# Why Simple Hash Functions Work: Exploiting the Entropy in a Data Stream

Michael Mitzenmacher* Salil Vadhan†

School of Engineering & Applied Sciences
Harvard University
Cambridge, MA 02138
{michaelm,salil}@eecs.harvard.edu
http://eecs.harvard.edu/~{michaelm,salil}

July 9, 2007

**Abstract**

Hashing is fundamental to many algorithms and data structures widely used in practice. For theoretical analysis of hashing, there have been two main approaches. First, one can assume that the hash function is truly random, mapping each data item independently and uniformly to the range. This idealized model is unrealistic because a truly random hash function requires an exponential number of bits to describe. Alternatively, one can provide rigorous bounds on performance when explicit families of hash functions are used, such as 2-universal or $O(1)$-wise independent families. For such families, performance guarantees are often noticeably weaker than for ideal hashing.

In practice, however, it is commonly observed that weak hash functions, including 2-universal hash functions, perform as predicted by the idealized analysis for truly random hash functions. In this paper, we try to explain this phenomenon. We demonstrate that the strong performance of universal hash functions in practice can arise naturally from a combination of the randomness of the hash function and the data. Specifically, following the large body of literature on random sources and randomness extraction, we model the data as coming from a "block source," whereby each new data item has some "entropy" given the previous ones. As long as the (Renyi) entropy per data item is sufficiently large, we note that the resulting behavior when choosing a hash function from a 2-universal family is essentially the same as for a truly random hash function. We describe results for several sample applications, including linear probing, balanced allocations, and Bloom filters.

**Keywords:** Hashing, Pairwise Independence, Bloom Filters, Linear Probing, Balanced Allocations, Randomness Extractors.

*Supported by NSF grant CCF-0634923 and grants from Yahoo! Inc. and Cisco, Inc.
†Supported by ONR grant N00014-04-1-0478, NSF grant CCF-0133096, and US-Israel BSF grant 2002246.

# Contents

# 1 Introduction

Hashing is at the core of many fundamental algorithms and data structures, including all varieties of hash tables [Knu], Bloom filters and their many variants [BM2], summary algorithms for data streams [Mut], and many others. Traditionally, applications of hashing are analyzed as if the hash function is a truly random function (a.k.a. "random oracle") mapping each data item independently and uniformly to the range of the hash function. However, this idealized model is unrealistic, because a truly random function mapping $\{0,1\}^n$ to $\{0,1\}^m$ requires an exponential number of bits to describe.

For this reason, a line of theoretical work, starting with the seminal paper of Carter and Wegman [CW] on universal hashing, has sought to provide rigorous bounds on performance when explicit families of hash functions are used, e.g. ones whose description and computational complexity are polynomial in $n$ and $m$. While many beautiful results of this type have been obtained, they are not always as strong as we would like. In some cases, the types of hash functions analyzed can be implemented very efficiently (e.g. universal or $O(1)$-wise independent), but the performance guarantees are noticeably weaker than for ideal hashing. (A motivating recent example is the analysis of linear probing under 5-wise independence [PPR].) In other cases, the performance guarantees are (essentially) optimal, but the hash functions are more complex and expensive (e.g. with a super-linear time or space requirement). For example, if at most $T$ items are going to be hashed, then a $T$-wise independent hash function will have precisely the same behavior as an ideal hash function. But a $T$-wise independent hash function mapping to $\{0,1\}^m$ requires at least $T \cdot m$ bits to represent, which is often too large. For some applications, it has been shown that less independence, such as $O(\log T)$-wise independence suffices, e.g. [SS, PR], but such functions are still substantially less efficient than 2-universal hash functions. A series of works [Sie, OP, DW] have improved the time complexity of (almost) $T$-wise independence to a *constant* number of word operations, but the space complexity necessarily remains at least $T \cdot m$.

In practice, however, the performance of standard universal hashing seems to match what is predicted for ideal hashing. This phenomenon was experimentally observed long ago in the setting of Bloom filters [Ram2]; other reported examples include [BM1, DKSL, PR, Ram1, RFB]. Thus, it does not seem truly necessary to use the more complex hash functions for which this kind of performance can be proven. We view this as a significant gap between the theory and practice of hashing.

In this paper, we aim to bridge this gap. Specifically, we suggest that it is due to the use of worst-case analysis. Indeed, in some cases, it can be proven that there exist sequences of data items for which universal hashing does not provide optimal performance. But these bad sequences may be pathological cases that are unlikely to arise in practice. That is, the strong performance of universal hash functions in practice may arise from a *combination* of the randomness of the hash function and the data.

Of course, doing an average-case analysis, whereby each data item is independently and uniformly distributed in $\{0,1\}^n$ is also very unrealistic (not to mention that it trivializes many applications). Here we propose that an intermediate model, previously studied in the literature on randomness extraction [CG], may also be an appropriate data model for hashing applications. Under the assumption that the data fits this model, we show that relatively weak hash functions achieve essentially the same performance as ideal hash functions.

**Our Model.** We model the data as coming from a random source in which the data items can be far from uniform and have arbitrary correlations, provided that each (new) data item is sufficiently unpredictable given the previous items. This is formalized by Chor and Goldreich's notion of a *block source* [CG],[1] where we require that the $i$'th item $X_i$ has at least some $k$ bits of "entropy" conditioned on the previous items $X_1, \ldots, X_{i-1}$. There are various choices for the entropy measure that can be used here; Chor and Goldreich use *min-entropy*, but most of our results hold even for the less stringent measure of *Renyi entropy*.

Our work is very much in the same spirit as previous works that have examined intermediate models between worst-case and average-case analysis of algorithms for other kinds of problems. Examples include the semi-random graph model of Blum and Spencer [BS], and the smoothed analysis of Spielman and Teng [ST]. Interestingly, Blum and Spencer's semi-random graph models are based on Santha and Vazirani's model of semi-random sources [SV], which in turn were the precursor to the Chor–Goldreich model of block sources [CG]. Chor and Goldreich suggest using block sources as an input model for communication complexity, but surprisingly it seems that no one has considered them as an input model for hashing applications.

**Our Results.** Our first observation is that standard results in the literature on randomness extractors already imply that universal hashing performs nearly as well ideal hashing, provided the data items have enough entropy [BBR, ILL, CG, Zuc]. Specifically, if we have $T$ data items coming from a block source $(X_1, \ldots, X_T)$ where each data item has (Renyi) entropy at least $m + 2\log(T/\varepsilon)$ and $H$ is a random universal hash function mapping to $\{0,1\}^m$, then $(H(X_1), \ldots, H(X_T))$ has statistical difference at most $\varepsilon$ from $T$ uniform and independent elements of $\{0,1\}^m$. Thus, any event that would occur with some probability $p$ under ideal hashing now occurs with probability $p \pm \varepsilon$. This allows us to automatically translate existing results for ideal hashing into results for universal hashing in our model.

In our remaining results, we focus on reducing the amount of entropy required from the data items. Assuming our hash function has a description size $o(mt)$, then we must have $(1 - o(1))m$ bits of entropy per item for the hashing to "behave like" ideal hashing (because the entropy of $(H(X_1), \ldots, H(X_T))$ is at most the sum of the entropies of $H$ and the $X_i$'s). The standard analysis mentioned above requires an additional $2\log(T/\varepsilon)$ bits of entropy per block. In the randomness extraction literature, the additional entropy required is typically not significant because $\log(T/\varepsilon)$ is much smaller than $m$. However, it can be significant in our applications. For example, a typical setting is hashing $T = \Theta(M)$ items into $2^m = M$ bins. Here $m + 2\log(T/\varepsilon) \geq 3m - O(1)$ and we require 3 times more entropy than the lower bound of $(1 - o(1))m$.

We use a variety of general techniques to reduce the entropy required. These include switching from statistical difference (equivalently, $\ell_1$ distance) to Renyi entropy (equivalently, $\ell_2$ distance or collision probability) throughout the analysis and decoupling the probability that a hash function is "good" from the uniformity of the hashed values $h(X_i)$. We can reduce the entropy required even further if we use 4-wise independent hash functions, which also have very fast implementations [TZ].

**Applications.** We illustrate our approach with several specific applications. Here we informally summarize the results; definitions and discussions appear in Sections 3 and 4.

[1] Chor and Goldreich called these *probability-bounded sources,* but the term *block source* has become more common in the literature.

The classic analysis of Knuth [Knu] gives a tight bound for the insertion time for a hash table with *linear probing* in terms of the 'load' of the table (the number of items divided by the size of the table), under the assumption of using an idealized, truly random hash function. Resolving a longstanding open problem, Pagh, Pagh, and Ruzic [PPR] showed that with pairwise independence it is not possible to bound the insertion time in terms of the load alone (for worst-case data), but achieve such a bound for 5-wise independent hashing. However, their bound for 5-wise independent hash functions is polynomially larger than the bound for ideal hashing. We show that 2-universal hashing actually achieves the same performance as ideal hashing asymptotically, provided that the data comes from a block source with roughly $4 \log M$ bits of (Renyi) entropy per item, where $M$ is the size of the hash table. For 4-wise independent hashing, we only need roughly $3 \log M$ bits of entropy per item.

With the *balanced allocation* paradigm [ABKU], it is known that when $T$ items are hashed to $T$ buckets, with each item being sequentially placed in the least loaded of $d$ choices (e.g. $d = 2$), the maximum load is $\log\log T/\log d + O(1)$ with high probability. We show the same results holds when the hash function is chosen from a 2-universal hash family, provided the data items come from a block source with roughly $(d+2)\log T$ bits of entropy per data item. For 4-wise independence, the entropy requirement is reduced to roughly $(d+1)\log T$.

Bloom filters [Blo] are data structures for approximately storing sets in which membership tests can result in false positives with some bounded probability. We begin by showing that there is a constant gap in the false positive probability for worst-case data when $O(1)$-wise independent hash functions are used instead of truly random hash functions. On the other hand, we show if the data comes from a block source with roughly $4 \log M$ bits of (Renyi) entropy per item, where $M$ is the size of the hash table, then the false positive probability with 2-universal hashing asymptotically matches that of ideal hashing. For 4-wise independent hashing, we only need roughly $3 \log M$ bits of entropy per item.

## 2 Preliminaries

**Notation.** $[N]$ denotes the set $\{1, \ldots, N\}$. All logs are base 2. For a random variable $X$ and an event $E$, $X|_E$ denotes $X$ conditioned on $E$. The *support* of $X$ is $\mathrm{Supp}(X) = \{x : \Pr[X = x] > 0\}$. For a set $S$, $U_S$ denotes a random variable uniformly distributed on $S$.

**Hashing.** Let $\mathcal{H}$ be a multiset of hash functions $h : [N] \to [M]$ and let $H$ be uniformly distributed over $\mathcal{H}$. We say that $\mathcal{H}$ is a *truly random family* if $\mathcal{H}$ is the set all functions mapping $[N]$ to $[M]$, i.e. the $N$ random variables $\{H(x)\}_{x\in[N]}$ are independent and uniformly distributed over $[M]$. For $s \in \mathbb{N}$, $\mathcal{H}$ is *$s$-wise independent* (a.k.a. *strongly $s$-universal* [WC]) if for every sequence of distinct elements $x_1, \ldots, x_s \in [N]$, the random variables $H(x_1), \ldots, H(x_s)$ are independent and uniformly distributed over $[M]$. $\mathcal{H}$ is *$s$-universal* if for every sequence of distinct elements $x_1, \ldots, x_s \in [N]$, we have $\Pr[H(x_1) = \cdots = H(x_s)] \le 1/M^s$. For a hash family $\mathcal{H}$ mapping $[N] \to [M]$ and $k \in \mathbb{N}$, we define $\mathcal{H}^k$ to be the family mapping $[N] \to [M]^k$ consisting of the functions of the form $h(x) = (h_1(x), \ldots, h_k(x))$, where each $h_i \in \mathcal{H}$. Observe that if $\mathcal{H}$ is $s$-wise independent (resp., $s$-universal), then so is $\mathcal{H}^k$. However, description and computation time for functions in $\mathcal{H}^k$ are $k$ times larger than for $\mathcal{H}$.

# 3 Hashing Worst-Case Data

In this section, we describe the three main hashing applications we study in this paper — linear probing, balanced allocations, and Bloom filters — and describe mostly known results about what can be achieved for worst-case data. Here and throughout the paper, where appropriate we leave proofs to the appendix because of space.

## 3.1 Linear Probing

A hash table using linear probing stores a sequence $\overline{x} = (x_1, \ldots, x_T)$ of data items from $[N]$ using $M$ memory locations. Given a hash function $h : [N] \to [M]$, we place the elements $x_1, \ldots, x_T$ sequentially as follows. The element $x_i$ first attempts placement at $h(x_i)$, and if this location is already filled, locations $h(x_i) + 1 \bmod M$, $h(x_i) + 2 \bmod M$, and so on are tried until an empty location is found. The ratio $\alpha = T/M$ is referred to as the *load* of the table. The efficiency of linear probing is measured according to the insertion time for a new data item. (Other measures, such as the average time to search for items already in the table, are also often studied, and our results can be generalized to these measures as well.)

**Definition 3.1** *Given $h : [N] \to [M]$, a sequence $\overline{x} = (x_1, \ldots, x_T)$ of data items from $[N]$ stored via linear probing using $h$, we define the* insertion time $\mathrm{Time}_{\mathrm{LP}}(\overline{x}, h)$ *to be the value of $j$ such that $x_T$ is placed at location $h(x_i) + (j - 1) \bmod M$.*

It is well known that with ideal hashing, the expected time can be bounded quite tightly as a function of the load [Knu].

**Theorem 3.2 ([Knu])** *Let $H$ be a truly random hash function mapping $[N]$ to $[M]$. For every sequence $\overline{x} \in [N]^T$, we have $\mathrm{E}[\mathrm{Time}_{\mathrm{LP}}(\overline{x}, H)] \leq 1/(1 - \alpha)^2$, where $\alpha = T/M$ is the load.*

Resolving a longstanding open problem, Pagh, Pagh, and Ruzic [PPR] recently showed that the expected lookup time could be bounded in terms of $\alpha$ using only $O(1)$-wise independence. Specifically, with 5-wise independence, the expected time for an insertion is $O\left(\frac{1}{(1-\alpha)^3}\right)$ for any sequence $\overline{x}$. On the other hand, in [PPR] it is also shown that there are examples of families of sets $\overline{x}$ and pairwise independent hash functions such that the expected time for a lookup is logarithmic in $T$. In contrast, our work demonstrates that pairwise independent hash functions yield expected lookup times that are asymptotically the same as under the idealized analysis, assuming there is sufficient entropy in the data items themselves.

## 3.2 Balanced Allocations.

A hash table using the *balanced allocation paradigm* [ABKU] with $d \in \mathbb{N}$ choices stores a sequence $\overline{x} = (x_1, \ldots, x_T) \in [N]^T$ in an array of $M$ buckets. Let $h$ be a hash function mapping $[N]$ to $[M]^d \cong [M^d]$, where we denote the components of $h(x)$ as $(h_1(x), \ldots, h_d(x))$. We place the elements sequentially by putting $x_i$ in the least loaded of the $d$ buckets $h_1(x_i), \ldots, h_d(x_i)$ at time $i$ (breaking ties arbitrarily), where the *load* of a bucket at time $i$ is the number elements from $x_1, \ldots, x_{i-1}$ placed in it.

**Definition 3.3** *Given $h : [N] \to [M]^d$, a sequence $\overline{x} = (x_1, \ldots, x_T)$ of data items from $[N]$ stored via the balanced allocation paradigm (with d choices) using h, we define the* maximum load $\mathrm{MaxLoad}_{\mathrm{BA}}(\overline{x}, h)$ *to be the maximum load of among the buckets at time $T+1$.*

In the case that the number $T$ of items is the same as the number $M$ of buckets and we do balanced allocation with $d = 1$ choice (i.e. standard load balancing), it is known that the maximum load is $\Theta(\log T/\log\log T)$ with high probability. Remarkably, when the number of choices $d$ is two or larger, the maximum load drops to be double-logarithmic.

**Theorem 3.4 ([ABKU, Vöc])** *For every $d \geq 2$ and $\gamma > 0$, there is a constant c such the following holds. Let H be a truly random hash function mapping $[N]$ to $[T]^d$. For every sequence $\overline{x} \in [N]^T$ of distinct data items, we have*

$$\Pr[\mathrm{MaxLoad}_{\mathrm{BA}}(\overline{x}, H) > \frac{\log\log T}{\log d} + c] \leq \frac{1}{T^\gamma}.$$

There are other variations on this scheme, including the asymmetric version do to Vöcking [Vöc] and cuckoo hashing [PR]; we choose to study the original setting for simplicity.

The asymmetric scheme has been recently studied under explicit functions [Woe], similar to those of [DW]. At this point, we know of no non-trivial upper or lower bounds for the balanced allocation paradigm using families of hash functions with constant independence, although performance has been tested empirically [BM1]. Such bounds have been a long-standing open question in this area.

### 3.3 Bloom Filters

A *Bloom filter* [Blo] represents a set $\overline{x} = \{x_1, \ldots, x_T\}$ where each $x_i \in [N]$ using an array of $M$ bits and $\ell$ hash functions. For our purposes, it will be somewhat easier to work with a *segmented Bloom filter*, where the $M$ bits are partitioned into $\ell$ disjoint subarrays of size $M/\ell$, with one subarray for each hash function. We assume that $M/\ell$ is an integer. (This splitting does not substantially change the results from the standard approach of having all hash functions map into a single array of size $M$.) As in the previous section, we denote the components of a hash function $h : [N] \to [M/\ell]^\ell \cong [(M/\ell)^\ell]$, as providing $\ell$ hash values $h(x) = (h_1(x), \ldots, h_\ell(x)) \in [M/\ell]^\ell$ in the natural way. The Bloom filter is initialized by setting all bits to 0, and then setting the $h_i(x_j)$'th bit to be 1 in the $i$'th subarray for all $i \in [\ell]$ and $j \in [T]$. Given an element $y$, one tests for membership in $\overline{x}$ by accepting if the $h_i(y)$'th bit is 1 in the $i$'th subarray for all $i \in [\ell]$, and rejecting otherwise. Clearly, if $y \in \overline{x}$, then the algorithm will always accept. However, the algorithm may err if $y \notin \overline{x}$.

**Definition 3.5** *Given $h : [N] \to [M/\ell]^\ell$ (where $\ell$ divides $M$), a sequence $\overline{x} = (x_1, \ldots, x_T)$ of data items from $[N]$ stored in an $\ell$-segment Bloom filter using h, and an additional element $y \in [M]$, we define the* false positive predicate $\mathrm{FalsePos}_{\mathrm{BF}}(h, \overline{x}, y)$ *to be 1 if $y \notin \overline{x}$ and the membership test accepts. That is, if $y \notin \overline{x}$ yet $h_i(y) \in h_i(\overline{x}) \stackrel{\text{def}}{=} \{h_i(x_j) : j = 1, \ldots, T\}$ for all $i = 1, \ldots, \ell$.*

For truly random families of hash functions, it is easy to compute the false positive probability.

**Theorem 3.6 ([Blo])** *Let $H$ be a truly random hash family mapping $[N]$ to $[M/\ell]^\ell$ (where $\ell$ divides $M$). For every sequence $\overline{x} \in [N]^T$ of data items and every $y \notin \overline{x}$, we have*

$$\Pr[\mathrm{FalsePos}_{\mathrm{BF}}(H, \overline{x}, y) = 1] = \left(1 - \left(1 - \frac{\ell}{M}\right)^T\right)^\ell \approx \left(1 - e^{-\ell T/M}\right)^\ell .$$

In the typical case that $M = \Theta(T)$, the asymptotically optimal number of hash functions is $\ell = (M/T) \cdot \ln 2$, and the false positive probability is approximately $2^{-\ell}$.

Below we will describe the kinds of results that can be proven about Bloom filters on worst-case data using $O(1)$-wise independence. But the following more mild reduction in randomness, using 2 truly random hash functions instead of $\ell$, will be useful to use later.

**Theorem 3.7 ([KM])** *Let $H = (H_1, H_2)$ be a truly random hash function mapping $[N]$ to $[M/\ell]^2$, where $M/\ell$ is a prime integer. Define $H' = (H'_1, \ldots, H'_\ell) : [N] \to [M/\ell]^\ell$ by $H'_i(w) = H_1(w) + (i - 1)H_2(w) \bmod M/\ell$. Then for every sequence $\overline{x} \in [N]^T$ of $T$ data items and every $y \notin \overline{x}$, we have*

$$\Pr[\mathrm{FalsePos}_{\mathrm{BF}}(H', \overline{x}, y) = 1] \leq \left(1 - \left(1 - \frac{\ell}{M}\right)^T\right)^\ell + O(1/M).$$

The restriction to prime integers $M/\ell$ is not strictly necessary in general; for more complete statements of when 2 truly random hash functions suffice, see [KM].

We now turn to the worst-case performance of Bloom filters under $O(1)$-wise independence. It is folklore that 2-universal hash functions can be used with a constant-factor loss in space efficiency. Indeed, a union bound shows that $\Pr[h_i(y) \in h_i(\overline{x})]$ is at most $T \cdot (\ell/M)$, compared to $1-(1-\ell/M)^T$ in the case of truly random hash functions. This can be generalized to $s$-wise independent families using the following inclusion-exclusion formula.

**Lemma 3.8** *Let $H : [N] \to [M/\ell]$ be a hash function chosen at random from a family $\mathcal{H}$ (where $\ell | M$). For every sequence $\overline{x} \in [N]^T$, every $y \notin \overline{x}$, and every even $s \leq T$, we have*

$$\begin{aligned}
\Pr[H(y) \in H(\overline{x})] &= \sum_{j=1}^{T} (-1)^{j+1} \sum_{U \subseteq T, |U|=j} \Pr[H(y) = H(w)\ \forall w \in U]. \\
&\leq \sum_{j=1}^{s-1} (-1)^{j+1} \sum_{U \subseteq T, |U|=j} \Pr[H(y) = H(w)\ \forall w \in U]
\end{aligned}$$

If $H$ is an $s$-universal hash family, then the first $s-1$ terms of the outer sum above are the same as for a truly random function (namely $(-1)^{j+1} \cdot \binom{T}{j} (\ell/M)^j$). This gives the following bound.

**Proposition 3.9** *Let $s$ be an even constant. Let $\mathcal{H}$ be an $s$-universal family mapping $[N]$ to $[M/\ell]$ (where $\ell$ divides $M$), and let $H = (H_1, \ldots, H_\ell)$ be a random hash function from $\mathcal{H}^\ell$. For every sequence $\overline{x} \in [N]^T$ of $T \leq M/\ell$ data items and every $y \notin \overline{x}$, we have*

$$\Pr[\mathrm{FalsePos}_{\mathrm{BF}}(H, \overline{x}, y) = 1] \leq \left(1 - \left(1 - \frac{\ell}{M}\right)^T + O\left(\frac{\ell T}{M}\right)^s\right)^\ell .$$

**Proof:** By Lemma 3.8, for each $i = 1, \ldots, \ell$, we have:

$$\begin{aligned}
\Pr[H_i(y) \in H_i(\overline{x})] &\leq -\sum_{j=1}^{s-1}(-1)^j \sum_{U \subseteq T, |U|=j} \Pr[H_i(y) = H_i(w)\ \forall w \in U] \\
&= -\sum_{j=1}^{s-1}(-1)^j \cdot \binom{T}{j}\left(\frac{\ell}{M}\right)^j \qquad \text{(by } s\text{-universality)} \\
&= -\left[\left(1-\frac{\ell}{M}\right)^T - 1 - \sum_{j=s}^{T}(-1)^j \cdot \binom{T}{j}\left(\frac{\ell}{M}\right)^j\right] \\
&\leq 1-\left(1-\frac{\ell}{M}\right)^T + O\left(\frac{\ell T}{M}\right)^s
\end{aligned}$$

Thus,

$$\Pr[\mathrm{FalsePos}_{\mathrm{BF}}(H, \overline{x}, y) = 1] = \Pr[H_i(y) \in H_i(\overline{x})\ \forall i] \leq \left(1-\left(1-\frac{\ell}{M}\right)^T + O\left(\frac{\ell T}{M}\right)^s\right)^\ell .$$

■

Notice that in the common case that $\ell = \Theta(1)$ and $M = \Theta(T)$, so that the false positive probability is constant, the above bound differs from the one for ideal hashing by an amount that shrinks rapidly with $s$. However, when $s$ is constant, the difference remains an additive constant. Another way of interpreting this is that to obtain the guarantee on the false positive probability using $O(1)$-wise independence, one must pay a constant factor in the space for the Bloom filter. The following proposition shows that this loss is necessary if we use only $O(1)$-wise independence.

**Proposition 3.10** *Let $s$ be an even constant. For all $N, M, \ell, T \in \mathbb{N}$ such that $M/\ell$ is a prime power and $T < \min\{M/\ell, N\}$, there exists an $(s+1)$-wise independent family of hash functions $\mathcal{H}$ mapping $[N]$ to $[M/\ell]$ a sequence $\overline{x} \in [N]^T$ of data items, and a $y \in [N] \setminus \overline{x}$, such that if $H = (H_1, \ldots, H_\ell)$ is a random hash function from $\mathcal{H}^\ell$, we have*

$$\Pr[\mathrm{FalsePos}_{\mathrm{BF}}(H, \overline{x}, y) = 1] \geq \left(1-\left(1-\frac{\ell}{M}\right)^T - \Omega\left(\frac{\ell T}{M}\right)^s\right)^\ell .$$

**Proof:** Let $q = M/\ell$, and let $\mathbb{F}$ be the finite field of size $q$. Associate the elements of $[M/\ell]$ with elements of $\mathbb{F}$, and similarly for the first $M/\ell$ elements of $[N]$. Let $\mathcal{H}_1$ consist of all polynomials $f : \mathbb{F} \to \mathbb{F}$ of degree at most $s-1$; this is an $(s+1)$-wise independent family. Let $\mathcal{H}_2$ consist of any $(s+1)$-wise independent family mapping $[N] \setminus \mathbb{F}$ to $\mathbb{F}$. For a function $f \in \mathcal{H}_1$ and $g \in \mathcal{H}_2$, define $h = f \cup g : [N] \to [M/\ell]$ by $h(x) = f(x)$ if $x \in \mathbb{F}$ and $h(x) = g(x)$ if $x \notin \mathbb{F}$. Let $\mathcal{H}$ be the multiset of all such functions $f \cup g$. We let $\overline{x}$ be an arbitrary sequence of $T-1$ distinct elements of $\mathbb{F}$, and $y$ any other element of $\mathbb{F}$.

Again we compute the false positive probability using Lemma 3.8. The first $s$ terms can be computed exactly as before, using $(s+1)$-universality. For the terms beyond $s$, we observe that when $|U| \geq s$, it is the case that $h_i(y) = h_i(w)$ for all $w \in U$ if and only if $h_i = f \cup g$ for a *constant* polynomial $f$. The reason is that no nonconstant polynomial of degree at most $s$ can take on the same value more than $s$ times. The probability that a random polynomial of degree at most $s$ is a constant polynomial is $1/q^s = (\ell/M)^s$.

$$
\begin{aligned}
\Pr[H_i(y) \in H_i(\overline{x})] &= \sum_{j=1}^{T} (-1)^{j+1} \sum_{U \subseteq T, |U|=j} \Pr[H_i(y) = H_i(w)\ \forall w \in U] \\
&= \left[\sum_{j=1}^{s-1} (-1)^{j+1} \cdot \binom{T}{j} \left(\frac{\ell}{M}\right)^j\right] + \left[\sum_{j=s}^{T} (-1)^{j+1} \cdot \binom{T}{j} \left(\frac{\ell}{M}\right)^s\right] \\
&= \left[1 - \left(1 - \frac{\ell}{M}\right)^T + \sum_{j=s}^{T} (-1)^j \cdot \binom{T}{j} \left(\frac{\ell}{M}\right)^j\right] + \left[\sum_{j=0}^{s-1} (-1)^j \cdot \binom{T}{j} \left(\frac{\ell}{M}\right)^s\right] \\
&\geq \left[1 - \left(1 - \frac{\ell}{M}\right)^T + \Omega\left(\frac{\ell T}{M}\right)^s\right] - O\left(T^{s-1} \cdot \left(\frac{\ell}{M}\right)^s\right) \\
&= 1 - \left(1 - \frac{\ell}{M}\right)^T + \Omega\left(\frac{\ell T}{M}\right)^s
\end{aligned}
$$

Again, to bound the false positive probability, we simply raise the above to the $\ell$'th power. ■

# 4 Hashing Block Sources

## 4.1 Random Sources

We view our data items as being random variables distributed over a finite set of size $N$, which we identify with $[N]$. We use the following quantities to measure the amount of randomness in a data item. For a random variable $X$, the *max probability* of $X$ is $\mathrm{mp}(X) = \max_x \Pr[X = x]$. The *collision probability* of $X$ is $\mathrm{cp}(X) = \sum_x \Pr[X = x]^2$. Measuring these quantities is equivalent to measuring the *min-entropy* $\mathrm{H}_\infty(X) = \min_x \log(1/\Pr[X = x]) = \log(1/\mathrm{mp}(X))$ and the *Renyi entropy* $\mathrm{H}_2(X) = \log(1/\mathrm{E}_{x \leftarrow X}[\Pr[X = x]]) = \log(1/\mathrm{cp}(X))$. If $X$ is supported on a set of size $K$, then $\mathrm{mp}(X) \geq \mathrm{cp}(X) \geq 1/K$ (i.e. $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq \log K$), with equality iff $X$ is uniform on its support. It also holds that $\mathrm{mp}(X) \leq \mathrm{cp}(X)^{1/2}$ (i.e. $\mathrm{H}_\infty(X) \geq \mathrm{H}_2(X)/2$), so min-entropy and Renyi entropy are always within a factor of 2 of each other.

We model a sequence of data items as a sequence $(X_1, \ldots, X_T)$ of correlated random variables where each item is guaranteed to have some entropy even conditioned on the previous items.

**Definition 4.1** *A sequence of random variables $(X_1, \ldots, X_T)$ taking values in $[N]^T$ is a* block source with collision probability $p$ per block *(respectively,* max probability $p$ per block*) if for every $i \in [T]$ and every $(x_1, \ldots, x_{i-1}) \in \mathrm{Supp}(X_1, \ldots, X_{i-1})$, we have $\mathrm{cp}(X_i|_{X_1=x_1,\ldots,X_{i-1}=x_{i-1}}) \leq p$ (respectively, $\mathrm{mp}(X_i|_{X_1=x_1,\ldots,X_{i-1}=x_{i-1}}) \leq p$*

When *max probability* is used as the measure of entropy, then this is precisely the model of sources suggested by Chor and Goldreich [CG] in the literature on randomness extractors. We will mainly use the *collision probability* formulation as the entropy measure, since it makes our results more general.

A useful fact is that collision/max probability of the entire sequence is at most the product of the collision/max probabilities contributed by the blocks.

**Lemma 4.2** *If a sequence of random variables* $Y = (Y_1, \ldots, Y_T)$ *taking values in* $[N]^T$ *has collision probability* $1/N + 1/K$ *per block, then* $\mathrm{cp}(Y) \leq (1/N + 1/K)^T$.

**Proof:** By induction on $T$. The base case $T = 0$ follows by definition. For $T > 0$, let $Y' = (Y'_1, \ldots, Y'_T)$ be an iid copy of $Y$. Let $Z$ be the distribution of $(Y_1, \ldots, Y_{T-1})$ conditioned on $(Y_1, \ldots, Y_{T-1}) = (Y'_1, \ldots, Y'_{T-1})$.

$$\begin{aligned}
\mathrm{cp}(Y) &= \Pr[Y = Y'] \\
&= \Pr[(Y_1, \ldots, Y_{T-1}) = (Y'_1, \ldots, Y'_{T-1})] \cdot \Pr[Y_T = Y'_T | (Y_1, \ldots, Y_{T-1}) = (Y'_1, \ldots, Y'_{T-1})] \\
&= \mathrm{cp}(Y_1, \ldots, Y_{T-1}) \cdot \mathop{\mathrm{E}}_{(y_1,\ldots,y_{T-1}) \leftarrow Z} [\Pr[Y_T = Y'_T | Y_1 = Y'_1 = y_1, \ldots, Y_{T-1} = Y'_{T-1} = y_{T-1}]] \\
&= \mathrm{cp}(Y_1, \ldots, Y_{T-1}) \cdot \mathop{\mathrm{E}}_{(y_1,\ldots,y_{T-1}) \leftarrow Z} [\mathrm{cp}(Y_T | Y_1 = y_1, \ldots, Y_{T-1} = y_{T-1})] \\
&\leq (1/N + 1/K)^{T-1} \cdot (1/N + 1/K).
\end{aligned}$$

■

Another useful fact is that we can bound the collision probability of a random variable in terms of conditioned versions of that variable.

**Lemma 4.3** *Let* $(X, Y)$ *be jointly distributed random variables. Then* $\mathrm{cp}(X) \leq \max_{y \in \mathrm{Supp}(Y)} \mathrm{cp}(X | Y = y)$.

**Proof:** Then

$$\begin{aligned}
\mathrm{cp}(X) &= \sum_x \Pr[X = x]^2 \\
&= \sum_x (\sum_y \Pr[Y = y] \cdot \Pr[X = x | Y = y]) \cdot (\sum_{y'} \Pr[Y = y'] \cdot \Pr[X = x | Y = y']) \\
&= \sum_{y,y'} \Pr[Y = y] \cdot \Pr[Y = y'] \cdot \sum_x \Pr[X = x | Y = y] \cdot \Pr[X = x | Y = y'] \\
&\leq \sum_{y,y'} \Pr[Y = y] \cdot \Pr[Y = y'] \cdot \left( \sum_x \Pr[X = x | Y = y]^2 \right)^{1/2} \cdot \left( \sum_x \Pr[X = x | Y = y']^2 \right)^{1/2} \\
&= \sum_{y,y'} \Pr[Y = y] \cdot \Pr[Y = y'] \cdot \mathrm{cp}(X | Y = y)^{1/2} \cdot \mathrm{cp}(X | Y = y')^{1/2} \\
&\leq \max_{y \in \mathrm{Supp}(Y)} \mathrm{cp}(X | Y = y)
\end{aligned}$$

■

## 4.2 Extracting Randomness

A *randomness extractor* [NZ] can be viewed as a family of hash functions with the property that for any random variable $X$ with enough entropy, if we pick a random hash function $h$ from the family, then $h(X)$ is "close" to being uniformly distributed on the range of the hash function. Randomness extractors are a central object in the theory of pseudorandomness and have many applications in theoretical computer science. Thus there is a large body of work on the construction of randomness extractors. (See the surveys [NT, Sha]). A major emphasis in this line of work is constructing extractors where it takes extremely few (e.g. a logarithmic number of) random bits to choose a hash function from the family. This parameter is less crucial for us, so instead our emphasis is on using simple and very efficient hash functions (e.g. universal hash functions) and minimizing the amount of entropy needed from the source $X$. To do this, we will measure the quality of a hash family in ways that are tailored to our application, and thus we do not necessarily work with the standard definitions of extractors.

In requiring that the hashed value $h(X)$ be 'close' to uniform, the standard definition of an extractor uses the most natural measure of 'closeness'. Specifically, for random variables $X$ and $Y$, taking values in $[N]$, their *statistical difference*, is defined as $\Delta(X,Y) = \max_{S \subseteq [N]} |\Pr[X \in S] - \Pr[Y \in S]|$. $X$ and $Y$ are called $\varepsilon$*-close* if $\Delta(X,Y) \le \varepsilon$.

The classic Leftover Hash Lemma shows that universal hash functions are randomnness extractors with respect to statistical difference.

**Lemma 4.4 (The Leftover Hash Lemma [BBR, ILL])** *Let $H : [N] \to [M]$ be a random hash function from a 2-universal family $\mathcal{H}$. For every random variable $X$ taking values in $[N]$ with $\mathrm{cp}(X) \le 1/K$, we have $\mathrm{cp}(H, H(X)) \le (1/|\mathcal{H}|)\cdot(1/M+1/K)$, and thus $(H, H(X))$ is $(1/2)\cdot\sqrt{M/K}$-close to $(H, U_{[M]})$.*

Notice that the above lemma says that the *joint* distribution of $(H, H(X))$ is $\varepsilon$-close to uniform; a family of hash functions achieving this property is referred to as a "strong" randomness extractor. Up to some loss in the parameter $\varepsilon$ (which we will later want to save), this strong extraction property is equivalent to saying that with high probability over $h \leftarrow H$, the random variable $h(X)$ is close to uniform. The above formulation of the Leftover Hash Lemma, passing through collision probability, is attributed to Rackoff [IZ]. It relies on the fact that if the collision probability of a random variable is close to that uniform distribution, then the random variable is close to uniform in statistical difference. This fact is captured (in a more general form) by the following lemma.

**Lemma 4.5** *If $X$ takes values in $[M]$ and $\mathrm{cp}(X) \le 1/M + 1/K$, then:*

1. *For every function $f : [M] \to \mathbb{R}$,*

$$|\mathrm{E}[f(X)] - \mu| \le \sigma \cdot \sqrt{M/K},$$

*where $\mu$ is the expectation of $f(U_{[M]})$ and $\sigma$ is its standard deviation. In particular, if $f$ takes values in the interval $[a,b]$, then*

$$|\mathrm{E}[f(X)] - \mu| \le \sqrt{(\mu - a)\cdot(b-\mu)} \cdot \sqrt{M/K}.$$

2. *$X$ is $(1/2)\cdot\sqrt{M/K}$-close to $U_{[M]}$.*

**Proof:** Then,

$$\begin{aligned}
|\mathrm{E}[f(X)] - \mu| &= \left|\sum_x (f(x)-\mu)\cdot(\Pr[X=x]-1/M)\right| \\
&\le \sqrt{\sum_x (f(x)-\mu)^2}\cdot\sqrt{\sum_x (\Pr[X=x]-1/M)^2} \qquad \text{(Cauchy-Schwarz)} \\
&= \sqrt{M\cdot \mathrm{Var}[f(X)]}\cdot\sqrt{\sum_x (\Pr[X=x]^2 - 2\Pr[X=x]/M + 1/M^2)} \\
&\le \sqrt{M}\cdot\sigma\cdot\sqrt{\mathrm{cp}(X)-2/M+1/M} \\
&= \sigma\cdot\sqrt{M/K}.
\end{aligned}$$

The "in particular" follows from the fact that $\sigma[Y] \le \sqrt{(\mu-a)\cdot(b-\mu)}$ for every random variable $Y$ taking values in $[a,b]$ and having expectation $\mu$. Item 2 follows by noting that the statistical difference between $X$ and $U_{[M]}$ is the maximum of $|\,\mathrm{E}[f(X)] - \mathrm{E}[f(U_{[M]})]|$ over boolean functions $f$, which by Item 1 is at most $\sqrt{\mu(f)\cdot(1-\mu(f))}\cdot\sqrt{M/K} \le (1/2)\cdot\sqrt{M/K}$. ■

While the bound on statistical difference given by Item 2 is simpler to state, Item 1 often provides substantially stronger bounds. To see this, suppose there is a bad event $S$ of vanishing density, i.e. $|S| = o(M)$, and we would like to say that $\Pr[X\in S] = o(1)$. Using Item 2, we would need $K = \omega(M)$, i.e. $\mathrm{cp}(X) = (1+o(1))/M$. But applying Item 1 with $f$ equal to the characteristic function of $S$, we get the desired conclusion assuming only $K = O(M)$, i.e. $\mathrm{cp}(X) = O(1/M)$. Variations of Item 1 can be obtained by using Hölder's Inequality instead of Cauchy-Schwarz in the proof; these variations provide bounds in terms Renyi entropy of different orders (and different moments of $f(U_{[M]})$).

The classic approach to extracting randomness from block sources is to simply apply a (strong) randomness extractor, like the one in Lemma 4.4, to each block of the source. The distance from the uniform distribution grows linearly with the number of blocks.

**Theorem 4.6 ([CG, Zuc])** *If $\mathcal{H}$ is 2-universal, then for every block source $(X_1,\ldots,X_T)$ with collision probability $1/K$ per block, the random variable $(H, H(X_1),\ldots,H(X_T))$ is $(T/2)\cdot\sqrt{M/K}$-close to $(H, U_{[M]^T})$.*

Thus, if we have enough entropy per block, universal hash functions behave just like ideal hash functions. How much entropy do we need? To achieve an error $\varepsilon$ with the above theorem, we need $K \ge M\cdot T^2/(4\varepsilon^2)$. In the next section, we will explore how to improve the quadratic dependence on $\varepsilon$ and $T$.

## 4.3 Optimized Block-Source Extraction

Our approach to improving Theorem 4.6 is to change the order of steps in the analysis. As outlined above, Theorem 4.6 is proven by bounding the collision probability of each hashed block, passing to statistical difference via Lemma 4.5, and then summing the statistical difference over the blocks. Our approach is to try to work with collision probability for as much as the proof as possible, and only pass to statistical difference at the end if needed.

**Theorem 4.7** *Let $H : [N] \to [M]$ be a random hash function from a 2-universal family $\mathcal{H}$. For every block source $(X_1, \ldots, X_T)$ with collision probability $1/K$ per block and every $\varepsilon > 0$, the random variable $Y = (H(X_1), \ldots, H(X_T))$ is $\varepsilon$-close to a block source $Z$ with collision probability $1/M + T/(\varepsilon K)$ per block. In particular, if $K \geq MT^2/\varepsilon$, then $Z$ has collision probability at most $(1 + 2MT^2/(\varepsilon K))/M^T$.*

Note that the conclusion of this theorem is different than that of Theorem 4.6, in that we do not claim that $Y$ is close to uniform in statistical difference. However, if $K \geq MT/\varepsilon$, then *each block* of $Z$ has collision probability within a constant factor of the uniform distribution (conditioned on previous ones), and this property suffices for some applications (using Lemma 4.5, Item 1). In addition, if $K \geq MT^2/\varepsilon$, then globally, $Z$ is also has collision probability within a constant factor of uniform. Lemma 4.5 (Item 2) can then be used to deduce that $Y$ is close to uniform in statistical difference, but the resulting bound is no better than the classic one (Theorem 4.6). (We do not know whether this quadratic dependence on $T$ is necessary, and it would be interesting to remove it.)

The approach to proving the theorem is as follows. Intuitively, the Leftover Hash Lemma (Lemma 4.4) tells us that each hashed block $H(X_i)$ contributes collision probability at most $1/M + 1/K$, *on average* over the choice of the hash function $H$. We can use a Markov argument to say that the collision probability of each block is at most $1/M + T/(\varepsilon K)$ with probability at least $1 - \varepsilon/T$ over the choice of the hash function. Taking a union bound, the hash function is good for all blocks with probability at least $1 - \varepsilon$.

We begin with a reformulation of the (first part of the) Leftover Hash Lemma that refers to collision probability of the hashed block, both in expectation and with high probability over the hash function.

**Lemma 4.8** *Let $X$ be a random variable taking values in $[N]$ with $\mathrm{cp}(X) \leq 1/K$. For a fixed hash function $h : [N] \to [M]$, define $f(h) = \mathrm{cp}(h(X))$. Then if $H : [N] \to [M]$ is a random hash function from a 2-universal family, we have $\mathrm{E}[f(H)] \leq 1/M + 1/K$. Thus, with probability at least $1 - \delta$ over $h \leftarrow H$, we have $\mathrm{cp}(h(X)) \leq 1/M + 1/(\delta K)$.*

**Proof:** For a hash function $h \in \mathcal{H}$, define $f(h) = \mathrm{cp}(h(X))$. We compute the expectation of the random variable $f(H)$. Note that $f(h) = \Pr[h(X) = h(X')]$ where $X'$ is an iid copy of $X$. Thus,

$$\begin{aligned}
\mathrm{E}[f(H)] &= \Pr[H(X) = H(X')] \\
&\leq \Pr[X = X'] + \Pr[X \neq X'] \cdot \Pr[H(X) = H(X') | X \neq X'] \\
&= \mathrm{cp}(X) + (1 - \mathrm{cp}(X)) \cdot 1/M \\
&< 1/K + 1/M.
\end{aligned}$$

For the high probability statement, we note that $f(h)$ is always at least $1/M$ and apply Markov's Inequality to the random variable $f(H) - 1/M$. ■

The other lemma we need to establish Theorem 4.7 is the following.

**Lemma 4.9** *Suppose that for every random variable $X$ taking values in $[N]$ with $\mathrm{cp}(X) \leq 1/K$, we have $\mathrm{cp}(h(X)) \leq 1/M + 1/L$ with probability at least $1 - \delta$ over $h \leftarrow H$. Then for every block source $(X_1, \ldots, X_T)$ with collision probability $1/K$ per block, the random variable $Y =$*

$(H(X_1),\ldots,H(X_T))$ *is* $T\cdot\delta$*-close to a block source* $Y'$ *with collision probability* $(1/M+1/L)$ *per block. In particular, if* $L \geq MT$*, then* $Y'$ *has collision probability at most* $(1+2MT/L)/M^T$*.*

**Proof:** Let $X=(X_1,\ldots,X_T)$ be our block source with collision probability at most $1/K$ per block. For $x \in \mathrm{Supp}(X)$, $i \in [T]$, and a hash function $h:[N]\to[M]$, define

$$f_i(h,x) = \mathrm{cp}(h(X_i)|X_1=x_1,\ldots,X_{i-1}=x_{i-1}).$$

By hypothesis, it holds that for every $x$ and $i$ and a random hash function $H$, $\Pr[f_i(x,H) > 1/M+1/L] \leq \delta$. By a union bound, $\Pr[\exists i f_i(x,H) > 1/M+1/L] \leq T\cdot\delta$.

Define random variable $Z$ as follows: Sample $h \leftarrow H$, $x \leftarrow X$. If for all $i$, $f_i(x,h) \leq 1/M+1/L$, output $(h(x_1),\ldots,h(x_T))$. Otherwise, let $i \in [T]$ be the first index such that $f_i(x,h) > 1/M+1/L$. Choose $y_j,\ldots,y_T \leftarrow U_{[M]}$ and output $(h(x_1),\ldots,h(x_{j-1}),y_j,\ldots,y_T)$.

By what we have argued, $Z$ is $\varepsilon$-close to $Y=(H(X_1),\ldots,H(X_T))$. It remains to bound the collision probability of $Z$ per block. For every $j\in[T]$ and every $z \in \mathrm{Supp}(Z)$, we have

$$\begin{aligned}
\mathrm{cp}(Z_j|Z_1=z_1,\ldots,Z_{j-1}=z_{j-1}) &\leq \max_{h,x} \mathrm{cp}(Z_j|Z_1=z_1,\ldots,Z_{j-1}=z_{j-1},H=h,X_1=x_1,\ldots,X_{j-1}=x_{j-1}) \\
&\leq 1/M+1/L.
\end{aligned}$$

Here the maximum is taken over $h,x$ such that the event we condition on has nonzero probability. The first inequality follows from Lemma 4.3. We now argue that the second inequality follows from the definition of $Z$. Conditioned on $h$, $x_1,\ldots,x_{j-1}$, $z_1,\ldots,z_{j-1}$, there are two possibilities for the distribution of $Z_j$, according to the cases in the definition of $Z$. If for all $i \leq j$, we have $f_i(x,h) \leq 1/M+1/L$, then distribution is that of $h(X_j)$ (conditioned on $x_1,\ldots,x_{j-1}$), which has collision probability at most $f_j(x,h) \leq 1/M+1/L$. Otherwise, the distribution is simply $U_{[M]}$, which has collision probability at most $1/M$.

For the "in particular", we apply Lemma 4.2, noting that

$$\left(\frac{1}{M}+\frac{1}{L}\right)^T \leq \frac{e^{MT/L}}{M^T} \leq \frac{1+2MT/L}{M^T},$$

provided that $MT/L \leq 1$. ■

Using 4-wise independent hash functions, we can obtain the following stronger results.

**Theorem 4.10** *Let* $H:[N]\to[M]$ *be a random hash function from a 4-wise independent family* $\mathcal{H}$*. For every block source* $(X_1,\ldots,X_T)$ *with collision probability* $1/K$ *per block, the random variable* $Y=(H(X_1),\ldots,H(X_T))$ *has the following properties.*

1. *For every* $\varepsilon > 0$*,* $Y$ *is* $\varepsilon$*-close to a block source* $Z$ *with collision probability* $1/M+1/K+\sqrt{2T/(\varepsilon M)}\cdot 1/K$ *per block. In particular, if* $K \geq (MT+2M^{1/2}T^{3/2})/\varepsilon$*, then* $Z$ *has collision probability at most*
2. $Y$ *is* $O(((MT+M^{1/2}T^{3/2})/K)^{1/3})$*-close to* $U_{[M]^T}$

The improvement over Theorem 4.7 is that we only need $K$ to be on the order of $\max\{M, \sqrt{MT/\varepsilon}\}$ (as opposed to $MT$) for each block of $Z$ to have collision probability within a constant factor of uniform. Moreover, here we sometimes obtain an improvement over the classic block-source extraction (Theorem 4.6) in the total statistical difference from uniform. For example, in the common case that $T = \Theta(M)$, this theorem gives a statistical difference bound of $O(M^2/K)^{1/3}$ instead of $O(M^3/K)^{1/2}$; this is an improvement when $K = o(M^5)$.

The key idea in the proof is to show that, with 4-wise independence, the collision probability of each hashed block is concentrated around its expectation, which is at most $1/M + 1/K$ by Lemma 4.8. This concentration is obtained by bounding the variance.

**Lemma 4.11** *If $\mathcal{H}$ is 4-wise independent, then for every random variable $X$ taking values in $[N]$ with $\mathrm{cp}(X) \le 1/K$, we have $\mathrm{cp}(h(X)) \le 1/M + 1/K + \sqrt{2/(\delta M)} \cdot 1/K$ with probability at least $1 - \delta$ over $h \leftarrow H$.*

**Proof:** Let $f(h) = \mathrm{cp}(h(X))$. By Lemma 4.8, $\mathrm{E}[f(H)] \le 1/M + 1/K$. We compute the variance $\mathrm{Var}[f(H)] = \mathrm{E}[f(H)^2] - \mathrm{E}[f(H)]^2$. If we let $X', Y, Y'$ be iid copies of $X$, then we have the following.

$$
\begin{aligned}
&\mathrm{E}[f(H)^2] \\
&\quad= \Pr[H(X) = H(X') \wedge H(Y) = H(Y')] \\
&\quad\le \Pr[X = X' \wedge Y = Y'] \\
&\qquad + \Pr[(X = X' \wedge Y \ne Y') \vee (X \ne X' \wedge Y = Y')] \\
&\qquad\qquad \cdot \Pr[H(X) = H(X') \wedge H(Y) = H(Y') | (X = X' \wedge Y \ne Y') \vee (X \ne X' \wedge Y = Y')] \\
&\qquad + \Pr[H(X) = H(X') \wedge H(Y) = H(Y') | X \ne X' \wedge Y \ne Y' \wedge \{X, X'\} \ne \{Y, Y'\}] \\
&\qquad + \Pr[\{X, X'\} = \{Y, Y'\} \wedge X \ne X'] \cdot \Pr[H(X) = H(X') \wedge H(Y) = H(Y') | \{X, X'\} = \{Y, Y'\} \wedge X \ne X'] \\
&\quad\le \mathrm{cp}(X)^2 + 2\mathrm{cp}(X)(1 - \mathrm{cp}(X)) \cdot \frac{1}{M} + \frac{1}{M^2} + 2\mathrm{cp}(X)^2 \cdot \frac{1}{M} \\
&\quad\le \mathrm{E}[f(H)]^2 + \frac{2\mathrm{cp}(X)^2}{M}
\end{aligned}
$$

Thus, $\mathrm{Var}[f(H)] \le 2/(MK^2)$.

By Chebychev's Inequality,

$$\Pr[f(H) > 1/M + 1/K + \sqrt{2/(\delta M K^2)}] < \delta.$$

■

Theorem 4.10 follows from Lemma 4.11 via Lemma 4.9, analogous to the proof of Theorem 4.7.

# 5 Applications

## 5.1 Linear Probing

An immediate application of Theorem 4.6, using just a pairwise independent hash family, gives that if $K \ge MT^2/(2\epsilon)^2$, the resulting distribution of the element hashes is $\epsilon$-close to uniform. In this case, if we let $E_U$ be the expected time for an insertion when using a truly random hash function, and $E_P$ be the expected time for an insertion using pairwise independent hash functions, as the maximum number of collisions is $T$, we have

$$E_P \leq E_U + \epsilon T.$$

A natural choice is $\epsilon = o(1/T)$, so that the $\epsilon T$ term is lower order, giving that $K$ needs to be $\omega(MT^4) = \omega(M^5)$ in the standard case where $T = \Theta(M)$ (which we assume henceforth). This amount of entropy is not unreasonable in many applications. An alternative interpretation is that with probability $1-\epsilon$, our hash table behaves exactly as though a truly random hash function was used. In some applications, constant $\epsilon$ may be sufficient, in which case using Theorem 4.10 $K = O(M^2)$ suffices.

Better results can be obtained by applying Lemma 4.5, in conjunction with Theorem 4.7 or Theorem 4.10. In particular, for linear probing, the standard deviation $\sigma$ of the insertion time is known (see, e.g., [GB, p. 52]) and is $O(1/(1-\alpha)^2)$. With a 2-universal family, as long as $K \geq MT^2/\varepsilon$, from Theorem 4.7 the resulting hash values are $\varepsilon$-close to a block source with collision probability at most $(1+2MT^2/(\varepsilon K))/M^T$. Using this, we apply Lemma 4.5 to bound the expected insertion time as

$$E_P \leq E_U + \epsilon T + \sigma\sqrt{\frac{2MT^2}{\epsilon K}}.$$

Again, choosing $\varepsilon = o(1/T)$ gives that $E_P$ and $E_U$ are the same up to lower order terms when $K$ is $\omega(M^4)$. Theorem 4.10 gives further a small improvement; for $K \geq 3MT/\varepsilon$, we have

$$E_P \leq E_U + \epsilon T + \sigma\sqrt{\frac{3MT}{\epsilon K}}.$$

Choosing $\epsilon = o(1/T)$ now allows for $K$ to be only $\omega(M^3)$.

In other words, the Renyi entropy needs only to be $3\log M + \omega(1)$ bits when using 4-wise independent hash functions, and $4\log M + \omega(1)$ for 2-universal hash functions. These numbers seem quite reasonable for practical situations. We formalize the result for the case of 2-universal hash functions as follows:

**Theorem 5.1** *Let $H$ be a chosen at random from a 2-universal hash family $\mathcal{H}$ mapping $[N]$ to $[M]$. For every block source $X$ taking values in $[N]^T$ with collision probability at most $1/K$ and where $K \geq MT^2/\varepsilon$, we have $\mathrm{E}[\mathrm{Time}_{\mathrm{LP}}(X,H)] \leq 1/(1-\alpha)^2 + \epsilon T + \sigma\sqrt{\frac{2MT^2}{\epsilon K}}$. Here $\alpha = T/M$ is the load and $\sigma$ is the standard deviation in the insertion time in the case of truly random hash functions.*

## 5.2 Balanced Allocations

By combining the known analysis for ideal hashing (Theorem 3.4), our optimized bounds for block-source extraction (Theorems 4.7 and 4.10), and the effect of collision probability on expectations (Lemma 4.5), we obtain:

**Theorem 5.2** *For every $d \geq 2$ and $\gamma > 0$, there is a constant $c$ such the following holds. Let $H$ be chosen at random from a 2-universal hash family $\mathcal{H}$ mapping $[N]$ to $[T]^d$. For every block source $X$ taking values in $[N]^T$ with collision probability at most $1/2T^{d+2+\gamma}$ per block, we have*

$$\Pr[\mathrm{MaxLoad}_{\mathrm{BA}}(X,H) > \frac{\log\log T}{\log d} + c] \leq \frac{1}{T^\gamma}.$$

**Proof:** Set $M = T^d$. Note that the value of $\mathrm{MaxLoad}_{\mathrm{BA}}(\overline{x}, h)$ can be determined from the hashed sequence $(h(x_1), \ldots, h(x_T)) \in M^T$ alone, and does not otherwise depend on the data sequence $\overline{x}$ or the hash function $h$. Thus we can let $S \subseteq M^T$ be the set of all sequences of hashed values that produce an allocation with a max load greater than $(\log\log T)/(\log d) + c$. By Theorem 3.4, we can choose the constant $c$ such that

$$\Pr[U_{[M]^T} \in S] = \Pr[\mathrm{MaxLoad}_{\mathrm{BA}}(\overline{x}, I) > \frac{\log\log T}{\log d} + c] \leq \frac{1}{T^{3\gamma}},$$

where $I$ is an truly random hash function mapping $[N]$ to $[M] = [T]^d$ and $\overline{x}$ is an arbitrary sequence of distinct data items.

We are interested in the quantity:

$$\Pr[\mathrm{MaxLoad}_{\mathrm{BA}}(X, H) > \frac{\log\log T}{\log d} + c] = \Pr[(H(X_1), \ldots, H(X_T)) \in S].$$

Set $\varepsilon = 1/2T^\gamma$ and $K = 2T^{d+2+\gamma} = MT^2/\varepsilon$. By Theorem 4.7, $(H(X_1), \ldots, H(X_T))$ is $\varepsilon$-close to a random variable $Z$ with collision probability at most $(1 + 2MT^2/(\varepsilon K))/M^T = 3/M^T$ per block. Thus, applying Lemma 4.5 with $f$ as the characteristic function of $S$ and $\mu = \mathrm{E}[f(U_{[M]^T})] \leq 1/T^{3\gamma}$, we have

$$\begin{aligned}
\Pr[(H(X_1), \ldots, H(X_T)) \in S] &\leq \Pr[Z \in S] + \varepsilon \\
&\leq \mu + \sqrt{\mu \cdot (1-\mu)} \cdot \sqrt{3} + \varepsilon \\
&\leq \frac{1}{T^{3\gamma}} + \sqrt{\frac{3}{T^{3\gamma}}} + \frac{1}{2T^\gamma} \\
&\leq \frac{1}{T^\gamma},
\end{aligned}$$

for sufficiently large $T$. (Small values of $T$ can be handled by increasing the constant $c$ in the theorem.) ■

**Theorem 5.3** *For every $d \geq 2$ and $\gamma > 0$, there is a constant $c$ such the following holds. Let $H$ be chosen at random from a 4-wise independent hash family $\mathcal{H}$ mapping $[N]$ to $[T]^d$. For every block source $X$ taking values in $[N]^T$ with collision probability at most $1/6T^{d+1+\gamma}$ per block, we have*

$$\Pr[\mathrm{MaxLoad}_{\mathrm{BA}}(X, H) > \frac{\log\log T}{\log d} + c] \leq \frac{1}{T^\gamma}.$$

**Proof:** The proof is identical to that of Theorem 5.2, except we use Theorem 4.10 instead of Theorem 4.7 and set $K = 6T^{d+1+\gamma} \geq (MT + 2M^{1/2}T^{3/2})/\varepsilon$. ■

### 5.3 Bloom Filters

We consider there the following setting: our block source $X$ takes on values in $[N]^{T+1}$, producing a collection $(x_1, \ldots, x_T, y) = (\overline{x}, y)$, where $\overline{x}$ constitutes the set represented by the filter, and $y$ represents an additional element that we assume is not equal to any element of $\overline{x}$. We assume the element not in the set also comes from the block source as the set data. We could apply a hash function $h : [N] \to [M/\ell]^\ell \cong [(M/\ell)^\ell]$ to obtain $\ell$ hash values for each element. However, for brevity we will limit ourselves here to results taking advantage of [KM], as in Theorem 3.7, which utilizes only two hash functions two generate $\ell$ hash values, reducing the amount of randomness required.

If we allow the false positive probability to increase by some constant $\epsilon > 0$ over truly random hash functions, we can utilize Lemma 4.5 along with Theorems 4.7 and 4.10 to obtain the following parallels to Theorem 3.7:

**Theorem 5.4** *Let $H = (H_1, H_2)$ be chosen at random from a 2-universal hash family $\mathcal{H}$ mapping $[N]$ to $[M/\ell]^2$, where $M/\ell$ is a prime integer. Define $H' = (H'_1, \ldots, H'_\ell) : [N] \to [M/\ell]^\ell$ by $H'_i(w) = H_1(w) + (i-1)H_2(w) \bmod M/\ell$. For every $\epsilon > 0$, there is a constant $c$ such that if the block source $X$ taking values $(\overline{x}, y) \in [N]^{T+1}$ has collision probability at most $1/(cM^2(T+1)^2)$ per block, then*

$$\Pr[\text{FalsePos}_{\text{BF}}(H', \overline{x}, y) = 1] \leq \left(1 - \left(1 - \frac{\ell}{M}\right)^T\right)^\ell + \epsilon.$$

**Theorem 5.5** *Let $H = (H_1, H_2)$ be chosen at random from a 4-wise independent hash family $\mathcal{H}$ mapping $[N]$ to $[M/\ell]^2$, where $M/\ell > T$ is a prime integer. Define $H' = (H'_1, \ldots, H'_\ell) : [N] \to [M/\ell]^\ell$ by $H'_i(w) = H_1(w) + (i-1)H_2(w) \bmod M/\ell$. For every $\epsilon > 0$, there is a constant $c$ such that if the block source $X$ taking values $(\overline{x}, y) \in [N]^{T+1}$ has collision probability at most $1/(cM^2(T+1))$ per block, then*

$$\Pr[\text{FalsePos}_{\text{BF}}(H', \overline{x}, y) = 1] \leq \left(1 - \left(1 - \frac{\ell}{M}\right)^T\right)^\ell + \epsilon.$$

We can obtain false positive probabilities within $o(1)$ of truly random hash functions by requiring just slightly smaller collision probability. When $T = \Theta(M)$,the Renyi entropy needs only to be $3\log M + \omega(1)$ bits when using 4-wise independent hash functions, and $4\log M + \omega(1)$ for 2-universal hash functions.

# 6 Alternative Approaches

The results we have described in Section 4 rely on very general arguments, relying on the collision probability of the entire sequence of data values. We suggest, however, that it may prove useful in the future to view the results of Section 4 as a collection of tools that can be applied in various ways to specific arguments. For example, the fact derived in Theorems 4.7 and 4.10 that the hashed values are close to a block source with bounded collision probability *per block* may yield improved results in some cases.

We sketch an example of how these results can be applied to more specific arguments for an application. In the standard layered induction argument for balanced allocations [ABKU], the following key step is used. Suppose that there are at most $\beta_i n$ buckets with load at least $i$ throughout the process. Then (using truly random hash functions) the probability that an element with $d$ choices lands in a bin with $i$ or more balls already present is bounded above by $(\beta_i)^d$. When using 2-universal hash functions, we can bound this probability, but with slightly weaker results. The choices for an element corresponds to one of our blocks from the block source. Let $S$ be the set of size at most $(\beta_i)^d$ possible hash values for the element's choices that would place the element in a bin with $i$ or more balls. We can bound the probability that the element hashes to a value in $S$ by bounding the collision probability per block (via Theorem 4.7) and applying Lemma 4.5 with $f$

equal to the characteristic function of $S$. We have applied this technique to generalize the standard layered induction proof of [ABKU] to this setting. We find that while for example its application requires slightly less entropy from the source for 2-universal hash functions than Theorem 5.2, the bound from Lemma 4.5 is weak enough that the analysis only works for $d \geq 3$ choices and the the maximum load changes by a constant factor (although the $O(\log \log n)$ behavior is still apparent). We omit the details.

# 7 Conclusion

We have started to build a link between previous work on randomness extraction and the practical performance of simple hash functions, specifically 2-universal hash functions. While we expect that our results can be further improved, they already give bounds that appear to apply to many realistic hashing scenarios. In the future, we hope that there will be a collaboration between theory and systems researchers aimed at fully understanding the behavior of hashing in practice. Indeed, while our view of data as coming from a block source is a natural initial suggestion, such interaction could lead to more refined and realistic models for real-life data (and in particular, provide estimates for the amount of entropy in the data). Another complementary direction is to show that hash functions used in practice (such as those based on cryptographic functions, which may not even be 2-universal) behave similarly to truly random hash functions for these data models. Some results in this direction can be found in [DGH$^+$].

# Acknowledgments

We thank Adam Kirsch for his careful reading of and helpful comments on the paper.

# References

[ABKU] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced Allocations. *SIAM Journal on Computing*, 29(1):180–200, 2000.

[BBR] C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988. Special issue on cryptography.

[Blo] B. H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7):422–426, 1970.

[BS] A. Blum and J. Spencer. Coloring random and semi-random $k$-colorable graphs. *Journal of Algorithms*, 19(2):204–234, 1995.

[BM1] A. Broder and M. Mitzenmacher. Using multiple hash functions to improve IP lookups. In *INFOCOM 2001: Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1454–1463, 2001.

[BM2] A. Broder and M. Mitzenmacher. Network Applications of Bloom Filters: A Survey. *Internet Mathematics*, 1(4):485–509, 2005.

[CW] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.

[CG] B. Chor and O. Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM J. Comput.*, 17(2):230–261, Apr. 1988.

[DKSL] S. Dharmapurikar, P. Krishnamurthy, T. Sproull, and J. Lockwood. Deep packet inspection using parallel bloom filters. *IEEE Micro*, 24(1):52–61, 2004.

[DW] M. Dietzfelbinger and P. Woelfel. Almost random graphs with simple hash functions. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 629–638, New York, NY, USA, 2003. ACM Press.

[DGH+] Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin. Randomness Extraction and Key Derivation Using the CBC, Cascade and HMAC Modes. In M. K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 494–510. Springer, 2004.

[GB] G. Gonnet and R. Baeza-Yates. *Handbook of algorithms and data structures: in Pascal and C*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1991.

[ILL] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random Generation from one-way functions (Extended Abstracts). In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, Washington, 15–17 May 1989.

[IZ] R. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. In *30th Annual Symposium on Foundations of Computer Science*, pages 248–253, Research Triangle Park, North Carolina, 30 Oct.–1 Nov. 1989. IEEE.

[KM] A. Kirsch and M. Mitzenmacher. Less Hashing, Same Performance: Building a Better Bloom Filter. In *Proceedings of the 14th Annual European Symposium on Algorithms*, pages 456–467, 2006.

[Knu] D. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison Wesley Longman Publishing Co., Inc. Redwood City, CA, USA, 1998.

[Mut] S. Muthukrishnan. Data Streams: Algorithms and Applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

[NT] N. Nisan and A. Ta-Shma. Extracting Randomness: A Survey and New Constructions. *Journal of Computer and System Sciences*, 58(1):148–173, 1999.

[NZ] N. Nisan and D. Zuckerman. Randomness is Linear in Space. *J. Comput. Syst. Sci.*, 52(1):43–52, Feb. 1996.

[OP] A. Ostlin and R. Pagh. Uniform hashing in constant time and linear space. *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 622–628, 2003.

[PPR] A. Pagh, R. Pagh, and M. Ruzic. Linear probing with constant independence. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 318–327, New York, NY, USA, 2007. ACM Press.

[PR] R. Pagh and F. Rodler. Cuckoo hashing. *Journal of Algorithms*, 51(2):122–144, 2004.

[Ram1] M. V. Ramakrishna. Hashing practice: analysis of hashing and universal hashing. In *SIGMOD '88: Proceedings of the 1988 ACM SIGMOD international conference on Management of data*, pages 191–199, New York, NY, USA, 1988. ACM Press.

[Ram2] M. V. Ramakrishna. Practical performance of Bloom filters and parallel free-text searching. *Communications of the ACM*, 32(10):1237–1239, 1989.

[RFB] M. V. Ramakrishna, E. Fu, and E. Bahcekapili. Efficient Hardware Hashing Functions for High Performance Computers. *IEEE Trans. Comput.*, 46(12):1378–1381, 1997.

[SV] M. Santha and U. V. Vazirani. Generating Quasi-random Sequences from Semi-random Sources. *J. Comput. Syst. Sci.*, 33(1):75–87, Aug. 1986.

[SS] J. P. Schmidt and A. Siegel. The Analysis of Closed Hashing under Limited Randomness (Extended Abstract). In *STOC*, pages 224–234. ACM, 1990.

[Sha] R. Shaltiel. Recent Developments in Explicit Constructions of Extractors. *Bulletin of the European Association for Theoretical Computer Science*, 77:67–95, June 2002.

[Sie] A. Siegel. On universal classes of extremely random constant-time hash functions. *SIAM Journal on Computing*, 33(3):505–543 (electronic), 2004.

[ST] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463 (electronic), 2004.

[TZ] M. Thorup and Y. Zhang. Tabulation based 4-universal hashing with applications to second moment estimation. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 615–624 (electronic), New York, 2004. ACM.

[Vöc] B. Vöcking. How asymmetry helps load balancing. *Journal of the ACM*, 50(4):568–589, 2003.

[WC] M. N. Wegman and J. L. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265–279, 1981. Special issue dedicated to Michael Machtey.

[Woe] P. Woelfel. Asymmetric balanced allocation with simple hash functions. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithms*, pages 424–433, New York, NY, USA, 2006. ACM Press.

[Zuc] D. Zuckerman. Simulating BPP Using a General Weak Random Source. *Algorithmica*, 16(4/5):367–391, Oct./Nov. 1996.