Understanding the Moments of Tabulation Hashing via Chaoses

Jakob Bæk Tejs Houen
BARC, Department of Computer Science, University of Copenhagen, Denmark

Mikkel Thorup
BARC, Department of Computer Science, University of Copenhagen, Denmark

Abstract

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 $h_0, \dots, h_{c-1} \colon \Sigma \to \{0, \dots, 2^l - 1\}$, and a key $x = (x_0, \dots, x_{c-1})$ is hashed to $h(x) = h_0(x_0) \oplus \dots \oplus h_{c-1}(x_{c-1})$ 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 $\sum a_{i_0,\dots,i_{c-1}} X_{i_0} \cdot \dots \cdot X_{i_{c-1}}$ 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.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases hashing, concentration bounds, moment bounds

Digital Object Identifier 10.4230/LIPIcs.ICALP.2022.74

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2205.01453`

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

Acknowledgements We thank the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions.

1 Introduction

Hashing is a ubiquitous tool of randomized algorithms which dates all the way back to the 1950s [12]. 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 [27]. The scheme is efficient and easy to implement, and Pătraşcu and Thorup [22] 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., [3, 4]). 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 [22] is that it has strong concentration bounds for hash-based sums (will be defined shortly in Section 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. [2] 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, [21] introduced twisted tabulation strengtening simple tabulation, and [9] 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 [9] 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 [2] for tabulation-permutation. More precisely, the concentration bounds in [2] 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 1.6.

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., [1]. 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 [2, 11, 25] 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 [26] 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 $x_0, \ldots, x_{k-1} \in U$.

[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.

The p-th central moment $\mathrm{E}[(X-\mu)^p]$ of X for a k-independent 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. We shall, however, focus on simple and fast hashing schemes that are not even 4-independent, and yet we will show strong moment bounds.

1.2 Tabulation Hashing

Simple tabulation hashing dates back to 1970 and was first introduced by Zobrist for optimizing chess computers [27]. 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, [22] 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. [7] showed that for a fixed bin the 4th central moment is close to that achieved by truly random hashing. Dahlgaard et al. [10] 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. [9]. 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.

1.3 Relation between Simple Tabulation and Chaoses

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

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

where $(X_i)_{i\in[n]}$ are independent random variables and $(a_{i_0,\ldots,i_{c-1}})_{0\le i_0<\ldots<i_{c-1}<n}$ is a multi-indexed 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 [17, 18], when the variables have logarithmically concave tails [5], and when the variables have logarithmically convex tails [16].

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 Section 1.5.

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 1** (p-norm)**.** *Let $p \ge 1$ 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.

1.4.1 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 [8] and the more general Bennett's inequality [6] 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:

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

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

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

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

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.

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

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

▶ Remark 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 $2 \le p$, hence only case 2 and 3 apply. Similarly, if $e^2\frac{\sigma^2}{M^2} \le 2$ then $p \ge e^2\frac{\sigma^2}{M^2}$ for all $2 \le p$, 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 4.** *There exist universal constants K_1 and K_2 satisfying that for a Poisson distributed random variable, X, with $\lambda = \mathrm{E}[X]$*

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

for all $p \ge 2$.

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

Bennett's inequality shows that we can bound the tail of $\sum_{x \in U} v(x, h(x))$ and Lemma 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 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 $t > 0$

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

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 Equation (4) gives the same tail bound as Bennett's inequality, Equation (3), up to a constant in the exponent.

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

▶ **Theorem 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.

1.4.2 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. [2] 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 $q \in [1, \infty]$.

1.4.2.1 Simple Tabulation Hashing

Our main result for simple tabulation hashing is a version of Theorem 5.

▶ **Theorem 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 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. [2] 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 [2, 22]. 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 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 $l \le m$. 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 $p \ge 2$*

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

where $K_c = L_1(L_2c)^{c-1}$, 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 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 9.** *Let $h\colon \Sigma^c \to [m]$ be a simple tabulation hash function, and $2 \le p \le L_1|\Sigma|\log(m)$, 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*

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

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

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

1.4.2.2 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 10.** *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with $d \ge 1$ 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 $p \ge 2$ then*

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

where $K_c = L_1(L_2c)^c$, 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 $m \le |U|^\gamma$ for some constant $\gamma \ge 1$. If this is the case then we can obtain the following nice tail bound for mixed tabulation hashing by using Markov's inequality.

▶ **Corollary 11.** *Let $h\colon \Sigma^c \to [m]$ be a mixed tabulation function with $d \ge 1$ 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 $m \le |U|^\gamma$ for a value $\gamma \ge 1$ then for all $t \ge 0$*

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

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.

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

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

where we can bound γ_p by

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

So we have that

$$\left\|V_v^{\text{mixed}}\right\|_p \le \Psi_p\left(\left(L_2 c^2 \gamma\right)^c M_v, \left(L_2 c^2\gamma\right)^c \sigma_v^2\right) \ .$$

Now by the same method as in Equation (4), we get the result. ◀

1.4.2.3 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 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 $p \ge 2$

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

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

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

▶ **Theorem 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

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

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

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

1.5 Technical Overview

1.5.1 Fully Random Hashing

1.5.1.1 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

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

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 [20], we get that

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

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

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

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 Equation (8) and that it is stronger than Equation (7).

1.5.1.2 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 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 14 is tight up to a universal constant. Thus a natural question to ask is whether Lemma 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 14 is not tight. It would be nice if there existed a simple modification of Lemma 14 which had a matching lower bound.

1.5.1.3 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.

▶ **Lemma 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 15 is tight up to a constant factor.

[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 $t \geq 0$.

1.5.2 Tabulation Hashing

1.5.2.1 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

$$\begin{aligned}2^{-c}\left\|\sum_{i_0,\dots,i_{c-1}\in[n]} a_{i_0,\dots,i_{c-1}} \prod_{j\in[c]} \varepsilon_{i_j}^{(j)} X_{i_j}^{(j)}\right\|_p &\le \left\|\sum_{i_0,\dots,i_{c-1}\in[n]} a_{i_0,\dots,i_{c-1}} \prod_{j\in[c]} X_{i_j}^{(j)}\right\|_p \\ &\le 2^c \left\|\sum_{i_0,\dots,i_{c-1}\in[n]} a_{i_0,\dots,i_{c-1}} \prod_{j\in[c]} \varepsilon_{i_j}^{(j)} X_{i_j}^{(j)}\right\|_p ,\end{aligned}$$

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 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 $\varepsilon(\alpha_0,\dots,\alpha_{c-1}) = \prod_{i\in[c]} T(i,\alpha_i)$. We then prove that for all $p \ge 2$

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

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. [9] 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 Equation (10) we are able to circumvent this. Thus, just by using symmetrization and the insights of Dahlgaard et al. [9] we obtain the following result.

▶ **Lemma 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 $p \ge 2$*

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

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

1.5.2.2 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

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

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 Equation (9) and we prove the following moment bound, which is strictly stronger than Lemma 16.

▶ **Lemma 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 7 but perhaps a bit surprisingly, we will use Lemma 17 as an important step in the proof of Theorem 7.

1.5.2.3 Sum of squares of simple tabulation hashing

A key element when proving Theorem 7 is bounding the sums of squares

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

This was also one of the main technical challenges for the analysis of Aamand et al. [2]. Instead of analyzing Equation (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.

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

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

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

▶ **Lemma 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 $p \geq 2$*

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

where L is a universal constant.

1.5.2.4 Proving the main result

The proof of Theorem 7 is by induction on c. We will use Theorem 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 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 18. While this sketch is simple, the actual proof is quite involved and technical since one has to be very careful with the estimates.

1.6 Mixed Tabulation Hashing in Context

Our concentration bounds for mixed tabulation hashing are similar to those Aamand et al. [2] 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 1.6.1.

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. [9] 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. [13, 14, 15], and for fast set similarity in large-scale machine learning by Li et al. [19, 23, 24].

Selective full randomness with mixed tabulation

The main result of Dahlgaard et al. [9] 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. [9, 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 19** (Dahlgaard et al. [9, 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 $\mathrm{E}[|Y|] \le |\Sigma|/(1+\Omega(1))$, 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 [9], we also want the size of the set Y to be concentrated around its mean and by Corollary 11, the concentration is essentially as strong as with fully random hashing and it holds for any $d \ge 1$.

In [9] they only proved weaker concentration bounds for the set Y selected in Theorem 19. Based on the concentration bounds for simple tabulation by Pătraşcu and Thorup [22], they proved that if the set Y from 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

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

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

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

[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 (14) is the asymptotic bound we would get with fully-random hashing. Also, while Dahlgaard et al. only proved (13) for the special case of $\mathrm{E}[Y] \in [|\Sigma|/8, 3|\Sigma|/4]$, our (14) is just a special case of Corollary 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 19 is rather subtle.

1.6.1 Mixed Tabulation Hashing Versus Tabulation-Permutation Hashing

As mentioned earlier, our new concentration bounds are similar to those proved by Aamand et al. [2] 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 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. [2] 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. [2] 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 [2, 10] for more details. In particular, [10] 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.

References

1 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. *CoRR*, 2020. arXiv:2004.01156.

2 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, pages 1265–1278, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3357713.3384259.

3 Anders Aamand, Mathias Bæk Tejs Knudsen, and Mikkel Thorup. Power of d choices with simple tabulation. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 5:1–5:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.5.

4 Anders Aamand and Mikkel Thorup. Non-empty bins with simple tabulation hashing. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2498–2512. SIAM, 2019. doi:10.1137/1.9781611975482.153.

5 Radosław Adamczak and Rafał Latała. Tail and moment estimates for chaoses generated by symmetric random variables with logarithmically concave tails. *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, 48(4):1103–1136, 2012. doi:10.1214/11-AIHP441.

6 George Bennett. Probability inequalities for the sum of independent random variables. *Journal of the American Statistical Association*, 57(297):33–45, 1962. doi:10.1080/01621459.1962.10482149.

7 Vladimir Braverman, Kai-Min Chung, Zhenming Liu, Michael Mitzenmacher, and Rafail Ostrovsky. AMS without 4-wise independence on product domains. In Jean-Yves Marion and Thomas Schwentick, editors, *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France*, volume 5 of *LIPIcs*, pages 119–130. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPIcs.STACS.2010.2449.

8 Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23(4):493–507, 1952.

9 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*, pages 1292–1310, 2015. doi:10.1109/FOCS.2015.83.

10 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, pages 6618–6628, USA, 2017. Curran Associates Inc. URL: http://dl.acm.org/citation.cfm?id=3295222.3295407.

11 Martin Dietzfelbinger and Michael Rink. Applications of a splitting trick. In *Proceedings of the 36th ICALP*, pages 354–365, 2009.

12 A. I. Dumey. Indexing for rapid random access memory systems. *Computers and Automation*, 5(12):6–9, 1956.

13 Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985. Announced at FOCS'83.

14 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.

15 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*, pages 683–692, 2013.

16 Konrad Kolesko and Rafał Latała. Moment estimates for chaoses generated by symmetric random variables with logarithmically convex tails. *Statistics & Probability Letters*, 107:210–214, 2015. doi:10.1016/j.spl.2015.08.019.

17 Rafał Latała. Estimates of moments and tails of Gaussian chaoses. *The Annals of Probability*, 34(6):2315–2331, 2006. doi:10.1214/009117906000000421.

18 Joseph Lehec. *Moments of the Gaussian Chaos*, pages 327–340. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-15217-7_13.

19 Ping Li, Art B. Owen, and Cun-Hui Zhang. One permutation hashing. In *Proc. 26th NIPS*, pages 3122–3130, 2012.

20 Krzysztof Oleszkiewicz. On a nonsymmetric version of the Khinchine-Kahane inequality. In *Stochastic inequalities and applications*, pages 157–168. Springer, 2003.

21 Mihai Patrascu and Mikkel Thorup. Twisted tabulation hashing. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 209–228. SIAM, 2013. doi:10.1137/1.9781611973105.16.

22 Mihai Pătraşcu and Mikkel Thorup. The power of simple tabulation hashing. *J. ACM*, 59(3), June 2012. doi:10.1145/2220357.2220361.

23 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)*, pages 557–565, 2014.

24 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*, pages 732–741, 2014.

25 Mikkel Thorup. Simple tabulation, fast expanders, double tabulation, and high independence. In *54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 90–99, 2013.

26 Mark N. Wegman and Larry Carter. New classes and applications of hash functions. *Journal of Computer and System Sciences*, 22(3):265–279, 1981. Announced at FOCS'79.

27 Albert Lindsey Zobrist. A new hashing method with application for game playing. Technical Report 88, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, 1970.