2 \leq q$, Gaussian variables satisfies $\|g_i\|_q = ((q-1)!!)^{1/q}$. A simple lower bound for this follows by using Stirling's approximation,

$$((q-1)!!)^{1/q} \geq ((q/2)!)^{1/q} \geq \left(\left(\frac{q}{2e} \right)^{q/2} \right)^{1/q} = \sqrt{\frac{q}{2e}} \, .$$

For an upper bound we note that by the AM-GM inequality we have that $(q-2i-1)(2i+1) \leq \left(\frac{q}{2}\right)^2$ so we get that

$$((q-1)!!)^{1/q} \leq \left(\left(\frac{q}{2} \right)^{q/2} \right)^{1/q} = \sqrt{\frac{q}{2}} \, .$$

Now the lower bound gives us the estimate,

$$\|X_i\|_q \leq \sqrt{q} a_i \leq \sqrt{2e} \, \|g_i\|_q \, .$$

We start by proving the case where $p \geq 2$ is an even integer. We then get that

$$\begin{aligned}
\mathrm{E}\left[\left(\sum_{i\in[n]} X_i \right)^p \right] &= \sum_{\sum_{i\in[n]} q_i = p} \binom{p}{(q_i)_{i\in[n]}} \prod_{i\in[n]} \mathrm{E}[(X_i)^{q_i}] \\
&= \sum_{\substack{\sum_{i\in[n]} q_i = p \\ \forall i \in [n] : q_i \text{ is even}}} \binom{p}{(q_i)_{i\in[n]}} \prod_{i\in[n]} \mathrm{E}[(X_i)^{q_i}] \\
&\leq \sum_{\substack{\sum_{i\in[n]} q_i = p \\ \forall i \in [n] : q_i \text{ is even}}} \binom{p}{(q_i)_{i\in[n]}} \prod_{i\in[n]} \mathrm{E}\Big[\Big(\sqrt{2e} a_i g_i \Big)^{q_i} \Big] \\
&= \sum_{\sum_{i\in[n]} q_i = p} \binom{p}{(q_i)_{i\in[n]}} \prod_{i\in[n]} \mathrm{E}\Big[\Big(\sqrt{2e} a_i g_i \Big)^{q_i} \Big] \\
&= \mathrm{E}\left[\left(\sum_{i\in[n]} \sqrt{2e} a_i g_i \right)^p \right] \, .
\end{aligned}$$

Now we use that if g, g', and g'' are independent standard Gaussian variables then $ag+bg'$ is distributed as $\sqrt{a^2+b^2}g''$. This give us that

$$\left\|\sum_{i\in[n]} X_i\right\|_p \leq \left\|\sum_{i\in[n]} \sqrt{2e}a_i g_i\right\|_p = \left\|\sqrt{\sum_{i\in[n]} 2ea_i^2}g\right\|_p \leq \sqrt{p}\sqrt{e\sum_{i\in[n]} a_i^2}\,.$$

If $p \geq 2$ is not an even integer then let $p' \geq p$ be the smallest even integer larger than p. We note that $p' \leq 2p$ and using Jensen's inequality we get that

$$\left\|\sum_{i\in[n]} X_i\right\|_p \leq \left\|\sum_{i\in[n]} X_i\right\|_{p'} \leq \sqrt{p}\sqrt{e\sum_{i\in[n]} a_i^2} \leq \sqrt{p}\sqrt{2e\sum_{i\in[n]} a_i^2}\,.$$

This finishes the proof. □

We can use this lemma to prove another useful bound for uniformly random functions.

Lemma A.39. *Let $h\colon U \to [m]$ be a uniformly random function, let $\varepsilon\colon U \to \{-1,1\}$ be a uniformly random sign function, and let $v : U \times [m] \to \mathbb{R}$ be a fixed value function. Then for all $p \geq 2$,*

$$\left\|\sum_{x\in U} \varepsilon(s)v(x,h(x))\right\|_p \leq L\sqrt{\frac{p}{\log\left(\frac{e^2 m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)}\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}\,,$$

where $L \leq e$ is a universal constant.

Before we prove Lemma A.39 we need the following technical lemma.

Lemma A.40. *Let $(a_i)_{i\in[n]}$ and $(b_i)_{i\in[n]}$ be two sequences of positive integers. If $\frac{a_i}{b_i} \geq 1$ for all $i \in [n]$ then,*

$$\sum_{i\in[n]} \frac{a_i}{\log\left(\frac{e^2 a_i}{b_i}\right)} \leq \frac{\sum_{i\in[n]} a_i}{\log\left(\frac{e^2\sum_{i\in[n]} a_i}{\sum_{i\in[n]} b_i}\right)}\,. \tag{A.25}$$

Proof. We define the sequence $(r_i)_{i\in[n]}$ by $r_i = \frac{a_i}{b_i}$ for all $i \in [n]$, define the random variable R by $\Pr[R = r_i] = \frac{b_i}{\sum_{j\in[n]} b_i}$, and the function $f\colon \mathbb{R}_+ \to \mathbb{R}_+$ by $f(r) = \frac{r}{\log(e^2 r)}$. Now we note that

$$\sum_{i\in[n]} \frac{a_i}{\log\left(\frac{e^2 a_i}{b_i}\right)} = \sum_{i\in[n]} b_i r_i \log\left(e^2 r_i\right) = \sum_{i\in[n]} b_i f(r_i) = \left(\sum_{j\in[n]} b_i\right)\mathrm{E}[f(R)]\,,$$

$$\frac{\sum_{i\in[n]} a_i}{\log\left(\frac{e^2\sum_{i\in[n]} a_i}{\sum_{i\in[n]} b_i}\right)} = \frac{\sum_{i\in[n]} b_i r_i}{\log\left(\frac{e^2\sum_{i\in[n]} b_i r_i}{\sum_{i\in[n]} b_i}\right)} = \left(\sum_{j\in[n]} b_i\right) f(\mathrm{E}[R])\,.$$

Thus we get that eq. (A.25) is equivalent with showing that $\mathrm{E}[f(R)] \leq f(\mathrm{E}[R])$. It easy to check that f is concave on the interval $[1, \infty)$ and since $R \geq \min_{i\in[n]} r_i = \min_{i\in[n]} \frac{a_i}{b_i} \geq 1$, Jensen's inequality implies the result. □

Proof of Lemma A.39. Let $x \in U$ be fixed and consider $2 \leq q \leq p$. We then have that

$$\begin{aligned}\|\varepsilon(x)v(x,h(x))\|_q &= \left(\frac{\sum_{j\in[m]} |v(x,h(x))|^q}{m}\right)^{1/q} \\ &\leq \left(\frac{\sum_{j\in[m]} v(x,h(x))^2}{m \max_{j\in[m]} v(x,h(x))^2}\right)^{1/q} \max_{j\in[m]} |v(x,h(x))| \\ &= \left(\frac{\|v[x]\|_2^2}{m\,\|v[x]\|_\infty^2}\right)^{1/q} \|v[x]\|_\infty \ .\end{aligned}$$

Now a simple estimate give us that $y^{1/q} \leq e(y/e^2)^{1/q} \leq e\sqrt{\frac{q}{2e\log\frac{e^2}{y}}} = \sqrt{\frac{eq}{2\log\frac{e^2}{y}}}$ for all $y \leq 1$ and all $q \geq 2$. Clearly, $\frac{\|v[x]\|_2^2}{m\|v[x]\|_\infty^2} \leq 1$, hence we get that

$$\|\varepsilon(x)v(x,h(x))\|_q \leq \sqrt{\frac{eq}{2\log\left(\frac{e^2 m\|v[x]\|_\infty^2}{\|v[x]\|_2^2}\right)}}\, \|v[x]\|_\infty \ .$$

This shows that $\varepsilon(x)v(x,h(x))$ is sub-Gaussian hence we can use Lemma A.38 to get that

$$\left\|\sum_{x\in U} \varepsilon(s)v(x,h(x))\right\|_p \leq e\sqrt{p}\sqrt{\sum_{x\in U} \frac{\|v[x]\|_\infty^2}{\log\left(\frac{e^2 m\|v[x]\|_\infty^2}{\|v[x]\|_2^2}\right)}} \ .$$

Now an application of Lemma A.40 finishes the proof. □

We end the section by bounding the simple case of weighted sums of Rademacher variables. We will need the lemma later and it is known as Khintchine's inequality. For completeness we include a proof the lemma.

Lemma A.41 (Khintchine's inequality)**.** *Let $(\varepsilon_i)_{i\in[n]}$ be a sequence of independent Rademacher variables, and let $(a_i)_{i\in[n]}$ be a sequence of real numbers. For all $p \geq 2$ we have that*

$$\left\|\sum_{i\in[n]} a_i\varepsilon_i\right\|_p \leq \sqrt{p}\sqrt{e\sum_{i\in[n]} a_i^2} \ .$$

Proof. We note that for all $i \in [n]$ and all $q \geq 2$ we have that $\|a_i\varepsilon_i\|_q = |a_i| \leq \frac{\sqrt{q}}{\sqrt{2}}|a_i|$. We now use Lemma A.38 to get that

$$\left\|\sum_{i\in[n]} a_i\varepsilon_i\right\|_p \leq \sqrt{p}\sqrt{2e\sum_{i\in[n]} \tfrac{a_i^2}{2}} = \sqrt{p}\sqrt{e\sum_{i\in[n]} a_i^2} \ .$$

This finishes the proof. □

A.3.3 Moments of Functions of Random Variables

The goal of this section is to prove Lemma A.15.

Lemma A.42. *Let $f\colon \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a non-negative function which is monotonically increasing in every argument, and assume that there exist positive reals $(\alpha_i)_{i\in[n]}$ and $(t_i)_{i\in[n]}$ such that for all $\lambda \geq 0$*

$$f(\lambda^{\alpha_0} t_0, \ldots, \lambda^{\alpha_{n-1}} t_{n-1}) \leq \lambda f(t_0, \ldots, t_{n-1}) \,.$$

Let $(X_i)_{i\in[n]}$ be non-negative random variables. Then for all $p \geq 1$ we have that

$$\|f(X_0, \ldots, X_{n-1})\|_p \leq n^{1/p} \max_{i\in[n]} \left(\frac{\|X_i\|_{p/\alpha_i}}{t_i} \right)^{1/\alpha_i} f(t_0, \ldots, t_{n-1}) \,.$$

Proof. We define $\lambda = \max_{i\in[n]} \left(\frac{X_i}{t_i}\right)^{1/\alpha_i}$ and note that $X_i \leq \lambda^{\alpha_i} t_i$ for all $i \in [n]$. Since f is an increasing function then $f(X_0, \ldots, X_{n-1}) \leq f(\lambda^{\alpha_0} t_0, \ldots, \lambda^{\alpha_{n-1}} t_{n-1})$. We can then use the condition on f to get that

$$\|f(X_0, \ldots, X_{n-1})\|_p \leq \|f(\lambda^{\alpha_0} t_0, \ldots, \lambda^{\alpha_{n-1}} t_{n-1})\|_p \leq \|\lambda\|_p \, f(t_0, \ldots, t_{n-1}) \,.$$

Now we just need to prove that $\|\lambda\|_p \leq n^{1/p} \max_{i\in[n]} \left(\frac{\|X_i\|_{p/\alpha_i}}{t_i} \right)^{1/\alpha_i}$. We note that $\lambda = \max_{i\in[n]} \left(\frac{X_i}{t_i}\right)^{1/\alpha_i} \leq \left(\sum_{i\in[n]} \left(\frac{X_i}{t_i}\right)^{p/\alpha_i} \right)^{1/p}$. Hence we get that

$$\begin{aligned}
\|\lambda\|_p &\leq \left(\sum_{i\in[n]} \mathrm{E}\left[\left(\frac{X_i}{t_i} \right)^{p/\alpha_i} \right] \right)^{1/p} \\
&\leq \left(n \max_{i\in[n]} \mathrm{E}\left[\left(\frac{X_i}{t_i} \right)^{p/\alpha_i} \right] \right)^{1/p} \\
&= n^{1/p} \max_{i\in[n]} \left(\frac{\|X_i\|_{p/\alpha_i}}{t_i} \right)^{1/\alpha_i} ,
\end{aligned}$$

which finishes the proof. □

A.3.4 Decoupling of Adapted Sequences

In the paper, we will need to analyse sums of martingale differences which are not independent. This poses a problem because the lemmas of the previous section assumes that random variables are independent. We will handle this issue by using a powerful result of Hitczenko [Hit94] to reduce the sums of martingale differences to a sum of independent variables. Before the theorem, we need a bit of notation.

Definition A.42. Let $(X_i)_{i\in[n]}$ and $(Y_i)_{i\in[n]}$ be two sequences of random variables adapted to a filtration $(\mathcal{F}_i)_{i\in[n]}$. Then $(X_i)_{i\in[n]}$ and $(Y_i)_{i\in[n]}$ are *tangent* with respect to $(\mathcal{F}_i)_{i\in[n]}$ if $(X_i \mid \mathcal{F}_{i-1})$ and $(Y_i \mid \mathcal{F}_{i-1})$ has the same distribution for all $i \in [n]$.

Definition A.43. Let $(X_i)_{i\in[n]}$ be a sequence of random variables adapted to a filtration $(\mathcal{F}_i)_{i\in[n]}$ and let $\mathcal{G} \subseteq \mathcal{F}_{n-1}$ be a σ-algebra. Then $(X_i)_{i\in[n]}$ satisfies the *conditional independence condition* with respect to $\mathcal{G}$ if $(X_i \mid \mathcal{F}_{i-1})$ and $(X_i \mid \mathcal{G})$ have the same distribution for all $i \in [n]$, and $(X_i)_{i\in[n]}$ are conditionally independent given $\mathcal{G}$.

Definition A.44. Let $(X_i)_{i\in[n]}$ and $(Y_i)_{i\in[n]}$ be two sequences of random variables which are tangent with respect to the filtration $(\mathcal{F}_i)_{i\in[n]}$. Let $\mathcal{G} \subseteq \mathcal{F}_{n-1}$ be a σ-algebra. If $(Y_i)_{i\in[n]}$ satisfies the conditional independence condition with respect to $\mathcal{G}$ then we say that $(Y_i)_{i\in[n]}$ is a *decoupled sequence tangent* to $(X_i)_{i\in[n]}$.

We can now state the theorem of Hitczenko [Hit94].

Theorem A.45 (Hitczenko [Hit94])**.** *There exists a universal constant $0 < M < \infty$ such that, for all $p \geq 1$ and all sequences of random variables $(X_i)_{i\in[n]}$ and $(Y_i)_{i\in[n]}$ where $(Y_i)_{i\in[n]}$ is a decoupled sequence tangent to $(X_i)_{i\in[n]}$, then*

$$\left\| \sum_{i\in[n]} X_i \right\|_p \leq M \left\| \sum_{i\in[n]} Y_i \right\|_p .$$

Instead of using the result directly, we will instead use the following consequence of the theorem.

Lemma A.46. *Let $(X_i, \mathcal{F}_i)_{i\in[n]}$ be a filtered sequence. Assume there exists a sequence of random variables $(Y_i)_{i\in[n]}$ satisfying the following:*

1. *$(X_i \mid \mathcal{F}_{i-1})$ and $(Y_i \mid \mathcal{F}_{i-1})$ have the same distribution for every $i \in [n]$.*
2. *The sequence $(Y_i)_{i\in[n]}$ is conditionally independent given $\mathcal{F}_{n-1}$.*
3. *$(Y_i \mid \mathcal{F}_{i-1})$ and $(Y_i \mid \mathcal{F}_{n-1})$ have the same distribution for every $i \in [n]$.*
4. *$(X_i \mid \mathcal{F}_{i-1})$ and $(X_i \mid \sigma(\mathcal{F}_{i-1}, (Y_j)_{j\in[i+1]})$ have the same distribution for every $i \in [n]$.*

Then for all $p \geq 1$,

$$\left\| \sum_{i\in[n]} X_i \right\|_p \leq M \left\| \sum_{i\in[n]} Y_i \right\|_p ,$$

where M is a universal constant.

Proof. We define the filtration $(\mathcal{H}_i)_{i\in[n]}$ by $\mathcal{H}_i = \sigma(\mathcal{F}_i, (Y_j)_{j\in[i+1]})$ for $i \in [n]$. We then clearly have that $(X_i)_{i\in[n]}$ and $(Y_i)_{i\in[n]}$ are adapted to $(\mathcal{H}_i)_{i\in[n]}$. We will also see that they are tangent. Let $A \subseteq \mathbb{R}$ then,

$$\begin{aligned}
\Pr[Y_i \in A \,|\, \mathcal{H}_{i-1}] &= \mathrm{E}[[Y_i \in A] \,|\, \mathcal{H}_{i-1}] \\
&= \mathrm{E}\Big[\mathrm{E}\big[[Y_i \in A] \,\big|\, \sigma(\mathcal{F}_{n-1}, (Y_j)_{j\in[i]})\big] \,\Big|\, \mathcal{H}_{i-1}\Big] \\
&= \mathrm{E}\Big[\mathrm{E}[[Y_i \in A] \,|\, \mathcal{F}_{n-1}] \,\Big|\, \mathcal{H}_{i-1}\Big] \\
&= \mathrm{E}\Big[\mathrm{E}[[Y_i \in A] \,|\, \mathcal{F}_{i-1}] \,\Big|\, \mathcal{H}_{i-1}\Big] \\
&= \mathrm{E}\Big[\mathrm{E}[[X_i \in A] \,|\, \mathcal{F}_{i-1}] \,\Big|\, \mathcal{H}_{i-1}\Big] \\
&= \Pr[X_i \in A \,|\, \mathcal{F}_{i-1}] \\
&= \Pr[X_i \in A \,|\, \mathcal{H}_{i-1}]
\end{aligned}$$

The first equality uses the power property of conditional expectation, the second equality uses the conditional independence property, the next two equalities follow by the equivalences of distributions, the second last equality follows by $\mathcal{F}_{i-1} \subseteq \mathcal{H}_{i-1}$, and the last equality follows by the equivalences of distributions.

We have that $(Y_i)_{i\in[n]}$ are conditionally independent given $\mathcal{F}_{n-1} \in \mathcal{H}_{n-1}$, hence $(Y_i)_{i\in[n]}$ is a decoupled sequence tangent to $(X_i)_{i\in[n]}$. Now Theorem A.45 give us the result. □

A.4 Strong Concentration for Tabulation Hashing

The goal of this chapter is to prove strong concentration results for tabulation based hashing. The chapter is divided into three parts: In the first part we generalize some of the results by Aamand. et al. [AKKR+20] to the case where we have partial keys, and we prove some auxiliary results which will be used in the later parts. In the second part we improve the analysis of simple tabulation and provide a moment bound which holds for all moments. In order to prove this result we first have to bound a technical quantity which will show up as a conditional variance in the proof. Finally, in the last part we prove moment bounds for mixed tabulation. They can be thought versions of Khintchine's inequality and Chernoff bound for mixed tabulation.

One of the main insights we use that differs from the previous analyses is that we work with symmetrized versions of simple tabulation and mixed tabulation. We will in their respective section argue that this assumption is valid.

We need the following simple lemma that compares the growth rate of powers and logarithms. This will be used extensively.

Lemma A.47. *Let $a > 0$ and $b > 0$ be positive reals. It then holds that for all $x > 1$,*

$$\frac{x^a}{\log(x)^b} \geq \left(\frac{e}{b/a}\right)^b.$$

Proof. We write $\frac{x^a}{\log(x)^b} = \left(\frac{x^{a/b}}{\log(x)}\right)^b$ so we just need to minimize $\frac{x^{a/b}}{\log(x)}$. Taking the derivative we get,

$$\frac{d}{dx}\frac{x^{a/b}}{\log(x)} = \frac{\frac{a}{b}x^{a/b-1}\log(x) - x^{a/b-1}}{\log(x)} .$$

From this it is clear that $\frac{x^{a/b}}{\log(x)}$ is minimized at $\hat{x} = e^{b/a}$. We then get that

$$\left(\frac{x^{a/b}}{\log(x)}\right)^b \geq \left(\frac{\hat{x}^{a/b}}{\log(\hat{x})}\right)^b = \left(\frac{e}{b/a}\right)^b .$$

□

A.4.1 Improved Analysis for Simple Tabulation

We start the section by arguing why we can assume that the simple tabulation functions are symmetrized.

Lemma A.48. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation function, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Then for every $p \geq 2$,*

$$2^{-c}\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p \leq \left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \leq 2^c\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p ,$$

where $\varepsilon\colon \Sigma^c \to \{-1,1\}$ a simple tabulation sign function.

Proof. We will prove the result by induction on c. The case $c = 1$ corresponds to Lemma A.37.

Now assume that $c > 1$ and that the results is true for values less than c. We define the σ-algebra $\mathcal{G} = \sigma((T(c-1,\alpha))_{\alpha\in\Sigma})$. Fix $\mathcal{G}$ and define $v'\colon \Sigma^{c-1} \times [m] \to \mathbb{R}$ by

$$v'(x,j) = \sum_{\alpha\in\Sigma} v(x \cup \{(c-1,\alpha)\}, T(c-1,\alpha) \oplus j) .$$

Clearly, we have that $v'(x,j)$ is $\mathcal{G}$-measurable for $x \in \Sigma^{c-1}, j \in [m]$ and $\mathrm{E}[v'(x,h(x) \mid \mathcal{G}] = 0$ for all $x \in \Sigma^{c-1}$. We fix $\mathcal{G}$ and use the induction hypothesis to get that

$$\begin{aligned} 2^{-(c-1)}\left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v'(x,h(x)) \;\middle|\; \mathcal{G}\right\|_p &\leq \left\|\sum_{x\in\Sigma^{c-1}} v'(x,h(x)) \;\middle|\; \mathcal{G}\right\|_p \\ &\leq 2^{c-1}\left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v'(x,h(x)) \;\middle|\; \mathcal{G}\right\|_p . \end{aligned}$$

So if we unfix $\mathcal{G}$ then we have that

$$2^{-(c-1)}\left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v'(x,h(x))\right\|_p \le \left\|\sum_{x\in\Sigma^{c-1}}v'(x,h(x))\right\|_p \le 2^{c-1}\left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v'(x,h(x))\right\|_p . \tag{A.26}$$

Now we define the σ-algebra $\mathcal{H} = \sigma\big((h(\{(j,\alpha)\}),\varepsilon(\{(j,\alpha)\}))_{j\in[c-1],\alpha\in\Sigma}\big)$, and define $v'' \colon \Sigma\times[m]\to\mathbb{R}$ by $v''(\alpha,j) = \sum_{x\in\Sigma^{c-1}}\varepsilon(x)v(x\cup\{(c-1,\alpha)\},h(x)\oplus j)$. We then get that

$$\begin{aligned}\left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v'(x,h(x))\right\|_p &= \left\|\sum_{x\in\Sigma^{c-1}}\varepsilon(x)\sum_{\alpha\in\Sigma}v(x\cup\{(c-1,\alpha)\},T(c-1,\alpha)\oplus h(x))\right\|_p \\ &= \left\|\sum_{\alpha\in\Sigma}v''(\alpha,T(c-1,\alpha))\right\|_p .\end{aligned}$$

We fix $\mathcal{H}$ and use Lemma A.37 to get that

$$\begin{aligned}&2^{-1}\left\|\sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})v''(\alpha,T(c-1,\alpha))\;\middle|\;\mathcal{H}\right\|_p \\ &\quad\le \left\|\sum_{\alpha\in\Sigma}v''(\alpha,T(c-1,\alpha))\;\middle|\;\mathcal{H}\right\|_p \le 2\left\|\sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})v''(\alpha,T(c-1,\alpha))\;\middle|\;\mathcal{H}\right\|_p .\end{aligned}$$

We unfix $\mathcal{G}$ and get that

$$\begin{aligned}&2^{-1}\left\|\sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})v''(\alpha,T(c-1,\alpha))\right\|_p \\ &\quad\le \left\|\sum_{\alpha\in\Sigma}v''(\alpha,T(c-1,\alpha))\right\|_p \le 2\left\|\sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})v''(\alpha,T(c-1,\alpha))\right\|_p .\end{aligned} \tag{A.27}$$

We now note that

$$\begin{aligned}&\sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})v''(\alpha,T(c-1,\alpha)) \\ &\qquad= \sum_{\alpha\in\Sigma}\varepsilon(\{(c-1,\alpha)\})\sum_{x\in\Sigma^{c-1}}\varepsilon(x)v(x\cup\{(c-1,\alpha)\},h(x)\oplus T(c-1,\alpha)) \\ &\qquad= \sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x)) .\end{aligned}$$

Thus combining eq. (A.26) and eq. (A.27) finishes the proof. □

We can then generalize a result by Aamand et al. [AKKR+20]. The previous bound was only valid for $p = O(1)$ constant while we expand the applicability to all $p \geq 2$. Surprisingly, the main insight is that by symmetrizing, the combinatorial arguments become much simpler.

Lemma A.49. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation function, $\varepsilon\colon \Sigma^c \to \{-1, 1\}$ be a simple tabulation sign function, and $v_i\colon \Sigma^c \times [m] \to \mathbb{R}$ be value functions for $i \in [k]$. Then for every $p \geq 2$,*

$$\left\| \sum_{x_0,\dots,x_{k-1}\in\Sigma^c} \sum_{\substack{j_0,\dots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_p \leq \sqrt{pk}^{ck} \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2-1/k}. \tag{A.28}$$

Proof. We will argue that for every even integer $q \geq 2$,

$$\left\| \sum_{x_0\in\Sigma^{I_0},\dots,x_{k-1}\in\Sigma^{I_{k-1}}} \sum_{\substack{j_0,\dots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_q \leq \sqrt{\frac{qk}{2}}^{ck} \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2-1/k} \tag{A.29}$$

We claim that the result follows from this. Let $p \geq 2$ be a real number and let $q \geq 2$ be the unique even number such that $q \leq p < q + 2$. Since $q \geq 2$ then $p \leq 2q$ and we can

then use Jensen's inequality to get that

$$\begin{aligned}
&\left\| \sum_{x_0 \in \Sigma^{I_0}, \ldots, x_{k-1} \in \Sigma^{I_{k-1}}} \sum_{\substack{j_0, \ldots, j_{k-1} \in [m] \\ \bigoplus_{i \in [k]} j_i = 0}} \prod_{i \in [k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_p \\
&\quad \leq \left\| \sum_{x_0 \in \Sigma^{I_0}, \ldots, x_{k-1} \in \Sigma^{I_{k-1}}} \sum_{\substack{j_0, \ldots, j_{k-1} \in [m] \\ \bigoplus_{i \in [k]} j_i = 0}} \prod_{i \in [k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_{2q} \\
&\quad \leq \sqrt{\frac{2qk}{2}}^{ck} \prod_{i \in [k]} \|v_i\|_2^2 \left(\frac{\sum_{x \in \Sigma^c} \|v_i[x]\|_1}{\sum_{x \in \Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2 - 1/k} \\
&\quad \leq \sqrt{pk}^{ck} \prod_{i \in [k]} \|v_i\|_2 \left(\frac{\sum_{x \in \Sigma^c} \|v_i[x]\|_1^2}{\sum_{x \in \Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2 - 1/k} .
\end{aligned}$$

All we need to do now is to prove eq. (A.29). Let $q \geq 2$ be an even integer. The goal is to apply Lemma A.29 to prove the claim. First we define $f \colon \prod_{i \in [k]} \Sigma^{I_i} \to \mathbb{R}$ by

$$f(x_0, \ldots, x_{k-1}) = \sum_{\substack{j_0, \ldots, j_{k-1} \in [m] \\ \bigoplus_{i \in [k]} j_i = 0}} \prod_{i \in [k]} v_i(x_i, j_i \oplus h(x_i)) .$$

We then want to bound

$$\left\| \sum_{x_0, \ldots, x_{k-1} \in \Sigma^c} \left(\prod_{i \in [k]} \varepsilon(x_i) \right) f(x_0, \ldots, x_{k-1}) \right\|_q$$

If we fix h then we get that

$$\begin{aligned}
&\left\| \sum_{x_0, \ldots, x_{k-1} \in \Sigma^c} \left(\prod_{i \in [k]} \varepsilon(x_i) \right) f(x_0, \ldots, x_{k-1}) \,\middle|\, h \right\|_q^q \\
&\qquad = \sum_{\substack{x_0^{(0)}, \ldots, x_{k-1}^{(0)}, \ldots, x_0^{(q-1)}, \ldots, x_{k-1}^{(q-1)} \in \Sigma^c \\ \bigoplus_{j \in [q], i \in [k]} x_i^{(j)} = \emptyset}} \prod_{j \in [q]} f(x_0^{(j)}, \ldots, x_{k-1}^{(j)}) .
\end{aligned}$$

Now we want to bound f to a form such that we can use Lemma A.29. This will be done

by use of the Cauchy-Schwartz inequality.

$$\begin{aligned}
f(x_0,\ldots,x_{k-1}) &= \sum_{\substack{j_0,\ldots,j_{k-1}\in[m]\\ \bigoplus_{i\in[k]} j_i=0}} \prod_{i\in[k]} v_i(x_i, j_i \oplus h(x_i)) \\
&= \sum_{j_0,\ldots,j_{k-3}\in[m]} \left(\prod_{i\in[k-2]} v_i(x_i, j_i \oplus h(x_i))\right) \\
&\quad \cdot \sum_{j_{k-2}\in[m]} v_{k-2}(x_{k-2}, j_{k-2}\oplus h(x_{k-2})) v_{k-1}\left(x_{k-1}, h(x_{k-1}) \oplus \bigoplus_{s\in[k-1]} j_s\right) \\
&\leq \sum_{j_0,\ldots,j_{k-3}\in[m]} \left|\prod_{i\in[k-2]} v_i(x_i, j_i\oplus h(x_i))\right| \|v_{k-2}[x_{k-2}]\|_2 \, \|v_{k-1}[x_{k-1}]\|_2 \\
&= \left(\prod_{i\in[k-2]} \|v_i[x_i]\|_1\right) \|v_{k-2}[x_{k-2}]\|_2 \, \|v_{k-1}[x_{k-1}]\|_2 \\
&= \left(\prod_{i\in[k-2]} \frac{\|v_i[x_i]\|_1}{\|v_i[x_i]\|_2}\right) \prod_{i\in[k]} \|v_i[x_i]\|_2 \ .
\end{aligned}$$

Similarly, for all $i_1 \neq i_2 \in [k]$ we can prove that

$$f(x_0,\ldots,x_{k-1}) \leq \left(\prod_{i\in[k]\setminus\{i_1,i_2\}} \frac{\|v_i[x_i]\|_1}{\|v_i[x_i]\|_2}\right) \prod_{i\in[k]} \|v_i[x_i]\|_2 \ .$$

This implies that

$$f(x_0,\ldots,x_{k-1}) \leq \prod_{i\in[k]} \|v_i[x_i]\|_2 \left(\frac{\|v_i[x_i]\|_1}{\|v_i[x_i]\|_2}\right)^{1-2/k}$$

We are now ready to use Lemma A.29.

$$\begin{aligned}
&\left\| \sum_{x_0,\ldots,x_{k-1}\in\Sigma^c} \left(\prod_{i\in[k]} \varepsilon(x_i)\right) f(x_0,\ldots,x_{k-1}) \;\middle|\; h \right\|_q^q \\
&\quad \leq \sum_{\substack{x_0^{(0)},\ldots,x_{k-1}^{(0)},\ldots,x_0^{(q-1)},\ldots,x_{k-1}^{(q-1)}\in\Sigma^c \\ \bigoplus_{j\in[q],i\in[k]} x_i^{(j)}=\emptyset}} \prod_{j\in[q],i\in[k]} \left\|v_i[x_i^{(j)}]\right\|_2 \left(\frac{\left\|v_i[x_i^{(j)}]\right\|_1}{\left\|v_i[x_i^{(j)}]\right\|_2}\right)^{1-2/k} \\
&\quad \leq \sqrt{\frac{qk}{2}}^{qck} \left(\prod_{i\in[k]} \sqrt{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2 \left(\frac{\|v_i[x]\|_1^2}{\|v_i[x]\|_2^2}\right)^{1-2/k}}\right)^q \ .
\end{aligned}$$

Now we define the random variables R_i by $\Pr\left[R_i = \frac{\|v_i[x]\|_1^2}{\|v_i[x]\|_2^2}\right] = \frac{\|v_i[x]\|_2^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}$, and note that

$$\begin{aligned}
\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2\left(\frac{\|v_i[x]\|_1^2}{\|v_i[x]\|_2^2}\right)^{1-2/k} &= \left(\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2\right)\mathrm{E}\left[R_i^{1-2/k}\right] \\
&\leq \left(\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2\right)\mathrm{E}[R_i]^{1-2/k} \\
&= \left(\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2\right)\left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1-2/k} \\
&= \|v_i\|_2^2\left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1-2/k}.
\end{aligned}$$

The inequality follows by Jensen's inequality. This implies that

$$\begin{aligned}
&\left\|\sum_{x_0,\ldots,x_{k-1}\in\Sigma^c}\left(\prod_{i\in[k]}\varepsilon(x_i)\right)f(x_0,\ldots,x_{k-1})\;\middle|\;h\right\|_q^q \\
&\qquad\leq \sqrt{\tfrac{qk}{2}}^{qck}\left(\prod_{i\in[k]}\|v_i\|_2\left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1/2-1/k}\right)^q.
\end{aligned}$$

Now taking the q'th root, give us that

$$\begin{aligned}
&\left\|\sum_{x_0,\ldots,x_{k-1}\in\Sigma^c}\left(\prod_{i\in[k]}\varepsilon(x_i)\right)f(x_0,\ldots,x_{k-1})\right\|_q \\
&\qquad= \left\|\left\|\sum_{x_0,\ldots,x_{k-1}\in\Sigma^c}\left(\prod_{i\in[k]}\varepsilon(x_i)\right)f(x_0,\ldots,x_{k-1})\;\middle|\;h\right\|_q\right\|_q \\
&\qquad\leq \left\|\sqrt{\tfrac{qk}{2}}^{ck}\left(\prod_{i\in[k]}\|v_i\|_2\left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1/2-1/k}\right)\right\|_q \\
&\qquad= \sqrt{\tfrac{qk}{2}}^{ck}\prod_{i\in[k]}\|v_i\|_2\left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1/2-1/k},
\end{aligned}$$

which finishes the proof of the lemma. □

Bounding the Sum of Squares

The goal of this section is to prove Lemma A.52 from which we then get a bound of sum of squares of simple tabulation hashing. We start by proving a result for simple tabulation hashing that will serve as the base for the proof.

Lemma A.50. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation function, $\varepsilon\colon \Sigma^c \to \{-1,1\}$ be a simple tabulation sign function, and $v\colon \Sigma^c \times [m] \to \mathbb{R}$ be value function. Then for every real number $p \geq 2$*

$$\left\| \sum_{x\in\Sigma^c} \varepsilon(x)v(x,h(x)) \right\|_p \leq \sqrt{K_c p \left(\max\{p,\log(m)\}\right)^{c-1} \frac{\sum_{x\in\Sigma^c} \|v[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)^c}},$$

where $K_c = (Lc)^c$ for a universal constant L.

Proof. The proof will be by induction on c. For $c = 1$ the result follows by Lemma A.39.

Now we assume that the result is true for $c-1$. We define $v'\colon \Sigma^{c-1} \times [m] \to \mathbb{R}$ by $v'(x,j) = \sum_{\alpha\in\Sigma} \varepsilon_{c-1}(\alpha) v(x \cup \{(c-1,\alpha)\}, j \oplus T(c-1,\alpha))$. The induction hypothesis then give us that

$$\begin{aligned}
&\left\| \sum_{x\in\Sigma^c} \varepsilon(x)v(x,h(x)) \right\|_p \\
&\quad= \left\| \left\| \sum_{x\in\Sigma^{c-1}} \varepsilon(x)v'(x,h(x)) \;\middle|\; T(\{c-1\}\times\Sigma) \right\|_p \right\|_p \\
&\quad\leq \left\| \sqrt{K_{c-1} p \left(\max\{p,\log(m)\}\right)^{c-2} \frac{\sum_{x\in\Sigma^{c-1}} \|v'[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^{c-1}}\|v'[x]\|_\infty^2}{\sum_{x\in\Sigma^{c-1}}\|v'[x]\|_2^2}\right)^{c-1}}} \right\|_p \\
&\quad\leq \sqrt{K_{c-1} p \left(\max\{p,\log(m)\}\right)^{c-2}} \left\| \frac{\sum_{x\in\Sigma^{c-1}} \|v'[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^{c-1}}\|v'[x]\|_\infty^2}{\sum_{x\in\Sigma^{c-1}}\|v'[x]\|_2^2}\right)^{c-1}} \right\|_{p/2}^{1/2}.
\end{aligned} \tag{A.30}$$

We define the function $f\colon \mathbb{R}_{\geq 0}^2 \to \mathbb{R}_{\geq 0}$ by

$$f(x,y) = \begin{cases} 0 & \text{if } y = 0 \\ \frac{x}{\log\left(\frac{e^2 x}{y}\right)^{c-1}} & \text{if } 0 < y \leq x \\ \frac{x}{2^{c-1}} & \text{otherwise} \end{cases}.$$

Clearly, $\sum_{x\in\Sigma^{c-1}} \|v'[x]\|_\infty^2 \geq \frac{1}{m} \sum_{x\in\Sigma^{c-1}} \|v'[x]\|_2^2$, hence we have that

$$\frac{\sum_{x\in\Sigma^{c-1}} \|v'[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^{c-1}}\|v'[x]\|_\infty^2}{\sum_{x\in\Sigma^{c-1}}\|v'[x]\|_2^2}\right)^{c-1}} = f\left(\sum_{x\in\Sigma^{c-1}} \|v'[x]\|_\infty^2, \tfrac{1}{m} \sum_{x\in\Sigma^{c-1}} \|v'[x]\|_2^2 \right).$$

It is easy to check that $f(\lambda x, \lambda y) = \lambda f(x,y)$ so by Lemma A.15 we get that

$$\begin{aligned}
&\left\| f\left(\sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_\infty^2, \tfrac{1}{m} \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_2^2 \right) \right\|_{p/2} \\
&\qquad \le 2^{1/p} f\left(\left\| \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_\infty^2 \right\|_{p/2}, \tfrac{1}{m} \left\| \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_2^2 \right\|_{p/2} \right) \qquad \text{(A.31)} \\
&\qquad \le \sqrt{2} f\left(\left\| \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_\infty^2 \right\|_{p/2}, \tfrac{1}{m} \left\| \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_2^2 \right\|_{p/2} \right) .
\end{aligned}$$

We define $v_x \colon \Sigma \times [m] \to \mathbb{R}$ for every $x \in \Sigma^{c-1}$ by $v_x(\alpha, j) = v(x \cup \{(c-1,\alpha)\}, j)$. We then have that $v'(x,j) = \sum_{\alpha\in\Sigma} \varepsilon_{c-1}(\alpha) v_x(\alpha, j \oplus T(c-1,\alpha))$.

Let $\bar{p} = \max\{p, \log(m)\}$.

$$\begin{aligned}
\left\| \sum_{x\in\Sigma^{c-1}} \left\|v'[x]\right\|_\infty^2 \right\|_{p/2} &= \left\| \sum_{x\in\Sigma^{c-1}} \max_{j\in[m]} v'(x,j)^2 \right\|_{p/2} \\
&\le \sum_{x\in\Sigma^{c-1}} \left\| \max_{j\in[m]} v'(x,j)^2 \right\|_{p/2} \\
&= \sum_{x\in\Sigma^{c-1}} \left\| \max_{j\in[m]} \left|v'(x,j)\right| \right\|_p^2 \\
&\le \sum_{x\in\Sigma^{c-1}} \left\| \max_{j\in[m]} \left|v'(x,j)\right| \right\|_{\bar{p}}^2 \\
&\le \sum_{x\in\Sigma^{c-1}} \left(\sum_{j\in[m]} \left\|v'(x,j)\right\|_{\bar{p}}^{\bar{p}} \right)^{2/\bar{p}} \\
&\le \sum_{x\in\Sigma^{c-1}} \left(m \max_{j\in[m]} \left\|v'(x,j)\right\|_{\bar{p}}^{\bar{p}} \right)^{2/\bar{p}} \\
&\le e \sum_{x\in\Sigma^{c-1}} \max_{j\in[m]} \left\|v'(x,j)\right\|_{\bar{p}}^2
\end{aligned}$$

Now we will use that $v'(x,j) = \sum_{\alpha\in\Sigma} \varepsilon(\{(c-1,\alpha)\}) v_x(\alpha, j \oplus T(c-1,\alpha))$ and Lemma A.39.

$$\left\|v'(x,j)\right\|_{\bar{p}}^2 = \left\| \sum_{\alpha\in\Sigma} \varepsilon_{c-1}(\alpha) v_x(\alpha, j \oplus T(c-1,\alpha)) \right\|_{\bar{p}}^2 \le C_1 \bar{p} \frac{\sum_{\alpha\in\Sigma} \left\|v_x[\alpha]\right\|_\infty^2}{\log\left(\frac{e^2 m \sum_{\alpha\in\Sigma} \|v_x[\alpha]\|_\infty^2}{\sum_{\alpha\in\Sigma} \|v_x[\alpha]\|_2^2} \right)}$$

So we have that

$$\left\| \sum_{x \in \Sigma^{c-1}} \left\| v'[x] \right\|_\infty^2 \right\|_{p/2} \leq e \sum_{x \in \Sigma^{c-1}} \max_{j \in [m]} \left\| v'(x,j) \right\|_{\bar{p}}^2$$
$$\leq C_1 e \bar{p} \sum_{x \in \Sigma^{c-1}} \frac{\sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_2^2} \right)}.$$

Now we use Lemma A.40 to get that

$$\begin{aligned} \left\| \sum_{x \in \Sigma^{c-1}} \left\| v'[x] \right\|_\infty^2 \right\|_{p/2} &\leq C_1 e \bar{p} \sum_{x \in \Sigma^{c-1}} \frac{\sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_2^2} \right)} \\ &\leq C_1 e \bar{p} \frac{\sum_{x \in \Sigma^{c-1}} \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^{c-1}} \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_\infty^2}{\sum_{x \in \Sigma^{c-1}} \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_2^2} \right)} \\ &= C_1 e \bar{p} \frac{\sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)} \end{aligned} \tag{A.32}$$

We again use that $v'(x,j) = \sum_{\alpha \in \Sigma} \varepsilon_{c-1}(\alpha) v_x(\alpha, j \oplus T(c-1,\alpha))$ to get that

$$\begin{aligned} \left\| \sum_{x \in \Sigma^{c-1}} \left\| v'[x] \right\|_2^2 \right\|_{p/2} &= \left\| \sum_{x \in \Sigma^{c-1}} \sum_{j \in [m]} v'(x,j)^2 \right\|_{p/2} \\ &\leq \sum_{x \in \Sigma^{c-1}} \left\| \sum_{j \in [m]} v'(x,j)^2 \right\|_{p/2} \\ &= \sum_{x \in \Sigma^{c-1}} \left\| \sum_{j \in [m]} \left(\sum_{\alpha \in \Sigma} \varepsilon_{c-1}(\alpha) v_x(\alpha, j \oplus T(c-1,\alpha)) \right)^2 \right\|_{p/2} \\ &= \sum_{x \in \Sigma^{c-1}} \left\| \sum_{j \in [m]} \sum_{\alpha,\beta \in \Sigma} \varepsilon_{c-1}(\alpha)\varepsilon_{c-1}(\beta) v_x(\alpha, j \oplus T(c-1,\alpha)) v_x(\beta, j \oplus T(c-1,\beta)) \right\|_{p/2} \end{aligned}$$

Now we can use Lemma A.49 to get that

$$\left\| \sum_{j \in [m]} \sum_{\alpha,\beta \in \Sigma} \varepsilon_{c-1}(\alpha)\varepsilon_{c-1}(\beta) v_x(\alpha, j \oplus T(c-1,\alpha)) v_x(\beta, j \oplus T(c-1,\beta)) \right\|_{p/2}$$
$$\leq 2p \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_2^2$$

This implies that

$$\left\| \sum_{x \in \Sigma^{c-1}} \|v'[x]\|_2^2 \right\|_{p/2} \leq 2p \sum_{x \in \Sigma^{c-1}} \sum_{\alpha \in \Sigma} \|v_x[\alpha]\|_2^2 = 2p \sum_{x \in \Sigma^c} \|v[x]\|_2^2 \leq 2\bar{p} \sum_{x \in \Sigma^c} \|v[x]\|_2^2 \ . \tag{A.33}$$

Combining eq. (A.30), eq. (A.31), eq. (A.32), and eq. (A.33) we get that

$$\begin{aligned} \left\| \sum_{x \in \Sigma^c} \varepsilon(x) v(x, h(x)) \right\|_p \leq & \sqrt{K_{c-1} p \left(\max\{p, \log(m)\}\right)^{c-2}} \\ & \cdot f\left(C_1 e \bar{p} \frac{\sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)}, \frac{1}{m} 2\bar{p} \sum_{x \in \Sigma^c} \|v[x]\|_2^2 \right)^{1/2} . \end{aligned}$$

We will now argue that

$$f\left(C_1 e \bar{p} \frac{\sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)}, \frac{1}{m} 2\bar{p} \sum_{x \in \Sigma^c} \|v[x]\|_2^2 \right) \leq Lc \frac{\bar{p}}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)^c} ,$$

which will finish the proof.

We will use Lemma A.47 to get that

$$\frac{\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)^{1/c}}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)} \geq \frac{e}{c} .$$

This implies that

$$\frac{C_1 e \bar{p} \frac{\sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)}}{\frac{1}{m} 2\bar{p} \sum_{x \in \Sigma^c} \|v[x]\|_2^2} \geq \frac{C_1}{2c} \left(\frac{e^2 m \sum_{x \in \Sigma^c} \|v[x]\|_\infty^2}{\sum_{x \in \Sigma^c} \|v[x]\|_2^2} \right)^{1 - 1/c} .$$

We then get that

$$
\begin{aligned}
f\left(C_1 e\bar{p}\frac{\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\log\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)}, \frac{1}{m}2\bar{p}\sum_{x\in\Sigma^c}\|v[x]\|_2^2\right) \\
&\leq f\left(2ec\bar{p}\frac{\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\log\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)}, \frac{1}{m}2\bar{p}\sum_{x\in\Sigma^c}\|v[x]\|_2^2\right) \\
&= 2ec\frac{\bar{p}}{\log\left(\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)^{1-1/c}\right)^c} \\
&= 2ec\frac{\bar{p}}{\log\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)^c\left(1-\frac{1}{c}\right)^c} \\
&\leq 8ec\frac{\bar{p}}{\log\left(\frac{e^2m\sum_{x\in\Sigma^c}\|v[x]\|_\infty^2}{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}\right)^c}\,.
\end{aligned}
$$

This finishes the proof. □

We will now expand the result of Lemma A.17 to chaoses of simple tabulation hashing. For this we need the following decoupling lemma of de la Pena et al. [PMS94].

Lemma A.50 (Decoupling [PMS94])**.** *Let $(f^{(j)}_{i_0,\ldots,i_{k-1}})_{i_0,\ldots,i_{k-1}\in[n],j\in[m]}$ be a multiindexed array of real numbers, and assume that $f^{(j)}_{i_0,\ldots,i_{k-1}} = 0$ if $i_l = i_{l'}$ for some $l \neq l'$. Let $(X^{(j)}_i)_{i\in[n]}$ be a sequence of independent and identically distributed random variables for every $j \in [m]$.*

Define $(Y^{(j)}_{i,l})_{i\in[n],l\in[k]}$ to be a sequence of independent and identically distributed random variables which has the same distribution as $X^{(j)}_0$ for every $j \in [m]$. Then for every $p \geq 2$,

$$
\left\|\sum_{j\in[m]}\sum_{i_0,\ldots,i_{k-1}\in[n]} f^{(d)}_{i_0,\ldots,i_{k-1}}\prod_{l\in[k]}X^{(j)}_{i_l}\right\|_p \leq L_k\left\|\sum_{j\in[m]}\sum_{i_0,\ldots,i_{k-1}\in[n]} f^{(d)}_{i_0,\ldots,i_{k-1}}\prod_{l\in[k]}Y^{(j)}_{i_l,l}\right\|_p,
$$

where $L_k \leq k^k$ if $\mathrm{E}\left[X^{(j)}_0\right] = 0$ for all $j \in [m]$, and $L_k \leq (2k+1)^k$ otherwise.

If we specialize it to simple tabulation hashing we get the following corollary.

Corollary A.51. *Let $(F_{\alpha_0,\ldots,\alpha_{k-1}})_{\alpha_0,\ldots,\alpha_{k-1}\in\Sigma}$ be a multiindexed array of real functions $F_{\alpha_0,\ldots,\alpha_{k-1}}\colon [m] \to \mathbb{R}$, and assume that $F_{\alpha_0,\ldots,\alpha_{k-1}} = 0$ if $x_l = x_{l'}$ for some $l \neq l'$. Let*

$h\colon \Sigma \to [m]$ be a fully random function and let $\varepsilon\colon \Sigma \to \{-1,1\}$ be a fully random sign function. Let $h'\colon \Sigma^k \to [m]$ be a simple tabulation hash function and let $\varepsilon'\colon \Sigma^k \to \{-1,1\}$ be a simple tabulation sign function. Then for every $p \geq 2$,

$$\left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} F_{\alpha_0,\dots,\alpha_{k-1}}(h(\alpha_0)\oplus\ldots\oplus h(\alpha_{k-1})) \prod_{l\in[k]} \varepsilon(\alpha_l) \right\|_p \leq k^k \left\| \sum_{x\in\Sigma^k} F_x(h'(x))\varepsilon'(x) \right\|_p .$$

Proof. We start by noticing that we can write the expression as follows,

$$\begin{aligned}&\left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} F_{\alpha_0,\dots,\alpha_{k-1}}(h(\alpha_0)\oplus\ldots\oplus h(\alpha_{k-1})) \prod_{l\in[k]} \varepsilon(\alpha_l) \right\|_p \\ &\qquad = \left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} \sum_{j_0,\dots,j_{k-1}} F_{\alpha_0,\dots,\alpha_{k-1}}(j_0\oplus\ldots\oplus j_{k-1}) \prod_{l\in[k]} \varepsilon(\alpha_l)\,[h(\alpha_l)=j_l] \right\|_p .\end{aligned}$$

We can then use Lemma A.50 to get that

$$\begin{aligned}&\left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} \sum_{j_0,\dots,j_{k-1}} F_{\alpha_0,\dots,\alpha_{k-1}}(j_0\oplus\ldots\oplus j_{k-1}) \prod_{l\in[k]} \varepsilon(\alpha_l)\,[h(\alpha_l)=j_l] \right\|_p \\ &\quad \leq k^k \left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} \sum_{j_0,\dots,j_{k-1}} F_{\alpha_0,\dots,\alpha_{k-1}}(j_0\oplus\ldots\oplus j_{k-1}) \prod_{l\in[k]} \varepsilon'(\{l,\alpha_l\})\,[T(l,\alpha_l)=j_l] \right\|_p .\end{aligned}$$

We can then finish the proof by reversing the rewriting,

$$\begin{aligned}&\left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} \sum_{j_0,\dots,j_{k-1}} F_{\alpha_0,\dots,\alpha_{k-1}}(j_0\oplus\ldots\oplus j_{k-1}) \prod_{l\in[k]} \varepsilon'(\{l,\alpha_l\})\,[T(l,\alpha_l)=j_l] \right\|_p \\ &\quad = \left\| \sum_{\alpha_0,\dots,\alpha_{k-1}\in\Sigma} F_{\alpha_0,\dots,\alpha_{k-1}}(T(0,\alpha_0)\oplus\ldots\oplus T(k-1,\alpha_{k-1}))\varepsilon'(\alpha_0,\dots,\alpha_{k-1}) \right\|_p \\ &\quad = \left\| \sum_{x\in\Sigma^k} F_x(h'(x))\varepsilon'(x) \right\|_p .\end{aligned}$$

□

We can now prove our result for chaoses of simple tabulation hashing.

Lemma A.52. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation function, $\varepsilon\colon \Sigma^c \to \{-1,1\}$ be a simple tabulation sign function, and $v_i\colon \Sigma^c \times [m] \to \mathbb{R}$ be value function for $i \in [k]$. For*

every real number $p \geq 2$,

$$\left\| \sum_{x_0,\ldots,x_{k-1}\in\Sigma^c} \sum_{\substack{j_0,\ldots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_p$$

$$\leq \left(\frac{Lck^3 \max\{p, \log(m)\}}{\log\left(\prod_{i\in[k]} \left(\frac{e^2 m \sum_{x\in\Sigma^c} \|v_i[x]\|_\infty^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2} \right)^{1/k} \right)} \right)^{ck/2} \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\|v_i\|_1}{\|v_i\|_2} \right)^{1-2/k},$$

where L is a universal constant.

Proof. For every $j \in [c]$ we define $\pi_j \colon \Sigma^c \to \{i\} \times \Sigma$ to be the projection onto the i'th position character, i.e., for a key $x = \{(0, \alpha_0), \ldots, (c-1, \alpha_{c-1})\}$ we have that $\pi_i(x) = (i, \alpha_i)$.

We define $\bar{p} = \max\{p, \log(m)\}$ to ease notation.

We make the observation that $\sum_{\substack{j_0,\ldots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i))$ depends only on $\bigoplus_{i\in[k]} h(x_i)$. More precisely, we note that if we define $v' \colon \Sigma^{ck} \times [m] \to \mathbb{R}$ by,

$$v'((x_0, \ldots, x_{k-1}), j) = \sum_{\substack{j_0,\ldots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = j}} \prod_{i\in[k]} v_i(x_i, j_i)$$

then we have that

$$\sum_{\substack{j_0,\ldots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) = v'((x_0, \ldots, x_{k-1}), \bigoplus_{i\in[k]} h(x_i))$$

This implies that we can the expression into sub-expressions depending on the number of distinct characters at each position.

Let $t_0, \ldots, t_{c-1}$ be even integers less than k. Now fix pairs $((s_l^{(i)}, r_l^{(i)}))_{l\in[t_i/2]}$ for $i \in [k]$ and define the set X by,

$$X = \Big\{ (x_0, \ldots, x_{k-1}) \in (\Sigma^c)^k \;\Big|\; \forall i \in [c] \colon \Big(\forall l \in [t_i] \colon \pi_i(x_{s_l^{(i)}}) = \pi_i(x_{r_l^{(i)}}) \\ \wedge (\pi_i(x_j))_{j\in[k]\setminus\bigcup_{l\in[t_i]}\{s_l^{(i)}, r_l^{(i)}\}} \text{ are all distinct} \Big) \Big\}.$$

We define $T^{(i)} \colon [c] \times \Sigma \to [m]$ to be independent copies of T for $i \in [k]$, and similarly, define $\varepsilon^{(i)} \colon \Sigma \to \{-1, 1\}$ to be independent copies of ε for $i \in [k]$. We define the set,

$$R_i = \left\{ v \in [c] \;\middle|\; i \notin \bigcup_{l\in[t_i/2]} \left\{ s_l^{(v)}, r_l^{(v)} \right\} \right\},$$

for $i \in [k]$. We now use Corollary A.51 to get that

$$\left\| \sum_{x_0,\dots,x_{k-1} \in X} v'((x_0,\dots,x_{k-1}), \bigoplus_{i\in[k]} h(x_i)) \prod_{i\in[k]} \varepsilon(x_i) \right\|_p \leq \prod_{i\in[c]} (k-t_i)^{k-t_i} \left\| \sum_{x_0,\dots,x_{k-1} \in X} v'((x_0,\dots,x_{k-1}), \bigoplus_{i\in[k]} \bigoplus_{l\in R_i} T^{(i)}(l,\pi_l(x_i))) \prod_{i\in[k]} \prod_{l\in R_i} \varepsilon^{(i)}(l,\pi_l(x_i)) \right\|_p .$$

This corresponds to a simple tabulation function with $ck - \sum_{i\in[k]} t_i$ characters. We can then use Lemma A.17 to get that

$$\left\| \sum_{x_0,\dots,x_{k-1} \in X} v'((x_0,\dots,x_{k-1}), \bigoplus_{i\in[k]} \bigoplus_{l\in R_i} T^{(i)}(l,\pi_l(x_i))) \prod_{i\in[k]} \prod_{l\in R_i} \varepsilon^{(i)}(l,\pi_l(x_i)) \right\|_p \leq \sqrt{\left(L(ck-\sum_{i\in[k]} t_i)\bar{p}\right)^{ck-\sum_{i\in[k]} t_i} \frac{\sum_{x\in X} \|v'[x]\|_\infty^2}{\log\left(\frac{e^2 m \sum_{x\in X}\|v'[x]\|_\infty^2}{\sum_{x\in X}\|v'[x]\|_2^2}\right)^{ck-\sum_{i\in[k]} t_i}}}$$

Now repeated use of Cauchy-Schwartz as in the proof of Lemma A.49 implies that

$$\sum_{x\in X} \|v'[x]\|_\infty^2 \leq \prod_{i\in[k]} \|v_i\|_2^2 \left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1-2/k},$$

$$\sum_{x\in X} \|v'[x]\|_2^2 \leq \prod_{i\in[k]} \|v_i\|_2^2 \left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1-1/k}.$$

We then get that

$$\left\| \sum_{x_0,\dots,x_{k-1} \in X} v'((x_0,\dots,x_{k-1}), \bigoplus_{i\in[k]} \bigoplus_{l\in R_i} T^{(i)}(l,\pi_l(x_i))) \prod_{i\in[k]} \prod_{l\in R_i} \varepsilon^{(i)}(l,\pi_l(x_i)) \right\|_p \leq \left(\frac{L(ck-\sum_{i\in[k]} t_i)\bar{p}}{\log\left(\prod_{i\in[k]}\left(\frac{e^2 m\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}\right)^{1/k}\right)}\right)^{(ck-\sum_{i\in[k]} t_i)/2} \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\sum_{x\in\Sigma^c}\|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c}\|v_i[x]\|_2^2}\right)^{1/2-1/k}.$$

Now we note that given $(t_i)_{i\in[k]}$ we can choose the pairs $((s_l^{(i)}, r_l^{(i)}))_{l\in[t_i/2]}$ for $i \in [k]$ in $\prod_{i\in[k]} \binom{k}{t_i}(t_i-1)!!$ ways. Summing all the possible values for $(t_i)_{i\in[k]}$ we get that

$$\begin{aligned}
&\left\| \sum_{x_0,\dots,x_{k-1}\in\Sigma^c} \sum_{\substack{j_0,\dots,j_{k-1}\in[m] \\ \bigoplus_{i\in[k]} j_i = 0}} \prod_{i\in[k]} \varepsilon(x_i) v_i(x_i, j_i \oplus h(x_i)) \right\|_p \\
&\quad \le \sum_{t_0,\dots,t_{k-1}} \left(\prod_{i\in[k]} \binom{k}{t_i} (t_i-1)!! (k-t_i)^{k-t_i} \right) \\
&\qquad \cdot \left(\frac{L(ck - \sum_{i\in[k]} t_i)\bar{p}}{\log\left(\prod_{i\in[k]} \left(\frac{e^2 m \sum_{x\in\Sigma^c} \|v_i[x]\|_2^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2} \right)^{1/k} \right)} \right)^{(ck - \sum_{i\in[k]} t_i)/2} \\
&\qquad\qquad \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2-1/k} \\
&\quad \le \left(\frac{L_2 c k^3 \bar{p}}{\log\left(\prod_{i\in[k]} \left(\frac{e^2 m \sum_{x\in\Sigma^c} \|v_i[x]\|_2^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2} \right)^{1/k} \right)} \right)^{ck/2} \prod_{i\in[k]} \|v_i\|_2 \left(\frac{\sum_{x\in\Sigma^c} \|v_i[x]\|_1^2}{\sum_{x\in\Sigma^c} \|v_i[x]\|_2^2} \right)^{1/2-1/k}
\end{aligned}$$

This finishes the proof. □

It now becomes easy to prove Lemma A.18.

Lemma A.53. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation function, $\varepsilon\colon \Sigma^c \to \{-1, 1\}$ be a simple tabulation sign function, and $v_i\colon \Sigma^c \times [m] \to \mathbb{R}$ be a value function for $i \in [k]$. For every real number $p \ge 2$*

$$\left\| \sum_{j\in[m]} \left(\sum_{x\in\Sigma^c} \varepsilon(x) v(x, h(x) \oplus j) \right)^2 \right\|_p \le \left(\frac{Lc \max\{p, \log(m)\}}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\sum_{x\in\Sigma^c} \|v[x]\|_1^2} \right)} \right)^c \sum_{x\in\Sigma^c} \|v[x]\|_2^2 ,$$

where L is a universal constant.

Proof. This follows by Lemma A.52 since,

$$\left\| \sum_{j\in[m]} \left(\sum_{x\in\Sigma^c} \varepsilon(x) v(x, h(x)) \right)^2 \right\|_p = \left\| \sum_{x,y\in\Sigma^c} \sum_{j\in[m]} \varepsilon(x)\varepsilon(y) v(x, h(x)) v(y, h(y)) \right\|_p .$$

□

Concentration Result for Simple Tabulation Hashing

We are now ready to prove the main result of the section. Note that by using Lemma A.48, the result can be extended to the case without symmetrization, which proves Theorem A.7. We warn the reader that the proof is long and technical.

Theorem A.7. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x,j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{simple}} = \sum_{x\in\Sigma^c} v(x,h(x))$. Then for all $p \geq 2$*

$$\left\| V_v^{\text{simple}} \right\|_p \leq L_1 \Psi_p \left(K_c \gamma_p^{c-1} M_v, K_c \gamma_p^{c-1} \sigma_v^2 \right) ,$$

where $K_c = (L_2 c)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{ \log(m) + \log\left(\frac{\sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\max_{x\in\Sigma^c} \|v[x]\|_2^2} \right) / c, p \right\}}{\log\left(e^2 m \left(\max_{x\in\Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2} \right)^{-1} \right)}$$

Proof. We will prove the result by induction on c. For $c = 1$ it corresponds to using a fully random hash function, and the result follows by Theorem A.5.

Now we assume that $c > 1$ and that the result is true for values less than c. We note that without loss of generality we can assume that $M_v = 1$. Let $w\colon \Sigma^c \to \mathbb{R}$ be a function defined by $w(x) = \|v[x]\|_2^2$ and for $X \subseteq \Sigma^c$ we overload the notation of w to write $w(X) = \sum_{x\in X} w(x)$. Furthermore, we define $w_\infty(X) = \max_{x\in X} w(x)$.

Now applying Lemma A.28 we obtain an ordering of position characters $\{\alpha_0, \ldots, \alpha_{r-1}\} = [c] \times \Sigma$ where $r = c|\Sigma|$, satisfying that the groups $G_i = \{x \in \Sigma^c \mid \alpha_i \in x \wedge x \subseteq \{\alpha_0, \ldots, \alpha_i\}\}$ has the property that $w(G_i) \leq w(\Sigma^c)^{1-1/c} w_\infty(\Sigma^c)^{1/c}$ for every $i \in [r]$.

We define the random variables

$$X_i^{(j)} = \sum_{x\in G_i} \varepsilon(x \setminus \{\alpha_i\}) v(x, j \oplus h(x \setminus \{\alpha_i\})) ,$$
$$Y_i = \varepsilon(\alpha_i) X_i^{(h(\alpha_i))} ,$$

for all $i \in [r], j \in [m]$. With this notation we have that

$$V_v^{\text{simple}} = V = \sum_{i\in[r]} Y_i .$$

We let $(\mathcal{F}_i)_{i\in[r]}$ be a filtration where $\mathcal{F}_i = \sigma((h(\alpha_k), \varepsilon(\alpha_k))_{k\in[i+1]})$ for $i \in [r]$. It is easy to see that $X_i^{(j)}$ is $\mathcal{F}_{i-1}$-measurable, $Y_i^{(j)}$ is $\mathcal{F}_i$-measurable, and $\mathrm{E}\left[Y_i^{(j)} \,\middle|\, \mathcal{F}_{i-1} \right] = 0$ for all $j \in [m], i \in [r]$. We thus have that $(Y_i, \mathcal{F}_i)_{i\in[r]}$ is a martingale difference sequence. We furthermore notice that

$$\mathrm{Var}[Y_i \mid \mathcal{F}_i] = \frac{1}{m} \sum_{k\in[m]} \left(X_i^{(k)} \right)^2 ,$$

for all $j \in [m], i \in [r]$.

Let $h' \colon \Sigma^c \to [m]$ be a simple tabulation hash function independent of h, and $\varepsilon' \colon \Sigma^c \to [m]$ be a simple tabulation sign function independent of ε. We define the random variables $Z_i^{(j)} = \varepsilon'(\alpha_i) X^{(j \oplus h'(\alpha_i))}$. We can now easily check that $(Z_i)_{i \in [r]}$ satisfies the properties needed for Lemma A.46:

1. $(Y_i \mid \mathcal{F}_{i-1})$ and $(Z_i \mid \mathcal{F}_{i-1})$ have the same distribution for every $i \in [r]$.
2. The sequence $(Z_i)_{i \in [r]}$ is conditionally independent given $\mathcal{F}_{r-1}$.
3. $(Z_i \mid \mathcal{F}_{i-1})$ and $(Z_i \mid \mathcal{F}_{r-1})$ have the same distribution for every $i \in [r]$.
4. $(Y_i \mid \mathcal{F}_{i-1})$ and $(Y_i \mid \sigma(\mathcal{F}_{i-1}, (Z_k)_{k \in [i+1]})$ have the same distribution for every $i \in [r]$.

Now Lemma A.46 then implies that

$$\left\| \sum_{i \in [r]} Y_i \right\|_p \leq M \left\| \sum_{i \in [r]} Z_i \right\|_p . \tag{A.34}$$

We now use that $(Z_i)_{i \in [r]}$ are conditionally independent given h, so fixing h and using Theorem A.5 we get that

$$\left\| \sum_{i \in [r]} Z_i \;\middle|\; h \right\|_p \leq 16 e \Psi_p \left(\max_{i \in [r], j \in [m]} \left| X_i^{(j)} \right|, \frac{\sum_{i \in [r], j \in [m]} \left(X_i^{(j)} \right)^2}{m} \right) . \tag{A.35}$$

Since $\Psi_p(\lambda M, \lambda^2 \sigma^2) = \lambda \Psi_p(M, \sigma^2)$, we then use Lemma A.15 to get that

$$\begin{aligned} &\left\| \Psi_p \left(\max_{i \in [r], j \in [m]} \left| X_i^{(j)} \right|, \frac{\sum_{i \in [r], j \in [m]} \left(X_i^{(j)} \right)^2}{m} \right) \right\|_p \\ &\qquad \leq 2^{1/p} \Psi_p \left(\left\| \max_{i \in [r], j \in [m]} \left| X_i^{(j)} \right| \right\|_p, \frac{1}{m} \left\| \sum_{i \in [r], j \in [m]} \left(X_i^{(j)} \right)^2 \right\|_{p/2}^{1/2} \right) \\ &\qquad \leq \sqrt{2} \Psi_p \left(\left\| \max_{i \in [r], j \in [m]} \left| X_i^{(j)} \right| \right\|_p, \frac{1}{m} \left\| \sum_{i \in [r], j \in [m]} \left(X_i^{(j)} \right)^2 \right\|_{p/2} \right) . \end{aligned} \tag{A.36}$$

We set $\bar{p} = \max\left\{ p, \log(m) + \log\left(\frac{w(\Sigma^c)}{w_\infty(\Sigma^c)} \right) / c \right\}$. With this notation we have that

$$\gamma_p = \frac{\bar{p}}{\log\left(\min_{x \in \Sigma^c} \frac{e^2 m \sum_{j \in [m]} v(x,j)^2}{\left(\sum_{j \in [m]} |v(x,j)| \right)^2} \right)} .$$

We start by bounding $\left\|\max_{i\in[r],j\in[m]}\left|X_i^{(j)}\right|\right\|_p$. By the induction hypothesis we get that

$$\begin{aligned}
\left\|\max_{i\in[r],j\in[m]}\left|X_i^{(j)}\right|\right\|_p &\leq \left\|\max_{i\in[r],j\in[m]}\left|X_i^{(j)}\right|\right\|_{\bar{p}} \\
&\leq \left(\sum_{i\in[r],j\in[m]}\left\|X_i^{(j)}\right\|_{\bar{p}}^{\bar{p}}\right)^{1/\bar{p}} \\
&\leq \left(m\sum_{i\in[r]}\max_{j\in[m]}\left\|X_i^{(j)}\right\|_{\bar{p}}^{\bar{p}}\right)^{1/\bar{p}} \\
&\leq \left(m\sum_{i\in[r]}L_1^p\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(G_i)}{m}\right)^{\bar{p}}\right)^{1/\bar{p}} \\
&\leq L_1\left(m\sum_{i\in[r]}\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(G_i)}{m}\right)^{\bar{p}}\right)^{1/\bar{p}}.
\end{aligned}$$

An easy observation is that $\Psi_{\bar{p}}(M,\sigma^2)^{\bar{p}}$ is a convex function in σ^2. So using that $\max_{i\in[r]} w(G_i) \leq w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}$ and that $\sum_{i\in[r]} w(G_i) = w(\Sigma^c)$ we get that

$$\begin{aligned}
&\left\|\max_{i\in[r],j\in[m]}\left|X_i^{(j)}\right|\right\|_p \\
&\quad\leq L_1\left(m\left(\tfrac{w(\Sigma^c)}{w_\infty(\Sigma^c)}\right)^{1/c}\right)^{1/\bar{p}}\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right) \\
&\quad\leq L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right).
\end{aligned} \tag{A.37}$$

The last inequality follows since $\bar{p} \geq \log(m) + \log\left(\frac{w(\Sigma^c)}{w_\infty(\Sigma^c)}\right)/c$.

We will bound $\left\|\sum_{i\in[r],j\in[m]}\left(X_i^{(j)}\right)^2\right\|_{p/2}$ by using the triangle inequality and

Lemma A.18.

$$\begin{aligned}
\left\| \sum_{i\in[r],j\in[m]} \left(X_i^{(j)}\right)^2 \right\|_{p/2} &\le \sum_{i\in[r]} \left\| \sum_{j\in[m]} \left(X_i^{(j)}\right)^2 \right\|_{p/2} \\
&\le K'_{c-1} \max\left\{ 1, \left(\frac{p/2}{\log\left(\frac{e^2 m \sum_{x\in\Sigma^c} \|v[x]\|_2^2}{\sum_{x\in\Sigma^c} \|v[x]\|_1^2} \right)} \right)^c \right\} \sum_{i\in[r]} w(G_i) \\
&\le K'_{c-1} \max\left\{ 1, \left(\frac{p}{\log\left(\min_{x\in\Sigma^c} \frac{e^2 m \|v[x]\|_2^2}{\|v[x]\|_1^2} \right)} \right)^c \right\} \sum_{i\in[r]} w(G_i) \\
&\le K'_{c-1} \gamma_p^{c-1} w(\Sigma^c) \\
&\le K_c \gamma_p^{c-1} w(\Sigma^c) \,.
\end{aligned} \tag{A.38}$$

Here K'_{c-1} is the constant depending on $c-1$ which we get from Lemma A.18.

Now combining eq. (A.36), eq. (A.37), and eq. (A.38) we get that

$$\begin{aligned}
&\left\| \Psi_p \left(\max_{i\in[r],j\in[m]} \left| X_i^{(j)} \right|, \frac{\sum_{i\in[r],j\in[m]} \left(X_i^{(j)}\right)^2}{m} \right) \right\|_p \\
&\quad \le \sqrt{2}\Psi_p \left(L_1 e \Psi_{\bar{p}} \left(K_{c-1}\gamma_p^{c-2}, K_{c-1} \frac{\gamma_p^{c-2} w(\Sigma^c)^{1-1/c} w_\infty(\Sigma^c)^{1/c}}{m} \right), K_c \gamma_p^{c-1} \frac{w(\Sigma^c)}{m} \right) \\
&\quad = \sqrt{2}\Psi_p \left(L_1 e \Psi_{\bar{p}} \left(K_{c-1}\gamma_p^{c-2}, K_{c-1} \frac{\gamma_p^{c-2} w(\Sigma^c)^{1-1/c} w_\infty(\Sigma^c)^{1/c}}{m} \right), K_c \gamma_p^{c-1} \sigma_v^2 \right) \,.
\end{aligned} \tag{A.39}$$

Now we will consider two cases depending on $w(\Sigma^c)$.

Case 1. $w(\Sigma^c) \le \left(\bar{p} e^{-2} K_{c-1} \gamma_p^{c-2}\right)^{c/(c-1)} m \left(\frac{m}{w_\infty(\Sigma^c)}\right)^{1/(2c-1)}$. In this case we will show that,

$$L_1 e \Psi_{\bar{p}} \left(K_{c-1}\gamma_p^{c-2}, K_{c-1} \frac{\gamma_p^{c-2} w(\Sigma^c)^{1-1/c} w_\infty(\Sigma^c)^{1/c}}{m} \right) \le K_c \gamma_p^{c-1} \,. \tag{A.40}$$

We first notice that

$$\begin{aligned}
&\frac{K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}{K_{c-1}^2\gamma_p^{2c-4}} \\
&\quad= \frac{w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{mK_{c-1}\gamma_p^{c-2}} \\
&\quad\le \frac{\left(\bar{p}e^{-2}K_{c-1}\gamma_p^{c-2}\right)m^{(2c-2)/(2c-1)}w_\infty(\Sigma^c)^{-(c-1)/c(2c-1)}w_\infty(\Sigma^c)^{1/c}}{mK_{c-1}\gamma_p^{c-2}} \\
&\quad= e^{-2}\bar{p}\left(\frac{w_\infty(\Sigma^c)}{m}\right)^{1/(2c-1)} \\
&\quad= e^{-2}\bar{p}\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_2^2}{m}\right)^{1/(2c-1)} \\
&\quad\le e^{-2}\bar{p}\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{m\,\|v[x]\|_2^2}\right)^{1/(2c-1)} \\
&\quad\le e^{-2}\bar{p}\,.
\end{aligned} \tag{A.41}$$

Now by Equation (A.18) we get that

$$\begin{aligned}
&L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right) \\
&\qquad\le L_1 e\frac{\bar{p}}{e\log\left(\frac{\bar{p}mK_{c-1}^2\gamma_p^{2c-4}}{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}\right)}K_{c-1}\gamma_p^{c-2} \\
&\qquad\le L_1\frac{\bar{p}}{\log\left(e^2\left(\min_{x\in\Sigma^c}\frac{m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{1/(2c-1)}\right)}K_{c-1}\gamma_p^{c-2} \\
&\qquad\le 2L_1cK_{c-1}\gamma_p^{c-1} \\
&\qquad\le K_c\gamma_p^{c-1}\,.
\end{aligned}$$

Where we have used that $\frac{K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}{K_{c-1}^2\gamma_p^{2c-4}} \le e^{-2}\bar{p}\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{m\|v[x]\|_2^2}\right)^{1/(2c-1)}$ which follows from eq. (A.41), and that $2L_1cK_{c-1}\le K_c$.

Now combining eq. (A.34), eq. (A.35), eq. (A.39), and eq. (A.40) we get the result.

Case 2. $w(\Sigma^c) > \left(\bar{p}e^{-2}K_{c-1}\gamma_p^{c-2}\right)^{c/(c-1)}m\left(\frac{m}{w_\infty(\Sigma^c)}\right)^{1/(2c-1)}$**.** In this case we will argue that

$$\begin{aligned}
&\Psi_p\left(L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right), K_c\gamma_p^{c-1}w(\Sigma^c)\right) \\
&\qquad\le \tfrac{1}{2}\sqrt{\bar{p}}\sqrt{K_c\gamma_p^{c-1}w(\Sigma^c)}\,.
\end{aligned} \tag{A.42}$$

We use eq. (A.18) to get that

$$L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right) \leq \max\left\{L_1\tfrac{e}{2}\sqrt{\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}, L_1\tfrac{1}{2}\bar{p}K_{c-1}\gamma_p^{c-2}\right\}.$$

We apply eq. (A.18) again to obtain that

$$\Psi_p\left(L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right), K_c\gamma_p^{c-1}w(\Sigma^c)\right)$$
$$\leq \max\left\{\tfrac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}, \tfrac{1}{2e}pL_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right)\right\}.$$

Combining the two estimates give us that

$$\Psi_p\left(L_1 e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right), K_c\gamma_p^{c-1}w(\Sigma^c)\right)$$
$$\leq \max\left\{\tfrac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}, \tfrac{L_1}{4}p\sqrt{\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}, \tfrac{L_1}{4e}p\bar{p}K_{c-1}\gamma_p^{c-2}\right\}.$$

We will show that the max is equal to $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}$ which will show eq. (A.42).

First we show that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}} \geq \frac{L_1}{4}p\sqrt{\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}$. We note that this equivalent with showing that $\frac{\frac{1}{4}pK_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16}p^2\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}} \geq 1$,

$$\begin{aligned}
&\frac{\frac{1}{4}pK_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16}p^2\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}} \\
&= \frac{4K_c}{L_1^2K_{c-1}}\cdot\frac{\gamma_p w(\Sigma^c)^{1/c}}{p\bar{p}w_\infty(\Sigma^c)^{1/c}} \\
&> \frac{4K_c}{L_1^2K_{c-1}}\cdot\frac{\gamma_p\left(\bar{p}e^{-2}K_{c-1}\gamma_p^{c-2}\right)^{1/(c-1)}m^{1/c}\left(\frac{m}{w_\infty(\Sigma^c)}\right)^{1/(c(2c-1))}}{\bar{p}^2w_\infty(\Sigma^c)^{1/c}} \\
&= \frac{4K_c}{L_1^2K_{c-1}}\frac{\left(e^{-2}K_{c-1}\right)^{1/(c-1)}}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\frac{m}{w_\infty(\Sigma^c)}\right)^{2/(2c-1)} \\
&\geq \frac{4K_c}{e^2L_1^2K_{c-1}^{1-1/(c-1)}}\frac{1}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\min_{x\in\Sigma^c}\frac{e^2m\,\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2/(2c-1)}
\end{aligned}$$

Now by using Lemma A.47 we get that

$$\begin{aligned}
&\frac{4K_c}{e^2L_1^2K_{c-1}^{1-1/(c-1)}}\frac{1}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\min_{x\in\Sigma^c}\frac{e^2m\,\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2/(2c-1)}\\
&\qquad\geq\frac{4K_c}{e^2L_1^2K_{c-1}^{1-1/(c-1)}}\left(\frac{2e}{(2-1/(c-1))(2c-1)}\right)^{2-1/(c-1)}\\
&\qquad\geq\frac{4K_c}{e^2L_1^2K_{c-1}^{1-1/(c-1)}}\left(\frac{e}{2c}\right)^2\\
&\qquad=\frac{K_c}{L_1^2K_{c-1}^{1-1/(c-1)}}c^{-2}\,.
\end{aligned}$$

Now $K_c=(L_2c)^c$ and $K_{c-1}\leq(L_2c)^{c-1}$ so we get that

$$\frac{K_c}{L_1^2K_{c-1}^{1-1/(c-1)}}c^{-2}\geq\frac{(L_2c)^c}{L_1^2\,(L_2c)^{c-2}}c^{-2}=\left(\frac{L_2}{L_1}\right)^2\geq 1\,.$$

The last inequality follows by choosing $L_2\geq L_1$. Combining it all we get that

$$\frac{\frac{1}{4}pK_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16}p^2\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}\geq 1\,.$$

This implies that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}\geq\frac{L_1}{4}p\sqrt{\bar{p}K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}}$ as we wanted.

Next we show that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}\geq\frac{L_1}{4e}p\bar{p}K_{c-1}\gamma_p^{c-2}$. Again we note that this equivalent with showing that $\frac{\frac{1}{4}K_cp\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16e^2}p^2\bar{p}^2K_{c-1}^2\gamma_p^{2c-4}}\geq 1$,

$$\begin{aligned}
&\frac{\frac{1}{4}K_cp\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16e^2}p^2\bar{p}^2K_{c-1}^2\gamma_p^{2c-4}}\\
&\qquad=\frac{4e^2K_c}{L_1^2K_{c-1}^2}\frac{w(\Sigma^c)}{mp\bar{p}^2\gamma_p^{c-3}}\\
&\qquad>\frac{4e^2K_c}{L_1^2K_{c-1}^2}\frac{\left(\bar{p}e^{-2}K_{c-1}\gamma_p^{c-2}\right)^{c/(c-1)}m\left(\frac{m}{w_\infty(\Sigma^c)}\right)^{1/(2c-1)}}{m\bar{p}^3\gamma_p^{c-3}}\\
&\qquad\geq\frac{4K_c}{eL_1^2K_{c-1}^{1-1/(c-1)}}\frac{1}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\min_{x\in\Sigma^c}\frac{e^2m\,\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{1/(2c-1)}\\
&\qquad\geq\frac{4K_c}{eL_1^2K_{c-1}^{1-1/(c-1)}}\frac{1}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\min_{x\in\Sigma^c}\frac{e^2m\,\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{1/(2c-1)}
\end{aligned}$$

Again we use Lemma A.47 to obtain that

$$\begin{aligned}
&\frac{4K_c}{eL_1^2K_{c-1}^{1-1/(c-1)}}\frac{1}{\log\left(\min_{x\in\Sigma^c}\frac{e^2m\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{2-1/(c-1)}}\left(\min_{x\in\Sigma^c}\frac{e^2m\,\|v[x]\|_2^2}{\|v[x]\|_1^2}\right)^{1/(2c-1)}\\
&\quad\geq \frac{4K_c}{eL_1^2K_{c-1}^{1-1/(c-1)}}\left(\frac{e}{(2c-1)(2-1/(c-1))}\right)^{2-1/(c-1)}\\
&\quad\geq \frac{4K_c}{eL_1^2K_{c-1}^{1-1/(c-1)}}\left(\frac{e}{4c}\right)^{2-1/(c-1)}\\
&\quad\geq \frac{4K_c}{L_1^2K_{c-1}^{1-1/(c-1)}}\left(\frac{1}{4c}\right)^{2-1/(c-1)}\\
&\quad\geq \frac{K_c}{4L_1^2K_{c-1}^{1-1/(c-1)}}c^{-2}\,.
\end{aligned}$$

Now $K_c = (L_2c)^c$ and $K_{c-1} \leq (L_2c)^{c-1}$ so we get that

$$\frac{K_c}{4L_1^2K_{c-1}^{1-1/(c-1)}}c^{-2} \geq \frac{(L_2c)^c}{4L_1^2\,(L_2c)^{c-2}}c^{-2} = \left(\frac{L_2}{2L_1}\right)^2 \geq 1\,.$$

The last inequality follows by choosing $L_2 \geq 2L_1$. Combining it all we get that

$$\frac{\frac{1}{4}K_cp\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}}{\frac{L_1^2}{16e^2}p^2\bar{p}^2K_{c-1}^2\gamma_p^{2c-4}} \geq 1\,.$$

This implies that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}} \geq \frac{L_1}{4e}p\bar{p}K_{c-1}\gamma_p^{c-2}$.

This proves eq. (A.42) and combining this with eq. (A.19) we get that

$$\begin{aligned}
&\Psi_p\left(L_1e\Psi_{\bar{p}}\left(K_{c-1}\gamma_p^{c-2}, K_{c-1}\frac{\gamma_p^{c-2}w(\Sigma^c)^{1-1/c}w_\infty(\Sigma^c)^{1/c}}{m}\right), K_c\gamma_p^{c-1}w(\Sigma^c)\right)\\
&\quad\leq \Psi_p\left(K_c\gamma_p^{c-1}, K_c\gamma_p^{c-1}\frac{w(\Sigma^c)}{m}\right)\\
&\quad= \Psi_p\left(K_c\gamma_p^{c-1}, K_c\gamma_p^{c-1}\sigma_v^2\right)\,.
\end{aligned} \tag{A.43}$$

Now combining eq. (A.34), eq. (A.35), eq. (A.39), and eq. (A.43) we get the result. □

A.4.2 Concentration Results for Mixed Tabulation Hashing

In this section, we will prove two different concentration results for mixed tabulation hashing. The first is a version of Khintchine's inequality for mixed tabulation hashing, and the proof is the simpler of the two. The other result is a strengthening of Theorem A.7 by using the strength of mixed tabulation hashing.

We will first introduce some notation.

Definition A.53. Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with d derived characters, and let $h_1\colon \Sigma^c \to [m]$, $h_2\colon \Sigma^c \to \Sigma$, and $h_3\colon \Sigma^d \to [m]$ be the three simple tabulation function defining h, i.e., $h(x) = h_1(x) \oplus h_3(h_2(x))$.

Let $\varepsilon_1\colon \Sigma^c \to \{-1,1\}$ and $\varepsilon_3\colon \Sigma^d \to \{-1,1\}$ be independent simple tabulation sign functions. We define $\varepsilon\colon \Sigma^c \to \{-1,1\}$ by

$$\varepsilon(x) = \varepsilon_1(x)\varepsilon_3(h_2(x)) \ .$$

We say that ε is a mixed tabulation sign function associated with h.

Theorem A.54. *Let $\varepsilon\colon \Sigma^c \to \{-1,1\}$ be a mixed tabulation sign function with $d \geq 1$ derived characters, and let $w\colon \Sigma^c \to \mathbb{R}$ be a weight function. For all $p \geq 2$ then,*

$$\left\| \sum_{x\in\Sigma^c} w(x)\varepsilon(x) \right\|_p \leq \sqrt{e} K_c \sqrt{p} \gamma_p^{c/2} \sqrt{\sum_{x\in\Sigma^c} w(x)^2} \tag{A.44}$$

Here K_c is as defined in Lemma A.18 and $\gamma_p = \max\left\{1, \frac{p}{\log(|\Sigma|)}\right\}$.

Proof. We will prove the result for $d = 1$. For $d > 1$ we fix the last $d-1$ derived characters and incorporate them into the weight function. This will only change the sign of the weight function for some keys, thus the result follows from the case with $d = 1$.

We let $\varepsilon_1\colon \Sigma^c \to \{-1,1\}$, $h\colon \Sigma^c \to \Sigma$, and $\varepsilon_2\colon \Sigma \to \{-1,1\}$ be the three simple tabulation functions used to define ε, i.e., $\varepsilon(x) = \varepsilon_1(x)\varepsilon_2(h(x))$.

We can now write,

$$\begin{aligned}\left\| \sum_{x\in\Sigma^c} w(x)\varepsilon(x) \right\|_p &= \left\| \sum_{x\in\Sigma^c} w(x)\varepsilon_1(x)\varepsilon_2(h(x)) \right\|_p \\ &= \left\| \sum_{\alpha\in\Sigma} \varepsilon_2(\alpha) \sum_{x\in\Sigma^c} w(x)\left[h(x)=\alpha\right]\varepsilon_1(x) \right\|_p .\end{aligned}$$

We fix h and ε_1 and use Lemma A.41 to get that

$$\begin{aligned}&\left\| \sum_{\alpha\in\Sigma} \varepsilon_2(\alpha) \sum_{x\in\Sigma^c} w(x)\left[h(x)=\alpha\right]\varepsilon_1(x) \;\middle|\; h, \varepsilon_1 \right\|_p \\ &\leq \sqrt{p}\sqrt{e\sum_{\alpha\in\Sigma}\left(\sum_{x\in\Sigma^c} w(x)\left[h(x)=\alpha\right]\varepsilon_1(x)\right)^2} \ .\end{aligned}$$

We define the value function $v\colon \Sigma^c \times \Sigma \to \mathbb{R}$ by $v(x,\alpha) = w(x)\left[\alpha = 0\right]$. We can then write,

$$\left\| \sqrt{\sum_{\alpha\in\Sigma}\left(\sum_{x\in\Sigma^c} w(x)\left[h(x)=\alpha\right]\varepsilon_1(x)\right)^2} \right\|_p = \left\| \sum_{\alpha\in\Sigma}\left(\sum_{x\in\Sigma^c} \varepsilon_1(x)V(x,\alpha\oplus h(x))\right)^2 \right\|_{p/2}^{1/2} .$$

Now we use Lemma A.18 to get that

$$\begin{aligned}\left\|\sum_{\alpha\in\Sigma}\left(\sum_{x\in\Sigma^c}\varepsilon_1(x)V(x,\alpha\oplus h(x))\right)^2\right\|_{p/2} &\leq K_c\gamma_{p/2}^c\sum_{x\in\Sigma^c}w(x)^2\\ &\leq K_c\gamma_p^c\sum_{x\in\Sigma^c}w(x)^2\,.\end{aligned}$$

Putting it all together, we get that

$$\begin{aligned}\left\|\sum_{x\in\Sigma^c}w(x)\varepsilon(x)\right\|_p &\leq \sqrt{ep}\left(K_c\gamma_p^c\sum_{x\in\Sigma^c}w(x)^2\right)^{1/2}\\ &= \sqrt{e}K_c\sqrt{p}\gamma_p^{c/2}\sqrt{\sum_{x\in\Sigma^c}w(x)^2}\,.\end{aligned}$$

□

Before proving the next result, we will first argue that we only need the symmetric case, similarly, as we did for simple tabulation.

Lemma A.55. *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with d derived characters and $v\colon \Sigma^c\times[m]\to\mathbb{R}$ a value function. Then for every $p\geq 2$,*

$$\left\|\sum_{x\in\Sigma^c}\Big(v(x,h(x))-\mathrm{E}[v(x,h(x))]\Big)\right\|_p \leq 2^{c+d}\left\|\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x))\right\|_p\,,$$

where $\varepsilon\colon\Sigma^c\to\{-1,1\}$ a mixed tabulation sign function associated with h.

Proof. The result follows by two uses Lemma A.48. Fixing h_2 and h_3 and using Lemma A.48 we get that

$$\begin{aligned}&\left\|\sum_{x\in\Sigma^c}\Big(v(x,h(x))-\mathrm{E}[v(x,h(x))]\Big)\;\middle|\;h_2,h_3\right\|_p\\ &\quad=\left\|\sum_{x\in\Sigma^c}\Big(v(x,h_1(x)\oplus h_3(h_2(x)))-\mathrm{E}[v(x,h_1(x)\oplus h_3(h_2(x)))]\Big)\;\middle|\;h_2,h_3\right\|_p\\ &\quad\leq 2^c\left\|\sum_{x\in\Sigma^c}\varepsilon_1(x)\Big(v(x,h_1(x)\oplus h_3(h_2(x)))-\mathrm{E}[v(x,h_1(x)\oplus h_3(h_2(x)))]\Big)\;\middle|\;h_2,h_3\right\|_p\,.\end{aligned}$$

Now we fix h_1, h_2, and ε_1 and use Lemma A.48,

$$\begin{aligned}
2^c \left\| \sum_{x\in\Sigma^c} \varepsilon_1(x) \left(v(x, h_1(x) \oplus h_3(h_2(x))) - \mathrm{E}[v(x, h_1(x) \oplus h_3(h_2(x)))] \right) \;\middle|\; h_1, h_2, \varepsilon_1 \right\|_p \\
\leq 2^{c+d} \left\| \sum_{x\in\Sigma^c} \varepsilon_1(x)\varepsilon_3(h) v(x, h_1(x) \oplus h_3(h_2(x))) \;\middle|\; h_1, h_2, \varepsilon_1 \right\|_p \\
= 2^{c+d} \left\| \sum_{x\in\Sigma^c} \varepsilon(x) v(x, h(x)) \;\middle|\; h_1, h_2, \varepsilon_1 \right\|_p .
\end{aligned}$$

□

We can now prove the concentration result for mixed tabulation. The proof is very similar to the proof of Theorem A.7 and is again quite long and technical.

Theorem A.10. *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with $d \geq 1$ derived characters, $v\colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j\in[m]} v(x, j) = 0$ for all keys $x \in \Sigma^c$. Define the random variable $V_v^{\text{mixed}} = \sum_{x\in\Sigma^c} v(x, h(x))$. For all $p \geq 2$ then*

$$\left\| V_v^{\text{mixed}} \right\|_p \leq \Psi_p \left(K_c \gamma_p^c M_v, K_c \gamma_p^c \sigma_v^2 \right) \tag{A.45}$$

where $K_c = L_1 (L_2 c)^c$, L_1 and L_2 are universal constants, and

$$\gamma_p = \max\left\{ 1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)} \right\} .$$

Proof. We will prove the result for $d = 1$. For $d > 1$ we fix the last $d-1$ derived characters and incorporate them into the value function. This does not change the variance and the result then follows from the case with $d = 1$.

We can assume without loss of generality that $M_v = 1$.

We let $h_1\colon \Sigma^c \to [m]$, $h_2\colon \Sigma^c \to \Sigma$, and $h_3\colon \Sigma \to [m]$ be the three simple tabulation functions used to define h, i.e., $h(x) = h_1(x) \oplus h_3(h_2(x))$. Similarly, we let $\varepsilon_1\colon \Sigma^c \to \{-1, 1\}$ and $\varepsilon_3\colon \Sigma \to \{-1, 1\}$ be the two simple tabulation sign functions used to define ε, i.e., $\varepsilon(x) = \varepsilon_1(x)\varepsilon_3(h_2(x))$.

We can then write,

$$\begin{aligned}
V_v^{\text{mixed}} &= \sum_{x\in\Sigma^c} \varepsilon(x) v(x, j \oplus h(x)) \\
&= \sum_{x\in\Sigma^c} \varepsilon_1(x)\varepsilon_3(h_2(x)) v(x, j \oplus h_1(x) \oplus h_3(h_2(x))) \\
&= \sum_{\alpha\in\Sigma} \varepsilon_3(\alpha) \sum_{x\in\Sigma^c} \varepsilon_1(x) [h_2(x) = \alpha] v(x, j \oplus h_1(x) \oplus h_3(\alpha)) .
\end{aligned}$$

We define the value function $v_h \colon \Sigma \times [m] \to \mathbb{R}$ by $v_h(\alpha, j) = \sum_{x \in \Sigma^c} \varepsilon_1(x) \, [h_2(x) = \alpha] \, v(x, j \oplus h_1(x))$. This allows us to write,

$$V_v^{\text{mixed}} = \sum_{\alpha \in \Sigma} \varepsilon_3(\alpha) v_h(\alpha, j \oplus h_3(\alpha)) \; .$$

If we fix h_1 and h_2 then Theorem A.5 give us that

$$\left\| V_v^{\text{mixed}} \;\middle|\; h_1, h_2 \right\|_p \leq 8e\Psi_p \left(\max_{\alpha \in \Sigma, j \in [m]} |v_h(\alpha, j)| \, , \frac{\sum_{\alpha \in \Sigma, j \in [m]} v_h(\alpha, j)^2}{m} \right) \; . \tag{A.46}$$

As in the proof of Theorem A.7 we will use Lemma A.15.

$$\begin{aligned} &\left\| \Psi_p \left(\max_{\alpha \in \Sigma, j \in [m]} |v_h(\alpha, j)| \, , \frac{\sum_{\alpha \in \Sigma, j \in [m]} v_h(\alpha, j)^2}{m} \right) \right\|_p \\ &\qquad \leq \sqrt{2} \Psi_p \left(\left\| \max_{\alpha \in \Sigma, j \in [m]} |v_h(\alpha, j)| \right\|_p , \frac{1}{m} \left\| \sum_{\alpha \in \Sigma, j \in [m]} v_h(\alpha, j)^2 \right\|_{p/2} \right) . \end{aligned} \tag{A.47}$$

Now we want to bound $\left\| \max_{\alpha \in \Sigma, j \in [m]} |v_h(\alpha, j)| \right\|_p$ and $\left\| \sum_{\alpha \in \Sigma, j \in [m]} v_h(\alpha, j)^2 \right\|_{p/2}$. We define the value function $v' \colon \Sigma^c \times ([m] \times \Sigma) \to \mathbb{R}$ by $v'(x, (j, \alpha)) = [\alpha = 0] \, v(x, j)$. We then get that

$$v_h(\alpha, j) = \sum_{x \in \Sigma^c} \varepsilon_1(x) v'(x, (j \oplus h_1(x), \alpha \oplus h_2(x))) \; .$$

Clearly, we have that the support of v' is at most m for all $x \in \Sigma^c$, thus $\frac{\|v'[x]\|_1^2}{\|v'[x]\|_2^2} \leq m$ for all $x \in \Sigma^c$.

We define $\bar{p} = \max\{q, \log(m \, |\Sigma|)\}$. This implies that $\gamma_p = \frac{\bar{p}}{\log(e|\Sigma|)}$.

We can now bound the moments of $\max_{\alpha\in\Sigma,j\in[m]}|v_h(\alpha,j)|$ by using Theorem A.7.

$$\begin{aligned}
\left\|\max_{\alpha\in\Sigma,j\in[m]}|v_h(\alpha,j)|\right\|_p &\le \left\|\max_{\alpha\in\Sigma,j\in[m]}|v_h(\alpha,j)|\right\|_{\bar{p}} \\
&\le \left(\sum_{\alpha\in\Sigma,j\in[m]}\|v_h(\alpha,j)\|_{\bar{p}}^{\bar{p}}\right)^{1/\bar{p}} \\
&\le \left(\sum_{\alpha\in\Sigma,j\in[m]}\left\|\sum_{x\in\Sigma^c}\varepsilon_1(x)v'(x,(j\oplus h_1(x),\alpha\oplus h_2(x)))\right\|_{\bar{p}}^{\bar{p}}\right)^{1/\bar{p}} \\
&\le \left(\sum_{\alpha\in\Sigma,j\in[m]} M_1^{\bar{p}}\Psi_{\bar{p}}\left(L_c^{(1)}\left(\gamma_p'\right)^{c-1},L_c^{(1)}\left(\gamma_p'\right)^{c-1}\sigma_{v'}^2\right)^{\bar{p}}\right)^{1/\bar{p}} \\
&= (m\,|\Sigma|)^{1/\bar{p}}\,M_1\Psi_{\bar{p}}\left(L_c^{(1)}\left(\gamma_p'\right)^{c-1},L_c^{(1)}\left(\gamma_p'\right)^{c-1}\sigma_{v'}^2\right) \\
&= (m\,|\Sigma|)^{1/\bar{p}}\,M_1\Psi_{\bar{p}}\left(L_c^{(1)}\left(\gamma_p'\right)^{c-1},L_c^{(1)}\left(\gamma_p'\right)^{c-1}\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{m\,|\Sigma|}\right) \\
&\le eM_1\Psi_{\bar{p}}\left(L_c^{(1)}\left(\gamma_p'\right)^{c-1},L_c^{(1)}\left(\gamma_p'\right)^{c-1}\frac{\sigma_v^2}{|\Sigma|}\right)\ ,
\end{aligned}$$

where $L_c^{(1)}$ is a constant depending on c as given by Theorem A.7, M_1 is universal constant, and

$$\gamma_p' = \max\left\{\frac{\log(m\,|\Sigma|)+\log\left(\frac{\sum_{x\in\Sigma^c}\|v'[x]\|_2^2}{\max_{x\in\Sigma^c}\|v'[x]\|_2^2}\right)/c}{\log(e\,|\Sigma|)},\frac{p}{\log\left(e\,|\Sigma|\right)}\right\}$$

We note that $\gamma_p' \le 2\gamma_p$ since,

$$\begin{aligned}
\gamma_p' &= \max\left\{\frac{\log(m\,|\Sigma|)+\log\left(\frac{\sum_{x\in\Sigma^c}\|v'[x]\|_2^2}{\max_{x\in\Sigma^c}\|v'[x]\|_2^2}\right)/c}{\log(e\,|\Sigma|)},\frac{p}{\log(e\,|\Sigma|)}\right\} \\
&\le \max\left\{\frac{\log(m\,|\Sigma|)+\log(m\,|\Sigma|^c)/c}{\log\left(e\,|\Sigma|\right)},\frac{p}{\log\left(e\,|\Sigma|\right)}\right\} \\
&\le 2\max\left\{\frac{\log(m\,|\Sigma|)}{\log\left(e\,|\Sigma|\right)},\frac{p}{\log\left(e\,|\Sigma|\right)}\right\} \\
&= 2\gamma_p\ .
\end{aligned}$$

So we have that

$$\left\|\max_{\alpha\in\Sigma,j\in[m]}|v_h(\alpha,j)|\right\|_p \le eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1},2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right)\ . \tag{A.48}$$

We bound the moments of $\sum_{\alpha\in\Sigma,j\in[m]} v_h(\alpha,j)^2$ by using Lemma A.18. We get that for all $q \geq 2$,

$$\begin{aligned}\left\|\sum_{\alpha\in\Sigma,j\in[m]} v_h(\alpha,j)^2\right\|_p &= \left\|\sum_{\alpha\in\Sigma,j\in[m]}\left(\sum_{x\in\Sigma^c} v'(x,(j\oplus h_1(x),\alpha\oplus h_2(x)))\right)^2\right\|_p \\ &\leq L_c^{(2)}\max\left\{1,\left(\frac{p}{\log(e\,|\Sigma|)}\right)^c\right\}\sum_{x\in\Sigma^c}\left\|v'[x]\right\|_2^2 \\ &= L_c^{(2)}\gamma_p^c\sum_{x\in\Sigma^c}\|v[x]\|_2^2 \\ &\leq K_c\gamma_p^c\sum_{x\in\Sigma^c}\|v[x]\|_2^2\ .\end{aligned} \tag{A.49}$$

Where $L_c^{(2)}$ is constant depending on c as given by Lemma A.18.

Now combining eq. (A.47), eq. (A.48), and eq. (A.49), and we get that

$$\begin{aligned}&\left\|\Psi_p\left(\max_{\alpha\in\Sigma,j\in[m]}|v_h(\alpha,j)|\,,\frac{\sum_{\alpha\in\Sigma,j\in[m]} v_h(\alpha,j)^2}{m}\right)\right\|_p \\ &\qquad\leq e\Psi_p\left(eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1},2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right),K_c\gamma_p^c\sigma^2\right)\ .\end{aligned} \tag{A.50}$$

We will now consider three different cases.

Case 1. $\sigma^2 \leq (2e^{-3}L_c^{(1)})\bar{p}\gamma_p^{c-1}\left(e\,|\Sigma|\right)^{1-\frac{1}{4(2c+1)}}$**.** In this case we will show that

$$eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1},2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right)\leq K_c\gamma_p^c\,. \tag{A.51}$$

We first see that

$$\begin{aligned}\frac{2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}{\left(2L_c^{(1)}\gamma_p^{c-1}\right)^2} &= \frac{\sigma^2}{2L_c^{(1)}\gamma_p^{c-1}\,|\Sigma|} \\ &\leq \frac{(2e^{-3}L_c^{(1)})\bar{p}\gamma_p^{c-1}\left(e\,|\Sigma|\right)^{1-\frac{1}{4(2c+1)}}}{2L_c^{(1)}\gamma_p^{c-1}\,|\Sigma|} \\ &\leq e^{-2}\bar{p}\,|\Sigma|^{-\frac{1}{4(2c+1)}} \\ &\leq e^{-2}\bar{p}\,.\end{aligned} \tag{A.52}$$

By eq. (A.20) we get that

$$\begin{aligned} eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right) &\leq eM_1\frac{\bar{p}}{\log\left(\frac{\bar{p}\left(2L_c^{(1)}\gamma_p^{c-1}\right)^2}{2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}\right)}2L_c^{(1)}\gamma_p^{c-1} \\ &\leq 2eM_1L_c^{(1)}\frac{\bar{p}}{\log\left(e^2\left|\Sigma\right|^{\frac{1}{4(2c+1)}}\right)}\gamma_p^{c-1} \\ &\leq 8eM_1L_c^{(1)}(2c+1)\gamma_p^c \\ &\leq 24ecM_1L_c^{(1)}\gamma_p^c \\ &\leq K_c\gamma_p^c\,. \end{aligned}$$

Here we have used that $\frac{\bar{p}\left(2L_c^{(1)}\gamma_p^{c-1}\right)^2}{2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}} \geq e^2\left|\Sigma\right|^{\frac{1}{4(2c+1)}}$ which follows from eq. (A.52) and that $K_c \geq 24ecM_1L_c^{(1)}$.

Now combining eq. (A.46), eq. (A.50), and eq. (A.51) we get the result.

Case 2. $\sigma^2 > (2e^{-3}L_c^{(1)})\bar{p}\gamma_p^{c-1}\left(e\left|\Sigma\right|\right)^{1-\frac{1}{4(2c+1)}}$ **and** $p \leq \left|\Sigma\right|^{1-\frac{1}{2(2c+1)}}$**.** We will show that

$$\Psi_p\left(eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right), K_c\gamma_p^c\sigma^2\right) \leq \sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2}\,. \tag{A.53}$$

We use eq. (A.18) to get that

$$\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right) \leq \max\left\{\tfrac{1}{2}\sqrt{\bar{p}2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}, \tfrac{L_c^{(1)}}{e}\bar{p}\gamma_p^{c-1}\right\}\,.$$

We again apply eq. (A.18) to obtain that

$$\begin{aligned} &\Psi_p\left(eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right), K_c\gamma_p^c\sigma^2\right) \\ &\qquad \leq \max\left\{\tfrac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2}, \tfrac{M_1}{2}\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right)\right\}\,. \end{aligned}$$

Combining the two estimates give us that

$$\begin{aligned} &\Psi_p\left(eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right), K_c\gamma_p^c\sigma^2\right) \\ &\qquad \leq \max\left\{\tfrac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2}, \tfrac{M_1\sqrt{L_c^{(1)}}}{2\sqrt{2}}p\sqrt{\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}, \tfrac{M_1L_c^{(1)}}{2e}p\bar{p}\gamma_p^{c-1}\right\} \end{aligned}$$

We will show that the max is equal to $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2}$ which will show eq. (A.53).

First we show that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2} \geq \frac{M_1\sqrt{L_c^{(1)}}}{2\sqrt{2}}p\sqrt{\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}$. We note that this equivalent with showing that $\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2L_c^{(1)}}{8}p^2\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}} \geq 1$,

$$\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2L_c^{(1)}}{8}p^2\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}} = \frac{2K_c}{M_1^2L_c^{(1)}}\cdot\frac{\gamma_p|\Sigma|}{p\bar{p}} = \frac{2K_c}{M_1^2L_c^{(1)}}\cdot\frac{|\Sigma|}{p\log(e|\Sigma|)} \geq \frac{2K_c}{M_1^2L_c^{(1)}}\cdot\frac{|\Sigma|^{\frac{1}{2(2c+1)}}}{\log(e|\Sigma|)}.$$

Now by using Lemma A.47 we get that

$$\frac{2K_c}{M_1^2L_c^{(1)}}\cdot\frac{|\Sigma|^{\frac{1}{2(2c+1)}}}{\log(e|\Sigma|)} \geq \frac{2K_c}{eM_1^2L_c^{(1)}}\cdot\frac{(e|\Sigma|)^{\frac{1}{2(2c+1)}}}{\log(e|\Sigma|)} \geq \frac{2K_c}{eM_1^2L_c^{(1)}}\left(\frac{e}{2(2c+1)}\right) = \frac{K_c}{3M_1^2L_c^{(1)}c}.$$

Now $K_c = (L_2c)^c$ and $L_c^{(1)} = (T_1c)^c$ where T_1 is universal constant determined by Theorem A.7. Now choosing L_2 large enough we get that so we get that

$$\frac{K_c}{3M_1^2L_c^{(1)}c} = \frac{L_2^c}{3M_1^2T_1^{c-1}} \geq 1.$$

Combining it all we get that

$$\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2L_c^{(1)}}{8}p^2\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}} \geq 1.$$

This implies that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2} \geq \frac{M_1\sqrt{L_c^{(1)}}}{2\sqrt{2}}p\sqrt{\bar{p}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}}$ as we wanted.

Next we show that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2} \geq \frac{M_1L_c^{(1)}}{2e}p\bar{p}\gamma_p^{c-1}$. Again we note that this equivalent with showing that $\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2\left(L_c^{(1)}\right)^2}{4e^2}p^2\bar{p}^2\gamma_p^{2c-2}} \geq 1$,

$$\begin{aligned}
\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2\left(L_c^{(1)}\right)^2}{4e^2}p^2\bar{p}^2\gamma_p^{2c-2}} &= \frac{e^2K_c}{\left(M_1L_c^{(1)}\right)^2}\frac{\sigma^2}{p\bar{p}^2\gamma_p^{c-2}} \\
&\geq \frac{e^2K_c}{\left(M_1L_c^{(1)}\right)^2}\frac{(2e^{-3}L_c^{(1)})\bar{p}\gamma_p^{c-1}(e|\Sigma|)^{1-\frac{1}{4(2c+1)}}}{p\bar{p}^2\gamma_p^{c-2}} \\
&\geq \frac{2K_c}{eM_1^2L_c^{(1)}}\frac{(e|\Sigma|)^{1-\frac{1}{4(2c+1)}}}{p\log(e|\Sigma|)} \\
&\geq \frac{2K_c}{eM_1^2L_c^{(1)}}\frac{(e|\Sigma|)^{\frac{1}{4(2c+1)}}}{\log(e|\Sigma|)}.
\end{aligned}$$

Again we use Lemma A.47 to obtain that

$$\frac{2K_c}{eM_1^2L_c^{(1)}}\frac{(e\,|\Sigma|)^{\frac{1}{4(2c+1)}}}{\log(e\,|\Sigma|)} \geq \frac{2K_c}{eM_1^2L_c^{(1)}}\left(\frac{e}{4(2c+1)}\right) \geq \frac{K_c}{6eM_1^2L_c^{(1)}c}\,.$$

Now $K_c = (L_2c)^c$ and $L_c^{(1)} = (T_1c)^c$ where T_1 is universal constant determined by Theorem A.7. Now choosing L_2 large enough we get that so we get that

$$\frac{K_c}{6eM_1^2L_c^{(1)}c} = \frac{L_2^c}{6eM_1^2T_1^{c-1}} \geq 1\,.$$

Combining it all we get that

$$\frac{\frac{K_c}{4}p\gamma_p^c\sigma^2}{\frac{M_1^2\left(L_c^{(1)}\right)^2}{4e^2}p^2\bar{p}^2\gamma_p^{2c-2}} \geq 1\,.$$

This implies that $\frac{1}{2}\sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2} \geq \frac{M_1L_c^{(1)}}{2e}p\bar{p}\gamma_p^{c-1}$.

This proves eq. (A.53) and combining this with eq. (A.19) we get that

$$\Psi_p\left(eM_1\Psi_{\bar{p}}\left(2L_c^{(1)}\gamma_p^{c-1}, 2L_c^{(1)}\gamma_p^{c-1}\frac{\sigma^2}{|\Sigma|}\right), K_c\gamma_p^c\sigma^2\right) \leq \Psi_p\left(K_c\gamma_p^c, K_c\gamma_p^c\sigma^2\right)\,. \tag{A.54}$$

Now combining eq. (A.46), eq. (A.50), and eq. (A.54) we get the result.

Case 3. $\sigma^2 > (2e^{-3}L_c^{(1)})\bar{p}\gamma_p^{c-1}\,(e\,|\Sigma|)^{1-\frac{1}{4(2c+1)}}$ **and** $p > (e\,|\Sigma|)^{1-\frac{1}{2(2c+1)}}$**.** In this case we will exploit that $\left|V_v^{\text{mixed}}\right| \leq |\Sigma|^c$. This implies that $\left\|V_v^{\text{mixed}}\right\|_p \leq |\Sigma|^c$, so if we can prove that

$$L_1\Psi_p\left(K_c\gamma_p^c, K_c\gamma_p^c\sigma^2\right) \geq |\Sigma|^c\,, \tag{A.55}$$

then the result follows.

We use eq. (A.19) to get that

$$\begin{aligned}
\Psi_p\left(K_c\gamma_p^c, K_c\gamma_p^c\sigma^2\right) &\geq \sqrt{p}\sqrt{K_c\gamma_p^c\sigma^2} \\
&\geq \sqrt{p^2K_c(2e^{-3}L_c^{(1)})\gamma_p^{2c-1}\,(e\,|\Sigma|)^{1-\frac{1}{4(2c+1)}}} \\
&\geq \sqrt{2e^{-3}K_cL_c^{(1)}}\frac{(e\,|\Sigma|)^{\frac{2c+1}{2}\cdot\left(1-\frac{1}{2(2c+1)}\right)}}{\log(e\,|\Sigma|)^{\frac{2c-1}{2}}} \\
&\geq \sqrt{2e^{-3}K_cL_c^{(1)}}\frac{(e\,|\Sigma|)^{c+\frac{1}{4}}}{\log(e\,|\Sigma|)^{\frac{2c-1}{2}}}
\end{aligned}$$

Now we use Lemma A.47 to get that

$$\frac{(e\,|\Sigma|)^{1/4}}{\log(e\,|\Sigma|)^{\frac{2c-1}{2}}} \geq \left(\frac{e}{2(2c-1)}\right)^{\frac{2c-1}{2}} \geq \left(\frac{e}{4c}\right)^{\frac{2c-1}{2}} .$$

Combining this we get that

$$\Psi_p\left(K_c\gamma_p^c, K_c\gamma_p^c\sigma^2\right) \geq \sqrt{2e^{-3}K_cL_c^{(1)}}\,(e\,|\Sigma|)^c\left(\frac{e}{4c}\right)^{\frac{2c-1}{2}} \geq \sqrt{2e^{-3}}\,|\Sigma|^c\sqrt{K_cL_c^{(1)}\left(\frac{e}{4c}\right)^{2c-1}} .$$

We have that $K_cL_c^{(1)} = c^{2c-1}T_1^cT_2^{c-1}$ where T_1 is a universal constant and T_2 is a universal determined by Theorem A.7. Choosing T_1 large enough we get that $\sqrt{K_cL_c^{(1)}\left(\frac{e}{4c}\right)^{2c-1}} \geq \frac{1}{L_1}$. Thus eq. (A.55) follows and the result is proven. □

A.5 Lower Bounds

Similarly, as in the proof of Lemma A.4, we will use the following result by Latała [Lat97] that gives a tight bound for p-norms of sums of independent and identically distributed symmetric variables.

Lemma A.56 (Latała [Lat97])**.** *If $X, (X_i)_{i\in[n]}$ are independent and identically distributed symmetric variables and $p \geq 2$ then,*

$$\left\|\sum_{i\in[n]} X_i\right\|_p \leq K_1 \sup\left\{\frac{p}{s}\left(\frac{n}{p}\right)^{1/s}\|X\|_s \,\middle|\, \max\{2, \tfrac{p}{n}\} \leq s \leq p\right\} ,$$

and

$$\left\|\sum_{i\in[n]} X_i\right\|_p \geq K_2 \sup\left\{\frac{p}{s}\left(\frac{n}{p}\right)^{1/s}\|X\|_s \,\middle|\, \max\{2, \tfrac{p}{n}\} \leq s \leq p\right\} .$$

Here K_1 and K_2 are universal constants.

We start by proving the lower bound for uniformly random hash functions which will serve as a base for the other lower bounds. We show the following lemma.

Lemma A.57. *Let $h\colon U \to [m]$ be a uniformly random function and let $v\colon U \times [m] \to \mathbb{R}$ be a value function defined by,*

$$v(x,j) = \begin{cases} 1 - \frac{1}{m} & \text{if } j = 0 \\ -\frac{1}{m} & \text{otherwise} \end{cases} ,$$

for all $x \in U$. Then for all $2 \leq p \leq L_1\,|U|\log(m)$,

$$\left\|\sum_{x\in U} v(x, h(x))\right\|_p \geq L_2\Psi_p\left(M_v, \sigma_v^2\right) ,$$

where L_1 and L_2 is a universal constant.

It is easy to check that this implies Theorem A.9.

Proof. We will consider the value function, $v\colon U \times [m] \to \mathbb{R}$, defined by,

$$v(x,j) = \begin{cases} 1 - \frac{1}{m} & \text{if } j = 0 \\ -\frac{1}{m} & \text{otherwise} \end{cases},$$

We then have that $\sigma_v^2 = |U| \frac{1}{m}\left(1 - \frac{1}{m}\right)$ and $M_v = 1$. The goal is then to prove that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_p \geq L_2 \Psi_p(1, \sigma_v^2) \,. \tag{A.56}$$

We then use Lemma A.37 to get that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_p \geq \frac{1}{2} \left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p ,$$

where $\varepsilon\colon U \to \{-1, 1\}$ is uniformly random sign function. We can now use Lemma A.56,

$$\left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p \geq K_2 \sup \left\{ \frac{p}{s} \left(\frac{|U|}{p} \right)^{1/s} \|v(x, h(x))\|_s \,\middle|\, \max\left\{2, \tfrac{p}{|U|}\right\} \leq s \leq p \right\} .$$

We will argue that $\|v(x,h(x))\|_s \geq \frac{1}{2}\left(\frac{1}{m}\left(1 - \frac{1}{m}\right)\right)^{1/s}$ for all $s \geq 2$,

$$\begin{aligned} \|v(x,h(x))\|_s &= \left(\tfrac{1}{m}\left(1 - \tfrac{1}{m}\right)^s + \left(1 - \tfrac{1}{m}\right)\left(\tfrac{1}{m}\right)^s\right)^{1/s} \\ &= \left(\tfrac{1}{m}\left(1 - \tfrac{1}{m}\right)\right)^{1/s} \left(\left(1 - \tfrac{1}{m}\right)^{s-1} + \left(\tfrac{1}{m}\right)^{s-1}\right)^{1/s} \\ &\geq \left(\tfrac{1}{m}\left(1 - \tfrac{1}{m}\right)\right)^{1/s} \left(\max\left\{\left(1 - \tfrac{1}{m}\right)^{s-1}, \left(\tfrac{1}{m}\right)^{s-1}\right\}\right)^{1/s} \\ &\geq \tfrac{1}{2}\left(\tfrac{1}{m}\left(1 - \tfrac{1}{m}\right)\right)^{1/s} . \end{aligned}$$

This implies that

$$\begin{aligned} \left\| \sum_{x \in U} \varepsilon(x) v(x, h(x)) \right\|_p &\geq \frac{K_2}{2} \sup \left\{ \frac{p}{s} \left(\frac{|U| \frac{1}{m}\left(1 - \frac{1}{m}\right)}{p} \right)^{1/s} \,\middle|\, \max\left\{2, \tfrac{p}{|U|}\right\} \leq s \leq p \right\} \\ &= \frac{K_2}{2} \sup \left\{ \frac{p}{s} \left(\frac{\sigma_v^2}{p} \right)^{1/s} \,\middle|\, \max\left\{2, \tfrac{p}{|U|}\right\} \leq s \leq p \right\} . \end{aligned}$$

It is easy to see that the function $s \mapsto \frac{p}{s}\left(\frac{\sigma_v^2}{p}\right)^{1/s}$ is maximized at $s^* = \log\left(\frac{p}{\sigma_v^2}\right)$. It is easy to check that $\log\left(\frac{p}{\sigma_v^2}\right) \geq \frac{p}{|U|}$ when $p \leq L_1 |U| \log(m)$ for a sufficiently large universal

constant L_1. We then get that

$$\begin{aligned}\left\|\sum_{x\in U}\varepsilon(x)v(x,h(x))\right\|_p &\geq \frac{K_2}{2}\sup\left\{\frac{p}{s}\left(\frac{\sigma_v^2}{p}\right)^{1/s}\middle|\max\left\{2,\tfrac{p}{|U|}\right\}\leq s\leq p\right\}\\ &= \frac{K_2}{2}\sup\left\{\frac{p}{s}\left(\frac{\sigma_v^2}{p}\right)^{1/s}\middle|2\leq s\leq p\right\}\\ &= \frac{K_2}{2}\Psi_p(1,\sigma_v^2)\ .\end{aligned}$$

The last equality follows by eq. (A.17). This proves eq. (A.56) and finishes the proof. □

Now we are ready to prove the lower bound for simple tabulation hashing.

Theorem A.9. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, and $2 \leq p \leq L_1 |\Sigma| \log(m)$, then there exists a value function, $v\colon U\times[m]\to\mathbb{R}$, where $\sum_{j\in[m]} v(x,j)=0$ for all keys $x\in\Sigma^c$, and for which*

$$\left\|\sum_{x\in\Sigma^c} v(x,h(x))\right\|_p \geq K_c'\Psi_p\left(\gamma_p^{c-1}M_v,\gamma_p^{c-1}\sigma_v^2\right)\ ,$$

where $K_c'=L_1^c$ and L_1 is a universal constant, and

$$\gamma_p=\max\left\{1,\frac{p}{\log\left(e^2m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)}\right\}$$

Proof. We will define a value function, $v\colon \Sigma^c\times[m]\to\mathbb{R}$, for which $\min_{x\in\Sigma^c}\frac{\|v[x]\|_2^2}{\|v[x]\|_1^2}=\frac{1}{4\left(1-\frac{1}{m}\right)}$. We then have that

$$\gamma_p=\max\left\{1,\frac{p}{\log\left(\frac{e^2m}{4\left(1-\frac{1}{m}\right)}\right)}\right\}\ .$$

If $\gamma_p=1$ then the result follow by Theorem A.6, so we assume that $\gamma_p>1$. We let $S=[1+\lfloor\gamma_p\rfloor]^{c-1}\times\Sigma$ and formally define the value function, $v\colon\Sigma^c\times[m]\to\mathbb{R}$, by,

$$v(x,j)=\begin{cases}0 & \text{if } x\notin S\\ 1-\frac{1}{m} & \text{if } x\in S \text{ and } j=0\\ -\frac{1}{m} & \text{otherwise}\end{cases}\ .$$

For every $i \in [c-1]$, we define A_i to be the event that $T(i,\alpha) = T(i,0)$ for all $\alpha \in [1+\lfloor \gamma_p \rfloor]$. We note that

$$\Pr[A_i] = \left(\frac{1}{m}\right)^{\lfloor \gamma_p \rfloor} \geq \left(\frac{1}{m}\right)^{\gamma_p} = \exp\left(-p\frac{\log(m)}{\log\left(\frac{e^2 m}{4\left(1-\frac{1}{m}\right)}\right)}\right) \geq e^{-p} .$$

We then get that $\Pr\left[\bigwedge_{i\in[c-1]} A_i\right] \geq e^{-p(c-1)}$.

If $\bigwedge_{i\in[c-1]} A_i$ happens then we can set $j^* = \bigoplus_{i\in[c-1]} T(i,0)$. Now we define the value function $v' \colon \Sigma \times [m] \to \mathbb{R}$, by

$$v'(\alpha, j) = \begin{cases} 1 - \frac{1}{m} & \text{if } j = j^* \\ -\frac{1}{m} & \text{otherwise} \end{cases} .$$

We then get that

$$\sum_{x\in\Sigma^c} v(x,h(x)) = \sum_{x\in S} v(x,h(x)) = \sum_{\alpha\in\Sigma} (1+\lfloor\gamma_p\rfloor)^{c-1} v'(\alpha, T(c-1,\alpha)) .$$

This implies that

$$\begin{aligned}
&\left\| \sum_{x\in\Sigma^c} v(x,h(x)) \right\|_p \\
&\quad = \left\| \sum_{x\in S} v(x,h(x)) \right\|_p \\
&\quad = \left(\mathrm{E}\left[\left[\bigwedge_{i\in[c-1]} A_i \right] \left(\sum_{x\in S} v(x,h(x)) \right)^p \right] \right. \\
&\qquad \left. + \mathrm{E}\left[\left[\left(\bigwedge_{i\in[c-1]} A_i \right)^C \right] \left(\sum_{x\in S} v(x,h(x)) \right)^p \right] \right)^{1/p} \\
&\quad \geq \left\| \left[\bigwedge_{i\in[c-1]} A_i \right] \sum_{x\in S} v(x,h(x)) \right\|_p \\
&\quad = \Pr\left[\bigwedge_{i\in[c-1]} A_i \right]^{1/p} \mathrm{E}\left[\left(\sum_{\alpha\in\Sigma} (1+\lfloor\gamma_p\rfloor)^{c-1} v'(\alpha, T(c-1,\alpha)) \right)^p \,\middle|\, \bigwedge_{i\in[c-1]} A_i \right]^{1/p} \\
&\quad \geq e^{-c} (1+\lfloor\gamma_p\rfloor)^{c-1} \mathrm{E}\left[\left(\sum_{\alpha\in\Sigma} v'(\alpha, T(c-1,\alpha)) \right)^p \,\middle|\, \bigwedge_{i\in[c-1]} A_i \right]^{1/p} .
\end{aligned}$$

Now we use Theorem A.6 to get that

$$\left\| \sum_{\alpha \in \Sigma} v'(\alpha, T(c-1,\alpha)) \,\middle|\, \bigwedge_{i \in [c-1]} A_i \right\|_p \geq L\Psi_p(1, \sigma_{v'}^2) \ ,$$

where $\sigma_{v'}^2 = \frac{1}{m}\left(1 - \frac{1}{m}\right)|\Sigma|$. Combining it all we have that

$$\begin{aligned}
\left\| \sum_{x \in \Sigma^c} v(x, h(x)) \right\|_p &\geq e^{-c} \left(1 + \lfloor \gamma_p \rfloor\right)^{c-1} L\Psi_p\left(1, \frac{1}{m}\left(1 - \frac{1}{m}\right)|\Sigma|\right) \\
&= e^{-c} L\Psi_p\left(\left(1 + \lfloor \gamma_p \rfloor\right)^{c-1}, \left(1 + \lfloor \gamma_p \rfloor\right)^{2(c-1)} \frac{1}{m}\left(1 - \frac{1}{m}\right)|\Sigma|\right) \\
&\geq e^{-c} L\Psi_p\left(\gamma_p^{c-1}, \gamma_p^{c-1} \frac{1}{m}\left(1 - \frac{1}{m}\right)|\Sigma| \left(1 + \lfloor \gamma_p \rfloor\right)^{c-1}\right) \ .
\end{aligned}$$

The equality follows since $\Psi_p(\lambda M, \lambda^2 \sigma^2) = \lambda \Psi_p(M, \sigma^2)$. We have that $\frac{1}{m}\left(1 - \frac{1}{m}\right)|\Sigma|\left(1 + \lfloor \gamma_p \rfloor\right)^{c-1} = \sigma_v^2$ so this finishes the proof. □

A.6 Adding a Query Element

The goal of the section is to prove that the result holds even when you condition on a query element. We start by proving the result for simple tabulation.

Theorem A.12. *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function and let $q \in \Sigma^c$ be a designated query element. Let $v\colon \Sigma^c \times [m] \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j \in [m]} v(x,j,k) = 0$ for all keys $x \in U$ and all $k \in [m]$. Define the random variable $V_{v,q}^{\text{simple}} = \sum_{x \in \Sigma^c \setminus \{q\}} v(x, h(x), h(q))$ and the random variables*

$$\begin{aligned}
M_{v,q} &= \max_{x \in \Sigma^c \setminus \{q\}, j \in [m]} |v(x, j, h(q))| \ , \\
\sigma_{v,q}^2 &= \frac{1}{m} \sum_{x \in \Sigma^c \setminus \{q\}} \sum_{j \in [m]} v(x, j, h(q))^2 \ ,
\end{aligned}$$

which only depend on the randomness of $h(q)$. Then for all $p \geq 2$

$$\mathrm{E}\left[\left(V_{v,q}^{\text{simple}}\right)^p \,\middle|\, h(q)\right]^{1/p} \leq \Psi_p\left(K_c \gamma_p^{c-1} M_{v,q}, K_c \gamma_p^{c-1} \sigma_{v,q}^2\right) \ ,$$

where $K_c = L_1 \left(L_2 c\right)^{c-1}$, L_1 and L_2 are universal constants, and

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x \in \Sigma^c} \|v[x]\|_2^2}{\max_{x \in \Sigma^c} \|v[x]\|_2^2}\right)/c, p\right\}}{\log\left(e^2 m \left(\max_{x \in \Sigma^c} \frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)} \ .$$

Proof. We start by defining a partition of $\Sigma^c \setminus \{q\}$. Write $q = (\alpha_0, \ldots, \alpha_{c-1})$ and for $I \subsetneq [c]$, we define $G_I \subseteq \Sigma^c$ by

$$G_I = \{x \in \Sigma^c \setminus \{q\} \mid \forall i \in I : (i, \alpha_i) \in x \wedge \forall i \notin I : (i, \alpha_i) \notin x\}$$

This clearly gives a partition of $\Sigma^c \setminus \{q\}$. Using this we obtain,

$$\begin{aligned}
\left\| V_{v,q}^{\text{simple}} \;\middle|\; h(q) \right\|_p &= \left\| \sum_{x \in \Sigma^c \setminus \{q\}} v(x, h(x), h(q)) \;\middle|\; h(q) \right\|_p \\
&= \left\| \left\| \sum_{x \in \Sigma^c \setminus \{q\}} v(x, h(x), h(q)) \;\middle|\; (h((i, \alpha_i)))_{i \in [c]} \right\|_p \;\middle|\; h(q) \right\|_p \\
&\leq \left\| \sum_{I \subsetneq [c]} \left\| \sum_{x \in G_I} v(x, h(x), h(q)) \;\middle|\; (h((i, \alpha_i)))_{i \in [c]} \right\|_p \;\middle|\; h(q) \right\|_p .
\end{aligned}$$

Now using Theorem A.7 we get that,

$$\begin{aligned}
&\left\| \sum_{I \subsetneq [c]} \left\| \sum_{x \in G_I} v(x, h(x), h(q)) \;\middle|\; (h((i, \alpha_i)))_{i \in [c]} \right\|_p \;\middle|\; h(q) \right\|_p \\
&\leq \left\| \sum_{I \subsetneq [c]} \Psi_p \left(K_{c-|I|} \gamma_p^{c-1-|I|} M_{v,q}, K_{c-|I|} \gamma_p^{c-1-|I|} \tfrac{1}{m} \sum_{x \in G_I} \sum_{j \in [m]} v(x, j, h(q)) \right) \;\middle|\; h(q) \right\|_p \\
&= \sum_{I \subsetneq [c]} \Psi_p \left(K_{c-|I|} \gamma_p^{c-1-|I|} M_{v,q}, K_{c-|I|} \gamma_p^{c-1-|I|} \tfrac{1}{m} \sum_{x \in G_I} \sum_{j \in [m]} v(x, j, h(q)) \right) \\
&\leq \Psi_p \left(2^{2c} K_c \gamma_p^{c-1} M_{v,q}, 2^{2c} K_c \gamma_p^{c-1} \sigma_{v,q}^2 \right) .
\end{aligned}$$

The last bound follows by using eq. (A.22) from Lemma A.32. □

We now prove the result for mixed tabulation when conditioning on a query element.

Theorem A.13. *Let $h \colon \Sigma^c \to [m]$ be a mixed tabulation hash function and let $q \in \Sigma^c$ be a designated query element. Let $v \colon \Sigma^c \times [m] \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j \in [m]} v(x, j, k) = 0$ for all keys $x \in U$ and all $k \in [m]$. Define the random variable $V_{v,q}^{\text{simple}} = \sum_{x \in \Sigma^c \setminus \{q\}} v(x, h(x), h(q))$ and the random variables*

$$\begin{aligned}
M_{v,q} &= \max_{x \in \Sigma^c \setminus \{q\}, j \in [m]} |v(x, j, h(q))| , \\
\sigma_{v,q}^2 &= \frac{1}{m} \sum_{x \in \Sigma^c \setminus \{q\}} \sum_{j \in [m]} v(x, j, h(q))^2 ,
\end{aligned}$$

which only depend on the randomness of $h(q)$*. For all* $p \geq 2$ *then*

$$\mathrm{E}\Big[\Big(V_{v,q}^{\text{simple}}\Big)^p \,\Big|\, h(q)\Big]^{1/p} \leq \Psi_p\left(K_c\gamma_p^c M_{v,q}, K_c\gamma_p^c\sigma_{v,q}^2\right) \tag{A.57}$$

where $K_c = L_1\left(L_2 c\right)^c$*,* L_1 *and* L_2 *are universal constants, and*

$$\gamma_p = \max\left\{1, \frac{\log(m)}{\log(|\Sigma|)}, \frac{p}{\log(|\Sigma|)}\right\}.$$

Proof. We start by defining $\mathcal{H} = \sigma((h_1((i,\alpha_i)), h_2((i,\alpha_i)))_{i\in[c]})$. Now we get that,

$$\begin{aligned}
&\left\|V_{v,q}^{\text{mixed}} \,\middle|\, h(q)\right\|_p \\
&= \left\|\sum_{x\in\Sigma^c\setminus\{q\}} v(x,h(x),h(q)) \,\middle|\, h(q)\right\|_p \\
&= \left\|\left\|\sum_{x\in\Sigma^c\setminus\{q\}} v(x,h(x),h(q)) \,\middle|\, \mathcal{H}\right\|_p \,\middle|\, h(q)\right\|_p \\
&\leq \left\|\left\|\sum_{\alpha\in\Sigma\setminus\{h_2(q)\}} \varepsilon_2(\alpha) \sum_{x\in\Sigma^c} \varepsilon_1(x) v(x, h_1(x)\oplus h_3(\alpha), h(q))\left[h_2(x)=\alpha\right] \,\middle|\, \mathcal{H}\right\|_p \,\middle|\, h(q)\right\|_p \\
&+ \left\|\left\|\sum_{x\in\Sigma^c\setminus\{q\}} \varepsilon_2(h_2(q))\varepsilon_1(x) v(x, h_1(x)\oplus h_3(h_2(q)), h(q))\left[h_2(x)=h_2(q)\right] \,\middle|\, \mathcal{H}\right\|_p \,\middle|\, h(q)\right\|_p
\end{aligned}$$

For the second term we use Theorem A.12 to get that,

$$\begin{aligned}
&\left\|\left\|\sum_{x\in\Sigma^c\setminus\{q\}} \varepsilon_2(h_2(q))\varepsilon_1(x) v(x, h_1(x)\oplus h_3(h_2(q)), h(q))\left[h_2(x)=h_2(q)\right] \,\middle|\, \mathcal{H}\right\|_p \,\middle|\, h(q)\right\|_p \\
&\qquad \leq \Psi_p\left(2^{2c}K_c\gamma_p^{c-1}M_{v,q}, 2^{2c}K_c\gamma_p^{c-1}\sigma_{v,q}^2\right) \\
&\qquad \leq \Psi_p\left(2^{2c}K_c\gamma_p^{c}M_{v,q}, 2^{2c}K_c\gamma_p^{c}\sigma_{v,q}^2\right)
\end{aligned}$$

For the first term we use the same proof technique as in the proof of Theorem A.12. Write $q = (\alpha_0, \ldots, \alpha_{c-1})$ and for $I \subsetneq [c]$, we define $G_I \subseteq \Sigma^c$ by

$$G_I = \{x \in \Sigma^c \setminus \{q\} \,|\, \forall i \in I : (i,\alpha_i) \in x \wedge \forall i \notin I : (i,\alpha_i) \notin x\}$$

This clearly gives a partition of $\Sigma^c \setminus \{q\}$. Using this we obtain,

$$\begin{aligned}
&\left\| \sum_{\alpha \in \Sigma \setminus \{h_2(q)\}} \varepsilon_2(\alpha) \sum_{x \in \Sigma^c} \varepsilon_1(x) v(x, h_1(x) \oplus h_3(\alpha), h(q)) \left[h_2(x) = \alpha\right] \;\middle|\; \mathcal{H} \right\|_p \\
&\leq \sum_{I \subsetneq [c]} \left\| \sum_{\alpha \in \Sigma \setminus \{h_2(q)\}} \varepsilon_2(\alpha) \sum_{x \in G_I} \varepsilon_1(x) v(x, h_1(x) \oplus h_3(\alpha), h(q)) \left[h_2(x) = \alpha\right] \;\middle|\; \mathcal{H} \right\|_p \\
&= \sum_{I \subsetneq [c]} \left\| \sum_{\alpha \in \Sigma \setminus \{h_2(q)\}} \varepsilon_2(\alpha) \sum_{x \in G_I} \varepsilon_1(x) v(x, h_1(x) \oplus h_3(\alpha), h'(q)) \left[h_2(x) = \alpha\right] \;\middle|\; \mathcal{H} \right\|_p \\
&\leq \sum_{I \subsetneq [c]} \left\| \sum_{\alpha \in \Sigma} \varepsilon_2(\alpha) \sum_{x \in G_I} \varepsilon_1(x) v(x, h_1(x) \oplus h_3(\alpha), h'(q)) \left[h_2(x) = \alpha\right] \;\middle|\; \mathcal{H} \right\|_p .
\end{aligned}$$

We then use Theorem A.10 to get that,

$$\begin{aligned}
&\sum_{I \subsetneq [c]} \left\| \sum_{\alpha \in \Sigma} \varepsilon_2(\alpha) \sum_{x \in G_I} \varepsilon_1(x) v(x, h_1(x) \oplus h_3(\alpha), h'(q)) \left[h_2(x) = \alpha\right] \;\middle|\; \mathcal{H} \right\|_p \\
&\qquad \leq 2^c \Psi_p \left(K_c \gamma_p^c M_{v,q}, K_c \gamma_p^c \sigma_{v,q}^2\right) \\
&\qquad \leq \Psi_p \left(2^{2c} K_c \gamma_p^c M_{v,q}, 2^{2c} K_c \gamma_p^c \sigma_{v,q}^2\right) .
\end{aligned}$$

Now combining everything we get that,

$$\begin{aligned}
\left\| V_{v,q}^{\text{mixed}} \;\middle|\; h(q) \right\|_p &\leq 2\Psi_p \left(2^{2c} K_c \gamma_p^c M_{v,q}, 2^{2c} K_c \gamma_p^c \sigma_{v,q}^2\right) \\
&\leq \Psi_p \left(2^{2c+2} K_c \gamma_p^c M_{v,q}, 2^{2c+2} K_c \gamma_p^c \sigma_{v,q}^2\right),
\end{aligned}$$

which finishes the proof. □

A.7 Acknowledgement

Research supported by Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

References

[ADKK+20] Anders Aamand, Debarati Das, Evangelos Kipouridis, Jakob Bæk Tejs Knudsen, Peter M. R. Rasmussen, and Mikkel Thorup. "No Repetition: Fast Streaming with Highly Concentrated Hashing". In: *CoRR* (2020). arxiv.org/abs/2004.01156. arXiv: 2004.01156.

[AKKR+20] Anders Aamand, Jakob Bæk Tejs Knudsen, Mathias Bæk Tejs Knudsen, Peter Michael Reichstein Rasmussen, and Mikkel Thorup. "Fast Hashing with Strong Concentration Bounds". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. Chicago, IL, USA: Association for Computing Machinery, 2020, pp. 1265–1278. ISBN: 9781450369794.

[AKT18] Anders Aamand, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. "Power of d Choices with Simple Tabulation". In: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. Ed. by Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella. Vol. 107. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 5:1–5:14.

[AT19] Anders Aamand and Mikkel Thorup. "Non-empty Bins with Simple Tabulation Hashing". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*. Ed. by Timothy M. Chan. SIAM, 2019, pp. 2498–2512.

[AL12] Radosław Adamczak and Rafał Latała. "Tail and moment estimates for chaoses generated by symmetric random variables with logarithmically concave tails". In: *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques* 48.4 (2012), pp. 1103–1136.

[Ben62] George Bennett. "Probability Inequalities for the Sum of Independent Random Variables". In: *Journal of the American Statistical Association* 57.297 (1962), pp. 33–45. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1962.10482149.

[BRV11] Paolo Boldi, Marco Rosa, and Sebastiano Vigna. "HyperANF: Approximating the Neighbourhood Function of Very Large Graphs on a Budget". In: *Proc. 20th WWW*. ACM, 2011, pp. 625–634.

[BCLM+10] Vladimir Braverman, Kai-Min Chung, Zhenming Liu, Michael Mitzenmacher, and Rafail Ostrovsky. "AMS Without 4-Wise Independence on Product Domains". In: *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France*. Ed. by Jean-Yves Marion and Thomas Schwentick. Vol. 5. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010, pp. 119–130.

[Bro97] Andrei Z. Broder. "On the resemblance and containment of documents". In: *Proc. Compression and Complexity of Sequences (SEQUENCES)*. 1997, pp. 21–29.

[BCFM00] Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. "Min-Wise Independent Permutations". In: *jcss* 60.3 (2000). See also STOC'98, pp. 630–659.

[Che52] Herman Chernoff. "A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the sum of Observations". In: *Annals of Mathematical Statistics* 23.4 (1952), pp. 493–507.

[Coh14] Edith Cohen. "All-distances Sketches, Revisited: HIP Estimators for Massive Graphs Analysis". In: *Proc. 33rd PODSs*. ACM, 2014, pp. 88–99.

[DKRT15] S. Dahlgaard, M. B. T. Knudsen, E. Rotenberg, and M. Thorup. "Hashing for Statistics over K-Partitions". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. 2015, pp. 1292–1310.

[DKT17] Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. "Practical Hash Functions for Similarity Estimation and Dimensionality Reduction". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6618–6628. ISBN: 978-1-5108-6096-4.

[DR09] Martin Dietzfelbinger and Michael Rink. "Applications of a Splitting Trick". In: *Proceedings of the 36th ICALP*. 2009, pp. 354–365.

[Dum56] A. I. Dumey. "Indexing for rapid random access memory systems". In: *Computers and Automation* 5.12 (1956), pp. 6–9.

[FFGa07] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and et al. "Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm". In: *In Analysis of Algorithms (AOFA)*. 2007.

[FM85] Philippe Flajolet and G. Nigel Martin. "Probabilistic Counting Algorithms for Data Base Applications". In: *jcss* 31.2 (1985). Announced at FOCS'83, pp. 182–209.

[HNH13] Stefan Heule, Marc Nunkesser, and Alex Hall. "HyperLogLog in Practice: Algorithmic Engineering of a State of The Art Cardinality Estimation Algorithm". In: *Proceedings of the EDBT 2013 Conference*. 2013, pp. 683–692.

[Hit94] Pawel Hitczenko. "On a Domination of Sums of Random Variables by Sums of Conditionally Independent Ones". In: *The Annals of Probability* 22.1 (1994), pp. 453–468. ISSN: 00911798.

[KL15] Konrad Kolesko and Rafał Latała. "Moment estimates for chaoses generated by symmetric random variables with logarithmically convex tails". In: *Statistics & Probability Letters* 107 (2015), pp. 210–214. ISSN: 0167-7152.

[Lat97] Rafał Latała. "Estimation of moments of sums of independent real random variables". In: *The Annals of Probability* 25.3 (1997), pp. 1502–1513.

[Lat06] Rafał Latała. "Estimates of moments and tails of Gaussian chaoses". In: *The Annals of Probability* 34.6 (2006), pp. 2315–2331.

[Leh11] Joseph Lehec. "Moments of the Gaussian Chaos". In: *Séminaire de Probabilités XLIII*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 327–340. ISBN: 978-3-642-15217-7.

[LOZ12] Ping Li, Art B. Owen, and Cun-Hui Zhang. "One Permutation Hashing". In: *Proc. 26th NIPS*. 2012, pp. 3122–3130.

[Mar65] A. J. Maria. "A Remark on Stirling's Formula". In: *The American Mathematical Monthly* 72.10 (1965), pp. 1096–1098. ISSN: 00029890, 19300972.

[Ole03] Krzysztof Oleszkiewicz. "On a nonsymmetric version of the Khinchine-Kahane inequality". In: *Stochastic inequalities and applications*. Springer, 2003, pp. 157–168.

[PT13] Mihai Patrascu and Mikkel Thorup. "Twisted Tabulation Hashing". In: *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*. Ed. by Sanjeev Khanna. SIAM, 2013, pp. 209–228.

[PT10] Mihai Pătraşcu and Mikkel Thorup. "On the k-Independence Required by Linear Probing and Minwise Independence". In: *Proc. 37th International Colloquium on Automata, Languages and Programming (ICALP)*. 2010, pp. 715–726.

[PT12] Mihai Pătraşcu and Mikkel Thorup. "The Power of Simple Tabulation Hashing". In: *J. ACM* 59.3 (June 2012). ISSN: 0004-5411.

[PMS94] V. H. de la Pena, S. J. Montgomery-Smith, and Jerzy Szulga. "Contraction and Decoupling Inequalities for Multilinear Forms and U-Statistics". In: *The Annals of Probability* 22.4 (1994), pp. 1745–1765.

[Sch05] René L. Schilling. *Measures, Integrals and Martingales*. Cambridge University Press, 2005.

[SL14a] Anshumali Shrivastava and Ping Li. "Densifying One Permutation Hashing via Rotation for Fast Near Neighbor Search". In: *Proc. 31th International Conference on Machine Learning (ICML)*. 2014, pp. 557–565.

[SL14b] Anshumali Shrivastava and Ping Li. "Improved Densification of One Permutation Hashing". In: *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI 2014, Quebec City, Quebec, Canada, July 23-27, 2014*. 2014, pp. 732–741.

[Tho13] Mikkel Thorup. "Simple Tabulation, Fast Expanders, Double Tabulation, and High Independence". In: *54th Annual Symposium on Foundations of Computer Science (FOCS)*. 2013, pp. 90–99.

[WC81] Mark N. Wegman and Larry Carter. "New Classes and Applications of Hash Functions". In: *jcss* 22.3 (1981). Announced at FOCS'79, pp. 265–279.

[Zob70] Albert Lindsey Zobrist. *A New Hashing Method with Application for Game Playing*. Tech. rep. 88. Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 1970.

A.I Statistics over k-partitions using mixed tabulation

We will now describe how mixed tabulation hashing is used for statistics over k-partitions. The description of the application is largely copied from [DKRT15] but now we can use our new strong concentration bounds for mixed tabulation hashing to obtain stronger results.

We consider the generic approach where a hash function is used to k-partition a set into k bins. Statistics are computed on each bin, and then all these statistics are combined so as to get good concentration bounds. This approach was introduced by Flajolet and Martin [FM85] under the name *stochastic averaging* to estimate the number of distinct elements in a data stream. Today, a more popular estimator of this quantity is the HyperLogLog counter, which is also based on k-partitioning [FFGa07; HNH13]. These types of counters have found many applications, e.g., to estimate the neighborhood function of a graph with all-distance sketches [BRV11; Coh14]. Later k-partitions were used for set similarity in large-scale machine learning by Li et al. [LOZ12; SL14a; SL14b]. Under the name one-permutation hashing, Li et al. used k-partitions to gain a factor k in speed within the classic MinHash framework of Broder et al. [Bro97; BCFM00].

We will use MinHash for frequency estimation as an example to illustrate how mixed tabulation yields good statistics over k-partitions: suppose we have a fully random hash function applied to a set X of red and blue balls. We want to estimate the fraction f of red balls. The idea of the MinHash algorithm is to sample the ball with the smallest hash value. With a fully-random hash function, this is a uniformly random sample from X, and it is red with probability f. For better concentration, we may use k *independent repetitions*: we repeat the experiment k times with k independent hash functions. This yields a multiset S of k samples with replacement from X. The fraction of red balls in S concentrates around f and the error probability falls exponentially in k.

Consider now the alternative experiment based on *k-partitioning*, assuming that k is a power of two. We use a single hash function, where the first $\log k$ bits of the hash value partitions X into k bins, and then the remaining bits are used as a *local hash value* within the bin. We pick the ball with the smallest (local) hash value in each bin. This is a sample S from X without replacement, and again, the fraction of red balls in the non-empty bins is concentrated around f with exponential concentration bounds. We note that there are some differences. We do get the advantage that the samples are without replacement, which means better concentration. On the other hand, we may end up with fewer samples if some bins are empty.

The big difference between the two schemes is that the second one runs $\Omega(k)$ times faster. In the first experiment, each ball participated in k independent experiments, but in the second one with k-partitioning, each ball picks its bin, and then only participates in the local experiment for that bin. Thus, time-wise, we get k experiments for the price of one. Handling each ball, or key, in constant time is important in applications of high volume streams.

The above approach, however, requires a very powerful hash function. The main issue is the overall k-partitioning distribution between bins. It could be that if we get a lot of red balls in one bin, then this would be part of a general clustering of the red balls on a few bins (examples showing how such systematic clustering can happen with simple hash

functions are given in [PT10]). This clustering would disfavor the red balls in the overall average even if the sampling in each bin was uniform and independent. This is an issue of non-linearity, e.g., if there are already more red than blue balls in a bin, then doubling their number only increases their frequency by at most 3/2. We note that k-independence does not by itself suffices to give any guarantees in this situation.

Using mixed tabulation hashing We will now show mixed tabulation can be applied in the above application using selective full randomness from Theorem A.19 in tandem with the concentration from (A.14). Combined they give:

> Let $h : \Sigma^c \to \{0,1\}^\ell$ be a mixed tabulation hash function using d derived characters. Let M be an ℓ-bit mask with don't cares. For a given key set $X \subseteq \Sigma^c$, let Y be the set of keys from X with hash values matching M. If $\Omega(|\Sigma|) = \mathrm{E}[Y] = |\Sigma|(1-\Omega(1))$, then with probability $1-O(|\Sigma|^{1-\lfloor d/2 \rfloor})$, the free bits of the hash values in Y are fully random. Moreover, with this probability, $|Y| = \mathrm{E}[Y](1 \pm O(\sqrt{(\log|\Sigma|)/|\Sigma|}))$.

For the k-partitioning, we need k to be bounded relative to the alphabet size as $k \leq \frac{|\Sigma|}{4d \ln|\Sigma|}$. Recall here our hash tables need space $O(|\Sigma|)$ which then has to be a logarithmic factor bigger than k. This may motivate 16-bit characters rather than 8-bit characters, but on modern computers this still fits in fast cache.

We a set X of n red and blue balls, and for simplicity in our analysis, we assume that the ratio between them is at most a factor 2. If $n \leq |\Sigma|/2$, Theorem A.19 with all bits free states that the balls hash fully-randomly with high probability. This means that *any analysis* based on fully-random hashing applies directly. Otherwise let $q = \lceil \log(n/(2|\Sigma|/3) \rceil$. We study the set of balls Y that has q leading zeros in their local hash value. These q bits are the fixed bits in the mask Theorem A.19. The expected size of Y is between $|\Sigma|/3$ and $2|\Sigma|/3$, so by Theorem A.19, w.h.p., all other bits in the hash values of Y are fully random, including both the global index of $\log k$ bits and the tail of bits in the local hash value after the q leading zeros. Moreover, (A.14), imply that the size of Y is at most a factor

$$1 \pm O(\sqrt{(\log|\Sigma|)/|\Sigma|}) = 1 \pm O(\sqrt{1/k})$$

from its mean, and the same holds for the ratio between red and blue balls in our simple case where there are more than $n/4$ of each.

We claim that, w.h.p, all bins get a ball from Y. We know that $\mathrm{E}[|Y|] \geq |\Sigma|/3 \geq 2kd \ln|\Sigma|/3$, so w.h.p., $|Y| \geq kd \ln|\Sigma|/2$. Consider now any bin i. Since the bin indices are fully random, the probability that i gets no ball from Y is at most $(1-1/k)^{kd(\ln|\Sigma|/2)} < 1/|\Sigma|^{d/2}$. A union bound now implies that all bins get a ball from Y with probability $1 - k/|\Sigma|^{d/2} \geq 1 - |\Sigma|^{1-d/2}$.

Since every bin contains a ball from Y and these are the balls with q leading zeros in their local hash, the MinHash ball from X in each bin is from Y. Since all keys from Y have at least q leading zeros in their hash values and all other bits are fully random, this means that, w.h.p., the result of the experiment is exactly the same as if we applied it to

Y using a fully random hash function, and our concentration bounds imply that the ratio between red and blue balls is well-preserved from X to Y.

The important point in the above analysis is that it is only the analysis that needs to know anything about n. This is important in more complicated contexts, e.g., streaming where n is not known in advance, or in set similarity where red balls represent the intersection of sets while the blue balls represent their symmetric difference, and where we use the k-partitions to compare sets of many different sizes. We know that mixed tabulation hashing is almost as good fully random hashing as long as $k \leq \frac{|\Sigma|}{4d \ln |\Sigma|}$.

The above description is very similar to the one of Dahlgaard et al. [DKRT15]. The difference is that we now have the right concentration bounds for twisted tabulation, e.g., Dahlgaard et al., had the additional constraint that $k \leq \frac{|\Sigma|}{\log |\Sigma| (\log\log |\Sigma|)^2}$, which is completely unnecessary.

Appendix B

A Sparse Johnson-Lindenstrauss Transform using Fast Hashing

A Sparse Johnson-Lindenstrauss Transform using Fast Hashing

Jakob Bæk Tejs Houen*
University of Copenhagen
jakn@di.ku.dk

Mikkel Thorup*
University of Copenhagen
mthorup@di.ku.dk

Dansk resumé

The *Sparse Johnson-Lindenstrauss Transform* of Kane and Nelson (SODA 2012) provides a linear dimensionality-reducing map $A \in \mathbb{R}^{m \times u}$ in ℓ_2 that preserves distances up to distortion of $1 + \varepsilon$ with probability $1 - \delta$, where $m = O(\varepsilon^{-2} \log 1/\delta)$ and each column of A has $O(\varepsilon m)$ non-zero entries. The previous analyses of the Sparse Johnson-Lindenstrauss Transform all assumed access to a $\Omega(\log 1/\delta)$-wise independent hash function. The main contribution of this paper is a more general analysis of the Sparse Johnson-Lindenstrauss Transform with less assumptions on the hash function. We also show that the *Mixed Tabulation hash function* of Dahlgaard, Knudsen, Rotenberg, and Thorup (FOCS 2015) satisfies the conditions of our analysis, thus giving us the first analysis of a Sparse Johnson-Lindenstrauss Transform that works with a practical hash function.

Contents

*Research supported by Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

B.1 Introduction

Dimensionality reduction is an often applied technique to obtain a speedup when working with high dimensional data. The basic idea is to map a set of points $X \subseteq \mathbb{R}^u$ to a lower dimension while approximately preserving the geometry. The Johnson-Lindenstrauss lemma [JL84] is a foundational result in that regard.

Lemma B.1 ([JL84]). *For any $0 < \varepsilon < 1$, integers n, u, and $X \subseteq \mathbb{R}^u$ with $|X| = n$, there exists a map $f\colon X \to \mathbb{R}^m$ with $m = O(\varepsilon^{-2} \log n)$ such that*

$$\forall w, w' \in X,\ \left|\left\|f(w) - f(w')\right\|_2 - \left\|w - w'\right\|_2\right| \leq \varepsilon \left\|w - w'\right\|_2.$$

It has been shown in [AK17; LN17] that the target dimension m is optimal for nearly the entire range of n, u, ε. More precisely, for any n, u, ε there exists a set of points $X \subseteq \mathbb{R}^u$ with $|X| = n$ such that for any map $f\colon X \to \mathbb{R}^m$ where the Euclidean norm is distorted by at most $(1 \pm \varepsilon)$ must have $m = \Omega(\min\left\{u, n, \varepsilon^{-2} \log(\varepsilon^2 n)\right\})$.

All known proofs of the Johnson-Lindenstrauss lemma constructs a linear map f. The original proof of Johnson and Lindenstrauss [JL84] chose $f(x) = \Pi x$ where $\Pi \in \mathbb{R}^{m \times u}$ is an appropriately scaled orthogonal projection into a random m-dimensional subspace. Another simple construction is to set $f(x) = \frac{1}{\sqrt{m}} Ax$ where $A \in \mathbb{R}^{m \times u}$ and each entry is an independent Rademacher variable.[1] In both cases, it can be shown that as long as $m = \Omega(\varepsilon^{-2} \log 1/\delta)$ then

$$\forall w \in \mathbb{R}^u,\ \Pr\left[\left|\|f(w)\|_2^2 - \|w\|_2^2\right| \geq \varepsilon \|w\|_2^2\right] \leq \delta. \tag{B.1}$$

The Johnson-Lindenstrauss lemma follows by setting $\delta < 1/\binom{n}{2}$ and taking $w = z - z'$ for all pairs $z, z' \in X$ together with a union bound. (B.1) is also known as the distributional Johnson-Lindenstrauss lemma and it has been shown that the target dimension m is tight, more precisely, m must be at least $\Omega(\min\left\{u, \varepsilon^{-2} \log 1/\delta\right\})$ [JW13; KMN11].

Sparse Johnson-Lindenstrauss Transform. One way to speed up the embedding time is replacing the dense A of the above construction by a sparse matrix. The first progress in that regard came by Achlioptas in [Ach03] who showed that A can be chosen with i.i.d. entries where $A_{ij} = 0$ with probability $2/3$ and otherwise A_{ij} is chosen uniformly in $\pm\sqrt{\frac{3}{m}}$. He showed that this construction can achieve the same m as the best analyses of the Johnson-Lindenstrauss lemma. Hence this achieves essentially a 3x speedup, but the asymptotic embedding time is still $O(m \|x\|_0)$ where $\|x\|_0$ is number of non-zeros of x.

Motivated by improving the asymptotic embedding time, Kane and Nelson in [KN14], following the work in [DKS10; KN10; BOR10], introduced the Sparse Johnson-Lindenstrauss Transform which maps down to essentially optimal dimension $m = O(\varepsilon^{-2} \log n)$ and only has $s = O(\varepsilon^{-1} \log n)$ non-zeros entries per column. This speeds

[1] A Rademacher variables, X, is a random variable that is chosen uniformly in ± 1, i.e., $\Pr[X = 1] = \Pr[X = -1] = \frac{1}{2}$.

up the embedding time to $O(\varepsilon^{-1} \log n \|x\|_0) = O(\varepsilon m \|x\|_0)$ thus improving the embedding time by a factor of ε^{-1}. It nearly matches a sparsity lower bound by Nelson and Nguyen [NN13] who showed that any sparse matrix needs at least $s = \Omega(\varepsilon^{-1} \log(n)/\log(1/\varepsilon))$ non-zeros per column.

Using Hashing. When the input dimension, u, is large it is not feasible to store the matrix A explicitly. Instead, we use a hash function to calculate the non-zero entries of A. Unfortunately, the previous analyses of the Sparse Johnson-Lindenstrauss Transform [KN14; CJN18] assume access to a $\Omega(\log 1/\delta)$-wise independent hash function which is inefficient. This motivates the natural question:

What are the sufficient properties we need of the hash function for a Sparse Johnson-Lindenstrauss Transform to work?

The goal of this work is to make progress on this question. In particular, we provide a new analysis of a Sparse Johnson-Lindenstrauss Transform with fewer assumptions on the hash function. This improved analysis allows us to conclude that there exists a Sparse Johnson-Lindenstrauss Transform that uses Mixed Tabulation hashing which is efficient.

Mixed Tabulation Hashing. Before introducing Mixed Tabulation hashing, we will first discuss *Simple Tabulation hashing* which was introduced by Zobrist [Zob70]. Simple Tabulation hashing takes an integer parameter $c > 1$, and we view a key $x \in [u] = \{0, \ldots, u-1\}$ as a vector of c characters, $x_0, \ldots, x_{c-1} \in \Sigma = [u^{1/c}]$. For each character, we initialize a fully random table $T_i \colon \Sigma \to [2^r]$ and the hash value of x is then calculated as

$$h(x) = T_0[x_0] \oplus \ldots \oplus T_{c-1}[x_{c-1}],$$

where $\oplus$ is the bitwise XOR-operation. We say that h is a Simple Tabulation hash function with c characters.

We can now define *Mixed Tabulation hashing* which is a variant of Simple Tabulation hashing that was introduced in [DKRT15]. As with Simple Tabulation hashing, Mixed Tabulation hashing takes $c > 1$ as a parameter, and it takes a further integer parameter $d \geq 1$. Again, we view a key $x \in [u]$ as vector of c characters, $x_0, \ldots, x_{c-1} \in \Sigma = [u^{1/c}]$. We then let $h_1 \colon \Sigma^c \to [2^r]$, $h_2 \colon \Sigma^c \to \Sigma^d$, and $h_3 \colon \Sigma^d \to [2^r]$ be independent Simple Tabulation hashing. Mixed Tabulation hashing is then defined as follows

$$h(x) = h_1(x) \oplus h_3(h_2(x)).$$

We say that h a mixed tabulation hash function with c characters and d derived characters. We call $h_2(x) \in \Sigma^d$ the *derived* characters. Mixed Tabulation hashing can be efficiently implemented by storing h_1 and h_2 as a single table with entries in $[2^r] \times \Sigma^d$, so the whole hash function can be computed with just $c + d$ lookups.

Our Contributions. Our main contribution is a new analysis of a Sparse Johnson-Lindenstrauss Transform that does not rely on the high independence of the hash function. Instead we show that it suffices that the hash function supports a decoupling-decomposition combined with strong concentration bounds.

We show that Mixed Tabulation hashing satisfies these conditions. This gives the first instance of a practical hash function that can support a Sparse Johnson-Lindenstrauss Transform.

B.1.1 Sparse Johnson-Lindenstrauss Transform

As mentioned earlier, the Sparse Johnson-Lindenstrauss Transform was introduced by Kane and Nelson [KN14] and they provided two different constructions with the same sparsity. Later a simpler analysis was given in [CJN18] which also generalized the result to a more general class of constructions. In this paper, we will only focus on one of the constructions which is described below.

Before we discuss the construction of the Sparse Johnson-Lindenstrauss Transform, we will first consider the related CountSketch which was introduced in [CCF04] and was analyzed for dimensionality reduction in [TZ12]. In CountSketch, we construct the matrix A as follows: We pick a pairwise independent hash function, $h\colon [u] \to [m]$, and a 4-wise independent sign function $\sigma\colon [u] \to \{-1, 1\}$. For each $x \in [u]$, we set $A_{h(x),x} = \sigma(x)$ and the rest of the x'th column to 0. Clearly, this construction has exactly 1 non-zero entry per column. It was shown in [TZ12] that if $m = \Omega(\varepsilon^{-2}\delta^{-1})$ then it satisfies the distributional Johnson-Lindenstrauss lemma, eq. (B.1). The result follows by bounding the second moment of $\|Ax\|_2^2 - \|x\|_2^2$ for any $x \in \mathbb{R}^d$ and then apply Chebyshev's inequality.

The bad dependence in the target dimension, m, on the failure probability, δ, is because we only use the second moment. So one might hope that you can improve the dependence by looking at higher moments instead. Unfortunately, it is not possible to improve the dependence for general $x \in \mathbb{R}^d$, and it is only possible to improve the dependence if $\|x\|_\infty^2 / \|x\|_2^2$ is small. Precisely, how small $\|x\|_\infty^2 / \|x\|_2^2$ has to be, has been shown in [FKL18]. So to improve the dependence on δ, we need to increase the number of non-zero entries per column.

We are now ready to describe the construction of the Sparse Johnson-Lindenstrauss Transform. The construction is to concatenate s CountSketch matrices and scale the resulting matrix by $\frac{1}{\sqrt{s}}$. This clearly gives a construction that has s non-zero entries per column and as it has been shown in [KN14; CJN18] if $s = \Omega(\varepsilon^{-1}\log(1/\delta))$ then we can obtain the optimal target dimension $m = O(\varepsilon^{-2}\log(1/\delta))$. More formally, we construct the matrix A as follows:

1. We pick a hash function, $h\colon [s] \times [u] \to [m/s]$ and a sign function $\sigma\colon [s] \times [u] \to \{-1, 1\}$.

2. For each $x \in [u]$, we set $A_{i\cdot m/s + h(i,x),x} = \frac{\sigma(i,x)}{\sqrt{s}}$ for every $i \in [s]$ and the rest of the x'th column to 0.

In the previous analyses [KN14; CJN18], it was shown that if h and σ are $\Omega(\log 1/\delta)$-wise independent then the construction works. Unfortunately, it is not practical to use a $\Omega(\log 1/\delta)$-wise independent hash function so the goal of this work is to obtain an analysis of a Sparse Johnson-Lindenstrauss Transform with fewer assumptions about the hash function. In particular, we relax the assumptions of the hash function, h, and the sign function, σ, to just satisfying a decoupling-decomposition and a strong concentration property. The formal theorem is stated in Appendix B.3.

We also show that Mixed Tabulation satisfies these properties and thus that the Sparse Johnson-Lindenstrauss Transform can be implemented using Mixed Tabulation. Let us describe more formally, what we mean by saying that Mixed Tabulation can implement the Sparse Johnson-Lindenstrauss Transform. We let $h_1 \colon \Sigma^c = [u] \to [m/s]$, $h_2 \colon \Sigma^c \to \Sigma^d$, and $h_3 \colon \Sigma^d \to [m/s]$ be the independent Simple Tabulation hash functions that implement the Mixed Tabulation hash function, $h_1(x) \oplus h_3(h_2(x))$. We then extend it to the domain $[s] \times [u]$ as follows:

1. Let $h_2' \colon [s] \times \Sigma^c \to \Sigma^d$ be defined by $h_2'(i, x) = h_2(x) \oplus \underbrace{(i, \ldots, i)}_{d \text{ times}}$, i.e., each derived character gets xor'ed by i.
2. We then define $h \colon [s] \times [u] \to [m/s]$ and $\sigma \colon [s] \times [u] \to \{-1, 1\}$ by $h(i, x) = h_1(x) \oplus h_3(h_2'(i, x))$ and $\sigma(i, x) = \sigma_1(x) \cdot \sigma_3(h_2'(i, x))$, where h_1 and h_3 are the Simple Tabulation hash functions described above, and $\sigma_1 \colon \Sigma^c \to \{-1, 1\}$ and $\sigma_3 \colon \Sigma^d \to \{-1, 1\}$ are independent Simple Tabulation functions.

B.1.2 Hashing Speed

When we use tabulation schemes, it is often as a fast alternative to $\Omega(\log n)$-independent hashing. Typically, we implement a q-independent hash function using a degree $q - 1$ polynomial in $O(q)$ time, and Siegel [Sie04] has proved that this is best possible unless we use large space. More precisely, for some key domain $[u]$, if we want to do $t < q$ memory accesses, then we need space at least $u^{1/t}$. Thus, if we want higher than constant independence but still constant evaluation times, then we do need space $u^{\Omega(1)}$. In our application, we have to compute many hash values simultaneously, so an alternative strategy would be to evaluate the polynomial using multi-point evaluation. This would reduce the time per hash value to $O(\log^2 q)$ but this is still super constant time.

With tabulation hashing, we use tables of size $O(|\Sigma|)$ where $|\Sigma| = u^{1/c}$ and $c = O(1)$. The table lookups are fast if the tables fit in cache, which is easily the case for 8-bit characters. In connection with each lookup, we do a small number of very fast AC^0 operations: a cast, a bit-wise xor, and a shift. This is incomparable to polynomial in the sense of fast cache versus multiplications, but the experiments from [AKKR+20, Table 1] found Simple Tabulation hashing to be faster than evaluation a 2-wise independent polynomial hashing.

Tabulation schemes are most easily compared by the number of lookups. Storing h_1 and h_2 in the same table, Mixed Tabulation hashing uses $c + d$ lookups. With $d = c$, the experiments from [AKKR+20] found Mixed Tabulation hashing to be slightly more than

twice as slow as Simple Tabulation hashing, and the experiments from [DKT17] found Mixed Tabulation hashing to be about as fast as 3-wise independent polynomial hashing. This motivates our claim that Mixed Tabulation hashing is practical.

In theory, we could also use a highly independent hash function that uses large space, but we don't know of any efficient construction. Siegel states about his construction, it is "far too slow for any practical application" [Sie04], and while Thorup [Tho13] has presented a simpler construction than Siegel's, it is still not efficient. The experiments in [AKKR+20] found it to be more than an order magnitude slower than Mixed Tabulation hashing.

B.2 Related Work

Even Sparser Johnson-Lindenstrauss Transforms. As touched upon earlier, there is a lower bound by Nelson and Nguyen [NN13] that rules out significant improvements, but never the less there has been research into sparser embedding. In the extreme, Feature Hashing of [WDLS+09] considers the case of $s = 1$. The lower bound excludes Feature Hashing from working for all vectors, but in [FKL18] they gave tight bounds for which vectors it works in terms of the measure $\|w\|_\infty^2 / \|w\|_2^2$. This was later generalized in [Jag19] to a complete understanding between the tradeoff between s and the measure $\|w\|_\infty^2 / \|w\|_2^2$. In this paper, we will only focus on the case $s = \Theta(\varepsilon^{-1} \log 1/\delta)$ and $m = \Theta(\varepsilon^{-2} \log 1/\delta)$

Fast Johnson-Lindenstrauss Transform. Another direction to speed-up the evaluation of Johnson-Lindenstrauss transforms is to exploit dense matrices with fast matrix-vector multiplication. This was first done by Ailon and Chazelle [AC09] who introduced the Fast Johnson-Lindenstrauss Transform. Their original construction was recently [FHL22] shown to give an embedding time $O(u \log u + m(\log 1/\delta + \varepsilon \log^2(1/\delta)/\log(1/\varepsilon)))$.

This has generated a lot follow-up work that has tried to improve the running to a clean $O(u \log u)$. Some of the work sacrifice the optimal target dimension, $m = O(\varepsilon^{-2} \log 1/\delta)$, in order to speed-up the construction, and are satisfied with sub-optimal $m = O(\varepsilon^{-2} \log n \log^4 u)$ [KW11], $m = O(\varepsilon^{-2} \log^3 n)$ [DGCN+09], $m = O(\varepsilon^{-1} \log^{3/2} n \log^{3/2} u + \varepsilon^{-2} \log n \log^4 u)$ [KW11], $m = O(\varepsilon^{-2} \log^2 n)$ [HV11; Vyb11; FL20], and $m = O(\varepsilon^{-2} \log n \log^2(\log n) \log^3 u)$ [JPSS+22]. Another line of progress is to assume that the target dimension, m, is substantially smaller then the starting dimension, u. Under the assumption that $m = o(u^{1/2})$ the work in [AL08; BK21] achieves embedding time $O(u \log m)$. The only construction that for some regimes improves on the original Fast Johnson-Lindenstrauss Transform is the recent analysis [JPSS+22] of the Kac Johnson-Lindenstrauss Transform, which uses the Kac random walk [Kac56]. They show that it can achieve an embedding time of $O(u \log u + \min\{u \log n, m \log n \log^2(\log n) \log^3 u\})$.

Previous Work on Tabulation Hashing. The work by Patrascu and Thorup [PT12] initiated the study of tabulation based hashing that goes further than what 3-wise independence of constructions would suggest. A long line of papers have shown tabulation based

hashing to work for min-wise hashing [PT13; DT14], hashing for k-statistics [DKRT15], and the number of non-empty-bins [AT19]. Furthermore, multiple papers have been concerned with showing strong concentration results for tabulation based hashing [PT12; PT13; AKKR+20; HT22]. Tabulation based hashing has also been studied experimentally where they have been shown to exhibit great performance [DKT17; AKKR+20].

B.2.1 Preliminaries

In this section, we will introduce the notation which will be used throughout the paper. First we introduce p-norms.

Definition B.2 (p-norm)**.** Let $p \geq 1$ and X be a random variable with $\mathrm{E}[|X|^p] < \infty$. We then define the p-norm of X by $\|X\|_p = \mathrm{E}[|X|^p]^{1/p}$.

Throughout the paper, we will repeatedly work with value functions $v \colon U \times [m] \to \mathbb{R}$. We will allow ourself to sometime view them as vectors, and in particular, we will write

$$\|v\|_2 = \sqrt{\sum_{x \in U} \sum_{j \in [m/s]} v(x,j)^2},$$
$$\|v\|_\infty = \max_{x \in U, j \in [m/s]} |v(x,j)| .$$

We will also use the Ψ_p-function introduced in [HT22].

Definition B.3. For $p \geq 2$ we define the function $\Psi_p \colon \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ as follows,

$$\Psi_p(M, \sigma^2) = \begin{cases} \left(\frac{\sigma^2}{pM^2}\right)^{1/p} M & \text{if } p < \log \frac{pM^2}{\sigma^2} \\ \frac{1}{2}\sqrt{p}\sigma & \text{if } p < e^2 \frac{\sigma^2}{M^2} \\ \frac{p}{e \log \frac{pM^2}{\sigma^2}} M & \text{if } \max\left\{\log \frac{pM^2}{\sigma^2}, e^2 \frac{\sigma^2}{M^2}\right\} \leq p \end{cases} .$$

It was shown in [HT22] that $\Psi_p(1, \lambda)$ is within a constant factor of the p-norm of a Poisson distributed random variable with parameter λ. They also showed that $\Psi_p(M, \sigma^2)$ can be used to upper bound expressions involving a fully random hash function $h \colon U \to [m]$. Let $v \colon U \times [m] \to \mathbb{R}$ be a value function then they showed that

$$\left\| \sum_{x \in U} v(x, h(x)) \right\|_{\leq} C\Psi_p(\|v\|_\infty, \|v\|_2^2/m) ,$$

where C is a universal constant.

B.3 Overview of the New Analysis

Our main technical contribution is a new analysis of the Sparse Johnson-Lindenstrauss Transform that relaxes the assumptions on the hash function, h. We show that if h

satisfies a decoupling decomposition property and a strong concentration property then we obtain the same bounds for the Sparse Johnson-Lindenstrauss Transform. Both of these properties are satisfied by h if h is $\Omega(\log 1/\delta)$-wise independent so our assumptions are weaker than those of the previous analyses.

In this section, we will give an informal overview of new approach. The technical details and the formal statement of the result will be in Appendix B.4.

In order to describe our approach, we look at the random variable

$$Z = \|Aw\|_2^2 - 1 = \frac{1}{s} \sum_{i \in [s]} \sum_{x \neq y \in [u]} \sigma(i,x)\sigma(i,y) \left[h(i,x) = h(i,y)\right] w_x w_y. \tag{B.2}$$

Here $w \in \mathbb{R}^u$ is a unit vector. With this notation the goal becomes to bound $\Pr[|Z| \geq \varepsilon]$.

The first step in our analysis is that we want to decouple eq. (B.2). Decoupling was also used in one of the proofs in [CJN18], but since we want to prove the result for more general hash functions, we cannot directly use the standard decoupling inequalities. We will instead assume that our hash function allows a *decoupling-decomposition*. This will formally be defined in Appendix B.4 and we will for now assume that our hash function allows for the standard decoupling inequality. If we apply Markov's inequality and a standard decoupling inequality for fully random hashing we obtain the expression.

$$\begin{aligned} \Pr[|Z| \geq \varepsilon] &\leq \varepsilon^{-p} \operatorname{E}[|Z|^p] \\ &\leq \left(\varepsilon^{-1}\frac{4}{s}\right)^p \operatorname{E}\left[\left|\sum_{i \in [s]} \sum_{x,y \in [u]} \sigma(i,x)\sigma'(i,y) \left[h(i,x) = h'(i,y)\right] w_x w_y\right|^p\right] \end{aligned} \tag{B.3}$$

where (h', σ') are independent copies of (h, σ) and $p \geq 2$. The power of decoupling stems from the fact that it breaks up some of the dependencies and allows for a simpler analysis.

The goal is now to analyse $\left\|\sum_{i \in [s]} \sum_{x,y \in [u]} \sigma(i,x)\sigma'(i,y) \left[h(i,x) = h'(i,y)\right] w_x w_y\right\|_p$. This is done by first fixing (h', σ') and bounding $\left\|\sum_{i \in [s], j \in [m/s]} \sum_{x \in [u]} \sigma(i,x) \left[h(i,x) = j\right] w_x a_{ij}\right\|_p$ using the randomness of (h, σ) where $a_{ij} = \sum_{y \in [u]} \sigma'(i,y) \left[h'(i,y) = j\right] w_y$. In order to do this, we will assume that the pair (h, σ) is *strongly concentrated*. Again the formal definition of this is postponed to Appendix B.4, but informally, we say that the pair is strongly concentrated if it has concentration results similar to those of fully random hashing.

We now take the view that $|a_{ij}|$ is the load of the bin $(i, j) \in [s] \times [m/s]$. The idea is then to split $[s] \times [m/s]$ into heavy and light bins and handle each separately. We choose a parameter k and let I be the heaviest k bins. Using the triangle inequality, we then get

that

$$\left\| \sum_{i\in[s],j\in[m/s]} \sum_{x\in[u]} \sigma(i,x)\,[h(i,x)=j]\,w_x a_{ij} \right\|_p \le \left\| \sum_{(i,j)\in I} \sum_{x\in[u]} \sigma(i,x)\,[h(i,x)=j]\,w_x a_{ij} \right\|_p + \left\| \sum_{(i,j)\in[s]\times[m/s]\setminus I} \sum_{x\in[u]} \sigma(i,x)\,[h(i,x)=j]\,w_x a_{ij} \right\|_p .$$

We show that the contribution from the light bins is as if the collisions are independent. This should be somewhat intuitive since if we only have few collisions in each bin then the collisions behave as if they were independent. In contrast, we show that the contribution from the heavy bins is dominated by the heaviest bin. This turns out to be exactly what we need to finish the analysis.

B.4 Technical Results

In this section, we will expand on the description from Appendix B.3 and formalize the ideas.

Decoupling. Ideally, we would like to use the standard decoupling inequality, eq. (B.3). Unfortunately, we cannot expect more general hash functions to support such a clean decoupling. We therefore introduce the notion of a decoupling-decomposition.

Definition B.4 (Decoupling-decomposition)**.** Let $p \ge 2$, $L \ge 1$, and $0 \le \gamma \le 1$. We say that a collection of possibly randomized sets, (U_α), is a (p, L, γ)-*decoupling-decomposition* for a property P of a pair (h, σ), if there exist hash functions $h_\alpha \colon [s] \times U_\alpha \to [m/s]$ and sign functions $s \colon [s] \times U_\alpha \to \{-1, 1\}$ for all α such that

$$\Pr[|Z| \ge \varepsilon] \le \left(\varepsilon^{-1} \sum_\alpha \frac{L}{s} \left\| \sum_{i\in[s]} \sum_{x,y\in U_\alpha} \sigma_\alpha(i,x)\sigma'_\alpha(i,y)\left[h_\alpha(i,x) = h'_\alpha(i,y)\right] w_x w_y \right\|_p \right)^p + \gamma \tag{B.4}$$

where $(h_\alpha, \sigma_\alpha)$ and $(h'_\alpha, \sigma'_\alpha)$ has the same distribution, and $(h_\alpha, \sigma_\alpha)$ satisfies the property P when conditioned on $(h'_\alpha, \sigma'_\alpha)$ and U_α.

The reader should compare eq. (B.3) for fully random hashing with eq. (B.4). There are 3 main differences between the expressions.

1. The first thing to notice is that, in the decoupling-decomposition we sum over different sets (U_α), where this is not needed for fully random hashing. We allow the decoupling-decomposition to use a different decoupling on each of the sets U_α. This is very powerful since general hash functions are not necessarily uniform over the input domain.

2. For the decoupling-decomposition, we allow an additive error probability γ. This is useful if the hash function allows for decoupling most of the time except when some unprobable event is happening.

3. The last difference is that a much larger loss-factor is allowed by the decoupling-decomposition than eq. (B.3). In the case of fully random hashing, we only lose a factor of 4 but for more general hash functions this loss might be bigger.

Finally, we note that eq. (B.3) implies if (h, σ) is $2p$-wise independent for an integer $p \geq 2$ then $[u]$ is a decoupling-decomposition of (h, σ) for any property P that is satisfied by (h, σ).

Strong Concentration. The second property we need is that the hash function is strongly concentrated.

Definition B.5 (Strong concentration)**.** Let $h\colon [s] \times U \to [m/s]$ be a hash function and $\sigma\colon [s] \times U \to \{-1, 1\}$ be a sign function. We say that the pair (h, σ) is (p, L)-*strongly-concentrated* if

1. For all value functions, $v\colon [s] \times [m/s] \to \mathbb{R}$, and all vectors, $w \in \mathbb{R}^U$,

$$\left\| \sum_{i\in[s]} \sum_{x\in U} \sigma(i,x) v(i, h(i,x)) w_x \right\|_p \leq \Psi_p \left(L \|v\|_\infty \|w\|_\infty , L\frac{s}{m} \|v\|_2^2 \|w\|_2^2 \right) , \tag{B.5}$$

$$\left\| \sum_{i\in[s]} \sum_{x\in U} \sigma(i,x) v(i, h(i,x)) w_x \right\|_p \leq \sqrt{L\frac{p}{\log(m/s)} \|v\|_2^2 \|w\|_2^2} . \tag{B.6}$$

2. For all vectors, $w \in \mathbb{R}^U$,

$$\left\| \sum_{i\in[s]} \sum_{j\in[m/s]} \left(\sum_{x\in U} \sigma(i,x) [h(i,x) = j] w_x \right)^2 \right\|_{p/2} \leq L \max\left\{ s \|w\|_2^2 , \frac{p}{\log m/s} \|w\|_2^2 \right\} . \tag{B.7}$$

3. If $p \leq \log m$,

$$\left\| \max_{i\in[s], j\in[m/s]} \left| \sum_{x\in U} \sigma(i,x) [h(i,x) = j] w_x \right| \right\|_p \leq e\sqrt{L\frac{\log m}{\log m/s}} \|w\|_2 . \tag{B.8}$$

We need essentially 3 different properties of our hash function to say that it is strongly concentrated.

1. The first property is a concentration result on the random variable

$$\sum_{i\in[s]}\sum_{x\in U}\sigma(i,x)v(i,h(i,x))w_x.$$

Here we need two different concentration results: The first concentration result, eq. (B.5), roughly corresponds to a p-norm version of what you would obtain by applying Bennett's inequality to a fully random hash function, while the second concentration result, eq. (B.5), corresponds to the best hypercontractive result you can obtain for weighted sums of independent Bernoulli-Rademacher variables with parameter s/m.[2]

2. The second property bounds the sum of squares

$$W = \sum_{i\in[s]}\sum_{j\in[m/s]}\left(\sum_{x\in U}\sigma(i,x)\left[h(i,x)=j\right]w_x\right)^2.$$

The condition, eq. (B.7), bounds $\|W\|_{p/2}$ by the maximum of two cases. The first case corresponds to $\mathrm{E}[W]$, and the second case is motivated by applying eq. (B.6) to

$$\sup_{\substack{z\in\mathbb{R}^{[s]\times[m/s]},\\ \|z\|_2=1}}\left\|\sum_{i\in[s]}\sum_{j\in[m]}\sum_{x\in U}\sigma(i,x)z_{i,h(i,x)}w_x\right\|_p^2.$$

While this at first glance might seem odd, it is roughly the best you can do, since one can show that

$$\max\left\{\mathrm{E}[W],\ \sup_{\substack{z\in\mathbb{R}^{[s]\times[m/s]},\\ \|z\|_2=1}}\left\|\sum_{i\in[s]}\sum_{j\in[m]}\sum_{x\in U}\sigma(i,x)z_{i,h(i,x)}w_x\right\|_p^2\right\}\leq\|W\|_{p/2}.$$

3. The final property is a bound on the largest coordinate, $\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\left[h(i,x)=j\right]w_x\right|$. The bound is a natural consequence of eq. (B.6) for fully random hashing. Namely, for fully random hashing we

[2] A Bernoulli-Rademacher variable with parameter α is random variable, $X\in\{-1,0,1\}$, with $\Pr[X=1]=\Pr[X=-1]=\alpha/2$ and $\Pr[X=0]=1-\alpha$.

get that

$$\begin{aligned}
&\left\| \max_{i\in[s],j\in[m/s]} \left| \sum_{x\in U} \sigma(i,x)\,[h(i,x)=j]\,w_x \right| \right\|_p \\
&\qquad \leq \left\| \max_{i\in[s],j\in[m/s]} \left| \sum_{x\in U} \sigma(i,x)\,[h(i,x)=j]\,w_x \right| \right\|_{\log m} \\
&\qquad \leq e \max_{i\in[s],j\in[m/s]} \left\| \left| \sum_{x\in U} \sigma(i,x)\,[h(i,x)=j]\,w_x \right| \right\|_{\log m} \\
&\qquad \leq e\sqrt{L\frac{\log m}{\log m/s}}\,\|w\|_2\,.
\end{aligned}$$

This derivation is not true for general hash function, but the hash function can still satisfy eq. (B.8).

The results of [HT22] show that if the hash function $h\colon [s]\times U \to [m/s]$ and the sign function $\sigma\colon [s]\times U \to [m/s]$ is p-wise independent for an integer $p \geq 2$ then the pair (h,σ) is (p,K)-strongly-concentrated where K is a universal constant.

The Main Result. We are now ready to state our main result which is a new analysis of a Sparse Johnson-Lindenstrauss Transform that only assumes that the hash function has a decoupling-decomposition for the strong concentration property.

Theorem B.6. *Let $h\colon [s]\times[u] \to [m/s]$ be a hash function and $\sigma\colon [s]\times[u] \to \{-1,1\}$ be a sign function. Furthermore, let $0<\varepsilon<1$ and $0<\delta<1$ be given, and define $p=\log 1/\delta$.*

Assume that there exists constants L_1, L_2, L_3, and $0\leq\gamma<1$, that only depends on (h,σ) and p, such that

1. *There exists a (p,L_1,γ)-decoupling-decomposition, (U_α), for the (p,L_2)-strong-concentration property of (h,σ)*
2. *For all vectors $w\in\mathbb{R}^u$, $\sum_\alpha \sum_{x\in U_\alpha} w_x^2 \leq L_3\,\|w\|_2^2$.*
3. *$m \geq \left(16e^7 L_1^2 L_2^3 L_3^2\right)\cdot\varepsilon^{-2}\log(1/\delta)$.*
4. *$s \geq \left(64e^3 L_1 L_2^{3/2} L_3\right)\cdot\varepsilon^{-1}\log(1/\delta)$.*

Then the following is true

$$\Pr[|Z|\geq\varepsilon] \leq \delta+\gamma.$$

As discussed earlier, a fully random hash function satisfies all the property needed of the theorem and thus gives a new analysis of the Sparse Johnson-Lindenstrauss Transform

for fully random hashing. We will also later show that Mixed Tabulation satisfies the assumption of the theorem hence giving the first analysis of a Sparse Johnson-Lindenstrauss Transform with a practical hash function that works.

The main difficulty in the analysis of Theorem B.6 is contained in the following technical lemma. The idea in the proof of Theorem B.6 is to use the decoupling-decomposition and apply the following lemma to each part.

Lemma B.7. *Let $h, \overline{h}\colon [s] \times U \to [m/s]$ be hash functions and $\sigma, \overline{\sigma}\colon [s] \times U \to \{-1, 1\}$ be sign functions. Let $p \geq 2$ and assume that there exists a constant L such that (h, σ) is (p, L)-strongly concentrated when conditioning on $(\overline{h}, \overline{\sigma})$, and similarly, $(\overline{h}, \overline{\sigma})$ is (p, L)-strongly concentrated when conditioning on (h, σ). Then for all vectors $w \in \mathbb{R}^U$,*

$$\left\| \sum_{i \in [s]} \sum_{x,y \in U} \sigma(i,x)\overline{\sigma}(i,y) \left[h(i,x) = \overline{h}(i,y)\right] w_x w_y \right\|_p \leq \Psi_p\left(32e^3L^{3/2}\left\|w\right\|_2^2, 32e^6L^3\frac{s^2}{m}\left\|w\right\|_2^4\right) + 36e^3L\frac{p}{\log m/s}\left\|w\right\|_2^2.$$

The lemma shows that the expression has two different regimes. The first regime, $\Psi_p\left(32e^3L^{3/2}\left\|w\right\|_2^2, 32e^6L^3\frac{s^2}{m}\left\|w\right\|_2^4\right)$, is essentially what we would expect if each of the collisions, $\left[h(i,x) = \overline{h}(i,y)\right]$, are independent of each other. The other regime, $36e^3L\frac{p}{\log m/s}\left\|w\right\|_2^2$, is essentially what you expect the largest coordinate to contribute.

Our analysis is inspired by these two regimes and tries to exploit them explicitly. We start by fixing (h, σ) and divide the coordinates into heavy and light coordinates. We then show that contribution of the light coordinates is $\Psi_p\left(32e^3L^{3/2}\left\|w\right\|_2^2, 32e^6L^3\frac{s^2}{m}\left\|w\right\|_2^4\right)$ which matches the intuition that if we only have few collisions on each coordinate then the collisions behave as if they were independent. Similarly, we show that the contribution of the heavy coordinates is dominated by the heaviest coordinate, namely, the contribution is $36e^3L\frac{p}{\log m/s}\left\|w\right\|_2^2$.

B.4.1 Mixed Tabulation Hashing

Our main result for Mixed Tabulation hashing is the following.

Theorem B.8. *Let $h\colon [s] \times [u] \to [m/s]$ and $\sigma\colon [s] \times [u] \to \{-1, 1\}$ be Mixed Tabulation functions as described in Appendix B.1.1. Furthermore, let $0 < \varepsilon < 1$ and $0 < \delta < 1$ be given, and define $p = \log 1/\delta$.*

If $m \geq \gamma_p^{3c}\varepsilon^{-2}\log(1/\delta)$ and $s \geq \gamma_p^{3/2c}\varepsilon^{-1}\log(1/\delta)$ where $\gamma_p = Kc\max\left\{1, \frac{p}{\log|\Sigma|}\right\}$ for a universal constant K.

Then the following is true

$$\Pr[|Z| \geq \varepsilon] \leq \delta + \varepsilon 3^c\left|\Sigma\right|^{-d}.$$

The result follows by proving that Mixed Tabulation hashing has a $(p, 4^{c+2}, 4\varepsilon^{-2}3^c\frac{s}{m}|\Sigma|^{-d})$-decoupling-decomposition and that Mixed Tabulation has the strong concentration property. The main new part is in showing the decoupling-decomposition while the analysis of the strong concentration property is modification of the analysis in [HT22].

B.5 Analysis of the Sparse Johnson-Lindenstrauss Transform

Lets us start by showing how Lemma B.7 implies our main result, Theorem B.6.

Proof of Theorem B.6. We start by using eq. (B.4) of the decoupling decomposition to get that

$$\Pr[|Z| \geq \varepsilon] \leq \left(\varepsilon^{-1}\sum_{\alpha}\frac{L_1}{s}\left\|\sum_{i\in[s]}\sum_{x,y\in U_\alpha}\sigma_\alpha(i,x)\sigma'_\alpha(i,y)\left[h_\alpha(i,x)=h'_\alpha(i,y)\right]v_xv_y\right\|_p\right)^p+\gamma$$

Now we fix α and apply Lemma B.7 while fixing U_α

$$\left\|\sum_{i\in[s]}\sum_{x,y\in U_\alpha}\sigma_\alpha(i,x)\sigma'_\alpha(i,y)\left[h_\alpha(i,x)=h'_\alpha(i,y)\right]v_xv_y\right\|_p \leq \Psi_p\left(32e^3L_2^{3/2}, 32e^6L^3\frac{s^2}{m}\right)\sum_{x\in U_\alpha}w_x^2+36e^3L_2\frac{p}{\log m/s}\sum_{x\in U_\alpha}w_x^2$$

Using this we get that

$$\sum_{\alpha}\frac{L_1}{s}\left\|\sum_{i\in[s]}\sum_{x,y\in U_\alpha}\sigma_\alpha(i,x)\sigma'_\alpha(i,y)\left[h_\alpha(i,x)=h'_\alpha(i,y)\right]v_xv_y\right\|_p \leq \sum_{\alpha}\frac{L_1}{s}\left(\Psi_p\left(32e^3L_2^{3/2}, 32e^6L^3\frac{s^2}{m}\right)\sum_{x\in U_\alpha}w_x^2+36e^3L_2\frac{p}{\log m/s}\sum_{x\in U_\alpha}w_x^2\right)$$

We now use that $\sum_\alpha\sum_{x\in U_\alpha}w_x^2 \leq L_3\|w\|_2^2$ to get that

$$\sum_{\alpha}\frac{L_1}{s}\left(\Psi_p\left(32e^3L_2^{3/2}, 32e^6L^3\frac{s^2}{m}\right)\sum_{x\in U_\alpha}w_x^2+36e^3L_2\frac{p}{\log m/s}\sum_{x\in U_\alpha}w_x^2\right) \leq \frac{L_3L_1}{s}\left(\Psi_p\left(32e^3L_2^{3/2}, 32e^6L^3\frac{s^2}{m}\right)+36e^3L_2\frac{p}{\log m/s}\right)\|w\|_2^2$$

It can now be checked that if m and s satisfies the stated assumptions then

$$\frac{L_3 L_1}{s}\left(\Psi_p\left(32e^3L_2^{3/2}, 32e^6L^3\frac{s^2}{m}\right) + 36e^3L_2\frac{p}{\log m/s}\right)\|w\|_2^2 \leq e^{-1}\varepsilon$$

Combining all the facts, we get that

$$\Pr[|Z| \geq \varepsilon] \leq \left(\varepsilon^{-1}(e^{-1}\varepsilon)\right)^p + \gamma = \delta + \gamma.$$

This finishes the proof. □

The rest of the section is concerned with proving our main technical lemma, Lemma B.7. First we need the following two lemmas from [HT22].

Lemma B.9. *Let $f\colon \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a non-negative function which is monotonically increasing in every argument, and assume that there exists positive reals $(\alpha_i)_{i\in[n]}$ and $(t_i)_{i\in[n]}$ such that for all $\lambda \geq 0$,*

$$f(\lambda^{\alpha_0}t_0, \ldots, \lambda^{\alpha_{n-1}}t_{n-1}) \leq \lambda f(t_0, \ldots, t_{n-1})\,.$$

Let $(X_i)_{i\in[n]}$ be non-negative random variables. Then for all $p \geq 1$ we have that

$$\|f(X_0, \ldots, X_{n-1})\|_p \leq n^{1/p}\max_{i\in[n]}\left(\frac{\|X_i\|_{p/\alpha_i}}{t_i}\right)^{1/\alpha_i} f(t_0, \ldots, t_{n-1})\,.$$

Lemma B.10. *Let $p \geq 2$, $M > 0$, and $\sigma^2 > 0$ then*

$$\frac{1}{2}\sqrt{p}\sigma \leq \Psi_p(M, \sigma^2) \leq \max\left\{\frac{1}{2}\sqrt{p}\sigma, \frac{1}{2e}pM\right\}.$$

We are now ready to prove Lemma B.7.

Proof of Lemma B.7. We start by defining $v_h, v_{\bar{h}}\colon [s] \times [m/s] \to \mathbb{R}$ by,

$$v_h(i,j) = \sum_{x\in U}\sigma(i,x)w_x\left[h(i,x) = j\right]\,,$$
$$v_{\bar{h}}(i,j) = \sum_{y\in U}\overline{\sigma}(i,y)w_y\left[\overline{h}(i,y) = j\right]\,.$$

We then want to prove that

$$\left\|\sum_{i\in[s],j\in[m/s]} v_h(i,j)v_{\bar{h}}(i,j)\right\|_p \leq \Psi_p\left(32e^3L^{3/2}\|w\|_2^2, 32e^6L^3\|w\|_2^4\right) + 4e^3L\frac{p}{\log m/s}\|w\|_2^2\,.$$

First we consider the case where $\frac{p}{\log m/s}\|w\|_2^2 \geq s\|w\|_2^2$. By Cauchy-Schwartz and eq. (B.7) we get that

$$\left\|\sum_{i\in[s],j\in[m/s]} v_h(i,j)v_{\bar{h}}(i,j)\right\|_p \leq \left\|\sum_{i\in[s],j\in[m/s]} v_h(i,j)^2\right\|_p \leq L\frac{p}{\log m/s}\|w\|_2^2 \ .$$

We now focus on the case where $\frac{p}{\log m/s}\|w\|_2^2 < s\|w\|_2^2$. We define $\pi\colon [m] \to [s]\times[m/s]$ to be a bijection which satisfies that

$$|v_h(\pi(0))| \geq |v_h(\pi(1))| \geq \ldots \geq |v_h(\pi(m-1))| \ .$$

We note that π is a random function but we can define π such that it only depends on the randomness of h and σ. We define $k = \lfloor p/\log(m/p)\rfloor$, $I = \{\pi(i)\,|\,i\in[k]\}$, and the random functions $v'_h, v''_h\colon [s]\times[m/s]\to\mathbb{R}$ by

$$\begin{aligned} v'_h(i,j) &= v_h(i,j)\,[(i,j)\in I]\,,\\ v''_h(i,j) &= v_h(i,j)\,[(i,j)\notin I]\,. \end{aligned}$$

Again we note that v'_h and v''_h only depends on the randomness of h and σ. We can then write our expression as

$$\begin{aligned} \left\|\sum_{i\in[s],j\in[m/s]} v_h(i,j)v_{\bar{h}}(i,j)\right\|_p &= \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h(i,\overline{h}(i,y))w_y\right\|_p\\ &\leq \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v'_h(i,\overline{h}(i,y))w_y\right\|_p + \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v''_h(i,\overline{h}(i,y))w_y\right\|_p \ . \end{aligned}$$

We will bound each of the term separately. We start by bounding $\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v'_h(i,\overline{h}(i,y))w_y\right\|_p$. We fix h and σ and use eq. (B.6) to get that

$$\begin{aligned} \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v'_h(i,\overline{h}(i,y))w_y\right\|_p &\leq \left\|\sqrt{L\frac{p}{\log m/s}\|w\|_2^2\left\|v'_h\right\|_2^2}\right\|_p\\ &= \sqrt{L\frac{p}{\log m/s}}\|w\|_2\left\|\sqrt{\sum_{(i,j)\in I} v'_h(i,j)^2}\right\|_p \ . \end{aligned}$$

We note that $\sum_{(i,j)\in I} v'_h(i,j)^2 = \max_{J\subseteq[s]\times[m/s],|J|=k} \sum_{(i,j)\in J} v_h(i,j)^2$. We then get that

$$\begin{aligned}
\left\| \sqrt{\sum_{(i,j)\in I} v'_h(i,j)^2} \right\|_p &= \left\| \sqrt{\max_{J\subseteq[s]\times[m/s],|J|=k} \sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p \\
&\leq \left(\sum_{J\subseteq[s]\times[m/s],|J|=k} \left\| \sqrt{\sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p^p \right)^{1/p} \\
&\leq \binom{ms}{k}^{1/p} \max_{J\subseteq[s]\times[m/s],|J|=k} \left\| \sqrt{\sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p
\end{aligned}$$

We use Sterling's bound and get that $\binom{ms}{k}^{1/p} \leq \left(\frac{ems}{k}\right)^{k/p} \leq \left(\frac{ems\log(ms/p)}{p}\right)^{1/\log(ms/p)} \leq e^3$. So we get that

$$\left\| \sqrt{\sum_{(i,j)\in I} v'_h(i,j)^2} \right\|_p \leq e^3 \max_{J\subseteq[s]\times[m/s],|J|=k} \left\| \sqrt{\sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p$$

A standard volumetric argument gives that there exists a $1/4$-net, $Z \subseteq \mathbb{R}^J$, with $|Z| \leq 9^k$, such that

$$\begin{aligned}
\left\| \sqrt{\sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p &= \left\| \sup_{z\in\mathbb{R}^J,\|z\|_2=1} \sum_{(i,j)\in J} z_{i,j} v_h(i,j) \right\|_p \\
&\leq \left\| \sup_{z\in Z} \sum_{(i,j)\in J} z_{i,j} v_h(i,j) \right\|_p \\
&\quad + \left\| \sup_{z\in\mathbb{R}^J,\|z\|_2=1} \sum_{(i,j)\in J} (z_{i,j} - z'_{i,j}) v_h(i,j) \right\|_p \\
&\leq \left\| \sup_{z\in Z} \sum_{(i,j)\in J} z_{i,j} v_h(i,j) \right\|_p \\
&\quad + \sup_{z\in\mathbb{R}^J,\|z\|_2=1} \|z - z'\|_2 \left\| \sqrt{\sum_{(i,j)\in J} v_h(i,j)^2} \right\|_p
\end{aligned}$$

where $z' \in Z$ is the closest element to z, and as such $\|z - z'\|_2 \leq 1/4$. Since there are at most 9^k elements in Z then $\left\|\sup_{z\in Z} \sum_{(i,j)\in J} z_{i,j} v_h(i,j)\right\|_p \leq$

$9 \sup_{z \in Z} \left\| \sum_{(i,j) \in J} z_{i,j} v_h(i,j) \right\|_p$, where we used that $k \leq p$. Collecting the fact we get that

$$\left\| \sqrt{\sum_{(i,j) \in J} v_h(i,j)^2} \right\|_p \leq 36 \sup_{z \in Z} \left\| \sum_{(i,j) \in J} z_{i,j} v_h(i,j) \right\|_p$$

Using this we get that

$$\begin{aligned} e^3 & \max_{J \subseteq [s] \times [m/s], |J| = k} \left\| \sqrt{\sum_{(i,j) \in J} v_h(i,j)^2} \right\|_p \\ & \leq 36 e^3 \max_{J \subseteq [s] \times [m/s], |J| = k} \max_{z \in Z} \left\| \sum_{(i,j) \in J} z_{i,j} v_h(i,j) \right\|_p \\ & = 36 e^3 \max_{J \subseteq [s] \times [m/s], |J| = k} \max_{\substack{z \in \mathbb{R}^{s \times m/s}, \\ \|z\|_2 = 1}} \left\| \sum_{i \in [s]} \sum_{x \in U} \sigma(i,x) z_{i,h(i,x)} \left[(i, h(i,x)) \in J \right] w_x \right\|_p \end{aligned}$$

We can then use eq. (B.6) to get that

$$\begin{aligned} 36 e^3 & \max_{J \subseteq [s] \times [m/s], |J| = k} \max_{\substack{z \in \mathbb{R}^{s \times m/s}, \\ \|z\|_2 = 1}} \left\| \sum_{i \in [s]} \sum_{x \in U} \sigma(i,x) z_{i,h(i,x)} \left[(i, h(i,x)) \in J \right] w_x \right\|_p \\ & \leq 36 e^3 \max_{J \subseteq [s] \times [m/s], |J| = k} \max_{\substack{z \in \mathbb{R}^{s \times m/s}, \\ \|z\|_2 = 1}} \sqrt{L \frac{p}{\log m/s}} \|w\|_2 \|z\|_2 \\ & = 36 e^3 \sqrt{L \frac{p}{\log m/s}} \|w\|_2 \end{aligned}$$

Combining the facts, we get that $\left\| \sum_{i \in [s]} \sum_{y \in U} \overline{\sigma}(i,y) v'_h(i, \overline{h}(i,y)) w_y \right\|_p \leq 36 e^3 L \frac{p}{\log m/s} \|w\|_2^2$.

We will now bound $\left\| \sum_{i \in [s]} \sum_{y \in U} \overline{\sigma}(i,y) v''_h(i, \overline{h}(i,y)) w_y \right\|_p$. We fix h and ε and use eq. (B.5) to get that

$$\begin{aligned} \left\| \sum_{i \in [s]} \sum_{y \in U} \overline{\sigma}(i,y) v''_h(i, \overline{h}(i,y)) w_y \right\|_p & \leq \left\| \Psi_p \left(L \|w\|_\infty \left\| v'_h \right\|_\infty, L \frac{s}{m} \|w\|_2^2 \left\| v'_h \right\|_2^2 \right) \right\|_p \\ & \leq \left\| \Psi_p \left(L \|w\|_\infty \left| v'_h(\pi(k+1)) \right|, L \frac{s}{m} \|w\|_2^2 \|v_h\|_2^2 \right) \right\|_p . \end{aligned}$$

Now we use Lemma B.9 to get that,

$$\left\|\Psi_p\left(L\left\|w\right\|_\infty\left|v_h'(\pi(k+1))\right|, L\frac{s}{m}\left\|w\right\|_2^2\left\|v_h\right\|_2^2\right)\right\|_p \leq \sqrt{2}\Psi_p\left(L\left\|w\right\|_\infty\left\|\left|v_h'(\pi(k+1))\right|\right\|_p, L\frac{s}{m}\left\|w\right\|_2^2\left\|\left\|v_h\right\|_2^2\right\|_{p/2}\right).$$

Since we assume that $\frac{p}{\log m}\left\|w\right\|_2^2 < s\left\|w\right\|_2^2$ then eq. (B.7) give us that $\left\|\left\|v_h\right\|_2^2\right\|_{p/2} \leq Ls\left\|w\right\|_2^2$.

We will now bound $\left\|v_h'(\pi(k+1))\right\|_p$. For this, we will distinguish between two cases: Either $p \geq \log m$ or $p < \log m$. Let us first case where $p \geq \log m$. We will use that $|v_h'(\pi(k+1))| \leq \frac{\sum_{i\in[k+1]}|v_h'(\pi(i))|}{k+1}$. We then get that

$$\begin{aligned}
&\left\|v_h'(\pi(k+1))\right\|_p \\
&\leq \left\|\frac{\sum_{i\in[k+1]}|v_h'(\pi(i))|}{k+1}\right\|_p \\
&\leq \left(\binom{m}{k+1}2^{k+1}\max_{\substack{J\subseteq[s]\times[m/s],\\|J|=k+1}}\max_{(\sigma_{i,j})_{(i,j)\in J}\in\{-1,1\}^J}\left(\frac{\left\|\sum_{(i,j)\in J}\sigma_{i,j}v_h(i,j)\right\|_p}{k+1}\right)^p\right)^{1/p} \\
&\leq \max_{\substack{J\subseteq[s]\times[m/s],\\|J|=k+1}}\max_{(s_{i,j})_{(i,j)\in J}\in\{-1,1\}^J} 2\binom{m}{k+1}^{1/p}\frac{\left\|\sum_{(i,j)\in J}\sigma_{i,j}v_h(i,j)\right\|_p}{k+1}
\end{aligned}$$

We note that $\left\|\sum_{(i,j)\in J}\sigma_{i,j}v_h(i,j)\right\|_p = \left\|\sum_{x\in U}\sum_{(i,j)\in J}\sigma(i,x)s_{i,j}\left[h(i,x)=j\right]w_x\right\|_p$. Since we have that $p \geq \log m$ then $k \geq 1$ which implies that $k+1 \leq 2\frac{p}{\log(m/p)}$. We then get that $\binom{m}{k+1}^{1/p} \leq \left(\frac{em}{2p/\log(m/p)}\right)^{2/\log(m/p)} \leq 2e^3$. We now use eq. (B.6) to get that,

$$\begin{aligned}
\left\|\sum_{x\in U}\sum_{(i,j)\in J}\sigma(i,x)s_{i,j}\left[h(i,x)=j\right]w_x\right\|_p &= \left\|\sum_{i\in[s]}\sum_{x\in U}\sigma(i,x)\left[(i,h(i,x))\in J\right]s_{i,h(i,x)}w_x\right\|_p \\
&\leq \sqrt{L\frac{p}{\log m/s}}\sqrt{|J|}\left\|w\right\|_2 \\
&= \sqrt{L\frac{p}{\log m/s}}\sqrt{k+1}\left\|w\right\|_2
\end{aligned}$$

Combining this we get that $\|v_h'(\pi(k+1))\|_p \leq 4e^3\sqrt{L\frac{p}{\log m/s}}\frac{\|w\|_2}{\sqrt{k+1}}$. We then obtain that,

$$\begin{aligned}\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p &\\ \leq \sqrt{2}\Psi_p\left(4e^3L\sqrt{L\frac{p}{(k+1)\log m/s}}\|w\|_\infty\|w\|_2, L^2\frac{s^2}{m}\|w\|_2^4\right)&\\ \leq \sqrt{2}\Psi_p\left(4e^3L\sqrt{L\frac{\log m/p}{\log m/s}}\|w\|_2^2, L^2\frac{s^2}{m}\|w\|_2^4\right)&\end{aligned}$$

If $\log m/p \leq 4\log m/s$ then we get that,

$$\begin{aligned}\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p &\leq \sqrt{2}\Psi_p\left(16e^3L^{3/2}\|w\|_2^2, L^2\frac{s^2}{m}\|w\|_2^4\right)\\ &\leq \Psi_p\left(32e^3L^{3/2}\|w\|_2^2, 2L^2\frac{s^2}{m}\|w\|_2^4\right)\end{aligned}$$

If $\log m/p > 4\log m/s$ then $m/p > (m/s)^4$ which implies that $\frac{pm}{s^2} \leq \sqrt{p}m$. Using this we get that $\frac{p16e^6L^3\frac{\log m/p}{\log m/s}\|w\|_2^4}{L^2\frac{s^2}{m}\|w\|_2^4} \leq 16e^6L\sqrt{p}m\log m/p \leq 16e^6L$. Where we have used that $\sqrt{s}\log 1/x \leq 1$. Now we use Lemma B.10 to get that

$$\begin{aligned}\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p &\leq \sqrt{2}\Psi_p\left(32e^3L^{3/2}\|w\|_2^2, L^2\frac{s^2}{m}\|w\|_2^4\right)\\ &\leq \sqrt{2}\sqrt{pL^3 16e^6\frac{s^2}{m}\|w\|_2^4}\\ &\leq \Psi_p\left(8e^3L^{3/2}\|w\|_2, 32e^6L^3\frac{s^2}{m}\|w\|_2^4\right)\end{aligned}$$

Now let us consider the case where $p < \log m$. By eq. (B.8), we get that

$$\left\|v_h'(\pi(k+1))\right\|_p \leq \left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\left[h(i,x)=j\right]w_x\right|\right\|_p \leq e\sqrt{L\frac{\log m}{\log m/s}}\|w\|_2$$

We then obtain that,

$$\begin{aligned}\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p &\leq \sqrt{2}\Psi_p\left(eL\sqrt{L\frac{\log m}{\log m/s}}\|w\|_\infty\|w\|_2, L\frac{s^2}{m}\|w\|_2^4\right)\\ &\leq \sqrt{2}\Psi_p\left(eL\sqrt{L\frac{\log m}{\log m/s}}\|w\|_2^2, L\frac{s^2}{m}\|w\|_2^4\right)\end{aligned}$$

If $s \leq m^{3/4}$ then we get that $\log m \leq 4 \log m/s$ and

$$\sqrt{2}\Psi_p\left(eL\sqrt{L\frac{\log m}{\log m/s}}\left\|w\right\|_2^2, L\frac{s^2}{m}\left\|w\right\|_2^4\right) \leq \sqrt{2}\Psi_p\left(4eL^{3/2}\left\|w\right\|_2^2, L\frac{s^2}{m}\left\|w\right\|_2^4\right)$$

as wanted. If $s \geq m^{3/4}$ then we get that $\frac{peL^3 \log m}{L^2 s^2/m} \leq \frac{eLm \log^2 m}{s^2} \leq \frac{eL \log^2 m}{m^{1/2}} \leq 16/eL$, where we have used that $\sqrt{x}\log^2 1/x \leq 16/e^2$. Again we use Lemma B.10 to get that

$$\begin{aligned}\sqrt{2}\Psi_p\left(4eL^{3/2}\left\|w\right\|_2^2, L\frac{s^2}{m}\left\|w\right\|_2^4\right) &\leq \sqrt{p32/eL^3\frac{s^2}{m}}\left\|w\right\|_2^2 \\ &\leq \Psi_p\left(8L^{3/2}\left\|w\right\|_2^2, 32L^3\frac{s^2}{m}\left\|w\right\|_2^4\right).\end{aligned}$$

Combining everything we get that

$$\begin{aligned}\left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p &\leq \Psi_p\left(32e^3L^{3/2}\left\|w\right\|_2, 32e^6L^3\frac{s^2}{m}\left\|w\right\|_2^4\right), \\ \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h'(i,\overline{h}(i,y))w_y\right\|_p &\leq 4e^3L\frac{p}{\log m/s}\left\|w\right\|_2^2.\end{aligned}$$

Now we conclude that

$$\begin{aligned}&\left\|\sum_{i\in[s],j\in[m/s]} v_h(i,j)v_{\overline{h}}(i,j)\right\|_p \\ &\quad\leq \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h'(i,\overline{h}(i,y))w_y\right\|_p + \left\|\sum_{i\in[s]}\sum_{y\in U}\overline{\sigma}(i,y)v_h''(i,\overline{h}(i,y))w_y\right\|_p \\ &\quad\leq \Psi_p\left(32e^3L^{3/2}\left\|w\right\|_2^2, 32e^6L^3\left\|w\right\|_2^4\right) + 4e^3L\frac{p}{\log m/s}\left\|w\right\|_2^2.\end{aligned}$$

Thus finishing the proof. □

B.6 Analysis of Mixed Tabulation Hashing

The goal of this section is to prove our main result for Mixed Tabulation hashing. The main new results is in Appendix B.6.2 where we show that Mixed Tabulation has a decoupling-decomposition. In Appendix B.6.3, we show that Mixed Tabulation also has the strong concentration property. The proofs in Appendix B.6.3 are modifications of those found in [HT22] for Mixed Tabulation hashing.

B.6.1 Notation and Previous Results for Tabulation Hashing

We will need to reason about the individual characters of a key, $x \in \Sigma^c$, and for that, we need some notation.

Definition B.11 (Position characters)**.** Let Σ be an alphabet and $c > 0$ a positive integer. We call an element $(i, y) \in [c] \times \Sigma$ a position character of Σ^c.

We will view a key $x = (y_0, \ldots, y_{c-1}) \in \Sigma^c$ as a set of c position characters, $\{(0, y_0), \ldots (c-1, y_{c-1})\} \subseteq [c] \times \Sigma$. We define the sets $P_{partial}$ and P_{prefix} which contains partial keys.

$$P_{partial} = \left\{ \{(i, \alpha_i)\}_{i \in I} \,\middle|\, \emptyset \neq I \subseteq [c], \forall i \in I : \alpha_i \in \Sigma \right\}$$
$$P_{prefix} = \left\{ \{(i, \alpha_i)\}_{i \in I} \,\middle|\, \emptyset \neq I \subseteq [c],\ I \text{ is an interval containing 0, and } \forall i \in I : \alpha_i \in \Sigma \right\}$$

For a partial key, $\pi = \{(i_0, \alpha_0), \ldots, (i_{k-1}, \alpha_{k-1})\} \in P_{partial}$, we define $I_\pi = \{i_0, \ldots, i_{k-1}\}$ to be the set of used positions. We will also write $|\pi| = |I_\pi|$.

For a simple tabulation hashing function, $h \colon \Sigma^c \to R$, we will let h_I for $I \subseteq [c]$ to be the hash function that only looks at the positions in I, i.e., $h_I(x) = \bigoplus_{i \in I} T_i[x_i]$. Similarly, for $h_2 \colon \Sigma^c \to \Sigma^d$ we will define h^I for $I \subseteq [d]$ to be the partial key restricted to the positions I.

For $\pi \in P_{partial}$ and $\tau \in P_{prefix}$, we define the random set $U_{\pi,\tau}$ as follows

$$U_{\pi,\tau} = \{x \in \Sigma^c \mid \pi \subseteq x, \tau \subseteq h_2(x)\}$$

We also need the following lemmas from [HT22].

Lemma B.12. *Let $h \colon U \to [m]$ be a uniformly random function, let $v \colon U \times [m] \to \mathbb{R}$ be a fixed value function, and assume that $\sum_{j \in [m]} v(x, j) = 0$ for all keys $x \in U$. Define the random variable $X_v = \sum_{x \in U} v(x, h(x))$. Then for all $p \geq 2$,*

$$\|X_v\|_p \leq L \Psi_p \left(M_v, \sigma_v^2\right) ,$$

where $L \leq 16e$ is a universal constant.

Lemma B.13. *Let $h \colon U \to [m]$ be a uniformly random function, let $\varepsilon \colon U \to \{-1, 1\}$ be a uniformly random sign function, and let $v : U \times [m] \to \mathbb{R}$ be a fixed value function. Then for all $p \geq 2$,*

$$\left\| \sum_{x \in U} \varepsilon(s) v(x, h(x)) \right\|_p \leq L \sqrt{\frac{p}{\log(m)}} \|v\|_2 ,$$

where $L \leq e$ is a universal constant.

Lemma B.14. *Let $h \colon \Sigma^c \to [m]$ be a simple tabulation hash function, $v \colon \Sigma^c \times [m] \to \mathbb{R}$ a value function, and assume that $\sum_{j \in [m]} v(x, j) = 0$ for all keys $x \in U$. Then for all $p \geq 2$,*

$$\left\| \sum_{x \in \Sigma^c} v(x, h(x)) \right\|_p \leq L_1 \Psi_p \left(K_c \gamma_p^{c-1} \|v\|_\infty , K_c \gamma_p^{c-1} \frac{\|v\|_2^2}{m} \right) ,$$

where $K_c = (L_2 c)^{c-1}$, L_1 *and* L_2 *are universal constants, and*

$$\gamma_p = \frac{\max\left\{\log(m) + \log\left(\frac{\sum_{x\in\Sigma^c}\|v[x]\|_2^2}{\max_{x\in\Sigma^c}\|v[x]\|_2^2}\right)/c, p\right\}}{\log\left(e^2 m\left(\max_{x\in\Sigma^c}\frac{\|v[x]\|_1^2}{\|v[x]\|_2^2}\right)^{-1}\right)}$$

Lemma B.15. *Let* $h\colon \Sigma^c \to [m]$ *be a simple tabulation function,* $\varepsilon\colon \Sigma^c \to \{-1,1\}$ *be a simple tabulation sign function, and* $v_i\colon \Sigma^c \times [m] \to \mathbb{R}$ *be value function for* $i \in [k]$*. For every real number* $p \geq 2$*,*

$$\left\|\sum_{j\in[m]}\left(\sum_{x\in\Sigma^c}\varepsilon(x)v(x,h(x)\oplus j)\right)^2\right\|_p \leq \left(\frac{Lc\max\{p,\log(m)\}}{\log\left(\frac{e^2 m\sum_{x\in\Sigma^c}\sum_{j\in[m]}v(x,j)^2}{\sum_{x\in\Sigma^c}(\sum_{j\in[m]}|v(x,j)|)^2}\right)}\right)^c \|v\|_2^2 \ ,$$

where L *is a universal constant.*

B.6.2 Decoupling

Before proving the decoupling-decomposition for Mixed Tabulation hashing, we a general decoupling lemma. We start by stating a standard decoupling result as found in [Ver18].

Lemma B.16. *Let* $f_{ij}\colon T\times T \to \mathbb{R}$ *be functions for* $i,j \in [n]$*. Let* $(X_i)_{i\in[n]}$ *be independent random variables with values in* T *such that* $\mathrm{E}[f_{ij}(X_i,X_j)\,|\,X_j] = \mathrm{E}[f_{ij}(X_i,X_j)\,|\,X_i] = 0$ *for all* $i \neq j \in [n]$*. Then, for every convex function* $F\colon \mathbb{R}\to\mathbb{R}$*, one has*

$$\mathrm{E}\left[F(\sum_{i\neq j\in[n]} f_{ij}(X_i,X_j))\right] \leq \mathrm{E}\left[F(4\sum_{i,j\in[n]} f_{ij}(X_i,X'_j))\right] \ ,$$

where $(X'_i)_{i\in[n]}$ *is an independent copy of* $(X_i)_{i\in[n]}$*.*

We slightly generalize the decoupling result as follows.

Lemma B.17. *Let* $f_{i_0,\ldots,i_{c-1},j_0,\ldots,j_{c-1}}\colon T^c \times T^c \to \mathbb{R}$ *be functions for* $i_0\ldots,i_{c-1},j_0,\ldots,j_{c-1} \in [n]$*. Let* $(X_i^{(k)})_{i\in[n],k\in[c]}$ *be independent random variables with values in* T *such that*

$$\begin{aligned}
&\mathrm{E}\Big[f_{i_0,\ldots,i_{c-1},j_0,\ldots,j_{c-1}}((X_{i_0}^{(0)},\ldots,X_{i_{c-1}}^{(c-1)}),(X_{j_0}^{(0)},\ldots,X_{j_{c-1}}^{(c-1)}))\,\Big|\,(X_{i_k}^{(k)})_{k\in[c]\setminus\{l\}},(X_{j_k}^{(k)})_{k\in[c]}\Big] = 0,\\
&\mathrm{E}\Big[f_{i_0,\ldots,i_{c-1},j_0,\ldots,j_{c-1}}((X_{i_0}^{(0)},\ldots,X_{i_{c-1}}^{(c-1)}),(X_{j_0}^{(0)},\ldots,X_{j_{c-1}}^{(c-1)}))\,\Big|\,(X_{i_k}^{(k)})_{k\in[c]},(X_{j_k}^{(k)})_{k\in[c]\setminus\{l\}}\Big] = 0,
\end{aligned} \tag{B.9}$$

for all $i_0 \dots, i_{c-1}, j_0, \dots, j_{c-1} \in [n]$ *and all* $l \in [c]$ *with* $i_k \neq j_k$ *for all* $k \in [c]$*. Then, for every convex function* $F \colon \mathbb{R} \to \mathbb{R}$*, one has*

$$\mathrm{E}\left[F\left(\sum_{\substack{i_0\dots,i_{c-1},j_0,\dots,j_{c-1}\in[n]\\ \forall k\in[c]:i_k\neq j_k}} f_{i_0,\dots,i_{c-1},j_0,\dots,j_{c-1}}((X^{(0)}_{i_0},\dots,X^{(c-1)}_{i_{c-1}}),(X^{(0)}_{j_0},\dots,X^{(c-1)}_{j_{c-1}}))\right)\right]$$
$$\leq \mathrm{E}\left[F\left(4^c \sum_{i_0\dots,i_{c-1},j_0,\dots,j_{c-1}\in[n]} f_{i_0,\dots,i_{c-1},j_0,\dots,j_{c-1}}((X^{(0)}_{i_0},\dots,X^{(c-1)}_{i_{c-1}}),(\hat{X}^{(0)}_{j_0},\dots,\hat{X}^{(c-1)}_{j_{c-1}}))\right)\right], \tag{B.10}$$

where $(\hat{X}^{(k)}_i)_{i\in[n],k\in[c]}$ *is an independent copy of* $(X^{(k)}_i)_{i\in[n],k\in[c]}$*.*

Proof. The will by induction on c, and the induction start, $c = 1$, is exactly Lemma B.16. Now assume that $c > 1$ and that eq. (B.10) holds for $c - 1$.

Throughout the proof, we use $\mathbf{i}$ and $\mathbf{j}$ as shortcuts for $i_0, \dots, i_{c-2}$ and $j_0, \dots, j_{c-2}$ respectively. We also use $X_{\mathbf{i}}$, $X_{\mathbf{j}}$, and $\hat{X}_{\mathbf{j}}$ to denote $(X^{(0)}_{i_0}, \dots, X^{(c-2)}_{i_{c-2}})$, $(X^{(0)}_{j_0}, \dots, X^{(c-2)}_{j_{c-2}})$, and $(\hat{X}^{(0)}_{i_0}, \dots, \hat{X}^{(c-2)}_{i_{c-2}})$ respectively.

We start by defining the functions, $g_{i_{c-1},j_{c-1}} \colon T \times T \to \mathbb{R}$ for all $i_{c-1}, j_{c-1} \in [n]$, by

$$g_{i_{c-1},j_{c-1}}(x,y) = \sum_{\substack{i_0\dots,i_{c-2},j_0,\dots,j_{c-2}\in[n]\\ \forall k\in[c-1]:i_k\neq j_k}} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((X_{\mathbf{i}},x),(X_{\mathbf{j}},y)).$$

We note that eq. (B.9) implies that

$$\mathrm{E}\Big[g_{i_{c-1},j_{c-1}}(X^{(c-1)}_{i_{c-1}},X^{(c-1)}_{j_{c-1}}) \,\Big|\, X^{(c-1)}_{i_{c-1}},(X^{(k)}_i)_{i\in[n],k\in[c-1]}\Big] = 0\,,$$
$$\mathrm{E}\Big[g_{i_{c-1},j_{c-1}}(X^{(c-1)}_{i_{c-1}},X^{(c-1)}_{j_{c-1}}) \,\Big|\, X^{(c-1)}_{j_{c-1}},(X^{(k)}_i)_{i\in[n],k\in[c-1]}\Big] = 0\,,$$

for all $i_{c-1} \neq j_{c-1} \in [n]$. This implies that we can use Lemma B.16 while conditioning on

$(X_i^{(k)})_{i\in[n],k\in[c-1]}$ to get that

$$\begin{aligned}
&\mathrm{E}\left[F\left(\sum_{\substack{i_0\dots,i_{c-1},j_0,\dots,j_{c-1}\in[n]\\ \forall k\in[c]:i_k\neq j_k}} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((X_{\mathbf{i}},X_{i_{c-1}}^{(c-1)}),(X_{\mathbf{j}},X_{j_{c-1}}^{(c-1)}))\right)\right]\\
&\quad=\mathrm{E}\left[\mathrm{E}\left[F\left(\sum_{i_{c-1}\neq j_{c-1}\in[n]} g_{i_{c-1},j_{c-1}}(X_{i_{c-1}}^{(c-1)},X_{c-1}^{(c-1)})\right)\middle|(X_i^{(k)})_{i\in[n],k\in[c-1]}\right]\right]\\
&\quad\leq\mathrm{E}\left[\mathrm{E}\left[F\left(4\sum_{i_{c-1},j_{c-1}\in[n]} g_{i_{c-1},j_{c-1}}(X_{i_{c-1}}^{(c-1)},\hat{X}_{c-1}^{(c-1)})\right)\middle|(X_i^{(k)})_{i\in[n],k\in[c-1]}\right]\right]\\
&\quad=\mathrm{E}\left[F\left(4\sum_{\substack{i_0\dots,i_{c-1},j_0,\dots,j_{c-1}\in[n]\\ \forall k\in[c-1]:i_k\neq j_k}} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((X_{\mathbf{i}},X_{i_{c-1}}^{(c-1)}),(X_{\mathbf{j}},\hat{X}_{j_{c-1}}^{(c-1)}))\right)\right].
\end{aligned}$$

We then define the functions, $h_{\mathbf{i},\mathbf{j}}\colon T^{c-1}\times T^{c-1}\to\mathbb{R}$ for all $i_0\dots,i_{c-2},j_0,\dots,j_{c-2}\in[n]$, by

$$\begin{aligned}
&h_{\mathbf{i},\mathbf{j}}((x_0,\dots,x_{c-2}),(y_0,\dots,x_{c-2}))\\
&\qquad=\sum_{i_{c-1},j_{c-1}\in[n]} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((x_0,\dots,x_{c-2},X_{i_{c-1}}^{(c-1)}),(y_0,\dots,y_{c-2},\hat{X}_{j_{c-1}}^{(c-1)}))\,.
\end{aligned}$$

Now let $i_0\dots,i_{c-2},j_0,\dots,j_{c-2}\in[n]$ and $l\in[c-1]$ with $i_k\neq j_k$ for all $k\in[c-1]$ and let $H=h_{i_0,\dots,i_{c-2},j_0,\dots,j_{c-2}}$. Then again by eq. (B.9) we get that for

$$\begin{aligned}
\mathrm{E}\Big[h_{\mathbf{i},\mathbf{j}}(X_{\mathbf{i}},X_{\mathbf{j}})\,\Big|\,(X_{i_k}^{(k)})_{k\in[c-1]\setminus\{l\}},(X_{j_k}^{(k)})_{k\in[c-1]},(X_i^{(c-1)},\hat{X}_i^{(c-1)})_{i\in[n]}\Big]=0,\\
\mathrm{E}\Big[h_{\mathbf{i},\mathbf{j}}(X_{\mathbf{i}},X_{\mathbf{j}})\,\Big|\,(X_{i_k}^{(k)})_{k\in[c-1]},(X_{j_k}^{(k)})_{k\in[c-1]\setminus\{l\}},(X_i^{(c-1)},\hat{X}_i^{(c-1)})_{i\in[n]}\Big]=0.
\end{aligned}$$

We can then use the induction hypothesis conditioned on $(X_i^{(c-1)}, \hat{X}_i^{(c-1)})_{i\in[n]}$ to get that[3]

$$\begin{aligned}
&\mathrm{E}\left[F\left(4\sum_{\substack{i_0\ldots,i_{c-1},j_0,\ldots,j_{c-1}\in[n]\\ \forall k\in[c-1]:i_k\neq j_k}} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((X_{\mathbf{i}},X_{i_{c-1}}^{(c-1)}),(X_{\mathbf{j}},\hat{X}_{j_{c-1}}^{(c-1)}))\right)\right]\\
&=\mathrm{E}\left[\mathrm{E}\left[F\left(4\sum_{\substack{i_0\ldots,i_{c-2},j_0,\ldots,j_{c-2}\in[n]\\ \forall k\in[c-1]:i_k\neq j_k}} h_{\mathbf{i},\mathbf{j}}(X_{\mathbf{i}},X_{\mathbf{j}})\right)\middle|(X_i^{(c-1)},\hat{X}_i^{(c-1)})_{i\in[n]}\right]\right]\\
&\leq\mathrm{E}\left[\mathrm{E}\left[F\left(4^c\sum_{i_0\ldots,i_{c-2},j_0,\ldots,j_{c-2}\in[n]} h_{\mathbf{i},\mathbf{j}}(X_{\mathbf{i}},\hat{X}_{\mathbf{j}})\right)\middle|(X_i^{(c-1)},\hat{X}_i^{(c-1)})_{i\in[n]}\right]\right]\\
&=\mathrm{E}\left[F\left(4^c\sum_{i_0\ldots,i_{c-1},j_0,\ldots,j_{c-1}\in[n]} f_{\mathbf{i},i_{c-1},\mathbf{j},j_{c-1}}((X_{i_0}^{(0)},\ldots,X_{i_{c-1}}^{(c-1)}),(\hat{X}_{j_0}^{(0)},\ldots,\hat{X}_{j_{c-1}}^{(c-1)}))\right)\right].
\end{aligned}$$

This finishes the induction step and thus the proof. □

We are now ready to prove a decoupling lemma for Mixed Tabulation.

Lemma B.18.

$$\begin{aligned}
&\left\|\sum_{i\in[s]}\sum_{x,y\in[u]}[h_2(x)\neq h_2(y)]\,\sigma(i,x)\sigma(i,y)\,[h(i,x)=h(i,y)]\,w_xw_y\right\|_p\\
&\leq\sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} 4^{c+1}\left\|\sum_{i\in[s]}\sum_{x,y\in U_{\pi,\tau}}\sigma_{I_\pi,|\tau|}(i,x)\overline{\sigma_{I_\pi,|\tau|}}(i,y)\left[h_{I_\pi,|\tau|}(i,x)=\overline{h_{I_\pi,|\tau|}}(i,y)\right]w_xw_y\right\|_p
\end{aligned}\tag{B.11}$$

[3]

where

$$
\begin{aligned}
h_{I,r}(i,x) &= \Big(h_{1,I}(x) \oplus h_{3,\{r\}}(h_{2,I}^{\{r\}}(x) \oplus i \oplus h_{2,I^c}^{\{r\}}(x))\Big) \\
&\quad \oplus \Big(h_{1,I^c}(x) \oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(x) \oplus i^{\otimes(d-1)})\Big) \\
\sigma_{I,r}(i,x) &= \Big(\sigma_{1,I}(x) \oplus \sigma_{3,\{r\}}(h_{2,I}^{\{r\}}(x) \oplus i \oplus h_{2,I^c}^{\{r\}}(x))\Big) \\
&\quad \oplus \Big(\sigma_{1,I^c}(x) \oplus \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(x) \oplus i^{\otimes(d-1)})\Big) \\
\overline{h_{I,r}}(i,y) &= \Big(\overline{h_{1,I}}(y) \oplus \overline{h_{3,\{r\}}}(\overline{h_{2,I}^{\{r\}}}(y) \oplus i \oplus h_{2,I^c}^{\{r\}}(y))\Big) \\
&\quad \oplus \Big(h_{1,I^c}(y) \oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(y) \oplus i^{\otimes(d-1)})\Big) \\
\overline{\sigma_{I,r}}(i,y) &= \Big(\overline{\sigma_{1,I}}(y) \oplus \overline{\sigma_{3,\{r\}}}(\overline{h_{2,I}^{\{r\}}}(y) \oplus i \oplus h_{2,I^c}^{\{r\}}(y))\Big) \\
&\quad \oplus \Big(\sigma_{1,I^c}(y) \oplus \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(y) \oplus i^{\otimes(d-1)})\Big)
\end{aligned}
$$

Futhermore, we have that

$$
\sum_{\pi} \sum_{\tau} \sum_{x \in U_{\pi,\tau}} w_x^2 \leq d2^c \left\|w\right\|_2^2 \tag{B.12}
$$

for all $w \in \mathbb{R}^u$.

Proof. We will start by proving eq. (B.12). Let $x \in [u]$, now it is easy to see that there exists 2^c, $\pi \in P_{partial}$ such that $\pi \subseteq x$ and similarly there exists d, $\tau \in P_{prefix}$ such that $\tau \subseteq h_2(x)$. We there have that there $d2^c$ pairs $(\pi, \tau) \in P_{partial} \times P_{prefix}$ such that $x \in U_{\pi,\tau}$ and thus eq. (B.12) follows.

We will now focus on proving eq. (B.11). We start by defining the events $A_{x,y,\tau}$ for $x, y \in [u]$ and $\tau \in P_{prefix}$ by $[A_{x,y,\tau}] = \big[h_{2,[|\tau|]}(x) = h_{2,[|\tau|]}(y) = \tau\big]\big[h_{2,\{|\tau|\}}(x) \neq h_{2,\{|\tau|\}}(y)\big]$. We can then write

$$
\begin{aligned}
&\left\| \sum_{i \in [s]} \sum_{x,y \in [u]} \left[h_2(x) \neq h_2(y)\right] \sigma(i,x)\sigma(i,y) \left[h(i,x) = h(i,y)\right] w_x w_y \right\|_p \\
&= \left\| \sum_{\tau \in P_{prefix}} \sum_{i \in [s]} \sum_{x,y \in [u]} \left[A_{x,y,\tau}\right] \sigma(i,x)\sigma(i,y) \left[h(i,x) = h(i,y)\right] w_x w_y \right\|_p \\
&\leq \sum_{\tau \in P_{prefix}} \left\| \sum_{i \in [s]} \sum_{x,y \in [u]} \left[A_{x,y,\tau}\right] \sigma(i,x)\sigma(i,y) \left[h(i,x) = h(i,y)\right] w_x w_y \right\|_p
\end{aligned}
$$

We then define the functions $h_r, \overline{h_r} \colon [s] \times [u] \to [m/s]$ and $\sigma_r, \overline{\sigma_r} \colon [s] \times [u]$ by

$$\begin{aligned}
h_r(i,x) &= h_1(x) \oplus h_{3,\{r\}}(h_2^{\{r\}}(x) \oplus i) \oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(x) \oplus i^{\otimes(d-1)}) \\
\sigma_r(i,x) &= \sigma_1(x)\sigma_{3,\{r\}}(h_2^{\{r\}}(x) \oplus i) \cdot \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(x) \oplus i^{\otimes(d-1)}) \\
\overline{h_r}(i,y) &= h_1(y) \oplus \overline{h_{3,\{r\}}}(h_2^{\{r\}}(y) \oplus i) \oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(y) \oplus i^{\otimes(d-1)}) \\
\overline{\sigma_r}(i,y) &= \sigma_1(x)\overline{\sigma_{3,\{r\}}}(h_2^{\{r\}}(x) \oplus i) \cdot \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(x) \oplus i^{\otimes(d-1)})
\end{aligned}$$

For a fixed $\tau \in P_{prefix}$, we see that the expression

$$\sum_{i \in [s]} \sum_{x,y \in [u]} [A_{x,y,\tau}]\, \sigma(i,x)\sigma(i,y)\, [h(i,x) = h(i,y)]\, w_x w_y$$

satisfies the requirement of Lemma B.16 for the random variables $(h_{3,\{|\tau|\}}(\alpha), \sigma_{3,\{|\tau|\}}(\alpha))_{\alpha \in \Sigma}$. So applying Lemma B.16 we get that

$$\begin{aligned}
&\sum_{\tau \in P_{prefix}} \left\| \sum_{i \in [s]} \sum_{x,y \in [u]} [A_{x,y,\tau}]\, \sigma(i,x)\sigma(i,y)\, [h(i,x) = h(i,y)]\, w_x w_y \right\|_p \\
&\leq \sum_{\tau \in P_{prefix}} 4 \left\| \sum_{i \in [s]} \sum_{x,y \in [u]} [x \neq y]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\, \left[h_{|\tau|}(i,x) = \overline{h_{|\tau|}}(i,y)\right] w_x w_y \right\|_p
\end{aligned}$$

We can now rewrite this expression as follows,

$$\begin{aligned}
&\sum_{\tau \in P_{prefix}} \left\| \sum_{i \in [s]} \sum_{x,y \in [u]} [x \neq y]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\, \left[h_{|\tau|}(i,x) = \overline{h_{|\tau|}}(i,y)\right] w_x w_y \right\|_p \\
&= \sum_{\tau \in P_{prefix}} \left\| \sum_{\pi \in P_{partial}} \sum_{i \in [s]} \sum_{x,y \in [u]} [x \cap y = \pi]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\, \left[h_{|\tau|}(i,x) = \overline{h_{|\tau|}}(i,y)\right] w_x w_y \right\|_p \\
&\leq \sum_{\substack{\pi \in P_{partial} \\ \tau \in P_{prefix}}} \left\| \sum_{i \in [s]} \sum_{x,y \in [u]} [x \cap y = \pi]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\, \left[h_{|\tau|}(i,x) = \overline{h_{|\tau|}}(i,y)\right] w_x w_y \right\|_p
\end{aligned}$$

For fixed $\pi \in P_{partial}$ and $\tau \in P_{prefix}$, we see that the expression

$$\sum_{i \in [s]} \sum_{x,y \in [u]} [x \cap y = \pi]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\, \left[h_{|\tau|}(i,x) = \overline{h_{|\tau|}}(i,y)\right] w_x w_y$$

satisfies the requirement of Lemma B.17 for the random variables $(h_{1,I_\pi^c}(x), h_{2,I_\pi^c}(x), \sigma_{1,I_\pi^c}(x), \sigma_{2,I_\pi^c}(x))_{x\in[u]}$. So applying Lemma B.16 we get that

$$\sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} \left\| \sum_{i\in[s]} \sum_{x,y\in[u]} [x\cap y=\pi]\, \sigma_{|\tau|}(i,x)\overline{\sigma_{|\tau|}}(i,y)\left[h_{|\tau|}(i,x)=\overline{h_{|\tau|}}(i,y)\right] w_x w_y \right\|_p$$

$$\leq \sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} 4^c \left\| \sum_{i\in[s]} \sum_{x,y\in[u]} [x\cap y=\pi]\, \sigma_{I_\pi,|\tau|}(i,x)\overline{\sigma_{I_\pi,|\tau|}}(i,y)\left[h_{I_\pi,|\tau|}(i,x)=\overline{h_{I_\pi,|\tau|}}(i,y)\right] w_x w_y \right\|_p$$

This finishes the proof of eq. (B.11) and thus the lemma. □

Finally, we can prove that Mixed Tabulation has an $(p, 4^{c+2}, 4\varepsilon^{-2}3^c\frac{s}{m}|\Sigma|^{-d})$-decoupling-decomposition.

Lemma B.19.

$$\Pr[|Z|\geq\varepsilon]$$

$$\leq \left(\varepsilon^{-1} \sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} \frac{4^{c+4}}{s} \left\| \sum_{i\in[s]} \sum_{x,y\in U_{\pi,\tau}} \sigma_{I_\pi,|\tau|}(i,x)\overline{\sigma_{I_\pi,|\tau|}}(i,y)\left[h_{I_\pi,|\tau|}(i,x)=\overline{h_{I_\pi,|\tau|}}(i,y)\right] w_x w_y \right\|_p \right)^p + 4\varepsilon^{-2}3^c\frac{s}{m}|\Sigma|^{-d},$$

where

$$\begin{aligned}
h_{I,r}(i,x) &= \left(h_{1,I}(x)\oplus h_{3,\{r\}}(h_{2,I}^{\{r\}}(x)\oplus i\oplus h_{2,I^c}^{\{r\}}(x))\right)\\
&\quad\oplus\left(h_{1,I^c}(x)\oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(x)\oplus i^{\otimes(d-1)})\right)\\
\sigma_{I,r}(i,x) &= \left(\sigma_{1,I}(x)\oplus \sigma_{3,\{r\}}(h_{2,I}^{\{r\}}(x)\oplus i\oplus h_{2,I^c}^{\{r\}}(x))\right)\\
&\quad\oplus\left(\sigma_{1,I^c}(x)\oplus \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(x)\oplus i^{\otimes(d-1)})\right)\\
\overline{h_{I,r}}(i,y) &= \left(\overline{h_{1,I}}(y)\oplus \overline{h_{3,\{r\}}}(\overline{h_{2,I}^{\{r\}}}(y)\oplus i\oplus h_{2,I^c}^{\{r\}}(y))\right)\\
&\quad\oplus\left(h_{1,I^c}(y)\oplus h_{3,\{r\}^c}(h_2^{\{r\}^c}(y)\oplus i^{\otimes(d-1)})\right)\\
\overline{\sigma_{I,r}}(i,y) &= \left(\overline{\sigma_{1,I}}(y)\oplus \overline{\sigma_{3,\{r\}}}(\overline{h_{2,I}^{\{r\}}}(y)\oplus i\oplus h_{2,I^c}^{\{r\}}(y))\right)\\
&\quad\oplus\left(\sigma_{1,I^c}(y)\oplus \sigma_{3,\{r\}^c}(h_2^{\{r\}^c}(y)\oplus i^{\otimes(d-1)})\right)
\end{aligned}$$

Futhermore, we have that

$$\sum_\pi\sum_\tau\sum_{x\in U_{\pi,\tau}} w_x^2 \leq d2^c\|w\|_2^2$$

for all $w \in \mathbb{R}^u$.

Proof. We use a union bound to obtain that,

$$\Pr[|Z| \geq \varepsilon] \leq \Pr\left[\left|\sum_{i\in[s]}\sum_{x\neq y\in[u]} \sigma_1(x)\sigma_1(y)\left[(h_1(x),h_2(x)) = (h_1(y),h_2(y))\right] w_x w_y\right| \geq \varepsilon/2\right] + \Pr\left[\left|\sum_{i\in[s]}\sum_{x,y\in[u]} \left[h_2(x)\neq h_2(y)\right]\sigma(i,x)\sigma(i,y)\left[h(i,x)=h(i,y)\right] w_x w_y\right| \geq \varepsilon/2\right]$$

For the first term, we use that

$$\mathrm{E}\left[\left(\sum_{i\in[s]}\sum_{x\neq y\in[u]} \sigma_1(x)\sigma_1(y)\left[(h_1(x),h_2(x)) = (h_1(y),h_2(y))w_x w_y\right]\right)^2\right] \leq 3^c \left\|w\right\|_2^4 \frac{s}{m}\left|\Sigma\right|^{-d}.$$

So applying Markov's inequality, we get that,

$$\Pr\left[\left|\sum_{i\in[s]}\sum_{x\neq y\in[u]} \sigma_1(x)\sigma_1(y)\left[(h_1(x),h_2(x)) = (h_1(y),h_2(y))\right] w_x w_y\right| \geq \varepsilon/2\right] \leq 4\varepsilon^{-2}3^c\frac{s}{m}\left|\Sigma\right|^{-d}.$$

For the second term we apply Markov's inequality for p and then the result follows by Lemma B.18. □

B.6.3 Concentration

The goal of this section is to show that Mixed Tabulation is (p,γ_p^c)-strongly-concentrated where $\gamma_p = Kc\max\left\{1, \frac{p}{\log|\Sigma|}\right\}$ for a universal constant K. This is done in next 3 lemmas that prove the individual parts of the strong concentration. They all follow the same blueprint as the results in [HT22].

Lemma B.20. *For any value function,* $v\colon [s]\times[m/s]$*, and any vector* $w \in \mathbb{R}^U$ *the following concentration results for Mixed Tabulation hashing holds*

$$\left\|\sum_{i\in[s]}\sum_{x\in U}\sigma(i,x)v(i,h(i,x))w_x\right\|_p \leq \Psi_p\left(\gamma_p^c\left\|v\right\|_\infty\left\|w\right\|_\infty, \gamma_p^c\frac{s}{m}\left\|v\right\|_2^2\left\|w\right\|_2^2\right), \tag{B.13}$$

$$\left\|\sum_{i\in[s]}\sum_{x\in U}\sigma(i,x)v(i,h(i,x))w_x\right\|_p \leq \sqrt{\gamma_p^c\frac{p}{\log m/s}}\left\|v\right\|_2\left\|w\right\|_2. \tag{B.14}$$

Here $\gamma_p = Kc\max\left\{1, \frac{p}{\log|\Sigma|}\right\}$ *for a universal constant* K.

Proof. We start by rewriting the expression

$$\left\| \sum_{i\in[s]} \sum_{x\in U} \sigma(i,x) v(i,h(i,x)) w_x \right\|_p$$
$$= \left\| \sum_{\alpha\in\Sigma} \sigma_2(\alpha) \sum_{i\in[s]} \sum_{x\in U} \sigma_1(x) v(i, h_1(x)\oplus h_3(x)) \left[h_2(x) = \alpha \oplus i\right] w_x \right\|_p$$

We define the value function $v' \colon U \times (\Sigma \times [m/s]) \to \mathbb{R}$ by $v'(x,(i,j)) = v(i,j)\left[i \in [s]\right] w_x$. We can then write our expression as

$$\left\| \sum_{\alpha\in\Sigma} \sigma_2(\alpha) \sum_{i\in[s]} \sum_{x\in U} \sigma_1(x) v(i, h_1(x)\oplus h_3(x)) \left[h_2(x) = \alpha \oplus i\right] w_x \right\|_p$$
$$= \left\| \sum_{\alpha\in\Sigma} \sigma_2(\alpha) \sum_{x\in U} v'(x, (\alpha\oplus h_2(x), h_3(\alpha)\oplus h_1(x))) \right\|_p$$

Now we start by proving eq. (B.14). We use Lemma B.13 to get that

$$\left\| \sum_{\alpha\in\Sigma} \sigma_2(\alpha) \sum_{x\in U} v'(x, (\alpha\oplus h_2(x), h_3(\alpha)\oplus h_1(x))) \right\|_p$$
$$\leq e\sqrt{\frac{p}{\log m/s}} \left\| \sum_{\alpha\in\Sigma} \sum_{j\in[m/s]} \left(\sum_{x\in U} v'(x, (\alpha\oplus h_2(x), j\oplus h_1(x))) \right)^2 \right\|_{p/2}^{1/2}$$

Now we use Lemma B.15 to obtain that

$$\begin{aligned} \left\| \sum_{\alpha\in\Sigma} \sum_{j\in[m/s]} \left(\sum_{x\in U} v'(x, (\alpha\oplus h_2(x), j\oplus h_1(x))) \right)^2 \right\|_{p/2} & \\ \leq \gamma_p^c \sum_{x\in U} \sum_{i\in\Sigma} \sum_{j\in[m/s]} v'(x,(i,j))^2 & \\ = \gamma_p^c \sum_{x\in U} \sum_{i\in\Sigma} \sum_{j\in[m/s]} v(i,j)^2 \left[i\in[s]\right] w_x^2 & \\ = \gamma_p^c \left\|v\right\|_2^2 \left\|w\right\|_2^2 & \end{aligned} \tag{B.15}$$

This then give us that

$$\left\| \sum_{\alpha\in\Sigma} \sigma_2(\alpha) \sum_{x\in U} v'(x, (\alpha\oplus h_2(x), h_3(\alpha)\oplus h_1(x))) \right\|_p \leq e\sqrt{\gamma_p^c \frac{p}{\log m/s}} \left\|v\right\|_2 \left\|w\right\|_2$$

This finishes the proof of eq. (B.14).

Now we focus on eq. (B.13). We use Lemma B.12 to get that

$$\left\| \sum_{\alpha \in \Sigma} \sigma_2(\alpha) \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), h_3(\alpha) \oplus h_1(x))) \right\|_p \leq L \left\| \Psi_p \left(A, B \right) \right\|_p$$

where

$$A = \max_{\alpha \in \Sigma} \max_{j \in [m/s]} \left| \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), h_3(\alpha) \oplus h_1(x))) \right|$$

$$B = \sum_{\alpha \in \Sigma} \sum_{j \in [m/s]} \left(\sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right)^2$$

We apply Lemma A.15 to get that we just need to bound the two expressions

$$\left\| \max_{\alpha \in \Sigma} \max_{j \in [m/s]} \left| \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), h_3(\alpha) \oplus h_1(x))) \right| \right\|_p$$

$$\left\| \sum_{\alpha \in \Sigma} \sum_{j \in [m/s]} \left(\sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right)^2 \right\|_{p/2}$$

From eq. (B.15) we have that

$$\left\| \sum_{\alpha \in \Sigma} \sum_{j \in [m/s]} \left(\sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right)^2 \right\|_{p/2} \leq \gamma_p^c \left\| v \right\|_2^2 \left\| w \right\|_2^2$$

Let $\bar{p} = \max\{p, \log(m/s \cdot |\Sigma|\}$ and use Lemma B.14 to get that

$$\begin{aligned}
&\left\| \max_{\alpha \in \Sigma} \max_{j \in [m/s]} \left| \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right| \right\|_p \\
&\leq \left\| \max_{\alpha \in \Sigma} \max_{j \in [m/s]} \left| \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right| \right\|_{\bar{p}} \\
&\leq \left(\sum_{\alpha \in \Sigma} \sum_{j \in [m/s]} \left\| \sum_{x \in U} v'(x, (\alpha \oplus h_2(x), j \oplus h_1(x))) \right\|_{\bar{p}}^{1/\bar{p}} \right)^{1/\bar{p}} \\
&\leq e \Psi_{\bar{p}} \left(\gamma_p^c \max_{x \in U} \max_{i \in [s]} \max_{j \in [m/s]} \left| v(i,j) w_x \right|, \gamma_p^c \frac{s}{m \left| \Sigma \right|} \sum_{x \in U} \sum_{i \in [s]} \sum_{j \in [m/s]} v(i,j)^2 w_x^2 \right) \\
&\leq e \Psi_{\bar{p}} \left(\gamma_p^c \left\| v \right\|_\infty \left\| w \right\|_\infty, \gamma_p^c \frac{1}{|\Sigma|} \left\| v \right\|_2^2 \left\| w \right\|_2^2 \right)
\end{aligned}$$

We then get that

$$\begin{aligned}&\left\|\sum_{\alpha\in\Sigma}\sigma_2(\alpha)\sum_{x\in U}v'(x,(\alpha\oplus h_2(x),h_3(\alpha)\oplus h_1(x)))\right\|_p\\&\leq e\Psi_p\left(e\Psi_{\bar{p}}\left(\gamma_p^c\left\|v\right\|_\infty\left\|w\right\|_\infty,\gamma_p^c\frac{1}{|\Sigma|}\left\|v\right\|_2^2\left\|w\right\|_2^2\right),\gamma_p^c\frac{s}{m}\left\|v\right\|_2^2\left\|w\right\|_2^2\right)\end{aligned}$$

Since $s\leq\sqrt{|\Sigma|}$, the same case analysis as in the proof of [HT22, Theorem 7] give us that since

$$\begin{aligned}&\left\|\sum_{\alpha\in\Sigma}\sigma_2(\alpha)\sum_{x\in U}v'(x,(\alpha\oplus h_2(x),h_3(\alpha)\oplus h_1(x)))\right\|_p\\&\qquad\leq\Psi_p\left(\gamma_p^c\left\|v\right\|_\infty\left\|w\right\|_\infty,\gamma_p^c\frac{s}{m}\left\|v\right\|_2^2\left\|w\right\|_2^2\right)\end{aligned}$$

This finishes the proof. □

Lemma B.21. *For any vector $w\in\mathbb{R}^U$, the following concentration result holds for Mixed Tabulation hashing,*

$$\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\left[h(i,x)=j\right]w_x\right|\right\|_p\leq\sqrt{\gamma_p^c\frac{\log m}{\log m/s}}\left\|w\right\|_2$$

Here $\gamma_p=Kc\max\left\{1,\frac{p}{\log|\Sigma|}\right\}$ for a universal constant K.

Proof. We start by rewriting the expression,

$$\begin{aligned}&\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\left[h(i,x)=j\right]w_x\right|\right\|_p\\&=\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{\alpha\in\Sigma}\sigma_3(\alpha)\sum_{x\in U}\sigma_1(x)\left[h_1(x)\oplus h_3(\alpha)=j\right]\left[h_2(x)=\alpha\oplus i\right]w_x\right|\right\|_p.\end{aligned}$$

Now we fix the randomness of h_1,σ_1,h_2 and use Lemma B.13 to get that,

$$\begin{aligned}&\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{\alpha\in\Sigma}\sigma_3(\alpha)\sum_{x\in U}\sigma_1(x)\left[h_1(x)\oplus h_3(\alpha)=j\right]\left[h_2(x)=\alpha\oplus i\right]w_x\right|\right\|_p\\&\leq\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{\alpha\in\Sigma}\sigma_3(\alpha)\sum_{x\in U}\sigma_1(x)\left[h_1(x)\oplus h_3(\alpha)=j\right]\left[h_2(x)=\alpha\oplus i\right]w_x\right|\right\|_{\log m}\\&\leq e\max_{i\in[s],j\in[m/s]}\left\|\left|\sum_{\alpha\in\Sigma}\sigma_3(\alpha)\sum_{x\in U}\sigma_1(x)\left[h_1(x)\oplus h_3(\alpha)=j\right]\left[h_2(x)=\alpha\oplus i\right]w_x\right|\right\|_{\log m}\\&\leq e^2\sqrt{\frac{\log m}{\log m/s}}\max_{\substack{i\in[s]\\j\in[m/s]}}\sqrt{\sum_{\substack{\alpha\in\Sigma\\k\in[m/s]}}\left(\sum_{x\in U}\sigma_1(x)\left[h_1(x)=j\oplus k\right]\left[h_2(x)=\alpha\oplus i\right]w_x\right)^2}\end{aligned}$$

We now note that the expression

$$\sqrt{\sum_{\substack{\alpha\in\Sigma\\ k\in[m/s]}}\left(\sum_{x\in U}\sigma_1(x)\,[h_1(x)=j\oplus k]\,[h_2(x)=\alpha\oplus i]\,w_x\right)^2}$$

does not depend on i and j, so might as well look at $i=j=0$. We thus get that

$$\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\,[h(i,x)=j]\,w_x\right|\right\|_p \leq e^2\sqrt{\frac{\log m}{\log m/s}}\left\|\sum_{\alpha\in\Sigma}\sum_{k\in[m/s]}\left(\sum_{x\in U}\sigma_1(x)\,[h_1(x)=k]\,[h_2(x)=\alpha]\,w_x\right)^2\right\|_{p/2}^{1/2}$$

We can then apply Lemma B.15 and obtain

$$\left\|\max_{i\in[s],j\in[m/s]}\left|\sum_{x\in U}\sigma(i,x)\,[h(i,x)=j]\,w_x\right|\right\|_p \leq \sqrt{\gamma_p^c\frac{\log m}{\log m/s}}\,\|w\|_2$$

This finishes the proof. □

Lemma B.22. *For any vector $w\in\mathbb{R}^U$, the following concentration result holds for Mixed Tabulation hashing,*

$$\left\|\sum_{i\in[s]}\sum_{j\in[m/s]}\left(\sum_{x\in U}\sigma(i,x)\,[h(i,x)=j]\,w_x\right)^2\right\|_p \leq \gamma_p^c\max\left\{s\,\|w\|_2^2\,,\frac{p}{\log m/s}\,\|w\|_2^2\right\}$$

Here $\gamma_p = Kc\max\left\{1,\frac{p}{\log|\Sigma|}\right\}$ for a universal constant K.

Proof. We start by rewriting the expression

$$\begin{aligned}
&\left\| \sum_{i\in[s]} \sum_{j\in[m/s]} \left(\sum_{x\in U} \sigma(i,x)\,[h(i,x)=j]\,w_x \right)^2 \right\|_p \\
&\quad = \left\| \sum_{i\in[s]} \sum_{x,y\in U} \sigma(i,x)\sigma(i,y)\,[h(i,x)=h(i,y)]\,w_x w_y \right\|_p \\
&\quad \le \left\| \sum_{i\in[s]} \sum_{x,y\in U} \sigma(i,x)\sigma(i,y)\,[(h_1(x),h_2(x))=(h_1(y),h_2(y))]\,w_x w_y \right\|_p \\
&\qquad + \left\| \sum_{i\in[s]} \sum_{x,y\in U} \sigma(i,x)\sigma(i,y)\,[h(i,x)=h(i,y)]\,[h_2(x)\neq h_2(y)]\,w_x w_y \right\|_p \\
&\quad = s \left\| \sum_{z\in\Sigma^d} \sum_{j\in[m/s]} \left(\sum_{x\in U} \sigma_1(x)\,[(h_1(x),h_2(x))=(j,z)]\,w_x \right)^2 \right\|_p \\
&\qquad + \left\| \sum_{i\in[s]} \sum_{x,y\in U} \sigma(i,x)\sigma(i,y)\,[h(i,x)=h(i,y)]\,[h_2(x)\neq h_2(y)]\,w_x w_y \right\|_p
\end{aligned}$$

We will bound each of the two expressions separately. We use Lemma B.15 to get that

$$s \left\| \sum_{z\in\Sigma^d} \sum_{j\in[m/s]} \left(\sum_{x\in U} \sigma_1(x)\,[(h_1(x),h_2(x))=(j,z)]\,w_x \right)^2 \right\|_p \le \gamma_p^c s\,\|w\|_2^2$$

To bound the other expression, we use Lemma B.18 to get that,

$$\begin{aligned}
&\left\| \sum_{i\in[s]} \sum_{x,y\in U} \sigma(i,x)\sigma(i,y)\,[h(i,x)=h(i,y)]\,[h_2(x)\neq h_2(y)]\,w_x w_y \right\|_p \\
&\le \sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} 4^{c+1} \left\| \sum_{i\in[s]} \sum_{x,y\in U_{\pi,\tau}} \sigma_{I_\pi,|\tau|}(i,x)\overline{\sigma_{I_\pi,|\tau|}}(i,y)\left[h_{I_\pi,|\tau|}(i,x)=\overline{h_{I_\pi,|\tau|}}(i,y)\right] w_x w_y \right\|_p
\end{aligned}$$

For each $\pi \in P_{partial}$ and each $\tau \in P_{prefix}$ this corresponds to a Mixed Tabulation hash function with $c' \le 2c$ and $d' \le 2$. We can then use eq. (B.14) to get that

$$\begin{aligned}
&\sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} 4^{c+1} \left\| \sum_{i\in[s]} \sum_{x,y\in U_{\pi,\tau}} \sigma_{I_\pi,|\tau|}(i,x)\overline{\sigma_{I_\pi,|\tau|}}(i,y)\left[h_{I_\pi,|\tau|}(i,x)=\overline{h_{I_\pi,|\tau|}}(i,y)\right] w_x w_y \right\|_p \\
&\qquad\qquad \le \sum_{\substack{\pi\in P_{partial}\\ \tau\in P_{prefix}}} 4^{c+1}\gamma_p^c \sum_{x\in U_{\pi,\tau}} w_x^2
\end{aligned}$$

We then use eq. (B.12) to get that,

$$\sum_{\substack{\pi \in P_{partial} \\ \tau \in P_{prefix}}} 4^{c+1} \gamma_p^c \sum_{x \in U_{\pi,\tau}} w_x^2 \leq \gamma_p^c d 8^c \left\| w \right\|_2^2$$

Combing the bounds we get that

$$\begin{aligned} \left\| \sum_{i \in [s]} \sum_{j \in [m/s]} \left(\sum_{x \in U} \sigma(i,x) \left[h(i,x) = j \right] w_x \right)^2 \right\|_p &\leq \gamma_p^c + \gamma_p^c d 8^c \left\| w \right\|_2^2 \\ &\leq \gamma_p^c \max \left\{ s \left\| w \right\|_2^2, \frac{p}{\log m/s} \left\| w \right\|_2^2 \right\} \end{aligned}$$

as wanted, which finishes the proof. □

B.7 Acknowledgement

Research supported by Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

References

[AKKR+20] Anders Aamand, Jakob Bæk Tejs Knudsen, Mathias Bæk Tejs Knudsen, Peter Michael Reichstein Rasmussen, and Mikkel Thorup. "Fast Hashing with Strong Concentration Bounds". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2020. Chicago, IL, USA: Association for Computing Machinery, 2020, pp. 1265–1278. ISBN: 9781450369794.

[AT19] Anders Aamand and Mikkel Thorup. "Non-empty Bins with Simple Tabulation Hashing". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*. Ed. by Timothy M. Chan. SIAM, 2019, pp. 2498–2512.

[Ach03] Dimitris Achlioptas. "Database-friendly random projections: Johnson-Lindenstrauss with binary coins". In: *Journal of Computer and System Sciences* 66.4 (2003). Special Issue on PODS 2001, pp. 671–687. ISSN: 0022-0000.

[AC09] Nir Ailon and Bernard Chazelle. "The Fast Johnson–Lindenstrauss Transform and Approximate Nearest Neighbors". In: *SIAM Journal on Computing* 39.1 (2009), pp. 302–322. eprint: https://doi.org/10.1137/060673096.

[AL08] Nir Ailon and Edo Liberty. "Fast dimension reduction using Rademacher series on dual BCH codes". In: *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*. Ed. by Shang-Hua Teng. SIAM, 2008, pp. 1–9.

[AK17] Noga Alon and Bo'az Klartag. "Optimal Compression of Approximate Inner Products and Dimension Reduction". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. Oct. 2017, pp. 639–650.

[BK21] Stefan Bamberger and Felix Krahmer. "Optimal Fast Johnson-Lindenstrauss Embeddings for Large Data Sets". In: *Sampling Theory, Signal Processing, and Data Analysis* 19 (June 2021).

[BOR10] Vladimir Braverman, Rafail Ostrovsky, and Yuval Rabani. "Rademacher Chaos, Random Eulerian Graphs and The Sparse Johnson-Lindenstrauss Transform". In: *CoRR* abs/1011.2590 (2010). arXiv: 1011.2590.

[CCF04] Moses Charikar, Kevin Chen, and Martin Farach-Colton. "Finding frequent items in data streams". In: *Theoretical Computer Science* 312.1 (2004). Automata, Languages and Programming, pp. 3–15. ISSN: 0304-3975.

[CJN18] Michael B. Cohen, T. S. Jayram, and Jelani Nelson. "Simple Analyses of the Sparse Johnson-Lindenstrauss Transform". In: *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*. Ed. by Raimund Seidel. Vol. 61. OASIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 15:1–15:9.

[DKRT15] S. Dahlgaard, M. B. T. Knudsen, E. Rotenberg, and M. Thorup. "Hashing for Statistics over K-Partitions". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. 2015, pp. 1292–1310.

[DKT17] Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. "Practical Hash Functions for Similarity Estimation and Dimensionality Reduction". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6618–6628. ISBN: 978-1-5108-6096-4.

[DT14] Søren Dahlgaard and Mikkel Thorup. "Approximately Minwise Independence with Twisted Tabulation". In: Mar. 2014. ISBN: 978-3-319-08403-9.

[DKS10] Anirban Dasgupta, Ravi Kumar, and Tamás Sarlos. "A Sparse Johnson: Lindenstrauss Transform". In: STOC '10. Cambridge, Massachusetts, USA: Association for Computing Machinery, 2010, pp. 341–350. ISBN: 9781450300506.

[DGCN+09] Thong T. Do, Lu Gan, Yi Chen, Nam Nguyen, and Trac D. Tran. "Fast and efficient dimensionality reduction using Structurally Random Matrices". In: *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2009, pp. 1821–1824.

[FHL22] Ora Nova Fandina, Mikael Møller Høgsgaard, and Kasper Green Larsen. *Barriers for Faster Dimensionality Reduction*. 2022.

[FKL18] Casper Freksen, Lior Kamma, and Kasper Green Larsen. "Fully Understanding the Hashing Trick". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Montréal, Canada: Curran Associates Inc., 2018, pp. 5394–5404.

[FL20] Casper Benjamin Freksen and Kasper Green Larsen. "On Using Toeplitz and Circulant Matrices for Johnson-Lindenstrauss Transforms". In: *Algorithmica* 82.2 (2020), pp. 338–354.

[HV11] Aicke Hinrichs and Jan Vybíral. "Johnson-Lindenstrauss lemma for circulant matrices". In: *Random Structures & Algorithms* 39.3 (2011), pp. 391–398. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.20360.

[HT22] Jakob Bæk Tejs Houen and Mikkel Thorup. "Understanding the Moments of Tabulation Hashing via Chaoses". In: *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*. Ed. by Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff. Vol. 229. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 74:1–74:19.

[Jag19] Meena Jagadeesan. "Understanding Sparse JL for Feature Hashing". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. NeurIPS'19. Red Hook, NY, USA: Curran Associates Inc., 2019.

[JPSS+22] Vishesh Jain, Natesh S. Pillai, Ashwin Sah, Mehtaab Sawhney, and Aaron Smith. "Fast and memory-optimal dimension reduction using Kac's walk". In: *The Annals of Applied Probability* 32.5 (2022), pp. 4038–4064.

[JW13] T. S. Jayram and David P. Woodruff. "Optimal Bounds for Johnson-Lindenstrauss Transforms and Streaming Problems with Subconstant Error". In: *ACM Trans. Algorithms* 9.3 (June 2013). ISSN: 1549-6325.

[JL84] William Johnson and Joram Lindenstrauss. "Extensions of Lipschitz maps into a Hilbert space". In: *Contemporary Mathematics* 26 (Jan. 1984), pp. 189–206.

[Kac56] Mark Kac. "Foundations of kinetic theory". In: *Proceedings of The third Berkeley symposium on mathematical statistics and probability*. Vol. 3. 1956, pp. 171–197.

[KMN11] Daniel Kane, Raghu Meka, and Jelani Nelson. "Almost Optimal Explicit Johnson-Lindenstrauss Families". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Ed. by Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 628–639. ISBN: 978-3-642-22935-0.

[KN10] Daniel M. Kane and Jelani Nelson. *A Derandomized Sparse Johnson-Lindenstrauss Transform*. 2010.

[KN14] Daniel M. Kane and Jelani Nelson. "Sparser Johnson-Lindenstrauss Transforms". In: *J. ACM* 61.1 (Jan. 2014). ISSN: 0004-5411.

[KW11] Felix Krahmer and Rachel Ward. "New and Improved Johnson–Lindenstrauss Embeddings via the Restricted Isometry Property". In: *SIAM Journal on Mathematical Analysis* 43.3 (2011), pp. 1269–1281. eprint: https://doi.org/10.1137/100810447.

[LN17] Kasper Green Larsen and Jelani Nelson. "Optimality of the Johnson-Lindenstrauss Lemma". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017, pp. 633–638.

[NN13] Jelani Nelson and Huy L. NguyÅn. "Sparsity Lower Bounds for Dimensionality Reducing Maps". In: *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*. STOC '13. Palo Alto, California, USA: Association for Computing Machinery, 2013, pp. 101–110. ISBN: 9781450320290.

[PT13] Mihai Patrascu and Mikkel Thorup. "Twisted Tabulation Hashing". In: *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*. Ed. by Sanjeev Khanna. SIAM, 2013, pp. 209–228.

[PT12] Mihai Pătraşcu and Mikkel Thorup. "The Power of Simple Tabulation Hashing". In: *J. ACM* 59.3 (June 2012). ISSN: 0004-5411.

[Sie04] Alan Siegel. "On Universal Classes of Extremely Random Constant-Time Hash Functions". In: 33.3 (2004). Announced at FOCS'89, pp. 505–543.

[Tho13] Mikkel Thorup. "Simple Tabulation, Fast Expanders, Double Tabulation, and High Independence". In: *54th Annual Symposium on Foundations of Computer Science (FOCS)*. 2013, pp. 90–99.

[TZ12] Mikkel Thorup and Yin Zhang. "Tabulation-Based 5-Independent Hashing with Applications to Linear Probing and Second Moment Estimation". In: *SIAM Journal on Computing* 41.2 (2012), pp. 293–331. eprint: https://doi.org/10.1137/100800774.

[Ver18] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

[Vyb11] Jan Vybíral. "A variant of the Johnson–Lindenstrauss lemma for circulant matrices". In: *Journal of Functional Analysis* 260.4 (2011), pp. 1096–1105. ISSN: 0022-1236.

[WDLS+09] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. "Feature Hashing for Large Scale Multitask Learning". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 1113–1120. ISBN: 9781605585161.

[Zob70] Albert Lindsey Zobrist. *A New Hashing Method with Application for Game Playing*. Tech. rep. 88. Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 1970.

Appendix C

Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search

Subsets and Supermajorities: Optimal Hashing-based Set Similarity Search

Thomas D. Ahle
BARC, IT University of Copenhagen
thdy@itu.dk

Jakob B. T. Knudsen
BARC, University of Copenhagen
jakn@di.ku.dk

Dansk resumé

We formulate and optimally solve a new generalized Set Similarity Search problem, which assumes the size of the database and query sets are known in advance. By creating polylog copies of our data-structure, we optimally solve any symmetric Approximate Set Similarity Search problem, including approximate versions of Subset Search, Maximum Inner Product Search (MIPS), Jaccard Similarity Search and Partial Match.

Our algorithm can be seen as a natural generalization of previous work on Set as well as Euclidean Similarity Search, but conceptually it differs by optimally exploiting the information present in the sets as well as their complements, and doing so asymmetrically between queries and stored sets. Doing so we improve upon the best previous work: MinHash [J. Discrete Algorithms 1998], SimHash [STOC 2002], Spherical LSF [SODA 2016, 2017] and Chosen Path [STOC 2017] by as much as a factor $n^{0.14}$ in both time and space; or in the near-constant time regime, in space, by an arbitrarily large polynomial factor.

Turning the geometric concept, based on Boolean supermajority functions, into a practical algorithm requires ideas from branching random walks on $\mathbb{Z}^2$, for which we give the first non-asymptotic near tight analysis.

Our lower bounds follow from new hypercontractive arguments, which can be seen as characterizing the exact family of similarity search problems for which supermajorities are optimal. The optimality holds for among all hashing based data structures in the random setting, and by reductions, for 1 cell and 2 cell probe data structures. As a side effect, we obtain new hypercontractive bounds on the directed noise operator $T_\rho^{p_1 \to p_2}$.

Contents

C.1 Introduction

Set Similarity Search (SSS) is the problem of indexing sets (or sparse boolean data) to allow fast retrieval of sets, similar under a given similarity measure. The sets may represent one-hot encodings of categorical data, "bag of words" representations of documents, or "visual/neural bag of words" models, such as the Scale-invariant feature transform (SIFT), that have been discretized. The applications are ubiquitous across Computer Science, touching everything from recommendation systems to gene sequences comparison. See [CCT10; JZYY+18] for recent surveys of methods and applications.

Set similarity measures are any function, s that takes two sets and return a value in $[0, 1]$. Unfortunately, most variants of Set Similarity Search, such as Partial Match, are hard to solve assuming popular conjectures around the Orthogonal Vectors Problem [Wil05; APRS16; ARW17; CW19], which roughly implies that the best possible algorithm is to not build an index, and "just brute force" scan through all the data, on every query. A way to get around this is to study Approximate SSS: Given a query, q, for which the most similar set y has similarity$(q, y) \geq s_1$, we are allowed to return any set y'

with similarity$(q, y') > s_1$, where $s_2 < s_1$. In practice, even the best *exact* algorithms for similarity search use such an (s_1, s_2)-approximate[1] solution as a subroutine [CPT18].

Euclidean Similarity Search, where the data is vectors $x \in \mathbb{R}^d$ and the measure of similarity is "Cosine", has recently been solved optimally — at least in the model of hashing based data structures [AR15; ALRW17a]. Meanwhile, the problem on sets has proven much less tractable. This is despite that the first solutions date back to the seminal MinHash algorithm (a.k.a. min-wise hashing), introduced by Broder et al. [BGMZ97; Bro97] in 1997 and by now boasting thousands of citations. In 2014 MinHash was shown to be near-optimal for set intersection *estimation* [PSW14], but in a surprising, recent development, it was shown not to be optimal for similarity *search* [CP17]. The question thus remained: What *is* the optimal algorithm for Set Similarity Search?

The question is made harder by the fact that previous algorithms study the problem under different similarity measures, such as Jaccard, Cosine, or Braun-Blanquet similarity. The only thing those measures have in common is that they can be defined as a function f of the sets sizes, the universe size, and the intersection size. In other words, similarity$(q, y) = f(|q|, |y|, |q \cap y|, |U|)$ where $|U|$ is the size of the universe from which the sets are taken. In fact, any symmetric measure of similarity for sets must be defined by those four quantities.

Hence, to fully solve Set Similarity Search, we avoid specifying a particular similarity measure, and instead define the problem solely from those four parameters. This generalized problem is what we solve optimally in this paper, for all values of the four parameters:

Definition C.1 (The (w_q, w_u, w_1, w_2)-GapSS problem)**.** Given some universe U and a collection $Y \subseteq \binom{U}{w_u|U|}$ of $|Y| = n$ sets of size $w_u|U|$, build a data structure that for any query set $q \in \binom{U}{w_q|U|}$: either returns $y' \in Y$ with $|y' \cap q| > w_2|U|$; or determines that there is no $y \in Y$ with $|y \cap q| \geq w_1|U|$.

For the problem to make sense, we assume that $w_q|U|$ and $w_u|U|$ are integers, that $w_q, w_u \in [0, 1]$, and that $0 < w_2 < w_1 \leq \min\{w_q, w_u\}$. Note that $|U|$ may be very large, and as a consequence the values w_q, w_u, w_1, w_2 may all be very small.

At first sight, the problem may seem easier than the version where the sizes of sets may vary. However, the point is that making polylog(n) data-structures for sets and queries of progressively bigger sizes,[2] immediately yields data structures for the original problem. Similarly, any algorithm assuming a specific set similarity measure also yields an algorithm for (w_q, w_u, w_1, w_2)-GapSS, so our lower bounds also hold for all previously studied SSS problems.

Example 1 As an example, assume we want to solve the Subset Search Problem, in which we, given a query q, want to find a set y in the database, such that $y \subseteq q$. If we

[1] By classical reductions [HIM12] we can assume s_1 is known in advance.

[2] For details, see [CP17] Section 5. A similar reduction, called "norm ranging", was recently shown at NeurIPS to give state of the art results for Maximum Inner Product Search in $\mathbb{R}^d$ [YLDC+18], suggesting it is very practical.

allow a two-approximate solution, GapSS includes this problem by setting $w_1 = w_u$ and $w_2 = w_1/2$: The overlap between the sets must equal the size of the stored sets; and we are guaranteed to return a y' such that at least $|q \cap y'| \geq |y|/2$.

Example 2 In the (j_1, j_2)-Jaccard Similarity Search Problem, given a query, q, we must find y such that the Jaccard Similarity $|q \cap y|/|q \cup y| > j_2$ given that a y' exists with similarity at least j_1. After partitioning the sets by size, we can solve the problem using GapSS by setting $w_1 = \frac{j_1(w_q+w_u)}{1+j_1}$ and $w_2 = \frac{j_2(w_q+w_u)}{1+j_2}$. The same reduction works for any other similarity measure with polylog(n) overhead.

The version of this problem where $w_2 = w_q w_u$ is similar to what is in the literature called *"the random instance"* [Pan06; Laa15; ALRW17b]. To see why, consider generating $n-1$ sets independently at random with size $w_u|U|$, and a "planted" pair, (q, y), with size respectively $w_q|U|$ and $w_u|U|$ and with intersection $|q \cap y| = w_1|U|$. Insert the size $w_u|U|$ sets into the database and query with q. Since q is independent from the $n-1$ original sets, its intersection with those is strongly concentrated around the expectation $w_q w_u|U|$. Thus, if we parametrize GapSS with $w_2 = w_q w_u + o(1)$, the query for q is guaranteed to return the planted set y.

There is a tradition in the Similarity Search literature for studying such this independent case, in part because *it is expected that one can always reduce to the random instance*, for example using the techniques of "data-dependent hashing" [AINR14; AR15]. However, for such a reduction to make sense, we would first need an optimal "data-independent" algorithm for the $w_2 = w_q w_u$ case, which is what we provide in this paper. We discuss this further in the Related Work section.

For generality we still define the problem for all $w_2 \in (0, w_1)$, our upper bound holds in this general setting and so does the lower bound Theorem C.4.

We give our new results in Appendix C.1.2 and our new lower bounds in Appendix C.1.3, but first we would like to sketch the algorithm and some probabilistic tools used in the theorem statement.

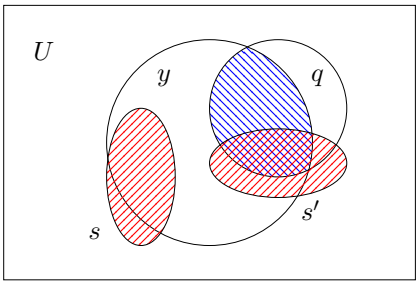
C.1.1 Supermajorities

In Social Choice Theory a supermajority is when a fraction strictly greater than 1/2 of people agree about something.[3] In the analysis of Boolean functions a t-supermajority function $f : \{0,1\}^n \to \{0,1\}$ can be defined as 1, if a fraction $\geq t$ of its arguments are 1, and 0 otherwise. We will sometimes use the same word for the requirement that a fraction $\leq t$ of the arguments are 1.[4]

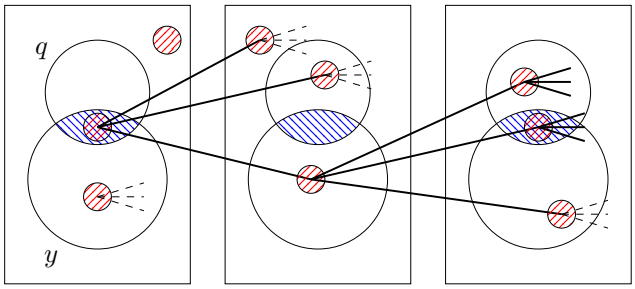
The main conceptual point of our algorithm is the realization that an optimal algorithm for Set Similarity Search must take advantage of the information present in the given sets,

[3] "America was founded on majority rule, not supermajority rule. Somehow, over the years, this has morphed into supermajority rule, and that changes things." – Kent Conrad.

[4] It turns out that defining everything in terms of having a fraction $t \pm o(1)$ of 1's is also sufficient. This is similar to Dubiner [Dub10].



(a) Two cohorts, y and q with a large intersection (blue). The first representative set, s, favours y, while the second, s', favours both y and q.



(b) Branching random walk run on two cohorts q and y. The bold lines illustrate paths considered by sets, while the dashed lines adorn paths only considered by only one of x or y. Here q has a higher threshold ($t_q = 2/3$) than y ($t_u = 1/2$), so q only considers paths starting with two favourable representatives.

Figure C.1: The representative sets, coloured in red, are scattered in the universe to provide an efficient space partition for the data.

as well as that present in their complement. A similar idea was leveraged by Cohen et al. [CK09] for Set Similarity *Estimation*, and we show in Appendix C.4.2 that the classical MinHash algorithm can be seen as an average of functions that pull varying amounts of information from the sets and their complements. In this paper, we show that there is a better way of combining this information and that doing so results in an optimal hashing based data structure for the entire parameter space of random instance GapSS.

This way of combining this information is by supermajority functions. While on the surface they will seem similar to the threshold methods applied for time/space tradeoffs in Spherical LSF [ALRW17a], our use of them is very different. Where [ALRW17a] corresponds to using small $t = 1/2 + o(1)$ thresholds (essentially simple majorities) our t may be as large as 1 (corresponding to the AND function) or as small as 0 (the NOT AND function). This way they are a sense as much a requirement on the complement as it is on the sets themselves.

The algorithm (idealized): While our data structure is technically a tree with a carefully designed pruning rule, the basic concept is very simple.

We start by sampling a large number of "representative sets" $R \subseteq \binom{U}{k}$. Here roughly $|R| \approx n^{\log n}$ and $k \approx \log n$. Given family $Y \subseteq \binom{U}{w_u|U|}$ of sets to store, which we call "cohorts", we say that $r \in R$ "t-favours" the cohort y if $|y \cap r|/|r| \geq t$. Representing sets as vectors in $\{0,1\}^d$, this is equivalent to saying $f_t(r \cap y) = 1$, where f_t is the t-supermajority function. (If t is less than w_u, the expected size of the overlap, we instead require $|y \cap r|/|r| \leq t$.)

Given the parameters $t_q, t_u \in [0,1]$, the data-structure is a map from elements of R to the cohorts they t_u-favour. When given a query $q \in \binom{U}{w_q|U|}$, (a $w_q|U|$ sized cohort), we compare it against all cohorts y favoured by representatives $r \in R$ which t_q-favour q (that is $|q \cap r|/|r| \geq t_q$). This set $R_{t_q}(q)$ is much smaller than $|R|$ (we will have $|R_{t_q}(q)| \approx n^{\varepsilon}$

and $E[|R_{t_u}(y) \cap R_{t_q}(q)|] \approx n^{\varepsilon-1}$), so the filtering procedure greatly reduces the number of cohorts we need to compare to the query from n to n^{ε} (where $\varepsilon = \rho_q < 1$ is defined later.)

The intuition is that while it is quite unlikely for a representative to favour a given cohort, and it is *very* unlikely for it to favour two given cohorts (q and y). So if it does, the two cohorts probably have a substantial overlap. Figure C.1a has a simple illustration of this principle.

In order to fully understand supermajorities, we want to understand the probability that a representative set is simultaneously in favour of two distinct cohorts given their overlap and representative sizes. This paragraph is a bit technical and may be skipped at first read. Chernoff bounds in $\mathbb{R}$ are a common tool in the community, and for iid. $X_i \sim \text{Bernoulli}(p) \in \{0,1\}$ the sharpest form (with a matching lower bound) is $\Pr[\sum X_i \geq tn] \leq \exp(-n\,\mathrm{d}(t \parallel p))$,[5] which uses the binary KL-Divergence $\mathrm{d}(t \parallel p) = t\log\frac{t}{p} + (1-t)\log\frac{1-t}{1-p}$. The Chernoff bound for $\mathbb{R}^2$ is less common, but likewise has a tight description in terms of the KL-Divergence between two discrete distributions: $\mathrm{D}(P \parallel Q) = \sum_{\omega\in\Omega} P(\omega)\log\frac{P(\omega)}{Q(\omega)}$ (summing over the possible events). In our case, we represent the four events that can happen as we sample an element of U as a vector $X_i \in \{0,1\}^2$. Here $X_i = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ means the ith element hit both cohorts, $X_i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ means it hit only the first and so on. We represent the distribution of each X_i as a matrix $P = \left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{smallmatrix}\right]$, and say $X_i \sim \text{Bernoulli}(P)$ iid. such that $\Pr[X_i = \left[\begin{smallmatrix}1-j\\1-k\end{smallmatrix}\right]] = P_{j,k}$. Then $\Pr[\sum X_i \geq \left[\begin{smallmatrix}t_q\\t_u\end{smallmatrix}\right]n] \approx \exp(-n\,\mathrm{D}(T \parallel P))$ where $T = \left[\begin{smallmatrix} t_1 & t_q - t_1 \\ t_u - t_1 & 1 - t_q - t_u + t_1 \end{smallmatrix}\right]$ and $t_1 \in [0, \min\{t_u, t_q\}]$ minimizes $\mathrm{D}(T \parallel P)$. (Here the notation $\left[\begin{smallmatrix}x\\y\end{smallmatrix}\right] \geq \left[\begin{smallmatrix}t_u\\t_q\end{smallmatrix}\right]$ means $x \geq t_u \wedge y \geq t_q$.)

The optimality of Supermajorities for Set Similarity Search is shown using a certain correspondence we show between the Information Theoretical quantities described above, and the hypercontractive inequalities that have been central in all previous lower bounds for similarity search.

These bounds above would immediately allow a cell probe version of our upper bound Theorem C.2, e.g. a query would require $n^{\frac{\mathrm{D}(T_1\parallel P_1) - \mathrm{d}(t_q\parallel w_q)}{\mathrm{D}(T_2\parallel P_2) - \mathrm{d}(t_q\parallel w_q)}}$ probes, where $P_i = \left[\begin{smallmatrix} w_i & w_q - w_1 \\ w_u - w_i & 1 - w_q - w_u + w_i \end{smallmatrix}\right]$ and T_i defined accordingly. The algorithmic challenge is that for optimal performance, $|R|$ must be in the order of $\Omega(n^{\log n})$, and so checking which representatives favour a given cohort takes super polynomial time!

The classical approach to designing an oracle to efficiently yield all such representatives, , is a product-code or "tensoring trick". The idea, (used by [Chr17; BDGL16]), is to choose a smaller $k' \approx \sqrt{k}$, make k/k' different R'_i sets of size $n^{\sqrt{\log n}}$ and take R as the product $R'_1 \times \cdots \times R'_{k/k'}$. As each R' can now be decoded in $n^{o(1)}$ time, so can R. This approach, however, in the case of Supermajorities, has a big drawback: Since $t_q k'$ and $t_u k'$ must be integers, t_q and t_u have to be rounded and thus distorted by a factor $1 + 1/k'$. Eventually, this ends up costing us a factor $w_1^{-k/k'}$ which can be much larger than n. For this reason, we need a decoding algorithm that allows us to use supermajorities with as large a k as possible!

[5] A special case of Hoeffding's inequality is obtained by $\mathrm{d}(p+\varepsilon \parallel p) \geq 2\varepsilon^2$, Pinsker's inequality.

We instead augment the above representative sampling procedure as follows: Instead of independent sampling sets, we (implicitly) sample a large, random height k tree, with nodes being elements from the universe. The representative sets are taken to be each path from the root to a leaf. Hence, some sets in R share a common prefix, but mostly they are still independent. We then add the extra constraint that *each of the prefixes of a representative has to be in favour of a cohort*, rather than only having this requirement on the final set. This is the key to making the tree useful: Now given a cohort, we walk down the tree, pruning any branches that do not consistently favour a supermajority of the cohort. Figure C.1b has a simple illustration of this algorithm and Algorithm 1 has a pseudo-code implementation. This pruning procedure can be shown to imply that we only spend time on representative sets that end up being in favour of our cohort, while only weakening the geometric properties of the idealized algorithm negligibly.

While conceptually simple and easy to implement (modulo a few tricks to prevent dependency on the size of the universe, $|U|$), the pruning rule introduces dependencies that are quite tricky to analyze sufficiently tight. The way to handle this will be to consider the tree as a "branching random walk" over $\mathbb{Z}_+^2$ where the value represents the size of the representative's intersection with the query and a given set respectively. The paths in the random walk at step i must be in the quadrant $[t_q i, i] \times [t_u i, i]$ while only increasing with a bias of $\left[\begin{smallmatrix} w_q \\ w_u \end{smallmatrix}\right]$ per step. The branching factor is carefully tuned to just the right number of paths survive to the end.

The "history" aspect of the pruning is a very important property of our algorithm, and is where it conceptually differs from all previous work.

Previous Locality Sensitive Filtering, LSF, algorithms [CP17; ALRW17b] can be seen as trees with pruning, but their pruning is on the individual node level, rather than on the entire path. This makes a big difference in which space partitions can be represented, since pruning on node level ends up representing the intersection of simple partitions, which can never represent Supermajorities in an efficient way. In [BDGL16] a similar idea was discussed heuristically for Gaussian filters, but ultimately tensoring was sufficient for their needs, and the idea was never analyzed.

C.1.2 Upper Bounds

As discussed, the performance of our algorithm is described in terms of KL-divergences. To ease understanding, we give a number of special cases, in which the general bound simplifies. The bounds in this section assume w_q, w_u, w_1, w_2 are constants. See Appendix C.2.1 for a version without this assumption.

Theorem C.2 (Simple Upper Bound)**.** *For any choice of constants $w_q, w_u \geq w_1 \geq w_2 \geq 0$ and $1 \geq t_q, t_u \geq 0$ we can solve the (w_q, w_u, w_1, w_2)-GapSS problem over universe U with query time $\tilde{O}(n^{\rho_q} + w_q|U|) + n^{o(1)}$ and auxiliary space usage $\tilde{O}(n^{1+\rho_u})$, where*

$$\rho_q = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)}, \quad \rho_u = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)}.$$

and T_1, T_2 are distributions with expectation $\left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]$ minimizing respectively $\mathrm{D}(T_1 \parallel P_1)$ and $\mathrm{D}(T_2 \parallel P_2)$, as described in Appendix C.1.1.

The two bounds differ only in the $\mathrm{d}(t_q \parallel w_q)$ and $\mathrm{d}(t_u \parallel w_u)$ terms in the numerator. The thresholds t_q and t_u can be chosen freely in $[0,1]^2$. Varying them compared to each other allows a full space/time trade-off with $\rho_q = 0$ in one end and $\rho_u = 0$ (and $\rho_q < 1$) in the other. Note that for a given GapSS instance, there are many (t_q, t_u) which are not optimal anywhere on the space/time trade-off. Using Lagrange's condition $\nabla\rho_q = \lambda\nabla\rho_u$ one gets a simple equation that all optimal (t_q, t_u) trade-offs must satisfy. As we will discuss later, it seems difficult to prove that a solution to this equation is unique, but in practice, it is easy to solve and provides an efficient way to optimize ρ_q given a space budget $n^{1+\rho_u}$. Figure C.2 and Figure C.3 provides some additional intuition for how the ρ values behave for different settings of GapSS.

Regarding the other terms in the theorem, we note that the $\tilde{O}$ hides only $\log n$ factors, and the additive $n^{o(1)}$ term grows as $e^{O(\sqrt{\log n \log\log n})}$, which is negligible unless $\rho_q = 0$. We also note that there is no dependence on $|U|$, other than the need to store the original dataset and the additive $w_q|U|$, which is just the time it takes to receive the query. The main difference between this theorem and the full version is that the full theorem does not assume the parameters (w_q, w_u, w_1, w_2) are constants but consider them potentially very small. In this more realistic scenario, it becomes very important to limit the dependency on factors like w_1^{-1}, which is what guides a lot of our algorithmic decisions.

Example 1: Near balanced ρ values. As noted, many pairs (t_q, t_u) are not optimal on the trade-off, in that one can reduce one or both of ρ_q, ρ_u by changing them. The pairs that are optimal are not always simple to express, so it is interesting to study those that are. One such particularly simple choice on the Lagrangian is $t_q = 1 - w_u$ and $t_u = 1 - w_q$.[6] This point is special because the values of t_q and t_u depend only on w_u and w_q, while in general they will also depend on w_1 and w_2. In this setting we have $T_i = \left[\begin{smallmatrix} 1-w_q-w_u+w_i & w_u-w_i \\ w_u-w_i & w_i \end{smallmatrix}\right]$, which can be plugged into Theorem C.2.

In the case $w_q = w_u = w$ we get the balanced ρ values $\rho_q = \rho_u = \log(\frac{w_1}{w}\frac{1-w}{1-2w+w_1})/\log(\frac{w_2}{w}\frac{1-w}{1-2w+w_2})$ in which case it is simple to compare with Chosen Path's ρ value of $\log(\frac{w_1}{w})/\log(\frac{w_2}{w})$. Chosen Path on balanced sets was shown in [CP17] to be optimal for w, w_1, w_2 small enough, and we see that Supermajorities do indeed recover this value for that range.

We give a separate lower bound in Appendix C.3.4 showing that this value is in fact optimal when $w_2 = w_q w_u$.

Example 2: Subset/superset queries. If $w_1 = \min\{w_u, w_q\}$ and $w_2 = w_u w_q$ we can take $t_q = \frac{-\alpha}{w_q - w_u} + \frac{w_q(1-w_u)}{w_q - w_u}$ and $t_u = \frac{w_u(1-w_u)w_q(1-w_q)}{w_q-w_u}\alpha^{-1} - \frac{w_u(1-w_q)}{w_q-w_u}$ for any $\alpha \in$

[6] To make matters complicated, this *is* a simple choice *and* on the Lagrangian, but that doesn't prove another point on the Lagrangian won't reduce both ρ_q and ρ_u and thus be better. That we have a matching lower bound for the algorithm doesn't help, since it only matches the upper bound for (t_q, t_u) minimal in Theorem C.2. In the case $w_q = w_u$ we can, however, prove that this t_q, t_u pair is optimal.

$[w_1 - w_q w_u, \max\{w_u, w_q\} - w_q w_u]$. Theorem C.2 then gives data structures with

$$\rho_q = \frac{t_q \log \frac{1-t_u}{1-w_u} - t_u \log \frac{1-t_q}{1-w_q}}{\mathrm{d}(t_u \parallel w_u)} \qquad \rho_u = \frac{(1-t_u)\log \frac{t_q}{w_q} - (1-t_q)\log \frac{t_u}{w_u}}{\mathrm{d}(t_u \parallel w_u)} \quad \text{if } w_1 = w_u,$$

$$\rho_q = \frac{(1-t_q)\log \frac{t_u}{w_u} - (1-t_u)\log \frac{t_q}{w_q}}{\mathrm{d}(t_u \parallel w_u)} \qquad \rho_u = \frac{t_u \log \frac{1-t_q}{1-w_q} - t_q \log \frac{1-t_u}{1-w_u}}{\mathrm{d}(t_u \parallel w_u)} \quad \text{if } w_1 = w_q.$$

This represents one of the cases where we can solve the Lagrangian equation to get a complete characterization of the t_q, t_u values that give the optimal trade-offs. Note that when $w_1 = w_u$ or $w_1 = w_q$, the P matrix as used in the theorem has 0's in it. The only way the KL-divergence $\mathrm{D}(T \parallel P)$ can then be finite is by having the corresponding elements of T be 0 and use the fact that $0 \log \frac{0}{q}$ is defined to be 0 in this context.

Example 3: Linear space/constant time. Setting t_1 in $T_1 = \begin{bmatrix} t_1 & t_q - t_1 \\ t_u - t_1 & 1 - t_q - t_u + t_1 \end{bmatrix}$ such that either $\frac{t_1}{w_1} = \frac{t_q - t_1}{w_q - w_1}$ or $\frac{t_1}{w_1} = \frac{t_u - t_1}{w_u - w_1}$ we get respectively $\mathrm{D}(T_1 \parallel P_1) = \mathrm{d}(t_q \parallel w_q)$ or $\mathrm{D}(T_1 \parallel P_1) = \mathrm{d}(t_u \parallel w_u)$. Theorem C.2 then yields algorithms with either $\rho_q = 0$ or $\rho_u = 0$ corresponding to either a data structure with $\approx e^{\tilde{O}(\sqrt{\log n})}$ query time, or with $\tilde{O}(n)$ auxiliary space. Like [ALRW17a] we have $\rho_q < 1$ for any parameter choice, even when $\rho_u = 0$. For very small w_q and $w_u < \exp(-\sqrt{\log n})$ there are some extra concerns which are discussed after the main theorem.

C.1.3 Lower Bounds

Results on approximate similarity search are usually phrased in terms of two quantities: (1) The "query exponent" $\rho_q \in [0, 1]$ which determines the query time by bounding it by $O(n^{\rho_q})$; (2) The "update exponent" $\rho_u \in [0, 1]$ which determines the time required to update the data structure when a point is inserted or deleted in Y and is given by $O(n^{\rho_u})$. The update exponent also bounds the space usage as $O(n^{1+\rho_u})$. Given parameters (w_q, w_u, w_1, w_2), the important question is for which pairs of (ρ_q, ρ_u) there exists data structures. E.g. given a space budget imposed by ρ_u, we ask how small can one make ρ_q?

Since the first lower bounds on Locality Sensitive Hashing [MNP06], lower bounds for approximate near neighbours have split into two kinds: (1) Cell probe lower bounds [PTW08; PTW10; ALRW17a] and (2) Lower bounds in restricted models [ODo14; AR16; ALRW17a; CP17]. The most general such model for data-independent algorithms was formulated by [ALRW17a] and defines a type of data structure called "list of points":

Definition C.3 (List-of-points)**.** Given some universes, Q, U, a similarity measure $S : Q \times U \to [0, 1]$ and two thresholds $1 \geq s_1 > s_2 \geq 0$,

1. We fix (possibly random) sets $A_i \subseteq \{-1, 1\}^d$, for $1 \leq i \leq m$; and with each possible query point $q \in \{-1, 1\}^d$, we associate a (random) set of indices $I(q) \subseteq [m]$;

2. For a given dataset P, we maintain m lists of points $L_1, L_2, \ldots, L_m$, where $L_i = P \cap A_i$.

3. On query q, we scan through each list L_i for $i \in I(q)$ and check whether there exists some $p \in L_i$ with $S(q,p) \geq s_2$. If it exists, return p.

The data structure succeeds, for a given $q \in Q, p \in P$ with $S(q,p) \geq s_1$, if there exists $i \in I(q)$ such that $p \in L_i$. The total space is defined by $S = m + \sum_{i \in [m]} |L_i|$ and the query time by $T = |I(q)| + \sum_{i \in I(q)} |L_i|$.

The List-of-points model contains all known Similarity Search data structures, except for the so-called "data-dependent algorithms". It is however conjectured [ALRW17b] that data-dependency does not help on random instances (recall this corresponds to $w_2 = w_q w_u$), which is the setting of Theorem C.5.

We show two main lower bounds: (1) That requires $w_q = w_u$ and $\rho_q = \rho_u$ and (2) That requires $w_2 = w_q w_u$. The second type is tight everywhere, but quite technical. The first type meanwhile is quite simple to state, informally:

Theorem C.4. *If $w_q = w_u = w$ and $\rho_u = \rho_q = \rho$, any data-independent LSF data structure must use space $n^{1+\rho}$ and have query time n^{ρ} where $\rho \geq \log(\frac{w_1 - w^2}{w(1-w)}) / \log(\frac{w_2 - w^2}{w(1-w)})$.*

The LSF Model defined in [BDGL16; CP17] generalizes [MNP06; OWZ14], but is slightly stronger than list-of-points. It is most likely that they are equivalent, so we defer its definition till Definition C.28. We will just note that previous bounds of this type [OWZ14; CP17] were only asymptotic, whereas our lower bound holds over the entire range of $0 < w_2 < w_1 < w < 1$. By comparison with $\rho = \log(\frac{w_1(1-w)}{w(1-2w+w_1)}) / \log(\frac{w_2(1-w)}{w(1-2w+w_2)})$ from Example 1 in the Upper Bounds section, we see that the lower bound is sharp when $w, w_1, w_2 \to 0$[7] and also for $w_1 \to w$, since $w(1-2w+w_1) = w(1-w) - w(w-w_1)$. However, for $w_2 = w^2$ (the random instance), Theorem C.4 just says $\rho \geq 0$, which means it tells us nothing.

For the random instances, we give an even stronger lower bound, which gets rid of the restrictions $w_q = w_u$ and $\rho_q = \rho_u$. This lower bound is tight for any $0 < w_q w_u < w_1 < \min\{w_q, w_u\}$ in the list-of-points model.

Theorem C.5. *Consider any list-of-point data structure for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem over a universe of size d of n points with $w_q w_u d = \omega(\log n)$, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $\alpha \in [0,1]$ we have that*

$$\alpha\rho_q + (1-\alpha)\rho_u \geq \inf_{\substack{t_q, t_u \in [0,1] \\ t_u \neq w_u}} \left(\alpha \frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{d}(t_u \parallel w_u)} + (1-\alpha) \frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{d}(t_u \parallel w_u)} \right),$$

where $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ *and* $T = \arg\inf_{T \ll P, \, \mathop{E}_{X \sim T}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]} \mathrm{D}(T \parallel P).$

[7] As $w, w_1, w_2 \to 0$ we recover the lower bound $\rho \geq \log(\frac{w_1}{w}) / \log\left(\frac{w_2}{w}\right)$ obtained for Chosen Path in [CP17].

Note that for $w_2 = w_q w_u$, the term $\mathrm{D}(T_2 \parallel P_2)$, in Theorem C.2, splits into $\mathrm{d}(t_q \parallel w_q) + \mathrm{d}(t_u \parallel w_u)$, and so the upper and lower bounds perfectly match. This shows that for any linear combination of ρ_q and ρ_u our algorithm obtains the minimal value. By continuity of the terms, this equivalently states as saying that no list-of-points algorithm can get a better query time than our Theorem C.2, given a space budget imposed by ρ_u. [8]

Example 1: Choices for t_q and t_u. As in the upper bounds, it is not easy to prove that a particular choice of t_q and t_u minimizes the lower bound. One might hope that having corresponding lower and upper bounds would help in this endeavour, but alas both results have a minimization. E.g. setting $t_q = 1 - w_u$ and $t_u = 1 - w_q$ the expression in Theorem C.5 we obtain the same value as in Theorem C.2, however it could be (though we strongly conjecture not) that another set of values would reduce both the upper and lower bound.

The good news is that the hypercontractive inequality by Oleszkiewicz [Ole03], can be used to prove certain optimal choices on the space/time trade-off.[9] In particular we will show that for $w_q = w_u = w$ the choice $t_q = t_u = 1 - w$ is optimal in the lower bound, and matches exactly the value $\rho = \log\left(\frac{w_1(1-w)}{w(1-2w+w_1)}\right) / \log(\frac{w_2(1-w)}{w(1-2w+w^2)})$ from Example 1 in the Upper Bounds section.

Example 2: Cell probe bounds Panigrahy et al. [PTW08; PTW10; KP12] created a framework for showing cell probe lower bounds for problems like approximate near-neighbour search and partial match based on a notion of "robust metric expansion". Using the hypercontractive inequalities shown in this paper with this framework, as well (as the extension by [ALRW17a]), we can show, unconditionally, that no data structure, which probes only 1 or 2 memory locations[10], can improve upon the space usage of $n^{1+\rho_u}$ obtained by Theorem C.2 as we let $\rho_q = 0$. In particular, this shows that the near-constant query time regime from Example 3 in the Upper Bounds is optimal up to $n^{o(1)}$ factors in time and space.

C.1.4 Technical Overview

The contributions of the paper are conceptual as well as technical. To a large part, what enables tight upper and lower parts is defining the right problem to study. The second part is realizing which geometry is going to work and proving it in a strong enough model.

[8] It is easy to see that $\rho_u = 0$ minimizes $\alpha\rho_q + (1-\alpha)\rho_u$ when $\alpha = 0$, and similarly $\rho_u = \rho_{max}$ minimizes $\alpha\rho_q + (1-\alpha)\rho_u$ when $\alpha = 1$, where ρ_{max} is the minimal space usage when $\rho_q = 0$. Furthermore, we note that when we change α from 0 to 1, then ρ_u will continuously and monotonically go from 0 to ρ_{max}. This shows that for every $\rho_u \in [0, \rho_{max}]$ there exists an α such that $\alpha\rho_q + (1-\alpha)\rho_u$ is minimized, where ρ_q is best query time given the space budget imposed by ρ_u.

[9] The generalizations by Wolff [Wol07] could in principle expand this range, but they are only tight up to a constant in the exponent.

[10] For 1 probe, the word size can be $n^{o(1)}$, whereas for the 2 probe argument, the word size can only be $o(\log n)$ for the lower bound to hold.

Lastly, a number of tricky algorithmic problems arise, requiring a novel algorithm and a new analysis of 2-dimensional branching random walks of exponentially tilted variables.

Supermajorities – why do they work? Representing sets $x \subseteq U$ as a vector $x \in \{0,1\}^{|U|}$ and scaling by $1/\sqrt{|x|}$, we get $\|x\|_2 = 1$, and it is natural to assume the optimal Similarity Search data structure for data on the unit-sphere — Spherical LSF — should be a good choice. Unfortunately this throws away two key properties of the data: that the vectors are sparse, and that they are non-negative. Algorithms like MinHash, which are specifically designed for this type of data, take advantage of the sparsity by entirely disregarding the remaining universe, U. This is seen by the fact that adding new elements to U never changes the MinHash of a set. Meanwhile Spherical LSF takes the inner product between x and a Gaussian vector scaled down by $1/\sqrt{|U|}$, so each new element added to U, in a sense, lowers the "sensitivity" to x.

In an alternative situation we might imagine $|x|$ being nearly as big as $|U|$. In this case we would clearly prefer to work with $U \setminus x$, since information about an element that is left out, is much more valuable than information about an element contained in x. What Supermajorities does can be seen as balancing how much information to include from x with how much to include from $U \setminus x$. A very good example of this is in Appendix C.4.2, which shows how to view MinHash as an average of simple algorithms that sample a specific amount from each of x and $U \setminus x$. Supermajorities, however, does this in a more clever way, that turns out to be optimal. A crucial advantage is the knowledge of the size of x, as well as the future queries, which allow us to use different thresholds on the storage and query side, each which is perfectly balanced to the problem instance.

As an interesting side effect, the extra flexibility afforded by our approach allows balancing the time required to perform queries with the size of the database. It is perhaps surprising that this simple balancing act is enough to be optimal across all hashing algorithms as well as 1 cell and 2 cell probe data structures.

The results turn out to be best described in terms of the KL-divergences $\mathrm{D}(T \parallel P) - \mathrm{d}(t_q \parallel w_q)$ and $\mathrm{D}(T \parallel P) - \mathrm{d}(t_u \parallel w_u)$, which are equivalent to $\mathrm{D}(T_{XY} \parallel P_{Y|X}T_X)$ and $\mathrm{D}(T_{XY} \parallel P_{X|Y}T_Y)$. Here P_{XY} is the distribution of a coordinated sample from both a query and a dataset, P_X and P_Y are the marginals, and T_{XY} is roughly the distribution of samples conditioned on having a shared representative set. Intuitively these describe the amount of information gained when observing a sample from T_{XY} given a belief that X (resp. Y) is distributed as T and Y (resp. X) is distributed as P. In this framework, Supermajorities can be seen as a continuation of the Entropy LSH approach by [Pan06].

Branching Random Walks Making Supermajorities a real algorithm (rather than just cell probe), requires, as discussed in the introduction, an efficient decoding algorithm of which representative sets overlap with a given cohort. Previous LSF methods can be seen as trees, with independent pruning in each leaf, going back to the LSH forest in 2005 [BCG05; ARN17]. Our method is the first to significantly depart from this idea: While still a tree, our pruning is highly dependent across the levels of the tree, carrying a state from the root to the leaf which needs be considered by the pruning as well as the

analysis. In "branching random walk", the state is represented in the "random walk", while the tree is what makes it branching. While considered heuristically in [BDGL16], such a stateful oracle has not before been analysed, partly because it wasn't necessary. For Supermajorities, meanwhile, it is crucially important. The reason is that failure of the "tensoring trick" employed previously in the literature, when working with thresholds.

The approach from [AI06; BDGL16; ALRW17a] when applied to our scheme would correspond to making our representatives have size just $\sqrt{k}$ (so there are only $|R'| \approx e^{\tilde{O}(\sqrt{\log n})}$ of them,) and then make $R'^{\otimes\sqrt{k}}$ our new R. Since R' can be decoded in $n^{o(1)}$ time, and the second step can be made to take only time proportional to the output, this works well for some cases. This approach has two main issues: (1) There is a certain overhead that comes from not using the optimal filters, but only an approximation. However, this gives only a factor $e^{\tilde{O}(\sqrt{\log n})}$, which is usually tolerated. Worse is (2): Since the thresholds $t_q k$ and $t_u k$ have to be integral, using representative sets of size $\sqrt{k}$ means we have to "repair" them by a multiplicative distortion of approximately $1 \pm 1/\sqrt{k}$, compared to $1 \pm 1/k$ for the "real" filters. This turns out to cost as much as $w_1^{-\sqrt{k}}$ which can easily be much larger than the polynomial cost in n. In a sense, this shows that supermajority functions must be applied to measure the entire representative part of a cohort at once! This makes tensoring not well fit for our purposes.

A pruned branching random walk on the real line can be described in the following way. An initial ancestor is created with value 0 and form the zeroth generation. The people in the ith generation give birth Δ times each and independently of one another to form the $(i+1)$th generation. The people in the $(i+1)$th generation inherit the value, v, of their parent plus an independent random variable X. If ever $v + X < 0$, the child doesn't survive. After k generations, we expect by linearity $\Delta^k \Pr[\forall_{i \le k} \sum_{j \in [i]} X_i \ge 0]$ people to be alive, where X_i are iid. random variables as used in the branching. A pruned 2d-branching random walk is simply one using values $\in \mathbb{R}^2$.

Branching random walks have been analysed before in the Brownian motion literature [Shi15]. They are commonly analysed using the second-moment method, however, as noted by Bramson [Big77]: "an immediate frontal assault using moment estimates, but ignoring the branching structure of the process, will fail." The issue is that the probability that a given pair of paths in the branching process survives is too large for standard estimates to succeed. If the lowest common ancestor of two nodes manages to accumulate much more wealth than expected, its children will have a much too high chance of surviving. For this reason we have to *counterintuitively add extra pruning when proving the lower bound* that a representative set survives. More precisely, we prune all the paths that accumulate much more than the expected value. We show that this does not lower the probability that a representative set is favour by much, while simultaneously decreasing the variance of the branching random walk a lot. Unfortunately, this adds further complications, since ideally, we would like to prune every path that gets below the expectation. Combined with the upper bound this would trap the random walks in a band to narrow to guarantee the survival of a sufficient number of paths. Hence instead, we allow the paths to deviate by roughly a standard deviation below the expectation.

Exponential Tilting and Non-asymptotic Central Limit Lemmas for Random Walks To analyse our algorithm, we need probability bounds for events such as "survival of k generations" that are tight up to polynomial factors. This contrast with many typical analysis approaches in Computer Science, such as Chernoff bounds, which only need to be tight up to a constant in the exponent. We also can't use Central Limit type estimates, since they either are asymptotic (which correspond to assuming w_q and w_u are constants) or too weak (such as Berry Esseen) or just don't apply to random walks.

The technical tool we employ is "Exponential Tilting", which allows coupling the real pruned branching random walk to one that is much more well behaved. This can be seen as a nicer way of conditioning the random walk on succeeding. This nicer random walk then needs to be analysed for properties such as "probability that the path is always above the mean." This is shown using a rearrangement lemma, known as the Truck Driver's Lemma: Assume a truck driver must drive between locations $l_1, l_2, \ldots, l_n, l_1$. At stop i they pick up g_i gas, and between stop i and $i+1$ they expand e_i gas. The lemma say, that if the sum of $g_i - e_i$ is non-negative, then there is a starting position $j \in \{1, \ldots, n\}$ so that the driver's gas level never goes below 0.

This lemma gives an easy proof that a random walk on $\mathbb{R}_+$ of n identically distributed steps, must be always non-negative with probability at least $1/n$ times the probability that it is eventually non-negative. That's because, if the location is eventually non-negative, and all arrangements of steps happen with the same probability, then we must hit the "always non-negative" rotation with probability $\geq 1/n$.

Extending this argument to two dimensions turns out to require a few extra conditions, such as a positive correlation between the coordinates, but as a surprisingly key result, we manage to show Lemma C.11, which says that for $k \in \mathbb{Z}_+$ and $p, p_1, p_2 \in [0,1]$, such that, pk, p_1k, and p_2k are integers and $p \geq p_1p_2$. Let $X^{(i)} \in \{0,1\}^2$ be independent identically distributed variables. We then get that

$$\Pr\left[\forall l \leq k : \sum_{i\in[k]} X^{(i)} \geq [\tfrac{p_1}{p_2}]\, l \;\middle|\; \sum_{i\in[k]} X^{(i)} = [\tfrac{p_1}{p_2}]\, k \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = pk\right] \geq k^{-3}\,.$$

Output-sensitive set decoding In our algorithm we are careful to not have factors of $|U|$ and $|X|$ (the size of the sets) on our query time and space bounds. When sampling our tree, at each level we must pick a certain number, Δ, of elements from the universe and check which of them are contained in the set being decoded. This is an issue, since Δ may be much bigger than $X \cap \Delta$, and so we need an "output-sensitive" sampling procedure. We do this by substituting random sampling with a two-independent hash function $h : U^k \to [q]$, where q is a prime number close to $|U|$. The sampling criterion is then $h(r \circ x) \leq \Delta$, where $\circ$ is string concatenation. The function $h(r)$ can be taken to be $\sum_{i=1}^{k} a_i x_i + b \pmod q$ for random values $a_1, \ldots, a_k, b \in [q]$, so we can expand $h(r \circ x)$ as $h(r) + a_k x \pmod q$.

Now

$$\begin{aligned}\{x \in X \mid (h(r \circ x) \mod q) < \Delta\} \\ &= \{x \in X \mid (h(r) + a_k x \mod q) < \Delta\} \\ &= \cup_{i=0}^{\Delta-1} \{x \in X \mid a_k x \equiv \Delta - h(r) \mod q\} \\ &= \{x \in X \mid (a_k x \mod q) \in [-h(r), \Delta - h(r)] \mod q\},\end{aligned}$$

where the last equation is adjusted in case $(-h(r) \mod q) > (\Delta - h(r) \mod q)$. By pre-computing $\{a_k x \mod q \mid x \in X\}$ (just has to be done one for each of roughly $\log n$ levels in the tree), and storing the result in a predecessor data-structure (or just sorting it), the sampling can be done it time proportional to the size of its output.

Lower Bounds and Hypercontractivity The structure of our lower bounds is by now standard: We first reduce our lower bound to random instances by showing that with high probability the random instances are in fact an instance of our problem. For this to work, we need $w_w |U| = \omega(\log n)$ and in particular $|U| = \omega(\log n)$, so we get concentration around the mean. This requirement is indeed known to be necessary, since the results of [BDGL16; Cha17] break the known lower bounds in the "medium dimension regime" when $|U| = O(\log n)$.

The main difference compared to previous bounds is that we study Boolean functions on so-called p-biased spaces, where the previous lower bounds used Boolean functions on unbiased spaces. This is necessary for us to lower bound every parameter choice for GapSS. In particular we are interested in tight hypercontractive inequalities on p-biased spaces. We say that a distribution $\mathcal{P}_{XY}$ on a space $\Omega_X \times \Omega_Y$ is (r, s)-hypercontractive if

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \leq \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[f(X)^r]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y}[g(Y)^s]^{1/s} ,$$

for all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$, where $\mathcal{P}_X$ and $\mathcal{P}_Y$ are the marginal distributions on the spaces Ω_X and Ω_Y respectively. On unbiased spaces, the classic Bonami-Beckner inequality [Bon70; Bec75] gives a complete understanding of the hypercontractivity. Unfortunately, this is not the case for p-biased spaces where the hypercontractivity is much less understood, with [Ole03] and [Wol07] being state of the art. We sidestep the issue of finding tight hypercontractive inequalities by instead showing an equivalence between hypercontractivity and KL-divergence, which is captured in the following lemma:[11]

Lemma C.6. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be the marginal distributions on the spaces Ω_X and Ω_Y respectively. Let $s, r \in [1, \infty)$, then the following is equivalent*

1. *For all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$,*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \leq \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[f(X)^r]^{1/r} \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y}[g(Y)^s]^{1/s} .$$

[11] It appears that one might prove a similar result using [Nai14] and [Fri15].

2. *For all probability distributions* $\mathcal{Q}_{XY} \ll \mathcal{P}_{XY}$,

$$\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s},$$

where $\mathcal{Q}_X$ *and* $\mathcal{Q}_Y$ *be the marginal distributions on the spaces* Ω_X *and* Ω_Y *respectively*

The main technical argument needed for proving Lemma C.6 is that, for all probability distributions $\mathcal{P}, \mathcal{Q}$, where $\mathcal{Q}$ is absolutely continuous with respect to $\mathcal{P}$, and all functions ϕ,

$$\mathrm{D}(\mathcal{Q} \parallel \mathcal{P}) + \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}}[\exp(\phi(X))] \geq \mathop{\mathrm{E}}_{X\sim\mathcal{Q}}[\phi(X)].$$

This can be seen as a version of Fenchel's inequality, which says that $f(x) + f^*(p) \geq xp$ for all convex functions f, f^*, where f^* is convex conjugate of f, and all $x, p \in \mathbb{R}$.

We use Lemma C.6 together with the "Two-Function Hypercontractivity Induction Theorem" [ODo14], which shows that if $\mathcal{P}_{XY}^{\otimes n}$ is (r,s)-hypercontractive if and only if $\mathcal{P}_{XY}$ is (r,s)-hypercontractive. This implies that $\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}^{\otimes n}}[f(X)g(Y)] \leq \mathrm{E}_{X\sim\mathcal{P}_X^{\otimes n}}[f(X)^r]^{1/r}\,\mathrm{E}_{X\sim\mathcal{P}_Y^{\otimes n}}[g(Y)^s]^{1/s}$ for all functions f, g if and only $\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X\|\mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y\|\mathcal{P}_Y)}{s}$ for all probability distributions $\mathcal{Q}_{XY}$. In the proof of Theorem C.5 we have $\mathcal{P}_{XY} = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ and consider all the probability distributions of the form $\mathcal{Q}_{XY} = \mathop{\arg\inf}_{\mathcal{Q}_{XY}\ll\mathcal{P}_{XY},\, \mathop{E}_{X\sim\mathcal{Q}_{XY}}[X]=\left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]} \mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY})$ for $t_q, t_u \in [0,1]$.

The obtained inequalities can be used directly with the framework by Panigrahy et al. [PTW08] to obtain bounds on "Robust Expansion", which has been shown to give lower bounds for 1-cell and 2-cell probe data structures, with word size $n^{o(1)}$ and $o(\log n)$ respectively.

The Directed Noise Operator We extend the range of our lower bounds further, by studying a recently defined generalization of the p-biased noise operator [ABGM14; AV15; Lif18; KLLM19]. This "Directed Noise Operator", $T_\rho^{p_1\to p_2} : L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d}) \to L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ has the property $\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(S) = \rho^{|S|}\hat{f}^{(p_1)}(S)$ for any $S \subseteq [d]$, where $\hat{f}^{(p)}(S)$ denotes the p-biased Fourier coefficient of f. Just like the Ornstein Uhlenbeck operator, we show that $T_\sigma^{p_2\to p_3} T_\rho^{p_1\to p_2} = T_{\rho\sigma}^{p_1\to p_3}$ and that $T_\rho^{p_2\to p_1}$ is the adjoint of $T_\rho^{p_1\to p_2}$. By connecting this operator to our hypercontractive theorem, we can integrate the results by Oleszkiewicz and obtain provably optimal points on the (t_q, t_u) trade-off.

We show that for p-biased distributions over $\{0,1\}^n$, we can add the following line to the list of equivalent statements in Lemma C.6:

3. For all functions $f : \{0,1\}^n \to \mathbb{R}$ it holds $\|T_\rho^{p_1\to p_2} f\|_{L_{s'}(p_1)} \leq \|f\|_{L_r(p_2)}$.

The operator allows us to prove some optimal choices for r and s in Lemma C.6 (and by effect for t_q and t_u.) Following [ABGM14] we use Pareseval's identity, to write

$\|T_\rho^{p_1\to p_2} f\|_{L_2(p_2)}^2$ as

$$\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(\emptyset)^2 + \widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(\{1\})^2 = \widehat{f}^{(p_1)}(\emptyset)^2 + \rho^2 \widehat{f}^{(p_1)}(\{1\})^2$$
$$= \|T^{p_1\to p_1} f\|_{L_2(p_1)}^2 \le \|f\|_{L_r(p_1)}^2 \ ,$$

where r is perfectly determined by Oleszkiewicz in [Ole03]. It is possible to prove further lower bounds using Hölder's inequality on T, however the bounds obtained this way turn out to be optimal only in the case $s = 2$ or $r = 2$ that also follow from Parseval. A particular simple case is $r = s =$, $w_q = w_u = w$, and $w_2 = w^2$, in which case the arguments above gives the lower bound $\rho \ge \log(\frac{w_1(1-w)}{w(1-2w+w_1)})/\log(\frac{1-w}{w})$ mentioned in Example 1 in the Upper Bounds section.

Another use of T is in proving lower bounds outside of the random instance $w_2 = w_q w_u$ regime. Using the power means inequality over p-biased Fourier coefficients, we show the relation

$$\left(\langle T_\alpha^{p\to p} f, f\rangle_{L_2(p)} / \|f\|_{L_2(p)}^2\right)^{1/\log(1/\alpha)} \le \left(\langle T_\beta^{p\to p} f, f\rangle_{L_2(p)} / \|f\|_{L_2(p)}^2\right)^{1/\log(1/\beta)} .$$

which is allows comparing functions under two different noise levels. This is stronger than hypercontractivity, even though we can prove it in fewer instances. The proof can been seen as a variation of [OWZ14] and we get a lower bound with a similar range, but without asymptotics and for Set Similarity instead of Hamming space Similarity Search.

C.1.5 Related Work

For the reasons laid out in the introduction, we will compare primarily against approximate solutions. The best of those are all able to solve GapSS, thus making it easy to draw comparisons. The guarantees of these algorithms are listed in Table C.1 and we provide plots in Figure C.2 and Figure C.3 for concreteness.

The methods known as Bit Sampling [IM98] and SimHash (Hyperplane rounding) [Cha02], while sometimes better than MinHash[BGMZ97] and Chosen Path [CP17] are always worse (theoretically) that Spherical LSF, so we won't perform a direct comparison to those.

It should be noted that both Chosen Path and Spherical LSF both have proofs of optimality in the restricted models. However these proofs translated to only a certain region of the (w_q, w_u, w_1, w_2) space, and so they may nearly always be improved.

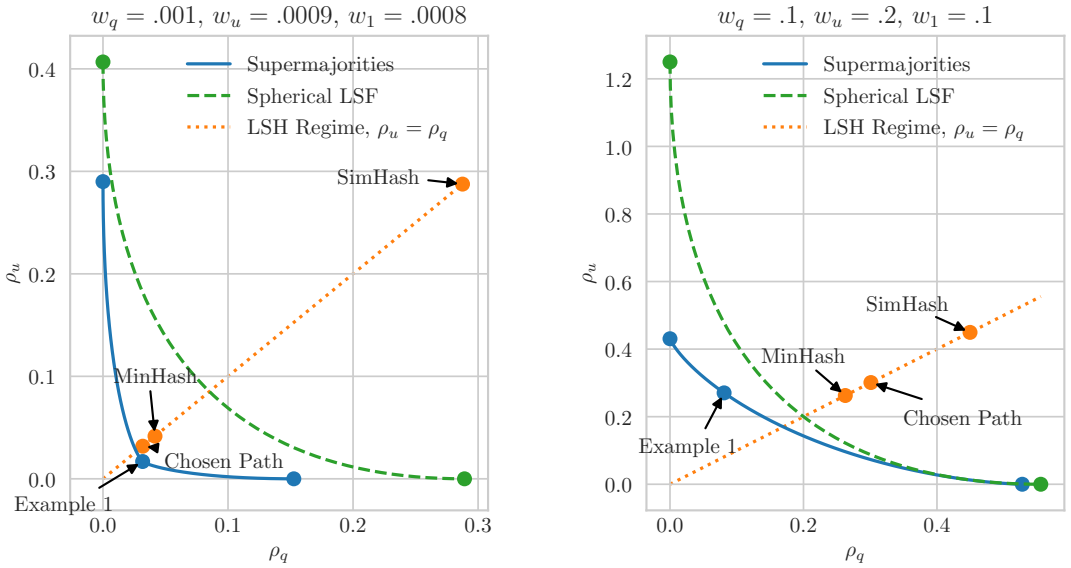
Arguably the largest break-through in Locality Sensitive Hashing, LSH, based data structures was the introduction of *data-dependent* LSH [AINR14; AR15; ARN17]. It was shown how to reduce the general case of α, β similarity search as described above, to the case $(\alpha, \beta) \mapsto (\frac{\alpha-\beta}{1-\beta}, 0)$, in which many LSH schemes work better. Using those data structures on GapSS with $w_2 > w_q w_u$ will often yield better performance than the algorithms described in this paper. However, in the "random instance" case $w_2 = w_q w_u$, which is the main focus of this paper, data-dependency has no effect, and so this issue won't show up much in our comparisons.

Method	**Balanced $\rho_q = \rho_u$**	**Space/time trade-offs**
Spherical LSF [TT07; Laa15] [Chr17; ALRW17a]	$\frac{1-\alpha}{1+\alpha}\frac{1+\beta}{1-\beta}$	$\rho_q = \frac{(1-\alpha^{1+\lambda})^2}{1-\alpha^2}\frac{1-\beta^2}{(1-\alpha^\lambda\beta)^2}$ (***) $\rho_u = \frac{(1-\alpha^{1+\lambda})^2}{1-\alpha^2}\frac{1-\beta^2}{(1-\alpha^\lambda\beta)^2}$
MinHash [BGMZ97]	$\frac{\log\frac{w_1}{w_q+w_u-w_1}}{\log\frac{w_2}{w_q+w_u-w_2}}$	Same as above(*) with $\alpha = \frac{w_1}{w_q+w_u-w_1}, \beta = \frac{w_2}{w_q+w_u-w_2}$
Chosen Path [CP17]	$\frac{\log\frac{w_1}{\max\{w_q,w_u\}}}{\log\frac{w_2}{\max\{w_q,w_u\}}}$	N/A
Supermajorities (This paper)	Theorem C.2, Example 1	Theorem C.2
Data-Dependent LSF [AR15; ALRW17a]	$\frac{1-\alpha}{1+\alpha-2\beta}$	$\sqrt{\rho_q} + \alpha'\sqrt{\rho_u} = \sqrt{1-\alpha'^2}$ where $\alpha' = 1 - \frac{1-\alpha}{1-\beta}$
SimHash [Cha02]	$\frac{\log(1-\arccos(\alpha)/\pi)}{\log(1-\arccos(\alpha)/\pi)}$	N/A(**)
Bit Sampling [IM98]	$\frac{\log(1-w_q-w_u+2w_1)}{\log(1-w_q-w_u+2w_2)}$	N/A(**)

Table C.1: Time and space exponents for the best similarity search data-structures. For Spherical LSF and SimHash, α and β are the inner products between sets represented as vectors, and can by Lemma C.26 be taken to be $\alpha = \frac{w_1 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$ and $\beta = \frac{w_2 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$.
(*): Space/time trade-offs for MinHash can be obtained using MinHash as an embedding for Spherical LSF. (**): Some space/time trade-offs can be obtained for LSH using Multi-probing [LJWC+07]. (***): $\lambda \in [-1, 1]$ controls the space/time trade-off.

We note that even without a reduction to the random instance, for many practical uses, it is natural to assume such "independence" between the query and most of the dataset. Arguably this is the main reason why approximate similarity search algorithms have gained popularity in the first place. In practice, some algorithms for Set Similarity Search take special care to handle "skew" data distributions [RSW20; ZLWZ+17; MMP18], in which some elements of the Universe are heavily over or under-represented. By special casing those elements, those algorithms can be seen as reducing the remaining dataset to the random instance. Curiously, even the early research on Partial Match by Ronald Rivest in his PhD thesis [Riv76], studied the problem on random data.

Many of the algorithms, based on the LSH framework, all had space usage roughly



(a) Example of search with very small sets.

(b) Example with larger sets of different sizes.

Figure C.2: Comparison to Spherical LSF: Plots of the achievable ρ_q (time exponent) and ρ_u (space exponent) achievable with Theorem C.2. Note that using our optimal spherical embedding from Lemma C.26 is critical to achieve the exponents shown for Spherical LSF. The plots are drawn in the "random setting", $w_2 = w_q w_u$ where Spherical LSF and Data-Dependent LSH coincide.

$n^{1+\rho}$ and query time n^{ρ} for the same constant ρ. This is known as the "balanced regime" or the "LSH regime". Time/space trade-offs are important, since $n^{1+\rho}$ can sometimes be too much space, even for relatively small ρ. Early work on this was done by Panigrahy [Pan06] and Kapralov [Kap15] who gave smooth trade-offs ranging from space $n^{1+o(1)}$ to query time $n^{o(1)}$. A breakthrough was the use of LSF (rather than LSH), which allowed time/space trade-offs with sublinear query time even for near linear space and small approximation [Laa15; Chr17; ALRW17b].

We finally compare our results to the classical literature on Partial Match and Super-/Subset search, which has some intriguing parallels to the work presented here.

Comparison to Spherical LSF We use "Spherical LSF" as a term for the algorithms [BDGL16] and [Laa15], but in particular section 3 of [ALRW17a], which has the most recent version. The algorithm solves the (r, cr)-Approximate Near Neighbour problem, in which we, given a dataset $Y \subseteq \mathbb{R}^d$ and a query $q \in \mathbb{R}^d$ must return $y \in Y$ such that $\|q - y\|_2 < cr$ or determinate that there is no $y' \in Y$ with $\|y - q\|_2 \leq r$.

The algorithm is a tree over the points, P. At each node they sample T i.i.d. Gaussian d-dimensional vectors $z_1, \ldots, z_T$ and split the dataset up into (not necessarily disjoint) "caps" $P_i = \{p \in P \mid \langle z_i, p \rangle \geq t_u\}$. They continue recursively and independently until the

expected number of leaves shared between two points at distance $\geq cr$ is $\approx n^{-1+\varepsilon}$.

The real algorithm also samples includes some caps that are dependent on an analysis of the dataset. This allows obtaining a query time of $n^{1/(2c^2-1)}$, for all values of r, rather than only in the "random instance", which, for data on the sphere, corresponds to $r = 1/(\sqrt{2}c)$. (To see this, notice that $rc = 1/\sqrt{2}$, which is the expected distance between two orthogonal points on a sphere.)

Whether we analyse the data-independent algorithm or not, however, a key property of Spherical LSF is that each node in the tree is independent of the remaining nodes. This allows a nice inductive analysis. In comparison, in our algorithm, the nodes are not independent. Whether a certain node gets pruned, depends on which elements from the universe were sampled at all the previous nodes along the path from the root. One could imagine doing Spherical LSF with a running total of inner products along each path, which would make the space partition more smooth, and possible better in practice. Something along these lines was indeed suggested in [BDGL16], however it wasn't analysed, as for Spherical LSF *the inner products at each node are continuous, and the thresholds can be set at any precision.*

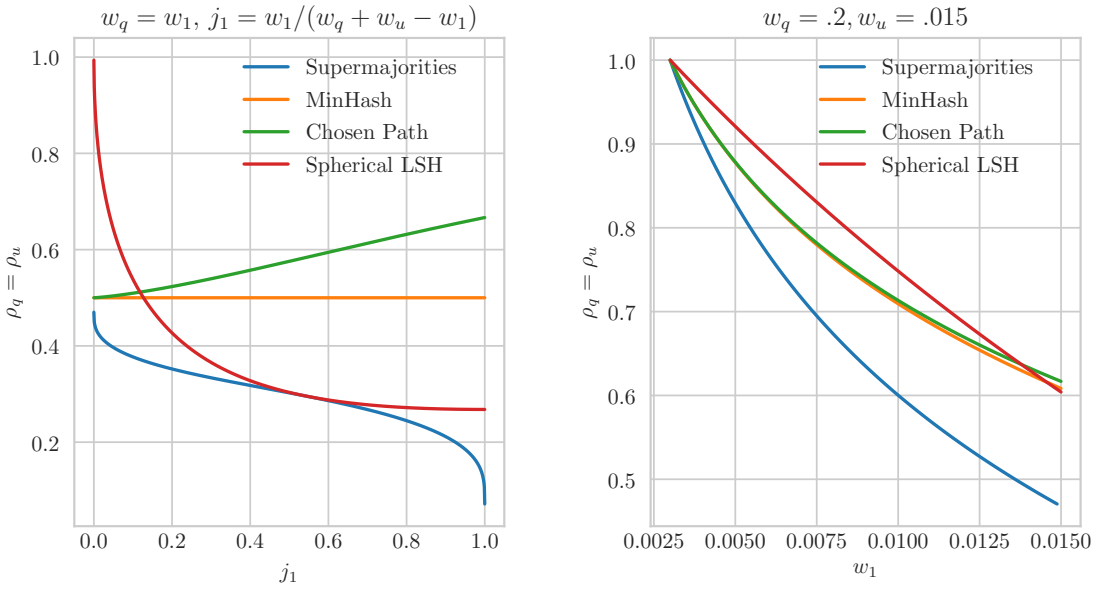
It is clear that Spherical LSF can solve GapSS – one simply needs an embedding of the sets onto the sphere. An obvious choice is $x \mapsto x/\|x\|_2$. This was used in [CP17] when comparing Chosen Path to Spherical LSF. However it is also clear that the choice of embedding matters on the performance one gets out of Spherical LSF. Other authors have considered $x \mapsto (2x-1)/\sqrt{d}$ and various asymmetric embeddings [SL14a].

We would like to find the most efficient embedding to get a fair comparison. However, we don't know how to do this optimally over all possible embeddings, which include using MinHash and possibly somehow emulating Supermajorities.[12] We instead find the most efficient *affine* embedding, which turns out to be surprisingly simple, and which encompasses all previously suggested approaches. In Lemma C.4.1 we prove a general result, implying that the embedding is optimal for Spherical LSF as well as other spherical data structures like SimHash. In Figure C.2 and Figure C.3 the ρ-values of Spherical LSF are obtained using this optimal embedding.

From the figures, we see the two main cases in which Spherical LSF is suboptimal. As the sets get very small ($w_q, w_u, w_1 \to 0$) the ρ value in the LSH regime goes to 1, whereas Supermajorities (as well as MinHash and Chosen Path) still obtain good performance. Similarly in the asymmetric case $w_q \neq w_u$, as we make ρ_q very small, the performance gap between Supermajorities and Spherical LSF can grow to arbitrarily large polynomial factors.

Comparison to MinHash Given a random function $h : \mathcal{P}(\{1,\ldots,d\}) \to [0,1]$, the MinHash algorithm hashes a set $x \subseteq \{1,\ldots,d\}$ to $m_h(x) = \arg\min_{i\in x} h(i)$. One can show that $Pr[m_h(x) = m_h(y)] = J(x,y) = \frac{|x\cap y|}{|x\cup y|}$. Using the LSH framework by Indyk and Motwani [IM98] this yields a data structure for Approximate Set Similarity Search over Jaccard similarity, J, with query time dn^ρ and space usage $n^{1+\rho} + dn$, where $\rho = \frac{\log j_1}{\log j_2}$ and

[12] We would also need some sort of limit on how much time the embedding takes to perform.



(a) Varying the Jaccard similarity, j_1, among close sets, while fixing the exponent of MinHash at $\rho = .5$ in the subset search instance, $w_q = w_1$. The plot shows the case of balanced exponents, $\rho_q = \rho_u$, between queries and updates.

(b) Varying the overlap w_1 among close sets while fixing the query and database set sizes. Note that at $w_1 = w_2 = .002$ there is no gap between close and far sets, and so all algorithms have exponents $\rho = 1$.

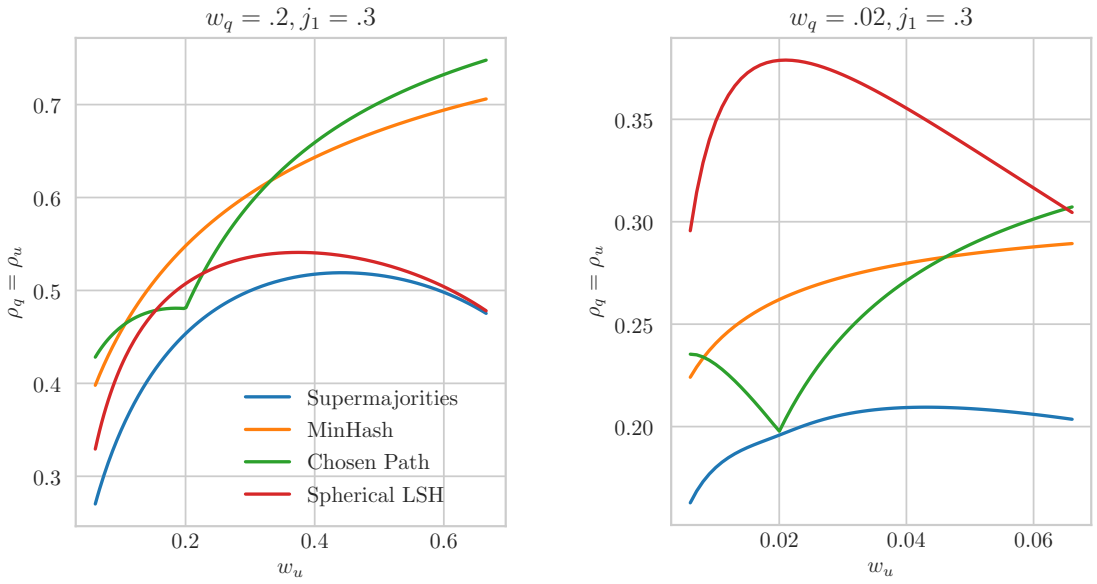
Figure C.3: Comparison to MinHash: Varying different parameters while searching on a background of random sets ($w_2 = w_q w_u$), Supermajorities regularly get substantially better time and space exponents. The plots are drawn in the "random setting", $w_2 = w_q w_u$ and use the optimal embedding for Spherical LSF.

j_1 and j_2 define the gap between "good" and "bad" search results. As Jaccard similarity is a set similarity measures, it is clear that MinHash yields a solution to the GapSS problem with $\rho_q = \rho_u = \log \frac{w_1}{w_q+w_u-w_1} / \log \frac{w_2}{w_q+w_u-w_2}$. Similarly, and that any solution to GapSS can yield a solution to Approximate SSS over Jaccard similarity.

MinHash has been very popular, since it gives a good, all-round algorithm for Set Similarity Search, that is easy to implement. In Figure C.3 we see how MinHash performant for different settings of GapSS. In particular we see that when solving the Superset Search problem, which is a common use case for MinHash, our new algorithm obtains quite a large polynomial improvement, except when the Jaccard similarity between the query and the sought after superset is nearly 0 (which is hardly an interesting situation.)

It is possible to use MinHash as an embedding (or densification) of sets into Hamming space or onto the Sphere. We can then use Spherical LSF to get space/time trade-offs. We have not plotted those, but we can notice that in the balanced case, $\rho_q = \rho_u$, this would give $\rho = \frac{1-j_1}{1+j_1}\frac{1+j_2}{1-j_2}$, which is worse than $\rho = \log j_1 / \log j_2$ obtained by the direct algorithm.

MinHash is quite different from the other algorithms considered in this section. For



(a) In the plot, Chosen Path never matches Supermajorities, even at $w_q = w_u$ since the sets are relatively large.

(b) In the plot, Chosen Path nearly matches Supermajorities when $w_u = w_q$ as the sets are relatively small.

Figure C.4: Comparison to Chosen Path: Fixing w_q and the Jaccard similarity so $w_1 = \frac{j_1}{1+j_1}(w_q + w_u)$, we vary w_u to see the performance of different algorithms at different levels of asymmetry in the set sizes. The plots are drawn in the "random setting", $w_2 = w_q w_u$ and use the optimal embedding for Spherical LSF.

some more intuition of why MinHash is not optimal for Approximate Set Similarity Search, we show in Appendix C.4.2 that MinHash can be seen as an average of a family of Chosen Path like algorithms. We also show that an average is always worse than simply using the best family member, which implies that MinHash is never optimal.

Comparison to Chosen Path The Chosen Path algorithm of [CP17], is virtually identical to Supermajorities, when parametrized with $t_q = t_u = 1$. Similar to Spherical LSF and our decoding algorithm, they build a tree on the datasets. For each node they sample iid. Elements $x_1, x_2, \cdots \in U$ from the universe, and split the data into (not necessarily disjoint) subsets $P_i = \{p \in P \mid x_i \in p\}$. They again continue recursively and independently until the expected number of leaves shared between two dissimilar points is sufficiently small.

The case $t_q = t_u = 1$ however, turns out to be a very special case of our algorithm, because one can decide which leaves of the tree to prune, without knowledge of what happened previously on the path from the root to the node. This allows a nice inductive analysis of Chosen Path based on second moments, which is a classic example literature on branching processes. Meanwhile, for our general algorithm, we need to analyse the

resulting branching random walk, a conceptually much different beast.

Doing the analysis, one gets a data structure for Approximate Set Similarity Search over Braun-Blanquet similarity, $B(x,y) = \frac{|x \cap y|}{\max\{|x|,|y|\}}$, with query time $|q|n^{\rho}$ and auxiliary space usage $n^{1+\rho}$, where $\rho = \frac{\log b_1}{\log b_2}$ and b_1 and b_2 define the gap between "good" and "bad" search results. Since $t_q = t_u = 1$ is sometimes the optimal choice for Supermajorities, it is clear that we must sometimes coincide in performance with Chosen Path. In particular, this happens as $w_q = w_u$ and $w_q, w_u, w_1 \to 0$. This is also one of the case where our lower bound Theorem C.4 is sharp, which confirms, in addition to the lower bound in [CP17] that both algorithms are sharp for LSF data structures in this setting. Figure C.2a shows how Chosen Path does nearly as well as Supermajorities on very small sets.

In the case $w_q = w_u$ the ρ value of Chosen Path can be equivalently written in terms of Jaccard similarities as $\log \frac{2j_1}{1+j_1} / \log \frac{2j_2}{1+j_2}$, which is always smaller than the $\log j_1 / \log j_2$ obtained by MinHash. (This value, $2j/(1+j)$, is also known as the Sørensen-Dice coefficient of two sets.) However, in the case $w_q \neq w_u$ Chosen Path can be much worse than MinHash, as seen in Figure C.2b and Figure C.3a. In [CP17] it was left as an open problem whether MinHash could be improved upon in general. It is a nice result that the balanced ρ value of Supermajorities (when $\rho_q = \rho_u$) can be shown (numerically) to always be less than or equal to $\log \frac{2j_1}{1+j_1} / \log \frac{2j_2}{1+j_2}$, even when $w_q \neq w_u$. It is a curious problem for which similarity measure, S, so the balanced ρ value of Supermajorities equal $\log s_1 / \log s_2$.

Partial Match (PM) and Super-/Subset queries (SQ) Partial Match asks to preprocess a database D of n points in $\{0,1\}^d$ such that, for all query of the form $q \in \{0,1,*\}^d$, either report a point $x \in D$ matching all non-$*$ characters in q or report that no such x exists. A related problem is Super-/Subset queries, in which queries are on the form $q \in \{0,1\}^d$, and we must either report a point $x \in D$ such that $x \subseteq q$ (resp. $q \subseteq x$) or report that no such x exists.

The problems are equivalent to the subset query problem by the following folklore reductions: (PM $\to$ SQ) Replace each $x \in D$ by the set $\{(i, p_i) : i \in [d]\}$. Then replace each query q by $\{(i, q_i) : q_i = *\}$. (SQ $\to$ PM) Keep the sets in the database as vectors and replace in each query each 0 by an $*$.

The classic approach, studied by Rivest [Riv76], is to split up database strings like `supermajority` and file them under `s`, `u`, `p` etc. Then when given query like `set` we take the intersection of the lists `s`, `e`, `t`. Sometimes this can be done faster than brute force searching each list. He also considered the space heavy solution of storing all subsets, and showed that when $d \leq 2 \log n$, the trivial space bound of 2^d can be somewhat improved. Rivest finally studied approaches based on tries and in particular the case where most of the database was random strings. The latter case is in some ways similar to the LSH based methods we will describe below.

Indyk, Charikar and Panigrahy [CIP02] also studied the exact version of the problem, and gave, for each $c \in [n]$, an algorithm with $O(n/2^c)$ time and $n2^{(O(d \log^2 d \sqrt{c/\log n})}$ space, and another with $O(dn/c)$ query time and nd^c space. Their approach was a mix between the shingling method of Rivest, building a look-up table of size $\approx 2^{\Omega(d)}$, and a brute force

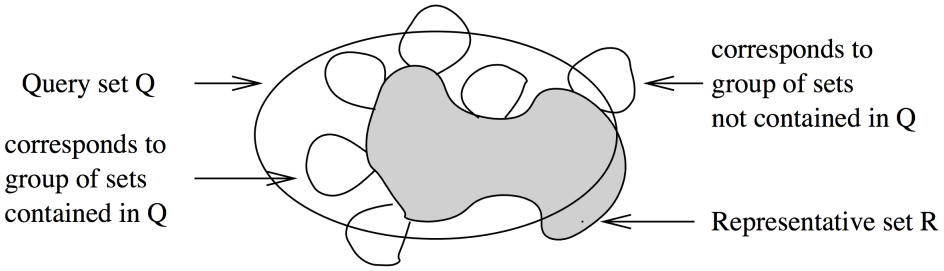


Figure C.5: This figure from the Partial Match algorithm of [CIP02] shares some of the same geometrical intuition visible in our own figure C.1a.

search. These bounds manage to be non-trivial for $d = \omega(\log n)$, however only slightly. (e.g. $n/\operatorname{poly}(\log n)$ time with polynomial space.)

There has also been a large number of practical papers written on Partial Match / Subset queries or the equivalent batch problem of subset joins [RPNK00; MG03; GG10; AAK10; FMNM19]. Most of these use similar methods to the above, but save time and space in various places by using bloom filters and sketches such as MinHash [BGMZ97] and HyperLogLog [FFGM07].

Maximum Inner Product (MIPS) is the Similarity Search problem with $S(x, y) = \langle x, y\rangle$ — the Euclidean inner product. For exact algorithms, most work has been done in the batch version (n data points, n queries). Here Alman et al. [ACW16] gave an $n^{2-1/\tilde{O}(\sqrt{k})}$ algorithm, when $d = k \log n$.

An approximative version can be defined as: Given $c > 1$, pre-process a database D of n points in $\{0,1\}^d$ such that, for all query of the form $q \in \{0,1\}^d$ return a point $x \in D$ such that $\langle q, x\rangle \geq \frac{1}{c} \max_{x' \in D} \langle q, x'\rangle$. Here [APRS16] gives a data structure with query time $\approx \tilde{O}(n/c^2)$, and [CW19] solves the batch problem in time $n^{2-1/O(\log c)}$ (both when d is $n^{o(1)}$.)

There are a large number of practical papers on this problem as well. Many are based on the Locality Sensitive Hashing framework (discussed below) and have names such as SIMPLE-LSH [NS15] and L2-ALSH [SL14a]. The main problem for these algorithms is usually that no hash family of functions $h : \{0,1\}^d \times \{0,1\}^d \to [m]$ such that $\Pr[h(q) = h(x)] = \langle q, x\rangle/d$ [APRS16] and various embeddings and asymmetries are suggested as solutions.

The state of the art is a paper from NeurIPS 2018 [YLDC+18] which suggests partitioning data by the vector norm, such that the inner product can be more easily estimated by LSH-able similarities such as Jaccard. This is curiously very similar to what we suggest in this paper.

We will not discuss these approaches further since, for GapSS, they all have higher exponents than the three LSH approaches we study next.

C.2 The Algorithm

We now describe the full algorithm that gives Theorem C.2. We state the full version of the theorem, discuss it and prove it. The section ends with an involved analysis of the survival probabilities of the branching random walk.

Notationally we define $[n] = \{1, \dots, n\}$ and let $(\cdot \circ \cdot) : A^{l_1} \times A^{l_2} \to A^{l_1+l_2}$ be the concatenation operator for any set A and integers l_1, l_2. We will use the Iversonian bracket, defined by $[P] = 1$ if P and 0 otherwise. For R and U sets, we have $R \times U = \{r \circ u \mid r \in R, u \in U\}$ $\mathcal{P}(U)$ is the power set of U.

The first step is to set up our assumptions. For $w_q, w_u, w_1, w_2, t_q, t_u \in [0, 1]$ given, we can assume $\min\{w_q, w_u\} \geq w_1 > w_2$ and $t_q \neq w_q, t_u \neq w_u$. We are also given a universe U and a family $Y \subseteq \binom{U}{w_u|U|}$ of size $|Y| = n$.

It will be nice to assume $|U| = q$ where q is a prime number. This can always be achieved by adding at most $|U|^{0.525}$ elements to U large enough[13] [BHP01]. Hence we only distort each of w_q, w_u, w_1, w_2 by roughly a factor $1 + O(|U|^{-1/2})$, which is insignificant for $|U| = \Omega(\log n)^2$, and we can always increase $|U|$ without changing the problem parameters by duplicating the set elements.

Let $k \in \mathbb{Z}_+$ be defined later. For all $i \in [k]$ we define $h_i(r) : [q]^i \to [q]$ by $h_i(r) = \sum_{j \in [i]} a_{i,j} r_j + b_i \mod q$ for some sequences of random numbers $a_{i,j} \in [q] \setminus \{0\}, b_i \in [q]$, such that each h_i is a 2-independent random function. (That means $\Pr[h_i(r) = h_i(r')] \leq 1/q$ for $r \neq r'$.)

Finally two sequences $(\Delta_i \in \mathbb{Z}_+)_{i \in [k]}$ and $(c_\ell \in \mathbb{R}^2)_{\ell \in [k]}$ to be specified later. We can now define the sets $R_i = \{r \circ x \in R_{i-1} \times U \mid h_i(r \circ x) < \Delta_i\}$, as well as the decoding functions

$$\mathcal{R}_i(X, t) = \left\{ r \in R_i \;\middle|\; \forall \ell \leq i : \sum_{j \in [\ell]} [r_j \in X] \geq t\ell - c_\ell \right\}$$

Intuitively R_i are our representative sets at level i in the tree, such that R_k is a close to iid. uniform sample from U^k. The decoding function takes a set $X \subseteq U$ and a value $t \in [0, 1]$, and returns all $r \in R_i$ such that all prefixes r' of r "$(t - \varepsilon)$-favours" X (as defined by $|r \cap X|/|r| \geq t - \varepsilon$ in the introduction), where $\varepsilon = \frac{c_{|r'|}}{|r'|}$ is some slack that helps ensure survival of at least one representative set. The slack won't be the same on each coordinate, but scaled by their variance. The algorithm is shown below as pseudo-code in Algorithm 1.

Our data structure now builds a hash-table M of lists of pointers and store each set $y \in Y$ in $M[r]$ for every $r \in R_k(y, t_u)$. One can think of this as storing the elements at the leafs of the tree represented by the sets R_i. On a new query $q \in \binom{U}{w_q|U|}$ we look at every list $M[r]$ for $r \in R_k(q, t_q)$. For each y in such a list, we compute the intersection with q and return y if $|q \cap y|/|U| \geq w_2$. This takes time $\min\{w_u, w_q\}|U|$, which would be a large multiplicative factor on our query time, so we may instead choose to sample just

$$O(\min\{w_q, w_u\} w_2^{-1} \log n) \tag{C.1}$$

[13] It is an open conjecture by Harald Cramér that $(\log |U|)^2$ suffices as well. [Cra36]

Algorithm 1: Pseudocode for the decoding function $\mathcal{R}$.

Input: Universe U, Set $X \subseteq U$, Threshold $t \in [0, 1]$
Result: Set $P_k \subseteq U^k$ of paths
$R_0 \leftarrow \{((), 0)\}$ `// These` R_i `values contain the paths and scores`
for $i = 1$ **to** k **do**
  $R_i \leftarrow \{\}$
  for $(r, s) \in R_{i-1}$ **do**
    for $x \in U$ *st.* $h_i(r \circ x) < \Delta_i$ **do** `// Sample the universe`
      $s' \leftarrow s + [x \in X]$
      if $s' \geq it - c_i$ **then** `// Trim to promising paths`
        $R_i \leftarrow R_i \cup \{(r \circ x, s')\}$
      end
    end
  end
end

elements, which suffices as a test with high probability.

This describes the entire algorithm, exception for an optimization for the "Sample the universe" step above, which naively implemented would take time $|X|$. This optimization is the reason $|U|$ was chosen to be a prime number.

An optimization In the "Sample the universe" step of Algorithm 1 a naive implementation spends time $|X|$ hashing all possible elements and comparing their value to Δ_i. We now show how to make this step output sensitive, using only time equal to the number of values for which the condition is true. [14]

The requirement $s' \geq it - c_i$ we call the "trimming condition". This allows us to trim away most prefix paths which would be very unlikely to ever reach our requirement for the final path. To speed up finding all $x \in U$ such that $h_i(r \circ x) < \Delta_i$ we note that there are two cases relevant to the trimming condition, depending on s in the algorithm: (1) s' has to be $s + 1$ or (2) $s' = s$ suffices. In the first case we are only interested in x values in X, while in the second case, all $x \in U$ values are relevant.

We have $h_i(r \circ x) = \eta + ax \mod q$ for some values η, a and b where $a > 0$. In case (2) the relevant x are simple $\{a^{-1}(v - \eta) \mod q \mid v \in [\Delta_i]\}$, where a^{-1} exists because q is prime. For the case (1) where x must be in X, we pre-process X by storing $ax \mod q$ for $x \in X$ in a sorted list. Using a single binary search, we can then find the relevant values with a time overhead of just $\lg |X|$. Using a more advanced predecessor data structure, this overhead can be reduced. See Algorithm 2 for a pseudocode version of this idea.

C.2.1 Full Theorem

We state the full version of Theorem C.2 and a discussion of the differences between it and the idealized version in the introduction.

[14] The subroutine is inspired by personal communications with Rasmus Pagh and Tobias Christiani.

Algorithm 2: Output sensitive sample

Input : $r \in [q]$, $\Delta \in [q]$
Pre-process: $s = \text{sorted}\{h(x) \mid x \in X\} \in [q]^{|X|}$ and $\kappa \in X^{|X|}$ st. $h(\kappa[i]) = s[i]$.
Result: $R = \{x \in X \mid (h(x) + r \mod q) < \Delta\}$
$i \leftarrow \min\{i \in [|X|] \mid s[i-1] < q - r \leq s[i]\}$ `// We assume` $s[i] = -\infty$ `for` $i < 0$
$R \leftarrow \{\}$
while $(s[i \mod |X|] + r \mod q) < \Delta$ **do**
  $R \leftarrow R \cup \{\kappa[i \mod |X|]\}$
  $i \leftarrow i + 1$
end

Theorem C.2 (Full version)**.** *Let $w_q, w_u \geq w_1 \geq w_2 \geq 0$ be given with $w_1 \geq w_q w_u$ and $1 \leq t_q, t_u \leq 0$. Set k to be the smallest even integer greater than or equal to $\frac{\log n}{\mathrm{D}(T_2 \| P_2) - \mathrm{d}(t_q \| w_q)}$ and assume that $t_q k/2$ and $t_u k/2$ are integers. The (w_q, w_u, w_1, w_2)-GapSS problem over a universe U can be solved with expected query time*

$$\begin{aligned}
\textit{query time} \quad & O\big(\varsigma_q\, k^{28}\, n^{\rho_q} + k w_q\, |U| + \big(\tfrac{t_q(1-w_q)}{(1-t_q)w_q}\big)^{\sqrt{t_q(1-t_q)k \cdot 6.5 \log(3k)}}\big), \\
\textit{space usage} \quad & O(\varsigma_u\, k^{28}\, n^{1+\rho_u} + n w_u\, |U|) \\
\textit{and update time} \quad & O\big(\varsigma_u\, k^{28}\, n^{\rho_u} + k w_u\, |U| + \big(\tfrac{t_u(1-w_u)}{(1-t_u)w_u}\big)^{\sqrt{t_u(1-t_u)k \cdot 6.5 \log(3k)}}\big),
\end{aligned}$$

$$\textit{where} \quad \rho_q = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)} \quad \textit{and} \quad \rho_u = \frac{\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)},$$

$$\textit{and} \quad \varsigma_q = \frac{\min\{w_q, w_u\}}{w_2} e^{2(\mathrm{D}(T_1 \| P_1) - \mathrm{d}(t_q \| w_q))}, \quad \varsigma_u = e^{2(\mathrm{D}(T_1 \| P_1) - \mathrm{d}(t_u \| w_u))}.$$

We stress that all previous Locality Sensitive algorithms with time/space trade-offs had $n^{o(1)}$ factors on n^{ρ_q} and n^{ρ_u}. These could be as large as $\exp(\sqrt{\log n})$ or even $\exp((\log n)/(\log \log n))$. In contrast, our algorithm is the first that only loses $k \approx \log(n)$ multiplicative factors!

In the statement of Theorem C.2 we have taken great effort to make sure that any dependence on $w_q, w_u, w_1, w_2, t_q, t_u$ is visible and only truly universal constants, like 4, are hidden in the $O(\cdot)$.

The main thing we do lose is the additive $(\frac{t_q(1-w_q)}{(1-t_q)w_q})^{\tilde{O}(\sqrt{t_q(1-t_q)k})}$. We may note the bound $(\frac{t_q}{1-t_q})^{\sqrt{t_q(1-t_q)}} \leq 2$, so the main eyesore is the $1/w_q$. For $w_q > e^{-\tilde{O}(\sqrt{\log n})}$ this is dominated by the main term, but for very small sets it could potentially be an issue. However, it turns out that as w_q and w_u get small, the optimal choices of t_q and t_u move towards 0 or 1. Since this effect is exponentially stronger we get that $(1/w_q)^{\sqrt{t_q(1-t_q)}}$ is usually never more than a small constant. It also means that we recover the performance of Chosen Path in the case $t_q = 1$, $t_u = 1$, which has no $\Omega(e^{\sqrt{\log n}})$ terms. [15]

[15] The authors know of a way to reduce the error term further, so it only appears in the $\rho_q = 0$ case, and only as $\exp((\log 1/w_q)^{2/3} k^{1/3})$ which is $o(n)$ for any $w_q = \omega(1/n)$.

In case w_q^{-1} is large, but w_2 is not too small, we can reduce w_q^{-1} to $\frac{w_u}{w_2}k$ by hashing! Sketch: Define a hash function $h : U \to [m]$ where $m = O(\frac{w_q w_u}{w_2}|U|k)$ and map each set y to $\{i \in [m] \mid \exists e \in y : h(e) = i\}$, that is the OR of the hashed values. With high probability this only distorts the size of the sets and their inner products by a factor $(1 + 1/k)$ which doesn't change ρ.

The constants of the size ς_q and ς_u are standard in all other similar algorithms since [IM98], as they come from the requirement that k is an integer. The terms $\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q)$ and $\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_u \parallel w_u)$ in ς_q and ς_u may be bounded by $\log \frac{w_u}{w_1}$ and $\log \frac{w_q}{w_1}$ respectively. The factor of 2 on those terms come from the tensoring step done on paths of length $k/2$. This can be removed at the cost of making the ratio-of-odds term multiplicative in the bounds above. The factor $\min\{w_q, w_u\}/w_2$ in ς_q comes from equation (C.1) and is the time it takes to verify a candidate identified by the filtering. Note that this factor would exist even in a brute force $O(n)$ algorithm and exists in any data structures known for similar problems. In fact, for small n, it is necessary due to communication complexity bounds.

Proof of Theorem C.2. Let $\mathcal{T}_q$ and $\mathcal{T}_u$ be the time it takes to compute $R_k(x, t_q)$ and $R_k(y, t_u)$ on given sets. When creating the data structure, decoding each $y \in Y$ takes time $n\mathcal{T}_u$ and uses $n\,\mathrm{E}[|R_k(Y, t_u)|]$ words of memory for space equivalent. When querying the data structure we first use time $\mathcal{T}_q$ to decode q, then $\mathrm{E}[|R_k(X, t_q)|]$ time to look in the buckets, and finally $\frac{\min\{w_q, w_u\} \log n}{w_2}$ time on each of $\mathrm{E}[|R_k(X, t_q) \cap R_k(Y, t_u)|]\, n$ expected collisions with far sets (the worst case is that we never find any y with $y \cap q > w_2|U|$ so we can't return early.)

The key to proving the theorem is thus bounding the above quantities. We do this using the following lemma, which we prove at the end of the section:

Lemma C.7. *In Algorithm 1 let $k \in \mathbb{Z}_+$ and let $w_q, w_u, w_1, w_2 \in [0, 1]$ be the Gap-SS parameters such that $w_1 \geq w_q w_u$. Now let $t_q, t_u \in [0, 1]$ be the thresholds such that $t_q k$ and $t_u k$ are integers, and let $\Delta > 0$ be the branching factor. Given a query set X, with $|X| = w_q\,|U|$, and data set $Y \subseteq U$, with $|Y| = w_u\,|U|$, then running Algorithm 1 with $c_\ell = \begin{bmatrix} \sqrt{t_q(1-t_q)} \\ \sqrt{t_u(1-t_u)} \end{bmatrix} \cdot \sqrt{6.5\ell \log(3k)}$ for $\ell < k$ and $c_k = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, gives that*

$$\mathrm{E}[|R_k(X, t_q)|] \leq 2\Delta^k \exp(-k\,\mathrm{d}(t_q \parallel w_q))\,. \tag{C.2}$$

$$\mathrm{E}[|R_k(Y, t_u)|] \leq 2\Delta^k \exp(-k\,\mathrm{d}(t_u \parallel w_u))\,. \tag{C.3}$$

$$\Pr[|R_k(X, t_q) \cap R_k(Y, t_u)| \geq 1] \geq 7^{-8}k^{-14}\Delta^k \exp(-k\,\mathrm{D}(T_1 \parallel P_1)) \qquad \text{if } |X \cap Y| \geq w_1\,|U|\,. \tag{C.4}$$

$$\mathrm{E}[|R_k(X, t_q) \cap R_k(Y, t_u)|] \leq 2\Delta^k \exp(-k\,\mathrm{D}(T_2 \parallel P_2)) \qquad \text{if } |X \cap Y| \leq w_2\,|U|\,. \tag{C.5}$$

where $P_j = \begin{pmatrix} w_j & w_q - w_j \\ w_u - w_j & 1 - w_q - w_u + w_j \end{pmatrix}$, $T_j = \begin{pmatrix} t_j & t_q - t_j \\ t_u - t_j & 1 - t_q - t_u + t_j \end{pmatrix}$, $t_j = \arg\inf \mathrm{D}(T_j \parallel P_j)$ for $j \in \{1, 2\}$.

Finally the expected running times, $\mathcal{T}_q$ and $\mathcal{T}_u$, it takes to compute $R_k(X, t_q)$ and $R_k(Y, t_u)$ respectively are bounded by

$$\begin{aligned}\mathrm{E}[\mathcal{T}_q] &\leq O\big(k\,|X| + k(k+\log(n))\Delta^k \exp(-k\,\mathrm{d}(t_q \parallel w_q))\left(\tfrac{t_q(1-w_q)}{(1-t_q)w_q}\right)^{(c_k)_1}\big)\,.\\ \mathrm{E}[\mathcal{T}_u] &\leq O\big(k\,|Y| + k(k+\log(n))\Delta^k \exp(-k\,\mathrm{d}(t_u \parallel w_u))\left(\tfrac{t_u(1-w_u)}{(1-t_u)w_u}\right)^{(c_k)_2}\big)\,.\end{aligned} \tag{C.6}$$

We define $\Delta = \exp(\mathrm{D}(T_1 \parallel P_1))$, and let k be the smallest even integer at least $\frac{\log n}{\mathrm{D}(T_2\|P_2)-\mathrm{d}(t_q\|w_q)}$. Define the sequence $\Delta_i = 2^{l_i}$ for some $l_i \in \mathbb{Z}_{\geq 0}$ such that $\prod_{j=1}^{i}\Delta_j \leq \Delta^i < 2\prod_{j=1}^{i}\Delta_j$ for all $i \in [k]$.

We make 2 initiations of Algorithm 1, M_1, M_2, with height $k/2$. Δ and c_ℓ are adjusted correspondingly. In we have $c_{k/2} = c_k = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$.

For each instance we have

$$\begin{aligned}\mathrm{E}\big[\big|R_{k/2}(X,t_q)\big|\big] &\leq 2\exp(k/2\ \mathrm{D}(T_1 \parallel P_1) - k/2\ \mathrm{d}(t_q \parallel w_q))\\ &\leq 2\exp\left(\left(\frac{\log n}{\mathrm{D}(T_2 \parallel P_2) - \mathrm{d}(t_q \parallel w_q)} + 2\right)\frac{(\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q))}{2}\right)\\ &= 2n^{\frac{1}{2}\frac{\mathrm{D}(T_1\|P_1)-\mathrm{d}(t_q\|w_q)}{\mathrm{D}(T_2\|P_2)-\mathrm{d}(t_q\|w_q)}}(\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_q \parallel w_q)).\end{aligned}$$

similarly we get

$$\mathrm{E}\big[\big|R_{k/2}(X,t_q)\big|\big] \leq 2n^{\frac{1}{2}\frac{\mathrm{D}(T_1\|P_1)-\mathrm{d}(t_u\|w_u)}{\mathrm{D}(T_2\|P_2)-\mathrm{d}(t_q\|w_q)}}(\mathrm{D}(T_1 \parallel P_1) - \mathrm{d}(t_u \parallel w_u)).$$

We combine the two data instances M_1 and M_2 by taking as representative sets returned the product of the sets returned by each of them. In particular, this means we successfully find a near set, if $\big|R_{k/2}(X,t_q) \cap R_{k/2}(Y,t_u)\big| \geq 1$ for both instances, which happens with probability at least

$$(7^{-8}k^{-14}\Delta^k \exp(-k\,\mathrm{D}(T_1 \parallel P_1)))^2 = (7^{-8}k^{-14})^2.$$

hence, repeating the algorithm Ck^{28} times, for some C, we can boost this probability to 99%.

Putting it all together now yields the full version of Theorem C.2 contingent on Lemma C.7.

□

C.2.2 Bounds on Branching

It now remains to prove Lemma C.7. The inequalities (C.2), (C.3) and (C.5) are all simple calculations based on linearity of expectation. The time bound (C.6) is also fairly simple, but we have to take the decoding optimization described above into account. We also need to bound the number of paths alive at some point during the decoding process, which requires being more careful about the trimming conditions.

Finally the proof of the probability lower bound (C.4) is the main star of the section. We do this using essentially a second-moment method, but a number of tricks are needed

in order to squeeze out acceptable bounds, taking into account that any of w_q, w_u, w_1, w_2 may be $o(1)$, which among other things forbid the use of many Central Limit Theorem type results.

Proof of (C.2) *and* (C.3). We only provide the proof for (C.2) since the proof (C.3) is analogous.

Let $r \in R_k$ be a representative string and define the random variables $\mathcal{X}^{(i)} = [r_i \in X]$ for $i \in [k]$, because the hash functions h_i used at each level of the tree are independent, so are the $(\mathcal{X}^{(i)})_{i\in[k]}$ independent.

We use linearity of expectation, and completely throw away the fact that some branches may have been cut early. Throwing away extra cuts of course only increases the probability of survival. Meanwhile, we do not expect to gain more than factors of k this way, compared to a sharp analysis, since the whole point of the algorithm is to efficiently approximate cuts done only at the leaf level.

$$\begin{aligned}
\mathrm{E}[|R_k(X,t_q)|] &\le |R_k| \Pr\left[\forall \ell \le k : \sum_{i\in[\ell]} \mathcal{X}^{(i)} \ge t_q \ell - c_1^{(\ell)}\right] \\
&\le |R_k| \Pr\left[\sum_{i\in[k]} \mathcal{X}^{(i)} \ge t_q k\right] \\
&\le |R_k| \exp(-k\,\mathrm{d}(t_q \parallel w_q)).
\end{aligned}$$

The final bound is the entropy Chernoff bound we use everywhere. Since $|R_k| = \prod_{i=1}^{k} \Delta_i \le 2\Delta^k$ we get the bound. □

Proof of (C.5). This is similar to the proof of (C.2) and (C.3), but two dimensional. Like in the those proofs we consider a single representative string $r \in R_k$ and define the random variables $\mathcal{X}^{(i)} = \left[\begin{smallmatrix}[r_i\in X]\\ [r_i\in Y]\end{smallmatrix}\right]$ for $i \in [k]$. By definition of Algorithm 1 $(\mathcal{X}^{(i)})_{i\in[k]}$ are independent.

We then bound using linearity of expectation:

$$\begin{aligned}
\mathrm{E}[|R_k(X,t_q) \cap R_k(Y,t_u)|] &\le |R_k| \Pr\left[\forall \ell \le k : \sum_{i\in[\ell]} \mathcal{X}^{(i)} \ge \left[\begin{smallmatrix}t_q\\ t_u\end{smallmatrix}\right]\ell - c^{(\ell)}\right] \\
&\le 2\Delta^k \Pr\left[\sum_{i\in[k]} \mathcal{X}^{(i)} \ge \left[\begin{smallmatrix}t_q\\ t_u\end{smallmatrix}\right]k\right] \\
&\le 2\Delta^k \exp(-\mathrm{D}(T_2 \parallel P_2))
\end{aligned}$$

□

Proof of (C.6). As a preprocessing stage we make k sorted lists of $(a_i x)_{x\in X}$ where a_i is the coefficient in $h_i(p \circ x) = h'_i(p) + a_i x \mod q$, this takes $O(k\,|X|)$ time.

We will argue that at each level of tree that we only use $O(k+\log|X|) = O(k+\log n)$ amortized time per active path. More precisely, at level l we use $O((k+\log n)\,|R_\ell(X,t_q)|)$ amortized time.

Let $\ell \in [k]$ be fixed and consider an active path $r \in R_\ell(X,t_q)$. If $\sum_{i\in[\ell]}[r_i \in X] \geq t_q(l+1) - c_1^{(l-1)}$ then every one of its children will be active. So we need to find $\{x \in U \mid h_\ell(p\circ x) < \Delta_\ell\} = h_\ell^{-1}([\Delta_\ell])$. Now $h_\ell(p\circ x) = h'_\ell(p) + ax \mod q$ where $a \neq 0 \mod q$ and $s = h'_\ell(p)$ can be computed in $O(k)$ time. We then get that $h_\ell^{-1}([\Delta_\ell]) = \left\{a^{-1}(i-s) \,\middle|\, i \in [\Delta_\ell]\right\}$, this we can find in time proportional with the number of active children, so charging the cost to them gives the result.

If $\sum_{i\in[\ell]}[r_i \in X] < t_q(l+1) - c_1^{(l-1)}$ then only the children $r\circ x \in R_{l+1}$ where $x \in X$ will be active. So we need to find $\{x \in X \mid h_\ell(p\circ x) < \Delta_\ell\}$. Again using that $h_\ell(p\circ x) = h'_\ell(p) + ax \mod q$ where $a \neq 0 \mod q$ and $s = h'_\ell(p)$ can be computed in $O(k)$ time, we have reduced the problem to finding $h_\ell^{-1}([\Delta_\ell]) = \{x \in X \mid s + ax \mod q < \Delta_\ell\}$. This we note we can rewrite as $h_\ell^{-1}([\Delta_\ell]) = \{x \in X \mid s \leq ax \vee ax < \Delta + s - q\}$, so using our sorted list this can be done in $O(\log n)$ time plus time proportional with the number of active children, so charging this cost to them gives the result.

We bound the expected number of active paths on a level $\ell \in [k]$. Let $r \in R_\ell$ be a representative string and define the random variables $\mathcal{X}^{(i)} = [r_i \in X]$ for $i \in [k]$, by definition of Algorithm 1 $(\mathcal{X}^{(i)})_{i\in[k]}$ are independent. We then bound

$$\begin{aligned}
\Pr\left[\sum_{i\in[l]} \mathcal{X}^{(i)} \geq t_q\ell - c_1^{(\ell)}\right] &\leq \Pr\left[\forall j \leq \ell : \sum_{i\in[j]} \mathcal{X}^{(i)} \geq t_q j - c_1^{(j)}\right] \\
&\leq \Pr\left[\sum_{i\in[\ell]} \mathcal{X}^{(i)} \geq t_q\ell - c_1^{(\ell)}\right] \\
&\leq \exp(-l\,\mathrm{d}(t_q - c^{(\ell)}/l \parallel w_q)) \\
&\leq \exp(-l\,\mathrm{d}(t_q \parallel w_q))\left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{c_1^{(\ell)}} .
\end{aligned}$$

The crucial step here was using the identity

$$\mathrm{d}(t_q - \varepsilon \parallel w_q) = \mathrm{d}(t_q \parallel w_q) - \varepsilon \log \tfrac{t_q(1-w_q)}{w_q(1-t_q)} + \mathrm{d}(t_q - \varepsilon \parallel t_q)$$

from which we can ignore the $\mathrm{d}(t_q - \varepsilon \parallel t_q)$ term, since it is positive.

Using linearity of expectation we get that

$$\begin{aligned}
\mathrm{E}[|R_\ell(X,t_q)|] &\leq |R_\ell| \exp(-\ell\,\mathrm{d}(t_q \parallel w_q))\left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{c_1^{(\ell)}} \\
&\leq 2\Delta^\ell \exp(-\ell\,\mathrm{d}(t_q \parallel w_q))\left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{c_1^{(\ell)}} .
\end{aligned}$$

Now the expected cost of the tree becomes

$$\mathrm{E}\left[\sum_{\ell\in[k]} O((k+\log(n))\,|R_\ell(X,t_q)|)\right] = O((k+\log(n))\sum_{\ell\in[k]}\mathrm{E}[|R_\ell(X,t_q)|])$$
$$\leq O(k(k+\log(n))\Delta^k \exp(-k\,\mathrm{d}(t_q \parallel w_q))\left(\tfrac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{c_1^{(\ell)}}).$$

□

Note that we throw away some leverage here by bounding the size of each level by the final level. We might have defined $c^{(\ell)}$ such that $\ell\Delta - \ell\,\mathrm{d}(t_q \parallel w_q) + c^{(\ell)}\log\frac{t_q(1-w_q)}{w_q(1-t_q)} - \ell\,\mathrm{d}(t_q - c^{(\ell)}/\ell \parallel t_q) = k\Delta - k\,\mathrm{d}(t_q \parallel w_q)$ and still used the same bound. The only later requirement we set the $c^{(\ell)}$ is that $\sum_{\ell\in[k]}\exp(-\ell\,\mathrm{d}(t_q - c^{(\ell)}/\ell \parallel t_q))$ sum to $1/\operatorname{poly}(k)$.

Making this change could potentially kill the $\left(\frac{t_q(1-w_q)}{w_q(1-t_q)}\right)^{c_1^{(\ell)}}$ factor, which is a bit of an eye sore. However in the near-constant query time case, which is really when this factor (or term once we using the tensoring trick) is relevant, this trick wouldn't work, since we then have exactly $\Delta = \mathrm{d}(t_q \parallel w_q)$.

For the final proof we need the following lemma, which bounds the probability that an unbiased Bernoulli $2d$ random walk stays entirely in the negative quadrant. A lemma like this is an exercise to show using the Central Limit Theorem and convergence to Brownian motion. However, our bound is non-asymptotic, making no assumptions about the relationship between the probability distribution of X_i and the size of n. There are non-asymptotic CLT bounds, like Berry Esseen, but unfortunately multivariate Berry Esseen bounds for random walks are not very developed.

Lemma C.8 (The probability that a random walk stays in a quadrant)**.** *Let $X_1,\ldots,X_k \in \{0,1\}^2$ be iid. Bernoulli 2d-random variables with probability matrix $\begin{bmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p \end{bmatrix}$. Assume that the coordinates are correlated, that is $p \geq p_1p_2$, and assume p_qk and p_2k are integers.*

Let $S_\ell = \sum_{i\in[\ell]} X_i$ be the associated random walk. Then

$$\Pr[\forall \ell \in [k] : S_\ell \leq 0] \geq \frac{1}{400\,k^{6.5}}.$$

The proof of this is in Appendix C.2.3.

Proof of (C.4)*.* We will prove this bound using the second moment method. For this to work, it is critical that we restrict our representative strings further and consider

$$S = \left\{ r \in R_k \;\middle|\; \forall \ell \leq k : \left[\begin{smallmatrix}[r_i\in X]\\ [r_i\in Y]\end{smallmatrix}\right]\ell - c^{(\ell)} \leq \sum_{i\in[\ell]}\left[\begin{smallmatrix}[r_i\in X]\\ [r_i\in Y]\end{smallmatrix}\right] \leq [\begin{smallmatrix}t_q\\ t_u\end{smallmatrix}]\ell \right\},$$

It is easy to check that $S \subseteq R_k(X,t_q)\cap R_k(Y,t_u)$, thus we have that

$$\Pr[|R_k(X,t_q)\cap R_k(Y,t_u)| \geq 1] \geq \Pr[|S| \geq 1] \geq \mathrm{E}[|S|]^2 \,/\, \mathrm{E}\left[|S|^2\right],$$

where the last bound is Paley-Zygmund's inequality. We then need to do two things: 1) Lower bound $\mathrm{E}[|S|]$, and 2) Upper bound $\mathrm{E}\left[|S|^2\right]$.

Lower bounding $\mathrm{E}[|S|]$**.** Let $r \in R_k$ be a representative string and define the random variables $\mathcal{X}^{(i)} = \begin{bmatrix} [r_i \in X] \\ [r_i \in Y] \end{bmatrix}$ for $i \in [k]$. Each one has distribution $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_q & 1 - w_q - w_u + w_1 \end{bmatrix}$. We then introduce variables $\tilde{\mathcal{X}}^{(i)}$ with law $T = \begin{bmatrix} t_1 & t_q - t_1 \\ t_u - t_q & 1 - t_q - t_u + t_1 \end{bmatrix}$, where t_1 minimizes $\mathrm{D}(T \parallel P)$ as defined in the algorithm.

We then use the following variation on Sanov's theorem:

Lemma C.9. *For any set $A \subseteq \mathbb{R}^{2\times n}$ we have*

$$\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] = \exp(-k\,\mathrm{D}(T \parallel P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in A\right]$$

Proof. Define the logarithmic moment generating function $\Lambda(\lambda) = \log \mathrm{E}[\exp(\langle \lambda, \mathcal{X}\rangle)]$, and let $z = (\nabla_x \Lambda^*)(t)$. By a standard correspondence, (see e.g. [PW14] Chapter 14 or [Din94] Chapter 6.2), we have that

$$\mathrm{d}T(x) = \exp(\langle z, x\rangle - \Lambda(z))\mathrm{d}P(x) \tag{C.7}$$

for Radon–Nikodym derivates $\mathrm{d}T$ and $\mathrm{d}P$. Now using the exponential change of measure, we get that

$$\begin{aligned}
&\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] \\
&\qquad = \int_{(x^{(i)})_{i\in[k]}\in A} \mathrm{d}P^{\otimes k} \\
&\qquad = \int_{(\tilde{x}^{(i)})_{i\in[k]}\in A} \exp\left(k\Lambda(z) - \left\langle z, \sum_{i\in[k]} \tilde{x}^{(i)} \right\rangle\right) \mathrm{d}T^{\otimes k} \\
&\qquad = \exp(-k\,\mathrm{D}(T \parallel P)) \int_{(\tilde{x}^{(i)})_{i\in[k]}\in A} \exp\left(-\left\langle z, \sum_{i\in[k]} \left(\tilde{x}^{(i)} - \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]\right)\right\rangle\right) \mathrm{d}T^{\otimes k} \\
&\qquad = \exp(-k\,\mathrm{D}(T \parallel P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in A\right],
\end{aligned}$$

where the last inequality follows from the fact that if $(\tilde{x}^{(i)})_{i\in[k]} \in A$ then $\sum_{i\in[k]} \tilde{x}^{(i)} = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right] k$. □

For convenience we will sometimes write $T = \left[\begin{smallmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{smallmatrix}\right]$. Note that by assumption $t_q k = (t_{12} + t_{11})k$ and $t_u k = (t_{21} + t_{11})k$ are integers, but values such as $t_{11}k$ and $t_{22}k$ need not be.

We define the sets

$$U = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \;\middle|\; \forall \ell \le k : \sum_{i\in[\ell]} x^{(i)} \le [\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}]\ell \right\}$$

$$\text{and} \quad L = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \;\middle|\; \forall \ell \le k : \sum_{i\in[\ell]} x^{(i)} \ge [\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}]\ell - c^{(\ell)} \right\}$$

such that $U \cap L$ are all sequences satisfying our path requirement. In other words $E|S| = \exp(-k\,\mathrm{D}(T \parallel P)) \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap L\right]$. Using a union bound we split up:

$$\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap L\right] \ge \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U\right] - \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in L\right].$$

The term is bounded by Lemma C.8 from the Appendix. Once we notice that $w_1 \ge w_q w_u$ implies that $t_1 \ge t_q t_u$. One way to see this is that t_1 minimizing $\mathrm{D}(T \parallel P)$ gives rise to the equation $\frac{w_1(1-w_q-w_u+w_1)}{(w_q-w_1)(w_u-w_1)} = \frac{t_1(1-t_q-t_u+t_1)}{(t_q-t_1)(t_u-t_1)} = \frac{t_1+t_1(t_1-t_q-t_u)}{t_q t_u+t_1(t_1-t_q-t_u)}$. If $w_1 \ge w_q w_u$ the left hand side is ≥ 1, and so we must have $t_1 \ge t_q t_u$.

Lemma C.8 then gives us

$$\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U\right] \ge \frac{1}{400} k^{-3.5}.$$

This is a pretty small value, so for the union bound to work we need an even smaller probability for the lower bound.

We bound each coordinate individually. The cases are symmetric, so we only consider the first coordinate. Using another union bound and Bernstein's inequality we get

$$\begin{aligned}
\Pr\left[\exists \ell \le k : \sum_{i=1}^{\ell} \tilde{\mathcal{X}}_1^{(i)} \le t_q \ell - c_1^{(\ell)}\right] &= \sum_{l\le k} \Pr\left[\sum_{i=1}^{\ell} \tilde{\mathcal{X}}_1^{(i)} \le t_q \ell - c_1^{(\ell)}\right] \\
&\le \sum_{l\le k} \exp\left(\frac{-(c_1^{(\ell)})^2/2}{(1-t_q)t_q\ell + (1-2t_q)c_1^{(\ell)}/3}\right) \\
&\le \frac{1}{1200} k^{-6.5}.
\end{aligned}$$

since $c_1^{(\ell)} = \Omega(\sqrt{t_q(1-t_q) l \log l} + |1-2t_q| \log l)$.

Similarly, we upper bound $\Pr\left[\exists \ell \le k : \sum_{i=1}^{\ell} \tilde{\mathcal{X}}_2^{(i)} \le t_u \ell - c_2^{(\ell)}\right] \le \frac{1}{1200} k^{-6.5}$. Putting it all together we get

$$\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in A\right] \ge \frac{1}{1200} \exp(-k\,\mathrm{D}(T_1 \parallel P_1)) k^{-6.5},$$

so by linearity of expectation we get that

$$\mathrm{E}[S] \ge |R_k| \frac{1}{1200} k^{-6.5} \exp(-k\,\mathrm{D}(T \parallel P)) \ge \frac{1}{1200} k^{-6.5} \Delta^k \exp(-k\,\mathrm{D}(T \parallel P)).$$

Upper bounding $\mathrm{E}\left[|S|^2\right]$

Consider two representative strings $r, r' \in R_k$ and let $q \in R_\ell$ be their common prefix, hence l is the length of their common prefix. Define the random variables $\mathcal{X}^{(i)} = \left[\begin{smallmatrix}[r_i \in X] \\ [r_i \in Y]\end{smallmatrix}\right]$, $\mathcal{Y}^{(j)} = \left[\begin{smallmatrix}[r'_j \in X] \\ [r'_j \in Y]\end{smallmatrix}\right]$, and $\mathcal{Z}^{(h)} = \left[\begin{smallmatrix}[q_h \in X] \\ [q_h \in Y]\end{smallmatrix}\right]$ for $i, j \in [k] \setminus [\ell]$ and $h \in [l]$. We then get that

$$\begin{aligned}
&\Pr\left[p, p' \in S\right] \\
&\quad\leq \Pr\left[\sum_{h\in[\ell]} \mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]} \mathcal{X}^{(i)} \geq tk \wedge \sum_{h\in[\ell]} \mathcal{Z}^{(h)} + \sum_{j\in[k]\setminus[l]} \mathcal{Y}^{(j)} \geq tk \wedge \sum_{h\in[\ell]} \mathcal{Z}^{(h)} \leq tl\right] \\
&\quad\leq \Pr\left[\sum_{h\in[\ell]} \mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]} \mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]} \mathcal{Y}^{(j)} \geq (2k-\ell)t\right] .
\end{aligned}$$

Now $\sum_{h\in[\ell]} \mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]} \mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]} \mathcal{Y}^{(j)}$ is almost a sum of independent random variable. We have that $\mathcal{X}^{(k-\ell+1)}$ and $\mathcal{Y}^{(k-\ell+1)}$ are correlated since they are chosen by sampling without replacement, but this implies that

$$\mathrm{E}\left[\exp(\langle\lambda, \mathcal{X}^{(k-\ell+1)} + \mathcal{Y}^{(k-\ell+1)}\rangle)\right] \leq \mathrm{E}\left[\exp(\langle\lambda, \mathcal{X}^{(k-\ell+1)}\rangle)\right] \mathrm{E}\left[\exp(\langle\lambda, \mathcal{Y}^{(k-\ell+1)}\rangle)\right]$$

We can then use a 2-dimensional Entropy-Chernoff bound and get that

$$\Pr\left[\sum_{h\in[\ell]} \mathcal{Z}^{(h)} + \sum_{i\in[k]\setminus[l]} \mathcal{X}^{(i)} + \sum_{j\in[k]\setminus[l]} \mathcal{Y}^{(j)} \geq (2k-\ell)t\right] \leq \exp(-(2k-\ell)\,\mathrm{D}(T \parallel P)) ,$$

Using this we can upper bound $\mathrm{E}\left[|S|^2\right] = \mathrm{E}\left[\sum_{r,r'\in R_k} [r, r' \in S]\right]$ by splitting the sum by the length of their common prefix.

$$\begin{aligned}
\mathrm{E}\left[|S|^2\right] &= \mathrm{E}\left[\sum_{r,r'\in S_k} [r, r' \in S]\right] \\
&\leq \sum_{i=1}^{k} \left(\prod_{j=1}^{i} \Delta_j\right) \binom{\Delta_{i+1}}{2} \left(\prod_{j=i+2}^{k} \Delta_j\right) \exp(-(2k-i)\,\mathrm{D}(T \parallel P)) \\
&\leq \left(\prod_{j=1}^{k} \Delta_j\right)^2 \exp(-2k\,\mathrm{D}(T \parallel P)) \sum_{i=1}^{k} \exp(i\,\mathrm{D}(T \parallel P)) \left(\prod_{j=1}^{i} \Delta_j\right)^{-1} \\
&\leq \left(\prod_{j=1}^{k} \Delta_j\right)^2 \cdot \exp(-2k\,\mathrm{D}(T \parallel P)) \cdot k \cdot \exp(k\,\mathrm{D}(T \parallel P)) \cdot \Delta^{-k} \\
&\leq 4k\Delta^k \exp(-k\,\mathrm{D}(T \parallel P))
\end{aligned}$$

Finishing the proof

Having lower bounded $\mathrm{E}[|S|]$ and upper bounded $\mathrm{E}\left[|S|^2\right]$ we can finish the proof.

$$\begin{aligned}\Pr[|R_k(X,t_q)\cap R_k(Y,t_u)|\geq 1] &\geq \Pr[|S|\geq 1]\\ &\geq \mathrm{E}[|S|]^2 \,/\, \mathrm{E}\left[|S|^2\right]\\ &\geq \frac{\frac{1}{1200^2}k^{-13}\Delta^{2k}\exp(-2k\,\mathrm{D}(T\parallel P))}{4k\Delta^k\exp(-k\,\mathrm{D}(T\parallel P))}\\ &= \frac{1}{2400^2}k^{-14}\Delta^{\ell}\exp(-k\,\mathrm{D}(T\parallel P))\ .\end{aligned}$$

□

C.2.3 Central Random Walks

The main goal of this section is to prove Lemma C.8, which polynomially in k lower bounds the probability that a biased random walk on $\mathbb{Z}^2$ always stays below its means. Asymptotically, this can be done in various ways using the Central Limit Theorem for Brownian Motion, but as far as we know there are no standard ways to prove such a result in a quantitative way.

What we would really want is a Multidimensional Berry Esseen for Random Walks. Instead we prove something specifically for walks where each iid. step $X_1,\ldots,X_k\in\{0,1\}^2$ be is a Bernoulli $2d$-random variables with probability matrix $\left[\begin{smallmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p\end{smallmatrix}\right]$. We need the further restrictions that the coordinates are correlated ($p\geq p_1p_2$), and that p_1k and p_2k are integers.

We will start by proving some partial results, simply bounding the probability that the final position of the random walk hits a specific value. We then prove the lemma conditioned on hitting those values, and finally put it all together.

Lemma C.10. *Let $k\in\mathbb{Z}_+$ and $p_1,p_2\in[0,1]$, such that, both p_1k and p_2k are integers. Choose $p\in[0,1]$, such that, $p\geq p_1p_2$. Let $X^{(i)}\in\mathbb{R}^2$ be independent identically distributed 2-dimensional Bernoulli variables, where their probability matrix is $P=\left[\begin{smallmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p\end{smallmatrix}\right]$. We then get that*

$$\Pr\left[\sum_{i\in[k]}X^{(i)}=\tbinom{p_1}{p_2}k\wedge\sum_{i\in[k]}X_1^{(i)}X_2^{(i)}=\lceil pk\rceil\right]\geq\frac{1}{400}k^{-3.5}\ ,$$

In the proof we will be using the Stirling's approximation

$$\sqrt{2\pi n}n^ne^{-n}\leq n!\leq e\sqrt{n}n^ne^{-n}\ . \tag{C.8}$$

This implies the following useful bounds on the binomial and multinomial coefficients.

$$\binom{n}{an} \geq \frac{\sqrt{2\pi}}{e^2} n^{-0.5} a^{-an} (1-a)^{-(1-a)n} \,. \tag{C.9}$$

$$\binom{n}{an, bn, cn} \geq \frac{\sqrt{2\pi}}{e^4} n^{-1.5} a^{-an} b^{-bn} c^{-cn} (1-a-b-c)^{-(1-a-b-c)n} \,. \tag{C.10}$$

Proof. If $p_1 = 1$ then $p = p_2$ and we get that

$$\Pr\left[\sum_{i\in[k]} X_2^{(i)} = p_2 k\right] = \binom{k}{p_2 k} p_2^{p_2 k} (1-p_2)^{(1-p_2)k} \geq \frac{\sqrt{2\pi}}{e^2} k^{-\frac{1}{2}} \,,$$

where we have used eq. (C.9). We get the same bound when $p_1 = 0$, $p_2 = 1$, or $p_2 = 0$.

Now assume that $p_1, p_2 \notin \{0, 1\}$, we then have that $\frac{1}{k} \leq p_1 \leq 1 - \frac{1}{k}$ and $\frac{1}{k} \leq p_2 \leq 1 - \frac{1}{k}$. We first note that

$$\begin{aligned}
&\Pr\left[\sum_{i\in[k]} X^{(i)} = (p_1 k, p_2 k) \wedge \sum_{i\in[k]} X_1^{(i)} X_2^{(i)} = \lceil pk \rceil\right] \\
&= \binom{k}{\lceil pk \rceil, p_1 k - \lceil pk \rceil, p_2 k - \lceil pk \rceil} \\
&\quad p^{\lceil pk \rceil} (p_1 - p)^{p_1 k - \lceil pk \rceil} (p_2 - p)^{p_2 k - \lceil pk \rceil} (1 - p_1 - p_2 + p)^{k - p_1 k - p_2 k + \lceil pk \rceil} \\
&\geq \frac{\sqrt{2\pi}}{e^4} \cdot \frac{1}{k^{3/2}} \exp\Big(- \lceil pk \rceil \log \frac{\lceil pk \rceil}{pk} - (p_1 k - \lceil pk \rceil) \log \frac{p_1 k - \lceil pk \rceil}{p_1 k - pk} \\
&- (p_2 k - \lceil pk \rceil) \log \frac{p_2 k - \lceil pk \rceil}{p_2 k - pk} - (k - p_1 k - p_2 k + \lceil pk \rceil) \log \frac{k - p_1 k - p_2 k + \lceil pk \rceil}{k - p_1 k - p_2 k + pk} \Big)
\end{aligned}$$

where we have used eq. (C.10). We will bound each of the terms $\lceil pk \rceil \log \frac{\lceil pk \rceil}{pk}$, $(p_1 k - \lceil pk \rceil) \log \frac{p_1 k - \lceil pk \rceil}{p_1 k - pk}$, $(p_2 k - \lceil pk \rceil) \log \frac{p_2 k - \lceil pk \rceil}{p_2 k - pk}$, and $(k - p_1 k - p_1 k + \lceil pk \rceil) \log \frac{k - p_1 k - p_2 k + \lceil pk \rceil}{k - p_1 k - p_2 k + pk}$ individually.

Using that $p \geq p_1 p_2 \geq \frac{1}{k^2}$ we get that

$$\lceil pk \rceil \log \frac{\lceil pk \rceil}{pk} = (1 + pk) \log \left(1 + \frac{1}{pk}\right) \leq 1 + \log(1 + k) \,.$$

Now using that $1 - p_1 - p_2 + p \geq (1 - p_1)(1 - p_2) \geq \frac{1}{k^2}$ we get that

$$\begin{aligned}
&(k - p_1 k - p_1 k + \lceil pk \rceil) \log \frac{k - p_1 k - p_2 k + \lceil pk \rceil}{k - p_1 k - p_2 k + pk} \\
&\qquad \leq (k(1 - p_1 - p_2 + p) + 1) \log \left(1 + \frac{1}{k(1 - p_1 - p_2 + p)}\right) \\
&\qquad \leq 1 + \log(1 + k) \,.
\end{aligned}$$

We easily get that

$$(p_1 k - \lceil pk \rceil) \log \frac{p_1 k - \lceil pk \rceil}{p_1 k - pk} \leq (p_1 k - \lceil pk \rceil) \log \frac{p_1 k - pk}{p_1 k - pk} = 0 \, .$$

Similarly, we get that $(p_2 k - \lceil pk \rceil) \log \frac{p_2 k - \lceil pk \rceil}{p_2 k - pk} = 0$.

Combining all this we get that

$$\Pr\left[\sum_{i \in [k]} X^{(i)} = (p_1 k, p_2 k) \wedge \sum_{i \in [k]} X_1^{(i)} X_2^{(i)} = \lceil pk \rceil\right] \geq \frac{\sqrt{2\pi}}{e^4} k^{-1.5} \exp(-(1 + \log(1+k)) - (1 + \log(1+k))) \geq \frac{1}{400} k^{-3.5} \, .$$

□

We now prove a result for the random walk, conditioned on the final position. In the last result of this section, we will remove those restrictions.

Lemma C.11. *Let $k \in \mathbb{Z}_+$ and $p, p_1, p_2 \in [0,1]$, such that, $pk, p_1 k$, and $p_2 k$ are integers and $p \geq p_1 p_2$. Let $X^{(i)} \in \{0,1\}^2$ be independent identically distributed variables. We then get that*

$$\Pr\left[\forall l \leq k : \sum_{i \in [k]} X^{(i)} \geq \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} l \,\middle|\, \sum_{i \in [k]} X^{(i)} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} k \wedge \sum_{i \in [k]} X_1^{(i)} X_2^{(i)} = pk\right] \geq k^{-3} \, .$$

In the proof we will use the folklore result.

Lemma C.12. *Let $k \in Z_+$ and $(a_i)_{i \in [k]}$ numbers such that $\sum_{i \in [k]} a_i \geq 0$ then there exists a $s \in [k]$ such that $\sum_{i \in [l]} a_{(s+i) \mod k} \geq 0$ for every $l \leq k$.*

Proof of Lemma C.11. Using Lemma C.12 we get that $\sum_{i \in [l]} X_1^{(i)} \geq p_1 l$ for every $l \leq k$ with probability at least k^{-1} since every variable identically distributed. Fixing $(X_1^{(i)})_{i \in [k]}$ and using Lemma C.12 2 times we get that $\sum_{i \in [l]} X_1^{(i)} X_2^{(i)} \geq \frac{p}{p_1} \sum_{i \in [l]} X_1^{(i)}$ and $\sum_{i \in [l]} (1 - X_1^{(i)}) X^{(i)} \geq \frac{p_2 - p}{1 - p_1} \sum_{i \in [l]} X_1^{(i)}$ for every $l \leq k$ with probability at least k^{-2}. If all these three events happens then for every $l \leq k$ we get that

$$\begin{aligned}
\sum_{i \in [l]} X_2^{(i)} &= \sum_{i \in [l]} X_1^{(i)} X_2^{(i)} + \sum_{i \in [l]} (1 - X_1^{(i)}) X_2^{(i)} \\
&\geq \frac{p}{p_1} \sum_{i \in [l]} X_1^{(i)} + \frac{p_2 - p}{1 - p_1} \sum_{i \in [l]} X_1^{(i)} \\
&= \frac{p - p_1 p_2}{p_1 (1 - p_1)} \sum_{i \in [l]} X_1^{(i)} + \frac{p_2 - p}{1 - p_1} l \\
&\geq \frac{p - p_1 p_2}{p_1 (1 - p_1)} p_1 l + \frac{p_2 - p}{1 - p_1} l \\
&= p_2 l \, .
\end{aligned}$$

So we conclude that with probability at least k^{-3} then $\sum_{i\in[l]} X^{(i)} \geq \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} l$ for every $l \leq k$ which finishes the proof. □

All that remains is proving Lemma C.8. We restate it and then prove it.

Lemma C.13. *Let $X_1, \ldots, X_k \in \{0,1\}^2$ be iid. Bernoulli 2d-random variables with probability matrix $\begin{bmatrix} p & p_1-p \\ p_2-p & 1-p_1-p_2+p \end{bmatrix}$. Assume that the coordinates are correlated, that is $p \geq p_1 p_2$, and assume $p_q k$ and $p_2 k$ are integers.*

Let $S_\ell = \sum_{i\in[\ell]} X_i$ be the associated random walk. Then

$$\Pr[\forall \ell \in [k] : S_\ell \leq 0] \geq \frac{1}{400\, k^{6.5}}.$$

Proof. We define the set $U = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \forall \ell \leq k : \sum_{i\in[\ell]} x^{(i)} \leq [\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}]\ell \right\}$ of all sequences satisfying our path requirement. In other words $\Pr[\forall k \in [n] : S_k \leq 0] = \Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in U\right]$. We then add even more restrictions by defining

$$A' = \left\{ (x^{(i)})_{i\in[k]} \in \mathbb{R}^{2\times k} \,\middle|\, \sum_{i\in[k]} x^{(i)} = [\begin{smallmatrix} p_1 \\ p_2 \end{smallmatrix}]k \wedge \sum_{i\in[k]} (1-x_1^{(i)})(1-x_2^{(i)}) = \lceil p_{22} k \rceil \right\}. \quad \text{(C.11)}$$

That is, we require the last final value of the path to completely match its expectation, rounded up. By monotonicity we have $\Pr\left[(\mathcal{X}^{(i)})_{i\in[k]} \in U\right] \geq \Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap A'\right]$.

We want to use Lemma C.10 and Lemma C.11 and to ease the notation we introduce the negated random variables $\mathcal{Y}^{(i)} = 1 - \tilde{\mathcal{X}}^{(i)}$. Define $p_{22} = 1 - p_1 - p_2 + p$. We then have that $\mathrm{E}\left[\mathcal{Y}^{(i)}\right] = [\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}]$ and $\Pr\left[\mathcal{Y}^{(i)} = (\begin{smallmatrix} 1 \\ 1 \end{smallmatrix})\right] = p_{22} = 1 - p_1 - p_2 + p \geq (1-p_1)(1-p_2)$ by the assumption of correlation.

We can then rewrite using $\mathcal{Y}^{(i)}$:

$$\begin{aligned} &\Pr\left[(\tilde{\mathcal{X}}^{(i)})_{i\in[k]} \in U \cap A'\right] \\ &\quad = \Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq [\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}]\ell \wedge \sum_{i\in[k]} \mathcal{Y}^{(i)} = [\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}]k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \end{aligned}$$

Now using Lemma C.10 we have that

$$\Pr\left[\sum_{i\in[k]} \mathcal{Y}^{(i)} = [\begin{smallmatrix} 1-p_1 \\ 1-p_2 \end{smallmatrix}]k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)} \mathcal{Y}_2^{(i)} = \lceil p_{22} k \rceil \right] \geq \frac{1}{400} k^{-3.5}.$$

Combining this with Lemma C.11 we get that

$$\begin{aligned}
&\Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq [\tfrac{1-p_1}{1-p_2}]\ell \wedge \sum_{i\in[k]} \mathcal{Y}^{(i)} = [\tfrac{1-p_1}{1-p_2}]k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)}\mathcal{Y}_2^{(i)} = \lceil p_{22}k \rceil\right] \\
&\quad = \Pr\left[\sum_{i\in[k]} \mathcal{Y}^{(i)} = [\tfrac{1-p_1}{1-p_2}]k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)}\mathcal{Y}_2^{(i)} = \lceil p_{22}k \rceil\right] \\
&\qquad \cdot \Pr\left[\forall \ell \leq k : \sum_{i\in[k]} \mathcal{Y}^{(i)} \geq [\tfrac{1-p_1}{1-p_2}]\ell \,\middle|\, \sum_{i\in[k]} \mathcal{Y}^{(i)} = [\tfrac{1-p_1}{1-p_2}]k \wedge \sum_{i\in[k]} \mathcal{Y}_1^{(i)}\mathcal{Y}_2^{(i)} = \lceil p_{22}k \rceil\right] \\
&\quad \geq \frac{1}{400}k^{-6.5} .
\end{aligned}$$

□

C.3 Lower Bounds

Our lower bounds all assume that $w_2 d = \omega(\log n)$, where d is the size of the universe. As discussed in the introduction is both standard and necessary.

We proceed to define the hard distributions for all further lower bounds.

1. A query $x \in \{0,1\}^d$ is created by sampling d random independent bits with Bernoulli(w_q) distribution.

2. A dataset $P \subseteq \{0,1\}^d$ is constructed by sampling $n-1$ vectors with random independent bits from such that $y_i \sim$ Bernoulli(w_2/w_q) if $x_i = 1$ and $y_i \sim$ Bernoulli($(w_u - w_2)/(1-w_q)$) otherwise, for all $y \in P$.

3. A 'close point', $y' \in \{0,1\}^d$, is created by $y'_i \sim$ Bernoulli(w_1/w_q) if $x_i = 1$ and $y'_i \sim$ Bernoulli($(w_u - w_1)/(1-w_q)$) otherwise. This point is also added to P.

The values are chosen such that $\mathrm{E}[|x|] = w_q d$, $\mathrm{E}[|z|] = w_u d$ for all $z \in P$, $\mathrm{E}[|x \cap y'|] = w_1 d$, and $\mathrm{E}[|x \cap y|] = w_2 d$ for all $y \in P \setminus \{y'\}$. By a union bound over P, the actual values are within factors $1 + o(1)$ of their expectations with high probability. Changing at most $o(\log n)$ coordinates we ensure the weights of queries/database points is exactly their expected value, while only changing the inner products by factors $1 + o(1)$. Since the changes do not contain any new information, we can assume for lower bounds that entries are independent. Thus any $(w_q, w_u, w_1(1-o(1)), w_2(1+o(1)))$-GapSS data structure on P must thus be able to return y' with at least constant probability when given the query x.

Model Our lower bounds are shown in slightly different models. The first lower bound follows the framework of O'Donnell et al. [OWZ14] and Christiani [Chr17] and directly

lower bound the quantity $\frac{\log(p_1/\min\{p_u,p_q\})}{\log(p_2/\min\{p_u,p_q\})}$ which lower bounds ρ_u and ρ_q in Definition C.28. This lower bound holds for all $w_2 \neq w_q w_u$, i.e., it gives a lower bound when we are not considering a random instance, and it only gives a lower bound in the case where $\rho_q = \rho_u$.

For the second lower bound we follow the framework of Andoni et al. [ALRW17b] and give a lower bound in the "list-of-points"-model (see Definition C.3). This is a slightly more general model, though it is believed that all bounds for the first model can be shown in the list-of-points model as well. Our lower bound shows that our upper bound is tight in the full time/space trade-off when $w_2 = w_q w_u$, i.e., when we are given a random instance.

The second bound can be extended to show cell probe lower bounds by the arguments in [PTW08].

C.3.1 p-biased Analysis

We first give some preliminaries on b-biased Boolean analysis, and then introduce the directed noise operator.

Preliminaries

We want analyse Boolean functions $f : \{0,1\}^d \to \{0,1\}$ but as is common, it turns out to be beneficial to consider a more general class of functions $f : \{0,1\}^d \to \mathbb{R}$.

The probability distribution π_p is defined on $\{0,1\}$ by $\pi_p(1) = p$ and $\pi_p(0) = 1 - p$, and we define $\pi_p^{\otimes d}$ to be the product probability distribution on $\{0,1\}^d$. We write $L_2(\{0,1\}^d, \pi_p^{\otimes d})$ for the inner product space of functions $f : \{0,1\}^d \to \mathbb{R}$ with inner product

$$\langle f, g \rangle_p = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}[f(x)g(x)] \ .$$

We will define the norm $\|f\|_{L_q(p)} = \left(\mathrm{E}_{x \sim \pi_p^{\otimes d}}[f(x)^q]\right)^{1/q}$.

We define the p-biased Fourier coefficients for a function $f : L_2(\{0,1\}^d, \pi_p^{\otimes d})$ by

$$\hat{f}^{(p)}(S) = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}\left[f(x)\phi_S^{(p)}(x)\right] \ ,$$

for every $S \subseteq [d]$ and where we define

$$\phi^{(p)}(x) = \frac{x - p}{\sqrt{p(1-p)}} \qquad \phi_S^{(p)}(x) = \prod_{i \in S} \phi^{(p)}(x_i) \ .$$

The Fourier coefficients have the nice property that

$$f(x) = \sum_{S \subseteq [d]} \hat{f}^{(p)}(S)\phi_S^{(p)}(x) \ . \tag{C.12}$$

The Fourier coefficients satisfy the Parseval-Plancherel identity, which says that for any $f, g \in L_2(\{0,1\}^d, \pi_p^d)$ we have that

$$\langle f, g \rangle_p = \sum_{S \subseteq [d]} \hat{f}^{(p)}(S)\hat{g}^{(p)}(S) \ .$$

In particular we have that $\mathrm{E}_{x \sim \pi_p^{\otimes d}}\left[f(x)^2\right] = \|f\|^2_{L_2(p)} = \sum_{S \subseteq [d]} \hat{f}^{(p)}(S)^2$. For Boolean functions $f : \{0,1\}^d \to \{0,1\}$ this is particularly useful since we get that

$$\Pr_{x \sim \pi_p^{\otimes d}}[f(x) = 1] = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}[f(x)] = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}\left[\sum_{S \subseteq [n]} \hat{f}^{(p)}(S)\phi_S^{(p)}(x)\right] = \hat{f}^{(p)}(\emptyset)$$
$$\Pr_{x \sim \pi_p^{\otimes d}}[f(x) = 1] = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}[f(x)] = \mathop{\mathrm{E}}_{x \sim \pi_p^{\otimes d}}\left[f(x)^2\right] = \sum_{S \subseteq [d]} \hat{f}^{(p)}(S)^2.$$

If we think of f as a filter in a Locality Sensitive data structure, $\Pr_{x \sim \pi_p^{\otimes d}}[f(x) = 1]$ is the probability that the filter accepts a random point with expected weight p ($d \cdot p$ of the coordinates being 1).

Noise

For $\rho \in [-1,1]$, $p_1, p_2 \in (0,1)$, and $x \in \{0,1\}^d$ we write $y \sim N_\rho^{p_1 \to p_2}(x)$ when $y \in \{0,1\}^d$ is randomly chosen such that for each $i \in [d]$ independently, we have that if $x_i \sim \pi_{p_2}$ then $y_i \sim \pi_{p_1}$ and (x_i, y_i) are ρ-correlated. We note that if $x \sim \pi_{p_2}^{\otimes d}$ and $y \sim N_\rho^{p_1 \to p_2}$ then we also have that $y \sim \pi_{p_1}^{\otimes d}$ and $x \sim N_\rho^{p_2 \to p_1}(y)$.

For $\rho \in [-1,1]$ and $p_1, p_2 \in (0,1)$ we define the *directed noise operator* $T_\rho^{p_1 \to p_2} : L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d}) \to L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ by

$$T_\rho^{p_1 \to p_2} f(x) = \mathop{\mathrm{E}}_{y \sim N_\rho^{p_1 \to p_2}}[f(y)] \ .$$

When $p_1 = p_2 = p$ then $T_\rho^{p \to p}$ is the *usual noise operator* on p-biased spaces and we denote it $T_\rho^{(p)}$. $T_\rho^{(p)}$ has the nice property that $\widehat{T_\rho^{(p)} f}^{(p)}(S) = \rho^{|S|}\hat{f}^{(p)}(S)$ for any $S \subseteq [d]$, and hence $T_\rho^{(p)}$ satisfies the semigroup property $T_\rho^{(p)} T_\sigma^{(p)} = T_{\rho\sigma}^{(p)}$. The following lemma shows that we have similar properties for $T_\rho^{p_1 \to p_2}$.

Lemma C.13. *For $\rho \in [-1,1]$, $p_1, p_2 \in (0,1)$ and $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes})$ we have that*

$$\widehat{T_\rho^{p_1 \to p_2}} f^{(p_2)}(S) = \rho^{|S|}\hat{f}^{(p_1)}(S) \ ,$$

for any $S \subseteq [d]$. Furthermore, for any $\sigma \in [-1,1]$ and $p_3 \in [0,1]$ we have that $T_\sigma^{p_2 \to p_3} T_\rho^{p_1 \to p_2} = T_{\rho\sigma}^{p_1 \to p_3}$ and $T_\rho^{p_2 \to p_1}$ is the adjoint of $T_\rho^{p_1 \to p_2}$.

Proof. We fix $S \subseteq [d]$ and get that

$$\begin{aligned}
\widehat{T_\rho^{p_1\to p_2} f}^{(p_2)}(S) &= \mathop{\mathrm{E}}_{x\sim\pi_{p_2}^{\otimes d}}\left[T_\rho^{p_1\to p_2} f(x)\phi_S^{(p_2)}(x)\right] \\
&= \mathop{\mathrm{E}}_{x\sim\pi_{p_2}^{\otimes d}}\left[\mathop{\mathrm{E}}_{y\sim N_\rho^{p_1\to p_2}(x)}[f(y)]\,\phi_S^{(p_2))}(x)\right] \\
&= \mathop{\mathrm{E}}_{x\sim\pi_{p_2}^{\otimes d}}\left[\mathop{\mathrm{E}}_{y\sim N_\rho^{p_1\to p_2}(x)}\left[\sum_{T\subseteq[d]}\widehat{f}^{(p_1)}(T)\phi_T^{(p_1)}(y)\right]\phi_S^{(p_2)}(x)\right] \\
&= \mathop{\mathrm{E}}_{x\sim\pi_{p_2}^{\otimes d}}\left[\mathop{\mathrm{E}}_{y\sim N_\rho^{p_1\to p_2}(x)}\left[\widehat{f}^{(p_1)}(S)\phi_S^{(p_1)}(y)\phi_S^{(p_2)}(x)\right]\right] \\
&= \widehat{f}^{(p_1)}(S)\prod_{i\in S}\mathop{\mathrm{E}}_{x_i\sim\pi_{p_2}}\left[\mathop{\mathrm{E}}_{y_i\sim N_\rho^{p_1\to p_2}}\left[\phi_i^{(p_1)}(y_i)\phi_i^{(p_2)}(x_i)\right]\right] \\
&= \rho^{|S|}\widehat{f}^{(p_1)}(S)\,,
\end{aligned}$$

where the last line uses that $\phi_i^{(p)}(x) = \frac{x-p}{\sqrt{p(1-p)}}$, which proves the first claim. For the second claim we note that

$$(T_\sigma^{p_2\to p_3}\widehat{T_\rho^{p_1\to p_2}} f)^{(p_3)}(S) = \sigma^{|S|}\widehat{T_\rho^{p_1\to p_2}} f^{(p_2)}(S) = (\rho\sigma)^{|S|}\widehat{f}^{(p_1)}(S) = \widehat{T_{\rho\sigma}^{p_1\to p_2}} f^{(p_3)}(S)\,,$$

for any $f \in f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes})$ and any $S \subseteq [d]$ which proves the second claim. For the last claim we use the Plancherel-Parseval identity and get that

$$\langle T_\rho^{p_1\to p_2} f, g\rangle_{L_2(p_2)} = \sum_{S\in[d]}\rho^{|S|}\widehat{f}^{(p_1)}\widehat{g}^{(p_2)} = \langle f, T_\rho^{p_2\to p_1} g\rangle_{L_2(p_1)}\,,$$

for any $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ and any $g \in L_2(\{0,1\}^d, \pi_{p_2}^{\otimes d})$ which shows that $T_\rho^{p_2\to p_1}$ is the adjoint of $T_\rho^{p_1\to p_2}$. □

We say that $(T_\rho^{p_1\to p_2})_{\rho>0}$ is (s,r)-hypercontractive if there exists $\rho_0 > 0$ such that for every $\rho \geq \rho_0$ and every $f \in L_r(\{0,1\}^d, \pi_{p_1}^d)$

$$\left\|T_\rho^{p_1\to p_2} f\right\|_{L_s(p_2)} \leq \|f\|_{L_r(p_1)}\,.$$

We define $\sigma_{s,r}(p_1,p_2)$ to be the smallest possible ρ_0 We are interested in the hypercontractivity of $T^{p_1\to p_2}$

C.3.2 Symmetric Lower bound

The most general, but sadly least tractable, approach to our lower bounds, is to bound the noise operator T_α in terms of a different level of noise, T_β. We do however manage to show one bound on this type, following an spectral approach first used by O'Donnell et

al. [OWZ14] to prove the first optimal LSH lower bounds of $\rho \geq 1/c$ for data-independent hashing. Besides handling the case of set similarity with filters rather than hash functions, we slightly generalize the approach a big by using the power-means inequality rather than log-concavity. [16]

We will show the following inequality

$$\left(\frac{\Pr_{x,y',f}[f(x)=1, f(y')=1]}{\Pr_{x,f}[f(x)=1]}\right)^{1/\log\alpha} \leq \left(\log\frac{\Pr_{x,y,f}[f(x)=1, f(y)=1]}{\Pr_{x,f}[f(x)=1]}\right)^{1/\log\beta}$$

where $\alpha = \frac{w_1-w^2}{w(1-w)}$ and $\beta = \frac{w_2-w^2}{w(1-w)}$, and y' and y are sampled as respectively a close and a far point (see the top of the section). By rearrangement, this directly implies a lower bound in the LSF model as defined in Definition C.28.

First we prove a general lemma about Boolean functions, which contains the most important arguments.

Lemma C.14. *Let $f : \{0,1\}^n \to \mathbb{R}$ be a function and $p \in (0,1)$. Then for any $1 > \alpha \geq \beta > 0$ we have that*

$$\left(\frac{\langle T_\alpha^{(p)} f, f\rangle_{L_2(p)}}{\|f\|^2_{L_2(p)}}\right)^{1/\log(1/\alpha)} \leq \left(\frac{\langle T_\beta^{(p)} f, f\rangle_{L_2(p)}}{\|f\|^2_{L_2(p)}}\right)^{1/\log(1/\beta)}.$$

Proof. We use the Parseval-Plancherel identity and the power-mean inequality to get that

$$\begin{aligned}
\left(\frac{\langle T_\alpha^{(p)} f, f\rangle_{L_2(p)}}{\|f\|^2_{L_2(p)}}\right)^{1/\log(1/\alpha)} &= \left(\frac{\sum_{S\subseteq[n]} \alpha^{|S|}\hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]}\hat{f}^{(p)}(S)^2}\right)^{1/\log(1/\alpha)} \\
&= \left(\sum_{k=0}^{n}\frac{\sum_{\substack{S\subseteq[n]\\|S|=k}}\hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]}\hat{f}^{(p)}(S)^2}\left(e^{-k}\right)^{\log(1/\alpha)}\right)^{1/\log(1/\alpha)} \\
&\leq \left(\sum_{k=0}^{n}\frac{\sum_{\substack{S\subseteq[n]\\|S|=k}}\hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]}\hat{f}^{(p)}(S)^2}\left(e^{-k}\right)^{\log(1/\beta)}\right)^{1/\log(1/\beta)} \\
&= \left(\frac{\sum_{S\subseteq[n]} \beta^{|S|}\hat{f}^{(p)}(S)^2}{\sum_{S\subseteq[n]}\hat{f}^{(p)}(S)^2}\right)^{1/\log(1/\beta)} \\
&= \left(\frac{\langle T_\beta^{(p)} f, f\rangle_{L_2(p)}}{\|f\|^2_{L_2(p)}}\right)^{1/\log(1/\beta)}.
\end{aligned}$$

The first and the last equality follows from the Parseval-Plancherel identity and the inequality follows from the power-mean inequality since $\log(1/\alpha) \leq \log(1/\beta)$. □

[16] This widens the range in which the bound is applicable – the O'Donnell bound is only asymptotic for $r \to 0$. However the values we obtain outside this range, when applied to Hamming space LSH, aren't sharp against the upper bounds.

The proof of Theorem C.4 is then simply a few a rearrangements such that we can use Lemma C.14.

Corollary C.15. *Any data-independent LSF data structure for the (w, w, w_1, w_2)-GapSS problem with expected query time n^{ρ_q} and expected space usage $n^{1+\rho_u}$ where $\rho_q = \rho_u = \rho$ must have*

$$\rho \geq \log\left(\frac{w_1 - w^2}{w(1-w)}\right) \Big/ \log\left(\frac{w_2 - w^2}{w(1-w)}\right) .$$

Proof. Let $\mathcal{F}$ be any fixed LSF-family and let $f : \{0,1\}^n \to \{0,1\}$ be a random function such that $f^{-1}(1) = Q$ for $Q \sim \mathcal{F}$. Now we define the deterministic function $\overline{f} : \{0,1\}^n \to \mathbb{R}$ by $\overline{f}(x) = \sum_{S \subseteq [d]} \sqrt{\mathrm{E}_f\left[\hat{f}^{(w)}(S)^2\right]}\phi_S(x)$. Using the Parseval-Plancherel identity we get that

$$\mathrm{E}_f\left[\langle T_\rho^{(w)} f, f\rangle_{L_2(w)}\right] = \sum_{S \subseteq [d]} \rho^{|S|} \mathrm{E}_f\left[\hat{f}^{(w)}(S)^2\right] = \langle T_\rho^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)} .$$

for every ρ. We set $\alpha = \frac{w_1 - w^2}{w(1-w)}$ and $\beta = \frac{w_2 - w^2}{w(1-w)}$ and note that $\Pr_{x,y',f}[f(x) = 1, f(y') = 1] = \mathrm{E}_f\left[\langle T_\alpha^{(w)} f, f\rangle_{L_2(w)}\right]$, $\Pr_{x,y,f}[f(x) = 1, f(y) = 1] = \mathrm{E}_f\left[\langle T_\beta^{(w)} f, f\rangle_{L_2(w)}\right]$, and $\Pr_{x,f}[f(x) = 1] = \mathrm{E}_f\left[\|f\|_{L_2(w)}^2\right]$. Then using Lemma C.14 we get that

$$\begin{aligned}
\left(\frac{\Pr_{x,y',f}[f(x) = 1, f(y') = 1]}{\Pr_{x,f}[f(x) = 1]}\right)^{1/\log 1/\alpha} &= \left(\frac{\mathrm{E}_f\left[\langle T_\alpha^{(w)} f, f\rangle_{L_2(w)}\right]}{\mathrm{E}_f\left[\|f\|_{L_2(w)}^2\right]}\right)^{1/\log 1/\alpha} \\
&= \left(\frac{\langle T_\alpha^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)}}{\|\overline{f}\|_{L_2(w)}^2}\right)^{1/\log 1/\alpha} \\
&\leq \left(\frac{\langle T_\beta^{(w)} \overline{f}, \overline{f}\rangle_{L_2(w)}}{\|\overline{f}\|_{L_2(w)}^2}\right)^{1/\log 1/\beta} \\
&= \left(\frac{\Pr_{x,y,f}[f(x) = 1, f(y) = 1]}{\Pr_{x,f}[f(x) = 1]}\right)^{1/\log 1/\beta} .
\end{aligned}$$

By rearrangement this implies that

$$\rho = \frac{\log \frac{\Pr_{x,y',f}[f(x)=1, f(y')=1]}{\Pr_{x,f}[f(x)=1]}}{\log \frac{\Pr_{x,y,f}[f(x)=1, f(y)=1]}{\Pr_{x,f}[f(x)=1]}} \geq \frac{\log \alpha}{\log \beta} .$$

□

As noted the bound is sharp against our upper bound when w_u, w_q, w_1, w_2 are all small. Also notice that $\log \alpha / \log \beta \leq \frac{1-\alpha}{1+\alpha}\frac{1-\beta}{1+\beta}$ is a rather good approximation for α and β close to 1. Here the right hand side is the ρ value of Spherical LSH with the batch-normalization embedding discussed in Appendix C.4.1.

Note that the lower bound becomes 0 when we get close to the random instance, $w_2 \to w_q w_u$. In the next sections we will remedy this, by showing a lower bound tight exactly when $w_2 = w_q w_u$.

C.3.3 General Lower Bound

Our second lower bound will be proven in the "list-of-points" model. We follow and expand upon the approach by Andoni et al. [ALRW17b]. The main idea is to lower bound random instances with planted points. If the random instances correspond to a Similarity Search problem with high probability then we have a lower bound for the Similarity Search problem. We formalize the notion of random instances in the following general definition.

Definition C.16 (Random instance)**.** For spaces Q and U we describe a distribution of dataset-query pairs (P, q) where $P \subseteq U$ and $q \in Q$. Let $\mathcal{P}_{QU}$ be a probability distribution on $Q \times U$, a $\mathcal{P}_{QU}$-random instance is a dataset-query pair drawn from the following distribution.

1. A dataset $P \subseteq U$ is constructed by sampling n points where $p \sim \mathcal{P}_U$ for all $p \in P$.
2. A dataset point $p' \in P$ is fixed and a $q \in Q$ is sampled such that $(q, p') \sim \mathcal{P}_{QU}$.
3. The goal of the data structure is to pre-process P such that it recovers p' when given the query point q.

We can then generalize the result by Andoni et al. [ALRW17b], who proved a result specifically for random Hamming instances, to general random instances. We defer the proof to Appendix C.I.

Lemma C.17. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Let $r, s \in [1, \infty]$ satisfy*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{QU}}[f(X)g(Y)] \leq \|f(X)\|_{L_r(\mathcal{P}_Q)} \|g(Y)\|_{L_s(\mathcal{P}_U)} ,$$

for all functions $f : Q \to \mathbb{R}$ and $g : U \to \mathbb{R}$. Then

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{1}{r} + \frac{1}{s} - 1 ,$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r.

This gives a good way to lower bound random instances when one has tight hypercontractive inequalities. Unfortunately, for most probability distributions this is not the case but we can amplify the power of Lemma C.17 by combining it with Lemma C.18 which we recall from the introduction.

Lemma C.18. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be the marginal distributions on the spaces Ω_X and Ω_Y respectively. Let $s, r \in [1, \infty)$, then the following is equivalent*

1. *For all functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$ we have*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \leq \|f(X)\|_{L_r(\mathcal{P}_X)} \|g(Y)\|_{L_s(\mathcal{P}_Y)} . \tag{C.13}$$

2. *For all probability distributions $\mathcal{Q}_{XY}$ which are absolutely continuous with respect to $\mathcal{P}_{XY}$ we have*

$$\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s} . \tag{C.14}$$

We defer the proof to the end of the section and instead start by focusing on the effects of combining Lemma C.17 and Lemma C.18. First of all we can prove the following general lower bound for random instances.

Theorem C.18. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $r \in [1, \infty]$ we have that*

$$\begin{aligned}&\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \\ &\quad\geq \inf_{\mathcal{Q}_{QU}} \left(\frac{1}{r} \frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \parallel \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)} + \frac{1}{r'} \frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)} \right),\end{aligned}$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r and the infimum is over every probability distribution $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$.

Proof. Let $r \in [1, \infty]$ and choose $s = \arg\inf \{s \in [1, \infty] \mid \mathcal{P}_{QU}$ is (r, s)-hypercontractive$\}$. Lemma C.17 give us that

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{1}{r} + \frac{1}{s} - 1 . \tag{C.15}$$

Lemma C.18 give us that

$$\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s}$$

for every $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$. We can rewrite this as

$$\begin{aligned}
&\frac{1}{r}\,\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \parallel \mathcal{P}_Q) + \frac{1}{r'}\,\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U) \\
&\qquad \geq \left(\frac{1}{s} - \frac{1}{r'}\right)\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U) \qquad\qquad \Leftrightarrow \\
&\frac{1}{r}\frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \parallel \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)} + \frac{1}{r'}\frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)} \\
&\qquad \geq \frac{1}{s} - \frac{1}{r'} = \frac{1}{s} + \frac{1}{r} - 1
\end{aligned}$$

Now the minimality of s give us that

$$\begin{aligned}
\inf_{\mathcal{Q}_{QU}} \left(\frac{1}{r}\frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_Q \parallel \mathcal{P}_Q)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)} + \frac{1}{r'}\frac{\mathrm{D}(\mathcal{Q}_{QU} \parallel \mathcal{P}_{QU}) - \mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)}{\mathrm{D}(\mathcal{Q}_U \parallel \mathcal{P}_U)}\right) \\
= \frac{1}{s} + \frac{1}{r} - 1,
\end{aligned} \tag{C.16}$$

where the infimum is over every probability distribution $\mathcal{Q}_{QU}$ with $\mathcal{Q}_U \neq \mathcal{P}_U$ and which is absolutely continuous with respect to $\mathcal{P}_{QU}$. Now combining (C.15) and (C.16) give us the result. □

Combining the lemma with the "Hypercontractive Induction Theorem" [ODo14] we can prove Theorem C.5.

Lemma C.19. *Let $\mathcal{P}_{XY}$ be a probability distribution on a space $\Omega_X \times \Omega_Y$ and $\mathcal{P}_{XY}^{\otimes n}$ be a probability distribution consisting n independent copies of $\mathcal{P}_{XY}$. Then $\mathcal{P}_{XY}$ is (r,s)-hypercontractive if and only if $\mathcal{P}_{XY}^{\otimes n}$ is (r,s)-hypercontractive.*

We restate Theorem C.5 and prove it.

Theorem C.5. *Consider any list-of-point data structure for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem over a universe of size d of n points with $w_q w_u d = \omega(\log n)$, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Then for every $\alpha \in [0,1]$ we have that*

$$\alpha\rho_q + (1-\alpha)\rho_u \geq \inf_{\substack{t_q, t_u \in [0,1] \\ t_u \neq w_u}} \left(\alpha\frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_q \parallel w_q)}{\mathrm{d}(t_u \parallel w_u)} + (1-\alpha)\frac{\mathrm{D}(T \parallel P) - \mathrm{d}(t_u \parallel w_u)}{\mathrm{d}(t_u \parallel w_u)}\right),$$

where $P = \begin{bmatrix} w_1 & w_q - w_1 \\ w_u - w_1 & 1 - w_q - w_u + w_1 \end{bmatrix}$ *and* $T = \operatorname*{arg\,inf}_{T \ll P,\ \mathop{E}_{X \sim T}[X] = \left[\begin{smallmatrix} t_q \\ t_u \end{smallmatrix}\right]} \mathrm{D}(T \parallel P)$.

Proof. From the discussion at the beginning of the section it is enough to lower bound the $P^{\otimes d}$-random instance where $P = \text{Bernoulli}(\left[\begin{smallmatrix} w_1 & w_q - w_1 \\ w_u & 1 - w_q - w_u + w_1 \end{smallmatrix}\right])$, since this will imply a lower bound for the $(w_q, w_u, w_1, w_q w_u)$-GapSS problem. Combining Lemma C.19 and

Lemma C.18 we get that $P^{\otimes d}$ is (r,s)-hypercontractive if and only if $\mathrm{D}(T \parallel P) \geq \frac{\mathrm{d}(t_q \| w_q)}{r} + \frac{\mathrm{d}(t_u \| w_u)}{s}$ where $T = \operatorname*{arg\,inf}_{T \ll P, \mathop{E}_{X \sim T}[X] = [\substack{t_q \\ t_u}]} \mathrm{D}(T \parallel P)$. Now repeating the proof of Theorem C.18 give us the result. □

Proof of Lemma C.18 We now turn to the proof Lemma C.18. The main argument needed in the proof of is contained in the following lemma, which can be seen as a variation of Fenchel's inequality.

Lemma C.20. *Let P be a probability distribution on a space Ω, Q a probability which is absolutely continuous with respect to P, and $\phi : \Omega \to \mathbb{R}$ a function such that $\mathrm{E}_{X \sim P}[\exp(\phi(X))] < \infty$. Then*

$$\mathrm{D}(Q \parallel P) + \log \mathop{\mathrm{E}}_{X \sim P}[\exp(\phi(X))] \geq \mathop{\mathrm{E}}_{X \sim Q}[\phi(X)] \ .$$

and we have equality if and only if $\frac{dQ}{dP}(x) = \frac{\exp(\phi(x))}{\mathrm{E}_{X \sim P}[\exp(\phi(X))]}$.

Proof. We note that

$$\begin{aligned}
\mathrm{D}(Q \parallel P) &= \mathop{\mathrm{E}}_{X \sim Q}\left[\log \frac{dQ}{dP}(X)\right] \\
&= \mathop{\mathrm{E}}_{X \sim Q}\left[\log \frac{\frac{dQ}{dP}(X)}{\exp(\phi(X))}\right] + \mathop{\mathrm{E}}_{X \sim Q}[\phi(X)] \\
&= \mathop{\mathrm{E}}_{X \sim Q}[\phi(X)] - \mathop{\mathrm{E}}_{X \sim Q}\left[\log \frac{\exp(\phi(X))}{\frac{dQ}{dP}(X)}\right] \ .
\end{aligned}$$

Using Jensen's inequality we get that

$$\mathop{\mathrm{E}}_{X \sim Q}\left[\log \frac{\exp(\phi(X))}{\frac{dQ}{dP}(X)}\right] \leq \log \mathop{\mathrm{E}}_{X \sim Q}\left[\exp(\phi(X))\frac{dP}{dQ}(X)\right] = \log \mathop{\mathrm{E}}_{X \sim P}[\exp(\phi(X))] \ .$$

Combining these two equations give us the inequality. Now we note that we have equality if and only if $e^{\phi(x)}\frac{dP}{dQ}(x)$ is constant, and since Q is a probability distribution this is equivalent with $\frac{dQ}{dP}(x) = \frac{\exp(\phi(x))}{\mathrm{E}_{X \sim P}[\exp(\phi(X))]}$. □

We are now ready to prove Lemma C.18.

Proof of Lemma C.18. First we prove that (C.13) ⇒ (C.14). Let $\mathcal{Q}_{XY}$ be a probability distribution which is absolutely continuous with respect to $\mathcal{P}_{XY}$. We set $\exp(\phi_X(x)) = \frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x)$ and $\exp(\phi_Y(y)) = \frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y)$. From this we see that $\mathrm{E}_{X \sim \mathcal{P}_X}[\exp(\phi_X(X))] = \mathrm{E}_{X \sim \mathcal{P}_x}\left[\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(X)\right] = \mathrm{E}_{X \sim \mathcal{Q}_X}[1] = 1$ and similarly that $\mathrm{E}_{Y \sim \mathcal{P}_Y}[\exp(\phi_Y(Y))] = 1$, hence

we have that $\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x) = \frac{\exp(\phi_X(x))}{\mathrm{E}_{X\sim\mathcal{P}_X}[\exp(\phi_X(X))]}$ and $\frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y) = \frac{\exp(\phi_Y(y))}{\mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(\phi_Y(Y))]}$. Using (C.13) we get that

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \\
&\qquad \le \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]^{1/s} \qquad \Leftrightarrow \\
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \\
&\qquad \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]}{s} \,.
\end{aligned}$$

Using Lemma C.20 3 times we have that

$$\begin{aligned}
\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y))] &\ge \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}}[\phi_X(X)+\phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \\
\log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[\exp(\phi_X(X))] &= \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X}[\phi_X(X)] - \mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X) \\
\log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y}[\exp(\phi_Y(Y))] &= \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y}[\phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y) \,,
\end{aligned}$$

where the equalities hold since $\frac{d\mathcal{Q}_X}{d\mathcal{P}_X}(x) = \frac{\exp(\phi_X(x))}{\mathrm{E}_{X\sim\mathcal{P}_X}[\exp(\phi_X(X))]}$ and $\frac{d\mathcal{Q}_Y}{d\mathcal{P}_Y}(y) = \frac{\exp(\phi_Y(y))}{\mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(\phi_X(Y))]}$. We then get that $\log \mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]}{s}$ implies, that

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}}[\phi_X(X)+\phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \\
&\qquad \le \frac{\mathrm{E}_{X\sim\mathcal{Q}_X}[r\phi_X(X)] - \mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{E}_{Y\sim\mathcal{Q}_Y}[s\phi_Y(Y)] - \mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s} \,.
\end{aligned}$$

Now by rearrangement this is equivalent with

$$\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \ge \frac{\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s} \,,$$

which proves that (C.13) ⇒ (C.14).

We now prove that (C.14) ⇒ (C.13). Fix the functions $f : \Omega_X \to \mathbb{R}$ and $g : \Omega_Y \to \mathbb{R}$. We note that $\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[f(X)g(Y)] \le \mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[|f|(X)\,|g|(Y)]$ hence we can assume that f and g are non-negative. We define $\phi_X(x) = \log(f(x))$ and $\phi_Y(x) = \log(g(x))$[17]. Then (C.13) is equivalent with

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \\
&\qquad \le \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]^{1/r} \mathop{\mathrm{E}}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]^{1/s} \qquad \Leftrightarrow \\
&\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \\
&\qquad \le \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]}{r} + \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]}{s} \,.
\end{aligned}$$

[17] We define $\log(0) = -\infty$ and $\exp(-\infty) = 0$.

We define the probability distribution $\mathcal{Q}_{XY}$ by $\frac{d\mathcal{Q}_{XY}}{d\mathcal{P}_{XY}}(x,y) = \frac{\exp(\phi_X(X)+\phi_Y(Y)}{\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)]}$. It is easy to see that $\mathcal{Q}_{XY}$ is indeed a probability distribution. Using (C.14) we get that

$$\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y)}{s} .$$

Using Lemma C.20 3 times we have that

$$\begin{aligned}
\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) &= \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}}[\phi_X(X)+\phi_Y(Y)] - \log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y))] \\
\mathrm{D}(\mathcal{Q}_X \parallel \mathcal{P}_X) &\geq \mathop{\mathrm{E}}_{X\sim\mathcal{Q}_X}[r\phi_X(X)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))] \\
\mathrm{D}(\mathcal{Q}_Y \parallel \mathcal{P}_Y) &\geq \mathop{\mathrm{E}}_{Y\sim\mathcal{Q}_Y}[s\phi_Y(Y)] - \log \mathop{\mathrm{E}}_{X\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y))] ,
\end{aligned}$$

where the equality holds since $\frac{d\mathcal{Q}_{XY}}{d\mathcal{P}_{XY}}(x,y) = \frac{\exp(\phi_X(X)+\phi_Y(Y)}{\mathrm{E}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)]}$. We then get that $\mathrm{D}(\mathcal{Q}_{XY} \parallel \mathcal{P}_{XY}) \geq \frac{\mathrm{D}(\mathcal{Q}_X\|\mathcal{P}_X)}{r} + \frac{\mathrm{D}(\mathcal{Q}_Y\|\mathcal{P}_Y)}{s}$ implies, that

$$\begin{aligned}
&\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{Q}_{XY}}[\phi_X(X)+\phi_Y(Y)] - \log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y))] \\
&\quad\geq \frac{\mathrm{E}_{X\sim\mathcal{Q}_X}[r\phi_X(X)] - \log \mathrm{E}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]}{r} \\
&\qquad + \frac{\mathrm{E}_{Y\sim\mathcal{Q}_Y}[s\phi_Y(Y)] - \log \mathrm{E}_{X\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y))]}{s} .
\end{aligned}$$

Now if we rearrange we get that

$$\begin{aligned}
\log \mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{XY}}[\exp(\phi_X(X)+\phi_Y(Y)] \leq \frac{\log \mathrm{E}_{X\sim\mathcal{P}_X}[\exp(r\phi_X(X))]}{r} \\
+ \frac{\log \mathrm{E}_{Y\sim\mathcal{P}_Y}[\exp(s\phi_Y(Y)]}{s} ,
\end{aligned}$$

which proves that (C.14) ⇒ (C.13). □

C.3.4 Explicit Hypercontractive Bounds

In this section we show how to relate the directed noise operator to the lower bounds of Oleszkiewicz [Ole03], thereby giving direct lower bounds for a number of cases for s and r. By Theorem C.5 and Lemma C.17 this is the dual to proving optimal values (t_q, t_u) in our upper bound.

We start by with a standard lemma which shows that hypercontractivity of an operator implies hypercontractivity of its adjoint.

Lemma C.21. *Let $T : L_2(\Omega,\pi) \to L_2(\Omega,\pi')$ be an operator with $T^* : L_2(\Omega,\pi') \to L_2(\Omega,\pi)$ being its adjoint, and let $1 \leq r,s < \infty$ with r',s' being their convex conjugates. Then*

$$\|Tf\|_{L_{s'}(\pi')} \leq \|f\|_{L_r(\pi)}$$

holds for all $f \in L_2(\Omega, \pi)$*, if and only if*

$$\langle Tf, g\rangle_{L_2(\pi')} = \langle f, T^*g\rangle_{L_2(\pi)} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')}$$

holds for all $f \in L_2(\Omega, \pi)$ *and all* $g \in L_2(\Omega, \pi')$*, if and only if*

$$\|T^*g\|_{L_{r'}(\pi)} \leq \|g\|_{L_s(\pi')}$$

holds for all $g \in L_2(\Omega, \pi')$*.*

Proof. We assume that $\|Tf\|_{L_{s'}(\pi')} \leq \|f\|_{L_r(\pi)}$ holds for all $f \in L_2(\Omega, \pi)$. Let $f \in L_2(\Omega, \pi)$ and $g \in L_2(\Omega, \pi')$ then by Hölder's inequality we have that

$$\langle Tf, g\rangle_{L_2(\pi')} \leq \|Tf\|_{L_{s'}(\pi')} \|g\|_{L_s(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')} \ .$$

Similarly, we assume that $\|T^*g\|_{L_{r'}(\pi)} \leq \|g\|_{L_s(\pi')}$ holds for all $g \in L_2(\Omega, \pi')$. Let $f \in L_2(\Omega, \pi)$ and $g \in L_2(\Omega, \pi')$ then by Hölder's inequality we have that

$$\langle f, T^*g\rangle_{L_2(\pi)} \leq \|f\|_{L_r(\pi')} \|T^*g\|_{L_{r'}(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_{s'}(\pi')} \ .$$

Finally, we assume that $\langle Tf, g\rangle_{L_2(\pi')} \leq \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')}$ holds for all $f \in L_2(\Omega, \pi)$ and all $g \in L_2(\Omega, \pi')$. Let $f \in L_2(\Omega, \pi)$ then using that $L_s(\pi')$ is the dual norm of $L_{s'}(\pi')$ we get that

$$\|Tf\|_{L_{s'}(\pi')} = \sup_{\|g\|_{Ls(\pi')}=1} \langle Tf, g\rangle_{L_2(\pi')} \leq \sup_{\|g\|_{Ls(\pi')}=1} \|f\|_{L_s(\pi)} \|g\|_{L_r(\pi')} = \|f\|_{L_r(\pi)} \ .$$

Similarly, let $g \in L_2(\Omega, \pi')$ then using that $L_r(\pi)$ is the dual norm of $L_{r'}(\pi)$ we get that

$$\|T^*g\|_{L_{r'}(\pi)} = \sup_{\|f\|_{Lr(\pi)}=1} \langle f, T^*g\rangle_{L_2(\pi)} \leq \sup_{\|f\|_{Lr(\pi')}=1} \|f\|_{L_r(\pi)} \|g\|_{L_s(\pi')} = \|g\|_{L_s(\pi')} \ ,$$

which finishes the proof. □

Our hypercontractive results will be based on the tight hypercontractive inequality by Oleszkiewicz [Ole03].

Theorem C.22 ([Ole03])**.** *Let* $p \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ *and* $1 \leq r \leq 2$ *then for any function* $f \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$ *we have that*

$$\left\| T_\rho^{(p)} f \right\|_{L_2(p)} \leq \|f\|_{L_r(p)} \ ,$$

where $\rho = p^{-1/2}(1-p)^{-1/2}\sqrt{\frac{(1-p)^{2-2/r} - p^{2-2/r}}{p^{-2/r} - (1-p)^{-2/r}}}$ *which is best possible.*

From this we get following tight hypercontractive inequalities for $T_\rho^{p_1 \to p_2}$.

Corollary C.23. *Let $p_1, p_2 \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \leq r \leq 2$ then for any function $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ we have that*

$$\left\| T_\rho^{p_1 \to p_2} f \right\|_{L_2(p_2)} \leq \|f\|_{L_r(p_1)} \ ,$$

where $\rho = p_1^{-1/2}(1-p_1)^{-1/2}\sqrt{\frac{(1-p_1)^{2-2/r} - p_1^{2-2/r}}{p_1^{-2/r} - (1-p_1)^{-2/r}}}$ which is best possible.

Proof. Using the Parseval-Plancherel identity and Lemma C.13 we get that

$$\left\| T_\rho^{p_1 \to p_2} f \right\|_{L_2(p_2)}^2 = \sum_{S \subseteq [d]} \widehat{T_\rho^{p_1 \to p_2}} f^{(p_2)}(S)^2 = \sum_{S \subseteq [d]} \rho^{2|S|} \widehat{T_\rho^{p_1 \to p_2}} f^{(p_2)}(S)^2 = \left\| T^{(p_1)} f \right\|_{L_2(p_1)}^2 \ ,$$

hence the result follows from Theorem C.22. □

Corollary C.24. *Let $p_1, p_2 \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \leq s \leq 2$ with s' the convex conjugate of s, then for any function $f \in L_2(\{0,1\}^d, \pi_{p_1}^{\otimes d})$ we have that*

$$\left\| T_\rho^{p_1 \to p_2} f \right\|_{L_{s'}(p_2)} \leq \|f\|_{L_2(p_1)} \ ,$$

where $\rho = p_2^{-1/2}(1-p_2)^{-1/2}\sqrt{\frac{(1-p_2)^{2-2/s} - p_2^{2-2/s}}{p_2^{-2/s} - (1-p_2)^{-2/s}}}$ which is best possible.

Proof. By Lemma C.21 we get that the result is true if and only if

$$\left\| T_\rho^{p_2 \to p_1} g \right\|_{L_2(p_1)} \leq \|g\|_{L_s(p_2)} \ ,$$

for all functions $g \in L_2(\{0,1\}, \pi_{p_2})$. Now the result follows by using Corollary C.23. □

We also get a hypercontractive inequality for the standard noise operator $T_\rho^{(p)}$.

Corollary C.25. *Let $p \in (0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ and $1 \leq r \leq 2$ with convex conjugate r', then for any function $f \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$ we have that*

$$\left\| T_\rho^{(p)} f \right\|_{L_{r'}(p)} \leq \|f\|_{L_r(p)} \ ,$$

where $\rho = p(1-p)\frac{(1-p)^{2-2/r} - p^{2-2/r}}{p^{-2/r} - (1-p)^{-2/r}}$ which is best possible.

Proof. Using Lemma C.21 we get the result holds if and only if

$$\langle T_\rho^{(p)} f, g \rangle_{L_2(p)} \leq \|f\|_{L_r(p)} \|g\|_{L_r(p)} \ ,$$

holds for all $f, g \in L_2(\{0,1\}^d, \pi_p^{\otimes d})$. First we note that the result is true by using Cauchy-Schwartz and Theorem C.22

$$\langle T_\rho^{(p)} f, g \rangle_{L_2(p)} = \langle T_{\sqrt{\rho}}^{(p)} f, T_{\sqrt{\rho}}^{(p)} g \rangle_{L_2(p)} \leq \left\| T_{\sqrt{\rho}}^{(p)} f \right\|_{L_2(p)} \left\| T_{\sqrt{\rho}}^{(p)} g \right\|_{L_2(p)} \leq \|f\|_{L_r(p)} \|f\|_{L_r(p)}$$

Now to see that ρ is best possible we set $g = f$ which give us that

$$\left\| T^{(p)}_{\sqrt{\rho}} f \right\|^2_{L_2(p)} = \langle T^{(p)}_{\rho} f, f \rangle_{L_2(p)} \leq \|f\|^2_{L_r(p)} \ ,$$

so Theorem C.22 gives that ρ is best possible. □

We will use Corollary C.25 to show that setting $(t_q, t_u) = (1-w, 1-w)$ is an optimal threshold for the (w, w, w_1, w^2)-GapSS problem. First of we note that $\rho = p(1-p)\frac{(1-p)^{2-2/r} - p^{2-2/r}}{p^{-2/r} - (1-p)^{-2/r}}$ can be rewritten as $r = \frac{2\log\tau}{\log\frac{\rho+\tau}{\rho+\tau^{-1}}}$ where $\tau = \frac{1-p}{p}$. Using Lemma C.17 we get that $\rho = \frac{w_1 - w^2}{w(1-w)}$, $\tau = \frac{1-w}{w}$, and that

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{2}{r} - 1 \ .$$

Now we have that $\rho_q = \rho_u = \frac{\log\frac{w(w_1-2w+1)}{w_1(1-w)}}{\log\frac{w}{1-w}}$, and we find that

$$\frac{2}{r} - 1 = \frac{\log\frac{\rho+\tau}{\rho+\tau^{-1}}}{\log\tau} - 1 = \frac{\log\tau^{-1}\frac{\rho+\tau}{\rho+\tau^{-1}}}{\log\tau} \ .$$

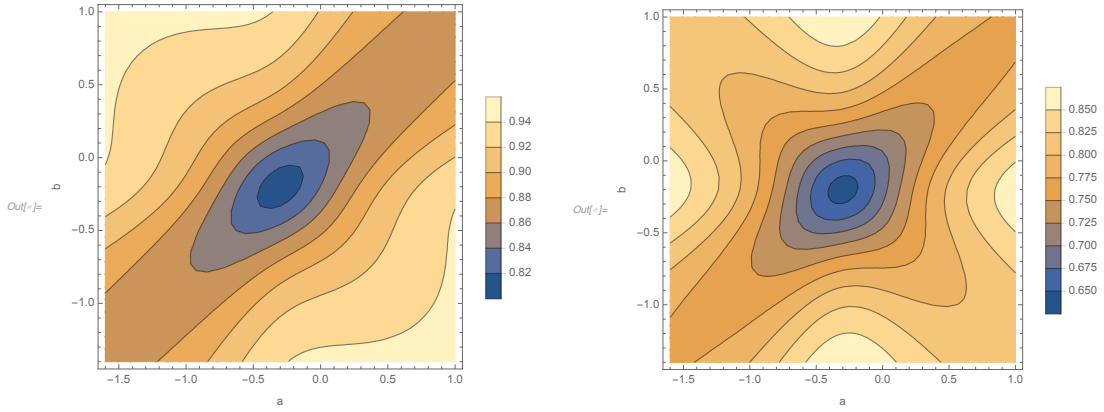
It is then easy to check that $\tau^{-1}\frac{\rho+\tau}{\rho+\tau^{-1}} = \frac{w_1(1-w)}{w(w_1-2w+1)}$, which then shows that $(t_q, t_u) = (1-w, 1-w)$ is an optimal threshold for the (w, w, w_1, w^2)-GapSS problem.

C.4 Other Algorithms

We show two results that, while orthogonal to Supermajorities, help us understand them and how they fit within the space of Similarity Search algorithms.

The first result is an optimal affine embedding of sets onto the sphere. This result is interesting in its own right, as it results in an algorithm that is in many cases better than the state of the art, and which can be implemented very easily in systems that can already solve Euclidean or Spherical Nearest Neighbours. The result gives a simple, general condition a Spherical LSH scheme must meet for the embedding to be optimal, and we show that both SimHash and Spherical LSH meets it.

The second result is also a new algorithm. In particular, it is a mix between Chosen Path and MinHash, which always achieves ρ values lower than both of them. It is in a sense a simple answer to the open problem in [CP17] about how to beat MinHash consistently. More interesting though, is that it sheds light on what makes Supermajorities work: It balances the amount of information pulled from sets vs. their complements. The proof is also conceptually interesting, since it proves that it is never advantageous to combine multiple Locality Sensitive Filter families.



(a) Query time/space exponent, ρ, for the SimHash algorithm [CIP02].

(b) Query time/space exponent, ρ, for the Spherical LSH algorithm [TT07].

Figure C.6: Given a GapSS instance with $w_q = .3$ and $w_u = .2$, the optimal affine embedding of the data (represented as vectors $x \in \{0,1\}^{|U|}$) onto the sphere, turns out to be normalizing the "mean" and "variance". That is, before scaling down to $\|x\|_2 = 1$, we subtract respectively w_q and w_u from all coordinates. The plot shows the "ρ-value" achieved by different spherical algorithms as the among subtracted is varied: The x-axis, a, is the amount subtracted from queries and the y-axis, b, is the amount subtracted from datasets.

C.4.1 Embedding onto the Sphere

We show, that if an algorithm has exponent $\rho(\alpha, \beta) = f(\alpha)/f(\beta)$ where α is the cosine similarity between good points and β is the similarity between bad points on the sphere; then assuming some light properties on f, which contain both Spherical and Hyperplane LSH, two affine embedding of sets $x \in \{0,1\}^d$ to S^{d-1} that minimizes ρ once the new cosine similarities are calculated, is $x \mapsto (x - w)/\sqrt{w(1-w)}$ where $w = |x|/d$. While the mapping is allowed to depend on any of the GapSS parameters, it curiously only cares about the weight of the set itself. For fairness, all our plots, such as Figure C.2, uses this embedding when comparing Supermajorities to Spherical LSH.

Lemma C.26 (Embedding Lemma)**.** *Let $g, h : \{0,1\}^d \to \mathbb{R}^d$ be function on the form $g(x) = a_1 x + b_1$ and $h(y) = a_2 y + b_2$. Let $\rho(x, y, y') = f(\alpha(x, y))/f(\alpha(x, y'))$ where $\alpha(x, y) = \langle x, y\rangle / \|x\|_2 \|y\|_2$ be such that*

$$f(z) \geq 0, \quad \tfrac{d}{dz}\left((\pm 1 - z)\tfrac{d}{dz}\log f(z)\right) \geq 0 \quad \text{and} \quad \tfrac{d^3}{dz^3}\log f(z) \leq 0$$

for all $z \in [-1, 1]$. Assume we know that $\|x\|_2^2 = w_q d$, $\|y\|_2^2 = w_u d$, $\langle x, y'\rangle = w_1 d$ and $\langle x, y\rangle = w_2 d$, then $\arg\min_{a_1,a_2,b_1,b_2} \rho(g(x), h(y), h(y')) = (1, 1, -w_q, -w_u)$.

In this section we will show that Hyperplane [Cha02] and Spherical [ALRW17b] LSH

both satisfy the requirements of the lemma. Hence we get two algorithms with ρ-values:

$$\rho_{\text{hp}} = \frac{\log(1 - \arccos(\alpha)/\pi)}{\log(1 - \arccos(\beta)/\pi)}, \quad \rho_{\text{sp}} = \frac{1-\alpha}{1+\alpha}\frac{1+\beta}{1-\beta}.$$

where $\alpha = \frac{w_1 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$ and $\beta = \frac{w_2 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}$, and space/time trade-offs using the ρ_q, ρ_u values in [Chr17]. [18] Figure C.6 shows how ρ varies with different translations a, b.

Taking $t_q = w_q(1+o(1))$ and $t_u = w_u(1+o(1))$ in theorem C.2 recovers ρ_{sp} by standard arguments. This implies that theorem C.2 dominates Spherical LSH (for binary data).

Lemma C.27. *The functions $f(z) = (1-z)/(1+z)$ for Spherical LSH and $f(z) = -\log(1 - \arccos(z)/\pi)$ for Hyperplane LSH satisfy lemma C.26.*

Proof. For Spherical LSH we have $f(z) = (1-z)/(1+z)$ and get

$$\frac{d}{dz}\left((\pm 1 - z)\frac{d}{dz}\log f(z)\right) = 2/(1 \pm z)^2 \geq 0,$$
$$\frac{d^3}{dz^3}\log f(z) = -4(1+3z^2)/(1-z^2)^3 \leq 0.$$

For Hyperplane LSH we have $f(z) = -\log(1 - \arccos(z)/\pi)$ and get

$$\frac{d}{dz}\left((\pm 1 - z)\frac{d}{dz}\log f(z)\right) = \frac{(\arccos(z) \mp \sqrt{1-z^2} - \pi)\log(1 - \arccos(z)/\pi) \mp \sqrt{1-z^2}}{(1 \pm z)\sqrt{1-z^2}(\pi - \arccos(z))^2 \log(1 - \arccos(z)/\pi)^2}.$$

In both cases the denominator is positive, and the numerator can be shown to be likewise by applying the inequalities $\sqrt{1-z^2} \leq \arccos(z)$, $\sqrt{1-z^2} + \arccos(z) \leq \pi$ and $x \leq \log(1+x)$.

The $\frac{d^3}{dz^3}\log f(z) \leq 0$ requirement is a bit trickier, but a numerical optimization shows that it's in fact less than -1.53. □

Finally we prove the embedding lemma:

Proof of lemma C.26. We have

$$\alpha = \frac{\langle x + a, y + b\rangle}{\|x + a\|_2 \|y + b\|_2} = \frac{w_1 + w_q b + w_u a + ab}{\sqrt{(w_q(1+a)^2 + (1-w_q)a^2)(w_u(1+b)^2 + (1-w_u)b^2)}}$$

and equivalent with w_2 for β. We'd like to show that $a = -w_q$, $b = -w_u$ is a minimum for $\rho = f(\alpha)/f(\beta)$.

Unfortunately the f's we are interested in are usually not convex, so it is not even clear that there is just one minimum. To proceed, we make the following substitution $a \to (c+d)\sqrt{w_q(1-w_q)} - w_q$, $b \to (c-d)\sqrt{w_u(1-w_u)} - w_u$ to get

$$\alpha(c,d) = \frac{cd + \frac{w_1 - w_q w_u}{\sqrt{w_q(1-w_q)w_u(1-w_u)}}}{\sqrt{(1+c^2)(1+d^2)}}.$$

[18] Unfortunately the space/time aren't on a form applicable to lemma C.26. From numerical experiments we however still conjecture that the embedding is optimal for those as well.

We can further substitute $cd \mapsto rs$ and $\sqrt{(1+c^2)(1+d^2)} \mapsto r+1$ or $r \geq 0$, $-1 \leq s \leq 1$, since $1 + cd \leq \sqrt{(1+c^2)(1+d^2)}$ by Cauchy Schwartz, and $(cd, \sqrt{(1+c^2)(1+d^2)})$ can take all values in this region.

The goal is now to show that $h = f\left(\frac{rs+x}{r+1}\right) / f\left(\frac{rs+y}{r+1}\right)$, where $1 \geq x \geq y \geq -1$, is increasing in r. This will imply that the optimal value for c and d is 0, which further implies that $a = -w_q$, $b = -w_u$ for the lemma.

We first show that h is quasi-concave in s, so we may limit ourselves to $s = \pm 1$. Note that $\log h = \log f\left(\frac{rs+x}{r+1}\right) - \log f\left(\frac{rs+y}{r+1}\right)$, and that $\frac{d^2}{ds^2} \log f\left(\frac{rs+x}{r+1}\right) = \left(\frac{r}{1+r}\right)^2 \frac{d^2}{dz^2} \log f(z)$ by the chain rule. Hence it follows from the assumptions that h is log-concave, which implies quasi-concavity as needed.

We now consider $s = \pm 1$ to be a constant. We need to show that $\frac{d}{dr} h \geq 0$. Calculating,

$$\frac{d}{dr} f\left(\frac{rs+x}{r+1}\right) / f\left(\frac{rs+y}{r+1}\right) = \frac{(s-x) f\left(\frac{rs+y}{r+1}\right) f'\left(\frac{rs+x}{r+1}\right) - (s-y) f\left(\frac{rs+x}{r+1}\right) f'\left(\frac{rs+y}{r+1}\right)}{(1+r)^2 f\left(\frac{rs+y}{r+1}\right)^2}.$$

Since $f \geq 0$ it suffices to show $\frac{d}{dx}(s-x) f'\left(\frac{rs+x}{r+1}\right) / f\left(\frac{rs+x}{r+1}\right) \geq 0$. If we substitute $z = \frac{rs+x}{r+1}$, $z \in [-1, 1]$, we can write the requirement as $\frac{d}{dz}(s-z) f'(z)/f(z) \geq 0$ or $\frac{d}{dz}\left((\pm 1 - z) \frac{d}{dz} \log f(z)\right) \geq 0$. □

C.4.2 A MinHash Dominating Family

Consider the classical MinHash scheme: A permutation $h : [d] \to [d]$ is sampled at random, and $y \subseteq \{0,1\}^d$ is placed in bucket $i \in [m]$ if $h(i) \in y$ and $\forall_{j<i} h(j) \notin y$. The probability for a collision between two sets q, y is then $|q \cap y| / (|q| + |y| - |q \cap y|)$ by a standard argument which implies an exponent of $\rho_{\text{mh}} = \log \frac{w_1}{w_q + w_u - w_1} \Big/ \log \frac{w_2}{w_q + w_u - w_2}$.

Now consider building multiple independent such MinHash tables, but *keeping only the kth bucket in each one.* That gives a Locality Sensitive Filter family, which we will analyse in this section.

The Locality Sensitive Filter approach to similarity search is an extension by Becker et al. [BDGL16] to the Locality Sensitive Hashing framework by Indyk and Motwani [IM98]. We will use the following definition by Christiani [Chr17], which we have slightly extended to support separate universes for query and data points:

Definition C.28 (LSF)**.** Let X and Y be some universes, let $S : X \times Y \to \mathbb{R}$ be a similarity function, and let $\mathcal{F}$ be a probability distribution over $\{(Q, U) \mid Q \subseteq X, U \subseteq Y\}$. We say that F is $(s_1, s_2, p_1, p_2, p_q, p_u)$-sensitive if for all points $x \in X, y \in Y$ and (Q, U) sampled randomly from $\mathcal{F}$ the following holds:

1. If $S(x, y) \geq s_1$ then $\Pr[x \in Q, y \in U] \geq p_1$.
2. If $S(x, y) \leq s_2$ then $\Pr[x \in Q, y \in U] \leq p_2$.
3. $\Pr[x \in Q] \leq p_q$ and $\Pr[x \in U] \leq p_u$.

We refer to (Q, U) as a filter and to Q as the query filter and U as the update filter.

We first state the LSF-Symmetrization lemma implicit in [CP17]:

Lemma C.29 (LSF-Symmetrization)**.** *Given a (p_1, p_2, p_q, p_u)-sensitive LSF-family, we can create a new family that is $(p_1 \frac{q}{p}, p_2 \frac{q}{p}, q, q)$-sensitive, where $p = \max\{p_q, p_u\}$ and $q = \min\{p_q, p_u\}$.*

For some values of p_1, p_2, p_q, p_u this will be better than simply taking $\max(\rho_u, \rho_q)$. In particular when symmetrization may reduce ρ_u by a lot by reducing its denominator.

Proof. W.l.o.g. assume $p_q \geq p_u$. When sampling a query filter, $Q \subseteq U$, pick a random number $\varrho \in [0, 1]$. If $\varrho > p_u/p_q$ use $\emptyset$ instead of Q. The new family then has $p'_q = p_q \cdot p_u/p_q$ and so on giving the lemma. □

Getting back to MinHash, we note that the "keeping only the ith bucket" family discussed above, corresponds sampling a permutation s of Y and taking the filter

$$U = \{x \mid s_i \in x \wedge s_0 \notin x \wedge \cdots \wedge s_{i-1} \notin x\}.$$

That is, the collection of x such that the first $i - 1$ values of s are not in x (since then x would have been put in that earlier bucket), but the ith element of s is in x (since otherwise x would have been put in a later bucket.)

Using just one of these families, combined with symmetrization, gives the ρ value:

$$\rho_i = \log \frac{(1 - w_q - w_u + w_1)^i w_1}{\max\{(1 - w_q)^i w_q, (1 - w_u)^i w_u\}} \Big/ \log \frac{(1 - w_q - w_u + w_2)^i w_2}{\max\{(1 - w_q)^i w_q, (1 - w_u)^i w_u\}}.$$

This scheme is a generalization of Chosen Path, since taking $i = 0$ recovers exactly that algorithm. However, as we increase i, we see that the weight gradually shifts from the *present* elements (symbolized by w_1, w_2, w_q and w_u) to the *absent* elements (symbolized by $(1 - w_q - w_u + w_q)$, etc.).

We will now show that for a given set of (w_q, w_u, w_1, w_2) there is always an optimal i which is better than using all of the i, which is what MinHash does. The exact goal is to show

$$\rho_{\text{mh}} = \log \frac{w_1}{w_q + w_u - w_1} \Big/ \log \frac{w_2}{w_q + w_u - w_2} \geq \min_{i \geq 0} \rho_i.$$

For this we show the following lemma, which intuitively says that it is never advantageous to combine multiple filter families:

Lemma C.30. *The function $f(x, y, z, t) = \log(\max\{x, y\}/z)/\log(\max\{x, y\}/t)$, defined for $\min\{x, y\} \geq z \geq t > 0$, is quasi-concave.*

This means in particular that

$$\frac{\log(\max\{x + x', y + y'\}/(z + z'))}{\log(\max\{x + x', y + y'\}/(t + t'))} \geq \min\left\{\frac{\log(\max\{x, y\}/z)}{\log(\max\{x, y\}/t)}, \frac{\log(\max\{x', y'\}/z')}{\log(\max\{x', y'\}/t')}\right\},$$

when the variables are in the range of the lemma.

Proof. We need to show that the set

$$\left\{(x,y,z,t) : \frac{\log(\max\{x,y\}/z)}{\log(\max\{x,y\}/t)} \geq \alpha\right\} = \{(x,y,z,t) : \max\{x,y\}^{1-\alpha}t^{\alpha} \geq z\}$$

is convex for all $\alpha \in [0,1]$ (since $z \geq t$ so $f(x,y,z,t) \in [0,1]$). This would follow if $g(x,y,t) = \max\{x,y\}^{1-\alpha}t^{\alpha}$ would be quasi-concave itself, and the eigenvalues of the Hessian of g are exactly 0, 0 and $-(1-\alpha)\alpha t^{\alpha-2}\max\{x,y\}^{-\alpha-1}\left(\max\{x,y\}^2 + t^2\right)$ so g is even concave! □

We can then show that MinHash is always dominated by one of the filters described, as

$$\begin{aligned}\rho_{\text{mh}} = \frac{\log\frac{w_1}{w_q+w_u-w_1}}{\log\frac{w_2}{w_q+w_u-w_2}} &= \frac{\log\frac{\sum_{i\geq 0}(1-w_q-w_u+w_1)^i w_1}{\max\{\sum_{i\geq 0}(1-w_q)^i w_q, \sum_{i\geq 0}(1-w_u)^i w_u\}}}{\log\frac{\sum_{i\geq 0}(1-w_q-w_u+w_2)^i w_2}{\max\{\sum_{i\geq 0}(1-w_q)^i w_q, \sum_{i\geq 0}(1-w_u)^i w_u\}}} \\ &\geq \min_{i\geq 0}\frac{\log\frac{(1-w_q-w_u+w_1)^i w_1}{\max\{(1-w_q)^i w_q,(1-w_u)^i w_u\}}}{\log\frac{(1-w_q-w_u+w_2)^i w_2}{\max\{(1-w_q)^i w_q,(1-w_u)^i w_u\}}},\end{aligned}$$

where the right hand side is exactly the symmetrization of the "only bucket i" filters. By monotonicity of $(1-w_q)^i w_q$ and $(1-w_u)^i w_u$ we can further argue that it is even possible to limit ourselves to one of $i \in \{0, \infty, \log(w_q/w_u)/\log((1-w_q)/(1-w_u))\}$, where the first gives Chosen Path, the second gives Chosen Path on the complemented sets, and the last gives a balanced trade-off where $(1-w_q)^i w_q = (1-w_u)^i w_u$.

C.5 Conclusion and Open Problems

For a long time there was a debate [SL14b] about why MinHash worked so well for sets, compared to other more general methods, like SimHash. It was a mystery why this method, so foreign to the frameworks of Spherical LSF and Chosen Path could still do so much better. For asymmetric problems like Subset Search, it was entirely open how far ρ could be reduced.

This paper finally solves the mystery of MinHash and unifies the ideas and frameworks of Euclidean and Set Similarity Search.

By showing that supermajorities indeed solve the general problem optimally, we not only unify and explain the performance of the previous literature, but also recover major performance improvements, space/time trade-offs, and the ability to solve Set Similarity Search for any similarity measure.

We propose the following open problems for future research:

LSH with polylog time When parametrized accordingly, we get a data structure with $e^{\tilde{O}(\sqrt{\log n})}$ query time and $n^{O(1)}$ space. Using Spherical LSH one can get similar runtime, though with a higher polynomial space usage. Employing a tighter analysis

of our algorithm, the query time can be reduced to $e^{\tilde{O}((\log n)^{1/3})}$, which by comparison with we conjecture is tight for the approach. A major open question is whether one can get $\tilde{O}(1)$?

Data-dependent Data-dependent LSH is able to reduce approximate similarity search problems to the case where far points are as far away as had they been random. For (w_q, w_u, w_1, w_2)-GapSS this corresponds to the case $w_2 = w_q w_u$. This would finally give the "optimal" algorithm for GapSS without any "non-data-dependent" disclaimers.

Sparse, non-binary data Our lower bounds really hold for a much larger class of problems, including cosine similarity search on sparse data in $\mathbb{R}^d$. However, our upper bounds currently focus on binary data only. It would be interesting to generalize our algorithm to this and other types of data for which Supermajorities are also optimal.

Sketching We have shown that Supermajorities can shave large polynomial factors of space and query time in LSH. Can they be used to give similar gains in the field of sketching sets under various similarity measures? Can one expand the work of [PSW14] and show optimality of some intersection sketching scheme?

C.5.1 Acknowledgements

We would like to thank Rasmus Pagh and Tobias Christiani on suggesting the problem discussed in this paper, as well as for reviews of previous manuscripts. Much of the initial work in the paper was done while the first author was visiting Eric Price at the University of Texas, and we would like to thank people there for encouragement and discussions on Boolean functions. Many people have helped reading versions of the manuscript, and we would like to thank Morgan Mingle, Morten Stöckel, John Kallaugher, Ninh Pham, Evangelos Kipouridis, Peter Rasmussen, Anders Aamand, Mikkel Thorup and everyone else who has given feedback. Finally, we really appreciate the comprehensive comments from the anonymous reviewers!

Thomas D. Ahle and Jakob B. T. Knudsen are partly supported by Thorup's Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

References

[ARW17] Amir Abboud, Aviad Rubinstein, and Ryan Williams. "Distributed PCP theorems for hardness of approximation in P". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 25–36.

[AV15] Amirali Abdullah and Suresh Venkatasubramanian. "A directed isoperimetric inequality with application to bregman near neighbor lower bounds". In: *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 2015, pp. 509–518.

[AAK10] Parag Agrawal, Arvind Arasu, and Raghav Kaushik. "On indexing error-tolerant set containment". In: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM. 2010, pp. 927–938.

[ABGM14] Daniel Ahlberg, Erik Broman, Simon Griffiths, and Robert Morris. "Noise sensitivity in continuum percolation". In: *Israel Journal of Mathematics* 201.2 (2014), pp. 847–899.

[APRS16] Thomas Dybdahl Ahle, Rasmus Pagh, Ilya Razenshteyn, and Francesco Silvestri. "On the complexity of inner product similarity join". In: *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. ACM. 2016, pp. 151–164.

[ACW16] Josh Alman, Timothy M Chan, and Ryan Williams. "Polynomial representations of threshold functions and algorithmic applications". In: *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2016, pp. 467–476.

[AI06] Alexandr Andoni and Piotr Indyk. "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions". In: *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*. IEEE. 2006, pp. 459–468.

[AINR14] Alexandr Andoni, Piotr Indyk, Huy L Nguyen, and Ilya Razenshteyn. "Beyond locality-sensitive hashing". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2014, pp. 1018–1028.

[ALRW17a] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. "Optimal hashing-based time-space trade-offs for approximate near neighbors". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2017, pp. 47–66.

[ALRW17b] Alexandr Andoni, Thijs Laarhoven, Ilya Razenshteyn, and Erik Waingarten. "Optimal hashing-based time-space trade-offs for approximate near neighbors". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 47–66.

[AR15] Alexandr Andoni and Ilya Razenshteyn. "Optimal data-dependent hashing for approximate near neighbors". In: *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*. ACM. 2015, pp. 793–801.

[ARN17] Alexandr Andoni, Ilya Razenshteyn, and Negev Shekel Nosatzki. "Lsh forest: Practical algorithms made theoretical". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 67–78.

[AR16] Alexandr Andoni and Ilya Razensteyn. "Tight Lower Bounds for Data-Dependent Locality-Sensitive Hashing". In: *32nd International Symposium on Computational Geometry (SoCG 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016.

[BHP01] Roger C Baker, Glyn Harman, and János Pintz. "The difference between consecutive primes, II". In: *Proceedings of the London Mathematical Society* 83.3 (2001), pp. 532–562.

[BCG05] Mayank Bawa, Tyson Condie, and Prasanna Ganesan. "LSH forest: self-tuning indexes for similarity search". In: *Proceedings of the 14th international conference on World Wide Web*. 2005, pp. 651–660.

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. "New directions in nearest neighbor searching with applications to lattice sieving". In: *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2016, pp. 10–24.

[Bec75] William Beckner. "Inequalities in Fourier analysis". In: *Annals of Mathematics* (1975), pp. 159–182.

[Big77] John D Biggins. "Martingale convergence in the branching random walk". In: *Journal of Applied Probability* 14.1 (1977), pp. 25–37.

[Bon70] Aline Bonami. " 'E study of the Fourier coefficients of the functions of $L\ p(G)$". In: *Annals of the Fourier Institute*. Vol. 20. 2. 1970, pp. 335–402.

[Bro97] Andrei Z Broder. "On the resemblance and containment of documents". In: *Compression and Complexity of Sequences 1997. Proceedings*. IEEE. 1997, pp. 21–29.

[BGMZ97] Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. "Syntactic clustering of the web". In: *Computer Networks and ISDN Systems* 29.8-13 (1997), pp. 1157–1166.

[Cha17] Timothy M Chan. "Orthogonal range searching in moderate dimensions: kd trees and range trees strike back". In: *33rd International Symposium on Computational Geometry (SoCG 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.

[CIP02] Moses Charikar, Piotr Indyk, and Rina Panigrahy. "New algorithms for subset query, partial match, orthogonal range searching, and related problems". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2002, pp. 451–462.

[Cha02] Moses S Charikar. "Similarity estimation techniques from rounding algorithms". In: *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*. ACM. 2002, pp. 380–388.

[CW19] Lijie Chen and Ryan Williams. "An equivalence class for orthogonal vectors". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 21–40.

[CCT10] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. "A survey of binary similarity and distance measures". In: *Journal of Systemics, Cybernetics and Informatics* 8.1 (2010), pp. 43–48.

[Chr17] Tobias Christiani. "A framework for similarity search with space-time tradeoffs using locality-sensitive filtering". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 31–46.

[CP17] Tobias Christiani and Rasmus Pagh. "Set similarity search beyond MinHash". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*. 2017, pp. 1094–1107.

[CPT18] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. "Confirmation Sampling for Exact Nearest Neighbor Search". In: *arXiv preprint arXiv:1812.02603* (2018).

[CK09] Edith Cohen and Haim Kaplan. "Leveraging discarded samples for tighter estimation of multiple-set aggregates". In: *ACM SIGMETRICS Performance Evaluation Review* 37.1 (2009), pp. 251–262.

[Cra36] Harald Cramér. "On the order of magnitude of the difference between consecutive prime numbers". In: *Acta Arithmetica* 2 (1936), pp. 23–46.

[Din94] Ian H. Dinwoodie. "Large Deviations Techniques and Applications (Amir Dembo and Ofer Zeitouni)". In: *SIAM Review* 36.2 (1994), pp. 303–304.

[Dub10] Moshe Dubiner. "Bucketing coding and information theory for the statistical high-dimensional nearest-neighbor problem". In: *IEEE Transactions on Information Theory* 56.8 (2010), pp. 4166–4179.

[FMNM19] Raul Castro Fernandez, Jisoo Min, Demitri Nava, and Samuel Madden. "Lazo: A Cardinality-Based Method for Coupled Estimation of Jaccard Similarity and Containment". In: *ICDE*. 2019.

[FFGM07] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. "Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm". In: *Discrete Mathematics and Theoretical Computer Science*. Discrete Mathematics and Theoretical Computer Science. 2007, pp. 137–156.

[Fri15] Ehud Friedgut. "An information theoretic proof of a hypercontractive inequality". In: *Entropy* 1/29 (2015).

[GG10] Ashish Goel and Pankaj Gupta. "Small subset queries and bloom filters using ternary associative memories, with applications". In: *ACM SIGMETRICS Performance Evaluation Review* 38.1 (2010), pp. 143–154.

[HIM12] Sariel Har-Peled, Piotr Indyk, and Rajeev Motwani. "Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality." In: *Theory of computing* 8.1 (2012), pp. 321–350.

[IM98] Piotr Indyk and Rajeev Motwani. "Approximate nearest neighbors: towards removing the curse of dimensionality". In: *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM. 1998, pp. 604–613.

[JZYY+18] Lianyin Jia, Lulu Zhang, Guoxian Yu, Jinguo You, Jiaman Ding, and Mengjuan Li. "A Survey on Set Similarity Search and Join." In: *International Journal of Performability Engineering* 14.2 (2018).

[Kap15] Michael Kapralov. "Smooth tradeoffs between insert and query complexity in nearest neighbor search". In: *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. ACM. 2015, pp. 329–342.

[KP12] Michael Kapralov and Rina Panigrahy. "NNS lower bounds via metric expansion for l_∞ and EMD". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2012, pp. 545–556.

[KLLM19] Peter Keevash, Noam Lifshitz, Eoin Long, and Dor Minzer. "Hypercontractivity for global functions and sharp thresholds". In: *arXiv preprint arXiv:1906.05568* (2019).

[Laa15] Thijs Laarhoven. "Tradeoffs for nearest neighbors on the sphere". In: *arXiv preprint arXiv:1511.07527* (2015).

[Lif18] Noam Lifshitz. "Hypergraph removal lemmas via robust sharp threshold theorems". In: *arXiv preprint arXiv:1804.00328* (2018).

[LJWC+07] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. "Multi-probe LSH: efficient indexing for high-dimensional similarity search". In: *Proceedings of the 33rd International Conference on Very Large Data Bases*. VLDB Endowment. 2007, pp. 950–961.

[MMP18] Samuel McCauley, Jesper W Mikkelsen, and Rasmus Pagh. "Set Similarity Search for Skewed Data". In: *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 2018, pp. 63–74.

[MG03] Sergey Melnik and Hector Garcia-Molina. "Adaptive algorithms for set containment joins". In: *ACM Transactions on Database Systems (TODS)* 28.1 (2003), pp. 56–99.

[MNP06] Rajeev Motwani, Assaf Naor, and Rina Panigrahi. "Lower bounds on locality sensitive hashing". In: *Proceedings of the twenty-second annual symposium on Computational geometry*. ACM. 2006, pp. 154–157.

[Nai14] Chandra Nair. "Equivalent Formulations of Hypercontractivity Using Information Measures". In: *International Zurich Seminar on Communications*. 2014, p. 42.

[NS15] Behnam Neyshabur and Nathan Srebro. "On Symmetric and Asymmetric LSHs for Inner Product Search". In: *International Conference on Machine Learning*. 2015, pp. 1926–1934.

[ODo14] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[OWZ14] Ryan O'Donnell, Yi Wu, and Yuan Zhou. "Optimal lower bounds for locality-sensitive hashing (except when q is tiny)". In: *ACM Transactions on Computation Theory (TOCT)* 6.1 (2014), p. 5.

[Ole03] Krzysztof Oleszkiewicz. "On a nonsymmetric version of the Khinchine-Kahane inequality". In: *Stochastic inequalities and applications*. Springer, 2003, pp. 157–168.

[PSW14] Rasmus Pagh, Morten Stöckel, and David P Woodruff. "Is min-wise hashing optimal for summarizing set intersection?" In: *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM. 2014, pp. 109–120.

[Pan06] Rina Panigrahy. "Entropy based nearest neighbor search in high dimensions". In: *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics. 2006, pp. 1186–1195.

[PTW08] Rina Panigrahy, Kunal Talwar, and Udi Wieder. "A geometric approach to lower bounds for approximate near-neighbor search and partial match". In: *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2008, pp. 414–423.

[PTW10] Rina Panigrahy, Kunal Talwar, and Udi Wieder. "Lower bounds on near neighbor search via metric expansion". In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 805–814.

[PW14] Yury Polyanskiy and Yihong Wu. "Lecture notes on information theory". In: *Lecture Notes for ECE563 (UIUC) and* 6.2012-2016 (2014), p. 7.

[RPNK00] Karthikeyan Ramasamy, Jignesh M. Patel, Jeffrey F. Naughton, and Raghav Kaushik. "Set Containment Joins: The Good, The Bad and The Ugly". In: *VLDB*. 2000.

[RSW20] Cyrus Rashtchian, Aneesh Sharma, and David P Woodruff. "LSF-Join: Locality Sensitive Filtering for Distributed All-Pairs Set Similarity Under Skew". In: *arXiv preprint arXiv:2003.02972* (2020).

[Riv76] Ronald L Rivest. "Partial-match retrieval algorithms". In: *SIAM Journal on Computing* 5.1 (1976), pp. 19–50.

[Shi15] Zhan Shi. *Branching random walks*. Springer, 2015.

[SL14a] Anshumali Shrivastava and Ping Li. "Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS)". In: *Advances in Neural Information Processing Systems*. 2014, pp. 2321–2329.

[SL14b] Anshumali Shrivastava and Ping Li. "In defense of minhash over simhash". In: *Artificial Intelligence and Statistics*. 2014, pp. 886–894.

[TT07] Kengo Terasawa and Yuzuru Tanaka. "Spherical lsh for approximate nearest neighbor search on unit hypersphere". In: *Workshop on Algorithms and Data Structures*. Springer. 2007, pp. 27–38.

[Wil05] Ryan Williams. "A new algorithm for optimal 2-constraint satisfaction and its implications". In: *Theoretical Computer Science* 348.2 (2005), pp. 357–365.

[Wol07] Paweł Wolff. "Hypercontractivity of simple random variables". In: *Studia Mathematica* 3.180 (2007), pp. 219–236.

[YLDC+18] Xiao Yan, Jinfeng Li, Xinyan Dai, Hongzhi Chen, and James Cheng. "Norm-Ranging LSH for Maximum Inner Product Search". In: *Advances in Neural Information Processing Systems*. 2018, pp. 2956–2965.

[ZLWZ+17] Yong Zhang, Xiuxing Li, Jin Wang, Ying Zhang, Chunxiao Xing, and Xiaojie Yuan. "An efficient framework for exact set similarity search using tree structure indexes". In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE. 2017, pp. 759–770.

C.I Proof of Lemma C.17

This proof in this section mostly follows [ALRW17a], with a few changes to work with separate spaces Q and U.

Lemma C.17. *Let Q and U be some spaces and $\mathcal{P}_{QU}$ a probability distribution on $Q \times U$. Consider any list-of-points data structure for $\mathcal{P}_{QU}$-random instances of n points, which uses expected space $n^{1+\rho_u}$, has expected query time $n^{\rho_q - o_n(1)}$, and succeeds with probability at least 0.99. Let $r, s \in [1, \infty]$ satisfy*

$$\mathop{\mathrm{E}}_{(X,Y)\sim\mathcal{P}_{QU}}[f(X)g(Y)] \leq \|f(X)\|_{L_r(\mathcal{P}_Q)} \|f(Y)\|_{L_s(\mathcal{P}_U)} ,$$

for all functions $f : Q \to \mathbb{R}$ and $g : U \to \mathbb{R}$. Then

$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{1}{r} + \frac{1}{s} - 1 ,$$

where $r' = \frac{r}{r-1}$ is the convex conjugate of r.

Proof. Fix a data structure D, where A_i specifies which dataset points are placed in L_i. Additionally, we define $B_i = \{v \mid i \in I(v)\}$ to the set of query points which scan L_i. We

sample a random dataset point u and then a random query point v from the neighborhood of u. Let

$$\gamma_i = \Pr[v \in B_i \,|\, u \in A_i]$$

represent the probability that query v scans the list L_i conditioned on u being in L_i. The query time for D is given by the following expression

$$\begin{aligned} T &= \sum_{i\in[m]} [v \in B_i] \left(1 + \sum_{j\in[n]} [u_j \in A_i]\right) \\ \mathrm{E}[T] &= \sum_{i\in[m]} \Pr[v \in B_i] + \sum_{i\in[m]} \gamma_i \Pr[u \in A_i] + (n-1) \sum_{i\in[m]} \Pr[u \in A_i] \Pr[v \in B_i] \ . \end{aligned}$$

We want to lower bound $\Pr[v \in B_i]$, so let $1 \leq r, s$ be any values such that $\mathcal{P}_{QU}$ is (r, s)-hypercontractive. We then get that

$$\begin{aligned} \gamma_i \Pr[u \in A_i] &= \Pr[u \in A_i \wedge v \in B_i] \\ &= \mathrm{E}[[u \in A_i][v \in B_i]] \\ &\leq \|[u \in A_i]\|_{L_s(p_1)} \|[v \in B_i]\|_{L_r(p_2)} \\ &= \Pr[u \in A_i]^{1/s} \Pr[v \in B_i]^{1/r} \end{aligned}$$

Hence we get that $\Pr[v \in B_i] \geq \gamma_i^r \Pr[u \in A_i]^{r/s'}$. We define $\tau_i = \Pr[u \in A_i]$ and get that

$$\mathrm{E}[T] \geq \sum_{i\in[m]} \gamma_i^r \tau_i^{r/s'} + \sum_{i\in[m]} \gamma_i \tau_i + (n-1) \sum_{i\in[m]} \gamma_i^r \tau_i^{1+r/s'} \ .$$

Since the data structure succeeds with probability γ we have that

$$\sum_{i\in[m]} \tau_i \gamma_i \geq \Pr[\exists i \in [m] : v \in B_i, u \in A_i] = \gamma \ .$$

Since D uses at most S space we have that

$$m + \sum_{i\in[m]} |A_i| \leq S \ \Rightarrow \ \sum_{i\in[m]} \tau_i \leq \frac{S}{n} \ .$$

We then get that we want to minimize

$$\begin{aligned} \mathrm{E}[T] &\geq \sum_{i\in[m]} \gamma_i^r \tau_i^{r/s'} + \sum_{i\in[m]} \gamma_i \tau_i + (n-1) \sum_{i\in[m]} \gamma_i^r \tau_i^{1+r/s'} \\ &\geq \sum_{i\in[m]} \gamma_i^r \tau_i^r (\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s}) \ , \end{aligned}$$

given the constraints

$$\sum_{i\in[m]} \tau_i\gamma_i \geq \gamma$$
$$\sum_{i\in[m]} \tau_i \leq \frac{S}{n}\,.$$

First we fix $(\tau_i)_{i\in[m]}$ and minimize the function with respect to $(\gamma_i)_{i\in[m]}$. Using Lagrange multipliers this is equivalent to minimizing the function

$$f((\gamma_i)_{i\in[m]},\lambda,\nu) = \sum_{i\in[m]} \gamma_i^r\tau_i^r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s}) - \lambda(\sum_{i\in[m]} \tau_i\gamma_i - \gamma - \nu^2)$$

We find the critical points $\nabla f = 0$:

$$r\gamma_i^{r-1}\tau_i^r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s}) = \lambda\tau_i$$
$$\sum_{i\in[m]} \tau_i\gamma_i = \gamma + \nu^2$$
$$2\lambda\nu = 0$$

for all $i \in [m]$. We note that since $\gamma > 0$ then $\lambda > 0$ and hence $\nu = 0$. The first inequality can be rewritten as

$$\gamma_i^{r-1}\tau_i^{r-1} = \frac{\lambda}{r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s})} \qquad \Leftrightarrow$$
$$\gamma_i\tau_i = \left(\frac{\lambda}{r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s})}\right)^{r'/r}$$

Combining this with $\sum_{i\in[m]} \tau_i\gamma_i = \gamma$ give us that

$$\sum_{i\in[m]} \left(\frac{\lambda}{r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s})}\right)^{r'/r} = \gamma \qquad \Leftrightarrow$$
$$\lambda^{r'/r} = \frac{\gamma}{\sum_{i\in[m]} \left(\frac{1}{r(\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s})}\right)^{r'/r}}$$

We define $t_i = \left(\frac{1}{\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s}}\right)^{r'/r}$ and get that

$$\gamma_i\tau_i = \gamma\frac{t_i}{\sum_{i\in[m]} t_i} \qquad \Leftrightarrow$$
$$\gamma_i^r\tau_i^r = \gamma^r\frac{t_i^r}{(\sum_{i\in[m]} t_i)^r}$$

We then get that our original function becomes

$$\gamma^r \sum_{i\in[m]} \frac{t_i^r}{(\sum_{i\in[m]} t_i)^r} t_i^{-r/r'} = \gamma^r \sum_{i\in[m]} \frac{t_i}{(\sum_{i\in[m]} t_i)^r} = \gamma^r (\sum_{i\in[m]} t_i)^{-(r-1)} = \gamma^r (\sum_{i\in[m]} t_i)^{-r/r'}$$

So we want to maximize

$$\sum_{i\in[m]} t_i = \sum_{i\in[m]} \left(\frac{1}{\tau_i^{-r/s} + (n-1)\tau_i^{1-r/s}} \right)^{r'/r} = \sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}}$$

We now consider two different cases.

Case 1. $r > s$**.** We know that $\tau_i \leq 1$ so we get that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \sum_{i\in[m]} \frac{\tau_i^{r'(1/s-1/r)}}{n^{r'/r}}$$

Since $r'(1/s - 1/r) > 0$ then we can use the power-mean inequality to get

$$\begin{aligned}
\sum_{i\in[m]} \frac{\tau_i^{r'(1/s-1/r)}}{n^{r'/r}} &\leq \frac{m}{n^{r'/r}} \left(\frac{\sum_{i\in[m]} \tau_i}{m} \right)^{r'(1/s-1/r)} \\
&\leq \frac{m^{r'-r'/s}}{n^{r'/r}} \left(\frac{S}{n} \right)^{r'(1/s-1/r)} \\
&= \frac{m^{r'/s'}}{n^{r'/s}} S^{r'(1/s-1/r)} \\
&\leq \frac{S^{r'/s'+r'(1/s-1/r)}}{n^{r'/s}} \\
&= \frac{S}{n^{r'/s}}
\end{aligned}$$

where we have used that $\max\left\{m, n\sum_{i\in[m]} \tau_i\right\} \leq S$.

Case 2. $r \leq s$ We find the derivatives

$$\begin{aligned}
&\frac{d}{d\tau_i} \sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \\
&\quad = \frac{\frac{r'}{s} t_i^{r'/s-1}(1+(n-1)\tau_i)^{r'/r} - (n-1)\frac{r'}{r}(1+(n-1)\tau_i)^{r'/r-1} t_i^{r'/s}}{(1+(n-1)\tau_i)^{2r'/r}} \\
&\quad = \frac{r' t_i^{r'/s-1}}{(1+(n-1)\tau_i)^{r'/r+1}} \left(\frac{1}{s}(1+(n-1)\tau_i) - (n-1)\frac{1}{r}\tau_i \right)
\end{aligned}$$

Case 2.1. $r < s$ We note that the function is maximized when we set $\tau_i = \frac{\frac{1}{s}}{(n-1)(\frac{1}{r}-\frac{1}{s})} = \frac{r}{(n-1)(s-r)}$. This give us that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq m \frac{\left(\frac{r}{(n-1)(s-r)}\right)^{r'/s}}{\left(1+\frac{r}{s-r}\right)^{r'/r}} \leq \frac{S}{n^{r'/s}} \frac{\left(\frac{2r}{s-r}\right)^{r'/s}}{\left(\frac{s}{s-r}\right)^{r'/r}}$$

where we have used that $m \leq S$ and $n \geq 2$.

$$m\frac{r}{(n-1)(s-r)} \leq \frac{S}{n} \Rightarrow m \leq S\frac{(n-1)(s-r)}{nr}$$

Case 2.2. $r = s$ We note that the function is increasing in τ_i so it is maximized when $\tau_i = 1$. Then we get that

$$\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \frac{m}{n^{r'/r}} = \frac{S}{n^{r'/r}} = \frac{S}{n^{r'/s}}$$

where we have used that $m \leq S$ and $r = s$.

From this we note that if we set $K = \max\left\{1, \frac{(\frac{2r}{s-r})^{r'/s}}{(\frac{s}{s-r})^{r'/r}}\right\}$ then $\sum_{i\in[m]} \frac{\tau_i^{r'/s}}{(1+(n-1)\tau_i)^{r'/r}} \leq \frac{S}{n^{r'/s}}K$. Now we can give the final lower bound on $\mathrm{E}[T]$:

$$\mathrm{E}[T] \geq \gamma^r (\sum_{i\in[m]} t_i)^{-r/r'} \geq \gamma^r \left(\frac{S}{n^{r'/s}}K\right)^{-r/r'} = \gamma^r K^{-r/r'} S^{-r/r'} n^{r/s}$$

From this we get the result we want

$$\rho_q \geq -\frac{r}{r'}(1+\rho_u) + \frac{r}{s} - o_n(1) \Leftrightarrow$$
$$\frac{1}{r}\rho_q + \frac{1}{r'}\rho_u \geq \frac{1}{s} - \frac{1}{r'} - o_n(1)$$

□

Appendix D

Load Balancing with Dynamic Set of Balls and Bins

Load Balancing with Dynamic Set of Balls and Bins

Anders Aamand
University of Copenhagen

Jakob Bæk Tejs Knudsen
University of Copenhagen

Mikkel Thorup
University of Copenhagen

Dansk resumé

In dynamic load balancing, we wish to distribute balls into bins in an environment where both balls and bins can be added and removed. We want to minimize the maximum load of any bin but we also want to minimize the number of balls and bins that are affected when adding or removing a ball or a bin. We want a hashing-style solution where we given the ID of a ball can find its bin efficiently.

We are given a user-specified balancing parameter $c = 1 + \varepsilon$, where $\varepsilon \in (0, 1)$. Let n and m be the current number of balls and bins. Then we want no bin with load above $C = \lceil cn/m \rceil$, referred to as the *capacity* of the bins.

We present a scheme where we can locate a ball checking $1 + O(\log 1/\varepsilon)$ bins in expectation. When inserting or deleting a ball, we expect to move $O(1/\varepsilon)$ balls, and when inserting or deleting a bin, we expect to move $O(C/\varepsilon)$ balls. Previous bounds were off by a factor $1/\varepsilon$.

The above bounds are best possible when $C = O(1)$ but for larger C, we can do much better: Let

$$f = \begin{cases} \varepsilon C & \text{if } C \leq \log 1/\varepsilon \\ \varepsilon\sqrt{C} \cdot \sqrt{\log(1/(\varepsilon\sqrt{C}))} & \text{if } \log 1/\varepsilon \leq C < \frac{1}{2\varepsilon^2} \\ 1 & \text{if } C \geq \frac{1}{2\varepsilon^2} \end{cases}$$

We show that we expect to move $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin. Moreover, when $C \geq \log 1/\varepsilon$, we can search a ball checking only $O(1)$ bins in expectation.

For the bounds with larger C, we first have to resolve a much simpler probabilistic problem. Place n balls in m bins of capacity C, one ball at the time. Each ball picks a uniformly random non-full bin. We show that in expectation and with high probability, the fraction of non-full bins is $\Theta(f)$. Then the expected number of bins that a new ball would have to visit to find one that is not full is $\Theta(1/f)$. As it turns out, this is also the complexity of an insertion in our more complicated scheme where both balls and bins can be added and removed.

Contents

D.1 Introduction

Load balancing in dynamic environments is a central problem in designing several networking systems and web services [SMLK+03; KLLP+97]. We wish to allocate *clients* (also referred to as *balls*) to *servers* (also referred to as *bins*) in such a way that none of the servers gets overloaded. Here, the *load* of a server is the number of clients allocated to it. We want a hashing-style solution where we given the ID of a client can efficiently find its server. Both clients and servers may be added or removed in any order, and with such changes, we do not want to move too many clients. Thus, while the dynamic allocation algorithm has to always ensure a proper load balancing, it should aim to minimize the number of clients moved after each change to the system. For every update in the system, we need to change the allocation of clients to servers. For simplicity, we assume that the updates (ball and bin insertions and removals) do not happen simultaneously and will be operated one at a time, so that we have time to finish changing the allocation before we get another update. Such allocation problems become even more challenging when we face hard constraints in the capacity of each server, that is, each server has a *capacity* and the load may not exceed this capacity. Typically, we want capacities close to the average loads.

There is a vast literature on solutions in the much simpler case where the set of servers is fixed and only the client set is updated. For now, we focus on solutions that are known to work in our fully-dynamic case where both clients and servers can be added and removed in an arbitrary order. This rules out solutions where only the last added server may be removed[1]. The above problem formulation is very general, and does not assume anything about the ratio between the number of clients n, and the number of servers m. Processors are cheap, so one could for instance imagine systems with a large number of servers. However, it is also conceivable having a system with many clients or a balanced system with $n \approx m$.

The classic solution to the scenario where both clients and servers can be added and removed is Consistent Hashing [SMLK+03; KLLP+97] where the current clients are assigned in a random way to the current servers. While consistent hashing schemes minimize the expected number of movements, they may result in hugely overloaded servers, and they do not allow for explicit capacity constraints on the servers. The basic point is that the load balancing of consistent hashing [KLLP+97; SMLK+03] is no better than a random assignment of clients to servers. The same issue holds for Highest Random Weight Hashing (popularly known as Rendezvous Hashing) [TR98]. Hence, with n clients and m servers, we expect good load balancing if $n/m = \omega(\log m)$, but the balance is lost with smaller loads, e.g., with $n \approx m$, we expect many servers to be overloaded with $\Theta(\log m / \log \log m)$ clients.

More recently, Mirrokni et al. [MTZ18] presented an algorithm that works with arbitrary capacity constraints on the servers. For the purpose of load balancing, the system designer can specify a balancing parameter $c = 1 + \varepsilon$, guaranteeing that the maximum load is at most $\lceil cn/m \rceil$. While maintaining this hard balancing constraint, they limit the expected number of clients to be moved when clients or servers are inserted or removed. From a more practical perspective, we think of the load balancing parameter $c = 1 + \varepsilon$ as a simple knob which captures the tradeoff between load balancing and stability upon changes in the system. This gives a more direct control to the system designer in meeting explicit balancing constraints.

Even without capacity constraints, the obvious general lower bounds for moves are as follows. When a client is added or removed, at least we have to move that client. When a server is added or removed, at least we have to move the clients belonging to it. On the average, we therefore have to move least $\frac{n}{m}$ clients when a server is added or removed.

With the algorithm from [MTZ18], while guaranteeing a balancing parameter $c = 1 + \varepsilon \leq 2$, when a client is added or removed, the expected number of clients moved is $O(\frac{1}{\varepsilon^2})$. When a server is added or removed, the expected number of clients moved is $O(\frac{n}{\varepsilon^2 m})$. These numbers are only a factor $O(\frac{1}{\varepsilon^2})$ worse than the general lower bounds without capacity constrains. For balancing parameter $c \geq 2$, the expected number of moves is increased by a factor $1 + O(\frac{\log c}{c})$ over the lower bounds. This implies that for superconstant c, we only expect to pay a negligible cost in extra moves.

Focusing on the challenging case where $c = 1 + \varepsilon \leq 2$, we present an algorithm which

[1] In particular, this rules out the external memory techniques [Lar88] where blocks (playing the role of fixed capacity servers) can only be added to and removed from the top of the current memory.

reduces the number of moves by a factor $1/\varepsilon$. When inserting or deleting a ball, we expect to move $O(1/\varepsilon)$ balls, and when inserting or deleting a bin, we expect to move $O(C/\varepsilon)$ balls. To search a ball we only need to consider $O(\log(1+1/\varepsilon))$ "consecutive" bins.

With $C := cn/m$, these bounds are essentially best possible when $C = O(1)$ is a constant. However, for larger C, we can do even better. In order to explain, this we first have to consider the following much simpler probabilistic problem: Consider placing n balls in m bins, each of capacity $C = (1+\varepsilon)n/m$, one ball at the time, where each ball picks a uniformly random non-full bin. We are interested in the number of non-full bins both in expectation and with concentration bounds. To our surprise, this relatively simple problem does not seem to have been analyzed before, and so, we believe our bounds to be of independent interest. To state our bounds, we define

$$f = \begin{cases} \varepsilon C & \text{if } C \leq \log 1/\varepsilon \\ \varepsilon\sqrt{C} \cdot \sqrt{\log(1/(\varepsilon\sqrt{C}))} & \text{if } \log 1/\varepsilon \leq C < \frac{1}{2\varepsilon^2} \\ 1 & \text{if } C \geq \frac{1}{2\varepsilon^2} \end{cases}, \tag{D.1}$$

whenever $0 < \varepsilon \leq 1$ and $C \geq 1$ is integral. We are going to prove the following result

Theorem D.1. *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$ be such that $C = (1+\varepsilon)n/m$ is integral. Moreover assume that that $1/\varepsilon = m^{o(1)}$. Suppose we distribute n balls sequentially into m bins each of capacity C, for each ball choosing a uniformly random non-full bin. The expected fraction of non-full bins is $\Theta(f)$.*

How does this result relate to our dynamic load allocation problem? We can think of the distribution scheme in the theorem as the algorithmically *weakest* way to assign the balls to the capacitated bins. Here, by algorithmically weak, we mean that it cannot be implemented in the dynamic setting where balls and bins can come and go. However, it is still helpful to think of it as the mathematically *ideal* way of solving dynamic load allocation with bounded loads in the following sense. Imagine that an insertion of a ball is carried out by repeatedly choosing a random bin until we find a non-full one where we place the ball. Then we avoid all the unpleasant dependencies between the loads of the bins visited during the insertion that arise in algorithmically stronger schemes. For example, one can compare to a scheme like linear probing where the cascading effect of balls causes heavy dependencies between the loads of bins visited during a search or an insertion. It follows from Theorem D.1 that in the simple scheme above, the expected number of bins visited when making an insertion is $O(1/f)$. The main contribution of this paper is to present a much stronger scheme which supports general insertions and deletions of both balls and bins, and which, nonetheless, achieves complexity bounds that are analogous to those in the mathematically ideal scheme above. To be precise, with our scheme, we expect to move $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin and this is tight. Similar bounds holds on the number of bins visited when performing any of these updates. Our main technical challenge is handling all the intricate dependencies that arise in the much more complicated probabilistic setting in our scheme.

Applications. Consistent hashing has found numerous applications [ÖV11; GF04] and early work in this area [KLLP+97; SMKK+01; SMLK+03] has been cited more than ten thousand times. To highlight the wide variety of areas in which similar allocation problems might arise, we mention a few more important references to applications: content-addressable networks [RFHK+01], peer-to-peer systems and their associated multicast applications [RD01; CDKR02]. Our algorithm and that from [MTZ18] are very similar to consistent hashing, and should work for most of the same applications, bounding the loads whenever this is desired. In fact, the algorithm from [MTZ18] already found two quite different industrial applications; namely Google's cloud system [MZ17] and Vimeo's video streaming [Rod16]. Both systems had to handle the lightly loaded case. Also, in both cases, load balancing was not an objective to maximize, but rather a hard constraint, e.g., in the Vimeo blog post [Rod16], Rodland describes how no server is allowed to be overloaded, and how he found a load balancing parameter $c = 1.25$ to be satisfactory for Vimeo's video steaming. We shall return to this later. With our algorithm, we get the same load balancing but with much fewer reallocations.

D.1.1 Background: Consistent Hashing

The standard solution to our fully-dynamic allocation problem is consistent hashing [SMLK+03; KLLP+97]. We shall use it as a starting point for own own solution, so we review it below.

Simple Consistent Hashing. In the simplest version of consistent hashing, we hash the active balls and bins onto a unit circle, that is, we hash to the unit interval, using the hash values to create a circular order of balls and bins. Assuming no collisions, a ball is placed in the bin succeeding it in the clockwise order around the circle. One of the nice features of consistent hashing is that it is history-independent, that is, we only need to know the IDs of the balls and the bins and the hash functions, to compute the distribution of balls in bins. If a bin is closed, we just move its balls to the succeeding bin. Similarly, when we open a new bin, we only have to consider the balls from the succeeding bin to see which ones belong in the new bin.

With n balls, m bins, and a fully random hash function h, each bin is expected to have n/m balls. This is also the number of balls we expect to move when a bin is opened or closed.

One problem with simple consistent hashing as described above is that the maximum load is likely to be $\Theta(\log m)$ times bigger than the average. This has to do with a big variation in the coverage of the bins. We say that bin b *covers* the interval of the cycle from the preceding bin b' to b because all balls hashing to this interval land in b. When m bins are placed randomly on the unit cycle, on the average, each bin covers an interval of size $1/m$, but we expect some bins to cover intervals of size $\Theta(\frac{\log m}{m})$, and such bins are expected to get $\Theta(\frac{n \log m}{m})$ balls. The maximum load is thus expected to be a factor $\Theta(\log m)$ above the average.

A related issue is that the expected number of balls landing in the same bin as any given ball is almost twice the average. More precisely, consider a particular ball x. Its

expected distance to the neighboring bin on either side is exactly $1/(m+1)$, so the expected size of the interval between these two neighbors is $2/(m+1)$. All balls landing in this interval will end in the same bin as x; namely the bin b succeeding x. Therefore we expect $2(n-1)/(m+1) \approx 2n/m$ other balls to land with x in b. Thus each ball is expected to land in a bin with load almost twice the average. If the load determines how efficiently a server can serve a client, the expected performance is then only half what it should be.

In [KLLP+97] they addressed the above issue using so called virtual bins. We will also employ these virtual bins in our solution and describe them below.

Consistent Hashing with Virtual Bins. To get a more uniform bin cover, [KLLP+97] suggests the use of *virtual bins*. The virtual bin trick is that the ball contents of $k = O(\log m)$ virtual bins is united in a single super bin. The super bins are the m bins seen by the user of the system. Internally it is the km virtual bins we place on the cycle together with the n balls. Each virtual bin has a pointer to its super bin. To place a ball, we go along the cycle to the first virtual bin, and then we follow the pointer to its super bin.

A super bin covers the union of the intervals covered by its k virtual bins. The point is that for any constant $\varepsilon > 0$, if we pick a large enough $k = O(\log m)$, then with high probability, each super bin covers a fraction $(1 \pm \varepsilon)/m$ of the unit cycle.

We note that many other methods have been proposed to maintain such a uniform bin cover as bins are added and removed (see, e.g., [BSS00; GH05; Man04; KM05; KR06; TR98]), and in our algorithms, we shall also employ such virtual bins.

With a uniform bin cover, balls distribute uniformly between bins. On the positive side, in the heavily loaded case when n/m is large, e.g., $n/m = \omega(\log m)$, all loads are $(1 \pm o(1))n/m$, w.h.p. However, with $n = m$, we still expect many bins with $\Theta((\log m)/(\log \log m))$ balls even though the average is 1. In this paper, we aim for good load balancing for all possible load levels.

D.1.2 Simple Consistent Hashing with Bounded Loads.

As we mentioned earlier, Mirrokni et al. [MTZ18] presented an algorithm that works with arbitrary capacity constraints on the bins. For the purpose of load balancing, the system designer can specify a balancing parameter $c = 1 + \varepsilon$, guaranteeing that the maximum load is at most $C = \lceil cn/m \rceil$.

Their idea is very simple. As in simple consistent hashing, we place balls and bins randomly on a cycle, but instead of placing balls in the first bin along the cycle, we place them in the first non-full bin. Thus we can think of the distribution as first placing all the bins on the cycle, and then placing the balls one-by-one, putting each in the first non-full bin found by going in clockwise around the cycle. If we have hash functions for placing arbitrary balls and bins along the cycle, and if we have a priority order on all balls, telling us the order in which we insert balls, then this completely determines the placement of any set of the balls in any set of capacitated bins. This means that the distribution is history independent as in [BG07]. It also means that we know exactly which balls to move if balls or bins are added or removed.

As terminology, we say a ball *hash to* the first bin following it in the clockwise order. However, the ball may be *placed* in a later bin if the bin it hashed to was full.

Note that the priority order makes the insertion of a new ball a bit more complicated since it may have higher priority than balls already in the system. To place it, we first place it in the bin it hashes to directly (that is, the one just after its hash location on the cycle). If the bin becomes overfull, we pop the lowest priority ball and place it in the next bin, and repeat. It is, however, important to notice that the bins we end up considering are exactly the bins from the one the ball hashes to, and to the first non-full bin.

The details of all the different system updates are described in Mirrokni et al. [MTZ18]. This also includes rolling adjustment of the capacities relative to average load n/m. Instead of giving all bins the maximal capacity $C = \lceil cn/m \rceil$, they always have $\lceil cn \rceil - m \lfloor cn/m \rfloor$ bins with capacity $\lfloor cn/m \rfloor$. The only exception is that we never drop any capacity below 1. A hash function choose which bins have which capacities, and this ensures that only few capacities have to be changed with each system update. In Mirrokni et al. [MTZ18] they show that their results hold, both when capacities are adjusted to ε, and when a joint capacity C is given, defining $\varepsilon = Cm/n - 1$. In this paper, for simplicity, we will focus on the latter model with fixed capacities.

Mirrokni et al. [MTZ18] also provided an analysis of their system. With $\varepsilon \leq 1$, they showed that starting from the hash location of any ball, the expected number of full bins passed on the way to the first non-full bin is $O(1/\varepsilon^2)$. From this they get that the expected number of balls that has to be moved when a ball is inserted or deleted is $O(1/\varepsilon^2)$. Likewise, the expected number of balls that has to be moved when a bin is inserted or deleted is $O(C/\varepsilon^2)$. These bounds are all tight for simple consistent hashing with bounded loads.

Finally, Mirrokni et al. [MTZ18] also discussed many potentially relevant techniques that could possibly be made to work for fully-dynamic load balancing where both balls and bins can be added and removed, and with strict requirements on the maximal load for each bin. In these comparisons, their scheme was the one with the best proven bounds on the number of moves needed in connection with the updates.

Faster Searches

Mirrokni et al. [MTZ18] states that to search a ball, they have to consider $O(1/\varepsilon^2)$ bins, but using an old trick [AK74; Knu73], this is easily improved to $O(1/\varepsilon)$. The idea is that when we search for a ball, we can stop as soon as we reach a bin that is not filled with balls of higher priority. This helps the searches if the priorities are random. We shall use the idea later, so let's elaborate. The bins considered in the search are exactly the bins from the bin hashed to and till the first non-full bin if only the balls of higher priority was inserted. Let $r(q, m, C)$ be expected number of bins considered if there are q balls of higher priority, and m bins of capacity C. Then with n balls in total, the expected cost with random priorities is $\sum_{q=0}^{n} r(q, m, C)/(n+1)$. The analysis in [MTZ18] implies $r(q, m, C) = O(1/\varepsilon_q^2)$ where $\varepsilon_q = C/\frac{q}{m} - 1$, implying an expected cost of $O(1/\varepsilon)$ with random priorities.

We note that random priorities do not help with updates, for if we, say, want to insert a ball, and meet a bin that is full including balls of lower priority, then we have to place the lowest priority ball in a later bin. However, finding the established server of a client if any, is often the most frequent operation in the system, so a faster search is very important in practice. As stated, a similar analysis gives that for our system, we have to consider fewer bins when searching than when inserting a ball. In particular, we only need to consider $O(1)$ bins in expectation when $C \geq \log 1/\varepsilon$.

D.1.3 Our Scheme: Consistent Hashing with Virtual Bins and Bounded Loads

Our algorithm basically just combines the bounded loads with virtual bins. When a ball is placed in a virtual bin, it is also placed in its super bin which has a limited capacity. In the following, we describe two different versions of our scheme. The first one, described in Appendix D.1.3, is conceptually the simplest to understand and easier to analyze mathematically. It is this version that we will analyze in the main body of the paper. The second one, described in Appendix D.1.3, is the version most suitable to be implemented in practice for several reasons to be described. Our results hold for both implementations, and in Appendix D.9, we sketch how to derive the results for the second more practical version. Common to both versions is that we fix some natural number k, which is the number of virtual bins for each super bin.

Mathematically Clean Version: Many Independent Cycles

For this version, we hash each super bin to k different cycles or levels using independent hash functions[2]. The k hash values on the k cycles will be the associated virtual bins of the given super bin. We also hash the balls to the cycles, but contrary to the bins, each ball gets just a single random hash value on a single random cycle.

The static placement of the balls can be described as follows: We start by placing all balls which hash to the first cycle using standard consistent hashing with bounded loads as described in Appendix D.1.2. We assume that we have priorities on the balls and we will simulate that they are inserted in priority order. After the first level, the balls hashing to this level have thus been distributed into the virtual bins and we put them in the corresponding super bins. Initially, each super bin had capacity C. If the virtual bin of such a super bin received a balls at the first level, its new capacity is then reduced accordingly to $C - a$. We continue this process on level $i = 2, \ldots, k$. At level i, each super bin has a certain remaining capacity and we use standard consistent hashing with bounded loads (with these capacities) to place the balls at level i into the virtual bins and thus, into the corresponding super bins. If a super bin had capacity C_0 before the hashing to level i, and it received a balls at level i, its remaining capacity for the next levels is $C_0 - a$. Traversing the levels one at a time like described, corresponds to enforcing that regardless

[2]For simplicity, we advice the reader to think of all our hash functions as fully random. However, our results hold even when the hashing is implemented with the practical mixed tabulation from [DKRT15]. We will later sketch how our proofs can be modified to show this.

of the initial priorities of the balls, if two balls hash to different levels, the ball hashing to the lower level will have the highest priority of the two. With these modified priorities, the static image at a given point can be obtained by simply inserting the balls one by one in priority order, placing each ball in the first virtual bin whose super bin is not full. This completely describes the placement of balls in bins if we know the hash functions and the priority order, so the system is history-independent as described in [BG07].

Searching for a ball x is almost the same as for normal consistent hashing. We calculate the hash value of x and visit the virtual bins starting from that hash value in cyclic order until we either find x in a corresponding super bin or we meet a ball of lower priority hashing to the same level.

Insertions are a bit more complicated. For inserting a ball x we calculate $h(x)$ which in particular indicates the level, i, that x hashes to. We traverse level i starting at $h(x)$ until we meet a bin, b, which either (a) is not full or (b) contains a ball of lower priority than x (all balls hashing to levels $j > i$ have lower priority than x by convention). We insert x in b. In case (a), the insertion is complete, but in case (b) we pop y from b and recurse the insertion starting with y (which happens at some level $j \geq i$).

Ball deletions are symmetric to ball insertions in the sense that the hash functions tells us exactly the placement of all balls in bins, both before and after the ball which we are to insert or delete is inserted or deleted. Deleting a bin is the same as re-inserting all balls in it, and inserting a bin is symmetric to deleting a bin. Therefore we get that the number of balls to be moved is essentially determined by the number that has to be moved in connection with an insertion (we shall discuss this in more detail later).

For most of our results, we will assume that the hashing of balls to the different levels is uniform, but in Appendix D.2 we will see an applications where the probability of hashing to level i is $1/2^i$ for $1 \leq i \leq k-1$ and 2^{-k+1} for $i = k$. In this setting we already obtain a big improvement over standard consistent hashing using just $\log 1/\varepsilon$ levels.

Practical Version: A Single Linear Order

We next describe the more practical implementation of our algorithm and here we will also give more details on the concrete ranges of the hash functions. As will be seen, it is very similar to the the version above having some minor alterations. For this implementation all balls and all virtual bins are hashed to a single range, which we think of not as a cyclic order but rather as a linear order. In order to describe the static image at given point, we would again consider the balls one by one in priority order, placing each ball in the first virtual bin whose super bin is not full. Again, this ensures that the system is history-independent.

We now provide some more details on the hash functions and the priority order. Generally the hash values are in some universe $[u] = \{0, \ldots, u-1\}$. We imagine u to be so large that we expect no collisions between hash values (if there are ties, we can break them in favour of the ID's of the balls, but we will ignore this detail). We also think of both balls and bins having ID's in $[u]$.

We have a single hash $h : [u] \to [u]$ describing the hash location of the balls. We also use h to give the random priority order of the balls, inserting those with smallest hash

values first.

For the super bins, and for some parameter k, each bin has $k+1$ associated virtual bins. Their hash locations are described via $k+1$ hash functions $h_i : [u] \to [u]$, $i \in [k] = \{0, \ldots, k\}$. We assume that k divides u, e.g., that both are powers of two, and we restrict h_i to map uniformly into $[iu/k, (i+1)u/k)$. This way each super bin gets exactly one virtual bin in each of the $k+1$ intervals $[iu/k, (i+1)u/k)$. Having this spread is important because of the priority order of the balls, which implies that virtual bins with larger hash values are more likely to be full.

The last interval $[u, u+u/k)$ is outside the normal hash range $[u]$. These last virtual bins will pick up any key that did not end in a bin in the normal range $[u]$. Since every super bin is represented in $[u, u+u/k)$, all balls are picked up unless there are more balls than the total capacity. As a result, we do no longer think of balls and bins as hashing to a cycle, but just to a linearly ordered universe with an extra set of representative virtual bins by the end making sure that all balls get placed.

We briefly explain why this system is preferable in practice. The first reason is that when using the hash values of the balls as their priorities we obtain a very simple description of the static distribution of balls in the bins: We may simply insert the balls in order from lowest to highest hash value, always placing the ball in the first non-full bins. A way of picturing this is to imagine that the balls of lower hash values are "pushing" balls of higher hash values ahead of them. On a line, it is very easy to implement this comparison as a standard comparison between hash values. In fact, it is possible to obtain a similar image for cycles, but for this one needs to impose a *cyclic* priority order of the balls hashing to a given level, and performing comparisons for such a cyclic order is a bit more technical to implement[3]. If on the other hand, we decided to stick with the linear priority order on each cycle, thus giving up on the nice image from above, we still encounter some technical issues with the implementation. With searches and insertions, everything works fine, but the issues come up when deleting balls and inserting bins. For instance, when deleting a ball which is placed in the "last" bin on the cycle, we may have to pull back balls that have been forwarded from this bin to the "first" bins in the cycle, and for deciding if such balls are to be pulled back, we have to use a different comparison of hash values. Thus, even with linear priorities the cyclic probing still muddies the implementation and makes it less efficient.

Again, we shall play a bit with the ranges of the hash functions for the virtual bins. However, they will always partition $[u]$ consecutively with the range of h_i following the range of h_{i-1}. With the *exponentially decreasing* hash ranges described by the end of Appendix D.1.3, h_i, maps uniformly to $[u - u/2^i, u - u/2^{i+1})$ for $i \in [k-1]$ and h_{k-1} maps uniformly to $[u - u/2^{k-1}, u)$. As above h_k is special, mapping to $[u, u+u/k)$.

Searches and insertions have similar descriptions to the ones given in Appendix D.1.3. Moreover, the history independence again implies that deletions are symmetric to insertions. Finally, deleting a bin corresponds to inserting the ball in the bin, and inserting a bin is symmetric to the deletion of the bin.

[3]For example, for just two balls, the notion of one hashing before the other is not well defined.

D.1.4 Main Results on Consistent Hashing

We now present our main results on consistent hashing with bounded loads and virtual bins.

$O(1/\varepsilon)$ Reallocated Balls, with $\log 1/\varepsilon$ Levels

Our first result, to be proved in Appendix D.2, uses a logarithmic number of virtual bins to achieve that the number of bins visited during an insertion (and thus the number of reallocated balls) is $O(1/\varepsilon)$. It uses a non-uniform distribution of the balls to the different levels, with the probability of a ball hashing to level i being 2^{-i} for $1 \leq i \leq k-1$ and 2^{-k+1} for $i = k$.

Theorem D.2. *Let $0 < \varepsilon < 1$ and suppose that we distribute n balls into m bins each of capacity $C = (1+\varepsilon)n/m$ using consistent hashing with bounded loads and $k = \lceil \log(1/\varepsilon) \rceil$ levels, where the probability, p_i, that a ball hashes to level i is*

$$p_i = \begin{cases} 2^{-i}, & 1 \leq i \leq k-1 \\ 2^{-k+1}, & i = k. \end{cases}$$

Assume that $1/\varepsilon = n^{o(1)}$. When inserting or deleting a ball, we expect to visit (and hence move) $O(1/\varepsilon)$ balls, and when inserting or deleting a bin, we expect to move $O(C/\varepsilon)$ balls. Finally, when searching a ball, we expect to visit $O(\log 1/\varepsilon)$ bins.

In the previous system of simple consistent hashing with bounded loads, but no virtual bins, Mirrokni et al. [MTZ18] proved that ball insertions and deletions are expected to move $O(1/\varepsilon^2)$ balls while bin insertions and deletions are expected to move $O(C/\varepsilon^2)$ balls. Those bounds are a factor $1/\varepsilon$ worse than ours. Mirrokni et al. [MTZ18] would also perform searches considering $O(1/\varepsilon^2)$ bins in expectation, but using the trick of assigning random priorities to the balls, one can get down to $O(1/\varepsilon)$ bins in expectation, still without the use of virtual bins. Combining our scheme using virtual bins, with the trick of random priorities the expected number of bins visited during a search drops exponentially to $O(\log 1/\varepsilon)$, as stated in the theorem.

When proving Theorem D.2, the main technical challenge is bounding the expected number of bins visited during an insertion. In fact, the remaining parts of the theorem follow once we have this bound. In Appendix D.7, we will argue why the results on ball deletions and bin insertions and deletions follow. Finally, in Appendix D.8, we will use the trick described in Appendix D.1.2 to prove the result on ball searches.

Better Bounds when the Capacities are Large

In classic consistent hashing without virtual bins, we obtain no advantage when the number of balls n are much larger than the number of bins m, or in other words, when the capacity of a bin, C, is large. The basic issue is that most of the uncertainty in the system without virtual bins stems from the uncertainty in the distance between a bin and its predecessor, which determines the expected number of balls hashing directly to the bin.

However, the use of virtual bins improves the concentration of the number of balls hashing directly to a super bin, and we do obtain an advantage of this improved concentration. This was in fact the whole point of introducing virtual bins in classic consistent hashing without load bounds [SMLK+03]. To be precise, fix $k = A(\log n)/\varepsilon^2$ for some appropriately large constant A. Then standard Chernoff bounds show that each bin cover a fraction $(1 \pm \lambda\varepsilon)/m$ of the combined hash range, where λ can be made arbitrarily small (by increasing A). If further the average load m/n is above k, then with high probability, no bin gets load above $C = (1+\varepsilon)m/n$ by balls hashing directly to them. In particular, all load bounds are satisfied without the having to forward a single ball. The result below (which is the main result of our paper) asymptotically settles the expected insertion time for general C, in particular for any $C \leq (\log n)/\varepsilon^2$. Before stating the theorem, we encourage the reader to recall the definition of f in eq. (D.1)

Theorem D.3. *Let $0 < \varepsilon < 1$ and suppose that we distribute n balls into m bins each of capacity $C = (1+\varepsilon)n/m$ using consistent hashing with bounded loads and $k = c/\varepsilon^2$ uniform levels for a sufficiently large constant c. Assume that $1/\varepsilon = n^{o(1)}$. In expectation we move $O(1/f)$ balls when inserting or deleting a ball, and $O(C/f)$ balls when inserting or deleting a bin. Finally, when searching a ball, we expect to visit $O(1)$ bins when $C \geq \log 1/\varepsilon$ and $O(\frac{\log 1/\varepsilon}{C})$ bins when $C < \log 1/\varepsilon$.*

Our bounds in Theorem D.3 show that we do get an advantage from bigger capacities even when C is smaller than $k = \Theta((\log n)/\varepsilon^2)$. In fact, already for $C = 1/\varepsilon^2$, the expected insertion time drops to $O(1)$.

Again, the hardest part of proving Theorem D.3, is bounding the expected number of bins visited during an insertion by $O(1/f)$. As for Theorem D.2, we argue that the remaining parts of the theorem follows in Appendices D.7 and D.8

High Probability Bounds Theorems D.2 and D.3 only bound the expected number of balls moved during the insertions and deletions of balls and bins. However, it is also possible to obtain high probability bounds. We will provide such high probability bounds in a later full version of the paper.

Distributing Balls Randomly into Capacitated Bins

To understand the strength of our bounds, we consider a much simpler problem where we place n balls in m bins, each of capacity $C = (1+\varepsilon)n/m$, one ball at the time. Each ball picks a uniformly random non-full bin. Letting X denote the fraction of non-full bins, we show in Appendix D.3 that $\mathrm{E}[X] = \Theta(f)$ and $X = \Theta(f)$ with high probability. Surprisingly, this relatively simple question has not been studied before.

What is the idea of considering this simpler distribution scheme? With a fraction of X non-full bins, the expected number of random bins visited in order to find one of the non-full ones is $1/X$. This is reminiscent to searching for a non-full bin using (any variation of) consistent hashing with bounded loads, except that we get rid of the intricate dependencies which arise in the more complicated schemes that can handle both insertions and deletions. In this way, the scheme above can be thought of as the simplest way of

achieving the desired load balancing, but of course it has no chance of working in a fully dynamic setting. We thus obtain, the same complexity bounds as the weakest system imaginable, at the same time being able to handle both insertions and deletions of balls and bins.

The Practical Implementation with Mixed Tabulation

When proving Theorems D.2 and D.3, we will assume that our scheme is implemented as described in Appendix D.1.3 and, moreover, using fully random hash functions. In Appendix D.9 we will sketch why our results hold even with the more practical implementation from Appendix D.1.3. We will also sketch how one can obtain the same results with the practical mixed tabulation scheme from [DKRT15]. In the implementation with mixed tabulation, we would use k independent mixed tabulation hash functions for the hashing of virtual bins, and a single independent mixed tabulation hash function for the hashing of balls.

D.1.5 The Model and its Applicability.

Consistent hashing with or without virtual bins is a simple versatile scheme that has been implemented in many different systems with different constraints and performance measures [ÖV11; GF04]. The most classic implementation of consistent hashing is the distributed system Chord [SMKK+01; SMLK+03] which has more than ten thousand citations. The Chord papers [SMKK+01; SMLK+03] give a thorough description of the many issues affecting the design. On the high level, they have a system of pointers so that given an arbitrary hash location, they can find the next bin in the clockwise order using $O(\log n)$ messages. This is how they find the (virtual) bin a ball hashes to. In simple consistent hashing, this is where the ball is to be found. With virtual bins, there are additional pointers between virtual bins and their super bins that we can follow using $O(1)$ messages. In fact, Chord does maintain explicit successor pointers between neighboring (virtual) bins, so we only have to pay $O(1)$ extra messages to find a next bin along the cycle.

As described by Mirrokni et al. [MTZ18], the successor pointers give immediate support for forwarding in case of capacitated bins. Mirrokni et al. only used this forwarding for simple consistent hashing without virtual bins, and this has been adopted both by Google's Cloud Pub/Sub [MZ17] and Vimeo [Rod16]. Both systems had to handle the lightly loaded case. Also, in both cases, load balancing was not an objective to maximize, but rather a hard constraint, e.g., in the Vimeo blog post [Rod16], Rodland describes how no server is allowed to be overloaded, and how he found a load balancing parameter $c = 1 + \varepsilon = 1.25$ to be satisfactory for Vimeo's video steaming.

The successor pointers in Chord work equally well for moving between virtual bins. In fact, Rodland from Vimeo has told (personal communication) the last author, Thorup, that their system does allow a combination of virtual bins and bounded loads, like what we suggest in this paper, so a system similar to ours is already running. Thorup had the general idea from much earlier (around the time of the first versions of [MTZ18]), but

deriving the mathematical understanding, presented here in Theorem D.3 took several years.

Let us now consider the time to search a ball in a Chord-like setting. By Theorem D.3, we expect to consider $O(\log(1/f))$ consecutive virtual bins with associated super bins. Finding the virtual bin succeeding the hash location uses $O(\log n)$ messages while each other bin is found with $O(1)$ messages. Then our message bottleneck is actually to find the first virtual bin.

Now it could be the case that balls/clients themselves remembered if they are in the system, and if so, what bin/server they belonged to. The latter requires that they are notified if they get moved due to other updates in the system, e.g., if their bin/server was removed.

Another way to circumvent the $O(\log n)$ messages for placing the hash location would be if we for some $\hat{m} = \Theta(m)$, placed the reference points $p_i = ui/\hat{m}$, $i \in [\hat{m}]$, in the doubly-linked list of virtual bins. For a ball x its hash reference point is $p_{\lfloor h(x)\hat{m}/u \rfloor}$. Regardless of system updates, it could remember its reference point, and from there follow in expectation $O(1)$ successor pointers to get the current virtual bin succeeding its real hash location. The reference points could be updated by background rebuilding to be ready every time m is halved or doubled, thus maintaining an $\hat{m}$ approximating m within a factor of 2.

In fact, our scheme is equally relevant for less distributed systems than Chord. In Google's Cloud Pub/Sub [MZ17], the most important aspects of the system was (1) that it has good load balance (2) that only few clients/balls have to be moved in connection with update, that is, a ball or bin insertion or deletion, and (3) history independence so that the placement of balls in bins can be computed by anyone knowing the hash functions and the current set of balls and bins. The fact that each system update only leads to few moves implies that even if we have a few mistakes in the set of balls and bins, then this only implies a few mistakes in the placement of balls in bins.

System updates, inserting or deleting a ball or a bins are hopefully not too frequent. As mentioned in [MZ17], the dominant concern is the actual reallocation of balls between bins; for in the real world, this means moving clients between servers disrupting service etc. Theorems D.2 and D.3 give us concrete bounds on how many balls we expect to move.

The computation of which balls are to be moved in connection with updates depends very much on the situation. As in [MZ17], thanks to history independence, we can compute the balls to be moved from scratch. We know the update to the set of balls and bins, and the hash functions tell us exactly which balls are placed in which bins before and after update. The difference tells us exactly which balls have to be moved. This solution if fine if the computation cost is small compared with the cost of actually moving the clients.

Alternatively, we may want a more distributed local identification of the moves as in in the Chord system. This is fairly straightforward for insertions, and we already described it earlier. It does, however, get a bit more complicated for the other updates, and we shall return to such a distributed implementation in Section D.1.6.

Stepping back, we offer a generic scheme for a load balanced distribution of balls in bins when both can be added and removed. We are not claiming to have a theoretical model that captures all the important aspects of performance since this depends very

much on the concrete implementation context. Our main contribution is a theoretical analysis of combinatorial parameters described in Theorems D.2 and D.3.

D.1.6 Computing Moves Locally in a Distributed Environment

We will now discuss how we could compute which balls have to be moved in connection with system updates in a distributed Chord-type system. Recall that sometimes it may be fast enough to identify the moves more centrally, simply by computing the placement of the balls in the bins before and after the update, and just identify the difference. However, in this subsection, we will discuss how to identify the moves locally, not spending much more time than the number of moves specified in Theorems D.2 and D.3.

We already discussed how to insert balls, but we want to do it in a way that also makes it fast and easy to delete balls. The basic idea to make deletions efficient is that we for every virtual bin store the number of balls that have passed it. More precisely, each bin has a pass count that starts at zero when there are no balls. We now consider the process where balls are inserted in priority order, each just placed in the first virtual bin with a non-empty super bin. This increases the count on all the virtual bins between the hash location and the virtual bin the ball ends in. Each super bin will also store which of its virtual bins that have a positive pass count.

The above pass counts are quite easy to maintain when balls arrive to the real system, that is, not in priority order. To see this, we review the insertion of a ball, adding when pass counts should be incremented. To insert a new ball, we first hash it to some location which also determines its priority. Starting from the hash location, we visit the virtual bins following, each time looking in the corresponding super bin. If the super bin is not full, we simply place the ball in it and terminate the insertion. If the super bin is filled with balls of higher priority, we increment the pass count of the virtual bin, and continue to the next virtual bin. However, if the super bin is filled and contains a ball of lower prioirty, we insert the new ball and pop the ball of lowest priority. The popped ball belongs to some virtual bin, which could be the same, but could also be only much later in the linear order than the virutal bin we just came from. The pass count is incremented from whichever virtual bin we pop the ball from, and then we recursiviely rinsert the popped ball, continuing from the next virtual bin. The $O(1/f)$ bound from Theorem D.3 actually bounds not only the number of moves, but also the number of bins considered during the above insertion.

Next we consider the deletion of a ball. Essentially, we just want to reverse the above process, systematically finding the balls the ball to be deleted have displaced. We think of deletions as first removing a ball, and then recursively, filling a hole. Finding the ball to be removed is easy, as described before, and when we remove it, we will have to decrement the pass count on all the virtual bins between its hash location and up to the virtual bin before the one it landed in. Next we want to see if we can refill the whole. Assuming that the bin we removed was in the level i virtual bin of a super bin. We now check corresponding super bin b to see if any ball has been displaced by the ball we deleted. This is the case if and only if at least one of its virtual bins has a positive pass count. Let j be the lowest level of a virtual bin with a positive pass count. It is not hard to see

that we must have $j \geq i$. We now consider the virtual bins following the level j virtual bin until we find a ball with hash location before $h_j(b)$. The virtual bins passed decrease their counts, and then we recursively delete the ball. As described above, our total work is within a constant factor of the symmetric insertion, that is, we consider $O(1/f)$ bins and spend $O(1/f)$ time in total.

We now consider the insertion of deletion of super bins. We think of these super bin or server updates as more rare than the ball or client updates.

Deleting a super bin b is relatively easy. Essentially, we just reinsert all the balls in it. A small detail is that if a ball x was in the level j virtual bin, then we insert it starting from $h_j(b)$ rather than from $h(x)$. This can only save work over the regular insertion of x and in particular, this means that we do not increase the pass count for virtual bins between $h(x)$ and $h_j(b)$. By Theorem D.3, the expected number of balls that has to be moved when deleting a super bin is $O(C/f)$. However, on top of that, we do have to spend at least $O(k)$ time on removing the k virtual bins from the system.

Inserting a super bin b is a bit more complicated. We would like to just fill it as we filled the holes arising when deleting a ball, but we have the issue that we do not know the pass counts for the k virtual bins representing the new super bin. To handle this, for $i = 1, \ldots, k$, we first find the hash location $h_i(b)$ of its virtual bin b_i, which takes $O(\log n)$ messages, including inserting it in the linked list of virtual bins. Next consider the virtual bin u following b_i. If bin u has no ball and pass count zero, then we can just set the pass count of $h_i(b)$ to zero. Otherwise, we continue along the virtual bins, counting the balls in them, until we find a ball that hash after $h_i(b)$. All but the last ball are the balls that have passed the level i virtual bin b_i, which now gets a pass count. Now that we have the pass count, we can move those balls to b_i, as long as super bin b has space for them, using the same procedure as described under deletions of balls.

We now first analyze the number of bins considered to compute the pass counts of the virtual bins b_i. We note that the bins considered are exactly the same as if we searched for a ball that hashed to $h_i(b)$. Now consider instead the case where we first generate a random $i \in [k]$, and then generate $h_i(b)$. With i random, $h_i(b)$ is uniformly random in $[u]$, and then the expected number of bins considered is exactly the same as those considered in the search of a ball with hash value uniformly random in $[u]$. We conclude that the expected total number of bins considered over all $i \in [k]$ is exactly k times bigger. Thus, by Theorem D.3, we expect to consider at most $O(k\frac{\log(1/\varepsilon)}{C})$ bins when $C \leq \log 1/\varepsilon$, and only $O(k)$ bins when $C \geq \log 1/\varepsilon$. Now that the pass counts are fixed, inserting a bin is symmetric to deleting it and has the same cost, yielding a bound of $O(C/f)$.

D.1.7 Dynamic Load Capacities

We now also consider what happens when we use self-adjusting capacities like Mirrokni et al. [MTZ18]. Below, the capacitated bins correspond to our super bins. Rather than fixed capacities, the user of the system specifies a balancing parameter $c = (1 + \varepsilon)$ and then the maximal capacity is $C = \lceil cn/m \rceil$. We do not want all bins to change capacity each time cn/m passes an integer.

Instead, as in Mirrokni et al. [MTZ18], assuming an arbitrary fixed ordering of the super bins, we let the lowest $q = \lceil cn \rceil - m \lfloor cn/m \rfloor$ super bins have capacity $C = \lceil cn/m \rceil$ while the remaining $r = m - q$ have capacity $C - 1$. We refer to the former bins as *big bins* and the latter bins as *small bins*, though the difference is only 1. Moreover, as an exception to the above rule, we will never let the capacity drop below 1, that is, if $cn < m$, then all bins have capacity 1.

The basic point in the above system is that a ball update changes at most $\lceil c \rceil = O(1)$ bin capacities while a bin update changes at most $O(C)$ capacities. Switching the capacity from large to small has the same effect as inserting an extra high priority ball in the super bin while leaving the capacity at C. In the other direction, switching the capacity from small to large corresponds to a deletion of an extra high priority ball.

From an analysis perspective, this means that we are essentially studying a system with $n' = r + n$ balls in bins of capacity C where $Cm = \lceil cn'/m \rceil$. In our analysis, this corresponds to having a 0th level which puts exactly one ball in each of r bins; 0 in the rest. Such a perfect level poses no issues for the analysis. Thus the cost per capacity change is the same as that of regular insertions/deletions, and therefore have no effect on our overall bounds.

A small point, elaborated in Mirrokni [MTZ18], is that for all the bounds to hold, we may always do things in the order that maximizes capacity in every step, so that we always have a total capacity of $Cm = \lceil c(n+q)/m \rceil$. For example, when inserting a ball, we increase capacities before inserting, while deleting a ball, we decrease capacities last. Likewise for a bin insertion, we insert it before decreasing capacities, while when deleting a bin, we start by increasing the capacities.

D.1.8 Roadmap of the Paper

We now present a brief roadmap of our paper as well as some of the theorems to be proven in the individual sections.

In Section D.2, we prove the part of Theorem D.2 concerning insertions of balls. That the statements about ball deletions and bin insertions and deletions follow, is covered in Appendix D.7. Finally, in Appendix D.8 we prove the statement of the theorem concerning ball searches.

To prove the main result of the paper, Theorem D.3, we first have to solve the much simpler problem of showing that when n balls are distributed into m bins each of capacity $C = (1+\varepsilon)n/m$, the expected fraction of non-full bins is $\Theta(f)$. This simpler problem is solved in Section D.3.

In Section D.4, we present a tail bound for sums of geometric random variables as well a technical lemma concerning consistent hashing with bounded loads and virtual bins. These results will be useful in the later sections towards the proof of Theorem D.3.

In Section D.5, we show that when distributing n balls into m bins using consistent hashing with bounded loads and enough levels, it similarly holds that the expected fraction of non-full bins is $\Theta(f)$, and moreover, that the number of non-full bins is concentrated around its mean. The exhibition is divided into two parts: In Section D.5.1, we prove the concentration result and in Section D.5.2, we determine the mean within a constant

factor. The following theorem is a corollary of the results from Appendix D.5 and we will require it to prove our main result in Section D.6.

Theorem D.4. *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$. Suppose we insert n balls into m bins, each of capacity $C = (1+\varepsilon)n/m$, using consistent hashing with bounded loads and virtual bins and k levels. For $(i,j) \in [k] \times [C+1]$, we let $X_{i,j}$ denote the number of bins with at most j balls after the hashing of balls to levels $0, \ldots, i-1$ and $\mu_{i,j} = \mathrm{E}[X_{i,j}]$. For any $\gamma = O(1)$ and $(i,j) \in [k] \times [C+1]$, it holds that $|X_{i,j} - \mu_{i,j}| \leq m^{1/2+o(1)}$ with probability $1 - n^{-\gamma}$.*

If moreover $k \geq c/\varepsilon^2$ for a sufficiently large universal constant c, it holds that $\mu_{k-1,C-1} = \Theta(fm)$.

In Section D.6, we show the part of Theorem D.3 which concerns ball insertions. Again, ball deletions, bin insertions, and bin deletions are handled in Appendix D.7, and searches are handled in Appendix D.8.

Finally, in Appendix D.9, we sketch why our results hold, even if we use the practical implementation described in Appendix D.1.3. We also sketch how to modify the proofs in the case where the hashing is implemented with the mixed tabulation scheme from [DKRT15].

D.2 Expected $O(1/\varepsilon)$ Insertion Time with $\lceil \log(1/\varepsilon) \rceil$ Levels

In this section we prove the part of Theorem D.2 concerning insertions, restated below. We will assume that we use the implementation described in Appendix D.1.3 but the result also holds with the other implementation in Appendix D.1.3 (see the Appendix D.9).

Theorem D.5. *Suppose that we distribute n balls into m bins each of capacity $C = (1+\varepsilon)n/m$ using consistent hashing with bounded loads and[4] $k = \lceil \log(1/\varepsilon) \rceil + 2$ levels, where the probability, p_i, that a ball hashes to level i is*

$$p_i = \begin{cases} 2^{-i}, & 1 \leq i \leq k-1 \\ 2^{-k+1}, & i = k. \end{cases}$$

Assume that $1/\varepsilon = n^{o(1)}$. The expected number of bins visited when inserting a ball is then $O(1/\varepsilon)$.

We remark that one way to implement the above hashing is by using an auxiliary hash function $s : U \to [2^{k-1}]$. Letting $h_1, \ldots, h_k$ denote the hash functions distributing balls at level $1, \ldots, k$, the hash value of a key $x \in U$ is then given by $h_{i+1}(x)$, where i is the number of leading 0's of $s(x)$.

Proof. Let Z denote the number of virtual bins visited in total and Z_i denote the number of virtual bins visited at level $i \in [k]$. Then $Z = \sum_{i=1}^{k} Z_i$. We will show that $\mathrm{E}[Z_i] = O(2^i)$ from which it follows that $\mathrm{E}[Z] = O(2^k) = O(1/\varepsilon)$.

[4]For simplicity, we have stated the theorem using $k = \lceil \log(1/\varepsilon) \rceil + 2$ levels as this makes the constants in the proof work out particularly nicely. However, a simple inspection of the proof of Theorem D.5 will show that the bound holds for any positive integer $k = \log(1/\varepsilon) - O(1)$.

First, it follows from a standard Chernoff bound that if X_i is the number of balls hashing to level i and $\mu_i = \mathrm{E}[X_i] = p_i n$, then for $\delta \leq 1$,

$$\Pr[|X_i - \mu_i| \geq \delta\mu_i] \leq \exp(-\delta^2\mu_i/3)$$

Thus , it holds that $|X_i - \mu_i| = O(\sqrt{\mu_i \log n})$ with probability at least $1 - n^{-3}$. Similarly, if $X_{<i} = \sum_{j<i} X_j$ and $\mu_{<i} = \sum_{j<i} \mu_j$, it holds that $|X_{<i} - \mu_{<i}| = O(\sqrt{\mu_{<i} \log n})$ with the same high probability.

For each $j \in [m]$, we define $C_j^{(i)}$ to be the remaining capacity of bin j after the distribution of balls to levels $1, \ldots, i-1$. Then $\sum_{j\in[m]} C_j^{(i)} = (1+\varepsilon)n - X_{<i}$, so it follows from the above that with probability $1 - O(n^{-2})$,

$$\sum_{j\in[m]} C_j^{(i)} \geq (1+\varepsilon)n - \mu_{<i} - O\left(\sqrt{\mu_{<i}\log n}\right) = (\varepsilon + 2^{-i+1})n - O(\sqrt{n\log n}).$$

For $i < k$ we have that $\mu_i = 2^{-i}n$, so it follows that, $\sum_{j\in[m]} C_j^{(i)} \geq 2X_i$ with probability $1 - O(n^{-2})$, where we used the assumption that $1/\varepsilon = n^{o(1)}$. In the case $i = k$, we instead have that

$$X_k \leq 2^{-k+1}n + O(\sqrt{n\log n}) \leq \varepsilon n/2 + (\sqrt{n\log n}),$$

with probability at least $1 - O(n^{-2})$, so again $\sum_{j\in[m]} C_j^{(i)} = X_k + \varepsilon n \geq 2X_k$, again using that $1/\varepsilon = n^{o(1)}$.

Now fix $i \in [k]$, and write $C_j = C_j^{(i)}$ for simplicity. Let $\mathcal{E}$ denote the event that $p_i n/2 \leq X_i \leq 2p_i n$ and that $\sum_{j\in[m]} C_j^{(i)} \geq 2X_k$. Then $\Pr[\mathcal{E}^c] = O(n^{-2})$, so that

$$\mathrm{E}[Z_i] \leq \mathrm{E}[Z_i \mid \mathcal{E}] + \mathrm{E}[Z_i \mid \mathcal{E}^c]\Pr[\mathcal{E}^c] \leq \mathrm{E}[Z_i \mid \mathcal{E}] + O(mn^{-2}) = \mathrm{E}[Z_i \mid \mathcal{E}] + O(1).$$

Thus, it will suffice to show that $\mathrm{E}[Z_i \mid \mathcal{E}] = O(2^i)$. Let b be the first bin visited at level i, i.e., during the insertion we at some level $j < i$ arrived at bin b and b is not full after the hashing of balls to level $1, \ldots, i-1$. Let I be a maximal interval at level i containing b and satisfying that all bins lying in I are full at level i. Let R denote the number of bins in I excluding b. Then $Z_i \leq R + 1$. We will show that $\mathrm{E}[R] = O(2^i)$ (for notational convenience the conditioning on $\mathcal{E}$ has been left out). Let $s \in \mathbb{N}$ be given and let A_s denote the even that $s + 1 \leq R \leq 2s$. We are now going to provide an upper bound on $\Pr[A_s]$. Let I_1^- and I_1^+ be the intervals respectively ending and starting at b and of lengths $\frac{s}{3m}$. Similarly, let I_2^- and I_2^+ be the intervals respectively ending and starting at b and of lengths $\frac{3s}{m}$. Let $I_1 = I_1^- \cup I_1^+$ and $I_2 = I_2^- \cup I_2^+$. Finally, partition I_2 into 54 intervals of equal lengths, $J_1, \ldots, J_{54}$. Let a be such that $(1-a)/(1+a) = 5/6$ (or $a = 1/11$) and $\overline{C} = \frac{1}{m}\sum_{j\in[m]} C_j$. We claim that if A^s holds then either of the following events must be true

B_1: I_2^- or I_2^+ contains at most $2s$ virtual bins different from b.

B_2: I_1^- or I_1^+ contains at least $s/2$ virtual bins different from b.

B_3: The total capacity of bins different than b hashing to J_j is at most $\frac{(1-a)s\overline{C}}{9}$ for some $1 \leq \ell \leq 54$.

B_4: The total number of balls hashing to J_ℓ is at least $\frac{(1+a)s\overline{C}}{18}$ for some $1 \leq \ell \leq 54$.

To see this, suppose that A_s occurs but neither of B_1, B_2, B_3 occurs. We show that then B_4 must occur. As $R \leq 2s$ and B_1 did not occur, $I \subseteq I_2$. As $R \geq s+1$ and B_2 did not occur, either $I_1^- \subseteq I$ or $I_1^+ \subseteq I$. Letting ℓ denote the number of j such that $J_j \subseteq I$ it therefore follows that $\ell \geq 3$. Since B_3 did not occur, the total capacity of bins hashing to I is at least $\frac{\ell\overline{C}(1-a)s}{9}$. Finally, since all balls which ends up in a bin in I must have hashed to I it follows that the total number of balls hashing to I is at least $\frac{\ell\overline{C}(1-a)s}{9}$. In particular, for some $1 \leq \ell \leq 54$, at least $\frac{\ell\overline{C}(1-a)s}{9(\ell+2)}$ balls must hash to J_ℓ. But

$$\frac{\ell\overline{C}(1-a)s}{9(\ell+2)} \geq \frac{\overline{C}(1-a)s}{15} = \frac{\overline{C}(1+a)s}{18},$$

so we conclude that B_4 holds.

Simple Chernoff bounds gives that inequality give that $\Pr[B_1] = \exp(-\Omega(s))$ and $\Pr[B_2] = \exp(-\Omega(s))$. To bound $\Pr[B_3]$, let $\ell \in [54]$ be fixed and define Y_j to be the indicator for bin j hashing to J_ℓ. Further, define $Y = \sum_{j\in[m]} Y_j$. Then $\mathrm{E}[Y] = \frac{\overline{C}s}{9}$. This time however, we only have that $|Y_j| \leq C$, so applying Chernoff we obtain that

$$\Pr[B_3] = \Pr[Y \leq (1-a)\,\mathrm{E}[Y]] = \exp\left(-\Omega\left(\frac{\mathrm{E}[Y]}{C}\right)\right) = \exp\left(-\Omega\left(\frac{s}{2^i}\right)\right)$$

For B_4, note that since we conditioned on $\mathcal{E}$, the expected number of balls hashing to an interval J_ℓ is $\frac{X_i s}{9m} \leq \frac{\overline{C}s}{18}$. Thus, another Chernoff bound yields that $\Pr[B_4] = \exp(-\Omega(s\overline{C}))$. Note that $\overline{C} \geq 1/2^i$, so that we in particular have that $\Pr[B_4] = \exp(-\Omega(s/2^i))$. Combining our bounds, it follows that for $s \geq 2^i$,

$$\Pr[A_s] = \exp\left(-\Omega\left(\frac{s}{2^i}\right)\right).$$

Now we can upper bound

$$\mathrm{E}[R] \leq 2^i + \sum_{j=0}^{\infty} \Pr[A_{2^{i+j}}]2^{i+j+1} = 2^i + 2^{i+1}\sum_{j=1}^{\infty} \exp\left(-\Omega(2^j)\right)2^j = O(2^i),$$

as desired. This completes the proof. □

D.3 Balls into Capacitated Bins

In this section we prove Theorem D.1. Let us start by recalling the setting of the theorem. We let $n, m \in \mathbb{N}$ and ε be given with $0 < \varepsilon < 1$ and suppose that we sequentially distribute n balls into m bins, each of capacity $C = (1+\varepsilon)n/m$. For simplicity, we assume that n, m

and ε are such that C is a positive integer. Each ball is placed in a uniformly random non-full bin, where a bin is *full* if it contains precisely C balls. The theorem claims that if $1/\varepsilon = m^{o(1)}$, then the expected fraction of non-full bins is $\Theta(f)$, where,

$$f = \begin{cases} \varepsilon C, & C \leq \log(1/\varepsilon) \\ \varepsilon\sqrt{C \log\left(\frac{1}{\varepsilon\sqrt{C}}\right)}, & \log(1/\varepsilon) < C \leq \frac{1}{2\varepsilon^2} \\ 1, & \frac{1}{2\varepsilon^2} \leq C. \end{cases}$$

To prove the theorem, we will take an alternative viewpoint on the distribution process. Instead of picking a non-full bin for each ball, we disregard the capacities and instead pick a uniformly random bin (full or non-full). Then a bin may receive more than C balls but if it does, we view it as having exactly C balls. To be precise, for $j \in [m]$, and $i \in \mathbb{Z}_{\geq 0}$, we denote by $X_j^{(i)}$ the number of balls in bin j after i balls have been placed. We further define $Y_j^{(i)} = \min(X_j^{(i)}, C)$. Let $T \in \mathbb{N}$ be minimal such that $\sum_{j\in[m]} Y_j^{(T)} = n$. Note that T is a random variable with $T \geq n$ and that $\Pr[T < \infty] = 1$. Further note that when the n balls are distributed into the m bins as in Theorem D.1, the joint distribution of balls in bins has the same distribution as $(Y_j^{(T)})_{i\in[m]}$. We will first prove concentration bounds on T and for this, we require Azuma's inequality.

Theorem D.6 (Azuma's inequality [Azu67])**.** *Suppose that $(X_i)_{i=0}^k$ is a martingale satisfying that $|X_{i+1} - X_i| \leq s_i$ almost surely for all $i = 0, \ldots, k-1$. Let $s = \sum_{i=1}^k s_i^2$. Then for any $t > 0$ it holds that*

$$\Pr(|X_k - X_0| \geq t) \leq 2\exp\left(\frac{-t^2}{2s}\right). \tag{D.2}$$

The concentration bound on T is as in the following lemma.

Lemma D.7. *For any $N \geq 2Cm$ and any $t > 0$ it holds that*

$$\Pr[|T - \mathrm{E}[T]| \geq t] \leq 2\exp\left(-\frac{t^2\varepsilon^2}{8N}\right) + m\exp(-N/(8m)).$$

Proof. For $i \in \mathbb{Z}_{\geq 0}$, we define $S_i \in [m]$ to be the randomly chosen bin for the i'th ball. We further define $\mathcal{F}_i = \sigma(S_1, \ldots, S_i)$ to be the σ-algebra generated by the random choices of bins for the first i balls. Finally, we put $X_i = \mathrm{E}[T|\mathcal{F}_i]$. Then $(X_i)_{i=0}^\infty$ is a martingale with $X_0 = \mathrm{E}[T]$. Now the random variable X_i is the expected value of T conditioned on the placements of the first i balls. We are going to prove that $|X_{i+1} - X_i| \leq \frac{1+\varepsilon}{\varepsilon}$ for each $i \geq 0$. To see this, fix i and write $n' = \sum_{j\in[m]} Y_j^{(i)}$. If $n' \geq n$, then $X_i = X_{i+1} = T$, so we may assume that $n' < n$, i.e., after distributing the first i balls we are still not done distributing the n balls into the capacitated bins. In this case, it trivially holds that $X_{i+1} \leq X_i + 1$ with equality holding if and only if the $(i+1)$'st ball is placed in a bin which is already full. On the other hand, we claim that $X_i \leq X_{i+1} + \frac{1+\varepsilon}{\varepsilon}$. To see this, let T' be minimal

such that $\sum_{j\in[m]} Y_j^{(T')} = n-1$ and let $T_i' = \max(T', i)$. From the assumption $n' < n$ it follows that $T_i' < T$ and we may write

$$X_i = \mathrm{E}[T_i'|\mathcal{F}_i] + \mathrm{E}[T - T_i'|\mathcal{F}_i].$$

Consider now any sequence of ball placements $s = (s_1, \ldots, s_\ell) \in [m]^\ell$ with $\ell > i$ satisfying that if $(S_1, \ldots, S_\ell) = s$, then $T = \ell$. Then, for any $s' \in [m]^\ell$ differing from s in at most the $(i+1)$'st coordinate, it holds that if $(S_1, \ldots, S_\ell) = s'$, then $T_i' \leq \ell$. From this it follows that $\mathrm{E}[T_i'|\mathcal{F}_i] \leq X_{i+1}$. We further claim that $\mathrm{E}[T - T_i'|\mathcal{F}_i] \leq \frac{1+\varepsilon}{\varepsilon}$. To see this, note that when placing n balls into m bins of capacity $C = (1+\varepsilon)n/m$, at least $\frac{\varepsilon}{1+\varepsilon}$ bins will be non-full regardless of the positions of the balls. Now $T - T_i'$ counts the number of times we have to select a random bin until we find a non-full bin. Therefore, $T - T_i'$ will be geometrically distributed with parameter $p \geq \frac{\varepsilon}{1+\varepsilon}$, and it follows that $\mathrm{E}[T - T_i' \mid \mathcal{F}_i] \leq \frac{1+\varepsilon}{\varepsilon}$. Combining our bounds, we conclude that

$$|X_{i+1} - X_i| \leq \frac{1+\varepsilon}{\varepsilon} \leq \frac{2}{\varepsilon}.$$

Plugging into Azuma's inequality, we see that for any $N \geq 0$ and any $t > 0$, it holds that

$$\Pr[|X_N - \mathrm{E}[T]| \geq t] = \Pr[|X_N - X_0| \geq t] \leq 2\exp\left(-\frac{t^2\varepsilon^2}{8N}\right).$$

Thus, for any $N \geq 0$,

$$\Pr[|T - \mathrm{E}[T]| \geq t] \leq \Pr[|X_N - \mathrm{E}[T]| \geq t] + \Pr[X_N \neq T] \leq 2\exp\left(-\frac{t^2\varepsilon^2}{8N}\right) + \Pr[N < T].$$

Suppose $N \geq 2Cm$. By a standard Chernoff bound it follows if N balls are distributed at random into m bins, the probability that a given bin receives less than C balls is upper bounded by $\exp(-N/(8m))$. Thus, we can trivially upper bound $\Pr[N < T] \leq m\exp(-N/(8m))$. Combining our bounds,

$$\Pr[|T - \mathrm{E}[T]| \geq t] \leq 2\exp\left(-\frac{t^2\varepsilon^2}{8N}\right) + m\exp(-N/(8m)),$$

as desired. □

Curiously, Lemma D.7 does not tell us anything about the value of $\mathrm{E}[T]$ and in fact, we will not need it when proving Theorem D.1. The bound in Lemma D.7 is a bit unwieldy, so below we state a corollary which is better suited for applications.

Corollary D.8. *Let $\gamma = O(1)$. If $C > \frac{3(1+\varepsilon)(1+\gamma)\log n}{\varepsilon^2}$, then $\Pr[T = n] = 1 - O(n^{-\gamma})$. Otherwise $|T - \mathrm{E}[T]| = O\left(\frac{\sqrt{m}\log n}{\varepsilon^2}\right)$ with probability $1 - O(n^{-\gamma})$, where the implicit constant in the O-notation depends on γ.*

Proof. Suppose first that $C > \frac{3(1+\varepsilon)(1+\gamma)\log n}{\varepsilon^2}$. Consider throwing n balls into m bins uniformly at random. Let X denote the number of balls landing in a given bin and $\mu = \mathrm{E}[X] = C/(1+\varepsilon)$. Then a standard Chernoff bound shows that

$$\Pr[X > C] = \Pr[X > (1+\varepsilon)\mu] \leq \exp(-\varepsilon^2\mu/3) \leq n^{-\gamma-1},$$

so the probability that any bin receives more than C ball is $O(n^{-\gamma})$ by a union bound. In particular $T = n$ with probability $1 - O(n^{-\gamma})$.

Suppose on the other hand that $C \leq \frac{3(1+\varepsilon)(1+\gamma)\log n}{\varepsilon^2} \leq \frac{6(1+\gamma)\log n}{\varepsilon^2}$. Applying Lemma D.7 with $N = \max(2Cm, 8m(\gamma+1)\log n)$, we obtain that

$$\Pr[|T - \mathrm{E}[T]| \geq t] = 2\exp\left(-\frac{t^2\varepsilon^2}{8N}\right) + n^{-\gamma}.$$

In particular $|T - \mathrm{E}[T]| = O(\sqrt{N\log n}/\varepsilon)$ with probability $1 - O(n^{-\gamma})$. The desired bound follows by observing that $N = O(\frac{m\log n}{\varepsilon^2})$. □

We need one further Lemma before proving Theorem D.1.

Lemma D.9. *Let $k \geq 0$ be fixed and define $Z = \sum_{j\in[m]} Y_j^{(k)}$. Then for any $t > 0$,*

$$\Pr[|Z - \mathrm{E}[Z]| \geq t] \leq 2\exp\left(-\frac{t^2}{2k}\right).$$

Proof. Let $S_1, S_2, \ldots$ and $\mathcal{F}_1, \mathcal{F}_2, \ldots$ be defines as in the proof of Lemma D.7. For $0 \leq i \leq k$, we define $Z_i = \mathrm{E}[Z \mid \mathcal{F}_i]$ so that $Z_0 = \mathrm{E}[Z]$ and $Z_k = Z$. Now it is easy to check that for $0 \leq i < k$ it holds that $|Z_{i+1} - Z_i| \leq 1$. Thus the desired result follows from Azuma's inequality. □

We will next prove Theorem D.1.

Proof of Theorem D.1. Note first, that if $\varepsilon = \Omega(1)$, then $f = \Theta(1)$, regardless of the relationship between ε and C. When placing n balls into m bins, each of capacity $C = (1+\varepsilon)n/m$, the fraction of non-full bins is at least $\varepsilon/(1+\varepsilon)$, regardless where the balls are placed. In the case $\varepsilon = \Omega(1)$, this is $\Theta(1)$, so Theorem D.1 is trivial. In the following, we may therefore assume that ε smaller than a sufficiently small constant.

We will again consider the alternative viewpoint where we throw an infinite sequence of balls uniformly at random into the bins. As before, we define $X_j^{(i)}$ to be the number of balls in bin j after throwing i balls, $Y_j^{(i)} = \min(X_j^{(i)}, C)$ and $T = \min(i \in \mathbb{N} : \sum_{j\in[m]} Y_j^{(i)} = n)$.

Let $\gamma > 1$ be a constant to be fixed. We are going to split the argument into three cases.

Case 1: $C \leq \gamma \log(1/\varepsilon)$**.** We will show that in this case, the expected fraction of non-full bins is $\Theta(\varepsilon C)$. To do this, we first show the following technical claim.

Claim D.10. *If* $C \leq \gamma \log(1/\varepsilon)$*, then* $\mathrm{E}[T] = (1 + \Omega(1))n$*.*

Proof of Claim. Fix a bin $j \in [m]$ and consider throwing $m \log(1/\varepsilon)/2$ balls into m bins. The probability that bin j is empty is

$$\left(1 - \frac{1}{m}\right)^{m \log(1/\varepsilon)/2} = \Omega(\sqrt{\varepsilon}).$$

As we will now argue, it follows that when throwing $N \geq m \log(1/\varepsilon)/2$ balls into m bins uniformly at random, a given bin receives at most $N/m - \log(1/\varepsilon)/2$ balls with probability $\Omega(\sqrt{\varepsilon})$. For this, we use the results of [GM14], stating that if $W \sim B(k, p)$ is binomially distributed with $p < 1 - 1/k$, then $\Pr[X \leq \mathrm{E}[X]] > 1/4$. Combining this result with the above, we obtain that the given bin receives none of the first $m \log(1/\varepsilon)/2$ balls with probability $\Omega(\sqrt{\varepsilon})$ and at most $(N - m \log(1/\varepsilon)/m = N/m - \log(1/\varepsilon)/2$ of the remaining balls with probability at least $1/4$. Moreover, these events are independent, happening simultaneously with probability $\Omega(\sqrt{\varepsilon})$, which gives the desired.

Now let $N = n + \log(1/\varepsilon)m/4$ and define $Z = \sum_{j \in [m]} Y_j^{(N)}$. From the above observation, it follows that

$$\mathrm{E}[Z] \leq Cm - \Omega(\sqrt{\varepsilon} \log(1/\varepsilon)m),$$

and by applying Lemma D.9 it follows that it similarly hold with high probability that $Z \leq Cm - \Omega(\sqrt{\varepsilon} \log(1/\varepsilon)m)$, with a potentially larger implicit constant in the Ω-notation. Assuming that ε is smaller than a sufficiently small constant we therefore have that with high probability,

$$Z \leq (C - \gamma\varepsilon \log(1/\varepsilon))m \leq C(1 - \varepsilon)m = (1 + \varepsilon)(1 - \varepsilon)n < n.$$

Thus $T > N$ with high probability, but this also means that

$$\mathrm{E}[T] \geq N = n + \frac{\log(1/\varepsilon)m}{4} \geq n + \frac{Cm}{4\gamma} = n\left(1 + \frac{1 + \varepsilon}{4\gamma}\right) = n(1 + \Omega(1)),$$

as desired. □

Using the claim and Corollary D.8 it follows that also $T = (1+\Omega(1))n$ with probability $1 - n^{-\gamma}$ for any constant γ and that $|T - \mathrm{E}[T]| = O\left(\frac{\sqrt{m} \log n}{\varepsilon^2}\right)$ with the same high probability.

We now choose $N = \mathrm{E}[T] + O\left(\frac{\sqrt{m} \log n}{\varepsilon^2}\right)$ so large that $\Pr[T \geq N] \leq n^{-2}$. Then $N = (1 + \Omega(1))n$ as well. Consider a bin $j \in [m]$ and let $A_k = [X_j^{(N)} = k]$ for each $k \geq 0$. Then

$$\Pr[A_k] = \binom{N}{k} \frac{1}{m^k} \left(1 - \frac{1}{m}\right)^{N-k}.$$

If $k = N^{1/2-\Omega(1)}$, then simple calculus yields that $\Pr[A_k]$ can be approximated with the Poisson distribution with mean $\mu = N/m$ as follows,

$$\Pr[A_k] = (1+o(1))\left(\frac{N}{m}\right)^k \frac{1}{k!}e^{-N/m} = (1+o(1))\frac{\mu^k}{k!}e^{-\mu}.$$

In particular, this holds when $k \le C$. Thus, for any $k \le C$ it holds that

$$\frac{\Pr[A_k]}{\Pr[A_{k-1}]} = (1+o(1))\frac{\mu}{k} = (1+\Omega(1))\frac{n}{km} \ge (1+\Omega(1))\frac{n}{Cm} = \frac{1+\Omega(1)}{1+\varepsilon} = 1+\Omega(1),$$

where the last inequality requires that ε is smaller than a sufficiently small constant which we may assume. Let $\alpha = \Omega(1)$ be the implicit constant in the Ω-notation above, such that for $k \le C$ (and $n, m, 1/\varepsilon$ sufficiently large), we have that $\Pr[A_k]/\Pr[A_{k-1}] \ge 1+\alpha$. It follows that,

$$\Pr[Y_j^{(N)} < C] = \sum_{k=1}^{C} \Pr[A_{C-k}] \le \sum_{k=1}^{C} (1+\alpha)^{k-1} \Pr[A_{C-1}] = O(\Pr[A_{C-1}]),$$

and

$$\mathrm{E}[C - Y_j^{(N)}] = \sum_{k=1}^{C} k \Pr[A_{C-k}] \le \sum_{k=1}^{C} k(1+\alpha)^{k-1} \Pr[A_{C-1}] = O(\Pr[A_{C-1}]).$$

It trivially holds that $\Pr[Y_j^{(N)} < C] \ge \Pr[A_{C-1}]$ and $\mathrm{E}[C - Y_j^{(N)}] \ge \Pr[A_{C-1}]$, so in fact we have proved that $\Pr[Y_j^{(N)} < C] = \Theta(\Pr[A_{C-1}])$ and $\mathrm{E}[C - Y_j^{(N)}] = \Theta(\Pr[A_{C-1}])$. By linearity of expectation,

$$\mathrm{E}\left[\sum_{j\in[m]} C - Y_j^{(N)}\right] = \Theta(m \Pr[A_{C-1}]) = \Theta(m \Pr[Y_j^{(N)} < C]). \tag{D.3}$$

Now with probability at least $1 - n^{-2}$, it holds that $N - O\left(\frac{\sqrt{m}\log n}{\varepsilon^2}\right) \le T \le N$. Since T is chosen such that $\sum_{j\in[m]} C - Y_j^{(T)} = \varepsilon n$, it follows that

$$\mathrm{E}\left[\sum_{j\in[m]} C - Y_j^{(N)}\right] = \Theta(\varepsilon n). \tag{D.4}$$

Thus, combining (D.3) and (D.4), we obtain that $\Pr[Y_j^{(N)} < C] = \Theta(\varepsilon C)$. Finally,

$$\Pr[Y_j^{(T)} < C] \ge \Pr[Y_j^{(N)} < C] - \Pr[N < T] = \Omega(\varepsilon C) - n^{-2} = \Omega(\varepsilon C).$$

Using the exact same argument but instead choosing $N = \mathrm{E}[T] - O\left(\frac{\sqrt{m}\log n}{\varepsilon^2}\right)$ so small that $\Pr[T \le N] \le n^{-2}$, we obtain that $\Pr[Y_j^{(T)} < C] = O(\varepsilon C)$, so in fact $\Pr[Y_j^{(T)} < C] = \Theta(\varepsilon C)$. But $\Pr[Y_j^{(T)} < C]$ is independent of j and is exactly the expected fraction of non-full bins. Thus the proof is complete in the case $C \le \gamma \log(1/\varepsilon)$.

Case 2: $\gamma \log(1/\varepsilon) < C \leq \frac{1}{\gamma\varepsilon^2}$**.** To make the argument work, we will assume that $\gamma = O(1)$ is sufficiently large. We can make this assumption since the argument from case 1 holds for any $\gamma = O(1)$. In general, the argument from case 1 serves as a nice warm up but for the present case we have to be more careful in our estimates. Again, we choose $N = \mathrm{E}[T] + O\left(\frac{\sqrt{m}\log n}{\varepsilon^2}\right)$ so large that $\Pr[T \geq N] \leq n^{-2}$ and put $\mu = N/m$. Let us state by proving some crude bounds on N as stated in the following claim.

Claim D.11. *If $\gamma = O(1)$ is sufficiently large, then $C + \sqrt{C} \leq N/m \leq 3C/2$.*

Proof. We first prove the lower bound. Suppose for contradiction that $N/m < C + \sqrt{C}$. Then $\mathrm{E}\left[\sum_{j\in[m]} C - Y_j^{(N)}\right] = \Omega(\sqrt{C}m) = \Omega(n/\sqrt{C}) = \Omega(n\gamma\varepsilon)$, so if γ is sufficiently large, $\mathrm{E}\left[\sum_{j\in[m]} C - Y_j^{(N)}\right] \geq 2\varepsilon n$ and this contradicts the fact that with high probability $T \leq N$. For the upper bound, note that if $N/m \geq 3C/2$, then for any $j \in [m]$,

$$\Pr[X_j^{(N)} \leq C] \leq \exp\left(-\frac{N}{18m}\right) \leq \exp\left(-\frac{C}{12}\right) \leq \exp\left(-\frac{\gamma\log(1/\varepsilon))}{12}\right) = \varepsilon^{\gamma/12} \leq \varepsilon^2$$

by a Chernoff bound and assuming $\gamma \geq 24$. Thus, $\mathrm{E}\left[\sum_{j\in[m]} C - Y_j^{(N)}\right] \leq \varepsilon^2 Cm \leq \varepsilon n/2$, where the last inequality assumes that ε is sufficiently small. Again this contradicts the fact that with high probability $T \geq N - O\left(\frac{\sqrt{m}\log n}{\varepsilon^2}\right)$ □

As before, we consider a bin $j \in [m]$ and define $\Pr[A_k] = \Pr[X_j^{(N)} = k]$. Then for $k \leq C$,

$$\frac{\Pr[A_k]}{\Pr[A_{k-1}]} = \frac{\binom{N}{k}}{\binom{N}{k-1}}\frac{1}{m-1} = \frac{N-k+1}{k}\frac{1}{m-1} = \frac{\mu}{k}\frac{m}{m-1}\frac{N-k+1}{N} = \frac{\mu}{k}\left(1 \pm O(1/m)\right).$$

It follows from the claim that $\mu/k \geq 1 + 1/\sqrt{C}$ for $k \leq C$. By our assumptions $C \leq 1/\varepsilon^2 = m^{o(1)}$ and thus $\Pr[A_k]/\Pr[A_{k-1}] = (\mu/k)^{1\pm o(1)}$. Let $\alpha \in \mathbb{N}$ be minimal satisfying that $\Pr[A_{C-1}]/\Pr[A_{C-\alpha}] \geq 2$. Using the crude bounds in the claim and simple calculations we obtain that $\alpha = \Theta(1/\log(\mu/C))$. Now,

$$\Pr[Y_j^{(N)} < C] = \Theta(\alpha \Pr[A_{C-1}]) = \Theta(\alpha \Pr[A_C]), \tag{D.5}$$

and

$$\mathrm{E}[C - Y_j^{(N)}] = \Theta(\alpha^2 \Pr[A_{C-1}]) = \Theta(\alpha^2 \Pr[A_C]). \tag{D.6}$$

As in case 1, $\Pr[Y_j^{(T)} < C] = \Theta(\Pr[Y_j^{(N)} < C])$ which is the the value we are looking for. Thus, if we can find the value of α, eq. (D.5) will give us the result we are looking for. The problem is that α depends of N and hence of $\mathrm{E}[T]$ which we as of now don't know the value of. However, we know that $\mathrm{E}[C - Y_j^{(N)}]$ is close to εn, so on a high level we can plug this into eq. (D.6) and solve for α.

Let us make the above argument precise. First, we write $\mu = C + \beta$ noting that by the claim, $\sqrt{C} \leq \beta \leq C/2$. Note for later use that $\alpha = \Theta\left(\frac{1}{\log(\mu/C)}\right) = \Theta\left(\frac{C}{\beta}\right)$. Using the Poisson approximation,

$$\Pr[A_C] = (1 + o(1))\frac{\mu^C}{C!}e^{-\mu} = \Theta\left(\left(\frac{\mu}{C}\right)^C \frac{1}{\sqrt{C}e^{\beta}}\right) = \Theta\left(\left(1 + \frac{\beta}{C}\right)^C \frac{1}{\sqrt{C}e^{\beta}}\right).$$

Write $f(x) = \log(1 + x)$, so that $\exp(f(\beta/C)) = 1 + \beta/C$. As $\beta/C \leq 1/2$, we can use a Taylor expansion to conclude that

$$f\left(\frac{\beta}{C}\right) = f(0) + f'(0)\frac{\beta}{C} - \Theta\left(f''(0)\left(\frac{\beta}{C}\right)^2\right) = \frac{\beta}{C} - \Theta\left(\left(\frac{\beta}{C}\right)^2\right).$$

Write $\Delta = \beta - Cf(\beta/C)$, so that $\Delta = \Theta(\beta^2/C) = \Theta(C/\alpha^2)$. Then

$$\Pr[A_C] = \Theta\left(\frac{1}{\sqrt{C}e^{\Delta}}\right).$$

On the other hand, it follows from Corollary D.8 that with high probability

$$\sum_{j\in[m]} C - Y_j^{(N)} = \Theta\left(\sum_{j\in[m]} C - Y_j^{(T)}\right) = \Theta(\varepsilon n),$$

so that, $\mathrm{E}[C - Y_j^{(N)}] = \Theta(\varepsilon C)$. Plugging all this into eq. (D.6), we find that

$$\frac{\alpha^2}{\sqrt{C}e^{\Delta}} = \Theta(\varepsilon C).$$

Using that $\alpha^2 = \Theta(C/\Delta)$, this reduces to $\Delta e^{\Delta} = \Theta\left(\frac{1}{\varepsilon\sqrt{C}}\right)$, so that $\Delta = \Theta\left(\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)\right)$, and thus,

$$\alpha = \Theta\left(\sqrt{C/\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)}\right).$$

Combining eq. (D.5) and eq. (D.6), we find that,

$$\Pr[Y_j^{(N)} < C] = \Theta(\mathrm{E}[C - Y_j^{(N)}]/\alpha) = \Theta\left(\varepsilon\sqrt{C}\sqrt{\left(\log\frac{1}{\varepsilon\sqrt{C}}\right)}\right).$$

A similar argument to that used in the first case shows that also $\Pr[Y_j^{(T)} < C] = \Theta(\Pr[Y_j^{(N)} < C]) = \Theta(f)$ which completes the proof.

Case 3: $C > \frac{1}{\gamma\varepsilon^2}$**.** We can reduce this case to case 2 as follows. Define the function, $f : \mathbb{R} \to \mathbb{R}$ by $f(x) = x(x - n/m)^2$. Then $f(n/m) = 0$ and $f(C) = C\varepsilon^2(n/m)^2 > \frac{(n/m)^2}{\gamma}$, so there exists $n/m < \hat{C} < C$ satisfying that $f(\hat{C}) = \frac{(n/m)^2}{\gamma}$. Let $\hat{\varepsilon}$ be such that $\hat{C} = (1 + \hat{\varepsilon})n/m$, so that $0 < \hat{\varepsilon} < \varepsilon$. Then $f(\hat{C}) = \hat{C}\hat{\varepsilon}^2(n/m)^2$ which implies that $\hat{C} = \frac{1}{\gamma\hat{\varepsilon}^2}$. Now define $\hat{Y}_j^{(i)} = \min(X_j^{(i)}, \hat{C})$ and $\hat{T} = \min(i \in \mathbb{N} : \sum_{j\in[m]} \hat{Y}_j^{(i)} = n)$. As $\hat{C} \leq C$, it follows that $\hat{T} \geq T$. We can now apply the result from Case 2 to conclude that

$$\Pr[Y_j^{(T)} < C] \geq \Pr[Y_j^{(\hat{T})} < C] = \Omega(1),$$

which completes the proof. □

D.4 Some Helpful Lemmas

In this section, we provide two helpful lemmas which will be useful in several of the later sections. The first is a tail bound for sums of geometric random variables, and the second can be seen as a high probability upper bound on the number of bins visited at a given level during an insertion with consistent hashing with bounded loads and virtual bins.

D.4.1 A Tail Bound for Sums of Geometric Variables

Recall that we say that Y is geometrically distributed with parameter p if for non-negative integers k it holds that $\Pr[Y = k] = p^k(1 - p)$. Then $\mathrm{E}[Y] = \frac{p}{1-p}$ and $\mathrm{Var}[Y] = \mathrm{E}[Y](1 + \mathrm{E}[Y])$. Let $(X_i)_{i\in[n]}$ be independent random variables such that X_i is geometrically distributed with parameter p_i. Let $X = \sum_{i\in[n]} X_i$. Define $\mu_i = \mathrm{E}[X_i]$, $\sigma_i^2 = \mathrm{Var}[X_i] = \mu_i(1+\mu_i)$, $\mu = \sum_{i\in[n]} \mu_i$, and $\sigma^2 = \sum_{i\in[n]} \sigma_i^2$. Finally let $W_0 : [0, \infty) \to \mathbb{R}$ be the Lambert function defined by $W_0(x)e^{W_0(x)} = x$. We have the following theorem.

Theorem D.12. *For any $t \geq 0$ it holds that*

$$\Pr\left[X \geq \mu + 4t\sigma^2\right] \leq \begin{cases} e^{-2\sigma^2 t W_0(t)}, & \text{if } t \leq \left(1 + \frac{1}{2\mu_0}\right)\log\left(1 + \frac{1}{2\mu_0}\right) \\ \left(1 - \frac{1}{1+2\mu_0}\right)^{2\sigma^2 t}, & \text{if } t > \left(1 + \frac{1}{2\mu_0}\right)\log\left(1 + \frac{1}{2\mu_0}\right) \end{cases}. \tag{D.7}$$

Proof. The idea of the proof is standard and uses the moment generating function of X. Let $0 \leq \lambda \leq \log\left(1 + \frac{1}{2\mu_0}\right)$ be a parameter which we will fix later. Then

$$\mathrm{E}\left[e^{\lambda(X_i - \mu_i)}\right] = \frac{e^{-\lambda\mu_i}}{1 - \mu_i(e^\lambda - 1)} = e^{-\lambda\mu_i - \log\left(1 - \mu_i(e^\lambda - 1)\right)}.$$

Define $f(\lambda) = -\lambda\mu_i - \log\left(1 - \mu_i(e^\lambda - 1)\right)$. Using a Taylor expansion,

$$f(\lambda) \leq f(0) + f'(0)\lambda + \frac{\max_{0\leq x\leq\lambda} f''(x)}{2}\lambda^2$$

It is easy to check that $f(0) = 0$, $f'(0) = 0$, and $f''(\lambda) = \sigma_i^2 \frac{e^\lambda}{(1-\mu_i(e^\lambda-1))^2}$. Now using that $\lambda \leq \log\left(1+\frac{1}{2\mu_0}\right)$ we get that $f''(\lambda) \leq 4\sigma_i^2 e^\lambda$, and hence

$$f(\lambda) \leq 2\sigma_i^2 e^\lambda \lambda^2.$$

We now use Markov's inequality to conclude that

$$\begin{aligned}\Pr\left[X \geq \mu + 4t\sigma^2\right] &= \Pr\left[e^{\lambda\sum_{i\in[n]}(X_i-\mu_i)} \geq e^{\lambda 4t\sigma^2}\right] \\ &\leq \frac{\prod_{i\in[n]} \mathrm{E}\left[e^{\lambda(X_i-\mu_i)}\right]}{e^{\lambda 4t\sigma^2}} \\ &\leq e^{2\sigma^2 e^\lambda\lambda^2 - 4\lambda t\sigma^2}\ .\end{aligned}$$

We will set $\lambda = \min\left\{\log\left(1+\frac{1}{2\mu_0}\right), W_0(t)\right\}$. Now, if $t \leq \left(1+\frac{1}{2\mu_0}\right)\log\left(1+\frac{1}{2\mu_0}\right)$ then $\lambda = W_0(t)$. This implies that,

$$\Pr\left[\sum_{i\in[n]} X_i \geq \mu + 4t\sigma^2\right] \leq e^{2\sigma^2 e^\lambda\lambda^2 - 4\lambda t\sigma^2} = e^{-2\sigma^2 t W_0(t)}\ .$$

On the other hand, if $t > \left(1+\frac{1}{2\mu_0}\right)\log\left(1+\frac{1}{2\mu_0}\right)$ then $\lambda = \log\left(1+\frac{1}{2\mu_0}\right)$. This implies that,

$$\Pr\left[\sum_{i\in[n]} X_i \geq \mu + 4t\sigma^2\right] \leq e^{2\sigma^2 e^\lambda\lambda^2 - 4\lambda t\sigma^2} \leq e^{-2\sigma^2\log\left(1+\frac{1}{2\mu_0}\right)t} = \left(1 - \frac{1}{1+2\mu_0}\right)^{2\sigma^2 t}$$

□

Defining $\mathcal{C} : [0,\infty) \to \mathbb{R}$ by $\mathcal{C}(x) = (1+x)\log(1+x) - x$, it follows from standard calculus that $\mathcal{C}(x) = \Theta(xW_0(x))$. In particular, the first bound in (D.7) takes the form

$$\Pr\left[X \geq \mu + 4t\sigma^2\right] = e^{-\Omega(\sigma^2\mathcal{C}(t))}.$$

Up to the constant delay in the exponential decrease, this is the same as the standard variance-based Chernoff bound for the sum of independent variables in $[0,1]$. Intuitively, the second bound of (D.7) corresponds to the event that the heaviest of the geometric variables, X_0, satisfies $X_0 = \mu_0 + \Omega(\sigma^2 t)$.

D.4.2 A High Probability Upper Bound on the Run Length at a Level

We next prove the general lemma on consistent hashing with bounded loads and virtual bins. Consider a bin b at level i that may be chosen dependently on the hashing of balls and bins to levels $1,\ldots,i-1$. We prove that if I is a maximal interval of level i containing b satisfying that all bins in I get full after the hashing of balls to levels $1,\ldots,i$, then with

probability $1-\delta$ the number of bins in I is $O(\log(1/\delta)/\varepsilon)$. This bound is quite crude but we require it for both the analyses of Section D.5 and Section D.6 which proceed by step by step revealing the history of how a bin obtained its balls at a given level. The result entails that at a given point in the process, we have only revealed an insignificant part of the system. On a high level, this means that even conditioning on what we already know about the system, the probabilities of the various relevant events only change very slightly. The result is as follows.

Lemma D.13. *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$ with $1/\varepsilon = n^{o(1)}$. Suppose we distribute n balls into m bins, each of capacity $C = (1+\varepsilon)n/m$, using consistent hashing with bounded loads and virtual bins and $k \geq 2/\varepsilon$ levels. Let b be a bin at level i which may be chosen dependently on the hashing of balls and bins to level $1, \ldots, i-1$. Let I be a maximal interval at level i containing b such that all bins lying in I are full after the hashing to level $1, \ldots, i$. Let $1/n^{O(1)} < \delta \leq 1/2$. The number of bins in I is $O(\log(1/\delta)/\varepsilon)$ with probability at least $1-\delta$.*

Proof. The proof is very similar to the proof of Theorem D.5, so we just provide a sketch of the proof. We may clearly assume that $i = k$ as this can only decrease the remaining capacities of the bins. Let $C_1, \ldots, C_m$ be the remaining capacities and $\overline{C} = \frac{1}{m}\sum_{i\in[m]} C_i$. Using a standard Chernoff bound and the assumptions that $k \geq 2/\varepsilon$ and $1/\varepsilon = n^{o(1)}$, we find that the number of balls hashing to level k is at most εn with probability $1 - O(n^{-\gamma})$ for any $\gamma = O(1)$. Letting X denote the number of such balls, it follows that $m\overline{C} \geq 2X$ with the same high probability. Condition on this event and let R denote the number of bins in I. For a given s, we find as in the proof of Theorem D.5, that there exists a constant number of intervals $I_1, \ldots, I_\ell$, all of length $\Theta(s/m)$ Such that the following holds. Let $X_j^{(1)}$, $X_j^{(2)}$, and $X_j^{(3)}$ denote respectively the number of bins, total capacity of bins, and number of balls hashing to I_j. Let further $\mu_j^{(1)} = \mathrm{E}[X_j^{(1)}]$, $\mu_j^{(2)} = \mathrm{E}[X_j^{(2)}]$, and $\mu_j^{(3)} = \mathrm{E}[X_j^{(3)}]$. If $s+1 \leq R \leq 2s$, then there is a $j \in [\ell]$ such that either

$$B_1\colon\ |X_j^{(1)} - \mu_j^{(1)}| = \Omega(\mu_j^{(1)}),$$

$$B_2\colon\ |X_j^{(2)} - \mu_j^{(2)}| = \Omega(\mu_j^{(2)}),$$

$$B_3\colon\ X_j^{(3)} - \mu_j^{(3)} = \Omega(\max(\mu_j^{(3)}, \overline{C}s)).$$

Note that $\mu_j^{(1)} = \Theta(s)$ and $\mu_j^{(2)} = \Theta(\overline{C}s)$. It therefore follows from standard Chernoff bounds that $\Pr[B_1] = \exp(-\Omega(s))$, $\Pr[B_2] = \exp(-\Omega(\overline{C}s/C))$, and $\Pr[B_3] = \exp(-\Omega(\overline{C}s))$. As $C = (1+\varepsilon)n/m$, we always have that $\overline{C} \geq \varepsilon n/m \geq \varepsilon C/2$. Therefore, we obtain the combined bound

$$\Pr[s+1 \leq R \leq 2s] = \exp(-\Omega(\varepsilon s)).$$

With $t = O(\log(1/\delta)/\varepsilon)$ sufficiently large, it follows that

$$\begin{aligned}\Pr[R' \geq t+1] &\leq \sum_{i=0}^{\infty} \Pr[A_{t2^i}] \leq \sum_{i=0}^{\infty} \exp(-\Omega(t2^i\varepsilon)) \\ &\leq \sum_{i=0}^{\infty} \exp(-\log(2/\delta)2^i) \leq \sum_{i=0}^{\infty} (\delta/2)^{2^i} \leq \delta.\end{aligned}$$

This completes the proof. □

Remark D.14. We will use the bound of Lemma D.13 to obtain the results in Section D.5 showing the concentration of the fraction of non-full bins around its mean $\mu = \Theta(f)$. In fact, this allows us to prove a stronger version of Lemma D.13 in Section D.6 which bounds the number of bins in I by $O(\log(1/\delta)/f)$, the only caveat being that here we have to use $k \geq 1/\varepsilon^2$ levels.

We finish the section with the following definition.

Definition D.15. For I as in the lemma above, we will call I the *run* at level i containing b.

Lemma D.13 shows that the number of bins in the run is $O(\frac{\log(1/\delta)}{\varepsilon})$ with probability $1-\delta$. It in particular follows that the number of bins visited at level i during an insertion is $O(\frac{\log(1/\delta)}{\varepsilon})$ with probability $1-\delta$. Indeed, if b is a bin encountered during the insertion which is not full at level $i-1$, then all the bins encountered at level i lie in the run at level i containing b.

D.5 Non-Full Bins: In Expectation and with Concentration

In this section we will show that with consistent hashing, for each level $d \in [k]$ and each and each $0 \leq t \leq C$, the number of bins at level d containing at most t balls is sharply concentrated around its mean. This goal is achieved in Appendix D.5.1. Next, in Appendix D.5.2 we prove that with $k \geq c/\varepsilon^2$ levels for a sufficiently large constant c, the expected number of non-full bins at the highest level $k-1$, is $\Theta(f)$ where f is as defined in Equation (D.1).

D.5.1 High Probability Bounds on the Number of Non-Full Bins

The goal of this section is to prove the first part of Theorem D.4. For this, we first require some notation. We define

$X_d^{(j)}$ The remaining capacity of bin j after distributing balls to all levels $i \leq d$.

$Y_d^{(j)}$ The number of balls landing in or forwarded by bin j at level d.

$Z_{d,s}^{(j)}$ The capacity of the bin s places before bin j just before the hashing of balls to level d.

$W_{d,s}^{(j)}$ The number of balls landing between the bins placed s and $s+1$ places before bin j at level i.

There are some important relations between the variables. $Y_d^{(j)}$ can be expressed in terms of $W_{d,s}^{(j)}$ and $Z_{d,s}^{(j)}$ as follows $Y_d^{(j)} = W_{d,0}^{(j)} + \max\left\{0, \max_{l\geq 1} \sum_{s=1}^{l}(W_{d,s}^{(j)} - Z_{d,s}^{(j)})\right\}$. Similarly, we can express $X_d^{(j)}$ in terms of $Y_d^{(j)}$ as follows $X_d^{(j)} = \max\left\{0, C - \sum_{i\leq d} Y_i^{(j)}\right\}$.

Now due to all the dependencies in the system, it is unwieldy to analyse it directly. Instead, we will analyse a simpler system which we then show can give us high probability bounds for $\sum_{j\in[m]}[X_d^{(j)} \leq t]$ for each $t \leq C$. First we define $\mathcal{X}_0^{(j)} = C$ for every bin j. We then define $\mathcal{X}_d^{(j)}$ for $0 < d < k$ recursively as follows: First define independent random variables $\mathcal{Z}_{d,s}^{(j)}$ and $\mathcal{W}_{d,s}^{(j)}$ for every bin j and every integer s by

$$\Pr\left[\mathcal{Z}_{d,s}^{(j)} = t_1\right] = \Pr\left[\mathcal{X}_{d-1}^{(j)} = t_1\right] \tag{D.8}$$

$$\Pr\left[\mathcal{W}_{d,s}^{(j)} = t_2\right] = \left(\frac{n/k}{n/k+m}\right)^{t_2} \frac{m}{n/k+m} \tag{D.9}$$

for every integers $0 \leq t_1 \leq C$ and $0 \leq t_2$. So $\mathcal{W}_{d,s}^{(j)}$ is geometrically distributed with parameter $\frac{n/k}{n/k+m}$. We then define $\mathcal{Y}_d^{(j)} = \mathcal{W}_{d,0}^{(j)} + \max\left\{0, \max_{l\geq 1} \sum_{s=1}^{l}(\mathcal{W}_{d,s}^{(j)} - \mathcal{Z}_{d,s}^{(j)})\right\}$ and finally $\mathcal{X}_d^{(j)} = \max\left\{0, C - \sum_{i=1}^{d} \mathcal{Y}_i^{(j)}\right\}$.

Clearly, the two systems have a lot of similarities. $\mathcal{X}_d^{(j)}$ and $\mathcal{Y}_d^{(j)}$ are defined analogously to how $X_d^{(j)}$ and $Y_d^{(j)}$ are defined. The difference between the two system is the difference between variables the $\mathcal{Z}_{d,s}^{(j)}$, $\mathcal{W}_{d,s}^{(j)}$ and the variables $Z_{d,s}^{(j)}$, $W_{d,s}^{(j)}$. Our goal is to show that two systems are in fact very comparable, yet leverage that the second system is much simpler to analyse due to the independence. This approach leads to the theorem below which provides concentration of $\sum_{j\in[m]}[X_i^{(j)} \leq t]$ around $m \Pr\left[\mathcal{X}_i^{(j)} \leq t\right]$.

Theorem D.16. *Let n and m be positive integers and set $\mu = \frac{n}{m}$. Let $0 \leq \varepsilon \leq 1$ be such that $C = (1+\varepsilon)\mu$ is in integer. If $\mu = m^{o(1)}$ and $\varepsilon = m^{o(1)}$, then with probability at least $1 - m^{-\gamma}$ we have that*

$$\left|\frac{\sum_{j\in[m]}[X_i^{(j)} \leq t]}{m} - \Pr\left[\mathcal{X}_i^{(j)} \leq t\right]\right| \leq m^{-1/2+o(1)} \tag{D.10}$$

for all levels $1 \leq i \leq k$ and all $0 \leq t \leq C$. The constant in the big-O notation depends on γ.

We define A_d to be the event that

$$\left| \frac{\sum_{j\in[m]}[X_i^{(j)} \le t]}{m} - \Pr\left[\mathcal{X}_i^{(j)} \le t\right] \right| \le m^{-1/2+o(1)} \tag{D.11}$$

for all $1 \le i \le d$ and all $0 \le t \le C$. The goal of Theorem D.16 is prove that $\Pr[A_k] \ge 1 - m^{-\gamma}$. An important step of the proof is the following lemma.

Lemma D.17. *Fix $0 \le t \le C$, $j \in [m]$, and a subset $S \subseteq [m] \setminus \{j\}$ of $l \le O(\log m)$ bins. Then*

$$\left| \Pr\left[Y_d^{(j)} \ge t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\mathcal{Y}_d^{(j)} \ge t\right] \right| \le m^{-1/2+o(1)} \tag{D.12}$$

We also need a couple of auxiliary lemmas. The first is a simple consequence of Lemma D.13:

Lemma D.18. *With probability at least $1 - m^{-\gamma}$ we have that the longest run at level d is at most $O((\log m)/\varepsilon)$.*

We will also need a bound on the number of balls between consecutive bins.

Lemma D.19. *With probability at least $1-m^{-\gamma}$ there are no more than $O(\log m(\mu/k+1))$ balls between any two consecutive virtual bins on level d.*

Proof. This is simple observation since the probability that is there lands l balls between consecutive virtual bins is at most

$$\prod_{i=0}^{l-1} \frac{n/k - i}{n/k - i + m} \le \left(1 - \frac{m}{n/k+m}\right)^l \le \exp\left(-\frac{m}{n/k+m} l\right)$$

It is now clear that if $l = \Theta(\log m(\mu/k+1))$ that there are no consecutive virtual bins which receives more that l balls with probability $1 - m^{-\gamma}$. □

The final lemma is a technical lemma which we will use to get tail bounds. The proof is deferred to the end of the section.

Lemma D.20. *Let $B_1, \ldots B_n$ be Bernoulli variables, $0 \le \delta \le 1$ a small real, and $r > 0$ be an even integer. Assume that for any $i \in [n]$ and any subset $S \subseteq [n] \setminus \{i\}$ of size at most $r-1$ we have that $|\Pr[B_i = 1 \mid (B_j)_{j\in S}] - p| \le \delta$, then*

$$\mathrm{E}\left[\left(\sum_{i\in[n]} (B_i - p)\right)^r\right]^{1/r} \le \delta n + O(\sqrt{rn}) \,,$$

and the following tail bound holds

$$\Pr\left[\left|\sum_{i\in[n]} (B_i - p)\right| \ge \delta n + r\sqrt{n}\right] \le \exp(-\Omega(r)) \,. \tag{D.13}$$

We will now prove Lemma D.17.

Proof of Lemma D.17. Let B_d be the event that the longest run on the level d is at most $r = O(\log m / \Pr\left[\mathcal{X}_d^{(j)} > 0\right]) \leq O(\log(m)/\varepsilon)$ and that there are at most $O(\log m(\mu/k + 1))$ balls between any two consecutive virtual bins on level d. By Lemma D.18 and Lemma D.19 we have that $\Pr[\neg B_d \,|\, A_{d-1}] \leq m^{-\gamma'}$ hence we get that

$$\left|\Pr\left[Y_d^{(j)} \geq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[Y_d^{(j)} \geq t \wedge B_d \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right]\right| \leq m^{-\gamma'}$$

The important observation now is that

$$Y_d^{(j)} = W_{d,0}^{(j)} + \max\left\{0, \max_{1\leq l\leq r} \sum_{s=1}^{l} (W_{d,s}^{(j)} - Z_{d,s}^{(j)})\right\}$$

when B_d is true. So we only reveal r virtual bins and at most $O(rO(\log m(\mu/k+1)))$ balls when determining $Y_d^{(j)}$.

We will introduce a third system which will act as an intermediate between the two systems. Let $\overline{X}_{d-1}^{(j)}$ be independent random variables where each of the variables has the same marginal distribution as $X_{d-1}^{(j)}$. Let $\overline{Z}_{d,s}^{(j)}$ and $\overline{W}_{d,s}^{(j)}$ be independent random variables where each of $\overline{Z}_{d,s}^{(j)}$ has the same marginal distribution as $Z_{d,s}^{(j)}$, and each of $\overline{W}_{d,s}^{(j)}$ is geometrically distributed with parameter $\frac{n/k}{n/k+m}$. We then define $\overline{Y}_d^{(j)} = \overline{W}_{d,0}^{(j)} + \max\left\{0, \max_{1\leq l\leq r} \sum_{s=1}^{l} (\overline{W}_{d,s}^{(j)} - \overline{Z}_{d,s}^{(j)})\right\}$. The difference between the intermediate system and the original system is that in the intermediate system we are sampling everything with replacement and in the original system everything is sampled without replacement.

Let D be the event that $X_{d-1}^{(j)}$ is a distinct bin from the bins $(Z_{d,s}^{(i)})_{i\in S, 1\leq s\leq r}$ and that the bins $(X_{d-1}^{(i)})_{i\in S}$ are distinct for the bins $(X_{d,s}^{(j)})_{1\leq s\leq r}$. It is easy to see that $\Pr\left[\neg D \wedge B \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] \leq O(\frac{rl}{m-(l+1)(r+1)}) = m^{-1+o(1)}$, hence we get that

$$\left|\Pr\left[Y_d^{(j)} \geq t \wedge B_d \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[Y_d^{(j)} \geq t \wedge B_d \wedge D \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right]\right| \leq m^{-1+o(1)}\,.$$

It is standard fact that if we sample $\overline{X}_{d-1}^{(j)}$ and $(\overline{Z}_{d,s}^{(j)})_{1\leq s\leq r}$ independently with replacement and condition on them sampling distinct bins which are also distinct from the bins for $(X_{d-1}^{(i)})_{i\in S}$ and $(Z_{d,s}^{(i)})_{i\in S, 1\leq s\leq r}$, then it has the same distribution as sampling without replacement. The probability that we make such a sampling error is bounded by $\frac{l(r+1)^2}{m} = m^{-1+o(1)}$.

Similarly, if we sample $(\overline{W}_{d,s}^{(j)})_{0\leq s\leq r}$ independently with replacement conditioned on all balls being distinct and distinct from the ball sampled for $(W_{d,s}^{(j)})_{i\in S, 0\leq s\leq r}$, then it has the same distribution as sampling without replacement. With probability $1 - m^{-\gamma}$ we

have that $\overline{W}_{d,s}^{(j)} \leq O(\log m(\mu/k+1))$ for all $0 \leq s \leq r$, hence the probability of making a sampling error with balls is bounded by

$$m^{-\gamma} + O\left(\frac{\log(m)^2(\mu/k+1)^2(r+1)^2}{n_e}\right) = m^{-\gamma} + O\left(\frac{k\log(m)^2(\mu/k+1)^2(r+1)^2}{n}\right) = m^{-1+o(1)} .$$

From this two facts we see that

$$\left|\Pr\left[Y_d^{(j)} \geq t \wedge B_d \wedge D \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\overline{Y}_d^{(j)} \geq t \wedge B_d \wedge D \,\middle|\, A_{d-1}\right]\right| \leq m^{-1+o(1)} .$$

Since $B_d \wedge D$ happens with probability at least $1 - m^{-\gamma}$ we get that,

$$\left|\Pr\left[\overline{Y}_d^{(j)} \geq t \wedge B_d \wedge D \,\middle|\, A_{d-1}\right] - \Pr\left[\overline{Y}_d^{(j)} \geq t \,\middle|\, A_{d-1}\right]\right| \leq m^{-1+o(1)} .$$

We then define $\overline{\mathcal{Y}}_d^{(j)} = \mathcal{W}_{d,0}^{(j)} + \max\left\{0, \max_{1\leq l\leq r}\sum_{s=1}^{l}(\mathcal{W}_{d,s}^{(j)} - \mathcal{Z}_{d,s}^{(j)})\right\}$. The difference between $\overline{\mathcal{Y}}_d^{(j)}$ and $\mathcal{Y}_d^{(j)}$ is that $\overline{\mathcal{Y}}_d^{(j)}$ looks at at most r bins in the tail while $\mathcal{Y}_d^{(j)}$ looks at all bins in the tail. If $\overline{Z}_{d,s}^{(j)} = \mathcal{Z}_{d,s}^{(j)}$ for all $1 \leq s \leq r$ then $\Pr\left[\overline{Y}_d^{(j)} \geq t \,\middle|\, A_{d-1}\right] = \Pr\left[\overline{\mathcal{Y}}_d^{(j)} \geq t\right]$. This observation imply that

$$\begin{aligned}
&\left|\Pr\left[\overline{Y}_d^{(j)} \geq t \,\middle|\, A_{d-1}\right] - \Pr\left[\overline{\mathcal{Y}}_d^{(j)} \geq t\right]\right| \\
&\leq \sum_{\tau_1,\ldots,\tau_r} \left|\Pr\left[(\overline{Z}_{d,1}^{(j)},\ldots,\overline{Z}_{d,r}^{(j)}) = (\tau_1,\ldots,\tau_r) \,\middle|\, A_{d-1}\right] - \Pr\left[(\mathcal{Z}_{d,1}^{(j)},\ldots,\mathcal{Z}_{d,r}^{(j)}) = (\tau_1,\ldots,\tau_r)\right]\right| \\
&\leq \sum_{\tau_1,\ldots,\tau_r} \left|\prod_{1\leq s\leq r} \Pr\left[\overline{Z}_{d,s}^{(j)} = \tau_s \,\middle|\, A_{d-1}\right] - \prod_{1\leq s\leq r} \Pr\left[\mathcal{Z}_{d,s}^{(j)} = \tau_s\right]\right| \\
&\leq 2\sum_{0\leq\tau\leq C} \left|\Pr\left[\overline{Z}_{d,1}^{(j)} = \tau \,\middle|\, A_{d-1}\right] - \Pr\left[\mathcal{Z}_{d,1}^{(j)} = \tau\right]\right| \\
&\leq r\sum_{0\leq\tau\leq C} m^{-1/2+o(1)} \\
&\leq m^{-1/2+o(1)}
\end{aligned}$$

Now the same arguments as in the proof of Lemma D.18 show that $\mathcal{Y}_d^{(j)} = \overline{\mathcal{Y}}_d^{(j)}$ with probability at least $1 - m^{-\gamma}$. Combining all these bounds proves the claim. □

Now having proved Lemma D.17 we are ready to prove Theorem D.16.

Proof of Theorem D.16. We note that

$$\Pr[A_k] = \Pr\left[\bigwedge_{1\leq i\leq k} A_i\right] = \prod_{1\leq i\leq k} \Pr[A_i \mid A_{i-1}]$$

If we can prove that $\Pr[A_d \,|\, A_{d-1}] \geq 1 - O(m^{-\gamma'})$ for all $1 \leq d \leq k$, where γ' is an appropriately chosen constant, then we would get that

$$\Pr[A_k] \geq (1 - m^{-\gamma'})^k \geq 1 - m^{-\gamma}$$

The rest of the proof is now to show that $\Pr[\neg A_d \,|\, A_{d-1}] \leq O(m^{-\gamma'})$.

Let $j \in [m]$ and $S \subseteq [m] \setminus \{s\}$ with $l = |S| \leq O(\log m)$. We will prove that,

$$\left|\Pr\left[X_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i \in S}\right] - \Pr\left[\mathcal{X}_d^{(j)} \leq t\right]\right| \leq m^{-1/2+o(1)} \,. \tag{D.14}$$

This will imply the result since if we combine eq. (D.14) with Lemma D.20 we get that,

$$\Pr\left[\left|\frac{\sum_{j \in [m]} [X_d^{(j)} \leq t]}{m} - \Pr\left[\mathcal{X}_d^{(j)} \leq t\right]\right| \geq m^{-1/2+o(1)} \,\middle|\, A_{d-1}\right] \leq m^{-\gamma''} \,.$$

For $0 \leq t \leq C$. Now a union bound over all $0 \leq t \leq C$ gives us that

$$\Pr[\neg A_d \wedge B_d \,|\, A_{d-1}] \leq (C+1) m^{-\gamma''} \leq m^{-\gamma'} \,.$$

We then get that $\Pr[A_k] \geq 1 - m^{-\gamma}$ as we wanted.

We just need to prove eq. (D.14). We note that $X_d^{(j)} \leq t$ if and only if $X_{d-1}^{(j)} - Y_d^{(j)} \leq t$. We thus get that,

$$\begin{aligned}\Pr\left[X_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i \in S}\right] = \sum_{s=0}^{t-1} \Pr\left[X_{d-1}^{(j)} \leq t - s \wedge Y_d^{(j)} = s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i \in S}\right] \\ + \Pr\left[Y_d^{(j)} \geq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i \in S}\right]\end{aligned}$$

If we fix the first $d-1$ levels then we get that

$$\Pr\left[X_{d-1}^{(j)} \leq t - s \,\middle|\, (X_d^{(i)})_{i \in S}\right] = \frac{\sum_{i \in [m] \setminus S} [X_{d-1}^{(j)} \leq t - s]}{m - |S|}$$

We condition on A_{d-1} so we know that,

$$\left|\frac{\sum_{i \in [m]} [X_{d-1}^{(i)} \leq t - s]}{m} - \Pr\left[\mathcal{X}_{d-1}^{(j)} \leq t - s\right]\right| \leq m^{-1/2+o(1)}$$

This implies that,

$$\begin{aligned}&\left|\frac{\sum_{i \in [m] \setminus S} [X_{d-1}^{(i)} \leq t - s]}{m - |S|} - \Pr\left[\mathcal{X}_{d-1}^{(j)} \leq t - s\right]\right| \\ &\qquad \leq m^{-1/2+o(1)} + \left|\frac{\sum_{i \in S} \left([X_{d-1}^{(i)} \leq t - s] - \Pr\left[\mathcal{X}_{d-1}^{(j)} \leq t - s\right]\right)}{m - |S|}\right| \\ &\qquad \leq m^{-1/2+o(1)}\end{aligned}$$

Here we have used that $|S| \leq O(\log(m))$. We thus get that

$$\begin{aligned}
&- \Pr\left[Y_d^{(j)} = s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] m^{-1/2+o(1)} \\
&\qquad \leq \Pr\left[X_{d-1}^{(j)} \leq t - s \wedge Y_d^{(j)} = s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] \\
&\qquad\qquad - \Pr\left[\mathcal{X}_{d-1}^{(j)} \leq t - s\right] \Pr\left[Y_d^{(j)} = s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] \\
&\qquad \leq \Pr\left[Y_d^{(j)} = s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] m^{-1/2+o(1)}
\end{aligned}$$

Using this we get that,

$$\left|\Pr\left[X_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\mathcal{X}_{d-1}^{(j)} - Y_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right]\right| \leq m^{-1/2+o(1)}$$

We now want to exchange $Y_d^{(j)}$ with $\mathcal{Y}_d^{(j)}$ and the approach is similar to what we just did. We note that,

$$\begin{aligned}
&\Pr\left[\mathcal{X}_{d-1}^{(j)} - Y_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] \\
&\qquad = \sum_{s=0}^{t-1} \Pr\left[\mathcal{X}_{d-1}^{(j)} = s\right] \Pr\left[Y_d^{(j)} \leq t - s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] \\
&\qquad + \Pr\left[\mathcal{X}_{d-1}^{(j)} \geq t\right]
\end{aligned}$$

We now use Lemma D.17 to get that,

$$\left|\Pr\left[Y_d^{(j)} \leq t - s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\mathcal{Y}_d^{(j)} \leq t - s\right]\right| \leq m^{-1/2+o(1)} + m^{-\gamma}$$

So

$$\begin{aligned}
&\left|\Pr\left[\mathcal{X}_{d-1}^{(j)} = s\right] \Pr\left[Y_d^{(j)} \leq t - s \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\mathcal{X}_{d-1}^{(j)} = s \wedge \mathcal{Y}_d^{(j)} \leq t - s\right]\right| \\
&\qquad \leq \Pr\left[\mathcal{X}_{d-1}^{(j)} = s\right] m^{-1/2+o(1)}
\end{aligned}$$

This implies that,

$$\left|\Pr\left[X_d^{(j)} \leq t \,\middle|\, A_{d-1} \wedge (X_d^{(i)})_{i\in S}\right] - \Pr\left[\mathcal{X}_d^{(j)} \leq t\right]\right| \leq m^{-1/2+o(1)}$$

Where we have use that $t \leq C \leq m^{o(1)}$. This proves eq. (D.14) and thus finishes the proof. □

Later in the paper we will need to bound the contribution to a bin while fixing the previous levels. The proof structure is very similar to the proof of We define $\mathcal{L}_{d-1}$ to be the sigma-algebra generated by the first $d-1$ levels.

Lemma D.21. *Let $0 \le t \le C$, $j \in [m]$, and $1 \le d \le k$. Then*

$$\left|\Pr\left[\sum_{i=d}^{k} Y_i^{(j)} \ge t \,\middle|\, \mathcal{L}_{d-1}\right] - \Pr\left[\sum_{i=d}^{k} \mathcal{Y}_i^{(j)} \ge t\right]\right| \le k\left[A_{d-1}^c\right] + 2km^{-1/2+o(1)} \tag{D.15}$$

Proof. We will prove that,

$$\begin{aligned}\left|\Pr\left[\sum_{i=k-r}^{k} Y_i^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-r-1}\right] - \Pr\left[\sum_{i=k-r}^{k} \mathcal{Y}_i^{(j)} \ge t\right]\right| \\ \le (1+r)\left[A_{k-r-1}^c\right] + (1+2r)m^{-1/2+o(1)}\end{aligned} \tag{D.16}$$

for $0 \le r \le k-d$ and all $0 \le t \le C$. We will prove the result by induction on r.

We first consider $r = 0$. We then have that,

$$\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-1}\right] = [A_{k-1}]\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{r-1}\right] + \left[A_{k-1}^c\right]\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{r-1}\right]$$

We now use Lemma D.17 to get that, $[A_{k-1}]\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-1}\right] = [A_{k-1}]\left(\Pr\left[\mathcal{Y}_k^{(j)} \ge t\right] + \delta\right)$ where $|\delta| \le m^{-1/2+o(1)}$. We then get that,

$$\begin{aligned}\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-1}\right] &= [A_{k-1}]\left(\Pr\left[\mathcal{Y}_k^{(j)} \ge t\right] + \delta\right) + \left[A_{k-1}^c\right]\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{r-1}\right] \\ &= \Pr\left[\mathcal{Y}_k^{(j)} \ge t\right] + \left[A_{k-1}^c\right]\left(\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{r-1}\right] - \Pr\left[\mathcal{Y}_k^{(j)} \ge t\right]\right) + [A_{k-1}]\,\delta\end{aligned}$$

We have that

$$\begin{aligned}\left|\left[A_{k-1}^c\right]\left(\Pr\left[Y_k^{(j)} \ge t \,\middle|\, \mathcal{L}_{r-1}\right] - \Pr\left[\mathcal{Y}_k^{(j)} \ge t\right]\right) + [A_{k-1}]\,\delta\right| &\le \left[A_{k-1}^c\right] + |\delta| \\ &\le \left[A_{k-1}^c\right] + m^{-1/2+o(1)}\end{aligned}$$

This proves eq. (D.16) for $r = 0$ which will be our induction start.

Now we consider $r \ge 1$ assume that eq. (D.16) is true for values less than r. We note that,

$$\begin{aligned}\Pr\left[\sum_{i=k-r}^{k} Y_i^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-r-1}\right] &= \sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \ge t-s \wedge Y_{k-r}^{(j)} = s \,\middle|\, \mathcal{L}_{k-r-1}\right] \\ &\quad + \Pr\left[Y_{k-r}^{(j)} \ge t \,\middle|\, \mathcal{L}_{k-r-1}\right]\end{aligned}$$

We now fix $0 \le s \le t-1$ and use the tower property of conditional expectation to get that,

$$\begin{aligned}&\Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \ge t-s \wedge Y_{k-r}^{(j)} = s \,\middle|\, \mathcal{L}_{k-r-1}\right] \\ &\qquad = \mathrm{E}\left[\left[Y_{k-r}^{(j)} = s\right]\Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \ge t-s \,\middle|\, \mathcal{L}_{k-r}\right] \,\middle|\, \mathcal{L}_{k-r-1}\right]\end{aligned}$$

By then induction hypothesis we have that $\Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \geq t-s \,\middle|\, \mathcal{L}_{k-r}\right] = \Pr\left[\sum_{i=k-r}^{k} \mathcal{Y}_i^{(j)} \geq t-s\right] + \delta_s$ where $|\delta_s| \leq r\left[A_{k-r}^c\right] + (2r-1)m^{-1/2+o(1)}$. This implies that,

$$\begin{aligned}
&\mathrm{E}\left[\left[Y_{k-r}^{(j)}=s\right]\Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \geq t-s \,\middle|\, \mathcal{L}_{k-r}\right] \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\qquad = \mathrm{E}\left[\left[Y_{k-r}^{(j)}=s\right]\left(\Pr\left[\sum_{i=k-r}^{k} \mathcal{Y}_i^{(j)} \geq t-s\right]+\delta_s\right) \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\qquad = \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} \geq t-s \wedge Y_{k-r}^{(j)}=s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \mathrm{E}\left[\left[Y_{k-r}^{(j)}=s\right]\delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right]
\end{aligned}$$

Hence, we get that,

$$\begin{aligned}
&\Pr\left[\sum_{i=k-r}^{k} Y_i^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\quad = \sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} Y_i^{(j)} \geq t-s \wedge Y_{k-r}^{(j)}=s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[Y_{k-r}^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\quad = \sum_{s=0}^{t-1} \left(\Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} \geq t-s \wedge Y_{k-r}^{(j)}=s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \mathrm{E}\left[\left[Y_{k-r}^{(j)}=s\right]\delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right]\right) \\
&\qquad\qquad + \Pr\left[Y_{k-r}^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\quad = \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} + Y_{k-r}^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] + \sum_{s=0}^{t-1} \mathrm{E}\left[\left[Y_{k-r}^{(j)}=s\right]\delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right]
\end{aligned}$$

Now we want to exchange $Y_{k-r}^{(j)}$ with $\mathcal{Y}_{k-r}^{(j)}$ and the method is similar to before. We write,

$$\begin{aligned}
&\Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} + Y_{k-r}^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] \\
&\quad = \sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \Pr\left[Y_{k-r}^{(j)} \geq t-s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} \geq t\right]
\end{aligned}$$

Now by analogous arguments as in the induction start we get that $\Pr\left[Y_{k-r}^{(j)} \geq t-s \,\middle|\, \mathcal{L}_{k-r-1}\right] = \Pr\left[\mathcal{Y}_{k-r}^{(j)} \geq t-s\right] + \delta'_s$ where $|\delta'_s| \leq \left[A_{k-r-1}^c\right] + m^{-1/2+o(1)}$.

We thus get that,

$$\begin{aligned}
&\sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \Pr\left[Y_{k-r}^{(j)} \geq t-s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} \geq t\right] \\
&\qquad = \sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \left(\Pr\left[\mathcal{Y}_{k-r}^{(j)} \geq t-s\right] + \delta'_s\right) + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} \geq t\right] \\
&\qquad = \Pr\left[\sum_{i=k-r}^{k} \mathcal{Y}_i^{(j)} \geq t\right] + \sum_{s=0}^{t-1} \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \delta'_s
\end{aligned}$$

Now combining it all we get that,

$$\begin{aligned}
\Pr\left[\sum_{i=k-r}^{k} Y_i^{(j)} \geq t \,\middle|\, \mathcal{L}_{k-r-1}\right] &= \Pr\left[\sum_{i=k-r}^{k} \mathcal{Y}_i^{(j)} \geq t\right] \\
&\quad + \sum_{s=0}^{t-1} \left(\mathrm{E}\left[\left[Y_{k-r}^{(j)} = s\right] \delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \delta'_s\right).
\end{aligned}$$

Now to finish the proof we just need to bound $\left|\sum_{s=0}^{t-1} \left(\mathrm{E}\left[\left[Y_{k-r}^{(j)} = s\right] \delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \delta'_s\right)\right|$.

$$\begin{aligned}
&\left|\sum_{s=0}^{t-1} \left(\mathrm{E}\left[\left[Y_{k-r}^{(j)} = s\right] \delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \delta'_s\right)\right| \\
&\quad \leq \max_{s=0}^{t-1} \mathrm{E}[|\delta_s| \,|\, \mathcal{L}_{k-r-1}] + \max_{s=0}^{t-1} |\delta'_s| \leq \left[A_{k-r-1}^c\right] + 2rm^{-1/2+o(1)} + r \Pr\left[A_{k-r}^c \,\middle|\, \mathcal{L}_{k-r-1}\right]
\end{aligned}$$

We now just need to bound $\Pr\left[A_{k-r}^c \,\middle|\, \mathcal{L}_{k-r-1}\right]$. From the proof of Theorem D.16 we have that $[A_{k-r-1}] \Pr\left[A_{k-r}^c \,\middle|\, \mathcal{L}_{k-r-1}\right] \leq m^{-\gamma}$. So we get that

$$r \Pr\left[A_{k-r}^c \,\middle|\, \mathcal{L}_{k-r-1}\right] \leq r\left[A_{k-r-1}^c\right] + rm^{-\gamma} \leq r\left[A_{k-r-1}^c\right] + m^{-1/2+o(1)}$$

This implies that,

$$\begin{aligned}
&\left|\sum_{s=0}^{t-1} \left(\mathrm{E}\left[\left[Y_{k-r}^{(j)} = s\right] \delta_s \,\middle|\, \mathcal{L}_{k-r-1}\right] + \Pr\left[\sum_{i=k-r+1}^{k} \mathcal{Y}_i^{(j)} = s\right] \delta'_s\right)\right| \\
&\qquad\qquad \leq (r+1)\left[A_{k-r-1}^c\right] + (1+2r)m^{-1/2+o(1)}
\end{aligned}$$

This finishes the induction step and thus the proof. □

We now turn to the proof Lemma D.20.

Proof of Lemma D.20. Let $j_1, \ldots, j_r$ be different indices and $a_1, \ldots, a_r$ be non-negative integers such that $\sum_{i=1}^{r} a_i = r$. We then want to estimate $\mathrm{E}[(X_{j_r} - p)^{a_r} \mid (X_{j_i})_{i<r}]$

$$\begin{aligned}\left|\mathrm{E}[(X_{j_r} - p)^{a_r} \mid (X_{j_i})_{i<r}]\right| &= \left|\mathrm{E}[X_{j_r} \mid (X_{j_i})_{i<r}]\,(1-p)^{a_r} + (1 - \mathrm{E}[X_{j_r} \mid (X_{j_i})_{i<r}])(-p)^{a_r}\right| \\ &\leq \delta\,|(1-p)^{a_r} - (-p)^{a_r}| + |p(1-p)^{a_r} + (1-p)(-p)^{a_r}|\end{aligned}$$

Now let X_i' be independent Bernoulli variables with parameter p' where $p' = p + \delta$ if $p < \frac{1}{2}$ and $p' = p - \delta$ when $p \geq \frac{1}{2}$. It is now easy to check that

$$\left|\mathrm{E}\left[(X_i' - p)^{a_r}\right]\right| = \delta\,|(1-p)^{a_r} - (-p)^{a_r}| + |p(1-p)^{a_r} + (1-p)(-p)^{a_r}|$$

Using this we see that

$$\left|\mathrm{E}\left[\prod_{i=1}^{r}(X_i - p)^{a_r}\right]\right| \leq \left|\mathrm{E}\left[\prod_{i=1}^{r}(X_i' - p)^{a_r}\right]\right|$$

From this we conclude that $\mathrm{E}[(\sum_i (X_i - p))^r] \leq \mathrm{E}[(\sum_i (X_i' - p))^r]$. Now by the triangle inequality and Hoeffding's inequality we get that

$$\begin{aligned}\mathrm{E}\left[\left(\sum_i (X_i - p)\right)^r\right]^{1/r} &\leq \mathrm{E}\left[\left(\sum_i (X_i' - p)\right)^r\right]^{1/r} \\ &\leq \delta n + \mathrm{E}\left[\left(\sum_i (X_i' - p')\right)^r\right]^{1/r} \\ &\leq \delta n + O(\sqrt{rn})\end{aligned}$$

Now using Markov's inequality give us the tail bound. □

D.5.2 The probability that a bin is not full

In this section we will bound the probability $\Pr\left[\mathcal{X}_k^{(j)} = 0\right]$ for any bin j. Since the bound is the same for all bins we will suppress j from the notation. We note that $\Pr[\mathcal{X}_k = 0] = \Pr\left[\sum_{i=1}^{k} \mathcal{Y}_i \geq C\right]$. Now an important observation is that if we define $1 - f_d = \Pr\left[\sum_{i=1}^{d} \mathcal{Y}_i \geq C\right]$, then $\mathcal{Y}_d$ is geometrically distributed with parameter $\alpha_d = \frac{n/k}{n/k + f_d m} = \frac{1}{1 + \frac{f_d k}{\mu}}$. The reason is that when generating $\mathcal{Y}_d$, we sample with replacement so when sampling a bin, the probability that it will be filled is $1 - f_d$ independently of the history. Thus, at any point, the probability of getting another ball is

$$\frac{n/k}{n/k+m}\sum_{i=0}^{\infty}\left(\frac{m}{n/k+m}(1-f_d)\right)^i = \frac{\frac{n/k}{n/k+m}}{1 - \frac{m}{n/k+m}(1-f_d)} = \frac{n/k}{n/k + f_d m} = \frac{1}{1 + \frac{f_d k}{\mu}}\,.$$

Which is exactly what we get from a geometrically distributed variable.

From simple facts about geometrically distributed variables we get that $\mu_d = \mathrm{E}[\mathcal{Y}_d] = \frac{\mu}{kf_d}$, and $\sigma_d^2 = \mathrm{Var}[\mathcal{Y}_d] = \frac{\mu}{kf_d}\left(1 + \frac{\mu}{kf_d}\right) \geq \mu_d$. We note that

$$\sum_{i=1}^{k} \sigma_i^2 = \sum_{i=1}^{k} \frac{\mu}{kf_i}\left(1 + \frac{\mu}{kf_i}\right) \geq \sum_{i=1}^{k} \frac{\mu}{kf_i} \tag{D.17}$$

We define $S_d = \sum_{i=1}^{d} \mathcal{Y}_i$ for $1 \leq d \leq k$. Our goal is to prove that there exists a constant L such that,

$$f_k \geq L \begin{cases} \varepsilon C & \text{if } C \leq \left(1/(\varepsilon\sqrt{C})\right) \\ \varepsilon\sqrt{C \log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} & \text{if } \log\left(1/(\varepsilon\sqrt{C})\right) \leq C \leq \frac{1}{2\varepsilon^2} \\ 1 & \text{if } \frac{1}{2\varepsilon^2} \leq C \end{cases}. \tag{D.18}$$

Combining this with Theorem D.16, the second part of Theorem D.4 will follow. Now to prove Equation (D.18), it suffices to consider the case $C \leq \gamma/\varepsilon^2$ for a sufficiently small constant γ. Indeed, otherwise, we apply a reduction similar to the one in Case 3 in the proof of Theorem D.1. We will make this assumption in what follows. We also note that we can assume that C is larger than $\frac{1}{4L}$ because if $C \leq \frac{1}{4L}$ then we get that,

$$\varepsilon/4 \geq L \begin{cases} \varepsilon C & \text{if } C \leq \log\left(1/(\varepsilon\sqrt{C})\right) \\ \varepsilon\sqrt{C \log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} & \text{if } \log\left(1/(\varepsilon\sqrt{C})\right) \leq C \leq \frac{1}{2\varepsilon^2} \\ 1 & \text{if } \frac{1}{2\varepsilon^2} \leq C \end{cases}.$$

We will argue that f_k is always larger than $\varepsilon/4$. We know that $\sum_{j\in[m]}\left[X_k^{(j)} = 0\right] \leq \frac{n}{C} = \frac{m}{1+\varepsilon}$ and $1 - f_k \leq \frac{\sum_{j\in[m]}\left[X_k^{(j)}=0\right]}{m} + m^{-1/2+o(1)}$ with probability at least $1 - m^{-\gamma}$ by Theorem D.4. Fixing such event give us that,

$$f_k \geq 1 - \frac{1}{1+\varepsilon} - m^{-1/2+o(1)} \geq \varepsilon/2 - \varepsilon/4 = \varepsilon/4\,.$$

Here we have used that $\varepsilon \leq 1$ and that $1/\varepsilon = m^{o(1)}$. So if $C \leq \frac{1}{4L}$ then eq. (D.18) holds and we in from now assume that $C > \frac{1}{4L}$.

We will prove the result by showing the stronger result that for all $1 \leq d \leq k$,

$$f_d \geq L \begin{cases} \varepsilon C & \text{if } C \leq \log\left(1/(\varepsilon\sqrt{C})\right) \\ \varepsilon\sqrt{C \log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} & \text{if } \log\left(1/(\varepsilon\sqrt{C})\right) \leq C \leq \frac{1}{2\varepsilon^2} \\ 1 & \text{if } \frac{1}{2\varepsilon^2} \leq C \end{cases}. \tag{D.19}$$

We will prove eq. (D.19) by induction on d.

First we note that $f_1 \geq f_2 \geq \ldots \geq f_k\varepsilon/4$. We then get that $\mathrm{E}[S_d] = \sum_{i=1}^{d} \mu_i \leq \frac{4d\mu}{\varepsilon k} \leq \frac{d\varepsilon C}{2}$, where we have used that $k \geq 8/\varepsilon^2$. So for $d \leq 1/\varepsilon$ we get that $\sum_{i=1}^{d} \mu_i \leq C/2$ and Markov's inequality give us that,

$$1 - f_d = \Pr[S_d \geq C] \leq \frac{1}{2} .$$

This shows that eq. (D.19) holds for $d \leq 1/\varepsilon$ and since $\varepsilon \leq 1$ then it holds for $d = 1$ which will be our induction start.

Now the previous argument shows that when $\mathrm{E}[S_d] \leq C/2$ then eq. (D.19) holds, so we can assume that $\sum_{i=1}^{d} \mu_i > C/2$. We note that $f_d \geq f_{d-1}/2$ since,

$$f_d = \Pr[S_d < C] \geq \Pr[S_{d-1} < C \wedge \mathcal{Y}_d = 0] = f_{d-1}(1 - \alpha_d) = f_{d-1}\frac{1}{1 + \frac{\mu}{f_d k}} .$$

This implies that $f_d \geq f_{d-1} - \frac{\mu}{k} \geq f_{d-1}/2$ where we use that $k \geq c/\varepsilon^2$ for some sufficiently large constant c and that $f_{d-1} \geq L\varepsilon^2 C$. We now note that if $f_i \geq \frac{L}{2}\min\left\{\varepsilon\sqrt{C}, 1\right\}$ then $\mu_i \leq 1$, since,

$$\mu_d = \frac{\mu}{k f_i} \leq \frac{2\varepsilon^2 C}{cL\min\left\{\varepsilon\sqrt{C}, 1\right\}} \leq 1 .$$

Here we have used that $k \geq c/\varepsilon^2$, that c is sufficiently large, and that $1/\varepsilon^2 \leq C$. We also note that

$$\begin{aligned}\mathrm{E}\left[|\mathcal{Y}_i - \mu_i|^3\right] &= \mathrm{E}\left[[\mathcal{Y}_i \leq \mu_i]\,(\mu_i - \mathcal{Y}_i)^3\right] + \mathrm{E}\left[[\mathcal{Y}_i > \mu_i]\,(\mathcal{Y}_i - \mu_i)^3\right] \\ &\leq \mu_i^3 + \alpha_i^{\lfloor \mu_i \rfloor}\,\mathrm{E}\left[\mathcal{Y}_i^3\right] \leq \mu_i^3 + \mathrm{E}\left[\mathcal{Y}_i^3\right] .\end{aligned}$$

In the first inequality we have used that the geometric distribution is memoryless. Now simple calculations give that $\mathrm{E}\left[\mathcal{Y}_i^3\right] \leq 6\mu_i(1+\mu_i)^2 \leq 24\mu_i$, so we get that $\mathrm{E}\left[|\mathcal{Y}_i - \mu_i|^3\right] \leq \mu_i^3 + 24\mu_i \leq 25\mu_i$.

Depending on $\mathrm{E}[S_d]$ we will prove different bounds on f_d. Let $M > 0$ be a large constant. We will prove that if $\mathrm{E}[S_d] < C + M\sqrt{C}$ then $f_d \geq L$, if $C + M\sqrt{C} \leq \mathrm{E}[S_d] < C + \frac{1}{4}C$ then $f_d \geq L\varepsilon\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)}$, and if $C + \frac{1}{4}C \leq \mathrm{E}[S_d]$ then $f_d \geq L\varepsilon C$. This will prove the result since $\varepsilon\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} \geq \varepsilon C$ if and only if $C \geq \left(1/(\varepsilon\sqrt{C})\right)$, and since $C \leq \gamma/\varepsilon^2$ for a small constant γ then $1 \geq \min\left\{\varepsilon\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)}, \varepsilon C\right\}$.

If $\mathrm{E}[S_d] < C + M\sqrt{C}$ then we will show that $f_d \geq L$. This will follow by a usage of the Berry-Esseen theorem.

Theorem D.22 (Berry Esseen theorem)**.** *Let $X_1, \ldots, X_d$ be independent random variables with $\mathrm{E}[X_i] = 0$, $\mathrm{E}\left[X_i^2\right] = \sigma_i^2 > 0$, and $\mathrm{E}\left[|X_i|^3\right] = \rho_i < \infty$. Let F_d be the cumulative distribution function of $\sum_{i=1}^{d} X_i$, let Φ be the cumulative distribution function of the standard*

normal distribution, and let $\sigma^2 = \sum_{i=1}^{d} \sigma^2$. *Then,*

$$\sup_{x\in\mathbb{R}} |F_d(x) - \Phi(x/\sigma)| \leq K_1 \frac{\sum_{i=1}^{d} \rho_i}{\sigma^3} .$$

where K_1 *is a universal constant.*

Since $\mathrm{E}[S_d] < C + M\sqrt{C}$ then $C \geq \mathrm{E}[S_d] - M\sqrt{\mathrm{E}[S_d]} \leq \mathrm{E}[S_d] - M\sqrt{\sum_{i=1}^{d} \sigma_i^2}$ and we get that $f_d = \Pr[S_d < C] \geq \Pr\left[S_d < \mathrm{E}[S_d] - M\sqrt{\sum_{i=1}^{d} \sigma_i^2}\right]$. Now the Berry-Esseen theorem give us that,

$$f_d \geq \Pr\left[S_d < \mathrm{E}[S_d] - \sqrt{\sum_{i=1}^{d} \sigma_i^2}\right] \geq \Phi(-M) - K_1 \frac{\sum_{i=1}^{d} \mathrm{E}\left[|\mathcal{Y}_i - \mu_i|^3\right]}{\left(\sum_{i=1}^{d} \sigma_i^2\right)^{3/2}}$$

We know that $\mathrm{E}\left[|\mathcal{Y}_i - \mu_i|^3\right] \leq 25\mu_i$ and that $\sigma_i^2 \geq \mu_i$ for all $1 \leq i \leq d$, so we get that,

$$f_d \geq \Phi(-M) - 25K_1 \frac{\mathrm{E}[S_d]}{\mathrm{E}[S_d]^{3/2}} \geq \Phi(-M) - 25K_1\sqrt{8L} \geq L .$$

Here we have used that $\mathrm{E}[S_d] \geq C/2 \geq \frac{1}{8L}$ and that L is sufficiently small.

Now we consider the case where $\mathrm{E}[S_d] \geq C + M\sqrt{C}$. We define $\beta_d = \mathrm{E}[S_d] - C$ and note that $\beta_d \geq M\sqrt{C}$. We will need the following claim.

Claim D.23. *For all* $1 \leq d \leq k$ *and all integers* $t \geq 1$ *we have that,*

$$\frac{\Pr[S_d = t+1]}{\Pr[S_d = t]} \leq \frac{\Pr[S_d = t]}{\Pr[S_d = t-1]} .$$

Proof. We define the sets $A_t = \left\{(a_1, \ldots, a_d) \in \mathbb{N}_0^d \,\middle|\, \sum_{i=1}^{d} a_i = t\right\}$ and get that

$$\Pr\left[\sum_{i=1}^{d} \mathcal{Y}_i = t\right] = \sum_{(a_1,\ldots,a_d)\in A_t} \prod_{i=1}^{d} \alpha_i^{a_i}(1-\alpha_i) .$$

We note that the result it is equivalent to showing that $\Pr[X = t+1]\Pr[X = t-1] \leq \Pr[X = t]^2$ which in turn is equivalent to

$$\sum_{(a,b)\in A_{t+1}\times A_{t-1}} \prod_{i=1}^{d} \alpha_i^{a_i+b_i}(1-\alpha_i)^2 \leq \sum_{(a,b)\in A_t\times A_t} \prod_{i=1}^{d} \alpha_i^{a_i+b_i}(1-\alpha_i)^2.$$

To see that this latter inequality holds, let $s \in A_{2t}$ and define the map $g_s : \mathbb{N}_0^d \to \mathbb{N}_0$ by $g_s(i) = |\{(a,b) \in A_i \times A_{2t-i} \mid a + b = s\}|$. We note that $g_s(i) > 0$ exactly when $i \in \{0, 1, \ldots, 2t\}$. The desired inequality is then equivalent to

$$\sum_{s\in A_{2t}} g_s(t+1) \prod_{i=1}^{d} \alpha_i^{s_i} \leq \sum_{s\in A_{2t}} g_s(t) \prod_{i=1}^{d} \alpha_i^{s_i}.$$

We will show that g_s is log-concave for each $s \in A_{2t}$. As g_s is clearly symmetric around $i = t$, it will in particular follow that $g_s(t+1) \leq g_s(t)$ which then leads to the desired inequality. To show that g_s is log-concave, we note that it is a convolution of log-concave functions. Indeed, fix s and define for $1 \leq j \leq d$, the map $h_j : \mathbb{N}_0 \to \mathbb{N}_0$ by $h_j(i) = 1$ if $0 \leq i \leq s_j$ and $h_j(i) = 0$ otherwise. Then each h_j is log-concave, and moreover, g_s is the convolution $g_s = h_1 * \cdots * h_k$, i.e.,

$$g_s(i) = \sum_{\substack{a \in \mathbb{Z}^k \\ a_1 + \cdots + a_d = i}} \prod_{j=1}^{d} h_j(a_j).$$

It is a standard fact that the convolution of log-concave functions is again log-concave, and the desired inequality follows. □

Now let $\ell_d \in \mathbb{N}$ be the minimal integer satisfying that $\Pr[S_d = C-1] / \Pr[S_d = C-1-\ell_d] \geq 2$. Now combining Claim D.23 with the definition of ℓ_d we get that,

$$\begin{aligned}
\Pr[S_d < C] &= \sum_{t=1}^{C} \Pr[S_d = C-t] \geq \sum_{t=1}^{\ell_d} \Pr[S_d = C-t] \geq \frac{\ell_d}{2} \Pr[S_d = C-1]\,, \\
\Pr[S_d < C] &= \sum_{t=1}^{C} \Pr[S_d = C-t] \leq \sum_{r=0}^{\lceil C/\ell_d \rceil} \ell_d \Pr[S_d = C-1-r\ell_d] \\
&\leq \ell_d \Pr[S_d = C-1] \sum_{r=0}^{\infty} 2^{-r} = 2\ell_d \Pr[S_d = C-1]\,, \\
\mathrm{E}[(C-S_d)[S_d < C]] &= \sum_{t=1}^{C} t \Pr[S_d = C-t] \\
&\leq \sum_{r=0}^{\lceil C/\ell_d \rceil} \left(r\ell_d^2 + \frac{\ell_d(\ell_d+1)}{2} \right) \Pr[S_d = C-1-r\ell_d] \\
&\leq \sum_{r=0}^{\lceil C/\ell_d \rceil} \left(r\ell_d^2 + \frac{\ell_d(\ell_d+1)}{2} \right) 2^{-r} \Pr[S_d = C-1] \\
&\leq 4\ell_d^2 \Pr[S_d = C-1]\,, \\
\mathrm{E}[(C-S_d)[S_d < C]] &= \sum_{t=1}^{C} t \Pr[S_d = C-t] \geq \sum_{t=1}^{\ell_d} t \Pr[S_d = C-1-t] \\
&\geq \frac{\ell_d^2}{4} \Pr[S_d = C-1]\,.
\end{aligned}$$

From this we get that $\frac{\mathrm{E}[(C-S_d)[S_d<C]]}{8\ell_d} \leq f_d \leq \frac{8\,\mathrm{E}[(C-S_d)[S_d<C]]}{\ell_d}$. Now it is clear that $\mathrm{E}[(C-S_d)[S_d<C]] \geq \mathrm{E}[(C-S_k)[S_k<C]]$ and we will argue that $\mathrm{E}[(C-S_k)[S_k<C]] \geq \frac{\varepsilon\mu}{2} \geq \frac{\varepsilon C}{4}$. This will imply that $f_d \geq \frac{\varepsilon C}{32\ell_d}$. Using Theorem D.16 we

get that $\Pr[S_d \geq C - t] \geq \frac{\sum_{j\in[m]}\left[X_k^{(j)} \leq t\right]}{m} - m^{-1/2+o(1)}$ for all $1 \leq t \leq C$ with probability $1 - m^{-\gamma}$, and we know that $\sum_{t=1}^{C} \frac{\sum_{j\in[m]}\left[X_k^{(j)} \leq t\right]}{m} = \varepsilon C$, so fixing such event give us that,

$$\begin{aligned}
\mathrm{E}[(C - S_k)[S_k < C]] &= \sum_{t=1}^{C} \Pr[S_d \geq C - t] \\
&\geq \sum_{t=1}^{C} \left(\frac{\sum_{j\in[m]} \left[X_k^{(j)} \leq t\right]}{m} - m^{-1/2+o(1)} \right) \\
&= \varepsilon\mu - Cm^{-1/2+o(1)} \\
&\geq \tfrac{\varepsilon C}{2} - Cm^{-1/2+o(1)} \\
&\geq \tfrac{\varepsilon C}{4} \,.
\end{aligned}$$

Here we have used that $\varepsilon \leq 1$ and that $1/\varepsilon = m^{o(1)}$.

Now we just need to upper bound ℓ_d. By Claim D.23 we get that $\ell_d \leq \left\lceil \frac{\log(2)}{\log\left(\frac{\Pr[S_d=C]}{\Pr[S_d=C-1]}\right)} \right\rceil$ so we want to lower bound $\frac{\Pr[S_d=C]}{\Pr[S_d=C-1]}$. To do this we will define exponentially tilted variables $(V_i)_{1\leq i\leq d}$. Let $\lambda \in \mathbb{R}$ satisfying $\mathrm{E}[e^{\lambda S_d}] < \infty$ be a parameter which will be determined later. We define V_i by $\Pr[V_i = t] = \frac{\Pr[\mathcal{Y}_i=t]e^{\lambda t}}{\mathrm{E}\left[e^{\lambda\mathcal{Y}_i}\right]}$ for $1 \leq i \leq d$. Clearly, this is well-defined since $\sum_{t=0}^{\infty} \Pr[\mathcal{Y}_i = t]\, e^{\lambda t} = \mathrm{E}\left[e^{\lambda\mathcal{Y}_i}\right]$. As pointed out in [AAKT21], each V_i is also geometric random variables (with parameter $\alpha_i e^{\lambda}$) and,

$$\Pr[S_d = C - t] = \frac{\mathrm{E}\left[e^{\lambda\sum_{i=1}^{d}\mathcal{Y}_i}\right]}{e^{\lambda C}} e^{\lambda t} \Pr\left[\sum_{i=1}^{d} V_i = C - t\right]. \tag{D.20}$$

for all integers t. Moreover, there is a unique λ maximizing $\lambda C - \log \mathrm{E}\left[e^{\lambda\sum_{i=1}^{d}\mathcal{Y}_i}\right]$, and with this choice of λ, it holds that $\sum_{i=1}^{d} \mathrm{E}[V_i] = C$. It is easy to see that $\lambda < 0$ since $\beta > 0$. We start by noticing that $\sum_{i=1}^{d} \mathrm{Var}[V_i] \geq \sum_{i=1}^{d} \mathrm{E}[V_i] = C$, and that $\mathrm{E}\left[|V_i - \mathrm{E}[V_i]|^3\right] \leq 25\,\mathrm{E}[V_i]$ by the same reasoning that gave us that $\mathrm{E}\left[|\mathcal{Y}_i - \mu_i|^3\right] \leq 25\,\mathrm{E}[\mathcal{Y}_i]$ since V_i is geometrically distributed with parameter $\alpha_i e^{\lambda} < \alpha_i$.

We will also need the following lemma by Aamand et al. [AAKT21]. We state a simplified version of their lemma which covers our use case.

Lemma D.24. *Let $X_1, \ldots, X_d$ be independent geometric distributed random variables with $\mathrm{Var}[X_i] = \sigma_i^2 > 0$ and $\mathrm{E}\left[|X_i - \mathrm{E}[X_i]|^3\right] = \rho_i < \infty$, and let $\sigma^2 = \sum_{i=1}^{d} \sigma_i^2$. Then for every t where $\mu + t\sigma$ is an integer,*

$$\left| \Pr[X = \mu + t\sigma] - \frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2} \right| \leq K_2 \left(\frac{\sum_{i=1}^{d} \rho_i}{\sigma^3} \right)^2 .$$

where K_2 is a universal constant.

We will also need the following claim. The proof is bit technical so we defer the proof till the end of the section.

Claim D.25. *If $\beta \geq C$ then,*

$$\frac{\Pr[S_d = C]}{\Pr[S_d = C-1]} \geq e^{1/8} ,$$

and if $\beta < C$ then,

$$\frac{\Pr[S_d = C]}{\Pr[S_d = C-1]} \geq e^{\frac{1}{8}\beta/C} ,$$

and

$$\frac{\Pr[S_d = C-1]}{\Pr\left[S_d = C - 1 - \frac{1}{2}\left\lceil \frac{C}{\beta} \right\rceil\right]} < 2 .$$

If $\beta \geq \frac{1}{4}C$ then using Claim D.25 we get that $\ell_d \leq \lceil 32\log(2) \rceil$ which implies that $f_d \geq \frac{\varepsilon C}{32\lceil 32\log(2)\rceil} \geq L\varepsilon C$. So now we just need to focus on the case where $\beta < \frac{1}{4}C$. We use Claim D.25 to get that $\ell_d \leq \left\lceil 8\log(2)\frac{C}{\beta} \right\rceil \leq 7\frac{C}{\beta}$ which implies that $f_d \geq \frac{\varepsilon\beta}{224}$.

We now just need to lower bound β. From Claim D.25 we know that $\ell_d > \frac{1}{4}\left\lceil \frac{C}{\beta} \right\rceil \geq \frac{C}{4\beta} > 1$, so $\Pr[S_d = C] \leq \Pr[S_d = C-1]/2$ and we get that $\mathrm{E}[(C - S_d)[S_d < C]] \geq \frac{\ell_d^2}{8}\Pr[S_d = C]$. We will argue that $\mathrm{E}[(C - S_d)[S_d < C]] \leq 2\varepsilon C$. Using Theorem D.16 we get that $\Pr[S_d \geq C - t] \leq \frac{\sum_{j\in[m]}\left[X_k^{(j)} \leq t\right]}{m} + m^{-1/2+o(1)}$ for all $1 \leq t \leq C$ with probability $1 - m^{-\gamma}$, and we know that $\sum_{t=1}^{C} \frac{\sum_{j\in[m]}\left[X_k^{(j)} \leq t\right]}{m} = \varepsilon C$, so fixing such event give us that,

$$\begin{aligned}
\mathrm{E}[(C - S_k)[S_k < C]] &= \sum_{t=1}^{C} \Pr[S_d \geq C - t] \\
&\leq \sum_{t=1}^{C} \left(\frac{\sum_{j\in[m]}\left[X_k^{(j)} \leq t\right]}{m} + m^{-1/2+o(1)} \right) \\
&= \varepsilon\mu + Cm^{-1/2+o(1)} \\
&\leq \varepsilon C - Cm^{-1/2+o(1)} \\
&\leq 2\varepsilon C .
\end{aligned}$$

Here we have used that $1/\varepsilon = m^{o(1)}$. Combing it all we have that,

$$\Pr[S_d = C] \leq 16\frac{\varepsilon C}{\ell_d^2} \leq 256\frac{\varepsilon\beta^2}{C} .$$

We will now prove that

$$\Pr[X = C] \geq \frac{\exp\left(-\frac{\beta^2}{C}\right)}{4\sqrt{C}} \,. \tag{D.21}$$

This will lead to the desired result. Indeed, combining with the bounds above, we then obtain that

$$\frac{\exp\left(-\frac{\beta^2}{C}\right)}{\sqrt{C}} \leq \frac{\varepsilon\beta^2}{1024C} \,,$$

or $\Delta e^{\Delta} \geq \frac{1}{1024\varepsilon\sqrt{C}}$, where we have put $\Delta = \beta^2/C$. Then $\Delta \geq \frac{1}{1024}\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)$, so that $\beta \geq \frac{1}{32}\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)}$, and finally

$$f_d = \Pr[S_d < C] \geq \tfrac{1}{32}\varepsilon\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} \geq L\varepsilon\sqrt{C\log\left(\frac{1}{\varepsilon\sqrt{C}}\right)} \,,$$

as desired.

We thus turn to prove eq. (D.21). By eq. (D.20) we have that,

$$\Pr[S_d = C] = \frac{\mathrm{E}\left[e^{\lambda\sum_{i=1}^{d}\mathcal{Y}_i}\right]}{e^{\lambda C}}\Pr\left[\sum_{i\in[k]} V_i = C\right] \,. \tag{D.22}$$

We start by focusing on bounding $\lambda C - \log\mathrm{E}\left[e^{\lambda\sum_{i=1}^{d}\mathcal{Y}_i}\right]$. First write $\psi_d(p) = \log\mathrm{E}\left[e^{p\sum_{i=1}^{d}\mathcal{Y}_i}\right]$ and define the function $g_d(t) = \sup_p(pt - \psi_d(p))$ which is the Fenchel-Legendre transform of $\psi_d(p)$. By our choice of λ, $g_d(C) = \lambda C - \log\mathrm{E}\left[e^{\lambda\sum_{i=1}^{d}\mathcal{Y}_i}\right]$. It is easy to check that $g_d(C+\beta) = 0$ and $g_d'(C+\beta) = 0$, and a standard result on the Fenchel-Legendre transformations is that $g_d''(t) = \frac{1}{\psi_d''(p_d(t))}$ where $p_d(t)$ is the unique number such that $g_d(t) = p_d(t)t - \psi_d(p_d(t))$. Now by Taylor's expansion formula we have that

$$g_d(C) \leq \left(\sup_{C\leq t\leq C+\beta} g_d''(t)\right)\frac{\beta^2}{2} = \left(\frac{1}{\inf_{C\leq t\leq C+\beta}\psi_d''(p_d(t))}\right)\frac{\beta^2}{2} \tag{D.23}$$

We have that $\psi_d'(p) = \sum_{i=1}^{d}\frac{\mathrm{E}\left[\mathcal{Y}_i e^{pY_i}\right]}{\mathrm{E}\left[e^{p\mathcal{Y}_i}\right]}$ and

$$\psi_d''(p) = \sum_{i=1}^{d}\left(\frac{\mathrm{E}\left[\mathcal{Y}_i^2 e^{p\mathcal{Y}_i}\right]}{\mathrm{E}\left[e^{p\mathcal{Y}_i}\right]} - \left(\frac{\mathrm{E}\left[\mathcal{Y}_i e^{p\mathcal{Y}_i}\right]}{\mathrm{E}\left[e^{p\mathcal{Y}_i}\right]}\right)^2\right) \geq \sum_{i=1}^{d}\frac{\mathrm{E}\left[\mathcal{Y}_i e^{p\mathcal{Y}_i}\right]}{\mathrm{E}\left[e^{p\mathcal{Y}_i}\right]} = \psi_d'(p).$$

Now, $p_d(t) \geq \lambda$ when $C \leq t \leq C+\beta$. This implies that $\psi_d''(p(t)) \geq \psi_d'(\lambda) = C$ when $C \leq t \leq C+\beta$. Combining this with eq. (D.22) and eq. (D.23) we get that

$$\Pr[S_d = C] \geq e^{-\frac{\beta^2}{2C}}\Pr\left[\sum_{i=1}^{d} V_i = C\right] \geq e^{-\frac{\beta^2}{C}}\Pr\left[\sum_{i=1}^{d} V_i = C\right] \,. \tag{D.24}$$

To complete the proof of eq. (D.21), it thus suffices to show that $\Pr\left[\sum_{i\in[k]} V_i = C\right] = \frac{1}{4\sqrt{C}}$. We use Lemma D.24 to get that,

$$\Pr\left[\sum_{i=1}^{d} V_i = C\right] \geq \frac{1}{\sqrt{2\pi \sum_{i=1}^{d} \operatorname{Var}[V_i]}} - K_2 \left(\frac{\sum_{i=1}^{d} \operatorname{E}\left[|V_i - \operatorname{E}[V_i]|^3\right]}{\left(\sum_{i=1}^{d} \operatorname{Var}[V_i]\right)^{3/2}}\right)^2$$

Now we use that $\operatorname{E}[V_i] \leq \operatorname{Var}[V_i] \leq 2\operatorname{E}[V_i]$, $|V_i - \operatorname{E}[V_i]|^3 \leq 25\operatorname{E}[V_i]$, and $\sum_{i=1}^{d} \operatorname{E}[V_i] = C$ to get that,

$$\Pr\left[\sum_{i=1}^{d} V_i = C\right] \geq \frac{1}{\sqrt{4\pi C}} - 25^2 K_2 \frac{1}{C^2}$$

We know that $C \geq \frac{1}{4L}$ so if we choose L sufficiently small we get that,

$$\Pr\left[\sum_{i=1}^{d} V_i = C\right] \geq \frac{1}{4\sqrt{C}} .$$

This leads to the desired bound.

We finish the section by proving Claim D.25.

Proof of Claim D.25. We start by using eq. (D.20) to get that,

$$\frac{\Pr[S_d = C]}{\Pr[S_d = C-1]} = e^{-\lambda} \frac{\Pr\left[\sum_{i=1}^{d} V_i = C\right]}{\Pr\left[\sum_{i=1}^{d} V_i = C-1\right]}$$

We want to argue that $\frac{\Pr[\sum_{i=1}^{d} V_i = C]}{\Pr[\sum_{i=1}^{d} V_i = C-1]} \geq \max\left\{e^{-1/8}, e^{-\frac{1}{8}\beta/C}\right\}$. First we use Lemma D.24 to get that,

$$\frac{\Pr\left[\sum_{i=1}^{d} V_i = C\right]}{\Pr\left[\sum_{i=1}^{d} V_i = C-1\right]} \geq \frac{\frac{1}{\sqrt{2\pi \sum_{i=1}^{d} \operatorname{Var}[V_i]}} - K_2 \left(\frac{\sum_{i=1}^{d} \operatorname{E}\left[|V_i - \operatorname{E}[V_i]|^3\right]}{\left(\sum_{i=1}^{d} \operatorname{Var}[V_i]\right)^{3/2}}\right)^2}{\frac{1}{\sqrt{2\pi \sum_{i=1}^{d} \operatorname{Var}[V_i]}} e^{-1/(2\sum_{i=1}^{d} \operatorname{Var}[V_i])} + K_2 \left(\frac{\sum_{i=1}^{d} \operatorname{E}\left[|V_i - \operatorname{E}[V_i]|^3\right]}{\left(\sum_{i=1}^{d} \operatorname{Var}[V_i]\right)^{3/2}}\right)^2}$$

Now we use that $\operatorname{E}[V_i] \leq \operatorname{Var}[V_i] \leq 2\operatorname{E}[V_i]$, $|V_i - \operatorname{E}[V_i]|^3 \leq 25\operatorname{E}[V_i]$, and $\sum_{i=1}^{d} \operatorname{E}[V_i] = C$ to get that,

$$\frac{\frac{1}{\sqrt{2\pi \sum_{i=1}^{d} \operatorname{Var}[V_i]}} - K_2 \left(\frac{\sum_{i=1}^{d} \operatorname{E}\left[|V_i - \operatorname{E}[V_i]|^3\right]}{\left(\sum_{i=1}^{d} \operatorname{Var}[V_i]\right)^{3/2}}\right)^2}{\frac{1}{\sqrt{2\pi \sum_{i=1}^{d} \operatorname{Var}[V_i]}} e^{-1/(2\sum_{i=1}^{d} \operatorname{Var}[V_i])} + K_2 \left(\frac{\sum_{i=1}^{d} \operatorname{E}\left[|V_i - \operatorname{E}[V_i]|^3\right]}{\left(\sum_{i=1}^{d} \operatorname{Var}[V_i]\right)^{3/2}}\right)^2} \geq \frac{1 - 25^2 K_2 \sqrt{8\pi \frac{1}{C}}}{e^{-1/(4C)} + 25^2 K_2 \sqrt{8\pi \frac{1}{C}}}$$

Using that $e^{-1/(4c)} \leq 1 - \frac{1}{8c}$ we the get that,

$$\frac{1 - 25^2 K_2 \sqrt{8\pi \frac{1}{C}}}{e^{-1/(4C)} + 25^2 K_2 \sqrt{8\pi \frac{1}{C}}} \geq \frac{1 - 25^2 K_2 \sqrt{8\pi \frac{1}{C}}}{1 - \frac{1}{8c} + 25^2 K_2 \sqrt{8\pi \frac{1}{C}}} = 1 - \frac{2 \cdot 25^2 K_2 \sqrt{8\pi \frac{1}{C}} - \frac{1}{8c}}{1 - \frac{1}{8c} + 25^2 K_2 \sqrt{8\pi \frac{1}{C}}}$$

We now that $C \geq \frac{1}{4L}$ so choosing L sufficiently small it holds that

$$\frac{2 \cdot 25^2 K_2 \sqrt{8\pi \frac{1}{C}} - \frac{1}{8c}}{1 - \frac{1}{8c} + 25^2 K_2 \sqrt{8\pi \frac{1}{C}}} \leq \frac{2 \cdot 25^2 K_2 \sqrt{8\pi}}{\sqrt{C}}$$

This implies that,

$$\frac{\Pr\left[\sum_{i=1}^{d} V_i = C\right]}{\Pr\left[\sum_{i=1}^{d} V_i = C - 1\right]} \geq 1 - \frac{2 \cdot 25^2 K_2 \sqrt{8\pi}}{\sqrt{C}}$$

Clearly, $1 - \frac{2 \cdot 25^2 K_2 \sqrt{8\pi}}{\sqrt{C}} \geq e^{-1/8}$ by choosing L small enough. We also note that,

$$1 - \frac{2 \cdot 25^2 K_2 \sqrt{8\pi}}{\sqrt{C}} \geq e^{-\frac{4 \cdot 25^2 K_2 \sqrt{8\pi}}{\sqrt{C}}} \geq e^{-\frac{M}{8\sqrt{C}}} \geq e^{-\frac{\beta}{8C}} .$$

By choosing M large enough. The last inequality follows since $\beta \geq M\sqrt{C}$.

We now have to bound λ. We define the function

$$h(x) = \sum_{i=1}^{d} \alpha_i e^x (1 - \alpha_i e^x)^{-1} .$$

We note that $h(0) = \sum_{i=1}^{d} \mathrm{E}[\mathcal{Y}_i] = C + \beta$. We take the derivative of h twice and get that,

$$h'(x) = \sum_{i=1}^{d} \alpha_i e^x (1 - \alpha_i e^x)^{-2}$$

$$h''(x) = \sum_{i=1}^{d} a e^x (1 + e^x)(1 - \alpha_i e^x)^{-3}$$

We note that $h'(x) \geq 0$ and $h''(x) \geq 0$ for all x so h is a monotonically increasing convex function, and $h'(0) = \sum_{i=1}^{d} \mathrm{Var}[\mathcal{Y}_i] \leq \sum_{i=1}^{d} 2\mu_i = 2(C + \beta)$.

If $\beta \geq C$ then again using that h is convex we get that,

$$h(-\tfrac{1}{4}) \geq h(0) - \tfrac{1}{4} h'(0) \geq C + \beta - 2\tfrac{1}{4}(C + \beta) \geq C .$$

Since h is increasing then it implies that $\lambda \leq -\frac{1}{4}$ and we get that,

$$\frac{\Pr[S_d = C]}{\Pr[S_d = C - 1]} = e^{-\lambda} \frac{\Pr\left[\sum_{i=1}^{d} V_i = C\right]}{\Pr\left[\sum_{i=1}^{d} V_i = C - 1\right]} \geq e^{\frac{1}{4}} e^{-\frac{1}{8}} = e^{-\frac{1}{8}} .$$

If $\beta < C$ then using that h is convex we get that,

$$\begin{aligned} h(-\tfrac{1}{4}\beta/C) &\geq h(0) - \tfrac{1}{4}\beta/Ch'(0) \geq C + \beta - 2\tfrac{1}{4}\beta/C(C+\beta) \\ &= C + (1 - 2\tfrac{1}{4} - 2\tfrac{1}{4}\beta/C)\beta \geq C + (1 - 4\tfrac{1}{4})\beta \geq C \ . \end{aligned}$$

Since h is increasing then it implies that $\lambda \leq -\frac{1}{4}\beta/C$ and we get that,

$$\frac{\Pr[S_d = C]}{\Pr[S_d = C-1]} = e^{-\lambda}\frac{\Pr\left[\sum_{i=1}^d V_i = C\right]}{\Pr\left[\sum_{i=1}^d V_i = C-1\right]} \geq e^{\frac{1}{4}\beta/C}e^{-\frac{1}{8}\beta/C} = e^{\frac{1}{8}\beta/C} \ .$$

We now focus on the upper bound. We use eq. (D.20) to get that,

$$\frac{\Pr[S_d = C-1]}{\Pr\left[S_d = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]} = e^{-\lambda\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil}\frac{\Pr\left[\sum_{i=1}^d V_i = C-1\right]}{\Pr\left[\sum_{i=1}^d V_i = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]}$$

We start by lower bounding λ. We first note that $h'(x) \leq h(x)$ for all x. Using that h is convex we get that,

$$C + \beta = h(0) \geq h(-\beta/C) + \beta/Ch'(-\beta/C) \geq \frac{C+\beta}{C}h(-\beta/C) \ .$$

This implies that $h(-\beta/C) \leq C$ and $\lambda \geq -\beta/C$. Now we will bound $\frac{\Pr\left[\sum_{i=1}^d V_i = C-1\right]}{\Pr\left[\sum_{i=1}^d V_i = C-1-\left\lceil\frac{C}{\beta}\right\rceil\right]}$. We will again use Lemma D.24.

$$\begin{aligned} &\frac{\Pr\left[\sum_{i=1}^d V_i = C-1\right]}{\Pr\left[\sum_{i=1}^d V_i = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]} \\ &\leq \frac{\frac{1}{\sqrt{2\pi\sum_{i=1}^d \mathrm{Var}[V_i]}}e^{-1/(2\sum_{i=1}^d \mathrm{Var}[V_i])} + K_2\left(\frac{\sum_{i=1}^d \mathrm{E}\left[|V_i - \mathrm{E}[V_i]|^3\right]}{\left(\sum_{i=1}^d \mathrm{Var}[V_i]\right)^{3/2}}\right)^2}{\frac{1}{\sqrt{2\pi\sum_{i=1}^d \mathrm{Var}[V_i]}}e^{-(1+\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil)^2/(2\sum_{i=1}^d \mathrm{Var}[V_i])} - K_2\left(\frac{\sum_{i=1}^d \mathrm{E}\left[|V_i - \mathrm{E}[V_i]|^3\right]}{\left(\sum_{i=1}^d \mathrm{Var}[V_i]\right)^{3/2}}\right)^2} \\ &\leq \frac{\frac{1}{\sqrt{2\pi C}}e^{-1/(2\sum_{i=1}^d \mathrm{Var}[V_i])} + 25^2K_2\frac{1}{C}}{\frac{1}{\sqrt{2\pi C}}e^{-(1+\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil)^2/(2\sum_{i=1}^d \mathrm{Var}[V_i])} - 25^2K_2\frac{1}{C}} \end{aligned}$$

Now we note that since $\beta \geq M\sqrt{C}$ then we get that $(1 + \frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil)^2/(2\sum_{i=1}^d \mathrm{Var}[V_i]) \leq \frac{1}{12}$. We then get that,

$$\frac{\Pr\left[\sum_{i=1}^d V_i = C-1\right]}{\Pr\left[\sum_{i=1}^d V_i = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]} \leq \frac{\frac{1}{\sqrt{2\pi C}} + 25^2K_2\frac{1}{C}}{\frac{1}{\sqrt{2\pi C}}e^{-1/12} - 25^2K_2\frac{1}{C}} \leq e^{1/6}$$

The last inequality follows by $C \geq \frac{1}{4L}$ and choosing L small enough. We then get that,

$$\begin{aligned}\frac{\Pr[S_d = C-1]}{\Pr\left[S_d = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]} &= e^{-\lambda\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil}\frac{\Pr\left[\sum_{i=1}^{d} V_i = C-1\right]}{\Pr\left[\sum_{i=1}^{d} V_i = C-1-\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil\right]} \\ &\leq e^{\frac{\beta}{C}\frac{1}{4}\left\lceil\frac{C}{\beta}\right\rceil}e^{1/6} \\ &\leq e^{1/2+1/6} \\ &< 2\end{aligned}$$

□

D.6 The Number of Bins Visited During an Insertion

This section is dedicated to proving the part of Theorem D.3 concerning insertions, which we restate below.

Theorem D.26. *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$ with $1/\varepsilon = n^{o(1)}$. Let $C = (1+\varepsilon)n/m$. Suppose we insert n balls into m bins, each of capacity C, using consistent hashing with bounded loads and virtual bins having k levels where $k = c/\varepsilon^2$ for c a sufficiently large universal constant. The expected number of bins visited during an insertion of a ball is $O(1/f)$.*

In fact, the proof uses only that the total number of non-full bins is $\Theta(f)$ with high probability, not the concrete value of f. Therefore the complicated expression for f will never occur in the proof of the theorem. All we will occasionally use is the fact that the number of non-full bins is $\Omega(\varepsilon)$, which follows trivially from a combinatorial argument.

The section is structured as follows: We start by providing some preliminaries for the proof of Theorem D.26 in Appendix D.6.1. In Appendix D.6.2, we use the results from Appendix D.5 to provide a strengthening of Lemma D.13. Finally, we provide the proof of Theorem D.26 in Appendix D.6.3.

D.6.1 Preliminaries For the Analysis

We start by making the following definition which will be repeatedly be useful in the analysis to follow.

Definition D.27. Consider any distribution of n balls into m bins. We say that a bin is *close to full* if it contains more than $(1+\varepsilon/2)n/m$ balls. Otherwise, we say that it is *far from full*.

Suppose we distribute n balls into m bins each of capacity $C = (1+\varepsilon)n/m$ using consistent hashing with bounded loads and virtual bins. By Theorem D.4, the number of non-full bins is $\Theta(fm)$ with high probability when $k = O(1/\varepsilon^2)$ is sufficiently large. We claim that it also holds that the number of far from full bins is $\Theta(fm)$ with high

probability. To see this, suppose that after distributing the n balls into the m bins of capacity $C = (1+\varepsilon)n/m$ each, we reduce the capacity of each bin to $C_0 = (1+\varepsilon/2)n/m$. This requires forwarding balls from the now overflowing bins and this forwarding can only increase the number of bins containing $(1+\varepsilon/2)n/m$ balls. By Theorem D.4, and with $\varepsilon_0 = \varepsilon/2$, the number of non-full bins after the relocating is $\Theta(f_0 m)$, where

$$f_0 = \begin{cases} \varepsilon_0 C_0, & C_0 \leq \log(1/\varepsilon_0) \\ \varepsilon_0 \sqrt{C_0 \log\left(\frac{1}{\varepsilon_0\sqrt{C_0}}\right)}, & \log(1/\varepsilon_0) < C_0 \leq \frac{1}{2\varepsilon_0^2} \\ 1, & \frac{1}{2\varepsilon_0^2} \leq C_0. \end{cases}$$

But clearly, $f_0 = \Theta(f)$, so we conclude that the number of far from full bins before modifying the system is $\Theta(fm)$ with high probability.

Summing up, we have the following corollary to Theorem D.4.

Corollary D.28. *In the setting of Theorem D.26, the number of far from full bins is $\Theta(fm)$ with high probability, i.e., with probability $1 - n^{-\gamma}$ for every γ=O(1).*

Finally, recall Definition D.15: The *run* at a given level i containing some virtual bin b, is the maximal interval at level i which contains b and satisfies that all bins lying in I gets full at level i.

D.6.2 High Probability Bound on the Number of Bins Visited in an Insertion

This section will be dedicated to prove the following strengthening of Lemma D.13.

Theorem D.29. *Let $n, m \in \mathbb{N}$ and $0 < \varepsilon < 1$ with $1/\varepsilon = n^{o(1)}$. Suppose we distribute n balls into m bins, each of capacity $C = (1+\varepsilon)n/m$, using consistent hashing with bounded loads and virtual bins and $k = c/\varepsilon^2$ levels for a sufficiently large constant c. Let b be a bin at level i which may be chosen dependently on the hashing of balls and bins to level $1, \ldots, i-1$ and I the run at level I containing b. Let X denote the number of bins in I. For any $t \geq 1/f$,*

$$\Pr[X \geq t] = \exp(-\Omega(tf)) + O(n^{-10})$$

The same statement holds even if b is given an extra start load of $\lambda t f \lceil C\varepsilon/2 \rceil$ 'artificial' balls before the hashing of balls and bins to level i, where λ is a sufficiently small constant.

Note that it in particular follows that the number of bins visited at a given level during an insertion is $O(\log(1/\delta)/f)$ with probability $1 - \delta$.

Proof. Let R denote the number of virtual bins in I. By Corollary D.28, the number of far from full bins after inserting balls at level $1, \ldots, i-1$ is at least $c_0 fm$ with high probability, where $c_0 > 0$ is some universal constant. Furthermore, by a standard Chernoff bound, the number of balls hashing to level i is at most $2n/k$ with high probability. Here we used the assumption that $1/\varepsilon = n^{o(1)}$, so $n \gg k \log n$. Condition on those two events and consider

the following modified process at level i where (1) b and every bin which was close to full after inserting the balls at level $1, \dots, i-1$ forwards every ball it receives at level i, i.e., has its remaining capacity reduced to zero (2) each far from full bin stores at most $\lceil C\varepsilon/2 \rceil$ balls from level i before it starts forwarding balls at level i, i.e., has its remaining capacity reduced to $\lceil C\varepsilon/2 \rceil$. Let I' denote the run containing b with such modified capacities. Letting R' denote the number of virtual bins lying in I it then holds that $R \leq R'$, so it suffices to provide a high probability upper bound on R'.

Let $s \in \mathbb{N}$ be given and let A_s be the event that $s+1 \leq R' \leq 2s+1$. Define J_1^- and J_1^+ to be respectively the intervals at level i ending and starting at b and having length $s/(4m)$. Similarly, let J_2^- and J_2^+ be respectively the intervals at level i ending and starting at b and having length $4s/m$. We observe that if A_s occur then either of the following events must hold.

B_1: J_2^- or J_2^+ contains at most $3s$ virtual bins.

B_2: J_1^- or J_1^+ contains at least $s/2$ virtual bins

B_3: J_1^- or J_1^+ contains at most $c_0 fs/8$ virtual bins which were far from full from levels $1, \dots, i-1$

B_4: $J_2^- \cup J_2^+$ contains at least $\lceil C\varepsilon/2 \rceil \cdot c_0 fs/8$ balls.

Indeed, suppose that A_s occur and that neither of B_1, B_2, B_3 occur. We show that then B_4 must occur. To see this observe that if B_1 does not occur, then since I' consists of at most $2s+1$ bins, $I' \subseteq J_2^- \cup J_2^+$. Since B_2 does not occur, I' must further fully contain J_1^- or J_1^+. Since B_3 does not occur, I' must then contain at least $c_0 fs/8$ virtual bins which were far from full from levels $1, \dots, i-1$. Finally any ball allocated to a bin of I' must also hash to I'. Since the at least $c_0 fs/8$ far from full bins from level $1, \dots, i-1$ which lie in I' each get full at level i and has a total capacity of $\lceil C\varepsilon/2 \rceil \cdot c_0 fs/8$, it follows that at least $\lceil C\varepsilon/2 \rceil \cdot c_0 fs/8$ balls must hash to $I' \subseteq J_2^- \cup J_2^+$. This is exactly the event B_4.

As in the proof of Lemma D.13, we can use standard Chernoff bounds to conclude that $\Pr[B_1] = \exp(-\Omega(s))$, $\Pr[B_2] = \exp(-\Omega(s))$ and $\Pr[B_3] = \exp(-\Omega(fs))$. For B_4, we observe that the expected number of balls, μ, hashing to $J_2^- \cup J_2^+$ is upper bounded by $2n/k \cdot 8s/m = O(Cs/k)$. As $f = \Omega(\varepsilon)$, we may assume that $k \geq c'/(\varepsilon f)$ for any constant c'. Thus, choosing c' sufficiently large, it follows that $\mu \leq Cs\varepsilon f c_0/32$. Using another Chernoff bound, it follows $\Pr[B_4] = \exp(-\Omega(fs))$. In conclusion, if $s \geq 1/f$, it holds that $\Pr[A_s] = \exp(-\Omega(fs))$ and the desired result follows as in the proof of Lemma D.13.

Finally, it is easy to modify the constants in the above argument, so that it carries through even when b is given an extra start load of $\lambda t f \lceil C\varepsilon/2 \rceil$ balls for a sufficiently small constant λ, and this gives the final statement of the Theorem.

□

D.6.3 The Proof of Theorem D.26

In this section, we provide the proof of Theorem D.26. In order to do so, we first require a technical lemma which for a given virtual bin, b, bounds the number of balls that are

either placed in b or forwarded from b at level i. The technique used to prove this lemma will be used for the final proof of Theorem D.26, but in a more sophisticated way. As such, the lemma below serves as a nice warm up to the proof of Theorem D.26. We start out by choosing $s^* = O(\log n/f)$ sufficiently large, such that Theorem D.29 yields that for a bin b at level i, the length of the run containing b at level i (see Definition D.15) has length at most s^* with probability $1 - 1/n^{10}$.

Lemma D.30. *Let $\lambda = O(1)$ be any constant. Let b be a virtual bin at level i that may depend on the distribution of balls into bins at level $1, \ldots, i-1$. Let n_i denote the number of balls hashing to level i and suppose that $n/(2k) \leq n_i \leq 2n/k$ where $k = O(1/\varepsilon^2)$ is sufficiently large (depending on λ). Let Z denote the number of the n_i balls hashing to level i that either are placed in b or are forwarded from b. Define $\alpha = \lceil C\varepsilon/2 \rceil$. For any $\ell \geq 1/5$ satisfying that $\alpha\ell$ is an integer[5], it holds that*

$$\Pr[Z \geq \alpha\ell] \leq e^{-\lambda\ell} + 1/n^{10}.$$

Proof. We define A_ℓ to be the event that $Z \geq \alpha\ell$. When upper bounding the probability of A_ℓ we may assume that every bin which was close to full at level $i-1$ forwards all balls landing in it at level i. We may further assume that any bin which was far from full at level $i-1$ stores exactly $\alpha = \lceil C\varepsilon/2 \rceil$ balls and then starts forwarding balls. Let Z' denote the number of balls landing in b or being forwarded from b at level i in this modified process. Then clearly, $Z' \geq Z$ so $\Pr[Z \geq \alpha\ell] \leq \Pr[Z' \geq \alpha\ell]$.

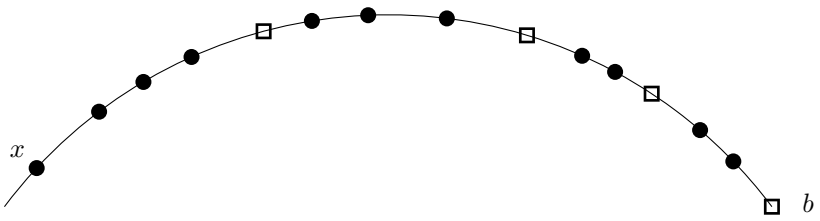


Figure D.1: s bins that are far from full and $\alpha s + \alpha\ell$ balls. The bins are represented as boxes and the balls as disks.

Next note that if $Z' \geq \alpha\ell$, then there must an integer $s \geq 0$ and an interval of the i'th level ending in b which contains exactly s virtual bins which are far from full and exactly $\alpha s + \alpha\ell$ balls. See Figure D.1. Indeed, of the ℓ balls landing or being forwarded from b consider the one hashing furthest behind b at level i, call it x. Let s be the the number of far from full bins hashing between x and b at level i. Aside from the $\alpha\ell$ balls landing in b or being forwarded from b, there must hash enough balls between x and s to put α balls in each of the far from full bins between x and b, and thus the interval between x and b contains exactly s far from full bins and $\alpha s + \alpha\ell$ balls. We denote the event that there exists such an interval by $A_{\ell,s}$ noting that we may then upper bound $\Pr[A_\ell] \leq \sum_{s=0}^{s^*} \Pr[A_{\ell,s}] + 1/n^{10}$. Here we used that the run containing b has length at most s^* with probability at least $1 - 1/n^2$. We proceed to upper bound $\Pr[A_{\ell,s}]$ for each $0 \leq s \leq s^*$.

[5]The constant 5 is arbitrary.

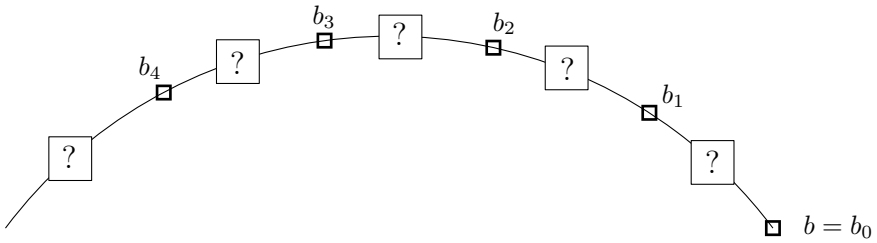


Figure D.2: We generate the number of balls hashing directly between b_i and b_{i+1} sequentially. In each step the number of such balls is dominated by a geometric distribution with parameter q.

So fix $s \geq 0$. We generate the sequence of the $s+1$ far from full bins $b = b_0, b_1, \ldots, b_{s+1}$ leading up to b and the balls hashing between them in a backwards order. Starting at b_0 we go backwards along the cyclic order. At some point we reach a bin, b_1 and we let X_0 be the number of balls met along the way in the between b_0 and b_1. We continue this was, going backwards until we have met $s+1$ bins $b_1, \ldots, b_{s+1}$ and for each $1 \leq i \leq s$ we let X_i be the number of balls met in the cyclic order between b_i and b_{i+1}. See Figure D.2 for an illustration of the process. Let f denote the fraction of bins which were far from full from level $1, \ldots, i-1$. As we saw after Definition D.27, $f \geq \varepsilon/3$. Now when going backwards from b_i until we get to b_{i+1}, the probability of meeting a ball in each step is upper bounded by $\frac{n_i}{n_i+mf-s} \leq \frac{n_i}{n_i+mf-s^*} := q$ regardless of the values of $X_0, \ldots, X_{i-1}$. Letting $X'_0, \ldots, X'_s$ be independent geometric variables with parameter $1-q$, $X = \sum_{i=0}^{s} X_i$, and $X' = \sum_{i=0}^{s} X'_i$ it follows tht for any $t > 0$, $\Pr[X \geq t] \leq \Pr[X' \geq t]$.

If $A_{\ell,s}$ holds, then $X \geq s\alpha + \ell\alpha$, so we may upper bound

$$\Pr[A_{\ell,s}] \leq \Pr[X' \geq s\alpha + \ell\alpha].$$

The expected value of X'_i is

$$\mathbb{E}[X'_i] = \frac{q}{1-q} = \frac{n_i}{mf - s^*} = O\left(\frac{n}{kmf}\right) = O\left(\frac{\alpha}{kf\varepsilon}\right) \leq \frac{\alpha}{\lambda_0}.$$

Here $\lambda_0 = O(1)$ is a sufficiently large constant which we will choose later. Here we again used the assumption that $1/\varepsilon = m^{o(1)}$ and moreover that $k = O(1/\varepsilon^2)$ is sufficiently large. It follows that $\mathbb{E}[X'] = \frac{(s+1)\alpha}{\lambda_0}$. Note in particular that we can ensure that $\mathrm{E}[X'] \leq \frac{s\alpha+\ell\alpha}{2}$, so that

$$\Pr[X' \geq s\alpha + \ell\alpha] \leq \Pr\left[X' \geq \mathrm{E}[X'] + \frac{s\alpha + \ell\alpha}{2}\right].$$

We apply Theorem D.12 to bound this quantity. If we are in the case, where we are to use the second bound of eq. (D.7), we obtain that

$$\Pr\left[X' \geq \mathrm{E}[X'] + \frac{s\alpha + \ell\alpha}{2}\right] \leq \left(1 - \frac{1}{1 + 2\alpha/\lambda_0}\right)^{\frac{s\alpha+\ell\alpha}{4}},$$

It is easy to check that $\left(1 - \frac{1}{1+2\alpha/\lambda_0}\right)^{\alpha}$ can be made smaller than any sufficiently small constant, just by choosing λ_0 sufficiently large. Thus it follows that

$$\Pr[X' \geq s\alpha + \ell\alpha] \leq e^{-\lambda_1(s+\ell)}, \tag{D.25}$$

where we can make $\lambda_1 = O(1)$ sufficiently large. However, we may have to use the first bound of eq. (D.7) and we investigate now which bound we obtain in this case. Relating back to Theorem D.12, we define $\mu_0 = \mathrm{E}[X'_i] \leq \alpha/\lambda_0$, $A = \left(1 + \frac{1}{2\mu_0}\right) \log\left(1 + \frac{1}{2\mu_0}\right)$ and $t = \frac{1}{4\sigma^2}\frac{s\alpha+\ell\alpha}{2}$. We further define $\sigma^2 = \mathrm{Var}[X'] = (s+1)\,\mathrm{Var}[X'_i] = (s+1)\mu_0(1+\mu_0)$. If $\mu_0 \geq 1$, then $A \leq 1/\mu_0$ and $\sigma^2 \leq (s+1)2\mu_0^2$, so that

$$A\sigma^2 \leq 2(s+1)\mu_0 \leq \frac{2(s+1)\alpha}{\lambda_0} < \frac{s\alpha+\ell\alpha}{8} = t\sigma^2,$$

by choosing λ_0 large enough. Thus, in this case we obtain the bound in eq. (D.25). If on the other hand $\mu_0 < 1$, then

$$t \geq \frac{s\alpha+\ell\alpha}{16(s+1)\mu_0} \geq \frac{\lambda_0(s+\ell)}{16(s+1)} \geq \lambda_2$$

for a sufficiently large constant λ_2. Then also $W_0(t)$ can be made larger than any given constant, so we obtain that the bound of eq. (D.25) holds in general.

We now sum over s to obtain that

$$\Pr[A_\ell] \leq 1/n^{10} + \sum_{s=0}^{s^*} \Pr[A_{\ell,s}] \leq 1/n^{10} + \sum_{s=0}^{s^*} e^{-\lambda_1(s+\ell)} \leq 1/n^{10} + e^{-\lambda\ell},$$

where again λ can be made sufficiently large. This completes the proof. □

With this lemma in hand we are ready to proceed with the proof of Theorem D.26. To guide the reader, we will start by providing a high level idea of how to obtain the result as follows. First of all, it will be helpful to recall in details how an insertion of a ball is handled using consistent hashing with bounded loads and virtual bins. When inserting a ball, x, we uniformly hash x to a random point at a random level. Suppose that the hash value of x, $h(x)$, lies in the i'th level i for some i. Starting at $h(x)$ we walk along level i until we arrive at a virtual bin. If the virtual bin is filled to its capacity with balls hashing to level $1, \ldots, i$, we forward a ball from that bin at level i (it could be x but it could also be another ball that hashed to level i of lower priority than x). We repeat the step, continuing to walk along level i until we meet a new virtual bin. The first time we meet a virtual bin, b, which was not filled to its capacity with balls hashing to level $1, \ldots, i$, we insert the forwarded ball and find the smallest level $j > i$ such that the virtual bin of b at level j received a ball at level j. If no such level exists, the insertion is completed. Otherwise b has an overflow of one ball at level j, and we continue the insertion walking along level j starting at b. Theorem D.26 claims that the expected number of bins visited during this entire process is upper bounded by $O(1/f)$.

The idea of in our proof of Theorem D.26 is to split the bins visited during the insertion of x into *epochs*. An epoch starts by visiting $\lceil 1/f \rceil$ virtual bins of the insertion (unless of course the insertion is completed before that many bins has been seen). The last of these $\lceil 1/f \rceil$ virtual bins lies at some level i and we finish the epoch by completing the forwarding of balls needed at level i. At this point, we are either done with the insertion or we need to forward a ball from some virtual bin at some level $j > i$. The next epochs are similar; having finished epoch $a-1$, in epoch a, we visit $\lceil 1/f \rceil$ virtual bins. At this point, we will be at some level ℓ if we are not already done with the insertion. We then finish the part of the insertion which takes place at level ℓ. Importantly, at the beginning of each epoch, we have just arrived at a virtual bin at a completely fresh level.

The proof shows that during the first $\lceil 1/f \rceil$ steps of an epoch, the probability of finishing the insertion in each step is $1 - \Omega(f)$. The intuition for this, is that when we reach a bin b at some level, i, the probability that b is far from full from other levels than i can be showed to be $\Omega(f)$. Since the number of levels $k = O(1/\varepsilon^2)$ is large, the contribution from level i to b only fills b with probability $1-\Omega(1)$. Thus, the probability of not finishing the insertion during the first $\lceil 1/f \rceil$ steps of an epoch is $(1-\Omega(f))^{\lceil 1/f \rceil} = e^{-\Omega(1)} = 1-\Omega(1)$. Now conditioning on not finishing the insertion during the first $\lceil 1/f \rceil$ steps of an epoch, we can still show that the expected number of bins visited during the rest of the epoch is $O(1/f)$. Letting $\mathcal{E}$ denote the event of finishing the insertion during the first $\lceil 1/f \rceil$ of an epoch and T, the total number of bins visited during the insertion, we have on a very high level that

$$\mathbb{E}[T] \leq \Pr[\mathcal{E}]\lceil 1/f \rceil + \Pr[\mathcal{E}^c](O(1/f) + \mathbb{E}[T]) = O(1/f) + \Pr[\mathcal{E}^c]\mathbb{E}[T] = O(1/f) + p\mathbb{E}[T], \tag{D.26}$$

where $p = 1 - \Omega(1)$. Solving this equation, we find that $\mathbb{E}[T] = O(1/f)$. Here it should be noted that the recursive formula (D.26) is a bit too simplified. In our analysis, the $\mathbb{E}[T]$ on the left hand side and on the right hand side of (D.26) will not exactly be the same. The point is that after finishing epoch a, and being ready to start epoch $a+1$ at a new level j, we will know a bit more about the hashing of balls to level $1, \ldots, j-1$ than we did before the beginning of epoch a. However, using Lemma D.29, we know that it is only a relatively small fraction of the system that we have any information about, and so we can argue that the expectation does not change much.

With this intuition in mind, our next goal is to obtain Theorem D.26.

Proof of Theorem D.26. As described above, we partition the insertion into *epochs* where an epoch consists of the following two steps.

1. We go through $\lceil 1/f \rceil$ bins of the insertion ending in a bin at some level ℓ.
2. We continue the insertion at level ℓ until we arrive at some bin b which does not get full at level ℓ.

After step 2. we will have to continue the insertion on some level $j > i$ (if b gets full at that level). Note that the insertion will complete during an epoch if along the way, we meet a bin which does not get full on either of levels $1, \ldots, k$. We will prove the following more technical claim which implies Theorem D.26.

Claim D.31. *Let $D > 0$ be any constant and $0 \leq t \leq D \log n$. Condition on the event that the insertion has been through t epochs so far. Let $\mathcal{E}$ denote the event that we finish the insertion at one of the first $\lceil 1/f \rceil$ bins met during step 1. of epoch $t+1$. Further define R to be the random variable which counts the number of bins visited during step 2. of epoch $t+1$ (if the insertion completes before we get to step 2. we put $R = 0$). Then*

$$\Pr[\mathcal{E}] \geq c, \tag{D.27}$$

for some universal constant $c > 0$ (which does not depend on D), and

$$\mathbb{E}[R \mid \mathcal{E}^c] = O(1/f). \tag{D.28}$$

Before proving the claim, we argue how the desired result follows. First of all, choosing $D = 2/c$, it follows from (D.27) that the probability of not finishing the insertion during the first $D \log n$ epochs is upper bounded by

$$(1-c)^{D \log n} \leq \exp(-2 \log n) \leq n^{-2}.$$

Conditioned on this extremely low probability event, the expected time for the insertion is crudely and trivially upper bounded by mk, but $mkn^{-2} \ll 1$, so this has no influence on the expected number of bins visited during the insertion, as we will now formalize. For $1 \leq i \leq D \log n$, we let X_i denote the expected number of bins visited during the insertion starting from epoch i. If the insertion finishes before epoch i, we let $X_i = 0$. Let further $\mathcal{E}_i$ denote the probability of finishing the insertion during step 1. of epoch i. Finally, let R_i denote the number of bins visited during step 2. of epoch i. Then, for any $0 \leq i \leq D \log n$, it holds that

$$\mathbb{E}[X_i] \leq \Pr[\mathcal{E}_i] \cdot \lceil 1/f \rceil + \Pr[\mathcal{E}_i^c](\mathbb{E}[R_i \mid \mathcal{E}_i^c] + \mathbb{E}[X_{i+1}]).$$

By the claim, $\Pr[\mathcal{E}_i^c] \leq 1 - c$ and $\mathbb{E}[R_i \mid \mathcal{E}_i^c] = O(1/f)$, so we obtain that

$$\mathbb{E}[X_i] \leq O(1/f) + (1-c)\mathbb{E}[X_{i+1}].$$

Solving this recursion, we obtain that

$$\mathbb{E}[X_0] = O(1/f) + (1-c)^i \mathbb{E}[X_{i+1}],$$

so putting $i = D \log n$, we obtain that $\mathbb{E}[X_0] = O(1/f) + n^{-2} \cdot \mathbb{E}[X_{C \log n + 1}] = O(1/f)$. But $\mathbb{E}[X_0]$ is exactly the expected number of bins visited during an insertion. It thus suffices to prove the claim which is the main technical challenge of the proof.

Proof of Claim D.31. We split the proof into the proofs of equations (D.27) and (D.28).

Proof of Equation (D.27)

It suffices to show that for each of the $\lceil 1/f \rceil$ bins visited during step 1. of the epoch, the probability of ending the insertion at that bin is $\Omega(f)$. More formally, we let $\mathcal{A}_i$ denote the event that the i'th of these bins, $1 \leq i \leq \lceil 1/f \rceil$ is still full, i.e., that we do not end the insertion at the i'th bin, and show that $\Pr[\mathcal{A}_i] \leq (1-\Omega(f))^i + im^{-1/2+o(1)}$. The probability of not completing the insertion during step 1. of the epoch is then upper bounded by $(1-\Omega(f))^{\lceil 1/f \rceil} + \lceil 1/f \rceil m^{-1/2+o(1)} \leq (1-\Omega(f))^{\lceil 1/f \rceil} + o(1) \leq e^{-\Omega(1)} := c$ which is the desired result. Here we used that $1/f \leq O(1/\varepsilon) = m^{o(1)}$.

We will condition on $\mathcal{A}_{i-1}$ so start by making the conditioning more precise by describing exactly how the bins met before the i'th bin of the epoch at the given level received enough ball to make them full. We then bound the probability of $\mathcal{A}_i$ conditioned on this history. So fix i with $1 \leq i \leq \lceil 1/f \rceil$. The conditioning on $\mathcal{A}_{i-1}$ means that we have already seen $i-1$ full bins during the epoch. Suppose that the i'th bin, call it b, is at some level ℓ. We then in particular know that the number of bins we have already visited at level ℓ is at most $i-1 \leq 1/f$. Let $a \geq 0$ denote the number of bins already visited on level ℓ. Going backwards from $b := b_a$, we denote these bins $b_{a-1}, \ldots, b_0$. Thus b_0 was the first bin ever visited at level ℓ. Note that possibly $b_a = b_0$. The conditioning $\mathcal{A}_{i-1}$ especially implies that after level ℓ, all bins $b_0, \ldots, b_{a-1}$ got filled. We now describe how these bins got filled at level ℓ as follows (see also Figure D.3 for an illustration of the process). Starting with $j = 0$, if the remaining capacity of b_0 after levels $1, \ldots, \ell-1$ is C_0, we go backwards until at some point we have met a set of bins of total remaining capacity C^* and exactly $C^* + C_0$ balls for some C^*. After this sequence, we insert a question mark **?**. This sequence of bins and balls describes how b_0 received its C_0 balls, and the **?** indicates a yet unknown history. We next go backwards from b_1 which has remaining capacity C_1, say. If we arrive at b_0 before having seen C_1 balls get we simply skip past the history of how b_0 got fills and continue the process after the **?**. If we obtain the description of how b_1 got filled at level ℓ before reaching b_0, there might still be more balls hashing between b_0 and b_1 (but no bins). In this case we insert a question mark, **?**, after the sequence of balls leading up to b_1. More generally, for $j = 1, \ldots, a-1$, we go backwards from b_j generating a sequence of balls. Whenever we reach a bin, we go back to the nearest **?** and start generating balls at that point until we find a new bin or are done with describing the filling of b_j — In the later case we insert a new **?**. The **?** before bin b_0 has a special status. If we ever reach it, and we still require C_j balls to be filled, we go backwards until we have found a set of bins of total remaining capacity C^* and exactly $C^* + C_0$ balls for some C^*. It should be remarked that there is nothing probabilistic going on here. We have simply explained a way to find the positions of a set of balls and bins which certify how bins $b_0, \ldots, b_{a-1}$ got filled at level ℓ. See Figure D.3 for an example of how this description of how bins $b_0, \ldots, b_{a-1}$ got filled at level ℓ can look.

We let $\mathcal{O}$ denote the event that bin b receives more than $\lceil C\varepsilon/2 \rceil$ bins from level ℓ. We also let $\mathcal{N}$ denote the event that b receives at least $\frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)$ balls from the levels different than ℓ. We then get that,

$$\Pr[\mathcal{A}_i] \leq \Pr[\mathcal{A}_{i-1} \wedge (\mathcal{O} \vee \mathcal{N})] \leq \Pr[\mathcal{A}_{i-1}] \Pr[\mathcal{O} \mid \mathcal{A}_{i-1}] + \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge \mathcal{N}] \ .$$

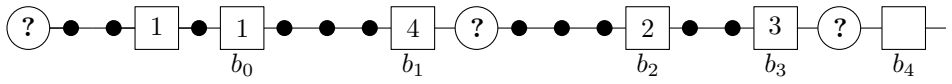


Figure D.3: The filling of bins $b_0, \ldots, b_3$ at level ℓ. The bins are represented as boxes an the numbers within them describes their remaining capacity at level ℓ. The balls are represented as disks and the question marks **?** in circles.

We will next show that $\Pr[\mathcal{O} \mid \mathcal{A}_{i-1}] = p$, where $p = 1 - \Omega(1)$, and

$$\Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge \mathcal{N}] \leq \Pr[\mathcal{A}_{i-1}] \Pr[\mathcal{O}^c \mid \mathcal{A}_{i-1}] (1 - c_0 f) + m^{-1/2+o(1)}$$

where $c_0 = \Omega(1)$ is a universal constant. This will then imply that,

$$\Pr[\mathcal{A}_i] \leq \Pr[\mathcal{A}_{i-1}] (1 - (1-p)c_0 f) + m^{-1/2+o(1)} \leq (1 - (1-p)c_0 f)^i + i m^{-1/2+o(1)} .$$

We again split the proof into two parts.

Bounding $\Pr[\mathcal{O} \mid \mathcal{A}_{i-1}]$: In the following we will omit the conditioning of $\mathcal{A}_{i-1}$ from the notation to avoid clutter. We have described how bins $b_0, \ldots, b_{a-1}$ got filled at level ℓ. This included a tail of balls behind each bin as well as some positions marked with **?**. Let $s+1$ be the number of such **?**-marks including the mark behind bin b_0. (See Figure D.3). Then $s \leq a$. Let X_0 denote the number of balls being forwarded to b_a from the backmost **?** before b_0 and let $X_1, \ldots, X_s$, denote the number of balls forwarded to b_a from the remaining positions marked with a **?**. The number of balls, n_ℓ , hashing to level ℓ lies between $n/(2k)$ and $2n/k$ with probability $1 - O(n^{-10})$ by a standard Chernoff bound. Moreover, the total number of bins lying in the history described so far is $s^* = O(\frac{\log n}{f})$ with probability $1 - O(n^{-10})$, by Lemma D.29 including those bins landing before b_0 in the description. Now conditioning on this history, for each $1 \leq j \leq s$

$$\mathrm{E}[X_j] \leq \frac{n_k}{m - s^*} = O(C/k).$$

It follows that

$$\mathrm{E}\left[\sum_{j=1}^{s} X_j\right] = O(sC/k) = O(C/(fk)) = O(C/(\varepsilon k)).$$

If, we choose $k = O(1/\varepsilon^2)$ sufficiently large, it in particular follows that $\mathrm{E}\left[\sum_{j=1}^{s} X_j\right] \leq \lceil C\varepsilon/2 \rceil/20$. Thus, by Markov's inequality,

$$\Pr\left[\sum_{j=1}^{s} X_j \geq \lceil C\varepsilon/2 \rceil/2\right] \leq 1/10. \tag{D.29}$$

Next, we show that $\Pr[X^* \geq \lceil C\varepsilon/2 \rceil/2] \leq 1/10$. From this it will follow that, $\Pr[\mathcal{O}] \leq 1/5$ which is what we need. For bounding this probability, we may use Lemma D.30. To get

into the setting of that lemma, we may simply contract the interval of the cyclic order from the most backwards **?** to b_a and remove all unresolved **?** in between except for the most backwards one. That the other places marked with **?** now cannot receive any balls only increases the probability that $X^* \geq t$ for any t. Now we are exactly in the setting of Lemma D.30, which we apply with $\ell = 1/2$ to conclude that if $k = O(1/\varepsilon^2)$ is sufficiently large, then

$$\Pr[X^* \geq \lceil C\varepsilon/2 \rceil/2] \leq 1/10.$$

The reader may note that as an alternative to the reduction above (contracting the so far described history of how the bins $b_0, \ldots, b_{a-1}$ received their balls), we may simply reprove Lemma D.30 in this a tiny bit more complicated setting. The arguments would remain exactly the same.

In conclusion, we have now argued that $\Pr[\mathcal{O} \mid \mathcal{A}_{i-1}] \leq 1/5$.

Bounding $\Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge \mathcal{N}]$: We start by defining notation which we used in Appendix D.5.1. Let Y_i be the number of balls which land in bin b or which are forwarded by bin b on level i. We define $Y_{<\ell} = \sum_{i<\ell} Y_i$ and $Y_{>\ell} = \sum_{i>\ell} Y_i$. With this notation we get that

$$\Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge \mathcal{N}] = \Pr\left[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + Y_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right]$$

We let let $\mathcal{L}_\ell$ be the sigma-algebra generated by the random choices on the first ℓ levels, and A_d will be the event as defined in Appendix D.5.1.

We recall the simpler system from Appendix D.5.1 which we will compare to. Let $\mathcal{Y}_i$ be the number of balls which land in bin b or which are forwarded by bin b on level i in the simpler system. We similarly define $\mathcal{Y}_{<\ell} = \sum_{i<\ell} \mathcal{Y}_i$ and $\mathcal{Y}_{>\ell} = \sum_{i>\ell} \mathcal{Y}_i$.

We will prove that,

$$\Pr\left[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + Y_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] \tag{D.30}$$

$$\leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\left[\mathcal{Y}_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] + m^{-1/2+o(1)} \tag{D.31}$$

This will imply the result since

$$\Pr\left[\mathcal{Y}_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] \leq \Pr\left[\sum_{i=1}^{k} \mathcal{Y}_i \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right]$$

Now using Theorem D.16 we get that $\Pr\left[\sum_{i=1}^{k} \mathcal{Y}_i \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] \leq \Pr\left[\sum_{i=1}^{k} Y_i \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] + m^{-1/2+o(1)}$, and the discussion at the start of Appendix D.6.1 give us that $\Pr\left[\sum_{i=1}^{k} Y_i \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\right] \leq 1 - c_0 f$. Thus we just need to prove eq. (D.30).

We start by noticing that,

$$\begin{aligned}\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_< + Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big] \\ = \sum_{s=0}^{\frac{n}{m}(1+\lceil C\varepsilon/2\rceil)-1} \Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} = s \wedge Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s\Big] \\ + \Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big]\end{aligned}$$

We fix $0 \leq s \leq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - 1$ and get that,

$$\begin{aligned}\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} = s \wedge Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s\Big] \\ = \mathrm{E}\Big[[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} = s] \Pr\Big[Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s \,\Big|\, \mathcal{L}_l\Big]\Big]\end{aligned}$$

Now we use Lemma D.21 and get that $\Pr\big[Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s \,\big|\, \mathcal{L}_l\big] \leq \Pr\big[\mathcal{Y}_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s\big] + k\,[A_\ell^c] + (1+2k)m^{-1/2+o(1)}$. Using this we get that,

$$\begin{aligned}\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + Y_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big] \\ \leq \Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big] \\ + \sum_{s=0}^{\frac{n}{m}(1+\lceil C\varepsilon/2\rceil)-1} \mathrm{E}\Big[[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} = s]\left(k\,[A_\ell^c] + (1+2k)m^{-1/2+o(1)}\right)\Big]\end{aligned}$$

Now we note that,

$$\begin{aligned}\sum_{s=0}^{\frac{n}{m}(1+\lceil C\varepsilon/2\rceil)-1} \mathrm{E}\Big[[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} = s]\left(k\,[A_\ell^c] + (1+2k)m^{-1/2+o(1)}\right)\Big] \\ \leq k\Pr[A_\ell^c] + (1+2k)m^{-1/2+o(1)} \\ \leq km^{-\gamma} + (1+2k)m^{-1/2+o(1)} \\ \leq m^{-1/2+o(1)}\end{aligned}$$

The second last inequality uses Theorem D.16 and last uses that $k = m^{o(1)}$.

We also want to also exchange $Y_{<\ell}$ with $\mathcal{Y}_{<\ell}$ and we will do this in similar fashion.

$$\begin{aligned}\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big] \\ = \sum_{s=0}^{\frac{n}{m}(1+\lceil C\varepsilon/2\rceil)-1} \Pr[\mathcal{Y}_{>\ell} = s] \Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil) - s\Big] \\ + \Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1+\lceil C\varepsilon/2\rceil)\Big]\end{aligned}$$

Again we fix s and get that,

$$\begin{aligned}
&\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] \\
&\quad = \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s \,\Big|\, \mathcal{A}_{i-1} \wedge \mathcal{O}^c\Big] \\
&\quad \leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s \,\Big|\, \mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge A_{\ell-1}\Big] + \Pr\big[A^c_{\ell-1}\big]
\end{aligned}$$

By Theorem D.16 we know that $\Pr\big[A^c_{\ell-1}\big] \leq m^{-\gamma}$. Now similarly to $Y_{<\ell}$ we define $Y^{(j)}_{<\ell}$ to be the number of balls which lands in j or which are forwarded by bin j on levels before level ℓ. We know that b is chosen uniformly from the set $[m] \setminus \{b_0, \ldots, b_{a-1}\}$ so if we fix the first $\ell - 1$ then the probability that $Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s$ is equal to

$$\frac{\sum_{j \in [m] \setminus \{b_0, \ldots, b_{a-1}\}} \Big[Y^{(j)}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big]}{m - a}$$

Since we condition on $A_{\ell-1}$ then we have that,

$$\left| \frac{\sum_{j \in [m]} \Big[Y^{(j)}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big]}{m} - \Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] \right| \leq m^{-1/2+o(1)}$$

This implies that,

$$\begin{aligned}
&\left| \frac{\sum_{j \in [m] \setminus \{b_0, \ldots, b_{a-1}\}} \Big[Y^{(j)}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big]}{m - a} - \Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] \right| \\
&\qquad\qquad \leq \frac{m}{m-a} m^{-1/2+o(1)} + \frac{a}{m-a}
\end{aligned}$$

Now we use Lemma D.13 to get that $a \leq O(\log(m)/\varepsilon) = m^{o(1)}$ with probability $1 - m^{-\gamma}$. Here we use that $1/\varepsilon = m^{o(1)}$. Combining this we get that,

$$\begin{aligned}
&\Pr\Big[Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s \,\Big|\, \mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge A_{\ell-1}\Big] \\
&\quad \leq \Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] + \frac{m}{m - m^{o(1)}} m^{-1/2+o(1)} + \frac{m^{o(1)}}{m - m^{o(1)}} + m^{-\gamma} \\
&\quad \leq \Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] + m^{-1/2+o(1)}
\end{aligned}$$

We then get that,

$$\begin{aligned}
&\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] \\
&\quad = \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[Y_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s \,\Big|\, \mathcal{A}_{i-1} \wedge \mathcal{O}^c\Big] \\
&\quad \leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \left(\Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] + m^{-1/2+o(1)}\right) + m^{-\gamma} \\
&\quad \leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[\mathcal{Y}_{<\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil) - s\Big] + m^{-1/2+o(1)}
\end{aligned}$$

Using this we get that,

$$\begin{aligned}
&\Pr\Big[\mathcal{A}_{i-1} \wedge \mathcal{O}^c \wedge Y_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\Big] \\
&\qquad \leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[\mathcal{Y}_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\Big] \\
&\qquad\quad + \sum_{s=0}^{\frac{n}{m}(1+\lceil C\varepsilon/2 \rceil)-1} \Pr[\mathcal{Y}_{>\ell} = s]\, m^{-1/2+o(1)} \\
&\qquad \leq \Pr[\mathcal{A}_{i-1} \wedge \mathcal{O}^c] \Pr\Big[\mathcal{Y}_{<\ell} + \mathcal{Y}_{>\ell} \geq \frac{n}{m}(1 + \lceil C\varepsilon/2 \rceil)\Big] + m^{-1/2+o(1)}
\end{aligned}$$

This finishes the proof eq. (D.30).

This concludes the proof that equation (D.27) of the claim holds.

Proof of Equation (D.28)

We restate what we have to prove, namely that

$$\mathbb{E}[R \mid \mathcal{E}^c] = O(1/f),$$

Where R is the number of bins visited during step 2. of epoch $t+1$ and $\mathcal{E}^c$ is the event that we did not finish the insertion during step 1. of epoch $t+1$. Let $b_0, \ldots, b_a$ denote the bins that we have visited so far at the level where we are currently at, call it ℓ. All bins $b_0, \ldots, b_a$ got filled from levels $1, \ldots, \ell$, and as in the proof of equation (D.27) of the claim, we may again describe the history of how the bins $b_0, \ldots, b_a$ got filled to their capacity at level ℓ. See Figure D.4 for an example of such a history.

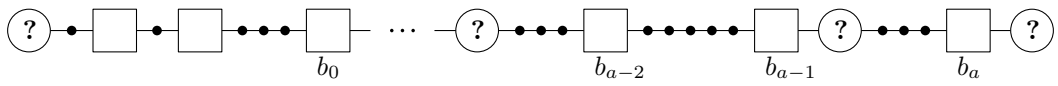


Figure D.4: An example of how the conditioning on $\mathcal{E}^c$ might look. Except for the **?** coming before b_0, the circled **?**'s, are parts of the cyclic order which has not yet been fixed, but which are known to consist solely of balls. The circled **?** appearing before b_0, which is the yet unknown history of how many balls b_0 are to further forward, has a special role. Indeed, this history does not have to consist solely of balls but can consist of a run of balls and bins such that all of the bins in the run gets filled at this level.

Let $s \geq 1/f$. We wish to argue that the conditional probability

$$\Pr[R \geq s \mid \mathcal{E}^c] = O(\exp(-\Omega(sf))) + O(n^{-10}). \tag{D.32}$$

Ignoring the unimportant $O(n^{-10})$ term, it will follow that

$$\begin{aligned}
\mathbb{E}[R \mid \mathcal{E}^c] &\leq 1/f + \sum_{i=0}^{\infty} \Pr[2^i/f \leq R \leq 2^{i+1}/f] 2^{i+1}/f \\
&= 1/f + O\left(\frac{1}{f} \sum_{i=0}^{\infty} \exp(-\Omega(2^i)) 2^i\right) = O(1/f).
\end{aligned}$$

and including the $O(n^{-10})$ term in the computation could only increase the bound with an additive n^{-8}, say, as we can here use the trivial bound on the length of a run of mk. Thus, this yields the desired result. For the bound on $\Pr[s \leq R \leq 2s \mid \mathcal{E}^c]$, it clearly suffices to assume that $s \geq c/f$ where $c = O(1)$ is a sufficiently large constant.

We start by noting that with probability $1 - O(n^{-10})$, the number of balls hashing to level ℓ is at most $2n/k$ which we assume to be the case in what follows. Let q denote the number of places marked with **?** between b_0 and b_a and let $X_1, \ldots, X_q$ denote the number of balls landing at these positions. Then $q \leq a \leq 1/f$. Let $\alpha = \lceil C\varepsilon/2 \rceil$. Let A denote the event that $X_1 + \cdots + X_q \geq \lambda s f \alpha$, where $\lambda = \Omega(1)$ is a sufficiently small constant to be chosen later. We start by providing an upper bound on $\Pr[A]$. For this, we let $X = \sum_{i=1}^{q} X_i$ and note, like in the proof of Lemma D.30, that for each i, X_i is dominated by a geometric variable with parameter q where $q = \frac{n_\ell}{m+n_\ell}$. Here $n_\ell = 2n/k$ is the upper bound on the number of ball hashing to level ℓ. Furthermore, this claim holds even conditioning on the values of $(X_j)_{j<i}$. Let $s' = s\lambda$. Letting $Y_1, \ldots, Y_{1/f}$ be independent such geometric variables and $Y = \sum_{i=1}^{1/f} Y_i$, we can thus upper bound

$$\Pr[A] \leq \Pr[Y \geq s' f \alpha].$$

Note that

$$\mathrm{E}[Y_i] \leq \frac{n_\ell}{m} \leq \frac{2C}{k} \leq \frac{4\alpha}{\varepsilon k}$$

for $1 \leq i \leq 1/f$, so that $\mathrm{E}[Y] \leq \frac{4\alpha}{\varepsilon f k} \leq \alpha$, were the last inequality follows by assuming that $k = O(1/\varepsilon^2)$ is sufficiently large. We may also assume that $s'f$ is larger than a sufficiently large constant, as described above, so we can upper bound

$$\Pr[A] \leq \Pr[Y \geq \mathrm{E}[Y] + s' f \alpha/2].$$

By applying the bound eq. (D.7) of Theorem D.12 similarly to how we did in the proof of Lemma D.30 it follows after some calculations that

$$\Pr[A] = \exp(-\Omega(sf)).$$

Now condition on A^c and let us focus on upper bounding $\Pr[R \geq s \mid \mathcal{E}^c \cap A^c]$. For this, we apply (D.29). To get into the setting of that theorem, we contract the part of the history revealed so far between the back-most **?**-mark before b_0 and up til and including b_a into a single *unified* bin. By the conditioning on A^c, this unified bin comes with an extra start load of at most $\lambda s f \alpha$ balls, where we can choose $\lambda = \Omega(1)$ to be any sufficiently small constant. Thus, with the conditioning, we are exactly in the setting to apply Theorem D.29, and we may thus bound

$$\Pr[R \geq s \mid \mathcal{E}^c \cap A^c] = \exp(-\Omega(sf)).$$

It follows that

$$\Pr[R \geq s \mid \mathcal{E}^c] \leq \Pr[A] + \Pr[R \geq s \mid \mathcal{E}^c \cap A^c] = \exp(-\Omega(sf)),$$

which is the desired. This completes the proof of Claim D.31. □

As explained before the proof of Claim D.31, this completes the proof of our theorem. □

D.7 Insertions of Bins and Deletions of Balls and Bins

In this section, we prove the statements of Theorems D.2 and D.3 concerning the deletions of balls and insertions and deletions of bins. Combined with the results of Appendices D.2, D.6 and D.8, this proves the two theorems in full.

Deletions of Balls. By the history independence, a deletion of a ball is symmetric to an insertion. The bins visited when deleting a ball x are the same as the bins visited if x had not been in the system and was inserted. Thus, we can upper bound the expected number of bins visited when deleting a ball by $O(1/\varepsilon)$ for Theorem D.2 and $O(1/f)$ for Theorem D.3. This also upper bounds the number of balls moved in a deletions.

Deletions of Bins. A deletion of a super bin is the same as reinserting the balls lying in that super bin. We claimed that that the expected cost of deleting a super bin is $O(C/f)$ in Theorem D.3. At first, this may seem completely obvious, since the cost of inserting a single ball is $O(1/f)$. However, this cost is for inserting a ball which is selected independently of the random choices of the hash function. Now, we are looking at the balls placed in a given super bin b, and those are highly dependent on the hash function. However, we do know that the expected average cost of all balls in the system is $O(1/f)$. Moreover, all bins are symmetric, so the bin b behaves like a random bin amongst those in the system. Thanks to our load balancing, the balls are almost uniformly spread between the bins, so a random ball from a random bin is almost a uniformly random ball, so a random ball from b has expected cost $O(1/f)$. There are at most C balls in b them, so the total expected cost is $O(C/f)$. A similar argument applies in the case of Theorem D.2.

Insertions of Bins Again, by the history independence an insertion of a bin is symmetric to its deletion. The balls that are moved when inserting a bin are thus the same as if that bin was in the system but was deleted. Thus we can use the result for deletions of bins to conclude the bound of $O(C/f)$ on the number of balls moved when inserting a bin. A similar argument applies in the case of Theorem D.2.

D.8 Faster Searches Using the Level-Induced Priorities

In this section we make the calculation demonstrating that giving the balls random priorities, we obtain the better bounds on the number of bins visited during an insertion as claimed in Theorems D.2 and D.3. This is not a new idea but is in fact an old trick [AK74; Knu73]. What we need to do is verify that applying it, with the particular formula for f in eq. (D.1), we obtain the stated search times. In fact, what we require for the analysis is only the fact that if two balls hash to different levels, the ball hashing to the lower level has the highest priority of the two. Within a given level, the priorities can be arbitrary. This is important for the practical version of our scheme described in Appendix D.1.3 where the priorities are not uniformly random and independent of the hashing of balls, but where the hashing of the balls in fact determines the priorities, with higher hash values

implying lower priorities. We start by arguing about the expected number of bins visited during a search as stated in Theorem D.2.

Number of Bins Visited During a Search: Theorem D.2. We encourage the reader to recall the setting described in the theorem. Define X to be the number of bins visited during the search for some ball x. Importantly, if x hashes to level i, then all virtual bins visited during the search of x also lie on level i. For $i \in [k]$, we let A_i denote the event that x hashes to level i, so that $\Pr[A_i] = p_i$. By a standard Chernoff bound, the number of balls hashing to the first i levels is $np_{\leq i} \pm m^{1/2+o(1)} = np_{\leq i}(1 \pm m^{-1/2+o(1)})$, with probability $1 - O(n^{-10})$, say. Here we used that $1/\varepsilon = m^{o(1)}$. Condition on this event and define $n_{<i}$ to be the number of balls hashing to the first i levels. Finally letting ε_i be such that that $(1+\varepsilon_i)n_{\leq i}/m = C$, we obtain from the part of Theorem D.2 concerning insertions (which was proved in Appendix D.2) that $\mathrm{E}[X_i \mid A_i] = O(1/\varepsilon_i)$. Moreover, $\Pr[A_i] \leq 2^{-i+1}$ for each i. It finally follows from the Chernoff bound above that $\varepsilon_i \geq 1/2^i$, and so

$$\mathrm{E}[X] = \sum_{i \in [k]} \mathrm{E}[X \mid A_i] \Pr[A_i] = O(k) = O(\log 1/\varepsilon)$$

as desired.

Number of Bins Visited During a Search: Theorem D.3. We now perform a similar calculation to the one above, in the more complicated setting of Theorem D.3. Let us for simplicity assume that the number of balls hashing to each level is exactly n/k. It is trivial to later remove this assumption. We also assume for simplicity that $k \geq 1/\varepsilon^2$ is a power of 2, $k = 2^a$ for some a. Let $\ell = \lceil \log(1/\varepsilon) \rceil$ noting that $\ell \leq a$. We partition $[k] = I_0 \cup \cdots I_\ell$, where $I_i = [2^a - 2^{a-i-1}] \setminus [2^a - 2^{a-i}]$ for $0 \leq i \leq \ell - 1$ and $I_\ell = [2^a] \setminus [2^a - 2^{a-\ell}]$. Let A_i be the event that the given ball to be searched x hashes to some level in I_i, so that $\Pr[A_i] = 2^{-i+1}$ for $0 \leq i \leq \ell - 1$ and $\Pr[A_\ell] = 2^{-\ell}$. For $0 \leq i \leq \ell$ we define $n_{\leq i}$ to be the number of balls hashing to some level in $I_0 \cup \cdots \cup I_i$. Finally, let ε_i be such that $(1 + \varepsilon_i)n_{\leq i}/m = C$ and note that $\Pr[A_i] = \Theta(\varepsilon_i)$.

We partition $[\ell + 1]$ into three sets, $[\ell + 1] = J_1 \cup J_2 \cup J_3$ where

$$J_1 = \{i \in [\ell+1] : C \leq \log 1/\varepsilon_i\}, \quad J_2 = \{i \in [\ell+1] : \log 1/\varepsilon_i < C \leq \frac{1}{2\varepsilon_i^2}\},$$
$$\text{and} \quad J_3 = \{i \in [\ell+1] : \frac{1}{2\varepsilon_i^2} < C\}.$$

It then follows from the part of Theorem D.3 dealing with insertions (proved in Ap-

pendix D.6) that

$$\mathrm{E}[X] = O\left(\sum_{i\in J_1}\frac{\Pr[A_i]}{\varepsilon_i C} + \sum_{i\in J_2}\frac{\Pr[A_i]}{\varepsilon_i\sqrt{C\log\left(\frac{1}{\varepsilon_i\sqrt{C}}\right)}} + \sum_{i\in J_3}\Pr[A_i]\right)$$
$$= O\left(1 + \frac{|J_1|}{C} + \sum_{i\in J_2}\frac{1}{\sqrt{C\log\left(\frac{1}{\varepsilon_i\sqrt{C}}\right)}}\right)$$

We have the trivial bound $|J_1| \leq \ell + 1 = O(\log 1/\varepsilon)$. Moreover, for $i \in J_2$, it holds that

$$e^{-C} \leq \varepsilon_i \leq \sqrt{\frac{1}{2C}},$$

and since $\varepsilon_i = \Theta(2^{-i})$, it follows that

$$\sum_{i\in J_2}\frac{1}{\sqrt{C\log\left(\frac{1}{\varepsilon_i\sqrt{C}}\right)}} = O\left(\frac{1}{\sqrt{C}}\sum_{i=1}^{O(C)}\frac{1}{\sqrt{i}}\right) = O(1).$$

In conclusion,

$$\mathrm{E}[X] = O\left(1 + \frac{\log 1/\varepsilon}{C}\right),$$

and splitting into the cases, $C \leq \log 1/\varepsilon$ and $C < \log 1/\varepsilon$, we obtain the desired result.

D.9 The Practical Implementation.

In this section we sketch why our results continue to holds when using the practical implementation described in Appendix D.1.3 even when the hashing is implemented using the practical mixed tabulation scheme from [DKRT15]. Let us call the implementation from Appendix D.1.3 the *practical implementation*.

We first discuss the practical implementation with fully random hashing. For this, recall the definition of a run (Definition D.15). Using a similar argumentation to the one used in the proof of Lemma D.13, it is easy to show that in this implementation, for any constant $\gamma = O(1)$, the maximal number of bins in a run is $O((\log n)/\varepsilon)$ with probability $1 - n^{-\gamma}$. Denote this high probability event $\mathcal{E}$. The number of balls lying in a run consisting of ℓ bins is trivially upper bounded by $C(\ell+1)$, so if $\mathcal{E}$ occurs, the maximal number of balls hashing to a fixed run is $O(C(\log n)/\varepsilon)$. It follows that the number of balls that are forwarded past any given point is $O(C(\log n)/\varepsilon)$. In particular for any level i, the number of balls that are forwarded from level i to level $i+1$ is $O(C(\log n)/\varepsilon)$ and the total number of such balls over all levels is $O(kC(\log n)/\varepsilon) = m^{o(1)}$. One can now modify our inductive proof of Theorem D.4 to check that its statement remains valid even with

the influence of these extra balls. Recall that in Theorem D.4, $X_{i,j}$ denoted the number of bins with at most j balls after the hashing of balls to levels $0, \dots, i-1$. Intuitively, in the inductive step, these $m^{o(1)}$ extra balls can only affect $m^{o(1)}$ bins which does not affect the high probability bound stating that $|X_{i,j} - \mu_{i,j}| \leq m^{1/2+o(1)}$. To exclude the bad event $\mathcal{E}^c$, we simply use a union bound and that $\mathcal{E}$ happened with very high probability. Once we have a version of Theorem D.4 which holds in the practical implementation, we can repeat the proof of Theorem D.26, again using union bounds for the event that the insertion interacts with the run of size $m^{o(1)}$ entering the given level from below.

Let us now discuss the implementation with mixed tabulation. A mixed tabulation hash function h is defined using two of the simple tabulation hash functions from [PT12], $h_1 : \Sigma^c \to \Sigma^d$ and $h_2 : \Sigma^{c+d} \to R$. Here Σ is some character alphabet with $\Sigma^c = [u]$ and $c, d = O(1)$ are constants. Then for a key x, $h(x) = h_2(x, h_1(x))$. An important property of mixed tabulation, proved in [DKRT15], is the following: Suppose X is a set of keys, $p_1, \dots, p_b$ are output bit positions and $v_1, \dots, v_b$ are desired bit values. Let Y be the set of keys $x \in X$ for which the p_i'th output bit $h(x)_{p_i} = v_i$ for all i. If $\mathrm{E}[|Y|] \leq |\Sigma|/(1+\Omega(1))$, then the remaining output bits of the hash values in Y are completely independent with probability $1 - O(|\Sigma|^{1-\lfloor d/2 \rfloor})$. Another important property is that mixed tabulation obeys the same concentration bounds as simple tabulation on the number of balls landing in an interval [PT12].

For the implementation with mixed tabulation, we use k independent mixed tabulation functions, $h_1, \dots, h_k$, to distribute the virtual bins, and a single mixed tabulation function h^* for the balls (independent of $h_1, \dots, h_k$). We moreover assume that $|\Sigma| = u^{1/c} = n^{\Omega(1)}$ which can be achieved using a standard universe reduction. To obtain our results using mixed tabulation, the idea is essentially the same as above. Again, we first need to prove an analogue of Theorem D.4, and we would do this using induction on the level, bounding $|X_{i,j} - \mu_{i,j}|$ with high probability for each level i. To do this, we partition level i into dyadic intervals where we expect at most $|\Sigma|/2$ balls or bins to hash. Then we can use the concentration bound from [PT12] (which also holds for mixed tabulation) to obtain concentration on the number of bins of a given capacity from the previous levels hashing to each interval. Moreover, we can use the result of [DKRT15] to conclude that restricted to such an interval the hashing of balls and bins is fully random. Again, we can prove a version of Lemma D.13 with mixed tabulation (by using that mixed tabulation provides concentration bounds) and conclude that the total number of balls that are forwarded from one interval to another is $O(C(\log n)/\varepsilon) = m^{o(1)} = |\Sigma|^{o(1)}$. Essentially, the good distribution of the $X_{i-1,j}$ ensures that we also obtain a good distribution of the number of bins with each capacity in each of the intervals of level i (using that the influence of the $|\Sigma|^{o(1)}$ balls passing between intervals can only affect $|\Sigma|^{o(1)}$ bins), and this gives a good distribution of the $X_{i,j}$. For this, it is important to be aware that there are now more intervals, essentially $n/|\Sigma|$, but since $|\Sigma| = n^{\Omega(1)}$, we still obtain that the total number of balls that are forwarded from one interval to another is $n^{1-\Omega(1)}$. The high probability bound we obtain on $|X_{i,j} - \mu_{i,j}|$ then instead takes the form $|X_{i,j} - \mu_{i,j}| = n^{1-\Omega(1)}$, but this still suffices for our purposes. Finally, we may prove a mixed tabulation version of Theorem D.26, again using the fully random hashing within each interval and using union bounds to bound away the probability that we interact with the $|\Sigma|^{o(1)}$ balls that

are forwarded between intervals. As such, showing that our results hold using mixed tabulation uses essentially the same ideas as is needed to show that the implementation in Appendix D.1.3 does, but with a finer partitioning into intervals.

D.10 Acknowledgement

Research supported by Investigator Grant 16582, Basic Algorithms Research Copenhagen (BARC), from the VILLUM Foundation.

References

[AAKT21] Anders Aamand, Noga Alon, Jakob Bæk Tejs Knudsen, and Mikkel Thorup. *On Sums of Monotone Random Integer Variables*. 2021. arXiv: 2104.03721 [math.PR].

[AK74] Ole Amble and Donald E. Knuth. "Ordered Hash Tables". In: *Comput. J.* 17.2 (1974), pp. 135–142.

[Azu67] Kazuoki Azuma. "Weighted sums of certain dependent random variables". In: *Tohoku Mathematical Journal* 19.3 (1967), pp. 357–367.

[BG07] G. E. Blelloch and D. Golovin. "Strongly History-Independent Hashing with Applications". In: *Proc. 48th IEEE Symposium on Foundations of Computer Science (FOCS)*. 2007, pp. 272–282.

[BSS00] André Brinkmann, Kay Salzwedel, and Christian Scheideler. "Efficient, distributed data placement strategies for storage area networks". In: *Proceedings of the Twelfth annual ACM Symposium on Parallel Algorithms and Architectures, SPAA*. 2000, pp. 119–128.

[CDKR02] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony IT Rowstron. "SCRIBE: A large-scale and decentralized application-level multicast infrastructure". In: *Selected Areas in Communications, IEEE Journal on* 20.8 (2002), pp. 1489–1499.

[DKRT15] Søren Dahlgaard, Mathias Bæk Tejs Knudsen, Eva Rotenberg, and Mikkel Thorup. "Hashing for Statistics over K-Partitions". In: *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS*. 2015, pp. 1292–1310.

[GH05] George Giakkoupis and Vassos Hadzilacos. "A scheme for load balancing in heterogenous distributed hash tables". In: *Proceedings of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing, PODC*. 2005, pp. 302–311.

[GM14] Spencer Greenberg and Mehryar Mohri. "Tight lower bound on the probability of a binomial exceeding its expectation". In: *Statistics & Probability Letters* 86 (2014), pp. 91–98. ISSN: 0167-7152.

[GF04] David A. Grossman and Ophir Frieder. *Information Retrieval - Algorithms and Heuristics, Second Edition*. Vol. 15. The Kluwer International Series on Information Retrieval. Kluwer, 2004. ISBN: 978-1-4020-3004-8.

[KLLP+97] David R. Karger, Eric Lehman, Frank Thomson Leighton, Rina Panigrahy, Matthew S. Levine, and Daniel Lewin. "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web". In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, STOC*. 1997, pp. 654–663.

[KR06] David R. Karger and Matthias Ruhl. "Simple Efficient Load-Balancing Algorithms for Peer-to-Peer Systems". In: *Theory Comput. Syst.* 39.6 (2006). Announced at SPAA'05, pp. 787–804.

[KM05] Krishnaram Kenthapadi and Gurmeet Singh Manku. "Decentralized algorithms using both local and random probes for P2P load balancing". In: *SPAA 2005: Proceedings of the 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures*. 2005, pp. 135–144.

[Knu73] Donald E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching*. Addison-Wesley, 1973.

[Lar88] Per-Åke Larson. "Dynamic Hash Tables". In: *Commun. ACM* 31.4 (1988), pp. 446–457.

[Man04] Gurmeet Singh Manku. "Balanced binary trees for ID management and load balance in distributed hash tables". In: *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing, PODC*. 2004, pp. 197–205.

[MZ17] Vahab Mirrokni and Morteza Zadimoghaddam. "Consistent Hashing with Bounded Loads". In: *Google Research Blog* April 3 (2017). https://research.googleblog.com/2017/04/consistent-hashing-with-bounded-loads.html.

[MTZ18] Vahab S. Mirrokni, Mikkel Thorup, and Morteza Zadimoghaddam. "Consistent Hashing with Bounded Loads". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*. Ed. by Artur Czumaj. SIAM, 2018, pp. 587–604.

[ÖV11] M. Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems, Third Edition*. Springer, 2011. ISBN: 978-1-4419-8833-1.

[PT12] Mihai Pătraşcu and Mikkel Thorup. "The Power of Simple Tabulation-Based Hashing". In: *Journal of the ACM* 59.3 (2012). See also STOC'11, Article 14.

[RFHK+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. *A scalable content-addressable network*. Vol. 31. ACM, 2001.

[Rod16] Andrew Rodland. "Improving load balancing with a new consistent-hashing algorithm". In: *Vimeo Engineering Blog* December 19 (2016). https://medium.com/vimeo-engineering-blog/improving-load-balancing-with-a-new-consistent-hashing-algorithm-9f1bd75709ed.

[RD01] Antony Rowstron and Peter Druschel. "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems". In: *Middleware 2001*. Springer. 2001, pp. 329–350.

[SMKK+01] Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. "Chord: A scalable peer-to-peer lookup service for internet applications". In: *ACM SIGCOMM Computer Communication Review* 31.4 (2001), pp. 149–160.

[SMLK+03] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. "Chord: a scalable peer-to-peer lookup protocol for internet applications". In: *IEEE/ACM Trans. Netw.* 11.1 (2003), pp. 17–32.

[TR98] David Thaler and Chinya V. Ravishankar. "Using name-based mappings to increase hit rates". In: *IEEE/ACM Trans. Netw.* 6.1 (1998), pp. 1–14.

Appendix E

On Sums of Monotone Random Integer Variables

On Sums of Monotone Random Integer Variables

Anders Aamand* Noga Alon† Jakob B. T. Knudsen*
Mikkel Thorup*

Dansk resumé

We say that a random integer variable X is *monotone* if the modulus of the characteristic function of X is decreasing on $[0, \pi]$. This is the case for many commonly encountered variables, e.g., Bernoulli, Poisson and geometric random variables. In this note, we provide estimates for the probability that the sum of independent monotone integer variables attains precisely a specific value. We do not assume that the variables are identically distributed. Our estimates are sharp when the specific value is close to the mean, but they are not useful further out in the tail. By combining with the trick of *exponential tilting*, we obtain sharp estimates for the point probabilities in the tail under a slightly stronger assumption on the random integer variables which we call *strong monotonicity*.

E.1 Introduction

In this note we provide sharp estimates for the probability that the sum of independent (not necessarily identically distributed) integer-valued random variables attains precisely a specific value. Our estimates hold under a fairly general assumption on the properties of the random variables, which for example is satisfied for Bernoulli, Poisson and geometric random variables. The bounds on the point probabilities derived in this paper have been used to understand the distribution of balls in capacitated bins [AKT21]. In the cleanest combinatorial variant of the problem, where the balls arrive sequentially and each ball picks a uniformly random non-full bin, they just needed the point probabilities of sums of i.i.d. Bernoulli variables. However, for a more dynamic distribution system, they had to apply the bounds for sums of a mix of Bernoulli and geometrically distributed variables.

Recall that for a real random variable X, the *characteristic function* of X is the map $f_X : \mathbb{R} \to \mathbb{C}$ given by $f_X(\lambda) = \mathrm{E}[e^{i\lambda X}]$. We say that a real random variable X is *monotone* if $|f_X|$ is decreasing on $[0, \pi]$. In the first part of this note (Section E.2), we provide estimates for the point probabilities of a sum, $X = \sum_{j\in[k]} X_j$, of independent monotone random integer variables[1]. To be precise, for any given $t \in \mathbb{Z}$, we estimate the probability

*Basic Algorithms Research Copenhagen (BARC), University of Copenhagen, Denmark. Emails: aa@di.ku.dk, jakn@di.ku.dk, and mikkel2thorup@gmail.com. BARC is supported by the VILLUM Foundation grant 16582.

†Department of Mathematics, Princeton University, Princeton, New Jersey, USA and Schools of Mathematics and Computer Science, Tel Aviv University, Tel Aviv, Israel. Email: nalon@math.princeton.edu. Research supported in part by NSF grant DMS-1855464, BSF grant 2018267 and the Simons Foundation.

[1]We define $[k] = \{0, \ldots, k-1\}$

$\Pr[X = t]$. Our estimates are sharp whenever t is close to the mean $\mathrm{E}[X]$, but they are not useful further out in the tail. To handle point probabilities in the tail, we require a slightly stronger assumption on the random variables which we now describe.

For a random integer variable X we define $I_X = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X}] < \infty\}$, to consist of those $\theta \in \mathbb{R}$ for which the moment generating function of X is defined. We note that I_X is an interval with $0 \in I_X$. For $\theta \in I_X$, we may define the *exponentially tilted* random variable X_θ by $\Pr[X_\theta = t] = \frac{\Pr[X=t]e^{\theta t}}{\mathrm{E}[e^{\theta X}]}$ for $t \in \mathbb{Z}$. We say that X is *strongly monotone* if (1) $I_X \neq \{0\}$ and (2) X_θ is monotone for each $\theta \in I_X$. In the second part of this note (Section E.3), we use the trick of exponential tilting to provide estimates for the point probabilities of a sum of independent strongly monotone random integer variables, $X = \sum_{j\in[k]} X_j$, which are also sharp in the tail.

It follows by direct computation that Bernoulli, Poisson, and geometric random variables are monotone, and moreover, that exponentially tilting these variables again yields Bernoulli, Poisson and geometric variables. In particular, these variables are all strongly monotone, so our results give sharp estimates for the point probabilities of the sum of (a mix of) such variables. In Section E.3, we provide examples of the estimates that can be obtained for such a sum using our results.

In the note we will consider the following setting. Let k be an integer and $(X_j)_{j\in[k]}$ independent integer-valued random variables with $\mathrm{E}[X_j] = \mu_j$ and $\mathrm{Var}[X_j] = \sigma_j^2$ for $j \in [k]$. Let $X = \sum_{i\in[k]} X_i$, and further $\mu = \sum_{j\in[k]} \mu_j$ and $\sigma^2 = \sum_{j\in[k]} \sigma_j^2$ be respectively the expectation and variance of X. The main result of the note is the following theorem.

Theorem E.1. *There exists a universal constant c, such that if X is monotone, then for every t for which $\mu + t\sigma$ is an integer, the probability that X is precisely $\mu + t\sigma$ satisfies,*

$$\left|\Pr[X = \mu + t\sigma] - \frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2}\right| \leq c\left(\frac{\sum_{j\in[k]} \mathrm{E}\left[|X_j - \mu_j|^3\right]}{\sigma^3}\right)^2. \tag{E.1}$$

Remark E.2. We note that if each X_j is monotone, then X is as well. Indeed, the characteristic function of X can be factorized as $f_X(\lambda) = \prod_{j\in[k]} f_{X_j}(\lambda)$. In particular, Theorem E.1 holds when each of the variables $(X_j)_{j\in[k]}$ is monotone.

Our result is reminiscent of the Berry-Esseen theorem, but instead of bounding the distance between the cumulative function of X and the cumulative function of the normal distribution as the Berry-Esseen theorem does, our result bounds the distance between the density function of X and the density function of the normal distribution. This setting has been studied before in the context of large deviation theory, e.g., by Blackwell and Hodges [BJ59] and by Iltis [Ilt95] in the d-dimensional case. They do not require X to be monotone but they only consider the case where $(X_j)_{j\in[k]}$ are identically distributed and are interested in the asymptotical behavior when $k \to \infty$. In particular the distribution of the variables $(X_j)_{j\in[k]}$ cannot depend on k. McDonald [McD79] considers variables that are not necessarily identically distributed but again in the limit $k \to \infty$ and with certain extra assumptions on the distribution of the variables. In this work we are not interested in such asymptotic bounds and our result is a uniform bound for monotone variables.

Another line of related work is the asymptotic expansions in the local limit theorem. See for instance [GK54] (Section 51) or [IL71] (Theorems 4.5.3 and 4.5.4) which for any given r provide expressions for the point probabilities of a sum of k i.i.d. random integer variables $(X_j)_{j\in[k]}$ up to an additive error of $o(k^{-r})$. Again, these results holds asymptotically as $k \to \infty$ and the distributions of the $(X_j)_{j\in[k]}$ cannot depend on k. The case of different distributions is considered in [DH22] which provide expansions of the point probabilities using regular and trigonometric polynomials up to an additive error of $o(\sigma^{-r})$ (without any assumptions of monotonicity).

E.2 Point Probabilities Near the Mean

The goal of this section is to prove Theorem E.1, but before diving into the proof, we provide some examples of random variables for which the condition of the lemma is satisfied. Let $p \in [0,1]$ and $\lambda > 0$. Let Y be a Bernoulli variable with $\Pr[Y = 1] = 1-\Pr[Y = 0] = p$, let Z be geometric with $\Pr[Z = k] = p^k(1-p)$ for $k \in \mathbb{N}_0$, and let W be Poisson with $\Pr[W = k] = \lambda^k e^{-\lambda}/k!$ for $k \in \mathbb{N}_0$. Let f_Y, f_Z, f_W be the characteristic functions for Y, Z, and W. Then for $\lambda \in \mathbb{R}$,

$$f_Y(\lambda) = 1 - p + pe^{i\lambda}, \quad f_Z(\lambda) = \frac{1-p}{1-pe^{i\lambda}}, \quad \text{and} \quad f_W(\lambda) = e^{\lambda(e^{i\lambda}-1)}.$$

Thus,

$$\begin{aligned}
|f_Y(\lambda)|^2 &= (1-p+pe^{i\lambda})(1-p+pe^{-i\lambda}) = 1 + 2p(1-p)(\cos\lambda - 1)\\
|f_Z(\lambda)|^2 &= \left(\frac{1-p}{1-pe^{i\lambda}}\right)\left(\frac{1-p}{1-pe^{-i\lambda}}\right) = \frac{(1-p)^2}{1+p^2-2p\cos\lambda}, \quad \text{and}\\
|f_W(\lambda)| &= e^{\lambda(\cos\lambda-1)},
\end{aligned}$$

which are all decreasing functions on $[0,\pi]$.

We will need the following simple Lemma on random integer variables.

Lemma E.3. *Let X be an integer random variable X with third moment. Then*[2]

$$\mathrm{E}[|X - \mathrm{E}[X]|^3] \geq \mathrm{Var}[X]/2.$$

Proof. Let $\mu = \mathrm{E}[X]$. We may clearly assume that $0 \leq \mu < 1$ by replacing X with $X - a$ for an appropriate integer a. Define $Z_0 = [X \leq 0]$ and $Z_1 = [X \geq 1] = 1 - Z_0$. Then,

$$0 = \mathrm{E}[X-\mu] = \mathrm{E}[Z_1 \cdot |X-\mu|] - \mathrm{E}[Z_0 \cdot |X-\mu|],$$

and,

$$\mathrm{E}[|X-\mu|] = \mathrm{E}[Z_1 \cdot |X-\mu|] + \mathrm{E}[Z_0 \cdot |X-\mu|].$$

[2]Originally, our bound was $\mathrm{E}[|X-\mathrm{E}[X]|^3] \geq \mathrm{Var}[X]/10$, which sufficed for our purposes. We thank Ahmad Beirami [Bei] for pointing out how we could replace the constant 10 with 2 which is optimal as can be seen by letting X be a Bernoulli variable with parameter $1/2$.

It follows that

$$\begin{aligned}\mathrm{Var}[X] &= \mathrm{E}[Z_1 \cdot (X-\mu)^2] + \mathrm{E}[Z_0 \cdot (X-\mu)^2] \\ &\geq (1-\mu)\,\mathrm{E}[Z_1 \cdot |X-\mu|] + \mu\,\mathrm{E}[Z_0 \cdot |X-\mu|] = \frac{\mathrm{E}[|X-\mu|]}{2}. \end{aligned} \tag{E.2}$$

Now finally,

$$\mathrm{Var}[X]^2 \leq \mathrm{E}[|X-\mu|] \cdot \mathrm{E}[|X-\mu|^3] \leq 2\,\mathrm{Var}[X]\,\mathrm{E}[|X-\mu|^3],$$

where the first inequality is by Cauchy-Schwartz and the second is an application of (E.2). The desired result follows. □

Proof of Theorem E.1. We start by noting that we may assume that $\sigma^2 > C$ for a sufficiently large constant C. Indeed, by Lemma E.3,

$$c\left(\frac{\sum_{j\in[k]} \mathrm{E}[|X_j-\mu_j|^3]}{\sigma^3}\right)^2 \geq \frac{c}{4\sigma^2},$$

so (E.1) is immediate when $\sigma^2 \leq C$ (by choosing c sufficiently large).

Now the proof proceeds, similarly to proofs of the Berry-Esseen theorem and uses simple properties of the Fourier transformation of X. Let f_j be the characteristic function of $X_j - \mu_j$ for $j \in [k]$ and let F be the characteristic function of $X - \mu$. Then

$$F(\lambda) = \prod_{j\in[k]} f_j(\lambda) = \sum_{n\in\mathbb{Z}} \Pr[X=n]\, e^{i(n-\mu)\lambda}\,.$$

For non-zero integers s, it holds that $\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{is\lambda}d\lambda = 0$ whereas $\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{is\lambda}d\lambda = 1$ if $s = 0$. It follows that for any integer $n \in \mathbb{Z}$,

$$\Pr[X=n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} F(\lambda)e^{-i(n-\mu)\lambda}d\lambda\,.$$

In particular, if $\mu + t\sigma$ is an integer, then

$$\Pr[X=\mu+t\sigma] = \frac{1}{2\pi}\int_{-\pi}^{\pi} F(\lambda)e^{-it\sigma\lambda}d\lambda\,.$$

We define $\tau = \frac{\sum_{j\in[k]} \mathrm{E}[|X_j-\mu_j|^3]}{\sigma^3}$ noting that we may assume that $\tau \leq c_0$ for a sufficiently small constant c_0 as otherwise the result is trivial. Split the interval $[-\pi,\pi]$ into three parts, $I_1 = [-\varepsilon,\varepsilon]$, $I_2 = [\varepsilon,\pi]$, and $I_3 = [-\pi,-\varepsilon]$. We will prove that if $\varepsilon = \frac{\sqrt{8\log 1/\tau}}{\sigma}$, then

$$\left|F(\lambda)e^{-it\sigma\lambda}\right| = |F(\lambda)| \leq \tau^2 \quad \text{for all } \lambda \in I_2 \cup I_3 \text{ and,} \tag{E.3}$$

$$\int_{-\varepsilon}^{\varepsilon} F(\lambda)e^{-it\sigma\lambda}d\lambda = \frac{1}{\sqrt{2\pi}\sigma}e^{-t^2/2} + O(\tau^2). \tag{E.4}$$

We note that we may assume that $\varepsilon \le \pi$. Indeed, by Lemma E.3, $\frac{\log 1/\tau}{\sigma^2} \le \frac{\log(8\sigma)}{\sigma}$, and σ is assumed to be sufficiently large. The desired result thus follows immediately from (E.3) and (E.4).

We start by proving (E.3). It is a general fact that the characteristic function f_Y of a random variable Y is Hermitian, i.e., $f_Y(-t) = \overline{f_Y(t)}$. In particular, $|F(\lambda)| = |F(-\lambda)|$, so it is enough to prove that $|F(\lambda)| \le \tau^2$ for $\lambda \in I_2$. As $|F|$ is decreasing on $[0,\pi]$, it in fact suffices to prove that $|F(\varepsilon)| \le \tau^2$. Now another standard fact about the characteristic function f_Y of a random variable Y is that for any n,

$$\left| f_Y(\lambda) - \sum_{j=0}^{n} \frac{(i\lambda)^j}{j!} \operatorname{E}[Y^j] \right| \le \frac{|\lambda|^{n+1} \operatorname{E}[|Y|^{n+1}]}{(n+1)!}. \tag{E.5}$$

By Jensen's inequality, $\sigma_j^2 \le (\operatorname{E}[|X_j - \mu_j|^3])^{2/3}$, so it follows that

$$\varepsilon^2 \sigma_j^2 \le \left(\varepsilon^3 \sum_{j \in [k]} \operatorname{E}\Big[|X_j - \mu_j|^3 \Big] \right)^{2/3} = (\varepsilon^3 \sigma^3 \tau)^{2/3} \le 8 \log(1/\tau) \tau^{2/3} \le 1, \tag{E.6}$$

where the last inequality used that $\tau \le c_0$ for a sufficiently small constant c_0. We may thus apply (E.5) with $n = 2$ to conclude that

$$|f_j(\varepsilon)| \le 1 - \frac{\varepsilon^2}{2}\sigma_j^2 + \operatorname{E}\Big[|X_j - \mu_j|^3 \Big] \frac{\varepsilon^3}{6} \le e^{-\frac{\varepsilon^2}{2}\sigma_j^2 + \operatorname{E}\left[|X_j - \mu_j|^3\right] \frac{\varepsilon^3}{6}}.$$

Thus, for $\lambda \in I_2$,

$$\left| F(\lambda) e^{-it\sigma\lambda} \right| = |F(\lambda)| \le |F(\varepsilon)| \le e^{-\frac{\varepsilon^2}{2}\sigma^2 + \left(\sum_{j \in [k]} \operatorname{E}\left[|X_j - \mu_j|^3\right]\right) \frac{\varepsilon^3}{6}} = e^{-\varepsilon^2 \sigma^2 (1/2 - \sigma\varepsilon\tau/6)}.$$

As $\sigma\varepsilon\tau = \tau\sqrt{8 \log 1/\tau} \le 3/2$, it therefore follows that for $\lambda \in I_2$,

$$|F(\lambda)| \le e^{-\frac{\varepsilon^2\sigma^2}{4}} = \tau^2 \,, \tag{E.7}$$

which proves (E.3).

Turning to (E.4), we again use the Taylor expansion formula to get

$$f_j(\lambda) = 1 - \frac{\lambda^2}{2}\sigma_j^2 + \operatorname{E}\Big[|X_j - \mu_j|^3 \Big] \lambda^3 g_j(\lambda),$$

for some (complex-valued) function $g_j(\lambda)$ with $|g_j(\lambda)| \le 1/6$ for all λ. As in (E.6), for $|\lambda| \le \varepsilon$,

$$\operatorname{E}\Big[|X_j - \mu_j|^3 \Big] |\lambda|^3 \le 1, \tag{E.8}$$

and

$$\lambda^4 \sigma_j^4 \le (|\lambda|^3 \operatorname{E}[|X_j - \mu_j|^3])^{4/3} \le |\lambda|^3 \operatorname{E}[|X_j - \mu_j|^3] \le 1. \tag{E.9}$$

It follows that $|f_j(\lambda)-1| \le 5/6$. Now for $z \in \mathbb{C}$ with $|z| \le 5/6$ it holds that $1+z = \exp(z+O(z^2))$. Also, if $a, b \in \mathbb{C}$ satisfy that $|a|^2 \le |b| \le 1$, then $|a+b|^2 \le 2(|a|^2+|b|^2) \le 4|b|$. Combining these observations with (E.8) and (E.9) we find that

$$f_j(\lambda) = e^{-\frac{\lambda^2}{2}\sigma_j^2 + \mathrm{E}\left[|X_j-\mu_j|^3\right]O(\lambda^3)}.$$

It follows that

$$F(\lambda)e^{-it\sigma\lambda} = e^{-\frac{\lambda^2}{2}\sigma^2 + \left(\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]\right)O(\lambda^3)} e^{-it\sigma\lambda}$$

We then get that

$$\begin{aligned} F(\lambda)e^{-it\sigma\lambda} &= e^{-\frac{\lambda^2}{2}\sigma^2 + \left(\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]\right)O(\lambda^3)} e^{-it\sigma\lambda} \\ &= e^{-\frac{\lambda^2}{2}\sigma^2}\left(1 + \left(\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]\right)O(\lambda^3)\right)e^{-it\sigma\lambda} \end{aligned} \tag{E.10}$$

for $|\lambda| \le \varepsilon$. Now we get that

$$\begin{aligned} &\frac{1}{2\pi}\int_{-\varepsilon}^{\varepsilon} \left| e^{-\frac{\lambda^2}{2}\sigma^2}\left(\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]\right)O(\lambda^3)e^{-it\sigma\lambda}\right| d\lambda \\ &\quad= \frac{1}{\pi}\left(\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]\right)\int_0^{\varepsilon} e^{-\frac{\lambda^2}{2}\sigma^2}O(\lambda^3)d\lambda \\ &\quad= \frac{1}{\pi}\left(\frac{\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]}{\sigma^4}\right)\int_0^{\sqrt{8\log 1/\tau}} e^{-\frac{s^2}{2}}O(s^3)ds \\ &\quad= O\left(\frac{\sum_{j\in[k]} \mathrm{E}\left[|X_j-\mu_j|^3\right]}{\sigma^4}\right) \\ &\quad= O(\tau^2) \end{aligned} \tag{E.11}$$

Here we used the substitution $s = \lambda\sigma$, and the last step uses Lemma E.3. Again using the same substitution we get that

$$\frac{1}{2\pi}\int_{-\varepsilon}^{\varepsilon} e^{-\frac{\lambda^2}{2}\sigma^2}e^{-it\sigma\lambda}d\lambda = \frac{1}{2\pi\sigma}\int_{-\sqrt{8\log 1/\tau}}^{\sqrt{8\log 1/\tau}} e^{-\frac{s^2}{2}}e^{-its}ds$$

Note that for any $u > 0$,

$$\int_u^{\infty} e^{-s^2/2}\,ds \le \frac{1}{u}\int_u^{\infty} s\cdot e^{-s^2/2}\,ds = \frac{1}{u}e^{-u^2/2},$$

so we can bound

$$\frac{1}{2\pi\sigma}\int_{|s|\geq\sqrt{8\log 1/\tau}}\left|e^{-\frac{s^2}{2}}e^{-its}\right|ds \leq \frac{\tau^4}{\pi\sigma\sqrt{8\log 1/\tau}} = O(\tau^2). \tag{E.12}$$

Calculating the Fourier transform of function of $e^{-\frac{s^2}{2}}$ we get that

$$\frac{1}{2\pi\sigma}\int_{-\infty}^{\infty} e^{-\frac{s^2}{2}}e^{-its}ds = \frac{1}{\sqrt{2\pi}\sigma}e^{-t^2/2} \tag{E.13}$$

Combining (E.10), (E.11), (E.12), and (E.13) proves (E.4). This finishes the proof. □

E.3 Point Probabilities in the Tail

As is, Theorem E.1 is only useful when $|t\sigma|$ is not too large. Indeed, for large $|t|$, the term $\frac{1}{\sqrt{2\pi}\sigma}e^{-t^2/2}$ will typically be much smaller than the error term on the right hand side of (E.1). We now show that if our variables satisfy the stronger property of being strongly monotone, we may also obtain precise estimates for the point probabilities in the tail by combining with the trick of exponential tilting.

Recall that we defined a real random variable X to be strongly monotone if $I_X \neq \{0\}$ and X_θ is monotone for each $\theta \in I_X$. Here, $I_X = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X}] < \infty\}$ consisted of those θ for which the moment generating function of X is defined, and X_θ was the exponentially tilted random variable defined by $\Pr[X_\theta = t] = \frac{\Pr[X=t]e^{\theta t}}{\mathrm{E}[e^{\theta X}]}$ for $t \in \mathbb{Z}$.

Many commonly encountered random variables have the property of being strongly monotone:

Lemma E.4. *Let X be Bernoulli, Y be geometric and Z be Poisson. Then X, Y and Z are each strongly monotone.*

Proof. We already saw that the classes of Bernoulli, geometric, and Poisson variables consists of monotone variables. The result follows by calculating the point probabilities of the tilted variables (when they exists) and observing that each class is closed under exponential tilts. □

Now suppose $X = \sum_{j\in[k]} X_j$ is a sum of independent random integer variables and moreover that X is not almost surely equal to a constant. We are interested in estimates for the probability $\Pr[X = t]$ for some $t \in \mathbb{Z}$. Let $I_j = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X_j}] < \infty\}$ and $I = \{\theta \in \mathbb{R} : \mathrm{E}[e^{\theta X}] < \infty\} = \cap_{j\in[k]} I_j$. We note each I_j and I are intervals containing 0. We define[3] $A = \operatorname{ess\,inf} X$ and $B = \operatorname{ess\,sup} X$. Let further $\psi_X : I \to \mathbb{R}$ be the cumulant generating function defined by $\psi_X : \theta \mapsto \log(\mathrm{E}[e^{\theta X}])$. It is well known that ψ_X is strictly convex and infinitely often differentiable for θ lying in the interior of I with $\psi'_X(\theta) = \frac{\mathrm{E}[Xe^{\theta X}]}{\mathrm{E}[e^{\theta X}]}$. For $t \in \mathbb{R}$, we define $g(t) = \sup_{\theta\in I}(\theta t - \psi_X(\theta))$. Now it is a standard

[3]Recall that the essential infimum and supremum of a random variable X are defined by $\operatorname{ess\,inf} X = \sup\{t : \Pr[X < t] = 0\}$ and $\operatorname{ess\,sup} X = \inf\{t : \Pr[X > t] = 0\}$ which are values in $\mathbb{R} \cup \{-\infty, \infty\}$.

fact about the cumulant generating function that if I contains a non-empty open interval (i.e., consists of more than a single point), then $\inf_{\theta\in I}\psi'_X(\theta) = A$ and $\sup_{\theta\in I}\psi'_X(\theta) = B$. If in particular $A < t < B$, there exists a θ_0 in the interior of I with $\psi'_X(\theta_0) = t$. Moreover, this θ_0 is unique since ψ_X is strictly convex.

Now let $(Y_j)_{j\in[k]}$ be independent random variables obtained by tilting each X_j by θ_0 as above. Let further $Y = \sum_{j\in[k]} Y_j$. For $s \in \mathbb{Z}$, we define $A_s = \{z \in \mathbb{Z}^k : z_1 + \cdots + z_k = s\}$. Then for any $t \in \mathbb{Z}$,

$$\Pr[X = t] = \sum_{z\in A_t}\prod_{j\in[k]}\Pr[X_j = z_j] = \frac{\mathrm{E}[e^{\theta_0 X}]}{e^{\theta_0 t}}\sum_{z\in A_t}\prod_{j\in[k]}\Pr[Y_j = z_j] = \frac{\mathrm{E}[e^{\theta_0 X}]}{e^{\theta_0 t}}\Pr[Y = t],$$

so Y is simply the variable obtained by tilting X by θ_0. Moreover, by the choice of θ_0,

$$\mathrm{E}[Y] = \sum_{z\in\mathbb{Z}}\frac{\Pr[X = z]e^{\theta_0 z}z}{\mathrm{E}[e^{\theta_0 X}]} = \frac{\mathrm{E}[Xe^{\theta_0 X}]}{\mathrm{E}[e^{\theta_0 X}]} = \psi'_X(\theta_0) = t.$$

Now the fact that $\mathrm{E}[Y] = t$, suggests using Theorem E.1 to estimate the probability that $\Pr[Y = t]$. Doing so, we immediately obtain the following result.

Theorem E.5. *Assume that X is strongly monotone and not almost surely equal to a constant. Moreover assume that $I \neq \{0\}$. Let t be an integer with with $A < t < B$ and θ be the unique real in the interior of I having $\psi'_X(\theta) = t$. Let Y be the exponential tilt of X by θ. Then $\mathrm{E}[Y] = t$ and*

$$\Pr[X = t] = \frac{\mathrm{E}[e^{\theta X}]}{e^{\theta t}}\left(\frac{1}{\sqrt{2\pi}\sigma_Y} \pm O\left(\frac{\eta_Y^2}{\sigma_Y^6}\right)\right), \tag{E.14}$$

where $\sigma_Y^2 = \mathrm{Var}[Y]$ and $\eta_Y = \sum_{j\in[k]}\mathrm{E}[|Y_j - \mathrm{E}[Y_j]|^3]$.

Remark E.6. We note that if either $A = \operatorname{ess\,inf} X \neq -\infty$ or $B = \operatorname{ess\,sup} X \neq \infty$, then $[0,\infty) \subset I$ or $(-\infty, 0] \subset I$, respectively, and we can therefore always apply the exponential tilt in the lemma. We moreover note that for $t < A$ and $t > B$, it trivially holds that $\Pr[X = t] = 0$ and it is an easy exercise to show that

$$\Pr[X = A] = \prod_{j\in[k]}\Pr[X_j = \operatorname{ess\,inf} X_j], \quad \text{and} \quad \Pr[X = B] = \prod_{j\in[k]}\Pr[X_j = \operatorname{ess\,sup} X_j],$$

whenever $A \neq -\infty$ and $B \neq \infty$. Even though the lemma does not provide estimates for these probabilities, they are therefore usually easy to determine for concrete families of random variables.

To apply Theorem E.5, for $X = \sum_{j\in[k]} X_j$ a concrete sum of strongly monotone random variables, say geometric variables, we would calculate ψ_X and find the unique θ with $\psi'_X(\theta) = t$. We would then determine the tilted random variables $(Y_j)_{j\in[k]}$. Typically Y_j comes from the same family of random variables as X_j, e.g., an exponential tilt of respectively a Bernoulli, geometric, and Poisson variable is again Bernoulli, geometric and Poisson. We would then determine the quantities η_Y and σ_Y^2 and plug into (E.14).

Example E.7. Let $X = \sum_{j\in[k]} X_j$, where $(X_j)_{j\in[k]}$ are independent Bernoulli variables with $\Pr[X_j = 1] = p_j$. We want to estimate $\Pr[X = t]$ for some $0 < t < k - 1$. For this, we define θ, $(Y_j)_{j\in[k]}$ and Y as in Theorem E.5. Then each Y_j is again Bernoulli. If $\Pr[Y_j = 1] = q_j$, then $\mathrm{E}[|Y_j - \mathrm{E}[Y_j]|^3] = q_j(1 - q_j)(q_j^2 + (1 - q_j)^2) \leq \mathrm{Var}[Y_j]$, so that $\eta_Y \leq \sigma_Y^2$. Thus, the bound of (E.14) becomes

$$\Pr[X = t] = \frac{1}{\sqrt{2\pi}\sigma_Y} \frac{\mathrm{E}[e^{\theta X}]}{e^{\theta t}} \left(1 \pm O\left(\frac{1}{\sigma_Y}\right)\right),$$

The bound on the error term can be shown to be asymptotically tight using known results. We in particular note that if $\sigma_Y = \omega(1)$, the bound on $\Pr[X = t]$ is within a factor of $1+o(1)$ of the true value. Consider as a very simple example[4], the case where the Bernoulli variables $(X_j)_{j\in[k]}$ are identically distributed. Then the same holds for the $(Y_j)_{j\in[k]}$, and since $\mathrm{E}[Y] = t$, we must have that $\sigma_Y^2 = t(1 - t/k)$. In particular, the bound is within a factor of $1 + o(1)$ of the true value as long as $\omega(1) < t < k - \omega(1)$.

Example E.8. Let $X = \sum_{j\in[k]} X_j$ be a sum of independent geometric variables such that for some probabilities $(p_j)_{j\in[k]}$ and each $s \in \mathbb{N}_0$, $\Pr[X_j = s] = p_j^s(1 - p_j)$. Let $\mu_j = \mathrm{E}[X_j] = \frac{p_j}{1-p_j}$ for $j \in [k]$ and $\mu = \mathrm{E}[X]$. Assume that $\mu_j = O(1)$ for $j \in [k]$. We want to estimate $\Pr[X = t]$ for some integer $t > 0$ using Theorem E.5, and we define θ, $(Y_j)_{j\in[k]}$ and Y accordingly. For simplicity, we will assume that $t = O(\mathrm{E}[X])$. By Lemma E.4, each Y_j is again geometric, say with $\Pr[Y_j = s] = q_j^s(1 - q_j)$ for $s \in \mathbb{N}_0$. Moreover, since $\mathrm{E}[X_j] = O(1)$ for $j \in [k]$ and $t = O(\mathrm{E}[X])$, it follows that also $\mathrm{E}[Y_j] = O(1)$ for $j \in [k]$. Now simple calculations yields that $\mathrm{Var}[Y_j] = \Theta(\mathrm{E}[Y_j])$ and $\mathrm{E}[|Y_j - \mathrm{E}[Y_j]|^3] = \Theta(\mathrm{E}[Y_j])$. Plugging into (E.14), we thus obtain that

$$\Pr[X = t] = \frac{1}{\sqrt{2\pi}\sigma_Y} \frac{\mathrm{E}[e^{\theta X}]}{e^{\theta t}} \left(1 \pm O\left(\frac{1}{\sqrt{\mu_Y}}\right)\right),$$

where $\mu_Y = \mathrm{E}[Y] = t$. In particular, the bound is within a factor of $1 + o(1)$ of the true value as long as $t = \omega(1)$.

References

[AKT21] Anders Aamand, Jakob Bæk Tejs Knudsen, and Mikkel Thorup. "Load balancing with dynamic set of balls and bins". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 1262–1275.

[Bei] Ahmad Beirami. *$E[|X - \mu|^n] \leq 2E[|X - \mu|^{n+1}]$ for Integer Random Variables*. Mathematics Stack Exchange. URL: https://math.stackexchange.com/q/4102849 (version: 2021-04-16).

[4] This example is to be seen as a proof of concept as one can obtain precise asymptotically tight estimates directly using Stirling's formula.

[BJ59] David Blackwell and J. L. Hodges Jr. "The Probability in the Extreme Tail of a Convolution". In: *The Annals of Mathematical Statistics* 30.4 (1959), pp. 1113–1120.

[DH22] Dmitry Dolgopyat and Yeor Hafouta. "Edgeworth expansions for independent bounded integer valued random variables". In: *Stochastic Processes and their Applications* 152 (2022), pp. 486–532. ISSN: 0304-4149.

[GK54] B. V. Gnedenko and A. N. Kolmogorov. *Limit Distributions for Sums of Independent Random Variables*. Cambridge: Addison-Wesley, 1954.

[IL71] I. A. Ibragimov and I. V. Linnik. *Independent and stationary sequences of random variables*. Groningen: Wolters-Noordhoff., 1971. ISBN: 9001418856.

[Ilt95] Michael Iltis. "Sharp asymptotics of large deviations in $\mathbb{R}^d$". In: *Journal of Theoretical Probability* 8.3 (1995), pp. 501–522.

[McD79] David McDonald. "A Local Limit Theorem for Large Deviations of Sums of Independent, Nonidentically Distributed Random Variables". In: *The Annals of Probability* 7.3 (1979), pp. 526–531. ISSN: 00911798.