Hashing: Construction of A Hash Family With Smaller Description Length

Mingkun Ni, Yuanhao Zhang
{m8ni, y2384zha}@uwaterloo.ca
University of Waterloo
Waterloo, ON, Canada

Abstract

This paper introduces and proves a new construction that could guarantee a $O(\log n/\log\log n)$ maximum load when throw n balls into n bins with a smaller description length. It is well-known that, with high probability, a traditional $O(\log n/\log\log n)$-wise independent hash family would guarantee maximum load of $O(\log n/\log\log n)$. The traditional analysis of $O(\log n/\log\log n)$-wise independent function can be described by $O(\log^2 n/\log\log n)$ bits, which already yields a dramatic improvement over a truly random function. This paper first introduces a specific hash family that requires $O(\log n)$ description length and $O(\log n)$ computation times. Based on this hash family, this research describes a construction whose overall hash function could be described in $O(\log n\log\log n)$ bits and computed in $O(\log n\log\log n)$ times while maintaining the guarantee of $O(\log n/\log\log n)$ maximum load. The special part of this research is that it constructs an innovative structure of a multi-layer random graph. Within the special construction, each layer is considered as a different hash process with different input and output sizes.

1. Introduction

A traditional analysis of the randomized algorithm of the Balls and Bins problem maps m balls into n bins independently and uniformly guarantees that each bin contains at most $O(\log n/\log\log n)$ balls with high probability, as known as the maximum load of the balls and bins problem. For a truly random hash function $h(x) : M \to N$, it would take $O(m\log n)$ space to store it. However, the traditional analysis with the use of truly random hash functions is impractical in various real-world applications because of the space to store the hash functions. Hence, a weaker notion of randomness, named k-wise independence, is introduced to solve this issue. It is specifically well-studied in the case of mapping

n balls into n bins that any $O(\log n/\log\log n)$-wise independent hash families can guarantee the maximum load of $O(\log n/\log\log n)$ with high probability.

This paper will continue to study the problems of mapping n balls into n bins with a construction of hash functions that require a smaller description length given the inspiration from the paper, titled *Ball and bins: smaller hash families and faster evaluation.* By using a $O(\log n/\log\log n)$-wise independent hash families, the hash functions can be described by $O(\log^2 n/\log\log n)$ bits, which itself yields a dramatic improvement over the description length of a truly random functions. We would like to provide an explicit family of hash functions to guarantee the same maximum load of $O(\log n/\log\log n)$ with high probability, and each hash function can be strictly described by $o(\log^2 n/\log\log n)$ bits. We provide an overview of the construction below.

1.1 - Construction:

This construction, obtained from the paper titled *Ball and bins: smaller hash families and faster evaluation*, concatenates the output of $O(\log\log n)$ functions. Each function $f(x)$ is described using $O(\log n\log\log n)$ bits, which is significantly smaller than the traditional method. Moreover, we can still guarantee the maximum load of $O(\log n/\log\log n)$ with high probability for overall hash function $f(x)$:

$$f(x) = h_1(x) \circ ... \circ h_d(x),$$

For each hash function, $h_i(x)$, the output is a binary string, and $\circ$ denotes the concatenation operator on binary strings. The first function $h_1(x)$ is $O(1)$-wise independent. The level of independence gradually increases to $O(\log n/\log\log n)$-wise independence for the last function $h_d(x)$. Similarly, these output length of the functions decrease from $\Omega(\log n)$ bits for h_1 to $O(\log\log n)$ bits for h_d. We will prove in Section 3.1 and Section 3.2 that this construction suffice the requirement of keeping the maximum load with a smaller description length of the hash family.

1.2 - Contribution:

The construction described in section 1.1 is originally from the paper, titled *Ball and bins: smaller hash families and faster evaluation.* This construction is the study results of L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. This paper is written to fully understand the construction presented in Section 1.1 and we will, in the end, present a full and complete tour guide in understanding that this construction indeed guaranteed the same maximum load with a smaller description length. In the original paper, key lemmas and corollaries that were used in proving the construction either had no proof or skipped many important steps which resulted in much confusion. By our own research, we added step-by-step details in proving these important lemmas and corollaries to help readers fully understand the construction.

1.3 - Outline:

In Section 2, we will introduce a few pieces of terminology, definitions, lemmas, and theorems that we will use in later sections. We will also include detailed proof of each theorem that is related to our construction. Section 3 is an essential part of this research paper. It will contain the formal introduction of our construction and formal proof of why this construction works. It will first give a formal description of the construction, and we will analyze the construction in which we will be essentially explaining the interpretation of such construction. Finally, we will prove step-by-step that the construction guarantees the description length and computation time of the overall hashing functions. Finally, Section 4 will introduce the extensional use of this construction. It will also analyze this construction with correspondence to the trade-off it makes to obtain the smaller description length of the function.

2. Preliminaries

In this section, we present the relevant definitions, lemmas, theorems required to prove the construction.

2.1 - Definitions:

We use the unit RAM model throughout the paper. In the RAM model, we assume that we can access an arbitrary position of an array in $O(1)$ time. We also assume that word size is large enough such that it takes $O(1)$ word operation. For the balls and bins problem, we are considering the case where there are exactly n balls and n bins. We want to achieve a maximum load of $O(\log n/\log\log n)$ with smaller than usual description length under this condition. The following are some definitions and terms that we will be using frequently throughout the paper.

1. We set log to be base of 2 as default if no base is specified.
2. For a natural number u, we define the set of integers $\{1, 2, ..., u\}$ as [u]. For example, $[5] = \{1, 2, 3, 4, 5\}$.
3. The term $x \in X$ represents a sample x from a random variable X.
4. $SD(X, Y)$ represents the statistical distance between two random variables over a finite domain

$$SD(X,Y) = \frac{1}{2}\sum_{\omega\in\Omega} |Pr(X=\omega) - Pr(Y=\omega)|$$

5. The term $x \in S$ means that we draw a random sample x uniformly from the finite set S.
6. For two bit-string x and y, $x \circ y$ represents the concatenation of x and y bit-string. For example, for $x = 1001$ and $y = 0111$,

$$x \circ y = 1001 \circ 0111 = 10010111$$

2.2 - More Definitions:

We present a few more definitions. These definitions are more complicated but essential to understanding the proof of the construction. Some definition will be used to prove the existence of the selected hash family

Definition 2.2.1: For a family of function f:$[u] \to [v]$, it is defined as a k-wise δ-dependent iff

$$SD(X, Y) \leq \delta$$

For $X =$ distribution $(f(x_1), f(x_2), ..., f(x_k))$ for any distinct $x_1, x_2, ..., x_k \in [u]$ and $Y =$ uniform distribution over $[v]^k$. It will take $O(k \max\{\log u, \log v\})$ bits to describe f and $O(k)$ times to calculate f in the unit RAM model.

Definition 2.2.2: A sequence of random variables $X_1, X_2, ..., X_n$ over $\{0, 1\}$ are ϵ-biased distribution if, for any non-empty set $S \subseteq [n]$,

$$|Pr(\oplus_{i \in S} X_i = 1) - Pr(\oplus_{i \in S} X_i = 0)| \leq \epsilon$$

Note that the following definitions will be used to understand Definition 2.2.2, which is ϵ-biased distribution. It's important to fully understand the definition and derivation of ϵ-biased distribution because it is used in the proof of our construction. This ϵ-biased distribution is based on the work did by Alon et al. included in the paper titled *Simple Constructions of Almost k-wise Independent Random Variables*(Alon et al., 1992)

Almost k-wise independence: Let S_n be a sample space and $X = \{x_1 ... x_n\}$ be selected uniformly from S_n

1. S_n is (ϵ, k)**-independent (in max form)** if for any k positions $i_1 < i_2 < ... < i_k$ and any k-bit string α, we have

$$|Pr[x_{i_1} x_{i_2} ... x_{i_k}] - 2^{-k}| \leq \epsilon$$

2. S_n is ϵ**-away (in L_1 form)** for k-wise independence if for any k positions $i_1 < i_2 < ... < i_k$, we have

$$\sum_{\alpha \in \{0,1\}^k} |Pr[x_{i_1} x_{i_2} ... x_{i_k}] - 2^{-k}| \leq \epsilon$$

Clearly, we could transfer between these two definitions of almost k-wise independence:

1. S_n is (ϵ, k)**-independent (in max form)**. Then, since there are at most 2^k different α, we could state this equations:

$$|Pr[x_{i_1} x_{i_2} ... x_{i_k}] - 2^{-k}| \leq \epsilon \Rightarrow \sum_{\alpha \in \{0,1\}^k} |Pr[x_{i_1} x_{i_2} ... x_{i_k}] - 2^{-k}| \leq 2^k \epsilon$$

Thus, S_n is at most $2^k \epsilon$**-away** from k-wise independence.

2. S_n is **ϵ-away (in L_1 form)** for k-wise independence. Then, we could simply state that:

$$\sum_{\alpha \in \{0,1\}^k} |Pr[x_{i_1}x_{i_2}...x_{i_k}] - 2^{-k}| \leq \epsilon \Rightarrow |Pr[x_{i_1}x_{i_2}...x_{i_k}] - 2^{-k}| \leq \epsilon$$

Thus, S_n is also **(ϵ, k)-independent**

Linear Test: Linear test refers to "linear Boolean tests" or test which take the **exclusive-or** of the bits in some fixed location in the strings. In particular, Linear test of size k means to perform linear boolean tests on k different select positions.

Definition Set: This set includes various definitions used to prove the existence of an useful ϵ-biased distribution:

1. Let $(\alpha, \beta)_2$ denote the **inner-product-mod-2** of the binary string $\alpha = \alpha_1\alpha_2...\alpha_n$ and $\beta = \beta_1\beta_2...\beta_n$.

$$(\alpha, \beta)_2 = (\alpha_1\alpha_2...\alpha_n, \beta_1\beta_2...\beta_n)_2 = (\sum_{i=1}^{n} \alpha_i\beta_i) \mod 2$$

2. A $0-1$ random variable X is **ϵ-biased** if

$$|Pr([x = 0]) - Pr([x = 1])| \leq \epsilon$$

3. Let S_n be a sample space and $X = x_1x_2...x_n$ be selected uniformly randomly from S_n. The sample space S_n is said to be **ϵ-biased** with respect to linear tests if, for every $\alpha = \alpha_1\alpha_2...\alpha_n \in \{0,1\}^n - \{0\}^n$, the random variable $(\alpha, X)_2$ is ϵ-biased.

4. The sample space S_n is said to be **ϵ-biased** with respect to linear tests of size **at most k** if, for every $\alpha = \alpha_1\alpha_2...\alpha_n \in \{0,1\}^n - \{0\}^n$ such that **at most k of the α_i are 1**, the random variable $(\alpha, X)_2$ is ϵ-biased.

Notice that the definition of ϵ-biased distribution and ϵ-biased sample space are actually the same with respect to linear tests using XOR operation since we could treat:

1. $X_1, X_2, ..., X_n$ in the definition of ϵ-biased distribution as $x_1, x_2, ..., x_n$ which is bits of random variable X in the definition of ϵ-biased sample

2. S in the definition of ϵ-biased distribution as α in the definition of ϵ-biased sample space

3. $\oplus$ XOR operation in the definition of ϵ-biased distribution as the inner-product-mod-2 operation in the definition of ϵ-biased sample space

2.3 - Theorems:

In this section, we will present the key lemmas, corollaries, and theorems that we use to prove the construction. Please refer to Section 2.1 and Section 2.2 if the reader encounters unknown definitions.

We will first state that there exists a family of k-wise δ-dependent function that meets our requirement about describe space and computation times. This hash family will be used as our selection of h_i for $i \in [d]$. By work conducted by Alon et all(1992), we will offer detail about this family of k-wise δ-dependent functions and proof of its space and time requirement. After that, we will state a useful tail bound for limited independence based on work did by Bellare(1994). It will help us limit the probability that the sum of n 2k-wise δ-dependent random variables diverges from its expectation.

First, we will use following family of k-wise δ-dependent function:
Corollary 2.3.1: For any integers a and integer b such that b is a power of 2, there exists a family of k-wise δ-dependent function f that maps from [a] to [b], that is $f : [a] \to [b]$. This function can be described by $O(\log a + k \log b + \log(1/\delta))$ bits. More importantly, in the RAM model, if a word size is $\Omega(\log a + k \log b + \log(1/\delta))$, then this function can be evaluated in $O(\log a + k \log b + \log(1/\delta))$ time.

To prove Corollary 2.3.1, we will need to understand the following fact:
Fact 2.3.2: For any k, an ϵ-biased distribution is also k-wise δ-dependent for $\delta = \epsilon 2^{\frac{k}{2}}$

To prove Fact 2.3.2, we will need to use the following two lemma and corollary. For the proofs of the Lemma 2.3.2.1 and Corollary 2.3.2.2, please refer to Appendix.
Lemma 2.3.2.1: Let $S_n \subseteq 0, 1^n$ be a sample space that is ϵ-biased with respect to linear tests of size at most k. Then, the sample space S_n is $((1 - 2^{-k})\epsilon, k)$-independent(in max form), and $(2^k - 1)^{\frac{1}{2}}\epsilon$-away(in L_1 norm) from k-wise independent.

Corollary 2.3.2.2: Let $S_n \in \{0, 1\}^n$ be a sample space that is ϵ-biased with respect to linear tests. Then, for every k, the sample space S_n is $((1 - 2^{-k})\epsilon, k)$-independent (in max norm), and $(2^k - 1)^{1/2}\epsilon$-away in (L_1 norm) from k-wise independence.

Proof of Fact 2.3.2
We know that from Lemma 2.3.2.1 and Corollary 2.3.2.2 (See Appendix),

$$\begin{aligned}
\text{any } \epsilon\text{-biased distribution} &\Rightarrow (2^k - 1)^{\frac{1}{2}}\epsilon\text{-away} \\
&\Rightarrow \sum_{\omega \in \Omega} |Pr(X = \omega) - \frac{1}{2^k}| \le (2^k - 1)^{\frac{1}{2}}\epsilon \le 2^{\frac{k}{2}+1}\epsilon \\
&\Rightarrow \sum_{\omega \in \Omega} |Pr(X = \omega) - \frac{1}{2^k}| \le 2\delta \\
&\Rightarrow \frac{1}{2}\sum_{\omega \in \Omega} |Pr(X = \omega) - \frac{1}{2^k}| \le \delta \\
&\Rightarrow SD(X, U) \le \delta, U = \text{Uniform Distribution over } [b]^k \\
&\Rightarrow k\text{-wise } \delta\text{-dependent}
\end{aligned}$$

Thus, the fact 2.3.2 stands. ■

Now given the fact 2.3.2, we could transfer any ϵ-biased distribution to a family of k-wise δ-dependent functions. In the work completed by Alon et al(1992), they introduced following construction that forms a sample space that is nearly ϵ-biased. We will introduce this construction below and show its connection with Lemma 2.3.1. Now, let S_n^{2m} represents the sample space of 2^{2m} strings whose length is n.

Construction(Alen et al, 1992)

1. Let $m = \log \frac{n}{\epsilon}$ and $bin : GF(2^m) \to \{0,1\}^m$ be a one-to-one mapping function satisfies that $bin(0) = \{0\}^m$ and $bin(u+v) = bin(u) \oplus bin(v)$, which should be satisfied by the standard representation of $GF(2^m)$.

2. We have two elements x, y that are used to calculate every string s in S_n^{2m}. Let $s = s_1 s_2 ... s_n$, then

$$s_i = (bin(x^i), bin(y))_2 \qquad i \in [n]$$

   where x^i is the ith power of x when considered as an element in $GF(2^m)$. This means x^i will become $x^i \mod 2^m$ iff $x^i \geq 2^m$.

Regards to this construction, we have following claim:
Claim: S_n^{2m} specified above is $\frac{n-1}{2^m}$-biased with respect to linear tests. For any non-zero α, the random variable $(\alpha, r)_2$ is $(n-1)2^{-m}$-biased when r is uniformly selected from S_n^{2m}.

Proof of Claim:
Let $r(x,y) = r_0(x,y) r_1(x,y) ... r_n(x,y)$. Then,

$$\begin{aligned}
(\alpha, r(x,y))_2 &= \sum_{i=0}^{n-1} \alpha_i r_i(x,y) \mod 2 \\
&= \sum_{i=0}^{n-1} \alpha_i (bin(x^i), bin(y))_2 \mod 2 \\
&= \sum_{i=0}^{n-1} \alpha_i \sum_{j=0}^{n-1} bin(x^i)_j bin(y)_j \mod 2 \mod 2 \\
&= \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \alpha_i bin(x^i)_j bin(y)_j \mod 2 \mod 2 \\
&= \sum_{j=0}^{n-1} bin \left(\sum_{i=0}^{n-1} \alpha_i x^i \mod 2 \right)_j bin(y)_j \mod 2
\end{aligned}$$

Now, we define $p_\alpha(t) = \sum_{i=0}^{n-1} \alpha_i t^i$ as a function over $GF(2)$, and we could get:

$$\begin{aligned}
(\alpha, r(x,y))_2 &= \sum_{j=0}^{n-1} bin \left(\sum_{i=0}^{n-1} \alpha_i x^i \mod 2 \right)_j bin(y_j) \mod 2 \\
&= (bin(\sum_{i=0}^{n-1} \alpha_i x^i), bin(y))_2
\end{aligned}$$

$$= (bin(p_\alpha(x)), bin(y))_2$$

When $x, y \in GF(2^m)$ are chosen uniformly, we fix x:

1. x is not a zero of $p_\alpha(x)$
   This means $p_\alpha(x) \neq 0$ and $bin(p_\alpha(x)) \neq \{0\}^m$. Set $bin(p_\alpha(x)) = p_1p_2...p_m$, then
   $$(bin(p_\alpha(x)), bin(y))_2 = \sum_{j=0}^{m} p_j bin(y)_j \mod 2$$
   Then, we could state that $(bin(p_\alpha(x)), bin(y))_2$ is unbiased since y is uniformly selected from $GF(2^m)$

2. x is a zero of $p_\alpha(x)$
   This means $p_\alpha(x) = 0$ and $bin(p_\alpha(x)) = \{0\}^m$. Therefore,
   $$(bin(p_\alpha(x)), bin(y))_2 = 0 \qquad \text{for any y} \in GF(2^m)$$
   However, $p_\alpha(t) hasatmostn - 1zeros$ and $x \in GF(2^m)$. Thus, there are at most $\frac{n-1}{2^m}$ of all values of x such that this case happens. Thus, S_n^{2m} is $\frac{n-1}{2^m}$ biased in this case.

In conclusion, S_n^{2m} is $\frac{n-1}{2^m}$ biased overall. ■

Now we will elaborate the connection between the construction above and Corollary 2.3.1. We will show that this construction will meet the space and time requirement specified in Corollary 2.3.1

Connection to Corollary 2.3.1:
We set $m = O(\log \frac{n}{\epsilon})$ where $\epsilon = \delta 2^{\frac{-k \log b}{2}}$, and for function $f : [a] \rightarrow [b]$, we set $n = a \log b$ which stands for a blocks and each block has length $\log b$. Each block represents a single output in $[b]$. For space requirement, since this construction outputs $\{0,1\}^m$ bits, we will use $O(m) = O(\log \frac{n}{\epsilon})$ bits to describe a function. For time requirement, we calculate the ith bit of output as $(bin(x^i), bin(y))_2$. This calculation will use also $O(m) = O(\log \frac{n}{\epsilon})$ times. On the other hand:

$$\begin{aligned} O(\log \frac{n}{\epsilon}) &= O(\log \frac{a \log b}{\delta 2^{\frac{-k \log b}{2}}}) \\ &= O(\log a + \log \log b + \log(\frac{1}{\delta}) + \frac{k \log b}{2}) \\ &= O(\log a + k \log b + \log(\frac{1}{\delta})) \end{aligned}$$

Thus, we could derive the space and time requirement mentioned in Lemma 2.3.1.

Now, we states that this sample space S_n^{2m} is ϵ-biased because S_n^{2m} is $\frac{n-1}{2^m}$-biased as the claim states, and:

$$\frac{n-1}{2^m} = \frac{n-1}{2^{O(\log \frac{n}{\epsilon})}} = O(\frac{n-1}{\frac{n}{\epsilon}}) \approx O(\epsilon) \Rightarrow \epsilon\text{-biased by definition}$$

■

Now, after proved Lemma 2.3.1, we will state a bound for random variables with limited independence. We will introduce Lemma 2.2 of paper titled *Randomness Efficient Oblivious Sampling*. This lemma is very important in proving our construction. We first state the Lemma below.

Lemma 2.3.2: Define n random variables, X_1, ..., $X_n \in \{0,1\}$ be 2k-wise δ-dependent random variables, for some $k \in \mathbb{N}$ and for some $0 \leq \delta < 1$. We represent $X = \sum_{i \in n} X_i$ and $\mu = E[X]$. Then for any $t > 0$, we have that:

$$Pr[|X-\mu| > t] \leq 2(\frac{2nk}{t^2})^k + \delta(\frac{n}{t})^{2k}$$

To prove Lemma 2.3.2, we will need a Lemma conducted by Bellare(1994) from the paper titled *Randomness Efficient Oblivious Sampling*. It is named as the First t-wise Independent Tail Inequality in the original paper and labeled as Lemma 2.3.3 here.

Lemma 2.3.3 (First t-wise Independent Tail Inequality): Let $t \geq 4$ be an even integer, Suppose $X_1, ..., X_n$ are t-wise independent random variables taking values in [0, 1]. Let $X = \sum_{i \in n} X_i$ and $\mu = E[x]$, then we have for $A > 0$,

$$Pr[|X-\mu| \geq A] \leq C_t(\frac{nt}{A^2})^{t/2}$$

Proof of Lemma 2.3.2:
With Lemma 2.3.3 setup, we can start proving Lemma 2.3.2. First apply Markov's Inequality, we obtain that

$$\begin{aligned} Pr\left[|X-\mu| > t\right] &\leq Pr\left[(X-\mu)^{2k} > t^{2k}\right] \\ &\leq \frac{E[(X-\mu)^{2k}]}{t^{2k}} \end{aligned}$$

For $\mu_i = E[X_i]$, we consider the the following terms $E[X-\mu]$, $E[(X-\mu)^2]$, ..., and $E[(X-\mu)^{2k}]$:

$$\begin{aligned} E[X-\mu] &= \sum_{i \in [n]} E[X_i - \mu_i] \\ E[(X-\mu)^2] &= \sum_{i \in [n]} \sum_{j \in [n]} E[(X_{ij} - \mu_{ij})^2] \\ &= \sum_{i_1, i_2 \in [n]} E[\Pi_{j \in [2]}(X_{i_j} - \mu_{i_j})] \end{aligned}$$

Then similarly, we can prove that

$$E[(X-\mu)^{2k}] = \sum_{i_1, i_2, \ldots, i_{2k} \in [n]} E[\Pi_{j \in [2k]}(X_{i_j} - \mu_{i_j})]$$

Then we will continue to prove Lemma 2.3.2 while substituting this result,

$$\begin{aligned} Pr\left[|X-\mu| > t\right] &\leq \frac{E[(X-\mu)^{2k}]}{t^{2k}} \\ &= \frac{\sum_{i_1, i_2, \ldots, i_{2k} \in [n]} E[\Pi_{j \in [2k]}(X_{i_j} - \mu_{i_j})]}{t^{2k}} \end{aligned}$$

we know that $\delta \geq 0$, $n \geq 0$, and $k \geq 0$

$$\leq \frac{\sum_{i_1,i_2,...,i_{2k}\in[n]} E[\Pi_{j\in[2k]}(\hat{X}_{i_j} - \mu_{i_j})] + \delta n^{2k}}{t^{2k}}$$

$$= \frac{E[(\hat{X} - \mu)^{2k}] + \delta n^{2k}}{t^{2k}}$$

For every $\hat{X}_i$ for $i \in [n]$, $\hat{X}_i$ are independent random variables that have the same marginal distribution $X_1, ..., X_n$. In addition, we apply Lemma 2.3.3 (First t-wise Independent Tail Inequality) here, which is the following:

$$Pr[(\hat{X} - \mu) > A) \leq \frac{E[(\hat{X} - \mu)^t]}{A^t} \leq C_t \times (\frac{n^t}{A^2})^{t/2}$$

Substituting t into A, we will obtain that

$$Pr[|X - \mu| > t] \leq \frac{E[(\hat{X} - \mu)^{2k}] + \delta n^{2k}}{t^{2k}} \leq 2(\frac{2nk}{t^2})^k + \delta(\frac{n}{t})^2 k \qquad \blacksquare$$

3. Construction

In this section, we will first present the construction, mentioned in Section 1.1, based on the gradually increasing independence. This construction will guarantee a maximum load of $O(\log n/\log\log n)$ using description length $O(\log n \log\log n)$ and computation time $O(\log n \log\log n)$. We state the following theorem:

Theorem 3.1: For any constant $c > 0$, integers n and $u = \text{poly}(n)$, there exists a family $\mathscr{F}$ of $f : [u] \to [v]$ such that:

1. could be described using $O(\log n \log\log n)$ bits
2. $f(x)$ can be computed in $O(\log n \log\log n)$ using unit cost RAM model for any $f \in \mathscr{F}$ and $x \in [u]$
3. $\exists\gamma > 0$ such that, for any $S \subseteq [u]$, $|S| = n$ and an error parameter c,

$$Pr_{f\leftarrow\mathscr{F}}\left[\max_{i\in[n]} |f^{-1}(i) \cap S| \leq \frac{\gamma \log n}{\log\log n}\right] > 1 - \frac{1}{n^c}$$

Under our construction, statement 3 of Theorem 3.1 could be interpreted in following way:

For any bin in the last layer,

the maximum load of this bin is $O(\frac{\log n}{\log\log n})$ with high probability.

In what follows we provide a more formal description of our construction (see Section 3.1), and then prove Theorem 3.1 (see Section 3.2).

3.1 - Formal Description of Construction

There are two cases about n: whether n is a power of 2 or not. We only consider the case that $n = 2^k$ since we could set the number of bins to be $m = 2^{\lfloor \log_2 n \rfloor}$ and the influence on maximum load will only be at most a factor of two. Let $d = O(\log \log n)$, and for every $i \in [d]$, let $\mathscr{H}_i$ be a family of k_i-wise δ-dependent functions $[u] \to 0, 1^{l_i}$, where:

1. $n_0 = n, n_i = \frac{n^{i-1}}{2^{l_i}}\ \forall i \in [d]$
2. $l_i = \lfloor \frac{\log n_{i-1}}{4} \rfloor\ \forall i \in [d-1]$, $l_d = \log n - \sum_{i=1}^{d-1} l_i$
3. $k_i l_i = \Theta(\log n)\ \forall i \in [d-1]$, $k_d = \Theta(\frac{\log n}{\log \log n})$
4. $\delta = \text{poly}(\frac{1}{n})$

We randomly select h_i from corresponding function family $\mathscr{H}_i$ for every $i \in [d]$, and generate our overall hash function $f(x)$ as:

$$f(x) = h_1(x) \circ ... \circ h_d(x),$$

which just simply concatenates all outputs of d k_i-wise δ-dependent functions.

We could visualize this construction as a reversed tree of d+1 layers. Given the sets of balls $S \subseteq [n]$, we will have:

$$\begin{aligned} &\text{layer 0:} && 1 \text{ bin with } n_0 = n \\ &\text{layer 1:} && 2^{l_1} \text{ bins with } n_1 = \frac{n_0}{2^{l_1}} \\ &\text{layer 2:} && 2^{l_1+l_2} \text{ bins with } n_2 = \frac{n_1}{2^{l_1}} \\ & && \text{.........} \\ &\text{layer i:} && 2^{\sum_{j=1}^{i} l_j} \text{ bins with } n_i = \frac{n_{i-1}}{2^{l_i}} \end{aligned}$$

Now, we start in layer 0 where we only have 1 bin containing all n balls. Then, we use function h_1 to hash all $n = n_0$ elements into 2^{l_1} bins in the next layer. Now, if h_1 is a uniformly independent hash function, the expected number of balls in one bin in layer 1 is $\frac{n_0}{2^{l_1}} = n_1$. Then, for each bin in layer 1, we run h_2 on all balls inside this bin and hash them to a unique group of 2^{l_2} bins from $2^{l_1+l_2}$ bins in layer 2. We repeat this step for all 2^{l_1} bins in layer 1. Since we expect each bin in layer 1 to have n_1 balls and we hash these balls to 2^{l_2} bins in layer 2, the expected number of balls in each bin in layer 2 is $\frac{n_1}{2^{l_1}} = n_2$ if function h_2 is uniformly independent. In general, layer i will have $2^{\sum_{j=1}^{i} l_j}$ bins, and we will hash all balls in each bin in layer $i-1$ to 2^{l_i} bins in layer i using hash function h_i. The expected load of each bin in layer i is n_i. Also, we could derive the relation between n_i and n based on previous analysis:

$$n_i = \frac{n_{i-1}}{2^l_i} = \frac{n_{i-2}}{2^{l_i+l_{i-1}}} \Rightarrow n_i = \frac{n}{2^{\sum_{j=0}^{i} l_j}}$$

On the other hand, we could rewrite l_d as

$$l_d = \log n - \sum_{i=1}^{d-1} l_i = \log \frac{n}{2^{\sum_{j=0}^{d-1} l_j}} = \log n_{d-1}$$

3.2 - Step-by-Step Proof of Theorem 3.1

To proof Theorem 3.1, we first need support from following Lemma.

Lemma 3.2: For any $i = 0, ..., d-2, \alpha = \Omega(\frac{1}{\log\log n}), 0 < \alpha_i < 1$, and set $S_i \subseteq [u]$ of size at most $(1+\alpha_i)n_i$,

$$Pr_{h_{i+1} \leftarrow \mathscr{H}_{i+1}} \left[\max_{y \in 0,1^{l_{i+1}}} |h_{i+1}^{-1}(y) \cap S_i| \leq (1+\alpha)(1+\alpha_i)n_{i+1} \right] > 1 - \frac{1}{n^{c+1}}$$

We could understand Lemma 3.2 in following way:

For any bin in layer $i+1$, given a set of elements S_i from a bin in layer i,

$$\text{Maximum Load} \leq \left(1 + \Omega(\frac{1}{\log\log n})\right)(1+\alpha_i)n_{i+1} \text{ with high probability.}$$

Proof of Lemma 3.2:
We first fix $y \in \{0,1\}^{l_{i+1}}$, let $X = |h_{i+1}^{-1}(y) \cap S|$. Assume without loss of generosity, $|S_i| \geq \lfloor (1+\alpha_i)n_i \rfloor$. If $|S_i| < \lfloor (1+\alpha_i)n_i \rfloor$, we could enlarge S_i by adding dummy elements. Then, we introduce indicator random variables X_i such that:

$$X_i = \begin{cases} 1 & \text{if element } j \in S_i \text{ is hashed into bin y by } h_{i+1} \\ 0 & \text{Otherwise} \end{cases}$$

Thus, we could rewrite X as the sum of X_i and, if h_{i+1} is uniformly independent, the expectation of X will become:

$$E[X] = \sum_{j}^{|S_i|} E[X_j] = \sum_{j}^{|S_i|} \frac{1}{2^{l_{i+1}}} = \frac{|S_i|}{2^{l_{i+1}}}$$

However, since our function h_{i+1} is k_{i+1}-wise δ-dependent, $X =$ sum of $|S_i|$ k_{i+1}-wise δ-dependent random variables. Then, we could directly apply **Lemma 2.3.2** with $k = \frac{k_{i+1}}{2}$ and $\mu = E[X] = \frac{|S_i|}{2^{l_{i+1}}}$,

$$\begin{aligned} Pr[X > (1+\alpha)\mu] &\leq 2\left(\frac{|S_i|k_{i+1}}{(\alpha\mu)^2}\right)^{\frac{k_{i+1}}{2}} + \delta\left(\frac{|S_i|}{\alpha\mu}\right)^{\frac{k_{i+1}}{2}} \\ &= 2\left(\frac{|S_i|k_{i+1}}{\alpha^2\frac{|S_i|^2}{2^{2l_{i+1}}}}\right)^{\frac{k_{i+1}}{2}} + \delta\left(\frac{|S_i|}{\alpha\frac{|S_i|}{2^{l_{i+1}}}}\right)^{\frac{k_{i+1}}{2}} \quad \text{since } \mu = \frac{|S_i|}{2^{l_{i+1}}} \\ &= 2\left(\frac{2^{2l_{i+1}}k_{i+1}}{\alpha^2|S_i|}\right)^{\frac{k_{i+1}}{2}} + \delta\left(\frac{2^{2l_{i+1}}}{\alpha}\right)^{\frac{k_{i+1}}{2}} \end{aligned}$$

Now, we will try to upper bound each item in this expression.

1. Notice that, in our construction,
   (a) $l_{i+1} = \lfloor \frac{\log n_i}{4} \rfloor \leq \frac{\log n_i}{4}$
   (b) $|S| \geq (1+\alpha_i)n_i - 1 \geq n_i$, and
   (c) $\alpha = \Omega(\frac{1}{\log\log n})$

Then, for the first item, we could derive:

$$\begin{aligned}
2\left(\frac{2^{2l_{i+1}}k_{i+1}}{\alpha^2|S_i|}\right)^{\frac{k_{i+1}}{2}} &\leq 2\left(\frac{2^{\frac{\log n_i}{2}}k_{i+1}}{\alpha^2|S_i|}\right)^{\frac{k_{i+1}}{2}} && \text{since (a)}\\
&= 2\left(\frac{\sqrt{n_i}k_{i+1}}{\alpha^2|S_i|}\right)^{\frac{k_{i+1}}{2}} && \text{since } 2^{\frac{\log n_i}{2}} = \sqrt{n_i}\\
&\leq 2\left(\frac{\sqrt{n_i}k_{i+1}}{\alpha^2 n_i}\right)^{\frac{k_{i+1}}{2}} && \text{since (b)} \to \frac{1}{S_i} \leq \frac{1}{n_i}\\
&= 2\left(\frac{k_{i+1}}{\alpha^2\sqrt{n_i}}\right)^{\frac{k_{i+1}}{2}}
\end{aligned}$$

Note that $2^{2l_{i+1}} \leq 2^{\frac{\log n_i}{2}} = \sqrt{n_i} \to \frac{1}{\sqrt{n_i}} \leq \frac{1}{2^{2l_{i+1}}}$, then

$$\begin{aligned}
&\leq 2\left(\frac{k_{i+1}}{\alpha^2 2^{2l_{i+1}}}\right)^{\frac{k_{i+1}}{2}}\\
&= 2\left(\frac{k_{i+1}^{k_{i+1}/2}}{\alpha^{k_{i+1}}2^{k_{i+1}l_{i+1}}}\right)
\end{aligned}$$

by construction, we set $k_{i+1}l_{i+1} = \log n^c \in \Theta(\log n)$, then

$$\begin{aligned}
&= 2\left(\frac{k_{i+1}^{k_{i+1}/2}}{\alpha^{k_{i+1}}2^{\log n^c}}\right)\\
&= 2\left(\frac{k_{i+1}}{\alpha^2}\right)^{k_{i+1}/2} \times \frac{1}{n^c}
\end{aligned}$$

Set $k_{i+1} = \frac{k_{i+1}l_{i+1}}{l_{i+1}} = \frac{4c\log n}{\log n_{i+1}}$ and $\alpha = \frac{2n}{\sqrt{k_{i+1}}} = \Omega(\frac{1}{\log\log n})$

$$\begin{aligned}
&= 2\left(\frac{k_{i+1}}{4n^2 \times \frac{1}{k_{i+1}}}\right)^{\frac{2c\log n}{\log n_{i+1}}} \times \frac{1}{n^c}\\
&= 2\left(\frac{1}{4n^2}\right)^{\frac{2c\log n}{\log n_{i+1}}} \times \frac{1}{n^c}\\
&\leq 2 \times \frac{1}{4n^2} \times \frac{1}{n^c} \quad \text{since } \frac{\log n}{\log n_i} \geq 1\\
&= \frac{1}{2n^{c+2}}
\end{aligned}$$

2. Notice that $\delta = \text{poly}(\frac{1}{n})$.Then, for the second item, we could derive:

$$\begin{aligned}
\delta\left(\frac{2^{2l_{i+1}}}{\alpha}\right)^{\frac{k_{i+1}}{2}} &= \delta\left(\frac{2^{k_{i+1}l_{i+1}}}{\alpha^k_{i+1}}\right)\\
&= \delta\left(\frac{2^{\log n^c}}{\alpha^k_{i+1}}\right) && \text{since } k_{i+1}l_{k+1} = \log n^c\\
&\leq \delta n^c
\end{aligned}$$

$$\begin{aligned}\text{Set } \delta = &\frac{1}{2n^{2c+2}}, \text{ then}\\ &\leq \frac{1}{2n^{2c+2}} \times n^c\\ &= \frac{1}{2n^{c+2}}\end{aligned}$$

Thus, we will get:

$$\begin{aligned}Pr\left[X > (1+\alpha)(1+\alpha_i)n_{i+1}\right] &= Pr\left[X > (1+\alpha)(1+\alpha_i)\frac{n_i}{2^{l_{i+1}}}\right] \quad \text{since } n_{i+1} = \frac{n_i}{2^{l_{i+1}}}\\ &\leq Pr\left[X > (1+\alpha)\frac{|S_i|}{2^{l_{i+1}}}\right] \quad \text{since } |S_i| \leq (1+\alpha_i)n_i\\ &= Pr\left[X > (1+\alpha)\mu\right] \quad \text{since } \mu = \frac{|S_i|}{2^{l_{i+1}}}\\ &\leq \frac{1}{2n^{c+2}} + \frac{1}{2n^{c+2}} \qquad \text{by upper bound derived above}\\ &= \frac{1}{n^{c+2}}\end{aligned}$$

We have at most $2^{l_{i+1}} \leq n$ different y since $2^{l_{i+1}} \leq 2^{\frac{\log n_{i-1}}{4}} \leq 2^{\log n} = n$. Then, we will apply a union bound on y. Then, we will get Lemma 3.2 by subtract the union bound result from 1. ■

Now we are ready to prove Theorem 3.1. We first start with the proof of the description length and evaluation time.

Proof of Theorem 3.1(Space and Time):

For the layer i, we have $h_i : [u] \to \{0,1\}^{l_i}$ which is a k_i-wise δ-dependent function. Thus, we know $v = 2^{l_i}$. Combining the fact that $u = \text{poly}(u)$ and $\delta = \text{poly}(\frac{1}{n})$, by Corollary 2.1, we know there exists a family of k_i-wise δ-dependent functions that require:

$$\begin{aligned}\textbf{Space} &= O(\log u + k\log v + \log(\frac{1}{\delta}))\\ &= O(\log \text{poly}(n) + k_i \log 2^{l_i} + \log(\frac{1}{\text{poly}(\frac{1}{n})}))\\ &= O(\log \text{poly}(n) + k_i l_i + \log(\frac{1}{\text{poly}(\frac{1}{n})})\\ &= O(\log \text{poly}(n) + \Theta(\log n) + \log(\frac{1}{\text{poly}(\frac{1}{n})})\\ &= O(\log n)\\ \textbf{Time} &= O(\log u + k\log v + \log(\frac{1}{\delta}))\\ &= O(\log n) \text{ by the same analysis}\end{aligned}$$

The overall hashing function $f(x)$ generates output by calculating d intermediate hash functions $h_1, h_2, ..., h_d$. Thus, with an appropriate choice of $d = O(\log\log n)$, we could describe $f(x)$ using $O(\log n \log\log n)$ bits and compute $f(x)$ in $O(\log n \log\log n)$ time. ■

Secondly, we need to limit the maximum load. We will use Lemma 3.2 to limit the maximum load of layer $i+1$ for $i \in [d-1]$ so that we could analyze the maximum load in the last layer.

Proof of Theorem 3.1(Maximum Load):

We fix a set $S \subseteq [u]$ of size n. We inductively argue that, for $\alpha = \Omega(\frac{1}{\log\log n})$:

$$Pr\left[\text{Maximum load of layer i per bin} \leq (1+\alpha)^i n_i\right] \geq 1 - \frac{i}{n^{c+1}}$$

The base case here is $i = 0$. In layer 0, we have only 1 bin with $n_0 = n = (1+\alpha)^0 n = (1+\alpha)^i n_i$. Thus, the claim holds for $i = 0$. Now, we assume the claim holds for layer i. We apply Lemma 3.2 on each bin of layer i with $(1+\alpha_i) = (1+\alpha)^i$. Based on our interpretation and the fact that there should be no more than n bins in one layer, we know:

$$Pr\left[\text{number of balls in one bin of layer } i+1 \leq (1+\alpha)^{i+1} n_{i+1}\right] \geq 1 - \frac{1}{n^{c+1}}$$
$$\Rightarrow Pr\left[\text{number of balls in one bin of layer } i+1 > (1+\alpha)^{i+1} n_{i+1}\right] \leq \frac{1}{n^{c+1}}$$
$$\Rightarrow_{\text{Union Bound}} Pr\left[\exists \text{ a bin whose load in layer } i+1 > (1+\alpha)^{i+1} n_{i+1}\right] \leq \frac{n}{n^{c+1}}$$
$$\Rightarrow Pr\left[\nexists \text{ a bin whose load in layer } i+1 > (1+\alpha)^{i+1} n_{i+1}\right] \geq 1 - \frac{n}{n^{c+1}}$$
$$\geq 1 - \frac{i+1}{n^{c+1}}$$
$$\Rightarrow Pr\left[\text{Maximum Load of layer } i+1 \text{ is } (1+\alpha)^{i+1} n_{i+1}\right] \geq 1 - \frac{i+1}{n^{c+1}}$$

Thus, the claim holds in inductive case.

Now, we want to upper bound n_{d-1} which is the expected load of layer $d-1$. By the claim of induction, we know that, with probability at least $1 - \frac{d-1}{n^{c+1}}$, the maximum load of layer $d-1$ is $(1+\alpha)^{d-1} n_{d-1} \leq 2n_{d-1}$ for appropriate $d = O(\log\log n)$. For every $i \in [d-1]$,

$$l_i = \lfloor \frac{\log n_{i-1}}{4} \rfloor \geq \frac{\log n_{i-1}}{4} - 1$$
$$\Rightarrow n_i = \frac{n_{i-1}}{2^{l_i}} \leq \frac{n_{i-1}}{2^{\frac{\log n_{i-1}}{4}-1}} = 2\frac{n_{i-1}}{2^{\frac{\log n_{i-1}}{4}}} = 2\frac{n_{i-1}}{n_{i-1}^{1/4}} = 2n_{i-1}^{3/4}$$
$$\Rightarrow n_{i-1} \leq 2n_{i-2}^{3/4} \Rightarrow n_i \leq 2(2n_{i-2}^{3/4})^{3/4} = 2^{1+3/4} n_{i-2}^{(3/4)^2}$$
$$\Rightarrow_{induction}\ n_{i-1} \leq 2^{\sum_{j=0}^{i-1}(3/4)^j} n^{(3/4)^i} <= 2^4 n^{(3/4)^i} = 16n^{(3/4)^i}$$

Thus, for an appropiate choice of $d = O(\log\log n)$, it holds

$$n_{d-1} \leq \log n$$

For example, if $d = \log_{3/4}\left(\frac{\log\frac{\log n}{16}}{\log n}\right) = O(\log\log n)$,

$$n_{d-1} \leq 16n^{(3/4)^d} = 16n^{(3/4)^{\log_{3/4}\left(\frac{\log\frac{\log n}{16}}{\log n}\right)}}$$

$$= 16n^{\frac{\log \frac{\log n}{16}}{\log n}}$$
$$= 16n^{\log_n \frac{\log n}{16}}$$
$$= 16 \times \frac{\log n}{16}$$
$$= \log n$$

Thus, we could state, with probability at least $1 - \frac{d-1}{n^{c+1}}$, the maximum load of layer $d-1$ is $(1+d)^{d-1} \leq 2n_{d-1} \leq 2\log n$. By the analysis on n_i and l_d in section 3.1, we know $l_d = \log n_{d-1}$. Thus, elements in each bin in layer $d-1$ are hashed into $2^{l_d} = 2^{\log n_{d-1}} = n_{d-1}$ bins using the function h_d which is k_d-wise δ-dependent, where $k_d = \Omega(\frac{\log n}{\log\log n})$ and with an appropriate choice of $d = O(\log\log n)$.

Therefore, the probability that any $t = \frac{\gamma \log n}{\log\log n} \leq k_d$ elements from layer $d-1$ are hashed into any specific bin in layer d is:

$$\begin{aligned}\binom{2n_{d-1}}{t}\left((\frac{1}{n_{d-1}}^{t} + \delta)\right) &\leq (\frac{2en_{d-1}}{t})^t \left((\frac{1}{n_{d-1}}^{t} + \delta)\right)\\ &= (\frac{2e}{t})^t + \delta\left(\frac{2en_{d-1}}{t}\right)^t\\ &= (\frac{2e\log\log n}{\gamma\log n})^{\frac{\gamma\log n}{\log\log n}} + \delta\left(\frac{2en_{d-1}\log\log n}{\gamma\log n}\right)^{\frac{\gamma\log n}{\log\log n}}\\ &\leq \frac{1}{2n^{c+3}} + \frac{1}{2n^{c+3}}\\ &= \frac{1}{n^{c+3}}\end{aligned}$$

Reason that $(\frac{2e\log\log n}{\gamma\log n})^{\frac{\gamma\log n}{\log\log n}}$ and $\delta\left(\frac{2en_{d-1}\log\log n}{\gamma\log n}\right)^{\frac{\gamma\log n}{\log\log n}}$ are less than $\frac{1}{2n^{c+3}}$ is simple. We will use $(\frac{2e\log\log n}{\gamma\log n})^{\frac{\gamma\log n}{\log\log n}}$ as example:

$$\begin{aligned}(\frac{2e\log\log n}{\gamma\log n})^{\frac{\gamma\log n}{\log\log n}} &= (\frac{2e}{\gamma})^{\frac{\gamma\log n}{\log\log n}}(\log\log n)^{\frac{\gamma\log n}{\log\log n}} \times (\log n)^{-\frac{\gamma\log n}{\log\log n}}\\ &= \eta(\log\log n)^{\frac{\gamma\log n}{\log\log n}} \times (\log n)^{-\frac{\gamma\log n}{\log\log n}} \quad \text{set } \eta = (\frac{2e}{\gamma})^{\frac{\gamma\log n}{\log\log n}}\\ &= \eta(\log\log n)^{\gamma\log_{\log n} n} \times (\log n)^{-\gamma\log_{\log n} n}\\ &= \eta(\log\log n)^{\gamma\log_{\log n} n} \times n^{-\gamma}\\ &= O(n^{-\gamma+1})\end{aligned}$$

For any $\gamma \leq c+4$, we could derive the result above. As we know $k_d = \Omega(\frac{\log n}{\log logn})$ and $t = \frac{\gamma\log n}{\log\log n}$, we know $\gamma \leq \frac{k_d\log\log n}{\log n} = \Omega(1)$. Thus, $\gamma \leq c+4$ is within the limit, and it is possible in this situation.

Thus, we have proved that the probability that any $t \leq k_d$ elements from layer $d-1$ are hashed into a specific bin in layer d is less than $\frac{1}{n^{c+3}}$. There exist at most n^2 such pair of bins exists between layer $d-1$ and layer d. Thus we could get:

$$
\begin{aligned}
&Pr(\text{a bin in layer d with more than t elements}) \leq \frac{1}{n^{c+1}} \\
&\Rightarrow_{\text{Union Bound}} Pr(\exists \text{ a bin in layer d with more than t elements}) \leq \frac{n}{n^{c+1}} \\
&\Rightarrow Pr(\nexists \text{ a bin in layer d with more than t elements}) \geq 1 - \frac{n}{n^{c+1}} \\
&\Rightarrow Pr(\text{Maximum load of layer d is t elements}) \geq 1 - \frac{n}{n^{c+1}} \geq 1 - \frac{1}{n^c}
\end{aligned}
$$

which completes the proof of Theorem 3.1. ■

Since we have proven Theorem 3.1, it implies that with high probability at least $1 - d/n^{c+1} > 1 - 1/n^c$, a randomly selected function from the hash family from our construction has a maximum load of $O(\log n/\log\log n)$ while guaranteeing each function can be described in $o(\log^2 n/\log\log n)$.

4. Extension and Acknowledgement

This section presents some extensional use of this construction, augmentation to this construction as well as acknowledgment. Since we have proven that this construction does provide the benefit it claims, we can employ this construction for storing elements using linear probing. More importantly, it also guarantees the maximum load that all other hash families provide. Existing algorithms put more of their focus on simplicity and fast computation, while this construction gives a much better description length of the hash family while sacrificing the simplicity factor.

4.1 - Augmenting The Construction

We can update the construction to offer $O(\log\log n)$-wise independence without affecting the benefits it offers, which is the description length of the hash family. According to L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder, this is especially useful in the case of any application that involves tail bounds for limited independence. For example, if for a function f built from this construction, we can modify it to be $(f(x) + h(x))$modn easily where $h(x)$ is sampled from a family of $O(loglogn)$-wise independent hash functions.

4.2 - Acknowledge

The construction described in this paper is originally from the paper, titled *Ball and bins: smaller hash families and faster evaluation.* This paper was written to fully understand the construction presented in Section 1.1, which is the study result of L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder.

5. Appendix

This section introduces a few more lemmas and corollaries that we used in the proof of the construction. However, these proofs are not strongly related to our paper, and they are study result of other papers; hence, we are directly introducing them as an appendix.

Lemma 2.3.2.1: Let $S_n \subseteq 0, 1^n$ be a sample space that is ϵ-biased with respect to linear tests of size at most k. Then, the sample space S_n is $((1-2^{-k})\epsilon, k)$-independent(in max form), and $(2^k-1)^{\frac{1}{2}}\epsilon$-away(in L_1 norm) from k-wise independent.

Proof of Lemma 2.3.2.1:
Without loss of generality, for variables $x_1, x_2, ...x_k$, for $\alpha \in \{0,1\}^k$, let p_α be the probability that $x_a = \alpha_i$ for all $1 \leq i \leq k$. For $\beta \in \{0,1\}^k$, let $\phi_\beta(\alpha) = (-1)^{\sum_{i=1}^k \alpha_i\beta_i}$. Then the discrete Fourier transformation of the sequence p_α is defined by

$$c_\beta = \sum_\alpha \phi_\beta(\alpha)p_\alpha$$

If $\beta \neq 0$, then this is exactly the same as the bias of the linear test given by β, and hence in the case of $|c_\beta| \leq \epsilon$, while $c_0 = 1$, by the standard Fourier analysis we have

$$p_\alpha = 2^{-k}\sum_\beta \phi_\beta(\alpha)c_\beta$$

and we know that:

$$\sum_\alpha p_\alpha^2 = 2^{-k}\sum_\beta c_\beta^2$$

Now with these information, we have:

$$|p_\alpha - 2^{-k}| = 2^{-k}\sum_\beta \phi_\beta(\alpha)c_\beta \leq (1-2^{-k})\epsilon,$$

This completes the first part of the lemma. The prove the second part, we do it similarly, and apply Cauchy-Schwarz inequality, then we will obtain the following:

$$\sum_\alpha |p_\alpha - 2^{-k}| \leq 2^{k/2}(\sum_\alpha(p_\alpha - 2^{-k})^2)^{1/2} = 2^{k/2}(2^{-k}\sum_{\beta\neq 0} c_\beta^2)^{1/2} \leq (2^k-1)^{1/2}\epsilon.$$

This completes the second part of the Lemma. Hence we have completed the proof of Lemma 2.3.2.1. ■

In particular, Lemma 2.3.2.1 will imply Corollary 2.3.2.2. It's clear to see how this following Corollary is a transformation of Lemma 2.3.2.1 and hence, we will not show the steps.

Corollary 2.3.2.2: Let $S_n \in \{0,1\}^n$ be a sample space that is ϵ-biased with respect to linear tests. Then, for every k, the sample space S_n is $((1-2^{-k})\epsilon, k)$-independent (in max norm), and $(2^k-1)^{1/2}\epsilon$-away in (L_1 norm) from k-wise independence.

6. References

[1] L. E. Celis, O. Reingold, G. Segev and U. Wieder, ”Balls and Bins: Smaller Hash Families and Faster Evaluation,” 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, Palm Springs, CA, 2011, pp. 599-608, doi: 10.1109/FOCS.2011.49.

[2] N. Alon, O. Goldreich, J. Hastad, and R. Peralta. Simple construction of almost kwise independent random variables. Random Structures and Algorithms, 3(3):289–304, 1992.

[3] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, pages 276–287, 1994.