Linear Probing with Constant Independence

Anna Pagh* Rasmus Pagh* Milan Ružić*

ABSTRACT

Hashing with linear probing dates back to the 1950s, and is among the most studied algorithms. In recent years it has become one of the most important hash table organizations since it uses the cache of modern computers very well. Unfortunately, previous analyses rely either on complicated and space consuming hash functions, or on the unrealistic assumption of free access to a truly random hash function. Already Carter and Wegman, in their seminal paper on universal hashing, raised the question of extending their analysis to linear probing. However, we show in this paper that linear probing using a pairwise independent family may have expected *logarithmic* cost per operation. On the positive side, we show that 5-wise independence is enough to ensure constant expected time per operation. This resolves the question of finding a space and time efficient hash function that provably ensures good performance for linear probing.

Categories and Subject Descriptors

E.2 [**Data**]: Data Storage Representations; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Performance, Theory

Keywords

Hashing, Linear Probing

*Computational Logic and Algorithms Group, IT University of Copenhagen, Rued Langgaards Vej 7, 2300 København S, Denmark.

1. INTRODUCTION

Hashing with linear probing is perhaps the simplest algorithm for storing and accessing a set of keys that obtains nontrivial performance. Given a hash function h, a key x is inserted in an array by searching for the first vacant array position in the sequence $h(x), h(x) + 1, h(x) + 2, \ldots$ (Here, addition is modulo r, the size of the array.) Retrieval of a key proceeds similarly, until either the key is found, or a vacant position is encountered, in which case the key is not present in the data structure. Deletions can be performed by moving elements back in the probe sequence in a greedy fashion (ensuring that no key x is moved beyond $h(x)$), until a vacant array position is encountered.

Linear probing dates back to 1954, but was first analyzed by Knuth in a 1963 memorandum [7] now considered to be the birth of the area of analysis of algorithms [10]. Knuth's analysis, as well as most of the work that has since gone into understanding the properties of linear probing, is based on the assumption that h is a truly random function. In 1977, Carter and Wegman's notion of universal hashing [2] initiated a new era in the design of hashing algorithms, where explicit and efficient ways of choosing hash functions replaced the unrealistic assumption of complete randomness. In their seminal paper, Carter and Wegman state it as an open problem to "Extend the analysis to [...] double hashing and open addressing."[1]

1.1 Previous results using limited randomness

The first analysis of linear probing relying only on limited randomness was given by Siegel and Schmidt in [11, 13]. Specifically, they show that $O(\log n)$-wise independence is sufficient to achieve essentially the same performance as in the fully random case. (We use n to denote the number of keys inserted into the hash table.) Another paper by Siegel [12] shows that evaluation of a hash function from a $O(\log n)$-wise independent family requires time $\Omega(\log n)$ unless the space used to describe the function is $n^{\Omega(1)}$. A family of functions is given that achieves space usage n^ϵ and constant time evaluation of functions, for any $\epsilon > 0$. However, this result is only of theoretical interest since the associated constants are very large (and growing exponentially with $1/\epsilon$).

[1]Nowadays the term "open addressing" refers to any hashing scheme where the data structure is an array containing only keys and empty locations. However, Knuth used the term to refer to linear probing in [7], and since it is mentioned here together with the double hashing probe sequence, we believe that it refers to linear probing.

A potentially more practical method is the "split and share" technique described in [4]. It can be used to achieve characteristics similar to those of linear probing, still using space n^ϵ, for any given $\epsilon > 0$. The idea is to split the set of keys into many subsets of roughly the same size, and simulate full randomness on each part. Thus, the resulting solution would be a *collection* of linear probing hash tables.

A significant drawback of both methods above, besides a large number of instructions for function evaluation, is the use of random accesses to the hash function description. The strength of linear probing is that for many practical parameters, almost all lookups will incur only a single cache miss. Performing random accesses while computing the hash function value may destroy this advantage.

1.2 Our results

We show in this paper that linear probing using a pairwise independent family may have expected *logarithmic* cost per operation. Specifically, we resolve the open problem of Carter and Wegman by showing that linear probing insertion of n keys in a table of size $2n$ using a function of the form $x \mapsto ((ax + b) \bmod p) \bmod 2n$, where $p = 4n + 1$ is prime and we randomly choose $a \in [p]\backslash\{0\}$ and $b \in [p]$, requires $\Omega(n \log n)$ insertion steps in expectation for a worst case insertion sequence (chosen independently of a and b). Since the total insertion cost equals the total cost of looking up all keys, the expected average time to look up a key in the resulting hash table is $\Omega(\log n)$. The main observation behind the proof is that if a is the multiplicative inverse (modulo p) of a small integer m, then inserting a certain set that consists of two intervals has expected cost $O(n^2/m)$.

On the positive side, we show that *5-wise independence* is enough to ensure constant expected time per operation, for load factor $\alpha = n/r$ bounded away from 1. Our proof is based on a new way of bounding the cost of linear probing operations, by counting intervals in which "many" probe sequences start.

Our analysis gives a bound of $O(1 + \frac{1}{(1-\alpha)^3})$ expected time per operation at load factor α. This implies a bound of $O(1 + \frac{1}{(1-\alpha)^2})$ expected time on average for successful searches. These bounds are a factor $\Omega(\frac{1}{1-\alpha})$ higher than for linear probing with full independence. However, this situation changes if r is a power of 2 and the probe sequence slightly changes to

$$h(x), h(x) \oplus 1, h(x) \oplus 2, h(x) \oplus 3, \ldots$$

where $\oplus$ denotes bitwise exclusive or. This probe sequence is arguably even more cache friendly than classical linear probing if we assume that memory block boundaries are at powers of 2. In fact, we analyze a slightly more general class of open addressing methods called *blocked probing*, which also includes a special kind of bidirectional linear probing. For this class we get the same dependence on α (up to constant factors) as for full independence, again using only 5-wise independent hash functions. A particularly precise analysis of successful searches is conducted, showing that the expected number of probes made during a search for a random element in the table is less than $1 + \frac{2}{1-\alpha}$.

1.3 Significance

Several recent experimental studies [1, 5, 9] have found linear probing to be the fastest hash table organization for moderate load factors (30-70%). While linear probing operations are known to require more instructions than those of other open addressing methods, the fact that they access an interval of array entries means that linear probing works very well with modern architectures for which sequential access is much faster than random access (assuming that the elements we are accessing are each significantly smaller than a cache line, or a disk block, etc.). However, the hash functions used to implement linear probing in practice are heuristics, and there is no known theoretical guarantee on their performance. Since linear probing is particularly sensitive to a bad choice of hash function, Heileman and Luo [5] advice *against* linear probing for general-purpose use. Our results imply that simple and efficient hash functions, whose description can be stored in CPU registers, can be used to give provably good performance.

2. PRELIMINARIES

2.1 Notation and definitions

Define $[x] = \{0, 1, \ldots, x - 1\}$. Throughout this paper S denotes a subset of some universe U, and h will denote a function from U to $R = [r]$. We denote the elements of S by $\{x_1, x_2, \ldots, x_n\}$, and refer to the elements of S as *keys*. We let $n = |S|$, and $\alpha = n/r$. For $Q \subseteq R$ we define $|h(S) \cap_m Q| = |\{x \in S \mid h(x) \in Q\}|$. For any integers x and a define $x \ominus a = x - (x \bmod a)$. The function $x \mapsto x - \lfloor x \rfloor$ is denoted by $\mathrm{frac}(x)$.

A family $\mathcal{H}$ of functions from U to R is k-wise independent if for any k distinct elements $x_1, \ldots, x_k \in U$ and h chosen uniformly at random from $\mathcal{H}$, the random variables $h(x_1), \ldots, h(x_k)$ are independent[2]. We refer to the variable

$$\bar{\alpha}_{\mathcal{H}} = n \max_{x \in U,\, \rho \in R} \Pr_{h \in \mathcal{H}}\{h(x) = \rho\}$$

as the *maximum load* of $\mathcal{H}$. When the hash function family in question is understood from the context, we omit the subscript of $\bar{\alpha}$. If $\mathcal{H}$ distributes hash function values of all elements of U uniformly on R, we will have $\bar{\alpha} = \alpha$, and in general $\bar{\alpha} \geq \alpha$.

Let $Q \subseteq R$. By $a + Q$ we denote the translated set

$$\{(a + y) \bmod r \mid y \in Q\} \ ,$$

and we define $a - Q$ analogously. An *interval* (modulo r) is a set of the form $a + [b]$, for integers a and b. We will later use sets of the form $h(x) + Q$, for a fixed x and with Q being an interval.

We introduce a function which simplifies statements of some upper bounds — those in which constant factors are explicit. Define $T(\bar{\alpha})$ as the function over the domain $(0, 1)$ with the value given by

$$T(\bar{\alpha}) = \begin{cases} \frac{5.2\bar{\alpha}}{(1-\bar{\alpha})^2} + \frac{1}{3} \ , & \bar{\alpha} \geq \frac{1}{3} \\ \frac{2.5\bar{\alpha}}{(1-\bar{\alpha})^4} \ , & \bar{\alpha} < \frac{1}{3} \end{cases} \ .$$

It can be checked that $T(\bar{\alpha}) < \frac{5.7\bar{\alpha}}{(1-\bar{\alpha})^2}$.

[2]We note that in some papers, the notion of k-wise independence is stronger in that it is required that function values are uniformly distributed in R. However, some interesting k-wise independent families have a slightly nonuniform distribution, and we will provide analysis for such families as well.

2.2 Hash function families

Carter and Wegman [15] observed that the family of degree $k-1$ polynomials in any finite field is k-wise independent. Specifically, for any prime p we may use the field defined by arithmetic modulo p to get a family of functions from $[p]$ to $[p]$ where a function can be evaluated in time $O(k)$ on a RAM, assuming that addition and multiplication modulo p can be performed in constant time. To obtain a smaller range $R = [r]$ we may map integers in $[p]$ down to R by a modulo r operation. This of course preserves independence, but the family is now only close to uniform. Specifically, the maximum load $\bar{\alpha}$ for this family is in the range $[\alpha; (1+n/p)\alpha]$. By choosing p much larger than n we can make $\bar{\alpha}$ arbitrarily close to α.

A recently proposed k-wise independent family of Thorup and Zhang [14] has uniformly distributed function values in $[r]$, and thus $\bar{\alpha} = \alpha$. From a theoretical perspective (ignoring constant factors) it is inferior to Siegel's highly independent family [12], since the evaluation time depends on k and the space usage is the same (though the dependence of ϵ is better). We mention it here because it is the first construction that makes k-wise independence truly competitive with popular heuristics, for small $k > 3$, in terms of evaluation time. In practice, the space usage can be kept so small that it does not matter. The construction for 4-wise independence has been shown to be particularly efficient. Though this is not stated in [14], it is not hard to verify that the same construction in fact gives 5-wise independence, and thus our analysis will apply.

2.3 A probabilistic lemma

Here we state a lemma that is essential for our upper bound results, described in sections 4 and 5. It gives an upper bound on the probability that an interval around a particular hash function value contains the hash function values of "many" keys. The proof is similar to the proof of [8, Lemma 4.19].

LEMMA 1. *Let $S \subseteq U$ be a set of size n, and $\mathcal{H}$ a 5-wise independent family of functions from U to R with maximum load at most $\bar{\alpha} < 1$. If h is chosen uniformly at random from $\mathcal{H}$, then for any $Q \subset R$ of size q, and any fixed $x \in U \setminus S$,*

$$Pr\{|h(S) \cap_m (h(x)+Q)| \geq \bar{\alpha}q + d\} < \frac{3(\bar{\alpha}q)^2 + \bar{\alpha}q}{d^4} \ .$$

PROOF. We will show a stronger statement, namely that for any $\rho \in R$ the bound holds if we choose h uniformly at random from the subfamily $\mathcal{H}_\rho = \{h \in \mathcal{H} \mid h(x) = \rho\}$. Notice that $\mathcal{H}_\rho$ is 4-wise independent on $U \backslash \{x\}$, and that the distribution of function values is identical to the distribution when h is chosen from $\mathcal{H}$.

Let $p_i = \Pr\{h(x_i) \in (h(x)+Q)\}$, and consider the random variables

$$X_i = \begin{cases} 1 - p_i, & \text{if } h(x_i) \in h(x) + Q \\ -p_i, & \text{otherwise} \end{cases} \ .$$

Let $X = \sum_i X_i$ and observe that

$$|h(S) \cap_m (h(x)+Q)| = X + \sum_i p_i \leq X + \bar{\alpha}q \ .$$

The last inequality above is by the definition of maximum load. So to prove the lemma it suffices to bound $\Pr\{X \geq d\}$. We will use the 4th moment inequality

$$\Pr\{X \geq d\} \leq \mathrm{E}(X^4)/d^4 \ .$$

Clearly, $\mathrm{E}(X_i) = 0$ for any i, and the variables $X_1, \ldots, X_n$ are 4-wise independent. Therefore we have $\mathrm{E}(X_{i_1}X_{i_2}X_{i_3}X_{i_4}) = 0$ unless $i_1 = i_2 = i_3 = i_4$ or (i_1, i_2, i_3, i_4) contains 2 numbers, both of them exactly twice. This means that

$$\begin{aligned} \mathrm{E}(X^4) &= \sum_{1 \leq i_1, i_2, i_3, i_4 \leq n} \mathrm{E}(X_{i1}X_{i2}X_{i3}X_{i4}) \\ &\leq \sum_{1 \leq i \leq n} \mathrm{E}(X_i^4) + \sum_{1 \leq i < j \leq n} \tbinom{4}{2} \mathrm{E}(X_i^2)\mathrm{E}(X_j^2). \end{aligned}$$

The first sum can be bounded as follows:

$$\begin{aligned} \sum_i \mathrm{E}(X_i^4) &= \sum_i (p_i(1-p_i)^4 + (1-p_i)p_i^4) \\ &= \sum_i p_i(1-p_i)((1-p_i)^3 + p_i^3) \\ &< \sum_i p_i \leq \bar{\alpha}q \ . \end{aligned}$$

The second sum is:

$$\begin{aligned} \sum_{1 \leq i < j \leq n} 6(p_i(1-p_i))(p_j(1-p_j)) &< 3 \sum_{1 \leq i, j \leq n} p_i p_j \\ &= 3(\sum_i p_i)^2 \leq 3(\bar{\alpha}q)^2 \ . \end{aligned}$$

In conclusion we have

$$\Pr\{X \geq d\} \leq \mathrm{E}(X^4)/d^4 < \frac{3(\bar{\alpha}q)^2 + \bar{\alpha}q}{d^4},$$

finishing the proof. □

3. PAIRWISE INDEPENDENCE

In this section we show that pairwise independence is not sufficient to ensure good performance for linear probing: Logarithmic time per operation is needed for a worst-case set. This complements our upper bounds for 5-wise (and higher) independence. We will consider two pairwise independent families: The first one is a very commonly used hash function family. The latter family is similar to the first, except that we have ensured function values to be uniformly distributed in R. To lower bound the cost of linear probing we use the following lemma.

LEMMA 2. *Suppose that n keys are inserted in a linear probing hash table of size r with probe sequences starting at $i_1, \ldots, i_n$, respectively. Further, suppose that $I_1, \ldots, I_\ell$ is any set of intervals (modulo r) such that we have the multiset equality $\bigcup_j \{i_j\} = \bigcup_j I_j$. Then the total number of steps to perform the insertions is at least*

$$\sum_{1 \leq j_1 < j_2 \leq \ell} |I_{j_1} \cap I_{j_2}|^2/2 \ .$$

PROOF. We proceed by induction on ℓ. Since the number of insertion steps is independent of the order of insertions, we may assume that the insertions corresponding to I_ℓ occur last. By the induction hypothesis, the total number of steps to do all preceding insertions is at least

$$\sum_{1 \leq j_1 < j_2 \leq \ell - 1} |I_{j_1} \cap I_{j_2}|^2/2 \ .$$

Let S_j denote the set of keys corresponding to I_j. For any $j < \ell$, and any $x \in S_\ell$ with probe sequence starting in $I_j \cap I_\ell$, the insertion of x will pass all keys of S_j with probe sequences starting in $I_j \cap I_\ell$ that have hash value $h(x)$ or above (mod r). This means that at least $|I_j \cap I_\ell|^2/2$ steps are used during the insertion of the keys of S_ℓ to pass locations occupied by keys of S_j. Summing over all $j < \ell$ and adding to the bound for the preceding insertions finishes the induction step. $\square$

3.1 Linear congruential hash functions

We first consider the following family of functions, introduced by Carter and Wegman [2] as a first example of a universal family of hash functions:

$$\mathcal{H}(p,r) = \{x \mapsto ((ax+b) \bmod p) \bmod r \mid 0 < a < p,\ 0 \le b < p\}$$

where p is any prime number and $r \le p$ is any integer. Functions in $\mathcal{H}(p,r)$ map integers of $[p]$ to $[r]$.

THEOREM 1. *For $r = \lceil p/2 \rceil$ there exists a set $S \subseteq [p]$, $|S| \le r/2$, such that the expected cost of inserting the elements of S in a linear probing hash table of size r using a hash function chosen uniformly at random from $\mathcal{H}(p,r)$ is $\Omega(r \log r)$.*

PROOF. We give a randomized construction of S, and show that when choosing h at random from $\mathcal{H}(p,r)$ the expected total insertion cost for the keys of S is $\Omega(r \log r)$. This implies the existence of a fixed set S with at least the same expectation for random $h \in \mathcal{H}(p,r)$. Specifically, we subdivide $[p]$ into 8 intervals $U_1,\ldots,U_8$, such that $\bigcup_i U_i = [p]$ and $r/4 \ge |U_i| \ge r/4 - 1$ for $i = 1,\ldots,8$, and let S be the union of two of the sets $U_1,\ldots,U_8$ chosen at random (without replacement). Note that $|S| \le r/2$, as required.

Consider a particular function $h \in \mathcal{H}(p,r)$ and the associated values of a and b. Let $\hat{h}(x) = (ax+b) \bmod p$, and let m denote the unique integer in $[p]$ such that $am \bmod p = 1$ (i.e., $m = a^{-1}$ in GF(p)). Since $\hat{h}$ is a permutation on $[p]$, the sets $\hat{h}(U_i)$, $i = 1,\ldots,8$, are disjoint. We note that for any x, $\hat{h}(x+m) = (\hat{h}(x)+1) \bmod p$. Thus, for any k, $\hat{h}(\{x, x+m, x+2m, \ldots, x+km\})$ is an interval (modulo p) of length $k+1$. This implies that for all i there exists a set $\hat{L}_i$ of m disjoint intervals such that $\hat{h}(U_i) = \bigcup_{I \in \hat{L}_i} I$. Similarly, for all i there exists a set L_i of at most $m+1$ intervals (not necessarily disjoint) such that we have the multiset equality $h(U_i) = \bigcup_{I \in L_i} I$. Since all intervals in $\bigcup_i \hat{L}_i$ are disjoint, an interval in $\bigcup_i L_i$ can intersect at most two other intervals in $\bigcup_i L_i$. We now consider two cases:

1. Suppose there is some i such that
$$\sum_{I_1, I_2 \in L_i, I_1 \ne I_2} |I_1 \cap I_2| \ge r/16 \ .$$
Then with constant probability $U_i \subseteq S$, and we apply the bound of Lemma 2. The sum is minimized if all $O(m)$ nonzero intersections have the same size, $\Omega(r/m)$. Thus Lemma 2 implies that the number of insertion steps is $\Omega(r^2/m)$.

2. Now suppose that for all i,
$$\sum_{I_1, I_2 \in L_i, I_1 \ne I_2} |I_1 \cap I_2| < r/16 \ .$$
Note that any value in $[r-1]$ is contained in exactly two intervals of $\bigcup_i L_i$, and by the assumption at most half occur in two intervals of L_i for some i. Thus there exist i_1, i_2, $i_1 \ne i_2$, such that $|h(U_{i_1}) \cap h(U_{i_2})| = \Omega(r)$. With constant probability we have $S = U_{i_1} \cup U_{i_2}$. We now apply Lemma 2. Consider just the terms in the sum of the form $|I_1 \cap I_2|^2/2$, where $I_1 \in L_{i_1}$ and $I_2 \in L_{i_2}$. As before, this sum is minimized if all $O(m)$ intersections have the same size, $\Omega(r/m)$, and we derive an $\Omega(r^2/m)$ lower bound on the number of insertion steps.

For a random $h \in \mathcal{H}(p,r)$, m is uniformly distributed in $\{1,\ldots,p\}$ (the map $a \mapsto a^{-1}$ is a permutation of $\{1,\ldots,p\}$). Therefore, the expected total insertion cost is:

$$\Omega\left(\frac{1}{p}\sum_{m=1}^{p} r^2/m\right) = \Omega\left(\frac{r^2}{p}\log p\right) = \Omega(r \log r) \ .$$

$\square$

3.2 Example with uniform distribution

One might wonder if the lower bound shown in the previous section also holds if the hash function values are uniformly distributed in R. We slightly modify $\mathcal{H}(p,r)$ to remain pairwise independent and also have uniformly distributed function values. Let $\hat{p} = \lceil p/r \rceil r$, and define a function g as follows: $g(y,\hat{y}) = \hat{y}$ if $\hat{y} \ge p$, and $g(y,\hat{y}) = y$ otherwise. For a vector v let v_i denote the $i+1$st component (indexes starting with zero). We define:

$$\mathcal{H}^*(p,r) = \{x \mapsto g((ax+b) \bmod p, v_x) \bmod r \mid 0 \le a < p,\ 0 \le b < p,\ v \in [\hat{p}]^p\}$$

LEMMA 3 (PAIRWISE INDEPENDENCE). *For any pair of distinct values $x_1, x_2 \in [p]$, and any $y_1, y_2 \in [r]$, if h is chosen uniformly at random from $\mathcal{H}^*(p,r)$, then*

$$Pr\{h(x_1) = y_1 \wedge h(x_2) = y_2\} = 1/r^2 \ .$$

PROOF. We will show something stronger than claimed, namely that the family

$$\mathcal{H}^{**} = \{x \mapsto g((ax+b) \bmod p, v_x) \mid 0 \le a < p,\ 0 \le b < p,\ v \in [\hat{p}]^p\}$$

is pairwise independent and has function values uniformly distributed in $[\hat{p}]$. Since r divides $\hat{p}$ this will imply the lemma. Pick any pair of distinct values $x_1, x_2 \in [p]$, and consider a random function $h \in \mathcal{H}^{**}$. Clearly, v_{x_1} and v_{x_2} are uniform in $[\hat{p}]$ and independent. Also, it follows by standard arguments [2] that $(ax_1+b) \bmod p$ and $(ax_2+b) \bmod p$ are uniform in $[p]$ and independent. We can think of the definition of $h(x)$ as follows: The value is v_x unless $v_x \in [p]$, in which case we substitute v_x for another random value in $[p]$, namely $(ax+b) \bmod p$. It follows that hash function values are uniformly distributed, and pairwise independent. $\square$

COROLLARY 1. *Theorem 1 holds also if we replace $\mathcal{H}(p,r)$ by $\mathcal{H}^*(p,r)$. In particular, pairwise independence is not a sufficient condition for linear probing to have expected constant cost per operation.*

PROOF. Consider the parameters a, b, and v of a random function in $\mathcal{H}^*(p,r)$. Since $r = \lceil p/2 \rceil$ we have $\hat{p} = p+1$, and $(p/\hat{p})^p > 1/4$. Therefore, with constant probability it holds that $a \ne 0$ and $v \in [p]^p$. Restricted to functions satisfying this, the family $\mathcal{H}^*(p,r)$ is identical to $\mathcal{H}(p,r)$. Thus, the lower bound carries over (with a smaller constant). By Lemma 3, $\mathcal{H}^*$ is pairwise independent with uniformly distributed function values. □

We remark that the lower bound is tight. A corresponding $O(n \log n)$ upper bound can be shown by applying the analysis of Section 4.1 and using Chebychev's inequality to bound the probability of a fully loaded interval, rather than using the 4th moment inequality as in Lemma 1.

4. 5-WISE INDEPENDENCE

We want to bound the cost of a given operation (insertion, deletion, or lookup of a key x) performed when the hash table contains the set S of keys. It is well known that for linear probing, the set P of occupied table positions depends only on the set S and the hash function, independently of the sequence of insertions and deletions performed. An upper bound for the cost of any operation on x is

$$O(1 + \max\{l \mid h(x) + [l] \subseteq P\}) \ .$$

This is because the cost is bounded by the distance from $h(x)$ to the next unoccupied position. To avoid two similar calculations for the cases $x \in S$ and $x \notin S$ we will consider the set $S' = S \cup \{x\}$ and the corresponding set P' of occupied table positions. We first show a lemma which intuitively says that if the operation on the key x goes on for at least l steps, then there are either "many" keys hashing to the interval $h(x) + [l]$, or there are "many" keys that hash to some interval having $h(x)$ as its right endpoint.

LEMMA 4. *For any $l > 0$ and $\bar{\alpha} \in (0,1)$, if $h(x) + [l] \subseteq P'$ and $l' = \max\{\ell \mid h(x) - [\ell] \subseteq P'\}$ then at least one of the following holds:*

1. $|h(S') \cap_m (h(x) + [l])| \geq \frac{1+\bar{\alpha}}{2} l - 1$, *or*
2. $|h(S') \cap_m (h(x) - [l'])| \geq l' + \frac{1-\bar{\alpha}}{2} l$.

PROOF. Suppose that $|h(S') \cap_m (h(x) + [l])| < \frac{1+\bar{\alpha}}{2} l - 1$. Now, fix any way of placing the keys in the hash table, e.g., suppose that keys are inserted in sorted order. Consider the set $S^* \subseteq S'$ of keys stored in the interval $I = (h(x) - [l']) \cup (h(x) + [l])$. By the choice of l' there must be an empty position to the left of I, so $h(S^*) \subseteq I$. This means:

$$\begin{aligned} |h(S') \cap_m (h(x) - [l'])| &\geq |h(S^*) \cap_m (h(x) - [l'])| \\ &\geq |S^*| - |h(S^*) \cap_m (h(x) + [l])| \\ &> |I| - (\tfrac{1+\bar{\alpha}}{2} l - 1) \\ &= l' + \tfrac{1-\bar{\alpha}}{2} l \ . \end{aligned}$$

□

THEOREM 2. *Consider any sequence of operations (insertions, deletions, and lookups) in a linear probing hash table where the hash function h used has been chosen uniformly at random from a 5-wise independent family of functions $\mathcal{H}$. Let n and $\bar{\alpha} < 1$ denote, respectively, the maximum number of keys in the table during a particular operation and the corresponding maximum load. Then the expected cost of that operation is $O(1 + (1-\bar{\alpha})^{-3})$. As a consequence, the expected average cost of successful lookups is $O(1 + (1-\bar{\alpha})^{-2})$.*

PROOF. We refer to x, S', and P' as defined previously in this section. As argued above, the expected cost of the operation is

$$O\left(1 + \sum_{l>0} \Pr\{h(x) + [l] \subseteq P'\}\right) \ .$$

Let $l_0 = \frac{6}{(1-\bar{\alpha})^3}$. For $l \leq l_0$ we use the trivial upper bound $\Pr\{h(x) + [l] \subseteq P'\} \leq 1$. In the following we consider the case $l > l_0$.

Let A_ℓ be the event that $|h(S') \cap_m (h(x) + [\ell])| \geq \frac{1+\bar{\alpha}}{2}\ell - 1$, and let A'_ℓ be the event that $|h(S') \cap_m (h(x) - [\ell])| \geq \ell$. Notice that if the second inequality of Lemma 4 holds then $A'_{l'}, \ldots, A'_{l' + \lceil \frac{1-\bar{\alpha}}{2} l \rceil}$ hold. Consequently, if

$$l' \text{ div } (\lceil \tfrac{1-\bar{\alpha}}{2} l \rceil + 1) = k \ ,$$

then $A'_{(k+1)\lceil \frac{1-\bar{\alpha}}{2} l \rceil}$ holds. Combining this observation with Lemma 4 shows that for any l such that $h(x) + [l] \subseteq P'$ the following event holds:

$$A_l \cup \Big(\bigcup_{k>0} A'_{k \lceil \frac{1-\bar{\alpha}}{2} l \rceil} \Big) \ .$$

Hence,

$$\sum_{l>l_0} \Pr\{h(x) + [l] \subseteq P'\} \leq \sum_{l>l_0} \big(\Pr(A_l) + \sum_{k>0} \Pr(A'_{k \lceil \frac{1-\bar{\alpha}}{2} l \rceil})\big) \ .$$

The event A_l implies $|h(S' \backslash \{x\}) \cap_m (h(x) + [l])| \geq \frac{1+\bar{\alpha}}{2} l - 2$. Thus, using Lemma 1 on $S' \backslash \{x\}$ we have

$$\begin{aligned} \Pr(A_l) &\leq \Pr\{|h(S' \backslash \{x\}) \cap_m (h(x) + [l])| \geq \bar{\alpha} l + (\tfrac{1-\bar{\alpha}}{2} l - 2)\} \\ &< \frac{3(\bar{\alpha} l)^2 + \bar{\alpha} l}{(\frac{1-\bar{\alpha}}{2} l - 2)^4} \\ &= O((1-\bar{\alpha})^{-4} l^{-2}) \ . \end{aligned}$$

In the last relation we used the fact that $l > \frac{6}{(1-\bar{\alpha})}$. Using the inequality $\Pr(A'_\ell) < O((1-\bar{\alpha})^{-4} \ell^{-2})$, again derived from Lemma 1, we get

$$\begin{aligned} \sum_{k>0} \Pr(A'_{k \lceil \frac{1-\bar{\alpha}}{2} l \rceil}) &< \sum_{k>0} O((1-\bar{\alpha})^{-4} (k \tfrac{1-\bar{\alpha}}{2} l)^{-2}) \\ &= O((1-\bar{\alpha})^{-6} l^{-2}) \ . \end{aligned}$$

Finally,

$$\begin{aligned} \sum_{l>l_0} \Pr\{h(x) + [l] \subseteq P'\} &< \sum_{l>l_0} O((1-\bar{\alpha})^{-6} l^{-2}) \\ &= O((1-\bar{\alpha})^{-3}) \ . \end{aligned}$$

For the average successful lookup cost, notice that the n keys in the hash table were inserted (in some order) at maximum loads $\bar{\alpha} i/n$, for $i = 1, \ldots, n$, so the expected total insertion cost of these keys is

$$O\left(\sum_{i=1}^{n} 1 + (1 - \bar{\alpha} i/n)^{-3}\right) = O(n + (1-\bar{\alpha})^{-2} n) \ .$$

Observing that the total insertion cost equals the total cost of looking up all keys, the claim follows. □

4.1 A bound with explicit constant factor

In earlier analyses of linear probing it was common to consider the number of probes into the table made during an operation. When the number of probes is analyzed, and not the number of machine instructions executed, it makes sense to be more precise and compute the constant factor involved. The analysis in the previous section hides a rather large constant factor. We will use a slightly different proof technique to obtain a better bound on the average successful search cost. This proof technique will be used again in section 5.

We estimate the total expected number of probes made by a sequence of n insertions into an empty table of size r. From this we can easily derive a bound on the number of probes made by a successful search of a random element in the set. Our bounds show that the expected probe count is not big, for moderate values of $\bar{\alpha}$, although definitely higher than in the case of truly random functions.

The number of probes made during an insertion does not depend on the order of insertions — or equivalently, on the policy of placing elements being inserted. We assume that the following policy is in effect: if x is the new element to be inserted, place x into the first slot $h(x) + i$ that is either empty or contains an element x' such that $h(x') \notin h(x) + [i + 1]$. If x is placed into a slot previously occupied by x' then the probe sequence continues as if x' is being inserted. The entire procedure terminates when an empty slot is found.

THEOREM 3. *Let $\mathcal{H}$ be a 5-wise independent family of functions which map U to R. When linear probing is used with a hash function chosen uniformly at random from $\mathcal{H}$, the expected total number of probes made by a sequence of n insertions into an empty table is less than $n(1 + T(\bar{\alpha}))$.*

PROOF. For every $x_i \in S$, we define d_i to be the displacement of x_i, i.e., the number such that x_i resides in slot $(h(x_i) + d_i) \bmod r$ after all keys have been inserted. The total number of probes made over all insertions is equal to $\sum_{i=1}^{n}(1 + d_i)$. From the way elements are inserted, we conclude that, for $1 \le i \le n$ and $1 \le l \le d_i$, every interval $h(x_i) + [l]$ is *fully loaded*, meaning that at least l elements of $S \setminus \{x_i\}$ hash into it. Let A_{il} be the event that the interval of slots $h(x_i) + [l]$ is fully loaded. Then,

$$\begin{aligned}
\mathrm{E}(d_i) &= \sum_{k=1}^{r} \Pr\{d_i \ge k\} \\
&\le \sum_{k=1}^{r} \Pr\left(\bigcap_{l=1}^{k} A_{il}\right) \\
&\le \sum_{j=0}^{\lfloor \lg r \rfloor} 2^j \cdot \Pr(A_{i\,2^j}) \ .
\end{aligned}$$

An upper bound on $\Pr(A_{i\,2^j})$ can be derived from Lemma 1. However, for small lengths (and $\bar{\alpha}$ not small) the bound is useless, so then we will simply use the trivial upper bound of 1. Let $K = \frac{3\bar{\alpha}^2}{(1-\bar{\alpha})^4}$. We first consider the case $K \ge 1$. Denoting $j_* = \lceil \frac{1}{2} \lg K \rceil$ we have

$$\mathrm{E}(d_i) \le 2^{j_*} - 1 + \sum_{j=j_*}^{\lg r} 2^j \left(\frac{K}{2^{2j}} + \frac{K}{3\bar{\alpha} \cdot 2^{3j}}\right)$$

$$\begin{aligned}
&= 2^{j_*} - 1 + K \sum_{j=j_*}^{\lg r} 2^{-j} + \frac{K}{3\bar{\alpha}} \sum_{j=j_*}^{\lg r} 2^{-2j} \\
&< 2^{j_*} - 1 + \frac{K}{2^{j_*}} \frac{1}{1 - \frac{1}{2}} + \frac{K}{3\bar{\alpha} \cdot 4^{j_*}} \frac{1}{1 - \frac{1}{4}} \\
&\le \sqrt{K} \cdot 2^{1-\mathrm{frac}(\lg \sqrt{K})} - 1 + 2\sqrt{K} \cdot 2^{-(1-\mathrm{frac}(\lg \sqrt{K}))} \\
&\quad + \tfrac{4}{9\bar{\alpha}} \\
&\le 3\sqrt{K} + \tfrac{4}{9\bar{\alpha}} - 1 \ .
\end{aligned}$$

The last inequality is true because $2^t + 2 \cdot 2^{-t} \le 3$, for $t \in [0, 1]$. The assumption that $K \ge 1$ implies that $\bar{\alpha} > 0.29$, but we decide to use the obtained bound for $\bar{\alpha} \ge \frac{1}{3}$. Thus, we may replace $\frac{4}{9\bar{\alpha}} - 1$ with the upper bound of $\frac{1}{3}$.

Doing an easier calculation without splitting of the sum at index j_* gives $\mathrm{E}(d_i) < K(2 + \frac{4}{9\bar{\alpha}})$. When $\bar{\alpha} < \frac{1}{3}$, we may replace $K(2 + \frac{4}{9\bar{\alpha}})$ with the upper bound of $\frac{2.5\bar{\alpha}}{(1-\bar{\alpha})^4}$. □

5. BLOCKED PROBING

In this section we propose and analyze a family of open addressing methods, containing among other a variant of bidirectional linear probing. The expected probe count for any single operation is within a constant factor from the corresponding value in linear probing with a fully random hash function. For successful searches we do a more precise analysis. It is shown that the expected number of probes made during a search for a random element of S is less than $1 + \frac{2}{1-\bar{\alpha}} \frac{\bar{\alpha}}{\alpha}$. The bounds for single operations require a 5-wise independent family of functions. The bound for average successful search requires 4-wise independence.

Suppose that keys are hashed into a table of size r by a function h. For simplicity we assume that r is a power of two. Let $V_j^i = \{j, j + 1, \ldots, j + 2^i - 1\}$ where j is assumed to be a multiple of 2^i. The intervals V_j^i may be thought of as sets of references to slots in the hash table. In a search for key x, intervals V_j^i that enclose $h(x)$ are examined in the order of increasing i. More precisely, $V_{h(x)}^0$ is examined first; if the search did not finish after traversing $V_{h(x)\ominus 2^i}^i$, then the search proceeds in the untraversed half of $V_{h(x)\ominus 2^{i+1}}^{i+1}$. The search stops after traversal of an interval if any of the following three cases hold:

a) key x was found,

b) the interval contained empty slot(s),

c) the interval contained key(s) whose hash value does not belong to the interval.

In case (a) the search may obviously stop immediately on discovery of x — there is no need to traverse through the rest of the interval. In cases (b) and (c) we will be able to conclude that x is not in the hash table.

Traversal of unexamined halves of intervals V_j^i may take different concrete forms — the only requirement is that every slot is probed exactly once. From a practical point of view, a good choice is to probe slots sequentially in a way that makes the scheme a variant of bidirectional linear probing. This concrete scheme defines a probe sequence that in probe numbers 2^i to $2^{i+1} - 1$ inspects either slots

$$(h(x) \ominus 2^i + 2^i,\ h(x) \ominus 2^i + 2^i + 1, \ldots,\ h(x) \ominus 2^i + 2^{i+1} - 1)$$

$$\text{or} \quad (h(x) \ominus 2^i - 1,\ h(x) \ominus 2^i - 2, \ldots,\ h(x) \ominus 2^i - 2^i)$$

depending on whether $h(x) \bmod 2^i = h(x) \bmod 2^{i+1}$ or not. A different probe sequence that falls in this class of methods, but is not sequential, is $(x, j) \mapsto h(x)$ xor j, with j starting from 0.

Insertions. Until key x which is being inserted is placed in a slot, the same probe sequence is followed as in a search for x. However, x may be placed in a non-empty slot if its hash value is closer to the slot number in a special metric which we will now define. Let $d(y_1, y_2) = \min\{i \mid y_2 \in V^i_{y_1 \ominus 2^i}\}$. The value of $d(y_1, y_2)$ is equal to the position of the most significant bit in which y_1 and y_2 differ. If during insertion of x we encounter a slot y containing key x' then key x is put into slot y if $d(h(x), y) < d(h(x'), y)$. In an implementation there is no need to evaluate $d(h(x), y)$ values every time. We can keep track of what interval $V^i_{h(x)\ominus 2^i}$ is being traversed at the moment and check whether $h(x')$ belongs to that interval.

When x is placed in slot y which was previously occupied by x', a new slot for x' has to be found. Let $i = d(h(x'), y)$. The procedure now continues as if x' is being inserted and we are starting with traversal of $V^i_{h(x')\ominus 2^i} \setminus V^{i-1}_{h(x')\ominus 2^{i-1}}$. If the variant of bidirectional linear probing is used, the traversal may start from position y, which may matter in practice.

Deletions. After removal of a key we have to check if the new empty slot can be used to bring some keys closer to their hash values, in terms of the metric d. Let x be the removed key, y be the slot in which it resided, and $i = d(h(x), y)$. There is no need to examine $V^{i-1}_{h(x)\ominus 2^{i-1}}$. If $V^i_{h(x)\ominus 2^i} \setminus V^{i-1}_{h(x)\ominus 2^{i-1}}$ contains another empty slot then the procedure does not continue in wider intervals. If it continues and an element gets repositioned then the procedure is recursively applied starting from the new empty slot.

It is easy to formally check that appropriate invariant holds and that the above described set of procedures works correctly. We leave this to the reader.

5.1 Analysis

We analyze the performance of operations on a hash table of size r when blocked probing is used. Suppose that the hash table stores an arbitrary fixed set of n elements, and let $\bar{\alpha}$ denote the maximum load of $\mathcal{H}$ on sets of size n. Let $C^U_{\bar{\alpha}}$, $C^I_{\bar{\alpha}}$, $C^D_{\bar{\alpha}}$, and $C^S_{\bar{\alpha}}$ be the random variables that represent, respectively: the number of probes made during an unsuccessful search for a fixed key, the number of probes made during an insertion of a fixed key, the number of probes made during a deletion of a fixed key, and the number of probes made during a successful search for a random element from the set. In the above notation we did not explicitly include the fixed set and fixed elements that are used in the operations – they have to be implied by the context. The upper bounds on the expectations of $C^{\Xi}_{\bar{\alpha}}$ variables, which are given by the following theorem, do not depend on choices of those elements.

THEOREM 4. *Let $\mathcal{H}$ be a 5-wise independent family of functions which map U to R. For a maximum load of $\bar{\alpha} < 1$, blocked probing with a hash function chosen uniformly at random from $\mathcal{H}$ provides the following expectations:*

$$\begin{aligned} E(C^U_{\bar{\alpha}}) &< 1 + T(\bar{\alpha}), \\ E(C^I_{\bar{\alpha}}) &< 1 + 2T(\bar{\alpha}), \\ E(C^D_{\bar{\alpha}}) &< 1 + 2T(\bar{\alpha}) \ . \end{aligned}$$

PROOF. Denote by x the fixed element from $U \setminus S$ that is being inserted or searched (unsuccessfully). Let $\bar{C}^U_{\bar{\alpha}}$ be the random variable that takes value 2^i when $2^{i-1} < C^U_{\bar{\alpha}} \le 2^i$, $0 \le i \le \lg r$. We can write $\bar{C}^U_{\bar{\alpha}} = 1 + \sum_{i=1}^{\lg r} 2^{i-1} T_i$, where T_i is an indicator variable whose value is 1 when at least $2^{i-1} + 1$ probes are made during the search. Let A_j be the event that the interval of slots $V^j_{h(x)\ominus 2^j}$ is *fully loaded*, meaning that at least 2^j elements of S are hashed into the interval. If $T_i = 1$ then A_{i-1} holds, so we have:

$$E(\bar{C}^U_{\bar{\alpha}}) \le 1 + \sum_{i=1}^{\lg r} 2^{i-1} \Pr(A_{i-1}) \ .$$

The sum $\sum_{i=0}^{\lg r} 2^i \Pr(A_i)$ appeared in the proof of Theorem 3, so we reuse the upper bound found there to conclude that $E(C^U_{\bar{\alpha}}) < 1 + T(\bar{\alpha})$.

We now move on to analyzing insertions. Let $\bar{C}^I_{\bar{\alpha}}$ be the random variable that takes value 2^i when $2^{i-1} < C^I_{\bar{\alpha}} \le 2^i$, $0 \le i \le \lg r$. The variable $C^U_{\bar{\alpha}}$ gives us the slot where x is placed, but we have to consider possible movements of other elements. If x is placed into a slot previously occupied by key x' from a "neighboring" interval $V^{i+1}_{h(x)\ominus 2^{i+1}} \setminus V^i_{h(x)\ominus 2^i}$, then as many as 2^i probes may be necessary to find a place for x' in $V^i_{h(x)\ominus 2^i}$, if there is one. If $V^{i+1}_{h(x)\ominus 2^{i+1}}$ is fully loaded, then as many as 2^{i+1} additional probes may be needed to find a place within $V^{i+2}_{h(x)\ominus 2^{i+2}} \setminus V^{i+1}_{h(x)\ominus 2^{i+1}}$, and so on. In general — and taking into account all repositioned elements — we use the following accounting to get an overestimate of $E(\bar{C}^I_{\bar{\alpha}})$: For every fully loaded interval $V^i_{h(x)\ominus 2^i}$ we charge 2^i probes, and for every fully loaded neighboring interval $V^{i+1}_{h(x)\ominus 2^{i+1}} \setminus V^i_{h(x)\ominus 2^i}$ we also charge 2^i probes. The probability of a neighboring interval of length 2^i being full is equal to $\Pr(A_i)$. As a result,

$$E(\bar{C}^I_{\bar{\alpha}}) \le 1 + \sum_{i=0}^{\lg r - 1} 2^i \cdot 2\Pr(A_i) < 1 + 2T(\bar{\alpha}) \ .$$

The analysis of deletions is analogous. $\square$

For higher values of $\bar{\alpha}$, the dominant term in the upper bounds is $O((1-\bar{\alpha})^{-2})$. The constants factors in front of term $(1-\bar{\alpha})^{-2}$ are relatively high compared to standard linear probing with fully random hash functions. This is in part due to approximative nature of the proof of Theorem 4, and in part due to tail bounds that we use, which are weaker than those for fully independent families. In the fully independent case, the probability that an interval of length q is fully loaded is less than $e^{q(1-\alpha+\ln\alpha)}$, according to Chernoff-Hoeffding bounds [3, 6]. Plugging this bound into the proof of Theorem 4 would give, e.g.,

$$E(C^U_{\alpha}) < 1 + \frac{e^{1-\alpha+\ln\alpha}}{\ln 2 \cdot |1-\alpha+\ln\alpha|} \ . \qquad (1)$$

For α close to 1, a good upper bound on (1) is $1 + \frac{2}{\ln 2}(1-\alpha)^{-2}$. The constant factor here is ≈ 2.88, as opposed to ≈ 5.2 from the statement of Theorem 4. As α gets smaller, the bound in (1) gets further below $1 + \frac{2}{\ln 2}(1-\alpha)^{-2}$.

5.2 Analysis of successful searches

As before, we assume that the function h is chosen uniformly at random from $\mathcal{H}$. For a subset Q of R, let X_i be the indicator random variable that has value 1 iff $h(x_i) \in Q$, $1 \le i \le n$. The variable $X = \sum_{i=1}^{n} X_i$ counts the number of elements that are mapped to Q. We introduce a random variable that counts the number of elements that have *overflowed* on Q. Define $Y = \max\{X - q, 0\}$, where $q = |Q|$. We will find an upper bound on $E(Y)$, such that it is expressed only in terms of q and $\bar{\alpha}_{\mathcal{H}}$. Denote such a bound by $M_q^{\bar{\alpha}}$. It is clear that

$$E(C_{\bar{\alpha}}^S) \le 1 + \frac{1}{n} \sum_{l=0}^{\lg r - 1} 2^l \frac{r}{2^l} M_{2^l}^{\bar{\alpha}} = 1 + \frac{1}{\alpha} \sum_{l=0}^{\lg r-1} M_{2^l}^{\bar{\alpha}} \ .$$

We are starting the analysis of $E(Y)$, for an arbitrary fixed set Q. The value of $E(Y)$ is $\sum_{j=1}^{n-q} j \cdot \Pr\{X = q + j\}$. A bound on $E(Y)$ can be obtained as the optimal value of an optimization problem, which we will introduce. We denote $E(X)$ shortly by μ. Let variables p_i, $0 \le i \le n$ have domain $[0, 1]$. Define the following optimization problem in variables $p_0, \ldots, p_n$:

$$\text{maximize} \qquad \sum_{j=1}^{n-q} j \cdot p_{q+j}$$

subject to constraints:

$$\begin{aligned} \sum_{i=0}^{n} p_i &= 1 \ , \\ \sum_{i=0}^{\lceil \mu \rceil - 1} (\mu - i) p_i &= \sum_{i=\lfloor \mu \rfloor + 1}^{n} (i - \mu) p_i \ , \\ \sum_{i=0}^{n} (\mu - i)^k p_i &= D_k \ . \end{aligned}$$

If we choose an even $k \ge 2$, and set D_k to be an upper bound on the value of the kth central moment of X, then the optimal value of the objective function is an upper bound on $E(Y)$. Remark that in an optimal solution $p_i = 0$ for $\lceil \mu \rceil \le i \le q$. We will actually not solve the above problem, but its relaxation. We introduce variables d_i, $i \in I = \{-\lceil \mu \rceil, \ldots, -2, -1, 1, 2, \ldots n - q\}$. For $i \in \{-\lceil \mu \rceil, \ldots, -1\}$ the domain of variables d_i is $(0, \mu]$; for $i \in \{1, \ldots n - q\}$ the domain of variables d_i is $(q - \mu, n]$. The variables representing probabilities are still denoted by p_i, but the index set is now I. Define the optimization problem Π over all variables p_i, d_i as:

$$\text{maximize} \qquad \sum_{j=1}^{n-q} p_j (d_j - (q - \mu))$$

subject to constraints:

$$\begin{aligned} \sum_{i \in I} p_i &= 1 \ , & (2) \\ \sum_{i=-1}^{\lceil \mu \rceil} d_i p_i &= \sum_{i=1}^{n-q} d_i p_i \ , & (3) \\ \sum_{i \in I} d_i^k p_i &= D_k \ . & (4) \end{aligned}$$

The optimal value of the objective function for problem Π is not smaller than the optimal value in the original problem.

Lemma 5. *Let $(\bar{p}, \bar{d})$ be an optimal solution to problem Π such that $\bar{d}_i \ne \bar{d}_j$ for $0 < i < j$, and $\bar{d}_i \ne \bar{d}_j$ for $0 > i > j$. Then only one value among $\bar{p}_1, \ldots, \bar{p}_{n-q}$ is not equal to 0, and only one value among $\bar{p}_{-1}, \ldots, \bar{p}_{-\lceil \mu \rceil}$ is not equal to 0.*

Proof. We will prove the claims for $i > 0$ and $i < 0$ separately. Suppose the contrary, that there are two non-zero values among $\bar{p}_1, \ldots, \bar{p}_{n-q}$. W.l.o.g. we may assume that $\bar{p}_1 > 0$ and $\bar{p}_2 > 0$. Define the function $f(d_1, d_2) = \bar{p}_1(d_1 - (q - \mu)) + \bar{p}_2(d_2 - (q - \mu))$ over $d_1, d_2 \ge 0$. We are interested in finding the maximum of function f subject to constraint $\bar{p}_1 d_1^k + \bar{p}_2 d_2^k - D = 0$, where $D = \bar{p}_1 \bar{d}_1^k + \bar{p}_2 \bar{d}_2^k$. According to the method of Lagrange multipliers, the only point in the interior of the domain that is a candidate for an extreme point is $(\hat{d}, \hat{d})$, where $\hat{d} = \sqrt[k]{\frac{D}{\bar{p}_1 + \bar{p}_2}}$. We cannot establish the character of the point directly through an appropriate quadratic form, because the required forms evaluate to zero (the form should be positive definite or negative definite to establish the local minimum or maximum, respectively). However, by comparing $f(\hat{d}, \hat{d})$ to the values of f at boundary points $(\sqrt[k]{D/\bar{p}_1}, 0)$ and $(0, \sqrt[k]{D/\bar{p}_2})$, we find that the maximum is reached in the interior of the domain and it must be at point $(\hat{d}, \hat{d})$. It is not hard to show that $\hat{d} \in (q - \mu, n]$, by looking at the set determined by equation $\bar{p}_1 d_1^k + \bar{p}_2 d_2^k - D = 0$ and using the fact that $(\bar{d}_1, \bar{d}_2) \in (q - \mu, n] \times (q - \mu, n]$.

We construct a new solution $(\tilde{p}, \tilde{d})$ to problem Π in two phases. First, set $\tilde{p}_i = \bar{p}_i$ and $\tilde{d}_i = \bar{d}_i$ for $i \notin \{1, 2\}$, set $\tilde{p}_2 = \tilde{d}_2 = 0$, $\tilde{d}_1 = \hat{d}$ and $\tilde{p}_1 = \frac{\bar{p}_1 \bar{d}_1 + \bar{p}_2 \bar{d}_2}{\hat{d}}$. From $(\hat{d}, \hat{d})$ being the point of maximum of f, it follows that

$$\frac{\bar{p}_1 \bar{d}_1 + \bar{p}_2 \bar{d}_2}{\hat{d}} < \bar{p}_1 + \bar{p}_2 \ .$$

Now $(\tilde{p}, \tilde{d})$ exactly satisfies (3) and it satisfies inequality conditions corresponding to (2) and (4). Already now, the value of the objective function is higher than the value at $(\bar{p}, \bar{d})$. It increases further when we increase values of $\tilde{p}_1$ and $\tilde{p}_{-1}$ in a way that satisfies all the conditions with equalities. We reached a contradiction, meaning that $(\bar{p}, \bar{d})$ cannot have two non-zero values among $\bar{p}_1, \ldots, \bar{p}_{n-q}$.

Suppose that there are two non-zero values among $\bar{p}_{-1}, \ldots, \bar{p}_{-\lceil \mu \rceil}$. W.l.o.g. we may assume that $\bar{p}_{-1} > 0$ and $\bar{p}_{-2} > 0$. To reach a contradiction we use an argument that is in some way dual to the argument for the first part. Define the function $g(d_{-1}, d_{-2}) = \bar{p}_{-1} d_{-1}^k + \bar{p}_{-2} d_{-2}^k$. Now we are interested in finding the minimum of g subject to constraint $\bar{p}_{-1} d_{-1} + \bar{p}_{-2} d_{-2} - (\bar{p}_{-1} \bar{d}_{-1} + \bar{p}_{-2} \bar{d}_{-2}) = 0$. The proof continues similarly to the first part. □

Lemma 6. *Let* OPT *be the optimal value of the objective function for problem Π. Then* OPT $< \frac{1}{2} \sqrt[k]{D_k}$, *and also*

$$\text{OPT} < \begin{cases} \frac{1}{2}^{\frac{k-1}{k}} \sqrt[k]{D_k} - \frac{1}{2}(q - \mu), & \frac{D_k (1 - \frac{1}{k})^k}{(q - \mu)^k} \ge \frac{1}{2} \\ \frac{D_k}{(q-\mu)^{k-1}} ((1 - \frac{1}{k})^{k-1} - (1 - \frac{1}{k})^k), & \text{otherwise} \end{cases}$$

Proof. According to Lemma 5, an optimal solution to problem Π can be obtained from the following simplified problem:

$$\text{maximize} \qquad p_1 (d_1 - (q - \mu))$$

subject to constraints:

$$\begin{aligned} p_1 d_1 &= (1 - p_1) d_{-1} \ , \\ p_1 d_1^k + (1 - p_1) d_{-1}^k &= D_k \ . \end{aligned}$$

Eliminating d_{-1} yields the constraint $d_1^k(p_1 + \frac{p_1^k}{(1-p_1)^{k-1}}) = D_k$. Expressing d_1 from this constraint shows that the objective function is equivalent to

$$f(p_1) = \frac{\sqrt[k]{D_k}}{\sqrt[k]{p_1^{1-k} + (1-p_1)^{1-k}}} - p_1(q - \mu)$$

The value of the first term is symmetrical around the point $\frac{1}{2}$. Therefore, the maximum of f is reached in the interval $(0, \frac{1}{2}]$. By analyzing only the first term, we get a simple upper bound of $\frac{1}{2}\sqrt[k]{D_k}$. To get the other bound we look at the function

$$\bar{f}(p_1) = \sqrt[k]{D_k} \cdot p_1^{1-1/k} - p_1(q - \mu) \ ,$$

which satisfies $f < \bar{f}$. Analyzing $\bar{f}'$ we easily find the maximum of $\bar{f}(p_1)$ over $p_1 \in (0, \frac{1}{2}]$. $\square$

Corollary 2. *Suppose that $\mathcal{H}$ is 4-wise independent. Define*

$$M_q^{\bar{\alpha}} = \begin{cases} \frac{1}{2}\sqrt{\bar{\alpha}q} \ , & \text{If } \frac{\bar{\alpha}}{(1-\bar{\alpha})^2 q} \geq (\sqrt{2}+1)^2 \\ \frac{1}{\sqrt{2}}\sqrt{\bar{\alpha}q} - \frac{1}{2}q(1-\bar{\alpha}) \ , & \text{If } 2 \leq \frac{\bar{\alpha}}{(1-\bar{\alpha})^2 q} < (\sqrt{2}+1)^2 \\ \frac{1}{4}\frac{\bar{\alpha}}{1-\bar{\alpha}} \ , & \text{If } \frac{1}{\sqrt{2}} \leq \frac{\bar{\alpha}}{(1-\bar{\alpha})^2 q} < 2 \\ 0.11 \cdot \frac{3\bar{\alpha}^2 q + \bar{\alpha}}{(1-\bar{\alpha})^3 q^2} \ , & \text{If } \frac{\bar{\alpha}}{(1-\bar{\alpha})^2 q} < \frac{1}{\sqrt{2}} \end{cases}$$

It holds that $E(Y) < M_q^{\bar{\alpha}}$.

Proof. For the first three cases we used $k = 2$, and for the fourth case $k = 4$ was used. The constants in problem Π were substituted as: $\mu = \bar{\alpha}q$, $D_2 = \bar{\alpha}q$, and $D_4 = 3\bar{\alpha}^2 q^2 + \bar{\alpha}q$. The bound on the fourth central moment is proved very similarly to the proof of of Lemma 1. The bound on the variance is easier to prove. $\square$

The analysis is finalized with the following theorem.

Theorem 5. *Let $\mathcal{H}$ be a 4-wise independent family of functions which map U to R. When blocked probing is used with a hash function chosen uniformly at random from $\mathcal{H}$, then*

$$E(C_{\bar{\alpha}}^S) < 1 + \frac{\bar{\alpha}}{\alpha} \cdot \begin{cases} \frac{2}{1-\bar{\alpha}} \ , & 0.5 \leq \bar{\alpha} \\ \frac{1.1}{1-\bar{\alpha}} \ , & 0.3 < \bar{\alpha} < 0.5 \\ \frac{0.85}{1-\bar{\alpha}} \ , & \bar{\alpha} \leq 0.3 \end{cases} .$$

Proof. As stated at the beginning of this section, $E(C_{\bar{\alpha}}^S)$ is upper-bounded by

$$1 + \frac{1}{\alpha}\sum_{l=0}^{\lg r - 1} M_{2^l}^{\bar{\alpha}} \ .$$

Now that we have $M_q^{\bar{\alpha}}$ values, we are only left to carry out summation of $\sum_l M_{2^l}^{\bar{\alpha}}$. We split the sum into four parts. The splitting indexes are set as follows: $l_1 = \lfloor \lg \frac{\bar{\alpha}}{(\sqrt{2}+1)^2(1-\bar{\alpha})^2} \rfloor$, $l_2 = \lfloor \lg \frac{\bar{\alpha}}{2(1-\bar{\alpha})^2} \rfloor$, $l_3 = \lfloor \lg \frac{\sqrt{2}\bar{\alpha}}{(1-\bar{\alpha})^2} \rfloor$. Some of the indices may be smaller than 0, in which case the sum is split into fewer parts. For tighter bounding, we will also need the following values: $t_1 = \operatorname{frac}(\lg \frac{\bar{\alpha}}{(\sqrt{2}+1)^2(1-\bar{\alpha})^2})$, $t_2 = \operatorname{frac}(\lg \frac{\bar{\alpha}}{2(1-\bar{\alpha})^2})$, and $t_3 = \operatorname{frac}(\lg \frac{\sqrt{2}\bar{\alpha}}{(1-\bar{\alpha})^2})$.

Suppose first that $l_1 \geq 0$. Simple calculations yield the following four inequalities:

$$\sum_{l=0}^{l_1} M_{2^l}^{\bar{\alpha}} < \frac{1}{\sqrt{2}}\frac{\bar{\alpha}}{1-\bar{\alpha}}2^{-t_1/2} \ ,$$

$$\sum_{l=l_1+1}^{l_2} M_{2^l}^{\bar{\alpha}} \leq \frac{\bar{\alpha}}{1-\bar{\alpha}}\Big((1+\frac{1}{\sqrt{2}})2^{-t_2/2} - 2^{-t_1/2} - \frac{1}{2}2^{-t_2} + \\ + \frac{1}{(\sqrt{2}+1)^2}2^{-t_1}\Big) \ ,$$

$$\sum_{l=l_2+1}^{l_3} M_{2^l}^{\bar{\alpha}} \leq \frac{1}{4}\frac{\bar{\alpha}}{1-\bar{\alpha}}(l_3 - l_2) \ ,$$

$$\sum_{l=l_3+1}^{\infty} M_{2^l}^{\bar{\alpha}} \leq 0.24\frac{\bar{\alpha}}{1-\bar{\alpha}}2^{t_3} + 0.02\frac{1-\bar{\alpha}}{\bar{\alpha}}2^{2t_3} \quad .$$

For the third inequality, we notice that $l_3 - l_2 = 2$ for $t_3 \leq \frac{1}{2}$, and $l_3 - l_2 = 1$ for $t_3 > \frac{1}{2}$. The assumption that $l_1 \geq 0$ implies that $\bar{\alpha} > \frac{1}{2}$ (we do not need the exact bound on $\bar{\alpha}$), which we use for the second term of the r.h.s. of the fourth inequality. Maximizing over t_1, t_2, t_3 shows that $\sum_{l=0}^{\infty} M_{2^l}^{\bar{\alpha}} < \frac{2\bar{\alpha}}{1-\bar{\alpha}}$.

Now suppose that $l_1 < 0$ and $l_2 \geq 0$. The first part of the sum is now

$$\sum_{l=0}^{l_2} M_{2^l}^{\bar{\alpha}} \leq \frac{\bar{\alpha}}{1-\bar{\alpha}}\Big((1+\frac{1}{\sqrt{2}})2^{-t_2/2} - \frac{1}{2}2^{-t_2}\Big) - \\ - (1+\frac{1}{\sqrt{2}})\sqrt{\bar{\alpha}} + \frac{1-\bar{\alpha}}{2} \ ,$$

while the bounds on the other two parts stay the same. Since $l_1 < 0$, it follows that $\bar{\alpha} < 0.7$. When $\bar{\alpha} < 0.7$, it holds that $3\sqrt{\bar{\alpha}} > \frac{\bar{\alpha}}{1-\bar{\alpha}}$. On the other hand, $l_2 \geq 0$ is equivalent to $\bar{\alpha} \geq \frac{1}{2}$. When $\bar{\alpha} \geq \frac{1}{2}$, it holds that $\sqrt{\bar{\alpha}} > 1 - \bar{\alpha}$. Simple calculations again show that $\sum_{l=0}^{\infty} M_{2^l}^{\bar{\alpha}} < \frac{2\bar{\alpha}}{1-\bar{\alpha}}$ (a slightly lower constant could also be stated).

Now suppose that $l_2 < 0$ and $l_3 \geq 0$. This assumption implies that $\bar{\alpha} > 0.3$, and we may write $\frac{1-\bar{\alpha}}{\bar{\alpha}} < \frac{6\bar{\alpha}}{1-\bar{\alpha}}$. In this case, we get $\sum_{l=0}^{\infty} M_{2^l}^{\bar{\alpha}} < 1.1\frac{\bar{\alpha}}{1-\bar{\alpha}}$.

In the final case, $l_3 < 0$, we have $\sum_{l=0}^{\infty} M_{2^l}^{\bar{\alpha}} < \frac{0.66\bar{\alpha}^2}{(1-\bar{\alpha})^3} + \frac{0.15\bar{\alpha}}{(1-\bar{\alpha})^3}$. Since $\bar{\alpha} < 0.33$ in this case, we may use $\frac{\bar{\alpha}}{(1-\bar{\alpha})^2} < 0.75$ and $\frac{1}{(1-\bar{\alpha})^2} < 2.25$ to get the stated bound. $\square$

6. OPEN PROBLEMS

An immediate question is whether it is possible to improve the dependence on α in the analysis of linear probing with constant independence. In general, the problem of finding practical, and provably good hash functions for a range of other important hashing methods remains unsolved. For example, cuckoo hashing [9] and its variants presently have no such functions. Also, if we consider the problem of hashing a set into $n/\log n$ buckets such that the number of keys in each bucket is $O(\log n)$ w.h.p., there is no known explicit class achieving this with function descriptions of $O(\log |U|)$ bits. Possibly, such families could be designed using efficient circuits, rather than a standard RAM instruction set.

Acknowledgment. We thank Martin Dietzfelbinger and an anonymous reviewer for numerous suggestions improving the presentation. In particular, we thank Martin Dietzfelbinger for showing us a simplified proof of Lemma 1.

7. REFERENCES

[1] J. R. Black, C. U. Martel, and H. Qi. Graph and hashing algorithms for modern architectures: Design and performance. In *Proceedings of 2nd International Workshop on Algorithm Engineering (WAE '92)*, pages 37–48. Max-Planck-Institut für Informatik, 1998.

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

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

[4] M. Dietzfelbinger and C. Weidling. Balanced allocation and dictionaries with tightly packed constant size bins. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP '05)*, volume 3580 of *Lecture Notes in Computer Science*, pages 166–178. Springer, 2005.

[5] G. L. Heileman and W. Luo. How caching affects hashing. In *Proceedings of the 7th Workshop on Algorithm Engineering and Experiments (ALENEX '05)*, pages 141–154. SIAM, 2005.

[6] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[7] D. E. Knuth. Notes on "open" addressing, July 22 1963. Unpublished memorandum. Available at `http://citeseer.ist.psu.edu/knuth63notes.html`.

[8] C. P. Kruskal, L. Rudolph, and M. Snir. A complexity theory of efficient parallel algorithms. *Theoretical Computer Science*, 71(1):95–132, Mar. 1990.

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

[10] H. Prodinger and W. Szpankowski (eds.). Special issue on average case analysis of algorithms. *Algorithmica*, 22(4), 1998. Preface.

[11] J. P. Schmidt and A. Siegel. The analysis of closed hashing under limited randomness (extended abstract). In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC '90)*, pages 224–234. ACM Press, 1990.

[12] A. Siegel. On universal classes of extremely random constant-time hash functions. *SIAM J. Comput.*, 33(3):505–543, 2004.

[13] A. Siegel and J. Schmidt. Closed hashing is computable and optimally randomizable with universal hash functions. Technical Report TR1995-687, New York University, Apr., 1995.

[14] M. Thorup and Y. Zhang. Tabulation based 4-universal hashing with applications to second moment estimation. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '04)*, pages 615–624, 2004.

[15] M. N. Wegman and J. L. Carter. New hash functions and their use in authentication and set equality. *J. Comput. System Sci.*, 22(3):265–279, 1981.