Double hashing is computable and randomizable with universal hash functions

Jeanette P. Schmidt
Incyte Pharmaceuticals, Inc.
3174 Porter Drive
Palo Alto, CA 94304

Alan Siegel
Department of Computer Science
Courant Institute
251 Mercer Street
N.Y.C., NY 10012

Abstract

Universal hash functions that exhibit $c \log n$-wise independence are shown to give a performance in double hashing and any reasonable generalization of double hashing that has an expected probe count of $\frac{1}{1-\alpha} + \epsilon + O(\frac{1}{n})$ for the insertion of the αn-th item into a table of size n, for any fixed $\alpha < 1$ and $\epsilon > 0$. This randomized performance bound is the first to be established for any classical closed hashing scheme, and is within ϵ of optimal.

The results are derived from a novel formulation that overestimates the expected probe count by underestimating the presence of keys already inserted into the hash table, and from a sharp analysis of the underlying stochastic structures formed by colliding keys.

Categories and Subject Descriptors: E.1 [**Data**]: Data Structures—*arrays*; *tables*; E.2 [**Data**]: Data Storage Representations—hash-table representations; F2.2 [**Analysis of Algorithms and Problem Complexity**]: Non-numerical Algorithms and Problems—*sorting and searching*.

General terms: Algorithms, Theory.

Additional Key Words and Phrases: Double hashing, hashing, limited independence, open addressing, closed hashing, uniform hashing, universal hash functions.

The work of the first author was supported in part by NSF grant CCR-9110255 and the New York State Science and Technology Foundation Center through its Center for Advanced Technology.

The work of the second author was supported in part by ONR grant N00014-85-K-0046, NSF grants CCR-8906949, CCR-8902221, CCR-9204202 and CCR-9504175.

Summary

This paper gives the first performance bounds for classical closed hashing schemes in the case of limited randomness, and thereby provides the first randomized performance analysis of these algorithms in a model that supports programmable computation. In contrast, traditional analyses have relied upon the use of mathematically random hash functions or the assumption that the input data is completely random. Unfortunately, real data is seldom provably random, and the average program size of a random hash function is so large it exceeds the size of the database it is intended to support. The bounds for limited randomness establish near optimal randomized performance for classical double hashing when restricted to programmable hash functions. Moreover, the proof technique unifies and significantly generalizes previous results even in the case of unlimited randomness.

Let $D = (x_1, x_2, \ldots, x_{\alpha n})$ be a sequence of αn distinct search keys, for $\alpha < 1$, belonging to the universe $U = \{0, 1, \ldots, m-1\}$. The objective is to hash D into a search table of n locations without the use of pointers and without relocating placed keys. We show that for any fixed load $\alpha < 1$, universal classes of $c \log n$-wise independent hash functions yield, up to an error of $\epsilon + O(\frac{1}{n})$, the same expected probe performance as fully random hash functions for double hashing. That is, the expected number of probes to insert the αn-th item is $\frac{1}{1-\alpha} + \epsilon + O(\frac{1}{n})$ when $c \log n$-wise independent functions are used instead of idealized mathematically random functions. The positive constant c depends on α and ϵ, but not n. The error term ϵ can be any fixed positive quantity. A consequence is that $O(\log\log m + \log^2 n)$ random bits suffice for these hash schemes. Subsequent work advances the formulations of this paper to reduce the total error to $O(\frac{1}{n})$ [17], to extend the results to additional hashing schemes, and to analyze higher moments of the insertion cost.

Our performance bound for double hashing readily applies to any generalization that exhibits approximate pairwise independence for the first $O(\log n)$ probes of any item, features statistically independent probe functions for any $O(\log n)$ keys, is robust in the sense that insertions must eventually succeed (unless the table is full), and typically does not revisit probe locations too often during the insertion of an individual key. Under such circumstances, the expected probe count

to locate the αn-th item is again bounded by $\frac{1}{1-\alpha} + \epsilon + O(\frac{1}{n})$. The performance bound for these generalizations is new even in the case of full randomness.

When combined with the highly independent fast hash functions of [16], these results give the first randomized classical closed hashing schemes featuring, for a word model of computation, a constant number of arithmetic operations per probe and nearly optimal probe performance.

These results are derived from a novel formulation that overestimates the expected probe count by underestimating the presence of local keys already inserted into the hash table, and from a sharp analysis of the underlying stochastic structures formed by colliding keys.

1.0 Introduction and background

Let $D = (x_1, x_2, \ldots, x_{\alpha n})$ be a sequence of αn distinct search keys, $\alpha < 1$, belonging to the universe $U = \{0, 1, \ldots, m-1\}$. We wish to hash D into a table $L[0..n-1]$. In closed hashing, which is also called open-address hashing, all data must be placed within the hash table, and pointers are not allowed. In this model, each key $x \in U$ is mapped into a probe sequence $p(x,1), p(x,2), ... \in [0, n-1]$ (which ideally would be a permutation of $[0, n-1]$), and the generic insertion scheme is to place x in the first vacant table location in its probe sequence. The search procedure is to traverse the same sequence until the item is located, or an empty table slot is identified, in which case the item is known to be absent.

Uniform hashing is an idealized model where the probe sequence $p(x,*)$, for each key $x \in U$, is assumed to be a fully independent random function (or permutation). Traditional double hashing, which originates in the 1968 Ph.D. thesis of Guy de Balbine [10], defines each probe sequence to be a random arithmetic progression:

$$p(x,j) = f(x) - (j-1)d(x) \bmod n,$$

where the table size n is prime, $f(x)$ is assumed to return an arbitrarily selected integer in $[0..n-1]$, and $d(x)$ is an arbitrarily selected value in $[1..n-1]$. The $2|U|$ random values $\{d(x), f(x)\}_{x \in U}$ are assumed to be fully independent and uniformly distributed over their respective ranges.

In the most common versions of these hashing models, a key is inserted into the table L at the first vacant location in $L[p(x,1)]$, $L(p(x,2)]$, $\ldots$, as opposed to some earlier probe position with the concomitant relocation of the former occupant. Relocation schemes originate with Brent (c.f. [4]), and were designed to reduce the expected search time in applications where search operations are much more frequent than insertions. Such schemes, however, are not the subject of this study. Accordingly, we shall hereafter refer solely to models of closed hashing without rearrangement.

Contributions begun by Knuth, [10], Ullman [18], Ajtai, Komlós, and Szemerédi [2] culminate in a proof by Yao [20], who showed that in terms of expected retrieval cost, uniform hashing is optimal: up to a factor of $(1+o(1))$, no family of hash functions gives a better average-case performance than the random functions of uniform hashing. For double hashing with any fixed load factor $\alpha < 1$, Lueker and Molodowitch [11] proved that if f and d are taken to be random uniformly distributed functions, then the expected number of probes to insert the $(\alpha n+1)$-st item is $\frac{1}{1-\alpha} + O(\frac{\log^{5/2} n}{\sqrt{n}})$, which is asymptotically equivalent to uniform hashing, and hence optimal in term of average case retrievals. Previously Guibas and Szemerédi had established a similar bound for loads $\alpha < .29$ [8]. Lueker and Molodowitch note that Ajtai, Guibas, Komlós and Szemerédi had also subsequently extended the range to fixed $\alpha < 1$.

An important consequence of these analyses is the certainty that only two random functions need to be defined to provide an optimal hashing scheme. Unfortunately, the question of what computable functions can be proven to behave like random hash functions has been open. The formal use of fully random functions on U leaves our understanding of computable hashing in an unsatisfactory state, since such a function has a program size (Kolmogorov complexity) of about $2|U|\log n$ bits, on average. We could use polynomials of degree $n-1$ to implement n-wise independent random functions (c.f. Definition 1) to reduce the spatial cost to $O(\log\log m + n\log n)$, while raising the time needed for evaluating the hash function to n, which is no better than the time needed to search an unordered list.

A more constructive perspective on the traditional analyses such as [11] is that they establish optimal performance bounds for hash schemes that use programmable functions, provided the measure of running time is taken to be the performance averaged over all possible input sequences. Of course, this is not the same as a randomized performance bound, where the expected running time for any fixed sequence of data is shown to be optimal.

Carter and Wegman [6] and [19], contributed to our understanding of limited randomness and randomized performance by introducing the concept of universal hash functions, and showing that these functions, when used for open hashing with separate chaining, result in an expected probe performance that is equivalent to the fully random formulation. In this model, $L[i]$ comprises a linked list of keys hashing to the value i: external storage is used to hold colliding keys and pointer information linking them together.

In particular, Carter and Wegman exhibited the universal classes of uniformly distributed h-wise independent hash functions (which they called ${}_h$):

$$F_{h,n} = \{f \mid f(x) = (\sum_{0 \le j < h} a_j x^j \mod p) \mod n,\ a_j \in [0, p-1]\}, \tag{1}$$

where $p \ge m$ is prime. They showed that if, for any $D \subset U$, a hash function is randomly selected from $F_{h,n}$ (independent of D), then the sum of the expected j-th moments of the chain (i.e., list) lengths is essentially the same as that resulting from fully random functions, for $j \le h$. For separate chaining, the second moment determines the expected retrieval time, whence pairwise independence guarantees optimal expected performance.

Carter and Wegman also posed as an open question whether a comparable result could be achieved for any form of closed hashing, such as double hashing. We resolve the question affirmatively if, for a sufficiently large c, $c \log n$-wise independent hash functions are used.

For our purposes, however, the evaluation time of their universal hash functions is too high, but the results of [16] exhibit, in principle, such families with constant evaluation time for a standard word model Random Access Machine. (It should be noted that while the functions of [16] are programmable, the performance is only of interest from a theoretical perspective because the constants are unacceptably large. On the other hand, the model is realistic, and an acceptable implementation might yet be discovered. The model uses the standard conventions of allowing indexed access to a size n array in constant time. A constant number of multiplications and integer divisions are deemed to require $O(1)$ time for keys in the universe U, but most of the (constant number of) operations are on $O(\log n)$-bit words. The requisite number of random bits still turns out to be $O(\log\log m + \log^2 n)$. However the $\log^2 n$ term is increased by a reasonably small constant factor, and an auxiliary storage of n^ϵ $(\log n)$-bit words is provably necessary, for some $\epsilon < 1$.)

The attraction of a universal hashing formulation is two-fold. First, the notion provides a way to construct randomized algorithms for hashing, which eliminates any requirement that the data be "random," for suitable performance. Second, it allows a fixed – presumably computable – set of functions to be used for hashing, as opposed to an axiomatically specified "random" function.

This work differs considerably from the analyses of [8] and [11] in that both analyze the intersection patterns of arithmetic progressions, whereas this work uses no notion of such a sequence. Byproducts of such an abstract proof formulation include:

1) Automatic generalization of the performance results to arbitrary hashing schemes that satisfy (minor) requirements regarding adequate table coverage and that (more importantly) exhibit approximate pairwise independent probing, which can be formalized as follows.
   $\forall x\ \forall i \neq k\ \forall \ell, j:\quad \mathrm{Prob}\{p(x,k)=\ell,\ p(x,i)=j\} = O(\frac{1}{n^2})$. Actually, the probe sequence can be defined by the equation $p(x,j) = h(f(x), d(x), j)$, for random f and d, and any **deterministic** function h as long as approximate pairwise independence is assured for the first $O(\log n)$ probes, and reasonable coverage occurs for later probes. The coverage requirements, as quantified later, simply ensure that overall probe coverage is adequate to guarantee that the insertion of a key x will fail only when the table is full, and that within any x's probe sequence, locations will not be repeated enough to degrade the resulting performance.

2) A proof that $O(\log n)$-wise independent hash functions are random enough to preserve the expected performance of the hashing schemes in 1), up to an error of $\epsilon + O(\frac{1}{n})$.

The rest of this section is organized as follows. Subsection 1.1 explains why a two-level hashing scheme requires only $O(\log\log m + \log^2 n)$ random bits. Subsection 1.2 formalizes a notion of limited independence with requirements that, in most respects, are necessarily a little stronger than the definitions generally encountered in the literature (and in other respects slightly weaker) but are still readily achievable. It also presents constructions of two-level hash functions that exhibit the statistical randomness required by our analyses. Subsection 1.3 outlines the rest of the paper.

1.1 Reducing the domain

Although our proofs show that any set of sufficiently well behaved hash functions can be used for double hashing, it is worth noting that a universal class of linear congruential hash functions can, with high probability, map the data D into a polynomial sized space such as $[0, n^4 - 1]$ in a collision-free manner. Then the universal class $F_{h,n}$,(as defined in Section 1.0), can be restricted to have coefficients of size $O(n^4)$, as opposed to size $O(m)$, which might be much larger. Such mappings can be pieced together from techniques in [6], [12] and [7]. Accordingly, we first exploit the following variation of Lemma 2 from [7]:

Fact 1: Let $P_k = \{p \mid p \text{ is prime and } p \in (n^k \log m, (2+\beta)n^k \log m)\}$, for some small suitably fixed $\beta > 0$. Then

$$\forall x \neq y \in D: \ \mathrm{Prob}_{p \in P_k}\{x = y \bmod p\} < n^{-k}.$$

Proof: [12],[7]. By the Prime Number Theorem, $|P_k| = \frac{(1+\beta)n^k \log m}{k \log n + \log \log m}(1 - o(1))$. The product of any $\gamma|P_k|$ primes in P_k is bounded below by $(n^k \log m)^{\gamma|P_k|} > (m)^{\gamma n^k}$, whence no more than $\gamma \leq 1/n^k$ of the elements of P_k can divide $|x - y|$. ∎

Fact 2: Let $F_0(p) = \{h \mid h(x) = (ax + b \bmod p) \bmod n^k,\ a \neq 0, b \in [0, p-1]\}$, where $p > n^k$ is prime. Then

$$\forall x \neq y \in [0, p-1]: \ \mathrm{Prob}_{f \in F_0(p)}\{f(x) = f(y)\} \leq n^{-k}.$$

Proof: [6]. Given x and y, $x, y \in [0, p-1]$, $x \neq y$, the number of different $f \in F_0(p)$ where $f(x) = f(y)$, is precisely the number of 2×2 linear systems in a and b:

$$\begin{cases} ax + b = c + dn^k \bmod p, \\ ay + b = c + en^k \bmod p, \end{cases} \qquad \text{where } c, d, e \geq 0;\ c + dn^k < p;\ c < n^k;\ e \neq d;\ c + en^k < p.$$

Now $c + dn^k$ can have p different values. The remaining parameter e cannot be set to d because this would give $a = 0$. Thus there are at most $\lceil p/n^k - 1 \rceil$ different values available for e. Since there are exactly $p(p-1)$ different functions in F_0, and the number of f where $f(x) = f(y)$ is at most $p\lceil p/n^k - 1\rceil = p\lfloor p/n^k \rfloor \leq p\frac{p-1}{n^k}$, the result follows. ∎

Combining Facts 1 and 2 shows that a hash function selected at random from $F_0^k = \cup_{p \in P_k} F_0(p)$ will, with probability exceeding $1 - 2\binom{\alpha n}{2}n^{-k}$, map D into $[0, n^k - 1]$ with no collisions at all among its $\binom{\alpha n}{2}$ pairs. We can take $k=4$, so that the probability of a collision is below $1/n^2$, and assume the functions $F_{h,n}$ are defined for $p \approx n^4$.

Because of this preprocessing, only $O(\log\log m + h\log n)$ random bits are needed for the composite universal hash functions $F_{h,n} \circ F_0^4$. Our applications will require h to be proportional to $\log n$

1.2 Limited randomness

Carter and Wegman defined a finite family of hash functions F with domain U and range R to be strongly *universal*$_h$ if

$$\forall\, y_1,\dots,y_h \in R, \forall \text{ distinct } x_1,\dots,x_h \in U: \quad |\{f \in F : f(x_i) = y_i,\, i = 1,2,\dots,h\}| = \frac{|F|}{|R|^h}, \qquad (2)$$

so that the fraction of functions in F that achieve the desired mapping of the x_i's is the same as that for fully random functions. This definition combines the requirements of uniformity and h-wise independence. The specification is a little stronger than that used by Carter and Wegman for open hashing, and was introduced by them for application in cryptography [19]. They also gave an application of *almost universal*$_h$ functions where the function density $\frac{|F|}{|R|^h}$ is increased by a constant factor μ, and used as an upper bound for the number of functions that satisfy the h constraints in (2).

Our bounds for closed hashing are so dependent on inclusion-exclusion that we need a very precise notion of *almost universal*$_h$, which separates the uniformity and independence requirements as follows.

Definition 1.

We say that a finite set of functions F with domain U and range R is an h-wise independent universal family of hash functions with *tolerance* β if F exhibits (h)-wise independence:

$\forall\, y_1,\dots,y_h \in R, \forall$ distinct $x_1,\dots,x_h \in U$:

$$\frac{|\{f \in F : f(x_i) = y_i,\, i = 1,2,\dots,h\}|}{|F|} = \prod_i \frac{|\{f \in F : f(x_i) = y_i\}|}{|F|},$$

and near uniformity:

$$\forall\, y \in R, \forall\, x \in U: \quad (1-\beta)\frac{|F|}{|R|} \le |\{f \in F : f(x) = y\}| \le (1+\beta)\frac{|F|}{|R|}.$$

Thus the family of hash functions has a nearly uniform distribution, and the joint probability distribution on any subset comprising h or fewer points in U exhibits the usual (multiplicative) independence. It is worth observing that the function classes $F_{h,n}$, from Section 1.0 are (h)-wise independent with *tolerance* $1/n^3$ for a universe of size n^4.

On the other hand, our premapping step for larger universes will not quite meet the multiplicative requirement of Definition 1 because F_0^4 will have too many hash functions that map two or more hash keys from D into the same value in $[0, n^4 - 1]$. If we ignore such unfortunate cases, and charge, say, an $O(n)$ cost per insertion for such instances, then our performance bounds will not change, and we can rely on the constructions of Section 2 to perform well enough in general. Accordingly, our final randomness characterization is as follows.

Definition 2.

We say that a finite family of hash functions F with domain U and range R is *effectively* (h)-*wise independent with* β-*tolerance* if for each $D \subset U$ with $|D| \leq n$, $\exists \widehat{F} \subset F$ where $\frac{(1+\beta)|\widehat{F}|}{|F|} > 1$ and $\widehat{F}$ is (h)-wise independent with tolerance $n\beta$ for domain D and range R.

It is worth remarking that β-tolerance serves two purposes. First, it allows simple premapping schemes (such as those used at the end of Subsection 1.0) to be included within our abstract computational models. Second, the formulation allows the model to include theoretical (but currently infeasible) models of constant-time highly random hash functions [16]. For large key domains, these constant-time hash functions would turn out to require greater computational resources were such premapping schemes not supported by the underlying performance calculations.

We will hereafter take the requirement of uniform distribution with β-tolerance to be understood, and simply refer to these schemes in terms of their limited independence. Section 1.1 gives a formal construction where for any fixed set $D \subset U$ of n input keys, all but $1/n^2$ of the F_0^4 map D into $[0, n^4 - 1]$ in a collision free way, and the subsequent hashing more than meets the requirements for approximate uniformity and near independence. The resulting hash functions are effectively (h)-wise independent with $\frac{1}{n^2}$-tolerance as specified by Definition 2.

In our formal models, a family of hash functions H will comprise a finite set of functions. Given the data sequence D, a specific hash function is selected by randomly choosing a function from H with each element equally likely to be selected. The statistical properties defined by Definitions 1, 2, and Definition 4 are with respect to H.

1.3 A proof outline

Traditional analyses of hashing view the state of a hash table as a stochastic process that evolves over a duration of αn probabilistic insertions. Lueker and Molodowitch, for example, analyze double hashing in the fully random case with an elegant scheme that keeps the table distribution uniform by introducing moderately improbable randomizing insertions of fake keys to correct the distribution at each insertion step [11]. By vigilantly maintaining a fully random table distribution, they establish a simple proof that double hashing and uniform hashing exhibit comparable collision statistics in a fairly strong sense, and this intuition has turned out to be invaluable in this current work, which establishes an even closer statistical equivalence. However, such an evolutionary approach seems to be inappropriate for instances where the randomness is limited, since all of the randomness would be used up after $\log n$ insertions. Instead, we are obliged to establish the bounds with a proof technique that can be extended from uniform hashing to double hashing with full independence to a comparable double hashing with limited independence. The hashing models are fully specified in Definition 4.

Let a fixed hashing model complete with a (probabilistically selected) hash function be specified, and consider how the i-th key x_i is hashed into the table. The proof is based on analyzing the dependency graph $G(x_i, D)$, which is defined as follows. The graph is the DAG comprising the root x_i, the dependency graphs for the keys that are encountered during the unsuccessful probes of L while trying to insert x_i, and edges from x_i to each of these encountered vertices. This DAG contains its vertex keys, their indices within D, and additional collision information, but not the actual probe locations $p(x, j)$. The dependency set $dep(x_i, D)$ is the vertex set of $G(x_i, D)$. Technically, the structure is a multigraph because a vertex can have multiple probes to the same location. These edges are distinct; adjacency lists are ordered by the probe count associated with each edge.

Given a subsequence $S \subset D$, one might ask, what is the probability that the specific keys in S are the precise and full cause for the number of probes needed to insert x? A necessary condition is that $dep(x, S) = S$ (c.f. Definition 8 as applied to $P(k, k)$ for $k = |S|$, and Section 2.1.1). The probability that the probe sequences for S have this behavior turns out to be, up to a factor of $(1 + O(\frac{|S|^3}{n}))$, the same in uniform hashing and our generalized double hashing schemes, as long as $6|S|$ does not exceed the amount of independence provided by our hash functions (Lemma 2).

Unfortunately, there can be many subsequences in D that, in the absence of other competing sequences, would satisfy the collision conditions for the set S. Consequently, we will sum, over all suitable subsets $S \subset D$, weighted terms that are products. Each term will have factors to account for the probability that S locally hashes into a dependency DAG, the number of probes needed to insert x into the hash structure built from S, and an overestimate of the probability that locally determined placements for S in L are actually empty at the correct times when the elements of S are hashed as members of D.

Thus, the expected number of probes will be an overestimate because the statistics will overcount the probability that various subsets are the true dependency set for the key $x_{\alpha n}$. The advantage of this formulation is that most of the analysis can be restricted to lightly loaded tables. In this statistical formulation, x will be most difficult to insert if the table is viewed as being statistically empty; overestimates of vacancy probabilities give overestimates of the expected probe count.

To make the problem tractable, we will take a weak definition of vacancy. The statistical notion of vacancy will be quantified by a vacancy estimator $q(t)$ (Definition 14), which overestimates the probability that a given location will be empty when the t-th key in D is about to be hashed into the table. The estimators will turn out to be *multiplicative*, which means that the probability that some location is vacant at insertion time t and some other location is vacant at time s is, apart from asymptotically negligible error, $q(s)q(t)$. This approximate multiplicativity will hold for collections of $O(\log n)$ different insertion times, and will be independent of the specific hash locations as long as they are distinct. For such a q, we will show that the expected number of probes needed to insert the t-th item into a hash table of size n is bounded by

$$\frac{1}{\sqrt{1-2\frac{1}{n}\sum_{0<j<\alpha n} q(t)}} + O(\frac{1}{n}).$$

This formulation (Theorem 4) gives overestimates for the expected insertion count in terms of overestimates of q, and is tight when $q(t) = 1 - \frac{t-1}{n}$. However, the greater part of the proof effort is dedicated to the subsidiary issues of defining satisfactory vacancy overestimators, showing that they are statistically equivalent across the different hashing models, and reducing the asymptotic error terms. And the question of whether upper bounds for $q(t)$, in our models of limited independence, can be shown to be asymptotically equal to $1 - \frac{t-1}{n}$ must await the further developments of [17].

2.0 Generic probe counts

Since our probe formulations are based on graphs that model all essential collision behavior, some preliminary definitions are needed to formalize the counting arguments.

2.1 Basic definitions

Definition 3.

- The hash keys $D = (x_1, x_2, \ldots, x_{\alpha n})$ comprise a sequence of αn distinct keys, $\alpha < 1$, belonging to the universe $U = \{0, 1, \ldots, m-1\}$, and $p(x, j) : U \to [0, n-1]$ denotes the j'th probe location for key x.
- The ith element in a sequence S is denoted by S_i. For D, we also have $D_i = x_i$. If T is a subsequence of the indices $1, 2, 3, \ldots, \alpha n$, then D_T is the subsequence of D that is indexed by T.
- The random variable giving the number of probes needed to insert x_i is defined to be $probe_i$. The randomness is due to the randomness in the hash functions as opposed to the data.
- *A rooted DAG* is a directed acyclic graph with only one root, (i.e. one vertex with indegree 0).
- Let $dgr(G)$ of a rooted DAG G be the outdegree of the root of G.

- Containment will be used in three ways. $A \subset B$ will have the usual meaning when A and B are sets. When B is a sequence, A will be a subsequence of B. We will also need to refer to all permutations of subsequences in a sequence B. Let $A \Subset B$ mean some permutation of the sequence A is a subsequence of B.

- Set difference will be denoted by $A - B$. If A and B are sequences with $B \subset A$, $A - B$ will denote the sequence A with the elements of B removed. The elements will always be unique, so no ambiguity will be possible. When x is an element in A, we will sometimes take the liberty of writing $A - x$ instead of the more precise $A - (x)$.

- Let "x is *embedded* at location ℓ" mean that as a consequence of hashing D into table L, $L[\ell] = x$. We extend this notion to include cases where only a subsequence $S \subset D$ of the data is hashed, in which case the embedding assignments based exclusively on S should be understood to be possibly incorrect when all of D is processed.

We will be analyzing how D is hashed into a table L of size n under three models: uniform hashing, a generalization of double hashing where fully random hash functions are used, and the same double hashing model where the hash functions are constructed from a family of (ψ)-wise independent universal hash functions.

Definition 4: The models UH, DH, DH_ψ, and DH_∞^ψ.

- In UH, the probe sequence $p(x,*)$ is an infinite family of independent random variables that are uniformly distributed over $[0, n-1]$. Any collection of sequences $p(x_1,*), p(x_2,*), \ldots, p(x_n,*)$ are mutually independent, for distinct x_i.

- DH relaxes the requirement that each individual probe sequence be fully random.

  1) Each probe sequence $p(x,*)$ exhibits approximate pairwise independence:

     $\forall x\ \forall i,\ \forall r \in [0, n-1]:\quad \text{Prob}\{p(x,i) = r\} = \frac{1}{n-O(1)}.$[†]

[†]We use the Big-Oh notation in the following standard way: $f = g + O(h)$ means that $|f - g| = O(|h|)$. Consequently, there is no distinction between $f + O(g)$ and $f - O(g)$. Nevertheless, we will occasionally use minus signs to suggest that the worst case error is negative, or to write, say, $\frac{1}{1+O(h)} = 1 - O(h)$, for $h = o(1)$.

$$\forall x\ \forall i,j\ i \neq j\ \forall r,s \in [0,n-1]: \quad \mathrm{Prob}\{p(x,i)=r,\ p(x,j)=s\} = \tfrac{O(1)}{n^2}.$$

2) Furthermore, the random sequences $\{p(x,*)\}_{x\in D}$ are mutually independent. This condition need only hold for a subset of hash functions $\widehat{F} \subset F$, where $\widehat{F}$ depends on D, and $\frac{|\widehat{F}|}{|F|} > 1 - \frac{O(1)}{n^2}$.

3) In addition, we have the following robustness requirements.

i) Extremely long probe sequences are rare: For a fixed c_0 that depends on α,

$$\forall x: \sum_{t>c_0 n} \mathrm{Prob}\{|\cup_{i=1}^{t}\{p(x,i)\}| < \alpha n\} = \tfrac{O(1)}{n}.$$

ii) Probe sequences are unlikely to reprobe locations too frequently.

$$\forall x\ \forall j<k,\ \forall h,\ \forall r \in [0,n-1]: \ \mathrm{Prob}\{p(x,j)=p(x,k),\ p(x,h)=r\} = \tfrac{O(1)}{n^2}.$$

$$\forall x\ \forall h<i, j<k,\ (h,i)\neq(j,k): \ \mathrm{Prob}\{p(x,h)=p(x,i),\ p(x,j)=p(x,k)\} = \tfrac{O(1)}{n^2}.$$

For DH_ψ, three slight variants are worth defining. For the few instances where it is necessary to distinguish among them, we use the notation $DH_\psi^{(a)}$, $DH_\psi^{(b)}$ and $DH_\psi^{(c)}$. Common characteristics will be attributed to DH_ψ.

- In DH_ψ, the statistical probe behavior of an individual probe sequence is subject to the same requirements as in DH for the first ψ probes, the global coverage requirement must still hold, and the joint distribution of initial probe sequences, for collections of ψ or fewer probes, is required to exhibit statistical independence, for distinct keys. More precisely, we have the following.

1) $\forall x\ \forall i,\ \forall r \in [0,n-1]: \quad \mathrm{Prob}\{p(x,i)=r\} = \frac{1}{n-O(1)}.$

$$\forall x\ \forall i,j \le \psi\ i\neq j\ \forall r,s \in [0,n-1]: \quad \mathrm{Prob}\{p(x,i)=r,\ p(x,j)=s\} = \tfrac{O(1)}{n^2}.$$

2a) Knowing the probe values for a small set of keys, in $DH_\psi^{(a)}$, gives no information about the probes for another key. Formally, let $\{z_j\}_{j=1}^{\zeta}$ be a set of $\zeta \le \psi/2$ distinct keys. Then

$$\mathrm{Prob}\{p(z_\zeta,j)=a_j, j=1,2,3,\ldots \,\big|\, p(z_\ell,j),\ 1\le \ell<\zeta,\ j=1,2,\ldots\}$$
$$= \mathrm{Prob}\{p(z_\zeta,j)=a_j, j=1,2,3,\ldots\}.$$

2b) For $DH_\psi^{(b)}$, the same statement holds, but the probe index j is uniformly bounded by

$\kappa \log n$ for a sufficiently large constant κ:

$$\text{Prob}\{p(z_\zeta, j) = a_j, j \le \kappa \log n \;\Big|\, p(z_\ell, j),\ 1 \le \ell < \zeta,\ j \le \kappa \log n\} \\ = \text{Prob}\{p(z_\zeta, j) = a_j, j \le \kappa \log n \ldots\}.$$

2c) In $DH_\psi^{(c)}$, the listing of probe values is for sequences of consecutive probes that are altogether bounded by ψ. Formally, let Z be a set of $\zeta \le \psi/6$ distinct keys with associated probe bounds $\beta_1, \beta_2, \ldots, \beta_\zeta$, where $\sum_j \beta_j \le \psi$. Then

$$\text{Prob}\{p(z_\zeta, j) = a_j,\ j \le \beta_\zeta \;\Big|\, p(z_\ell, i),\ i \le \beta_\ell,\ \ell \le \zeta - 1\} = \text{Prob}\{p(z_\zeta, j) = a_j, j \le \beta_\zeta\}.$$

The conditions given in part 2) need only hold for a subset of hash functions $\widehat{F} \subset F$, where $\widehat{F}$ depends on D, and $\frac{|\widehat{F}|}{|F|} = 1 - \frac{O(1)}{n^2}$.

3) For some fixed c_0 that depends on α, $\forall x:\ \sum_{t > c_0 n} \text{Prob}\{| \cup_{i=1}^t \{p(x,i)\}| < \alpha n\} = O(\frac{1}{n})$.

4) $\forall x\ \forall j < k \le \psi,\ h \le \psi,\ r \in [0, n-1]:\ \text{Prob}\{p(x,j) = p(x,k),\ p(x,h) = r\} = \frac{O(1)}{n^2}$.

5) $\forall x\ \forall h < i \le \psi, j < k \le \psi,\ (h,i) \ne (j,k):\ \text{Prob}\{p(x,h) = p(x,i),\ p(x,j) = p(x,k)\} = \frac{O(1)}{n^2}$.

6) In $DH_\psi^{(b)}$, there is little chance that a probe sequence will revisit lots of locations:

$$\text{Prob}\{| \cup_{j=1}^{\kappa \log n} \{p(x,j)\}| < \tfrac{\kappa \log n}{2}\} = O(\tfrac{1}{n^3}),\ \text{where } \kappa \text{ is as in 2b)}.$$

In $DH_\psi^{(c)}$, related events of this type are even less likely. Formally, let $\mathcal{Z} = \{y_j\}_{j=1}^{\zeta}$ be any set of $\zeta < \frac{\psi}{6}$ keys with associated probe bounds $\{\beta_j\}_{j=1}^{\zeta}$. Let $m \ge 1$ and the probe bounds satisfy $\sum_j \beta_j + 3m + 3 \le \psi$. Let $S = \cup_j \cup_{i \le \beta_j} \{p(y_j, i)\}$, and suppose that $x \notin \mathcal{Z}$. Then

$$\text{Prob}\{| \cup_{1 \le i \le 3m+3} \{p(x,i)\} - S| \le m \ \ | \, p(y_i, j),\ j \le \beta_i,\ i \le \zeta\} = O(\tfrac{1}{n^{m+3}}).$$

For DH, requirement 1) is worded to include both UH and standard double hashing, which happens to generate probe sequences $p(x,1), p(x,2), \ldots$ that are permutations. Similarly, all of DH is included in $DH_\psi^{(a)}$ and $DH_\psi^{(b)}$.

Requirement 2) is needed for inclusion-exclusion arguments with limited independence, and especially for applications of the Chernoff-Hoeffding bound given in Lemma A2 of the Appendix. The statement says nothing about how random the probe sequencing is for any individual key. Rather, it says that if you have a bunch of distinct keys, each has statistics that, within various limits, appear to be independent of each other. Thus, these formulations define kinds of limited key-wise independence. The variations among the $DH_{\psi}^{(*)}$ models deserve some further explanation, which is momentarily deferred.

The robustness requirements 3), 4), and 5) replace the (stronger) requirement that probe sequences be permutations. Evidently, some form of robustness is necessary to guarantee that hash functions do not fail. Consider the damage that would occur if the offset function $d(x)$, for standard double hashing, were allowed, for example, to be zero even with the tiny probability $\frac{1}{n^3}$: with probability $\frac{1}{n^3}$ the number of probes needed to insert x_i becomes ∞ and so does the expected probe count. Such degenerate functions must therefore be excluded from DH and DH_{ψ}. Mathematically, condition 3) ensures that the hashing cannot fail because αn locations will eventually be probed. Similarly, conditions 4) and 5) are technical requirements to ensure that probe locations are not revisited so often that they spoil the overall performance.

We also need to ensure algorithmic (as well as statistical) robustness for generalized hashing schemes so that they satisfy the requirements of DH_{ψ}. In particular, criterion 3), which requires that the insertion strategy cannot fail, demands algorithmic robustness. The easiest fix is to modify the hashing procedure when, say, a key's first $c \log n$ probes, for suitably large constant c (or first $n^{1/3}$ or first $O(n)$ probes) have failed to find a vacant location. A suitable strategy would be to switch to linear probing (where $p(x,j) = f(x) - j + 1 \bmod n$, for a random f), which would reduce the probability of failure to zero, and satisfy our global robustness requirement. We could even set $f \equiv 0$, in this case. Alternatively, one could select new random seeds and rehash the entire data set.

For the purposes of implementing programmable forms of true double hashing with limited independence, $DH_{\psi}^{(a)}$ and $DH_{\psi}^{(b)}$ are perfectly reasonable; the function pair $(f(x), d(x))$ can be defined to be $(\frac{\psi}{2})$-wise independent with domain U and range $[0, n-1] \times [1, n-1]$. The $DH_{\psi}^{(b)}$ model is intended to accommodate algorithmic robustness, which might change the probe sequence when too many probes have been unsuccessful. In reality, this change is insignificant. It will turn out that the probability of a prolonged insertion failure is $O(\frac{1}{n^2})$. Error terms will show that the robust requirements for DH_{ψ} ensure the additional expected probe count is just $O(\frac{1}{n})$, since these failures are so rare.

$DH_\psi^{(c)}$ is intended to include models of uniform hashing with limited independence. True double hashing does not satisfy specification 6) because, for example, the first $3m+3$ probes for x could all belong to S with a probability in excess of $\frac{1}{n^2}$. On the other hand, it is not difficult to see that 6) is a fairly mild constraint if the $3m+3$ probes are taken to be independent of S, uniformly distributed, and independent of each other. The purpose of criterion 6) is to prevent pairwise independent probe sequences from using up their independence and becoming deterministically malicious, (which is to say that they would use up their two degrees of freedom and then, say, follow the probe sequence of $x_{\alpha n}$). Although there are weaker formulations that yield the same performance bounds, the suite of DH_ψ characterizations as given would seem, for most practical (and many impractical) purposes, adequately general.

The differences among the DH_ψ models also serve to highlight a distinction between independence and randomness. In true double hashing, the randomness per key comprises a modest two degrees of freedom, but the independence across different keys (keywise independence) is fairly strong. Uniform hashing, however, attributes independence to statistical randomness. In pragmatic terms, this suggests that the degrees of freedom that are provably adequate for uniform hashing will exceed the statistical randomness needed to establish comparable performance bounds for double hashing. Conditions 2c) and 6) formalize these additional randomness/independence requirements in a way that can be used in a performance analysis.

There is also a kind of non-uniformity and mild ambiguity that should be resolved for DH_ψ. Technically, a statement such as $p(x,k) = p(y,j)$ defines one constraint, but will require two degrees of freedom in our formal accounting of the randomness for $DH_\psi^{(c)}$. In this case, we would model the event as $\bigvee_r \langle p(x,k) = r \bigwedge p(y,j = r)\rangle$. Consequently, we will allocate two degrees of freedom from the count ψ for such an event in $DH_\psi^{(c)}$. For $DH_\psi^{(a)}$ and $DH_\psi^{(b)}$, only one degree of freedom will be needed, due to the availability of high independence. It turns out that these differences only play a role (and a minor one at that) in Lemma 2. Consequently, we will take the most conservative accounting for ψ, which is for model $DH_\psi^{(c)}$.

More importantly, we note that any family of hash functions $\mathcal{F}_\psi \in DH_\psi$ has a corresponding family $\mathcal{F} \in DH$ where the probability products that correspond to the conjunction of statistically independent events are exactly the same for $\mathcal{F}_\psi$ and $\mathcal{F}$. The family in DH is created by selecting, for each $x \in D$, a new function $f_x \in \mathcal{F}_\psi$, so that $p(x,j) = f_x(x,j)$. If, in DH_ψ, property 2 holds only for a subset $\widehat{F}$ of the hash functions (depending on D), then this set should be used to define the hash functions in DH_∞^ψ. Thus, the model permits the hash functions to depend on D, but only in a very weak way. In any case, the correspondence between DH_ψ and DH_∞^ψ simplifies the arguments for DH_ψ. We will show that all of DH and UH have, in terms of our error bound $\epsilon + O(\frac{1}{n})$, the same performance. To extend this conclusion to DH_ψ, we need (and sometimes will) only show that each family in DH_ψ is comparable to its representative in DH.

For specificity, we define DH_∞^ψ as follows.

- Given D and $\mathcal{F} \in DH_\psi$, let DH_∞^ψ be this special model in DH, which is constructed by independently selecting a hash function from $\mathcal{F}$ (or a suitable $\widehat{\mathcal{F}} \subset \mathcal{F}$) for each hash key in D.

The statistical differences between DH_ψ and DH_ψ^∞ will be understood as being between corresponding families.

The independence ψ needed for the performance analyses will turn out to be $O(\log n)$ in all cases. An immediate consequence of this work is that standard double hashing will achieve near optimal performance if, for example, the probe and offset functions f, d are chosen from an *effectively ψ-wise independent* family of hash functions with $\frac{1}{n^2}$-tolerance. For example, an adequately independent family is given by $f \in F_{\psi,n} \circ g$, and $(d-1) \in F_{\psi,n-1} \circ g$, where g is a random function in F_0^4. Subject to the caveats needed to ensure the absence of failure, uniform hashing will achieve near optimal expected performance for the probe functions $p(x,j) = f(j + ng(x))$, for $g \in F_0^k$, and $f \in \bar{F}_{\psi,n}$, where $\bar{F}_{\psi,n} : [0, n^{k+1}-1] \to [0, n-1]$, for, say $k = 4$ as in Section 1.1. The function families $F_{\psi,n}$, $F_{\psi,n-1}$, and $\bar{F}_{\psi,n}$ could be the universal hash functions presented in Section 1.0, or the constant time functions of [16].

We have already seen that the presence of irregular hash functions, such as the small $1/n^2$ fraction in F_0^4 that have collisions on D are insignificant. We now drop all reference to them since they induce an expected probe cost of $O(1/n)$.

To analyze how the ordered data set D hashes into the table, we introduce a family of directed graphs to model the structure of the collision events.

Definition 5: The dependency graph $G(D)$**.**

Given a sequence D of hash keys, we say that a hashing of D defines a directed *dependency graph* $G(D)$ as follows. Before any elements are hashed, the vertex set of $G(D)$ is D and the edge set is empty. Then D is hashed element-by-element in order. Suppose that when inserting x into the hash table, x is placed in its k-th probe location l_k after probing $l_1, \ldots, l_{k-1}$. We add a directed edge from x to each of the keys $x_{T_1}, \ldots, x_{T_{k-1}}$ residing in table locations $l_1, \ldots, l_{k-1}$. Each edge is labeled with its corresponding probe number (i.e., an edge from x to $y = L[p(x,j)]$ has the label j), and each vertex $x_i = D_i$ bears the label i.

Notice that $G(D)$, despite extensive labeling, bears no information to indicate where nodes are embedded. (However, intermediate calculations will sometimes reference specific embedding locations.)

Definition 6: The dependency graph $G(x, D)$**.**

- The *dependency graph* of x in D, $G(x, D)$, is the restriction of $G(D)$ to the vertex set comprising x and all nodes reachable from x in $G(D)$. Its edges and vertices are both labeled.
- The *dependency set* of x, $dep(x, D)$, is the vertex set of $G(x, D)$.

It is convenient to refine this graph nomenclature to include every prefix of a probe sequence.

Definition 7: Partial dependency graphs $G_r(x, D)$**.**

- The *partial dependency* graphs of x_i are the $probe_i$ subgraphs of $G(x_i, D)$, where the r-th subgraph $G_r(x_i, D)$ comprises the portion of $G(x_i, D)$ that is reachable from the first r probes for x_i. Thus, $(\{x_i\}, \emptyset) = G_0(x_i, D) \subset G_1(x_i, D) \subset \cdots G_{probe_i - 1}(x_i, D) = G(x_i, D)$.

It should be noted that the graphs $G_r(x, D)$ and $G_{r+1}(x, D)$ might have the same vertex set and only differ in the outdegree of the root x.

- The vertex set of $G_r(x, D)$ is denoted by $dep_r(x, D)$.

The (random) set of all partial dependency graphs of x that occur from the hashing of D is denoted by $\mathcal{G}^*(x, D)$. The vertices in each of these graphs have labels in $[1, |D|]$. We relabel the vertices of any $G = (V, E), G \in \mathcal{G}^*(x, D)$, in the unique order preserving way to $1, 2, \ldots, |V|$. Let $\mathcal{G}(x, D)$ be the resulting set of relabeled graphs. Since each of the partial dependency structures that occur are different, $|\mathcal{G}(x, D)| = |\mathcal{G}^*(x, D)|$.

It is worth pointing out that partial dependency graphs will be used in two different contexts. First, they will be used as the actual dependency DAGs that occur, which is consistent with the definitions above, and the first application as displayed below. Second, they will be used in the form of candidate dependency DAGs, which are structures that could, with some probability, be an actual dependency DAG that occurs. In this case, issue will be how to estimate this probability.

These definitions provide immediate formulations for the expected number of probes to insert x_i, as a function of its dependency graph. We will count the expected number of partial dependency DAGs rooted at $x_{\alpha n}$. Suppose that $x_{\alpha n}$ has r exiting edges in the DAG $G(x_{\alpha n}, D)$. Then x will be the root of $r + 1$ partial dependency DAGs. (The first will have zero children since we do not require the root to be inserted when counting these structures.) Evidently, the number of such partial DAGs with root $x_{\alpha n}$ is precisely the number of probes needed to insert the key. Formally,

$$\begin{aligned} \mathrm{E}[probe_i] &= \sum_{k \geq 0} \mathrm{Prob}\{dgr(G(x_i, D)) \geq k\} \\ &= \mathrm{E}[|\mathcal{G}(x_i, D)|]. \end{aligned}$$

The probability that a given DAG structure G is a partial dependency graph for $x_{\alpha n}$ is determined by two compound events. First, the local hashing must exhibit the prescribed collision behavior to map a sequence $S = (x_{T_1}, x_{T_2}, \ldots, x_{T_{k-1}}, x_{\alpha n})$ onto G. Second, each of these vertices (apart from $x_{\alpha n}$) must end up in the same location regardless of whether only S or all of D is hashed. The probability of the first event can be computed quite easily, whereas the latter is considerably the more difficult to bound, since it concerns all of D. Moreover, estimating the probability that a sequence S is the right one involves estimating the probability that various locations are vacant at certain insertion times, which is our original performance estimation problem. There is, however, one critical difference: this formulation permits us to overestimate the probability that a specific DAG is the right one by overestimating the probability that the candidate vertices are hashed to vacant locations. The resulting expected number of such S gives an overestimate for $\mathrm{E}[probe_i]$.

The following definitions support the first step of this estimation process.

Definition 8: The probabilities $P(k,j)$ and $\widetilde{P}(k,j)$**.**

- Let $P(k,j)$ be the probability that a partial dependency graph of x_j, (the jth item to be hashed), contains exactly k vertices: $P(k,j) = \mathrm{Prob}\{|dep_r(x_j,D)| = k\}$, for some r.
- Let $\widetilde{P}(k,j)$ be the expected number of partial dependency graph of x_j that contain exactly k vertices: $\widetilde{P}(k,j) = \sum_r \mathrm{Prob}\{|dep_r(x_j,D)| = k\}$.

The technical reason for defining the $\widetilde{P}(k,j)$ as an expected number as opposed to a probability is that a single dependency graph can have two partial dependency graphs with the same number of vertices, due to a collision between x_j and some node already within its dependency graph. The $\widetilde{P}(k,j)$ also facilitate the following formulation.

Lemma 1. $\mathrm{E}[probe_j]$, the expected number of probes needed to insert the jth element x_j, equals $\sum_{0<k} \widetilde{P}(k,j)$.

Proof:

$$\sum_{0<k} \widetilde{P}(k,j) = \mathrm{E}[|\mathcal{G}(x_i,D)|]. \quad \blacksquare$$

Unfortunately, $\widetilde{P}(k,j)$ might be a little unruly in DH, because of the possibility of reprobing earlier probe locations. Accordingly, we account for such events as follows.

Definition 9.

Let $Err_0(j)$ be the expected number of probes from x_j to vertices already belonging to partial dependency sets of x_j: $Err_0(j) = \sum_{r=2}^{probe_j - 1} \mathrm{Prob}\{dep_r(x_j,D) = dep_{r-1}(x_j,D)\}$.

Lemma 1 can now be restated as follows.

Corollary 1.

$$\mathrm{E}[probe_j] = Err_0(j) + \sum_{0<\ell} P(\ell,j). \quad \blacksquare$$

Remark 1.

Note that the probabilities $P(\ell, j)$ (and, in fact, all performance statistics) are defined with respect to a universal class of hash functions. These probabilities are not, of course, exactly the same for all function families in DH_ψ, nor DH. However, since $UH \subset DH \subset DH_\psi^{(*)}$, where $* \in \{a, b\}$, and $UH \subset DH_\psi^{(c)}$, Corollary 1 shows that DH_ψ would be guaranteed to provide optimal performance if $P(\ell, j)$ and $Err_0(j)$ were shown to be asymptotically the same for all families DH_ψ. In the next subsection, we establish that $P(k, k)$ is indeed essentially the same for all members of DH_ψ, provided $\psi \geq 6k$.

2.1.1 The importance of $P(k, k)$

The value $P(k, k)$ is of special interest because the event $(|dep(x, D)| = k)$ corresponds to the existence of a subset $\delta \subset D$ of k keys that includes x, and which has the collision behavior $|dep(x, \delta)| = k$. The event $dep(x, \delta) = \delta$, as the next lemma will show, is nearly independent of the hashing scheme and specific keys being inserted. But before analyzing the probability distribution of dependency DAGs, we need a standard traversal procedure to extract a unique spanning tree from each DAG.

Definition 10.

The *spanning tree* of a rooted dependency DAG $G = (V, E)$ (such as $G(x, D)$) is the tree $Tr = (V, E_{Tr})$, where the tree edge into each vertex $v \in V$ is the incoming edge in E that originates from the vertex key with the highest numbered index in V. If x_i has multiple probes to x_j, and one of the edges should be a tree edge, let the edge with the lowest probe number be the tree edge. Thus, the spanning tree is created by a search of G where the vertices are scanned in order of decreasing indices in D.

Of course, any definition of a spanning tree will suffice. Lemma 2 below begins the task of counting the expected number of tree-like structures that occur under various hashing constraints. Although the counting is elementary, a few preliminary comments might be appropriate, since the underlying freedom for a given probe sequence is severely constrained by our definitions of DH and DH_ψ.

Suppose $D = (x_1, x_2, x_3, x_4)$, and we want to compute the probability that the dependency graph is a tree with root x_4 and the three children x_1, x_2 and x_3. There are several ways to represent this event.

A formal formulation might read: $\exists r_1, r_2, r_3$ and permutation π of (1,2,3):

$$p(x_4,\pi_1) = r_1,\ p(x_1,1) = r_1,\ p(x_4,\pi_2) = r_2,\ p(x_2,1) = r_2,\ p(x_4,\pi_3) = r_3,\ p(x_3,1) = r_3.$$

Additional constraints would require that r_1, r_2, r_3 be distinct.

The probability of such joint events can be bounded in DH_ψ by evaluating formulations where no key has more than two constraints. For example,

$$\begin{aligned}
&\text{Prob}\{\exists r_1, r_2, r_3 : \bigwedge_{i=1}^{3}((p(x_4,i) = r_i)\bigwedge(p(x_i,1) = r_i))\} \\
&\quad = \sum_{(r_1,r_2,r_3)\in[0,n-1]^3} \text{Prob}\{\bigwedge_{i=1}^{3}((p(x_4,i) = r_i)\bigwedge(p(x_i,1) = r_i))\} \\
&\quad = \sum_{(r_1,r_2,r_3)\in[0,n-1]^3} \text{Prob}\{\bigwedge_{i=1}^{3}(p(x_4,i) = r_i)\}\text{Prob}\{\bigwedge_{i=1}^{3}(p(x_i,1) = r_i) \mid \bigwedge_{i=1}^{3}(p(x_4,i) = r_i\} \\
&\quad = \sum_{(r_1,r_2,r_3)\in[0,n-1]^3} \text{Prob}\{\bigwedge_{i=1}^{3}(p(x_4,i) = r_i)\}\prod_{i=1}^{3}\text{Prob}\{p(x_i,1) = r_i\} \\
&\quad = (\frac{1}{n+O(1)})^3 \sum_{(r_1,r_2,r_3)\in[0,n-1]^3} \text{Prob}\{\bigwedge_{i=1}^{3}(p(x_4,i) = r_i)\} \\
&\quad = (\frac{1}{n+O(1)})^3,
\end{aligned}$$

where we used limited independence to replace $\text{Prob}\{\bigwedge_{i=1}^{3}(p(x_i,1) = r_i) \mid \bigwedge_{i=1}^{4}(p(x_4,i) = r_i\}$ with $\prod_{i=1}^{3}\text{Prob}\{p(x_i,1) = r_i\}$. Thus, we attain the right bound without knowing anything about the distribution for the first three probes of x_4. (Of course, this bound is a slight overestimate for the probability that the specific tree structure tree occurs, since the formulation did not require that r_1, r_2, and r_3 be distinct.) More generally, each tree edge is a constraint about a pair of probes, and the randomness in the destination vertex is used to account for the $\frac{1}{n}$ probability attributable to the edge. But each such edge requires (in our rather conservative model) two degrees of freedom from ψ.

The accounting for DAGs that are not tree-like is more complex. The difficulty is that in $DH_\psi^{(a)}$, for example, no event with four keys can be computed to have a probability that is less than $O(n^{-8})$, whereas the number of different multi-collision structures that could occur is infinite, since locations can be reprobed any number of times. The solution is to group the structures into sets.

For example, it is possible that key x_3 satisfies $p(x_3, 1) = p(x_1, 1)$, and $p(x_3, 2) = p(x_2, 1)$. These two events give a multiplicative factor of $(\frac{1}{n+o(1)})^2$ to the behavior of x_3 even if we drop the requirement that x_3's final location be in the predetermined table index dictated by root x_4. So the groupings can even include hashings that do not even form DAGs. We will, however, insist that the specific structures be dependency DAGs even if the probability estimates are so weak that they might include other structures as well.

More generally, we have the following lemma, which asserts that the local dependency DAGs, as unembedded structures, have associated probabilities that are virtually identical in all of DH_ψ.

Lemma 2. In DH_ψ, for $\psi \geq 6k$, $P(k, k)$ is within a factor of $(1 + O(k^3/n))$ of the same value for all k-tuples of distinct keys in D and any family that satisfies the requirements of DH_ψ. More precisely: suppose $\psi = O(\log n)$, $6k \leq \psi$, and $S = (S_1, S_2 \ldots S_k)$ is a sequence of k distinct elements in U. Let a dependency DAG be degenerate, in $DH_\psi^{(b)}$, if it contains a vertex with an outdegree of at least $\kappa \log n$. Let there be no degenerate DAGs in the other models. Then:

1) Let Tr be a dependency tree of k vertices. Then the probability that the local hashing of S yields the dependency structure Tr is, in DH_ψ, $\frac{1+O(k^2/n)}{n^{k-1}}$.

2) The set of all non-degenerate k-vertex dependency DAGs can be partitioned into equivalence classes so that each class is represented by a unique spanning tree that is contained as a subgraph of each element in the corresponding class, and the probability that S is hashed into a structure that is not a tree but belongs to the class corresponding to a given dependency tree Tr is, in DH_ψ, bounded by $O(k^3/n^k)$.

Proof: Let $\widehat{S} = (\widehat{S_1}, \widehat{S_2}, \ldots, \widehat{S_k}) = (S_k, S_{k-1}, \ldots, S_1)$, so that $\widehat{S}$ is S in reverse order. The lemma will be established by identifying all possible embeddings of Tr, and estimating the probability that the hashing of S builds a dependency structure with a particular spanning tree isomorphic to Tr. By definition, the vertex set for Tr is $\{1, \ldots, k\}$, and the adjacency lists are ordered. Let, for $j =$

$1, 2, \ldots, k$, h_j be the outdegree of vertex j in Tr. Let the intended mapping (graph subisomorphism) between $G(S)$ and Tr map $\widehat{S_j}$ to j. The statistical hashing of S into a dependency DAG will proceed one node at a time.

For specificity and uniformity of exposition, we will embed the (potential) root $\widehat{S_1}$, but will impose no constraint for the location. It will be evident that this embedding will have a nominal asymptotic effect on the overall probabilities for the embedding of all other nodes of $\widehat{S}$: the same formulations would result if this first embedding step were omitted.†

1) We begin by considering the case where $G(S)$ has no non-tree edges. First, the root $\widehat{S_1}$ is embedded in L. Tr prescribes that the root $\widehat{S_1}$ must be embedded in its $(h_1 + 1)$-st probe location, which is any one of n target locations, and then the h_1 children embeddings are specified. Whenever a vertex is embedded, its target embedding location will have been prespecified (unless the vertex is the root). Its unsuccessful probe locations are used to specify the target destinations for the subsequent embedding of its children. The probability of the actual embeddings are computed in the order prescribed by $\widehat{S}$. The probability that these first h_1+1 probes are to distinct locations is between 1 and $1-|O(h_1^2/n)|$. Subsequent node embedding will have two constraints: a node with h_j children must have its $(h_j + 1)$-st probe location prespecified, and the first h_j probes must be to h_j distinct locations that have not yet become specified targets for embedding. Let R_j be the set of target embedding locations that are specified prior to the embedding of $\widehat{S_j}$, and let r_j be the location for $\widehat{S_j}$.

   The probability that the hashing of a node $\widehat{S_j}$ satisfies its embedding specification and yields, as prescribed by Tr, h_j children specifications is:

$$\operatorname{Prob}\left\{(p(\widehat{S_j}, h_j + 1) = r_j) \bigwedge_{1\le i<\ell\le h_j} (p(\widehat{S_j}, i) \ne p(\widehat{S_j}, \ell)) \bigwedge_{1\le i\le h_j} (p(\widehat{S_j}, i) \notin R_j)\right\},$$

†This analysis of trees with embedded roots will not be for naught; Lemma 5 uses the statistics for dependency trees whose roots are embedded at specific locations.

which is at most $\text{Prob}\{(p(\widehat{S}_j, h_j+1) = r_j)\} \le \frac{1+O(1/n)}{n}$, and at least

$$\text{Prob}\left\{(p(\widehat{S}_j, h_j+1) = r_j)\right\} - \sum_{1\le i<\ell\le h_j} \text{Prob}\left\{(p(\widehat{S}_j, h_j+1) = r_j), (p(\widehat{S}_j, i) = p(\widehat{S}_j, \ell))\right\}$$
$$- \sum_{\substack{1\le i\le h_j\\ r\in R_j}} \text{Prob}\left\{(p(\widehat{S}_j, h_j+1) = r_j), (p(\widehat{S}_j, i) = r)\right\},$$

which is bounded below by $\frac{1}{n+O(1)} - \frac{O(h_j^2)}{n^2} - \frac{O(kh_j)}{n^2}$. We used our assumption of local robustness to derive the second term and approximate pairwise independence to derive the third term. We appeal to the independence of individual probe sequences to multiply all k factors to get a value between $(1 - \frac{|O(k^2)|}{n})(\frac{1}{n})^{k-1}$ and $(\frac{1}{n})^{k-1}(1 + |O(k/n)|)$, which proves 1).

2) If $G(S)$ is to have non-tree edges, then we must bound the probability that the hashing produces a DAG that has Tr as its canonical spanning tree as formalized by Definition 10. The probability considerations are similar to those in the preceding case, but some nodes $x \in \widehat{S}$, will have gaps in their probe sequences $p(x,1), p(x,2), \ldots$ to their tree children, since edges to nodes that are already embedded or that have embedding locations already specified will be omitted. In this setting, Tr only prescribes the tree edges emanating from each vertex of the DAG. Of course, any k keys have initial probe sequences that are mutually independent of each other as long as the total number of probes is bounded by ψ, in $DH_\psi^{(c)}$. The other DH_ψ models, will be seen to require less independence. So first suppose that the total number of probes is at most ψ. In this case, the probability that V is hashed into a DAG that (is Tr or) contains Tr as its spanning tree is at most $\prod_{j=1}^{k} pr_j$, where pr_j *overestimates* the probability that the j-th vertex is hashed to have the correct number of probes to locations that are available for its spanning tree children, and has an embedding location that satisfies its prescribed constraint.

To bound pr_j, we distinguish among three subcases: i) $\widehat{S}_j$ has no non-tree edges, ii) $\widehat{S}_j$ has fewer than h_j+2 non-tree edges and at least one, and iii) $\widehat{S}_j$ has at least h_j+2 non-tree edges. Note that if no location can appear twice in a probe sequence — as is the case for real double hashing — then ii) and iii) can be combined into the event that $\widehat{S}_j$ has at least one and at most $k-1$ non-tree edges.

i) This subcase has the same analysis as 1), and contributes a probability of at most 1 to pr_1, and at most $\frac{1}{n}(1+O(1/n))$ to pr_j, for $j>1$.

ii) In this subcase, there are different DAG structures, depending on which probe count within $(h_j+2,\ldots,2h_j+2)$ is the last and actually embeds $\widehat{S_j}$. Summing over all possible last probe counts, over the possible probe counts that might† correspond to the first non-tree edge, which must be among the first h_j+1 probes, and the set of possible destinations for this first non-tree edge, (which must be to a location already probed by $\widehat{S_j}$ or some other item in S), we get $\frac{O((h_j+1)(h_j+1)k)}{n^2}$ as an overestimate for the probability contributed to pr_j by case 2.

iii) In this subcase, there must be two consecutive non-tree edges among the first $2h_j+2$ probes of $\widehat{S_j}$. These edges might go to previously embedded keys or collide with earlier probes of $\widehat{S_j}$. To estimate this contribution to pr_j, we ignore the predetermined embedding requirement for $\widehat{S_j}$, and simply overcount the probability that such a pair of probes could occur, which is bounded by $(2h_j+1)\frac{O(k^2)}{n^2}$. There are, however, a few possible complications that we defer to case iv).

Combining terms from the three cases into factors and multiplying gives the following overestimate for the probability that S hashes into a DAG that is represented by (or is) *Tr*:

$$\prod pr_j = \prod_{1<j\le k}\left(\frac{1}{n}+O\left(\frac{(h_j+1)k^2}{n^2}\right)\right) = \left(\frac{1}{n}\right)^{k-1}(1+O(k^3/n)).$$

Hence, the probability that S hashes into a non-tree DAG represented by the spanning tree *Tr* is at most $(\frac{1}{n})^{k-1}(1+O(k^3/n)) - (1-\frac{O(k^2)}{n})(\frac{1}{n})^{k-1} = O(k^3/n^k)$. Moreover, reasoning used to establish iii) shows that the probability that the k vertices form such a DAG where the root has more than $2k$ edges is $O(k^3/n^{k+1})$ in all three models of DH_ψ.

iv) These multiplicative formulations, however, are not valid if the total number of probes needed to embed the structure exceeds ψ. This issue is resolved differently for different models of

† Technically, the event

$\langle p(\widehat{S_j},i)$ *is the first non-tree probe,* $p(\widehat{S_j},\ell)$ *is to the prespecified embedding location for* $\widehat{S_j}$*, and there are exactly* h_j *probes among the first* ℓ *to as yet unspecified locations*$\rangle$

is outside of the kinds of events we can specify with two degrees of freedom. Instead, we use overlapping superset decompositions of the event space (rather than a true partition) of the form:

$\langle p(\widehat{S_j},i)$ *is to a previously specified location for some vertex in the DAG, and* $p(\widehat{S_j},\ell)$ *is to the prespecified location for* $\widehat{S_j}\rangle$.

DH_ψ. The multiplicativity even deserves some justification for $DH_\psi^{(a)}$. In this case, let the possible probe events for $\widehat{S_j}$ be partitioned into the discrete disjoint events $e_1, e_2, \ldots$, so that $\sum_i \text{Prob}\{e_i\} = \frac{O((h_j+1)(h_j+1)k)}{n^2}$. While we cannot assert how much an individual $\text{Prob}\{e_i\}$ is, it suffices to observe that for each e_i, the probability that the h_j children of $\widehat{S_j}$ meet the constraints implied by e_i is multiplicative, whence the probability that, say, a child $\widehat{S_*}$ has its embedding probe in the location prescribed by e_i is just $\frac{1}{n+O(1)}$, in $DH_\psi^{(a)}$. Thus, the multiplicativity of the probabilities for each child gives the desired probability over each e_i, whence summing gives the desired probabilities in aggregate. Formally, the argument is based on a straightforward (and omitted) structural induction over subtrees.

For $DH_\psi^{(b)}$ and $DH_\psi^{(c)}$, it is possible that a DAG could have so many non-tree edges that the allocated independence can be used up before all of a dependency set's embedding locations have been discovered. In $DH_\psi^{(b)}$, the problem has been eliminated by definition, which introduces certain statistical undercounts for the expected number of probes due to undercounted dependency DAGs. The expected error due to this simplification will be bounded by the term Err_b as introduced in Definition 12.

For $DH_\psi^{(c)}$, an amortized attribution of probabilistic cause is needed when the dependency DAG G comprises $|V| \le \frac{\psi}{6}$ vertices and $|E| > \psi$ probes. Given a candidate dependency set $S \subset D$, with $|S| = k$ and candidate dependency DAG with the graph representation $G = (V = \{1, \ldots, k\}, E)$, let $\mathit{Tr} = (V, E_{\mathit{Tr}})$ be the spanning tree for G as formalized by Definition 10. Let each edge $e \in E$ be classified as a tree edge when $e \in E_{\mathit{Tr}}$, or not, when $e \notin E_{\mathit{Tr}}$. Let $parent[j]$ give the parent vertex of vertex j with respect to Tr. Let $GAdj[j]$ be an array that contains the neighbors of vertex j in G sorted by increasing (probe) index for vertex j. Let $|GAdj[j]|$ be the number of edges exiting j in G. To bound the probability that S is hashed into a dependency DAG that is not a tree, we redefine the partitioning of all such DAGs into equivalence sets indexed by dependency trees. The difficulty with Tr is that important tree edges might be omitted from some subsets of initial probe sequences for each key. The following procedure begins with a candidate DAG G and its spanning tree Tr as previously defined. It then substitutes one tree edge for another to get a sequence of spanning trees with more suitable edges. It also determines the number of probes for each key to use in establishing the probability estimates for the hashing of $\widehat{S}$ into a non-tree DAG. These counts are stored in the array $Degrees[*]$.

for $j \leftarrow 1$ to k do $Degrees[j] \leftarrow 3h_j + 3$ endfor

repeat

  make all vertices untouched

  for $j \leftarrow 1$ to k do

    $childex \leftarrow 1$

    while $childex \leq Degrees[j]$ and $childex \leq |GAdj[j]|$ do

      $jchild \leftarrow (GAdj[j])[childex]$

      if $jchild$ not touched then

        make $jchild$ touched

        $e \leftarrow (j, jchild)$

        if $e \notin E_{Tr}$ then $\{make\ e\ a\ tree\ edge\}$

          $\ell \leftarrow parent[jchild]$

          remove $(\ell, jchild)$ from E_{Tr}

          include e in E_{Tr}

          $parent[jchild] \leftarrow j$

          $Degrees[j] \leftarrow Degrees[j] + 3$

          $Degrees[\ell] \leftarrow Degrees[\ell] - 3$

        endif

      endif

      $childex \leftarrow childex + 1$

    endwhile

  endfor

until no changes occur in $Degrees$

It is easy to see that this procedure, if it terminates, yields a spanning tree for G, satisfies

$Degrees[j] = 3h_j + 3$, where h_j is the outdegree of vertex j in the modified tree. The salient facts are: a) the edges in the DAG (as opposed to their classifications) do not change, and b) each edge reclassification replaces one tree edge by an alternative that has the same destination, so that the indegree of each vertex other than root vertex 1 remains at 1.

Termination is established as follows. Since the original spanning tree was constructed so that every neighbor of $\widehat{S_1}$ was attached to $\widehat{S_1}$ by a tree edge, $Degrees[1]$ can never increase at any iteration. All of the first $Degrees[1]$ probes from $\widehat{S_1}$ (including those that are non-tree probes) will be to children of $\widehat{S_1}$, but the count can be reduced by subsequent vertices adopting other children of $\widehat{S_1}$ for their own. Consequently, this count is non-increasing and must eventually stabilize. The induction hypothesis supposes that after some iteration, $Degrees[i]$, for $i \le \ell$ are all constant. Subsequently, $Degrees[\ell+1]$ might be changing, but must have some maximum value. At that point, $\widehat{S_{\ell+1}}$ declares all vertices in its first $Degrees[\ell+1]$ probes as its children, except for vertices already parented by the restricted probes of $\widehat{S_1}, \ldots, \widehat{S_\ell}$. Thereafter, $Degrees[\ell+1]$ can only decrease since it will never see any additional vertices available for adoption. It follows that its count must also stabilize at some iteration. Hence all $Degree[*]$ counts stabilize, and the algorithm must terminate.

Now suppose that, after all tree children reassignments have been finalized, $Degrees[j] > |GAdj[j]|$. Then all of j's probes are revealed within its allotment of $Degrees[j]$ probes, and case i), ii), or iii) is applicable to establish pr_j. But there is a mild catch that we momentarily defer. If, on the other hand, $Degrees[j] \le |GAdj[j]|$, then there are at most $\frac{Degrees[j]}{3} - 1$. "new" locations that j visits with its first $Degrees[j]$ probes, and which have not been visited by the restricted probes of keys $1, \ldots, j-1$. Hence the excessive probe bound of Definition 4.6) shows that the probability that $\widehat{S_j}$ hashes to achieve this characteristic, given the initial probe sequences of $\widehat{S_1}, \ldots, \widehat{S_{j-1}}$, is $O(\frac{1}{n^{Degrees[j]/3+2}})$. This value is so small that we can award a factor of $\frac{1}{n}$ to each of the h_j children of j in the tree, and give j the factor $O(\frac{1}{n^3})$. With this allocation of probabilities, it does not matter if the hashing of vertex j is independent of the hashing of its children or not. The children can be viewed as meeting their terminating probe specifications with probability 1. And j will still retain a factor of $O(\frac{1}{n^3})$ regardless of whether or not its hashing is independent of its locally determined embedding constraint.

The mild catch is that when $Degrees[j] > |GAdj[j]|$, the processing of the edge set by cases i), ii), or iii) will use one or two probes by j to locations that are prespecified by the probing of other keys. But if these one or two other keys do not have sufficient probe allotments to reveal the prespecified location(s), we cannot assert that the constraint(s) are independent of the hashing of vertex j. The solution is straightforward. Think of a vertex as owning probe locations rather than children. Let x be an abstract hashing location that would be prespecified by a probe from x's parent — name it vertex i — and suppose that $Degrees[i] \leq |GAdj[i]|$. Let Σ be the collection of all probes to x that are selected to define probability constraints as described for the cases i), ii), or iii), which requires that the source vertices issuing these probes have *Degree* counts that exceed their outdegrees in G. Thus, the vertex that is supposed to reside in location x as defined by Tr would have its last probe belonging to Σ if its probes as recorded in G satisfy the requirements of cases i) or ii). Other vertices can have selected probes to location x based on cases ii) or iii). (Σ was defined in terms of probes rather than keys because a key could have two constraining probes to the same location. Thus, a probe in Σ should comprise the vertex name and its probe number for each probe to x in Σ.)

We use the probe in Σ that originates from the key with the smallest index (in the reverse ordering of keys, and smallest probe number, if the key has two such probes in Σ) to specify the location constraint for the $|\Sigma| - 1$ other probes in Σ. Σ will now represent $|\Sigma| - 1$ real constraints, which is one too few. But the missing factor of $\frac{1}{n}$ is available from x's owner — vertex i — due to its excessive probe characteristic. A more detailed argument would retrace the same reasonings given for cases i)—iii), but now applied to the members of Σ.

It is now easy to estimate the probability that $\widehat{S}$ hashes into any of the DAGs that results with the spanning tree Tr, in the case where $|E| > \psi$, but $|V| = k \leq \frac{\psi}{6}$. There are four cases per vertex. If $|GAdj[j]| < Degrees[j]$, then case i), ii), or iii) is applicable, and requires at most $Degrees[j]$ degrees of freedom for the hashing of $\widehat{S_j}$. If not, the rare probe requirement of Definition 4.6) for $DH_\psi^{(c)}$ will apply (and use $Degrees[j]$ degrees of freedom). Of course, the value for $Degrees[j]$ is defined by the final spanning tree computed for each DAG. Combining all of these events gives a bound of

$$p_j = (\frac{1}{n} + O(\frac{h_j k^2}{n^2}) + O(\frac{1}{n^3})),$$

and hence the probability that S hashed into a non-tree DAG mapped to Tr is at most

$$(\frac{1}{n})^{k-1}(1+O(k^3/n)) - (\frac{1}{n} - \frac{|O(k^2)|}{n^2})^{k-1} = O(k^3/n^k).$$

Lastly, we must account for the necessary limited independence. For dependency trees, each vertex uses a probe count equal to its number of children plus one. Of course, this sum is bounded by $2k$ degrees of freedom for a k-vertex tree. The analysis for DAGs used additional independence. The worst-case instances in these accounting schemes use $3h_j + 3$ probes for vertex $j \in \mathit{Tr}$. Summing $3(|V|+|E_{\mathit{Tr}}|)$ for a spanning tree gives less than $6|V|$ degrees of freedom that must be reserved from ψ, in $DH_\psi^{(c)}$. The accounting for $DH_\psi^{(a)}$ is less severe, and $DH_\psi^{(b)}$ just requires that $\kappa \log n$ be at least as large as $2k$, for a k-element dependency set.

Altogether, the independence requirement is bounded by $6k$. ∎

Although true double hashing can process a k-key dependency set with $2k$ degrees of freedom, the difference in these counts is so small that we will drop all distinctions among the DH_ψ, and determine the independence requirements for the more restrictive model $DH_\psi^{(c)}$. Similarly, the probe costs will use very conservative overcounts that apply to any model of DH_ψ.

Lemma 2 suggests that the hashing statistics associated with trees should be analyzed in greater detail. The amortized analysis and uniformity of tree distributions allows this analysis to be simplified by pretending that each probe is to a random vertex embedded at an anonymous location.

Definition 11.

- Let $N(k)$ be the number of distinct ordered (dependency) trees that can occur with k vertices.
- Let $P_{tree}(k,j)$ be the probability that some partial dependency graph of x_j is a *tree* of k nodes.

Remark 2.

Lemma 2 shows that for any k element subset $\delta = (\delta_1, \ldots, \delta_k) \subset D$, where $k \leq \psi/6 = O(n^{1/3})$:

(a) $P(k,k) = \text{Prob}\{dep(\delta_k, \delta) = \delta\} = n^{-k+1}N(k)(1 + O(k^3/n))$

(b) $P_{tree}(k,k) = n^{-k+1}N(k)(1 + O(k^2/n))$.

The next steps formalize these remarks, introduce $Err(k,j)$, a bound that will replace $Err_0(j)$ in the formula of Corollary 1, and define a truncation error that permits the $P(\ell, j)$ to be summed through the first k terms only.

Definition 12.

- Redefine $P(k,k)$ to be $n^{-k+1}N(k)(1 + O(k^3/n))$.
- Let $P_{tree}(k,k) = n^{-k+1}N(k)(1 + O(k^2/n))$.
- Let $Err_1(k,j)$ be the probability that the vertex count $|dep(x_j, D)| > k$.
- Let $RR(k,j)$ be the Boolean indicator function for the event
  $\langle |dep(x_j, D)| \leq k$ *and some unsuccessful probe for* x_j *does not increase the size of its partial dependency set*$\rangle$:
  $RR(k,j) = (|dep(x_j, D)| \leq k) \bigwedge (dep_r(x_j, D) = dep_{r+1}(x_j, D)$ for some $r \leq probe_j - 2)$.
  Let $Err_2(k,j) = 2\text{E}[|dep(x_j, D)| \times RR(k,j)]$, so that we take a penalty of $2|dep(x_j, D)|$ probes when root x_j has any non-tree edges.
- Let $Err_3(k,j)$ be the probability that $|dep(x_j, D)| \leq k$ and x_j has at least $2|dep(x_j, D)|$ probes to $dep(x_j, D)$.
- Let $Err_4(j) = \sum_{t \geq c_0 n} \text{Prob}\{probe_j > t\}$, where c_0 is used in Definition 4 for DH and DH_ψ.
- Let X_b be the number of keys in D that, in $DH_\psi^{(b)}$, have their first $\kappa \log n$ probes hit just $\frac{\kappa \log n}{2}$ or fewer locations as formalized in Definition 4.6), and set $Err_b(j) = c_0 n \text{E}[X_b] + Err_4(j)$. Thus, the penalty for x_j's probe count is $c_0 n$ if any vertex has this excessive local probing, plus the expected number of probes for x_j that exceed the first $c_0 n$ regardless of the cause.
- Let $Err(k,j) = c_0 n Err_1(k,j) + Err_2(k,j) + c_0 n Err_3(k,j) + Err_4(j) + Err_b(j)$.

Corollary 2. For $\psi \geq 6k$,

$$\mathrm{E}[probe_j] \leq Err(k,j) + \sum_{1\leq i<k} P(i,j).$$

Proof: In view of Lemma 1, we need only show that $Err_0(j) + \sum_{i>k} \widetilde{P}(i,j) \leq Err(k,j)$. But this follows from the definition of $Err(k,j)$. ∎

The following definitions enable $P(k, \alpha n)$ to be formulated as a function of $P(k,k)$.

Definition 13.

Let $T = (T_1, T_2, \ldots, T_k)$, with $T_k < \alpha n$, be an increasing sequence of k indices (or time stamps) that correspond to the keys D_T.

- Define $loc_D(D_T)$ to be the sequence of table indices occupied by D_T when D is hashed into L so that $L[loc_D(D_{T_i})] = x_{T_i}$. Let, for a data sequence D', $loc(D') = loc_{D'}(D')$, so that *loc* without a subscript, takes its argument to be the complete sequence being hashed.

Let I be a sequence of k distinct index locations in the hash table.

- Let the Boolean predicate $M(T,I)$ correspond to the event: all of D can be successfully hashed into L according to the following modified hashing process: for $j \notin T$, x_j is hashed at time j according to its specified probe sequence; for $j = 1, 2, \ldots, |T|$, x_{T_j} can be placed, at time T_j, in the vacant location $L[I_j]$ without concern for the probe sequence. If some $L[I_j]$ turns out to be already occupied at time T_j, then $M(T,I)$ is false. For sequences of one element, we will occasionally abuse the M notation by writing, for example, $M(T_1, I_1)$ instead of the more precise $M((T_1),(I_1))$.

  In any case, $M(T,I)$ depends on I, T and $D - D_T - x_{\alpha n}$, but is independent of $x_{\alpha n}$ and the values comprising D_T. Simply stated, $((loc(D_T) = I) \bigwedge (loc_D(D_T) = I)) = ((loc(D_T) = I) \bigwedge M(T,I))$. In models UH and DH, the events $(loc(D_T) = I)$ and $M(T,I)$ are independent, although they depend on I in DH.

- Let $\hat{q}(T,I)$ denote the probability that $M(T,I)$ is true.

One of our main tasks is to show that $\hat{q}(T,I)$, for the UH, DH, and DH_ψ, are close.

- Let $\hat{q}_{\psi-6k}(T,I)$ be an overestimate for $\hat{q}(T,I)$ in $DH_{\psi-6k}$.

The subscript in the expression $\hat{q}_{\psi-6k}$ is intended to restrict numerical computation of $\hat{q}$ to inclusion-exclusion calculations that use no more than $(\psi-6k)$-wise independence (and include error terms to ensure that the formulation is an overestimate). The point of this definition is that $\hat{q}_{\psi-6k}$ will be used to estimate the probability that the locally determined embedding locations of a k-node dependency DAG are actually vacant at the real insertion times of the DAG elements. Thus, the DAG "uses up" $6k$ of the available degrees of freedom in DH_ψ. By definition, $DH_\psi \subset DH_{\psi-6k}$. Our overestimates will satisfy $\hat{q}(T,I) \le \hat{q}_\psi(T,I) \le \hat{q}_{\psi-6k}(T,I)$. Similarly, let $M_{\psi-6k}(T,I)$ denote the event $M(T,I)$ in $DH_{\psi-6k}$.

- Given any sequence δ, let $\delta||x$ denote the sequence δ with x appended at the end.

These definitions can now be put to use to find additional formulations for the expected number of probes.

Lemma 3.

In UH, for any fixed $I \Subset [0..n-1]$, with $|I| = k-1$:

$$P(k,\alpha n) = P(k,k) \sum_{\substack{T\subset[1,\alpha n-1]\\ |T|=k-1}} \hat{q}(T,I). \tag{3}$$

In DH:

$$P(k,\alpha n) \le P(k,k) \sum_{\substack{T\subset[1,\alpha n-1]\\ |T|=k-1}} \max_{\substack{I\Subset[0,n-1]\\ |I|=k-1}} \hat{q}(T,I). \tag{4}$$

In DH_ψ, for $6k \le \psi$:

$$P(k,\alpha n) \le P(k,k) \sum_{\substack{T\subset[1,\alpha n-1]\\ |T|=k-1}} \max_{\substack{I\Subset[0,n-1]\\ |I|=k-1}} \text{Prob}\{M(T,I)_{\psi-6k} \mid (dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}) \bigwedge (loc(D_T) = I)\}, \tag{5}$$

$$\le P(k,k) \sum_{\substack{T\subset[1,\alpha n-1]\\ |T|=k-1}} \max_{\substack{I\Subset[0,n-1]\\ |I|=k-1}} \hat{q}_{\psi-6k}(T,I). \tag{6}$$

Proof: In all three models:

$$P(k,\alpha n) = \sum_{\substack{|I|=k-1\\|T|=k-1}} \mathrm{Prob}\{(dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}) \bigwedge (loc(D_T) = I) \bigwedge M(T,I)\}$$

$$\leq \sum_{|T|=k-1} \Big(\Big(\sum_{|I|=k-1} \mathrm{Prob}\{(dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}) \bigwedge (loc(D_T) = I)\}\Big)$$

$$\times \max_{\substack{I \Subset [0,n-1]\\|I|=k-1}} \mathrm{Prob}\{M(T,I) \mid \ (dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}), (loc(D_T) = I)\}\Big).$$

Now,

$$\sum_{|I|=k-1} \mathrm{Prob}\{(dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}) \bigwedge (loc(D_T) = I)\} = \mathrm{Prob}\{(dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n})\}.$$

We know (Lemma 2 and Remark 2) that in all three models and for any suitable T,

$$\mathrm{Prob}\{dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}\} = P(k,k).$$

Inequality (5) now follows.

In UH, the event $M(T,I)$ is independent of $(dep(x_{\alpha n}, D_{T||\alpha n}) = D_{T||\alpha n}) \bigwedge (loc(D_T) = I)$, and is uniformly distributed over all sets I that comprise $k-1$ elements. Hence (3) follows with equality.

In DH, $M(T,I)$ depends on the specific locations in I, but for any fixed I, $M(T,I)$ is also independent of $\{(dep(x_{\alpha n}, D_T||\alpha n) = D_T||\alpha n) \bigwedge (loc(D_T) = I)\}$, since hash values in $D_T||x_{\alpha n}$ do not disclose any information about hash values on $D - D_T - x_{\alpha n}$. Thus (4) follows.

In DH_ψ, large events are not necessarily independent, but our estimates of the conditional $M_{\psi-6k}(T,I)$ will be based on windows of $\psi - 6(|D_T|+1)$ probe events for keys in $D - D_T - x_{\alpha n}$, conditioned on information about the hash function's behavior on $D_T||x_{\alpha n}$. Since these event samples (and error bounds) are, for any fixed I, statistically independent of the conditioning, the numerical estimate in (6) now follows. ∎

Further analysis of $P(k,k)$ will show that $P(k,\alpha n)$ is negligible for $k > C\log n$, for a suitable constant C. Similarly, $Err(k)$, will turn out to be negligible. As a consequence, the behavior of the hash function can be analyzed by determining $P(k,k)$ for the very light loads $k = O(\log n)$ and estimating $\hat{q}(T,I)$ based on many small samples of points in $D - D_T - x_{\alpha n}$, for $|T| = O(\log n)$.

3. Good vacancy estimators and their generic performance equation

Suppose that T is a sequence of insertion times for a collection of keys that locally hash into a rooted dependency DAG. This dependency DAG is important if its apparent hash locations are really empty at respective times T. We need a function $q(T)$ that overestimates the probability, in DH, that these $|T|$ locations are empty at their respective times in T.

Definition 14: Multiplicative vacancy overestimators**.**

- We call a Boolean function $M'(T,I)$ a vacancy overestimator if, for any sequence of table locations I and key indices T, the event $M(T,I)$, (that locations I are empty at respective times T), implies the event $M'(T,I)$.
- We call a function $q(t)$ a *multiplicative vacancy overestimator* for a vacancy overestimator $M'(T,I)$ if q is nonincreasing and for any fixed $\alpha < 1$, the following holds.

For all $D: \ |D| = \alpha n$, $T \subset [1, \alpha n - 1] : \ |T| = O(\log n)$:

$$\max_{\substack{I \Subset [1,n], \\ |I|=|T|}} \text{Prob}\{M'(T,I) \mid [dep(x_{\alpha n}) = D_T || x_{\alpha n}] \bigwedge [loc(D_T) = I]\} \le \left(1 + \frac{O(|T| \log^2 n)}{n}\right) \prod_{i=1}^{|I|} q(T_i).$$

We will use the subscript s, in $q_s(t)$, and $q_s(T,I)$ to denote (possibly) multiplicative vacancy overestimators that have vacancy estimates computed from samples of s or fewer keys, and which, therefore, are valid in DH_s.

By definition, $(1+O(\frac{\log^2 n}{n}))q(t)$ is an overestimate of the probability that a given table location is empty after $t-1$ keys have been hashed into L. Obviously, such q's do exist: $q(t) = 1$ is certainly a (very uninteresting) multiplicative overestimator. Given a multiplicative estimator q that is valid in all of our hashing models, Lemma 1, Lemma 2 and Corollary 2 show that the expected probe count for x_j can be overestimated by a computation that is virtually identical in UH, DH, and DH_ψ, provided that for a suitable $k = O(\log n)$, $Err(k,j)$ is sufficiently small. This last requirement allows the computation to be restricted to dependency trees of size k or less. For DH_ψ, the independence ψ will be required to exceed $6k + s$, where the s-wise (conditional) independence of the probe sequences must guarantee that the behavior of q is as stated. Until these values are quantified, we will expose these implicit dependencies by writing q_s when appropriate. The dependence of s on k, however, will be implicit.

The presentation of a suitably multiplicative overestimator q is technical, and will be deferred even further. Meanwhile, the reader might prefer to view the following development as if it were for UH, although the conclusions will apply to DH and DH_ψ as well.

Corollary 3. Let q_s be a multiplicative vacancy overestimator for UH, DH, or DH_ψ, where $s+6k \le \psi$, and let $Q_s(a) = \frac{1}{n}\sum_{0<b<a}(1+O(\frac{\log^2 n}{n}))q_s(b)$. Then for $k = O(\log n)$,

$$P_{tree}(k, \alpha n) \le \sum_{\substack{|T|=k-1 \\ T\subset[1,\alpha n-1]}} P_{tree}(k,k)\prod_{i=1}^{k-1}(1+\frac{O(\log^2 n)}{n})q_s(T_i) \tag{7}$$

$$\le P_{tree}(k,k)\frac{n^{k-1}Q_s(\alpha n)^{k-1}}{(k-1)!}. \tag{8}$$

Proof: Lemmas 2 and 3 show that

$$P_{tree}(k,\alpha n) \le (1+\frac{O(k\log^2 n)}{n})P_{tree}(k,k)\sum_{\substack{T\subset[1,\alpha n-1] \\ |T|=k-1}} \max_{|I|=k-1} q_s(T,I).$$

Inequality (7) follows from Definition 14. Similarly, (8) is an immediate consequence, since $\frac{1}{(k-1)!}\left(\sum_{t=1}^{\alpha n-1} q_s(t)\right)^{k-1}$ includes, for all subsequences $T \subset [1, \alpha n-1]$ with $(|T| = k-1)$, each of the products $\prod_{i=1}^{k-1} q_s(T_i)$, exactly once. ∎

Corollary 3 provides a way to estimate $P_{tree}(k,\alpha n)$ and hence $\mathrm{E}[probe_{\alpha n}]$ from a vacancy estimator $q_s(t)$.

Theorem 1. Let q_s be a multiplicative vacancy estimator, for UH, DH, or DH_ψ, where $s+6k \le \psi$ and $k = O(\log n)$. Let $\widetilde{Q}_s(a) = \frac{1}{n}\sum_{0<b<a} q_s(b)$, and $Q_s(a) = (1+|\frac{O(\log^2 n)}{n}|)\widetilde{Q}_s(a)$. Then

1) $$\mathrm{E}[probe_{\alpha n}] < \frac{1}{\sqrt{1-2Q_s(\alpha n)}} + O(\frac{\log^2 n}{n}) + Err(k,\alpha n).$$

Furthermore, suppose that for $|I| = O(\log n)$,

$$\mathrm{Prob}\{M(T,I)\} \le (1+O(\frac{|I|^3}{n}))\prod_{i\in I} q_s(i).$$

Then

$$\text{2)} \qquad \mathrm{E}[probe_{\alpha n}] < \frac{1}{\sqrt{1-2\widetilde{Q}_s(\alpha n)}} + O(\frac{1}{n}) + Err(k, \alpha n).$$

Proof: By Lemma 2,

$$P_{tree}(k,k) = N(k)(\frac{1}{n})^{k-1}(1+O(k^2/n)), \tag{9}$$

where $N(k)$ counts the number of different dependency trees with the k vertices $x_1, x_2, \ldots, x_k$. To determine $N(k)$, we observe that the root of the dependency trees is fixed at x_k, the vertex with highest index. Both the rightmost subtree of x_k, as well as the tree consisting of root x_k plus the remaining vertices comprising its other subtrees, if any, constitute partial dependency trees. If the rightmost subtree contains j vertices, its elements can be chosen in $\binom{k-1}{j}$ ways. This results in the following recurrence equation for N:

$$N(1) = 1$$
$$N(k) = \sum_{1\le j\le k-1} \binom{k-1}{j} N(j)N(k-j),$$

which upon setting $j' = k - j$ gives:

$$N(k) = \sum_{1\le j'\le k-1} \binom{k-1}{j'-1} N(j')N(k-j').$$

Averaging these two formulations, and applying the equality $\binom{k-1}{j-1} + \binom{k-1}{j} = \binom{k}{j}$ gives:

$$N(1) = 1,$$
$$N(k) = \sum_{1\le j\le k-1} \frac{1}{2}\binom{k}{j} N(j)N(k-j), \quad k > 1. \tag{10}$$

Let

$$g(x) = \sum_{0<k} \frac{N(k)}{k!}x^k. \tag{11}$$

Then multiplying (10) by $\frac{x^k}{k!}$, summing, and applying (11) gives:

$$g(x) = x + \sum_{k>1}\sum_{1\le j\le k-1} \frac{1}{2}\frac{N(j)}{j!}\frac{N(k-j)}{(k-j)!}x^k = g^2(x)/2 + x,$$

and hence

$$g(x) = 1 - \sqrt{1-2x} = 1 - \sum_{k\ge 0}\binom{1/2}{k}(-2x)^k. \tag{12}$$

Equating coefficients of x^k in (11) and (12) gives:

$$N(k) = k!\binom{1/2}{k}2^k(-1)^{k-1} = (k-1)!\binom{-1/2}{k-1}(-2)^{k-1}.$$

Substituting for $N(k)$ in (9), and using Corollary 3 to define $P(k, \alpha n)$ in terms of $P_{tree}(k,k)$ and $Q_s(\alpha n)$ gives:

$$P(k,\alpha n) \le \binom{-1/2}{k-1}\left(1+\frac{O(k^3)}{n}\right)\left(\frac{-2nQ_s(\alpha n)}{n-O(1)}\right)^{k-1} = \binom{-1/2}{k-1}\left(1+\frac{O(k^3)}{n}\right)(-2Q_s(\alpha n))^{k-1}. \quad (13)$$

Summing the $P(i,\alpha n)$, as prescribed by Corollary 2 gives:

$$\begin{aligned}\mathrm{E}[probe_{\alpha n}] &< \sum_{i=1}^{k}(1+\frac{O(i^3)}{n})\binom{-1/2}{i-1}(-2Q_s(\alpha n))^{i-1} + Err(k,\alpha n)\\ &< \frac{1}{\sqrt{1-2Q_s(\alpha n)}} + \sum_i(\frac{O(i^3)}{n})\binom{-1/2}{i-1}(-2Q_s(\alpha n))^{i-1} + Err(k,\alpha n).\end{aligned} \quad (14)$$

The error attributable to non-tree DAGs is seen to be bounded by:

$$\sum_{i>0}\frac{O(i^3)}{n}\binom{-1/2}{i-1}(-2Q_s(\alpha n))^{i-1} = \sum_{i>3}\frac{O(1)}{n}\binom{-7/2}{i-4}(-2Q_s(\alpha n))^{i-1} = O(\frac{(Q_s(\alpha n))^3}{n(1-2Q_s(\alpha n))^{7/2}}), \quad (15)$$

which equals $O(\frac{1}{n})$. We conclude that for fixed $\alpha < 1$, and $2Q_s(\alpha n)$ bounded by some fixed value less than 1,

$$\mathrm{E}[probe_{\alpha n}] < \frac{1}{\sqrt{1-2Q_s(\alpha n)}} + O(\frac{1}{n}) + Err(k,\alpha n).$$

Since $Q_s = \widetilde{Q}_s + O(\frac{\log^2 n}{n})$, claim 1) follows from the asymptotic expansion $\frac{1}{\sqrt{1-x-\epsilon}} \approx \frac{1}{\sqrt{1-x}} + \frac{\epsilon}{2(1-x)^{3/2}}$. Claim 2) follows from the computation in equation (15), which shows that the asymptotic error terms $\frac{O(i^3)}{n}\binom{-1/2}{i-1}(-2\widetilde{Q}_s(\alpha n))^{i-1}$ give a total contribution of $O(\frac{1}{n})$. ∎

It is worth remarking that the results and computations are monotone in q_s; any error in its estimation carries through in Q_s.

It is also reassuring to observe that Theorem 1 gives the correct performance bound for UH. In UH, the probability that a location is vacant at time αn is $1-\alpha+\frac{1}{n}$, and according to the definition of $M(T,I)$, $\mathrm{Prob}(M(T,I)) = \prod_{i=1}^{|T|}(1-\frac{T_i - i}{n-i+1})$, whence $q(t) = 1-\frac{t-1}{n}$ is a multiplicative vacancy overestimator. In this case, $Q(\alpha n+1) = \frac{1+O(\frac{|I|^2}{n})}{n}\sum_1^{\alpha n}(1-\frac{j}{n}) = (\alpha - \alpha^2/2)(1+\frac{O(|I|^2)}{n})$.

Theorem 1.2 says that for UH,

$$\mathrm{E}[probe_{\alpha n}] < \frac{1}{\sqrt{1-2(\alpha-\alpha^2/2)}} + O(\frac{1}{n}) + Err(|I|,n) = \frac{1}{1-\alpha} + O(\frac{1}{n}) + Err(|I|,n),$$

which is tight, apart from $Err(|I|,n)$.

The calculations of Theorem 1 are useful for bounding $Err(k,\alpha n)$.

Corollary 4. Let q (with the subscript s suppressed, in the case of DH_{ψ},) be a multiplicative vacancy estimator in DH and DH_{ψ}, for $s+6k \leq \psi$. Set $2\tilde{Q}(\alpha n) = \frac{2}{n}\sum_{0<j<\alpha n} q(j)$, $Q(\alpha) = (1+O(\frac{\log^2 n}{n}))\tilde{Q}(\alpha)$, and suppose that $2Q(\alpha n) < \beta < 1$. Then $Err(k,\alpha n) = O(\frac{n^2\beta^k}{\sqrt{k}(1-\beta)})$, and hence for $k \geq 3\frac{\log n}{\log(1/\beta)}$,

$$Err(k,\alpha n) = O(\frac{1}{n}).$$

Proof:

1) Recall that $Err_1(k,j)$ is the probability that the vertex count $|dep(x_j,D)| > k$. We use $c_0 n$ as an overestimate of the first $c_0 n$ or fewer probes that occur, in this case, and show that $c_0 n Err_1(k,\alpha n) = \frac{O(1)}{n}$ for suitable k. So suppose that $|dep(x_{\alpha n},D)| > k$. Then some $x \in D$ has a partial dependency set of size $\hat{k}$, where $k < \hat{k} \leq 2k$. Indeed, let x_t be the first key in D to have a dependency set of size $k+1$ or more. Then each child of x_t in $G(x_t,D)$ must have a dependency set of size k or less. Sequencing over $G_1(x_t,D), G_2(x_t,D), \ldots$ gives a family of dependency graphs with vertex counts growing by steps of k or less, and eventually exceeding k. Hence one of these counts must be within $[k+1,2k]$. It follows that

$$Err_1(k,\alpha n) < \sum_{i\leq \alpha n}\sum_{k<j\leq 2k} P(j,i) < \alpha n \sum_{k<j\leq 2k} P(j,m) \quad \text{for some specific } m \leq \alpha n.$$

The proof of Theorem 1 (inequality (13)) allows us to bound this sum as follows.

$$
\begin{aligned}
c_0 n Err_1(k, \alpha n) &< \alpha c_0 n^2 \sum_{j=k+1}^{2k} (1 + O(j^3/n)) \binom{-1/2}{j-1} (-2Q(\alpha n))^{j-1} \\
&= O(n^2) \sum_{j \geq k} \beta^j \prod_{i=1}^{j} (1 - \frac{1}{2i}) \\
&= O(n^2) \sum_{j \geq k} \beta^j \prod_{i=1}^{j} e^{-1/2i} \\
&= O(n^2) \sum_{j \geq k} \beta^j e^{-(\log j)/2} \\
&= O(n^2) \sum_{j \geq k} \frac{1}{\sqrt{j}} \beta^j \\
&= O(n^2 \frac{\beta^k}{\sqrt{k}(1-\beta)}).
\end{aligned} \tag{16}
$$

Taking $k = 3\frac{\log n}{\log(1/\beta)}$ gives an additive error of $\frac{O(1)}{n}$.

2) Recall that $Err_2(\alpha n, k)$ equals $2|dep(x_{\alpha n}, D)|$ in the case that $|dep(x_{\alpha n}, D)| \leq k$ and $x_{\alpha n}$ has some unsuccessful probe that does not increase its partial dependency set. Let $|dep(x_{\alpha n}, D)| = j$. There are at most $j-1$ probes of $x_{\alpha n}$ that could be the first to touch a node already belonging to the dependency set. There are at most $j-1$ vertices that could be the probe's destination. Let the destination node be z. Key z can be reached by a direct probe edge from $x_{\alpha n}$, and by some earlier path from $x_{\alpha n}$ to z that comprises one or more edges. Let the node probed by $x_{\alpha n}$ along this path be w (which could equal z). In any case, we compute the probability that the DAG structure is hashed as specified by a traversal that begins with w, reaches all of its descendents, and then continues from $x_{\alpha n}$. The embedding of w is unconstrained, but $x_{\alpha n}$ will be constrained at two probe locations.

The expected number of ways these events can occur, in this case, is bounded by $(j-1)^2 \frac{j-1}{n} n^{-j+1}(1 + O(\frac{j^3}{n}))$. Upon taking the vacancy overestimator into consideration, the contribution from $Err_2(\alpha n, k)$ turns out to have a formulation that is equivalent to the calculation in (15): $Err_2(\alpha n, k) \leq \sum_{j=1}^{k} O(\frac{j^3}{n}) \binom{-1/2}{j-1} (-2Q_s(\alpha n))^{j-1} = O(\frac{1}{n})$.

3) Recall that $Err_3(k, \alpha n)$ is the probability that $|dep(x_{\alpha n}, D)| = j \leq k$ and $x_{\alpha n}$ has at least $2j$

probes to $dep(x_{\alpha n}, D)$. Evidently, the root must have two consecutive probes among the first $2j$ that comprise non-spanning tree edges, and the number of ways this can occur is at most $(2j-1)(j-1)^2$. The probability that a candidate dependency set of j vertices has the $2j$ probes as stated is therefore bounded by $(1+O(j^3/n))P(j,j)2j^3/n^2$, We can use $c_0 n$ as an estimate for the number of probes in this case. The $O(\frac{1}{n})$ bound again follows from calculations comparable to those in (15).

4) Recall that $Err_4(\alpha n)$ is the expected number of probes of length $c_0 n$ or more needed to insert $x_{\alpha n}$. Then the expected number of probes contributed in this case is $c_0 n$Prob{at least $c_0 n$ probes occur} + $\sum_{t>c_0 n}$ Prob{at least t probes occur}. The first term is already counted by $c_0 n Err_3 + c_0 n Err_1$. The second term equals $\frac{O(1)}{n}$, according to our robustness requirement that long probe sequences be rare.

Finally, recall that $Err(k,j) = c_0 n Err_1(k,j) + Err_2(k,j) + c_0 n Err_3(k,j) + Err_4(j) + Err_b(j)$. The definition of $DH_{\psi}^{(b)}$ and the bound for $Err_4(j)$ ensure that $Err_b(j) = O(\frac{1}{n})$. As shown in 1)—4), each of the remaining terms is $O(\frac{1}{n})$ in size, and hence $Err(k,j) = O(\frac{1}{n})$, provided $k \geq 3\frac{\log n}{\log(1/\beta)}$.

∎

The next step is to define a suitable vacancy overestimator.

Definition 15: The vacancy criterion $M^{(h)}(T,I)$ and its probability $q^{(h)}(T,I)$.

- Given D, T and I, the subgraph defined by the local hashing of the key tuple S is called an *occupancy tree for the pair* (T_i, I_i) if $|S| \leq h$, $S \subset (D_1, D_2, \ldots, D_{T_i}) - D_T$ and S hashes locally into a dependency tree with its root embedded at I_i, and no other key embedded in I.

- Let $M^{(h)}(T,I)$, be the vacancy criterion: for $j = 1, 2, \ldots, |I|$, D does not contain an occupancy tree for the pair (T_j, I_j).

- Let $q^{(h)}(T,I)$ denote the probability that the vacancy criterion $M^{(h)}(T,I)$ holds.

Thus, $M^{(h)}(T,I)$ is a vacancy criterion with limited backtracking. Loosely put, a location $\ell \in I$ is deemed occupied at a specific time only if some key has already probed the location as a consequence of belonging to a local dependency tree that contains h or fewer keys. Otherwise ℓ is deemed vacant.

The performance analysis will identify a superset of keys that might belong to some suitably small occupying tree. These candidate keys and their collision structure will be called witness graphs even if there is no occupancy tree. In this latter case, the witnesses declare the locations I to be vacant at their respective times T. It should also be noted that an occupancy tree, for some location I_j, might not represent the dependency graph actually rooted at I_j. Moreover, we will not actually determine $q^{(h)}(T, I)$; instead, Lemma 4 will estimate the simpler $q_{UH}^{(h)}(t_j, i_j)$, and do so with rather poor accuracy. However, the vacancy estimator will be virtually unaffected by the assumptions of limited independence, as well as the specific hashing model, and it will be strong enough to give good hashing bounds (once it is proven to be multiplicative and then used) in Theorem 1. The multiplicativity needed to estimate $q_{UH}^{(h)}(T, I)$ is deferred to Theorem 3 and the slightly stronger Theorem 4 at the end of this paper.

Lemma 4. For any fixed $\alpha < 1$ and fixed h, in uniform hashing,

$$q_{UH}^{(h)}(\alpha n, i) \le 1 - \alpha + \frac{(2\alpha - \alpha^2)^{h+1}}{h^{3/2}(1-\alpha)} + \frac{O(1)}{n}.$$

Proof: If $M^{(h)}(T_j, I_j)$ is true, then one of the following must hold:

1) I_j is vacant at time T_j.

2) The true dependency graph rooted at I_j has h or fewer vertices and is not a tree.

3) The true dependency graph rooted at I_j has at least $h+1$ vertices.

The probability of the event $M^{(h)}(T_j, I_j)$ is therefore bounded by the sum of the probabilities associated with these three events. In UH, the true vacancy probabilities satisfy $q_{UH}(j, \ell) = 1 - \frac{j-1}{n}$. Moreover, it is an easy exercise to show that the true vacancy probability $q_{UH}(j, \ell) = 1 - \frac{j-1}{n}$ is multiplicative (with respect to $q_{UH}(T, I)$) as formulated in Definition 14. Hence the calculations of (15) are applicable. Evidently, the occupancy trees used to define $q_{UH}^{(h)}$ are rooted dependency DAGs that differ from the dependency trees of Lemma 2 in just two respects. First, the root is required to be embedded at some fixed location I_j. Second, the root x_r is not fixed at x_{T_j}, but instead has an index $r < T_j$. There are also some minor embedding constraints for occupancy trees.

These observations can be used to bound the probability for case 2). Furthermore, the analysis is very simple because in UH, $M(T,I)$ is known, and the occupancy structure is restricted to be the true occupancy DAG rooted at I_j. Consequently, we can adapt inequality (15). Crude adjustments multiply the bound for the expected number of non-tree dependency DAGs by αn to account for the fact the root is some node x_r, for $r < \alpha n$, and also multiply by $\frac{1+O(1/n)}{n}$ to account for the fact that the root must be hashed to the fixed index I_j. The resulting probability, for case 2), is $O(\frac{1}{n})$ as before.

As for case 3), let $P_{trueloc}(k,T_j)$ be the probability that the true dependency graph rooted at location I_j by time T_j is a DAG with k nodes. It is easy to see that the computation for $P_{trueloc}(k,T_j)$ is very similar to that for $P(k,T_j)$. The reasoning of Lemmas 2 and 3 gives the following.

$$\begin{aligned} P_{trueloc}(k,\alpha n) &\le (1+O(k^3)/n) \sum_{\substack{|T|=k \\ T\subset[1,\alpha n-1]}} N(k) n^{-k} \prod_{i=1}^{k} q(T_i) \\ &\le (1+O(k^3)/n) N(k) \frac{Q(\alpha n)^k}{k!} \\ &= -\binom{1/2}{k}\left(1+\frac{O(k^3)}{n}\right)(-2Q(\alpha n))^k . \end{aligned} \tag{17}$$

We have seen that in UH, $2Q(\alpha n) \le 2\alpha - \alpha^2$, since the sum for Q includes fewer than αn terms. Hence for suitable K,

$$\text{Prob}\{M_{UH}^{(h)}(\alpha n, I_{\alpha n})\} \le (1-\alpha) - \sum_{k=h+1}^{K} \binom{1/2}{k}\left(1+\frac{O(k^3)}{n}\right)(-2Q(\alpha n))^k + \frac{O(h^3)}{n} + Err_1(K,\alpha n),$$

where each term has a positive contribution to the sum. As shown in (16) of Corollary 4, K can be chosen so that it is $O(\log n)$ in size and $Err_1(K,\alpha n) = O(\frac{1}{n})$. Similarly, we can infer from (15) that the asymptotic error term in (17) satisfies: $-\sum_{k=h+1}^{K}\binom{1/2}{k}\frac{k^3}{n}(-2Q(\alpha n))^k = O(\frac{1}{(1-\alpha)^{5/2}n}) = O(\frac{1}{n})$. Lastly, (16) suffices to bound the principal summation by $-\sum_{k=h+1}^{K}\binom{1/2}{k}(-2Q(\alpha n))^k < \frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)}$. Consequently,

$$\text{Prob}\{M_{UH}^{(h)}(\alpha n, I_{\alpha n})\} \le (1-\alpha) + \frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)} + \frac{O(1)}{n}. \quad \blacksquare$$

It is worth remarking that $q_{UH}^{(h)}$ actually convergences at a much faster rate (as a function of h) than the estimate given by Lemma 4. When $h=1$, for example, it is easy to verify that the probability that a location will not have been hit by a first probe of αn keys is $1 - e^{-\alpha n} + \frac{O(1)}{n}$, which is already smaller than the corresponding bound established in Lemma 4.

We will soon prove, for all three hashing models, that $q^{(h)}(t)$ is a multiplicative vacancy overestimator. The consequences are the following.

Theorem 2. Suppose that $q_s^{(h)}(t) = (1 - \frac{t}{n}) + \frac{(2\frac{t}{n}-(\frac{t}{n})^2)^{h+1}}{h^{3/2}(1-\frac{t}{n})} + \frac{O(1)}{n}$ is a multiplicative vacancy overestimator in DH_ψ for $\psi = d(\log n)$, where d is a sufficiently large constant. Then

$$\mathrm{E}[probe_{\alpha n}] < \frac{1}{1-\alpha} + O(\frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)^{5/2}}) + O(\frac{\log^2 n}{n}).$$

Proof: Let $2Q_s(\alpha n) = (1+O(\frac{\log^2 n}{n}))\frac{1}{n}\sum_{t\le\alpha n} q_s^{(h)}(t) \le 2\alpha-\alpha^2+2\frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)}+O(\frac{\log^2 n}{n})$. Choose h so large that $2\alpha-\alpha^2+2\frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)}$ is closer to $2\alpha-\alpha^2$ than 1.

The calculations in Corollary 4 show that $Err(k,\alpha n) = O(\frac{1}{n})$, for large enough $\psi = O(\log n)$. The result now follows from the fact that $Q_s(\alpha n) - (\alpha - \alpha^2/2) < \frac{(2\frac{t}{n}-(\frac{t}{n})^2)^{h+1}}{h^{3/2}(1-\frac{t}{n})} + O(\frac{\log^2 n}{n})$, Lemma 4, and the asymptotic expansion $\frac{1}{\sqrt{a^2-\epsilon}} \approx \frac{1}{a} + \frac{\epsilon}{2a^3}$, for ϵ close to zero. ∎

This theorem is our main result, although it is strengthened, somewhat, in Theorem 4. The remainder of this paper is mostly aimed at showing that the probability of $M^{(h)}(T,I)$, for any constant h, can be adequately estimated in DH, and even DH_ψ. The proof is based on a statistical analysis of the witness graph $WG^{(h)}(T,I)$, which is formalized in Definition 16.

Intuitively, the witness graph will contain all vertices (hash keys from D) that could possibly belong to some local occupancy tree that comprises h or fewer vertices and is rooted in I. As in the case of dependency DAGs, the graph structure will also record the hashing collisions that occur among its vertex keys. However, the witness graph will be bipartite, and have two vertex types, which are hash keys and abstract probe locations. Informally, the top level of the graph will comprise the hash locations I. Directed edges will go from I to keys that might hash into I. These keys, in turn, will have directed edges to earlier probe locations that must be occupied if the keys are to land in I. The process will continue for these new locations. The parameter h limits the size of the witness graph, by ensuring, for example, that it cannot contain a path with more than h keys.

The witness graph is constructed in a greedy top-down manner, much as the construction of dependency trees. Let $D' = (x'_1, \ldots, x'_{|D'|})$ be the sequence $(D - D_T - x_{\alpha n})$ in reverse order. The witness set will comprise the hash keys belonging to the witness graph. They are found by scanning D' to see, essentially, which keys might wind up hitting relevant locations within their first h probes. We call a location relevant (or actually an *eligible collision location*) if it represents a probe location for a key that belongs to the witness set as a consequence of that location being already occupied.

Procedurally, the *eligible collision set* (i.e., set of eligible collision locations) is initialized to I. Then the keys in $D - D_T - x_{\alpha n}$ are scanned in reverse order. If a key hits an eligible collision location at probe $i \leq h$, the key is entered into the witness set, and the first $i - 1$ locations are included in the eligible collision set. Then the next key is processed. As this procedure suggests, the witness set is defined without reference to the time constraints T; this simplification can only increase the number of relevant keys and eligible collision locations that are found.

We achieve better bounds by including the sum of the number of probes consumed by the sequence of collisions that is responsible for the presence of each eligible collision location. If some key x takes k probes to reach an eligible collision location, then only $h - k + 1$ probes are available for a witness key that (at some earlier real insertion time) hashes into one of the first $k - 1$ probe locations in x's probe sequence. Definition 16 will provide a formal construction of the witness graph. The eligible collision set will be stored in a family of index sets $\mathcal{L}(i, \tau)$, for $i \in [0, h - 1]$. The indices i and $\tau \in [0, |D'|]$ indicate that the locations in $\mathcal{L}(i, \tau)$ are available for collisions with the first $h - i$ probes of the candidate witness key $x'(\tau + 1)$. Initially, $\mathcal{L}(0, 0) = I$ and $\mathcal{L}(i, 0) = \emptyset$ for $i > 0$. After x'_τ is processed and hence $\mathcal{L}(i, \tau)$ determined, $\mathcal{L}(i, \tau + 1)$ is initialized to $\mathcal{L}(i, \tau)$ and additional locations might then be inserted into $\mathcal{L}(\tau + 1)$, depending on the first h probes of $x'_{\tau+1}$. In particular, if $x'_{\tau+1}$ hits a location in $\mathcal{L}(j, \tau + 1)$ on probe i, with $j + i \leq h$, then $x'_{\tau+1}$ is inserted into the witness set W and its first $i - 1$ probes are inserted into $\mathcal{L}(j + i - 1, \tau + 1)$.

Because the keys are processed in reverse order, with a greedy interpretation as to their eventual hash location, the procedure will include many elements in the witness set that will turn out to be irrelevant. Some keys are assumed to reside in probe locations for which earlier probe locations will turn out to be empty; for others, the dependency graph will turn out to be to big. On the other hand, these circumstances will only be evident after the structure is completed and all of the keys have been inserted.

For completeness, we are obliged to state what happens in the unlikely event that more than one of a key's first h probes land in eligible collision locations. The reason that this issue must be addressed is that once two probes are given specific values, the remaining probes might be completely deterministic—as can be the case, for example, with real double hashing.

If a key in $D' - D_T - x_{\alpha n}$ has several probes to eligible collision locations, we simply ignore the key, and refrain from recording the collisions. Similarly, we will ignore a key that has two probes to the same location and then has a probe to the eligible collision set. These restrictions produce a simplified witness graph that overestimates the probability that all locations in I are vacant at the corresponding times in T, based on the hashing behavior of h-tuples in $D - D_T - x_{\alpha n}$. On the other hand, we can underestimate this joint vacancy property by declaring the vacancy criterion to be false if any key was omitted from the witness set due to some kind of double collision. It is reasonable to anticipate that these events should have an insignificant affect on the overall vacancy statistics. The procedure is fully specified below.

Definition 16: The witness graph $WG^{(h)}(T, I) = (\mathcal{L}, W, E)$**.**

Let witness graph $WG^{(h)}(T, I) = (\mathcal{L}, W, E)$ be procedurally specified by the following algorithm.

1. $D' \leftarrow Reverse(D - D_T - x_{\alpha n})$;
2. $\mathcal{L}(0, 0) \leftarrow I_T$;
3. $W(0) \leftarrow \emptyset$; $E(0) \leftarrow \emptyset$;
4. for $i \leftarrow 1$ to $h - 1$ do $\mathcal{L}(i, 0) \leftarrow \emptyset$ endfor;
5. for $\tau \leftarrow 1$ to $|D'|$ do
6. for $i \leftarrow 0$ to $h - 1$ do $\mathcal{L}(i, \tau) \leftarrow \mathcal{L}(i, \tau - 1)$ endfor;
7. $W(\tau) \leftarrow W(\tau - 1)$;
8. if for exactly one pair (i, j) $i \in [1, h]$, $j \leq h - i$: $p(x'_\tau, i) \in \mathcal{L}(j, \tau - 1)$

   AND for all $r < s < i$: $p(x'_\tau, r) \neq p(x'_\tau, s)$

```
        AND for all r < i, s < h: p(x'_τ, r) ∉ ℒ(s, τ) then
              { A single eligible collision occurs at probe location p(x'_τ, i). }
9.            ℓ ← p(x'_τ, i);
10.           ℒ(i + j − 1, τ) ← ℒ(i + j − 1, τ) ∪_{k<i} {p(x'_τ, k)};
11.           W(τ) ← W(τ) ∪ {x'_τ};
12.           E ← E ∪ (ℓ, x'_τ; label i);
13.           E ← E ∪_{k<i} {(x'_τ, p(x'_τ, k); label k)}
14.       endif
15.   endfor;
16.   W ← W(|D'|);
17.   ℒ ← ∑_j ℒ(j, |D'|);
18.   Replace all location labels referencing indices in L − I by pointers to abstract vertices.
```

The location labels mentioned in 18 refer to the ℓ in 9 and 12, and the $p(x'_\tau, k)$ in 10 and 13.

Lemma 5 will compare the probabilities that a given witness graph structure occurs in UH, DH, and DH_ψ. To establish an asymptotic equivalence among all three models, it is essential that the actual probe locations apart from those hitting I be absent from the structure (line 18). All that is recorded in the structure is which probes collided with which. To show that the vacancy estimates are almost the same in the three models, we need, in part, estimates to bound, with high probability, the size of a witness graph as a function of the dependency set size $|I|$. Lemma 6 bounds the expected size of the witness graph, and shows that the witness graph is of size $O(\log n)$ with sufficiently high probability.

Let $W \subset D'$ be a candidate witness set of k keys with $W = \{x_{i_1}, \ldots, x_{i_k}\}$. Let WG be a candidate labeled witness graph for the pair (T, I) with key vertex set W, and (abstract) eligible location set $\mathcal{L}$, with $\mathcal{L}(i, \tau)$ as specified in Definition 16. Recall that by construction, any such WG is a forest; there will be no cycles, and each vertex has an indegree of 1 or 0.

In the following presentation, we suppress, for notational simplicity, some of the implicit dependence of various quantities on the sets D, T, I, and the randomness in the hashing strategy.

Similarly, we will use the notation WG in two contexts. When WG is selected from a set of candidate witness graphs, WG will represent a sample set of keys and a hash structure; in this case, we will compute the probability that WG is the actual witness graph that occurs. When WG is not bound as a candidate, it will represent the actual witness graph; here the computational issue is the probability that its size $|WG|$ is too large.

Definition 17.

- Let $\mathcal{L}(\tau) = \sum_{i=0}^{h-1} |\mathcal{L}(i,\tau)|$, and $|\mathcal{L}| = \mathcal{L}(|D'|)$.
- Let $|WG| = \max(|\mathcal{L}|, |W|)$.
- Let $\mathrm{Prob}^{W}_{UH}\{WG\}$, $\mathrm{Prob}^{W}_{DH}\{WG\}$, $\mathrm{Prob}^{W}_{DH_\psi}\{WG\}$ be the probabilities that WG is the actual witness graph for the pair (T, I) in, respectively, UH, DH and DH_ψ.
- Let $\mathrm{Prob}^{pure}_{UH}\{WG\}$, $\mathrm{Prob}^{pure}_{DH}\{WG\}$, $\mathrm{Prob}^{pure}_{DH_\psi}\{WG\}$ be the analogous probabilities that WG is the witness graph for the pair (T, I) and that no vertices were eliminated in its construction due to double hits.
- Let $\mathrm{Prob}^{W}_{\bullet}\{WG\}$ and $\mathrm{Prob}^{pure}_{\bullet}\{WG\}$ be used in expressions where $\bullet$ is a free subscript that holds when all $\bullet$-s are simultaneously replaced by the same label UH, DH, or DH_ψ.

Lemma 5. Let D be a sequence of αn keys, with D_T and I specified, where $|I| = O(\log n)$. Let WG be a candidate witness graph with $|\mathcal{L}| = O(\log n)$. Suppose that $\psi \geq h|W| + 6|I| + 5h^2|\mathcal{L}| + \log n$. Then

1) $\mathrm{Prob}^{W}_{DH}\{WG\} = \mathrm{Prob}^{W}_{UH}\{WG\}(1 + O(|\mathcal{L}| \cdot |WG|)/n)$.

2) $\mathrm{Prob}^{W}_{DH_\psi}\{WG\} = \mathrm{Prob}^{W}_{DH^{\psi}_{\infty}}\{WG\}(1 + O(\frac{1}{n}))$.

3) $\mathrm{Prob}^{pure}_{\bullet}\{WG\} = \mathrm{Prob}^{W}_{\bullet}\{WG\}\left(1 + \frac{O(|\mathcal{L}|^2)}{n}\right)$, for UH, DH, and DH_ψ.

Proof: The counting statistics for witness graphs is similar to that for dependency graphs, but differs from the latter in two aspects. The simplest change is that witness graphs have embedded roots (in locations of I). The other difference is that, whereas the combinatorial structure of dependency graphs was primarily based on local events that are characterized by the event $dep(x, S) = S$, witness graphs (forests) must be analyzed in terms of both local and global events. The probability that a forest WG is the witness forest in question involves both local hashing properties and the event that the many keys not included in WG either do *not hit* any of the eligible hit locations belonging to WG or hit the location set more than once. However, as before, these probabilities will turn out to be nearly identical in our four models.

Definition 18.

- Let $\mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\}$ be the probability that the local set comprising the keys $W \subset D$ hit the prescribed virtual locations in the manner prescribed by WG, and with no additional collisions in $\mathcal{L}$ by the restricted probes of keys in W.
- Let $\mathrm{Prob}_{UH}^{not1\text{-}hit}\{WG\}$ be the probability that vertices in $D - D_T - W - x_{\alpha n}$ incur either no hits or multiple hits to the eligible location set within the requisite number of probes, as they are processed by the algorithm.
- Let $\mathrm{Prob}_{UH}^{no\text{-}hit}\{WG\}$ be the probability that $D - D_T - x_{\alpha n} - W$ do not hit the eligible location set at all, within the requisite number of probes.
- Let these definitions be extended to models DH and DH_ψ.

Evidently,

$$\mathrm{Prob}_{UH}^{W}\{WG\} = \mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\} \times \mathrm{Prob}_{UH}^{not1\text{-}hit}\{WG\}. \tag{18}$$

In DH, such a simple formulation is not quite true, since $\mathrm{Prob}_{DH}^{\ell\text{-}hit}\{WG\}$ and $\mathrm{Prob}_{DH}^{not1\text{-}hit}\{WG\}$ can depend, somewhat, on just which locations are actually used for each possible embedding of WG. Now, the reasoning of Lemma 2.1 shows that in UH, DH, and DH_ψ, for large enough ψ,

$$\mathrm{Prob}_{\bullet}^{\ell\text{-}hit}\{WG\} = (1/n + O(h|\mathcal{L}|/n^2))^{|W|}, \tag{19}$$

where W is the set of keys belonging to WG, and we use the conventional interpretation of $O(h|\mathcal{L}|/n^2)$, which is arbitrarily signed. The exponent in (19) is $|W|$ rather than $|W| - 1$ because unlike dependency sets, the roots in witness forests are explicitly embedded in prespecified locations.

Let WG be decomposed into the disjoint trees $WG_1, WG_2, \ldots, WG_I$, where tree WG_i is rooted in location I_i. The formula for $\mathrm{Prob}_{\bullet}^{\ell-hit}\{WG\}$ shows that

$$\mathrm{Prob}_{\bullet}^{\ell-hit}\{WG\} = (1 + O(\frac{|W| \cdot |\mathcal{L}|}{n})) \prod_{i=1}^{|I|} \mathrm{Prob}_{UH}^{\ell-hit}\{WG_i\},$$

and this asymptotic equivalence will be used in Theorem 3.

In order to get good hashing estimates in DH, a formulation comparable to (18) must be established for $\mathrm{Prob}_{DH}^{W}\{WG\}$. Let XG denote an *explicit* embedding of WG that includes its collision structure, and the explicit locations of $\mathcal{L}(j,t)$, for all $j < h$, $t \le \alpha n$. Given D, D_T, I, and a candidate $XG = (D_W; E, E_{embedding})$, the probability, in UH and DH, that the candidate XG structure defines the real witness graph, is a consequence of two events: 1) the local hashing of D_W is as prescribed by E, which must name the vertex endpoints (as indices in D) and their physical embeddings (as indices in $[0, n-1]$) and 2) no key $x \in D - D_T - x_{\alpha n} - D_W$ has a relevant probe that causes x to be in the witness set. Now, the probability that XG defines the witness graph can vary tremendously between the two models, but in both cases, the probability that W is hashed as specified by E (and its probe locations $E_{embedding}$), and the probability that $D - D_T - x_{\alpha n} - W$ is hashed as specified (to have no relevant collisions) are independent of each other. The reason for the independence is that the hashing of each key is independent of the hashing of any other. Thus, $\mathrm{Prob}_{\bullet}^{W}\{XG\} = \mathrm{Prob}_{\bullet}^{\ell-hit}\{XG\} \times \mathrm{Prob}_{\bullet}^{not1-hit}\{XG\}$ where the subscript $\bullet$-s are all either DH or UH. The second factor in this product formulation for $\mathrm{Prob}_{\bullet}^{W}\{XG\}$ in DH and UH is readily written as $\mathrm{Prob}_{\bullet}^{not1-hit}\{XG\} = \mathrm{Prob}\{\bigwedge_{\tau:\ x'_\tau \in D'-W} \neg s_{\bullet}(\tau)\}$, where $s_{\bullet}(\tau)$ is the event that $x'_\tau \in D'$ experiences a single eligible hit with respect to the embedded sets $\mathcal{L}(0, \tau-1), \ldots, \mathcal{L}(h-1, \tau-1)$. Let $\sigma_{\bullet}(\tau, XG)$, with appropriate subscript UH or DH, denote $\mathrm{Prob}\{s_{\bullet}(\tau)\}$. Since, for any specific embedding XG, the events $s_{\bullet}(\tau)$ are mutually independent in both DH and UH,

$$\mathrm{Prob}_{\bullet}^{not1-hit}\{XG\} = \prod_{\tau:\ x'_\tau \in D'-W} (1 - \sigma_{\bullet}(\tau, XG))$$

in these two models. It is easy to see that for all τ, $|\sigma_{UH}(\tau, XG) - \sigma_{DH}(\tau, XG)| = O(\binom{h}{2}) \frac{|\widetilde{\mathcal{L}}(\tau-1)|^2}{n^2}$. Hence

$$1 - \sigma_{DH}(\tau, XG) = (1 - \sigma_{UH}(\tau, XG))(1 + \frac{O(|\mathcal{L}|^2)}{n^2})$$

for any embedding of the structure and any constant h. $\text{Prob}_{\bullet}^{not1-hit}\{XG\}$ is therefore the same in UH and DH to within a factor of $(1+O(|\mathcal{L}|^2/n^2))^{\tau} = (1+O(|\mathcal{L}|^2/n))$.

To complete the argument for DH, we sum over all embeddings of WG, which we denote below with the expression $XG \cong WG$.

$$\begin{aligned}
\text{Prob}_{DH}^{W}\{WG\} &= \sum_{XG\cong WG} \text{Prob}_{DH}^{W}\{XG\} \\
&= \sum_{XG\cong WG} \text{Prob}_{DH}^{\ell-hit}\{XG\} \times \text{Prob}_{DH}^{not1-hit}\{XG\} \\
&= (1+O(|\mathcal{L}|\cdot|WG|/n))\text{Prob}_{DH}^{not1-hit}\{WG\} \sum_{XG\cong WG} \text{Prob}_{DH}^{\ell-hit}\{XG\} \\
&= (1+O(|\mathcal{L}|\cdot|WG|/n))\text{Prob}_{DH}^{not1-hit}\{WG\} \times \text{Prob}_{DH}^{\ell-hit}\{WG\} \\
&= (1+O(|\mathcal{L}|\cdot|WG|/n))\text{Prob}_{UH}^{W}\{WG\}.
\end{aligned}$$

To extend the results to DH_ψ, the reasoning is almost the same. In all of the DH_ψ models, a candidate witness set W has exactly the same probability of hashing into an explicit witness graph XG in DH_ψ as in DH_∞^ψ. Similarly, the conditional probability that, given the hashing of D_T and W, a specific key $x_t \notin W$ does not probe an eligible collision location as specified by XG is likewise the same in both DH_ψ and DH_∞^ψ. In view of the independence (multiplicativity rules) 2) for DH, these events can be treated as Bernoulli trials, which are independent in DH_∞^ψ, and $(\frac{\psi-6|I|-h|W|}{h})$-wise independent in DH_ψ. Lemma A2 in the Appendix says that $\text{Prob}_{\bullet}^{not1-hit}\{XG\}$, in DH_ψ and DH_∞^ψ, will differ by a factor of just $(1+O(\frac{1}{n}))$ if the additional independence is adequate. Since the expected number of collisions with the witness set is less than $h|\mathcal{L}|$, Lemma A2 ensures that the probabilities will satisfy the asymptotic equivalences provided $\psi \geq 6|I| + h|W| + 5h^2|\mathcal{L}| + \log n$. (That is, the expected outcome of the fewer than n Bernoulli trials is less than $h|\mathcal{L}|$, and each trial is a statement about h probes of a key, which requires up to h degrees of independence.) Upon summing over all possible explicit embeddings XG, part 2) now follows.

The proof for part 3) is comparable to that for part 2). The main observation is that the probability that a key has two collisions (among its first h probes) with WG is $O(h^2\frac{|\mathcal{L}|^2}{n^2})$, and this bound also holds for the conditional probability of the two-collision event, given that the key does not have exactly one collision. ∎

We will now show that $q^{(h)}(T,I)$, the probability that the vacancy criterion holds for locations I, can be successfully approximated in all three models by the probability that these locations are declared empty by the witness graphs. It will follow that the probabilities $q^{(h)}_{UH}(T,I)$, $q^{(h)}_{DH}(T,I)$, and $q^{(h)}_{DH_\psi}(T,I)$ differ by a factor of at most $\left(1+\frac{O(\log^2 n)}{n}\right)$ in the three models, for an appropriate choice of ψ.

Definition 19.

- Let $w^{(h)}_{\bullet}(T,I)$, in UH, DH or DH_ψ, be the probability that $WG^{(h)}(T,I)$, by the completion of time step T_j-1, does not contain a local dependency tree of size h or less rooted at I_j, for any $j \le |I|$.

Thus, $w^{(h)}(T,I)$ is the probability that $WG^{(h)}(T,I)$ "declares" each location I_j to be empty at insertion time T_j.

Lemma 6. Let $\alpha<1$ be fixed, the integer h be a constant, and let $T=(T_1,\ldots,T_{|T|})$ be a sequence of increasing time stamps with $T_{|T|}=\alpha n$, and let I be an arbitrary sequence of distinct table locations with $|I|=|T|=O(\log n)$. Let $I_0=\frac{h^3}{2}[\ln(\frac{1}{1-\alpha})|I|+3\ln n]$, and $f(h)=e^{2(h+1/2)^{2/3}}$. Let the random structure $WG^{(h)}(T,I)=(\mathcal{L}(0,\alpha n),\mathcal{L}(1,\alpha n),\ldots,\mathcal{L}(h-1,\alpha n);W;E)$ be the witness graph for the pair (T,I), with $T_{|T|}=\alpha n$ and let $\widehat{\mathcal{N}}(j,\alpha n)=\mathrm{E}[\mathcal{L}(j,\alpha n)]$, for $j<h$.

Suppose that $\psi \ge \log n + 5h^2\lambda + 6|I| + h\omega$, where $\omega=\lambda=3hf(h)I_0+h$.

Then

1) $\widehat{\mathcal{N}}(j,\alpha n) \le |I|f(j)$ in UH, DH, and DH_ψ, provided n is sufficiently large.

2) $\mathrm{Prob}_{\bullet}\{|\mathcal{L}| \ge 3hf(h)I_0\} < \frac{(1-\alpha)^{|I|}}{2n}$, provided n is sufficiently large.
$\mathrm{Prob}_{\bullet}\{|W| \ge 3hf(h)I_0\} < \frac{(1-\alpha)^{|I|}}{n}$, provided n is sufficiently large.

3) $w^{(h)}_{DH}(T,I)=w^{(h)}_{UH}(T,I)\left(1+\frac{O(\log^2 n)}{n}\right)$,
$w^{(h)}_{DH_\psi}(T,I)=w^{(h)}_{DH^\psi_\infty}(T,I)(1+O(\frac{1}{n}))$.

4) The vacancy overestimator $q^{(h)}(T)$ (as defined in Definition 14) with respective subscripts UH, DH and DH_ψ are equal, up to a factor of $(1+O(\log^2 n/n))$:

$$q^{(h)}_{DH_\psi}(T,I) = q^{(h)}_{DH}(T,I)\left(1+\frac{O(1)}{n}\right) = q^{(h)}_{UH}(T,I)\left(1+\frac{O(\log^2 n)}{n}\right) = w^{(h)}_{\bullet}(T,I)\left(1+\frac{O(\log^2 n)}{n}\right).$$

Proof: We first establish 1) and 2) for UH and DH. Then the results are extended to DH_ψ. Parts 3) and 4) will have analogous extensions to DH_ψ.

We again analyze the construction of the witness forest as if each collision event $\langle p(x'_t, i) \in \mathcal{L}(j, t-1)\rangle$ were the outcome of a Bernoulli trial with probability of success $|\mathcal{L}(j,t-1)|/n$. Thus, each time step is viewed as contributing $\binom{h}{2}$ (somewhat dependent) Bernoulli trials, of which at most one can result in success. This simplification will have three insignificant consequences that we are obliged to mention. First, the requirement that at most one of the trials be successful can be ignored, since we are interested in establishing upper bounds on the number of successes. Second, a single real probe can hit at most one of the disjoint sets $\mathcal{L}(j,t-1)$, and modeling the outcome with h different Boolean trials undercounts the probability that at least one success occurs, since the conditional probability that $\mathcal{L}(j,t-1)$ is probed, given that $\mathcal{L}(0,t-1),\ldots,\mathcal{L}(j-1,t-1)$ are not probed is effectively increased by a factor of $\frac{n}{n-O(\sum_{0\le i<j}|\mathcal{L}(j,\alpha n)|)}$. The simplest fix is to include this factor in the model implicitly, by replacing the table size n by the parameter n_1, and recasting the probabilities in terms of n_1, which will be adjusted a posteriori. We follow this prescription, although technical arguments can establish that no such a rescaling is actually necessary. Lastly, these changes allow us to drop factors of $(1+O(|I|/n))$ from many of the probability estimates, and we do so for convenience of reader and writer alike.

Accordingly, we overestimate the probability that $p(x'_t, j-i+1)$ constitutes an eligible hit in $\mathcal{L}(i,t-1)$, and x'_t's first h probes have no other hits to $\mathcal{L}$. This event is represented as the outcome of a Bernoulli trial with (an overestimated) probability of success equal to $\frac{|\mathcal{L}(i,t-1)|}{n_1}$. The time restriction for locations in I as prescribed by the definition of $q(T,I)$ will now be dropped, which renders the estimate $\mathcal{N}$ for $\widehat{\mathcal{N}}$ larger still. Let $I_0 = |I|$. (We will soon increase I_0 to yield a final increase in $\mathcal{N}$.) The simplified system reads:

$$\mathcal{N}(0,t) = I_0, \text{ and } \mathcal{N}(j,0) = 0,\ j = 1,2,\ldots,h-1;$$
$$\mathcal{N}(j,t) = \mathcal{N}(j,t-1) + \sum_{i<j}(j-i)\mathcal{N}(i,t-1)/n_1. \tag{20}$$

The system will be analyzed with generating functions. Define $f(x) \preceq g(x)$ to mean that each coefficient in the Taylor expansion of $f(x)$ (about $x = 0$) is positive and at most the value of the corresponding coefficient for g. Let the final estimate $\mathcal{N}$ be increased as specified by the following system:

$$\begin{aligned}&\mathcal{N}(j,0) = I_0,\ j = 0,1,2,\ldots.\\&\mathcal{N}(j,t) = \mathcal{N}(j,t-1) + \sum_{i<j}(j-i)\mathcal{N}(i,t-1)/n.\end{aligned} \tag{21}$$

Let $\nu(t,x)$ be the generating function $\nu(t,x) = \sum_j \mathcal{N}(j,t)x^j$. The solution for ν is immediate. The initial condition is $\nu(0,x) = \sum_j I_0 x^j = \frac{I_0}{1-x}$, and the recurrence equation becomes

$$\begin{aligned}\nu(t,x) &= \nu(t-1,x) + \sum_j\sum_{i<j}(j-i)\mathcal{N}(i,t-1)x^j/n_1\\&= \nu(t-1,x) + \frac{1}{n_1}\sum_j \mathcal{N}(j,t-1)x^j\sum_k kx^k\\&= \nu(t-1,x) + \frac{\nu(t-1,x)}{n_1}(\sum_k kx^k) = \nu(t-1,x)\left(1+\frac{1}{n_1}\frac{x}{(1-x)^2}\right).\end{aligned}$$

Hence

$$\nu(t,x) = (1+\frac{x}{n_1(1-x)^2})^t\frac{I_0}{1-x}.$$

Now, $1+\frac{x}{n_1(1-x)^2} \preceq e^{\frac{x}{n_1(1-x)^2}}$, (since $e^{\frac{x}{(1-x)^2}} = 1+\frac{x}{(1-x)^2}+\sum_{j>1}x^j\frac{1}{j!(1-x)^{2j}}$), whence for all $t \geq 0$, $(1+\frac{x}{n_1(1-x)^2})^t \preceq e^{\frac{tx}{n_1(1-x)^2}}$. Thus $\nu(t,x) \preceq e^{\frac{tx}{n_1(1-x)^2}}\frac{I_0}{1-x}$, and $\mathcal{N}(j,t)$ is bounded by the jth term in the Taylor expansion of $e^{\frac{tx}{n_1(1-x)^2}}\frac{I_0}{1-x}$. Set $t = \alpha n = \alpha_1 n_1$. Then

$$\begin{aligned}\frac{\nu(\alpha_1 n_1,x)}{I_0} &\leq \sum_j\frac{\alpha_1^j x^j}{j!(1-x)^{2j+1}} = \sum_j\frac{x^j\alpha_1^j}{j!}\sum_{k\geq 2j}\binom{k}{2j}x^{k-2j}\\&= \sum_j\sum_{k\geq 2j}\frac{x^{k-j}\alpha_1^j}{j!}\binom{k}{2j} = \sum_j x^j\left(\sum_{\ell=0}^{j}\binom{\ell+j}{2\ell}\frac{\alpha_1^\ell}{\ell!}\right).\end{aligned}$$

Consequently, $\frac{\mathcal{N}(j,\alpha_1 n)}{I_0} \leq \sum_{\ell=0}^{j}\binom{\ell+j}{2\ell}\frac{\alpha_1^\ell}{\ell!}$. Now, $\binom{\ell+j}{2\ell}\frac{1}{\ell!} \leq (j+1/2)^{2\ell}\binom{3\ell}{\ell}\frac{1}{3\ell!}$, and $\binom{3\ell}{\ell} \leq \frac{3^{3\ell}}{2^{2\ell}}$, so the summation is bounded by $\sum_{\ell=0}^{j}\frac{\left(\frac{3}{2^{2/3}}(j+1/2)^{2/3}\alpha_1^{1/3}\right)^{3\ell}}{(3\ell)!} < e^{2\alpha_1^{\frac{1}{3}}(j+1/2)^{2/3}}$. Setting $\alpha_1 = 1$ establishes 1) and validates[†] our lack of concern over replacing n by n_1.

[†] Technically, setting $\alpha = 1$ improves the bound by a significant factor for sufficiently large n. Similarly, this setting validates the formulations of parts 1) and 2): since probabilities are only prescribed asymptotically, the statements would need asymptotic factors of $(1+\frac{O(\log^2 n)}{n})$ if they were to hold for all n.

Given a dependency set size $|I|$, let $\mathcal{N}(j,t)$ be defined by (21) with $I_0 = a|I| + b$, where a and b will be specified later. We now show that for suitable a and b, $|\mathcal{L}(j,t)|$ is, with very high probability, no larger than $3\mathcal{N}(j,t)$.

Formally, we analyze the following modified process: if at any time τ, $|\mathcal{L}(j,\tau)| > (1 + 1/h)^j\mathcal{N}(j,\tau)$, then the process is aborted, and failure declared. The probabilistic recurrence analogous to (20) is given below, where $\mathcal{X}\langle event\rangle$ denotes the indicator function for the *event*.

$$|\mathcal{L}(0,\tau)| = |I|, \text{ and } |\mathcal{L}(j,\tau)| = 0, \ j = 1, 2, \ldots, h-1.$$

$$|\mathcal{L}(j,\tau)| = |\mathcal{L}(j,\tau-1)| + \sum_{i<j}(j-i)\mathcal{X}\langle p(x'_\tau, j-i+1) \in \mathcal{L}(i,\tau-1)\rangle, \ 1 \le j < h.$$

Expanding the recursion gives:

$$|\mathcal{L}(j,\tau)| = \sum_{i=0}^{j-1}(j-i)[\sum_{\tau'\le\tau}\mathcal{X}\langle p(x'_{\tau'}, j-i+1) \in \mathcal{L}(i,\tau'-1)\rangle], \text{ for } 1 \le j < h, \tag{22}$$

where the event $\langle p(x'_{\tau'}, j-i+1) \in \mathcal{L}(i,\tau'-1)\rangle$ requires that no other probe of $x_{\tau'}$ be an eligible hit in some $\mathcal{L}(k,\tau'-1)$, and no probe $p(x'_{\tau'},\ell)$, for $\ell \le j-i$, be in any $\mathcal{L}(*,\tau'-1)$.

A similar expansion for $\mathcal{N}$ gives:

$$\mathcal{N}(j,\tau) = \sum_{i=0}^{j-1}(j-i)\sum_{\tau'<\tau}\mathcal{N}(i,\tau')/n_1 + I_0, \ 1 \le j < h. \tag{23}$$

We establish 2) for DH as follows. Let $\mathcal{H}(j,t)$ be the Boolean indicator function for the event: $\langle|\mathcal{L}(i,\tau)| \le (1+1/h)^i\mathcal{N}(i,\tau) \textit{ for } i<j,\ \tau<t\rangle$. We first use crude estimates to show that

$$\text{Prob}\{|\mathcal{L}(i,t)| > (1+1/h)^i\mathcal{N}(i,t) \mid \mathcal{H}(i,t)\} < ie^{-\frac{I_0}{h^3}}. \tag{24}$$

It is evident that (24) holds for $i = 0$ and for $t = 0$.

Let $X(\tau,i)$, for $i = 0, 1, 2, \ldots, h-1$, $\tau = 1, 2, \ldots, \alpha n$, be independent Bernoulli trials with

$$\text{Prob}\{X(\tau,i) = 1\} = (1+1/h)^i\frac{\mathcal{N}_i(\tau-1)}{n_1},$$

and let $X_i(\tau) = \sum_{\tau'=1}^{\tau}X(\tau',i)$. By construction, $X_i(\tau)$, for $i \le \ell$, $\tau \le s$, statistically dominates $\sum_{\tau'\le\tau}\mathcal{X}\langle p(x, j-i+1) \in \mathcal{L}(i,\tau'-1)\rangle$, if $\mathcal{H}(i+1,\tau)$ is true. That is, for all r:

$$\text{Prob}\Big\{(\sum_{\tau'\le\tau}\mathcal{X}\langle p(x,j-i+1) \in \mathcal{L}(i,\tau'-1)\rangle) > r \mid \mathcal{H}(i+1,\tau)\Big\} \le \text{Prob}\Big\{X_i(\tau) > r\Big\}.$$

By construction,

$$E[X_i(t)] = \sum_{\tau<t} (1+1/h)^i \frac{\mathcal{N}(i,\tau)}{n_1}.$$

We now use the following type of Chernoff-Hoeffding bound, (proven in Lemma A1 in the Appendix), to bound our Bernoulli process:

$$Prob\{X_i(t) \geq (1+1/h)E[X_i] + C\} \leq e^{-\frac{5C}{4h}}.$$

Let $C = 2I_0/h^2$, so that with probability exceeding $1 - e^{-\frac{2I_0}{h^3}}$, an individual $X_i(t)$, $(0 \leq i \leq j-1)$, is bounded by $\left[\sum_{\tau \leq t}(1+1/h)^{i+1}\frac{\mathcal{N}(i,\tau-1)}{n_1} + \frac{2I_0}{h^2}\right]$. According to the definition of $\mathcal{L}(j,t)$ (in Equation (22)) and the definition of X_j above, for $j > 0$, $|\mathcal{L}(j,t)|$ is statistically dominated by $\sum_{i=0}^{j-1}(j-i)X_i(t)$ if $\mathcal{H}(j,t)$ is true. So given that $\mathcal{H}(j,t)$ is true, the conditional probability is at least $1 - je^{-\frac{2I_0}{h^3}}$ that:

$$\begin{aligned}|\mathcal{L}(j,t)| &\leq \sum_{i=0}^{j-1}(j-i)\left[\sum_{\tau\leq t}(1+1/h)^{i+1}\frac{\mathcal{N}(i,\tau-1)}{n_1} + \frac{2I_0}{h^2}\right]\\ &\leq I_0 + (1+1/h)^j \sum_{i=0}^{j-1}(j-i)\left[\sum_{\tau \leq t}\frac{\mathcal{N}(i,\tau-1)}{n_1}\right],\end{aligned}$$

which by (23) is

$$\leq (1+1/h)^j \mathcal{N}(j,t).$$

Hence

$$\text{Prob}\{\forall \tau \leq \alpha n,\ j<h : |\mathcal{L}(j,\tau)| \leq (1+\frac{1}{h})^j \mathcal{N}(j,\tau)\} \geq 1 - \frac{h^2}{2}ne^{-\frac{2I_0}{h^3}}.$$

Now let $I_0 = \frac{h^3}{2}[\ln(1/(1-\alpha_1))|I| + 3\ln n]$, so that $\frac{h^2}{2}ne^{-\frac{2I_0}{h^3}} \leq \frac{(1-\alpha_1)^{|I|}}{2n}$. Upon substituting, we see that in DH (and UH), the size of the eligible collision set $|\mathcal{L}(\alpha_1 n)|$ is, with probability $1 - \frac{(1-\alpha)^{|I|}}{2n}$, bounded by $\sum_{j=0}^{h-1}(1+1/h)^h \mathcal{N}(j,\alpha n) < ehf(h)I_0 < 3hf(h)I_0$.

The size of the witness set W can be bounded similarly. Let $W_j(\tau)$ be the portion of the witness set that, by time τ, has been discovered, in Definition 16, to have eligible probes in some $\mathcal{L}(j,s)$, for $s < \tau$. Its size satisfies:

$$|W_j(\tau)| \leq \sum_{\tau' \leq \tau} \mathcal{X}\langle p(y_{\tau'}, i) \in \mathcal{L}(j,\tau'-1) \mathit{for\ some\ probe\ number}\ i \in [1, h-j]\rangle.$$

If we are given upper bounds $U_j(\tau'-1)$ for the size of the $\mathcal{L}(j,\tau'-1)$, then $|W_j(\tau)|$ is statistically dominated by the sum of τ independent Bernoulli trials $X_j(1), X_j(2), \ldots, X_j(\tau)$ with probabilities of success $\text{Prob}\{X_j(\tau')\} = (h-j)U_j(\tau'-1)/n_1$. We have just shown that with a probability exceeding $1 - \frac{(1-\alpha)^{|I|}}{2n}$: $|\mathcal{L}(i,\tau)|$ is bounded by $(1+1/h)^i|\mathcal{N}_i(\tau)|$ for all $i < h$, $\tau \le \alpha n$. The Chernoff-Hoeffding bound from Lemma A1 ensures that conditioned on the bounds for $\mathcal{L}(j,\tau)$ being as stated,

$$Prob\{|W_j(\tau)| > (1+1/h)\sum_{\tau' \le \tau}(1+1/h)^j(h-j)|\mathcal{N}_j(\tau'-1)|/n_1 + \frac{2I_0}{h^2}\} \le e^{-\frac{2I_0}{h^3}}.$$

Hence, given the bounds on $\mathcal{L}(j,t)$, the conditional is at least $1 - je^{-\frac{2I_0}{h^3}}$ that

$$|W_j(\alpha n)| \le \sum_{i=0}^{j-1}\left(\sum_{\tau \le \alpha n}(1+1/h)^{i+1}(j-i)|\mathcal{N}_i(\tau-1)|/n_1 + 2\frac{I_0}{h^2}\right) \le 3\mathcal{N}(j,\alpha n).$$

It follows that in DH, $\text{Prob}\{|W| \le 3hf(h)I_0\} > 1 - \frac{(1-\alpha)^{|I|}}{n}$, which establishes the bound as stated in 2).

The extension of parts 1) and 2) from DH to DH_ψ use Lemma A2. An easy way to establish the bound is as follows. Given D, I and T, let B_s, for $s \le t$, comprise the event that, in DH_∞^ψ:

$$\langle(|W(s)| \ge 3hf(h)I_0)\bigwedge(W(s-1) < 3hf(h)I_0)\rangle.$$

Thus, B_s, for $s \le t$, comprise a disjoint decomposition of the events (or hash functions) where the bound for $|W|$ as stated in 2) fails to hold. So $\sum_{s \le t}\text{Prob}\{B_s\} \le \frac{(1-\alpha)^{|I|}}{n}$.

The argument is now the same as that used to establish Lemma 5.2. A candidate (partial) witness set $W(s)$ that corresponds to a possible event in B_s is analyzed in terms of all explicit embeddings that have the same abstract partial witness graph. Each such embedded graph has the same probability of occuring in DH_∞^ψ and DH_ψ, provided ψ is large enough. Similarly, the probability that the graph does not have additional members in its partial witness set will be, up to a factor of $(1+\frac{1}{n})$, the same in the two models. Since the expected number of collisions from $D - D_T - x_{\alpha n} - X$ is less than $n \times \frac{h}{n} \times |\mathcal{L}|$, a sufficient count for ψ is $\log n + 5h^2|\mathcal{L}| + 6|I| + h\omega$, where ω is taken to be $\lfloor 3hf(h)I_0\rfloor + 1$.

The same reasoning is readily applied to bound $|\mathcal{L}|$ in DH_ψ, and bound for $|\mathcal{L}|$ is $\lfloor 3hf(h)I_0\rfloor + h - 1$, and a sufficient bound for ψ is $\log n + 5h^2|\lambda| + 6|I| + h\omega$, where $\lambda = \omega = \lfloor 3hf(h)I_0\rfloor + h$. Technically, the decompositions are over insertion times s, all witness graphs that are just one witness too big (as measured by $|W|$ or $|\mathcal{L}|$), and all embeddings of these graphs. Of course, the probability of various embeddings differ tremendously across the models in DH, but the extension to DH_ψ only has to be from its analog in DH_∞^ψ. The closeness among models in DH, of course, is at a higher level where the graph structure is recorded but not the embeddings.

The factors $(1+O(\frac{1}{n}))$, which arise from Lemma A2 are dropped from the statement of part 2) because they are subsumed by the overestimation of the probability calculations and the requirement that n be sufficiently large.

Part 3) will now be established by showing that the probability that the witness graph $WG^{(h)}(T,I)$ contains no dependency tree with h or fewer keys and with its root embedded at I_j by time T_j, for any $j \in [1,|I|]$, is identical in UH and DH, up to a factor of $(1+O(\frac{\log^2 n}{n}))$. Evidently, $w_\bullet^{(h)}(T,I)$ can be expressed as the sum, over all legal candidate witness graphs that declare locations I empty at the corresponding times in T, of the probability that each such graph occurs. Denote by $W^h_{empty}(T,I)$ the subset of all witness graphs that declare, with respect to trees built from h nodes or less, all I locations empty at the prescribed times. Then

$$w_{UH}^{(h)}(T,I) = \sum_{WG \in W^h_{empty}(T,I)} \mathrm{Prob}^W_{UH}\{WG\}$$

and (25)

$$w_{DH}^{(h)}(T,I) = \sum_{WG \in W^h_{empty}(T,I)} \mathrm{Prob}^W_{DH}\{WG\}.$$

Parts 1) and 2) of Lemma 5 say that for any witness graph WG, where $|WG| = O(\log n)$,

$$\mathrm{Prob}^W_{DH}\{WG\} = \mathrm{Prob}^W_{UH}\{WG\}(1+\frac{O(|WG|^2)}{n})$$

and

$$\mathrm{Prob}^W_{DH_\psi}\{WG\} = \mathrm{Prob}^W_{DH_\infty^\psi}\{WG\}(1+O(1/n)), \text{ provided } \psi \geq h|W| + 6|I| + 5h^2|\mathcal{L}| + \log n.$$

It follows that

$$w_{DH}^{(h)}(T,I) = \sum_{\substack{WG \in W^h_{empty}(T,I) \\ |\mathcal{L}|,\ |W| < 3hf(h)I_0}} \mathrm{Prob}^W_{UH}\{WG\}(1+\frac{O(|WG|^2)}{n}) + O(\mathrm{Prob}\{\max(|W|,\,|\mathcal{L}|) \geq 3hf(h)I_0\}),$$

$$= \sum_{\substack{WG \in W^h_{empty}(T,I)\\ |\mathcal{L}|,\ |W| < 3hf(h)I_0}} \mathrm{Prob}^W_{UH}\{WG\}(1 + \frac{O(|WG|^2)}{n}) + O(\frac{(1-\alpha)^{|I|}}{n}),$$

$$= w^{(h)}_{UH}(T,I)(1 + O(\frac{\log^2 n}{n})),$$

which establishes the first part of 3). The second part: $w^{(h)}_{DH_\psi}(T,I) = w^{(h)}_{DH^\psi_\infty}(T,I)(1+O(\frac{1}{n}))$, follows from Lemma 5.2, the decomposition argument used in the extension of 2) from DH to DH_ψ, the bounds on $|W|$ and $|\mathcal{L}|$ as stated in 2), and the fact that $(1-\alpha)^I < q(T,I)$.

The argument for $q^{(h)}_\bullet(T,I)$ is straightforward. The difference between $q^{(h)}_\bullet(T,I)$ and $w^{(h)}_\bullet(T,I)$ is that the vertex set W and graph WG that are used to define $w^{(h)}_\bullet(T,I)$ might be missing a few keys that have more than one collision with locations in $\mathcal{L}$. Of course, these missing keys can cause other keys to fail to have eligible collisions. (The reason we have to consider this possibility is that a tree rooted in I might have keys with non-tree probes to other nodes of the witness graph, and which would therefore be omitted from W.) No matter. Ignoring keys can only cause $w^{(h)}_\bullet(T,I)$ to increase, so $q^{(h)}_\bullet(T,I) \le w^{(h)}_\bullet(T,I)$. However, the probability that a key $x \in D - D_T - x_{\alpha n} - W$ has, among its first h probes, two or more collisions with $\mathcal{L}$ is bounded by $O(\binom{h}{2})(\frac{|\mathcal{L}|}{n})^2$. So the probability that there is such a key is bounded by $O(\binom{h}{2})\frac{|\mathcal{L}|^2}{n}$. The only consequence of these additional keys is that a witness graph might switch from declaring vacancy to non-vacancy. Let

$$\underline{w}^{(h)}_\bullet(T,I) = \sum_{\substack{WG \in W^h_{empty}(T,I)\\ |\mathcal{L}|,\ |W| < 3hf(h)I_0}} \mathrm{Prob}^W_\bullet\{WG\}. \tag{26}$$

It follows that

$$\underline{w}^{(h)}_\bullet(T,I)(1 - |O(h^2\frac{|\mathcal{L}|^2}{n})|) \le q^{(h)}_\bullet(T,I) \le \underline{w}^{(h)}_\bullet(T,I) + \mathrm{Prob}\{|W| > 3hf(h)I_0 \text{ or } |\mathcal{L}| > 3hf(h)I_0\}. \tag{27}$$

Lemma 6 says that $|W| < 3hf(h)I_0$ and $|\mathcal{L}| < 3hf(h)I_0$ with a probability exceeding $1 - \frac{(1-\alpha)^{|I|}}{n}$. Hence, for $|I| = O(\log n)$:

$$\underline{w}^{(h)}_\bullet(T,I)(1 - O(\frac{\log^2 n}{n})) - \frac{(1-\alpha)^{|I|}}{n} \le q^{(h)}_\bullet(T,I) \le \underline{w}^{(h)}_\bullet(T,I)(1 + O(\frac{1}{n})).$$

This inequality even applies to DH_ψ, since the probability of two or more collisions just depends on h probes per key. It now follows that the closeness of the $w^{(h)}_\bullet(T,I)$-s as established in 3) also applies to the $q^{(h)}_\bullet(T,I)$-s as stated in 4). ∎

The only remaining step is to prove that witness graphs give multiplicative vacancy overestimators, and it suffices to establish the property in UH, since the vacancy overestimators are all equivalent, up to factors of $(1+O(\frac{\log^2 n}{n}))$.

Theorem 3.

1) $q_{UH}^{(h)}(t) \equiv q_{UH}^{(h)}(t,1)$ is a multiplicative vacancy overestimator for $M_{UH}^{(h)}(T,I)$. Moreover, for any fixed $\alpha < 1$, fixed h, and $T \subset [1, \alpha n]$ with $|I| = |T| = O(\log n)$,

$$q_{UH}^{(h)}(T,I) = (1 + \frac{O(|I|\log^2 n)}{n}) \prod_{i \le |T|} q_{UH}^{(h)}(T_i).$$

Let $\beta = \frac{1}{2}(1 + 2\alpha - \alpha^2 + \frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)})$. Let h be so large that $\beta < 1 - 1/h$. Let $\phi = 3\frac{\log n}{\log(1/\beta)}$, $\phi_0 = \frac{h^3}{2}[\phi \ln(\frac{1}{1-\alpha}) + 3\ln n]$, $f(h) = e^{2(h+1/2)^{2/3}}$, $\omega = 3hf(h)\phi_0 + h$, and suppose that $\psi \ge \log n + 6\phi_0 + 5h^2\omega + h\omega$. Then:

2) In DH and DH_ψ, $q_\bullet^{(h)}(t) = \mathrm{Prob}_\bullet\{M(T,I)\}$ is multiplicative, and

$$q_\bullet^{(h)}(t) \le 1 - \frac{t}{n} + \frac{(2\frac{t}{n} - (\frac{t}{n})^2)^{h+1}}{h^{3/2}(1-\frac{t}{n})} + \frac{O(\log^2 n)}{n}.$$

Proof: Let $\underline{W}_{empty}^h(T,I)$ denote the set of all witness graphs WG for (T,I) with $|\mathcal{L}| < 3hf(h)I_0$, $|W| < 3hf(h)I_0$, and which have a structure that declares all I-locations to be empty at the times prescribed by T. Recall that $\mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\}$ is the probability that the vertices included in the witness graphs behave as prescribed and $\mathrm{Prob}_{UH}^{no\text{-}hit}\{WG\}$ is the probability that no other key in $D - D_T - x_{\alpha n} - W$ has an eligible hit. Lemmas 5 and 6 establish that

$$\begin{aligned} q_{UH}^{(h)}(T,I) &= \left(1 + \frac{O(\log^2 n)}{n}\right) \sum_{WG \in \underline{W}_{empty}^h(T,I)} \mathrm{Prob}_{UH}^{pure}\{WG\} \qquad (28) \\ &= \left(1 + \frac{O(\log^2 n)}{n}\right) \sum_{WG \in \underline{W}_{empty}^h(T,I)} \mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\} \times \mathrm{Prob}_{UH}^{no\text{-}hit}\{WG\}, \end{aligned}$$

Let the forest WG be partitioned into the $|I|$ trees $WG_1, \ldots, WG_{|I|}$, rooted at the respective locations $I_1, \ldots, I_{|I|}$. It is easy to see that $\mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\} = (1 + O(\frac{|WG|^2}{n})) \prod_{i=1}^{|I|} \mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG_i\}$, where the factor $(1+O(\frac{|WG|^2}{n}))$ accounts for the constraint that the trees have disjoint embeddings. That is, by definition, WG requires the embeddings to be disjoint, but each individual WG_i as the witness structure for $M(T_i, I_i)$ has no such constraint for its embedding. Of course, the probability that WG_1, WG_2, …, $WG_{|I|}$ have a common embedding location is bounded by $O(h^2 \frac{|W|^2}{n})$.

There is another minor anomaly that has to be addressed. When considering all witness graphs for the event $M(T, I)$, the candidate keys must come from the sequence $D - D_T - x_{\alpha n}$. When doing the same for $M(T_j, I_j)$, the candidate key set is $D - D_{T_j} - x_{\alpha n}$. Thus, the number of candidate keys increases by as much as $|I| - 1$. If WG_i is a witness structure comprising keys from $D - D_T - x_{\alpha n}$, the probability that members from $D_T - D_{T_j}$ have an eligible hit is bounded by $\frac{h|WG_i| \times |I|}{n}$. So suppose WG is the witness set for $M(T, I)$, and suppose that WG declares $M(T, I)$ to be true. WG_i, when viewed as a candidate witness set for $M(T_i, I_i)$, will have almost the same statistics as it does as a part of the witness set for $M(T, I)$. The difference is that the items in $D_T - D_{T_i}$ might have an eligible hit with WG_i and produce a non-vacancy for I_i at T_i. But the probability of an eligible collision is just $O(\frac{h|WG_i| \times |I|}{n})$, which can lower the probability of vacancy by at most a factor of $1 - O(\frac{h|WG_i| \cdot |I|}{n})$. Thus, $\mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG\} = (1 + O(\frac{|WG| \cdot |I|}{n})) \prod_{1 \le i \le |I|} \mathrm{Prob}_{UH}^{\ell\text{-}hit}\{WG_i\}$ The exact same issue is revisited when we consider an approximate factorization of $\mathrm{Prob}_{UH}^{no\text{-}hit}\{WG\}$.

Let $\sigma^*(\tau, WG)$, for $x'_\tau \in D - D_T - x_{\alpha n} - W$, be the probability that x'_τ has an eligible hit to WG. Similarly, $\sigma^*(\tau, WG_i)$ is the analogous probability for $x'_\tau \in D - x_{T_i} - W_i$, where W_i is the vertex set associated with WG_i. Let $\sigma^{**}(\tau, WG_i)$, for $x'_\tau \in D - x_{T_i} - W_i$, be the probability that x'_τ has an eligible hit to WG_i conditioned on x_τ not having an eligible hit to $WG_1, WG_2, WG_3, \ldots, WG_{i-1}$. For the purposes of using multiplicative statistics of vacancy overestimators, we can define the behavior of x_j, $j \in I - (i)$ to be the same as any key $x \notin D_T$ (i.e. without conditioning based on the membership in D_T) when computing $\sigma^*(x_j, WG_i)$ and $\sigma^{**}(x_j, WG_i)$.

The proof of Lemma 5 notes that in UH, $\mathrm{Prob}_{UH}^{no\text{-}hit}\{WG\}$ can be expressed as $\prod_{\tau:\ x'_\tau \in D - D_T - x_{\alpha n} - W} (1 - \sigma^*(\tau, WG))$. By definition, $\prod_{\tau:\ x'_\tau \in D - D_T - x_{\alpha n} - W} (1 - \sigma^*(\tau, WG)) = \prod_{\tau:\ x'_\tau \in D - D_T - x_{\alpha n} - W} \prod_{1 \le j \le |I|} (1 - \sigma^{**}(\tau, WG_i))$.

It is straightforward to show that for $x'_\tau \in D - D_T - x_{\alpha n} - W$, $1 - \sigma^{**}(\tau, WG_i) = (1 - \sigma^*(\tau, WG_i))(1 + O(\frac{|WG| \cdot |WG_i|}{n^2}))$, since the conditioning effectively multiplies $\sigma^*(\tau, WG)$ by $\frac{n}{n - O(|WG|)}$: $\sigma^{**}(\tau, WG_i) = \sigma^*(\tau, WG_i)(1 - O(\frac{|WG|}{n}))$.

Similarly, $\prod_{\tau\in I-i}(1-\sigma^{**}(\tau,WG_i)) = 1-O(\frac{|WG_i|I|}{n})$.

Putting these observations together gives:

$$
\begin{aligned}
q^{(h)}_{UH}(T,I) = &\sum_{WG\in \underline{W}^h_{empty}(T,I)} \\
&(1+\frac{O(|WG|^2)}{n})\mathrm{Prob}^{\ell\text{-}hit}_{UH}\{WG\}\prod_{i=1}^{|I|}\prod_{\tau:\ x'_\tau\in D'-W}(1+\frac{O(|WG||WG_i|)}{n^2})(1-\sigma^*(\tau,WG_i)) \\
= &\sum_{WG\in \underline{W}^h_{empty}(T,I)} \\
&(1+\frac{O(|WG|^2)}{n})\prod_{i=1}^{|I|}\mathrm{Prob}^{\ell\text{-}hit}_{UH}\{WG_i\}(1+\frac{O(|WG||WG_i|)}{n})\prod_{\tau:\ x'_\tau\in D'-W}(1-\sigma^*(\tau,WG_i)) \\
= &\sum_{WG\in \underline{W}^h_{empty}(T,I)}(1+\frac{O(|WG|^2)}{n})\prod_{i=1}^{|I|}\mathrm{Prob}^{\ell\text{-}hit}_{UH}\{WG_i\}\prod_{\tau:\ x'_\tau\in D-\{x_{T_i}\}-W_i}(1-\sigma^*(\tau,WG_i)) \\
\le &\prod_{i=1}^{|I|}\sum_{WG_i\in \underline{W}^h_{empty}(T_i,I_i)}(1+\frac{O(|WG||WG_i|)}{n})\mathrm{Prob}^{\ell\text{-}hit}_{UH}\{WG_i\}\prod_{\tau:\ x'_\tau\in D-\{x_{T_i}\}-W_i}(1-\sigma^*(\tau,WG_i)),
\end{aligned}
$$

where the inequality in the last line accounts for the fact that outer factors have the constraint $WG_i = O(\log n)$ rather than $\sum_i |WG_i| = O(\log n)$. Of course the additional terms are statistically negligible, since they correspond to instances where $|WG| > 3hf(h)I_0$. It follows that

$$q^{(h)}_{UH}(T,I) = (1+O(\frac{\log^2 n}{n})\prod_{i=1}^{|I|} w^{(h)}_{UH}(T_i,I_i).$$

Appealing to Lemma 6.4 gives:

$$q^{(h)}_{UH}(T,I) = (1+O(\frac{|I|\log^2 n}{n}))\prod_{i=1}^{|I|} q^{(h)}_{UH}(T_i), \tag{29}$$

which establishes 1).

Claim 2) is a direct consequence of 1), the definition of a multiplicative vacancy overestimator, Lemma 4, Corollary 4, and Lemma 6. Lastly, Corollary 4 shows that $Err(k,\alpha n) = O(1/n)$, provided we set $\phi = 3\frac{\log n}{\log(1/\beta)}$. Lemma 6 determines a sufficient size for ψ. ▮

4. Reducing the errors

In retrospect, it is clear that many of the counting steps used crude overestimates and approximations. Thus, it is reasonable to expect that the performances as calculated in the three hashing models are even closer than the bounds we have established. There is one improvement that requires rather little work, and which turns out to be quite useful. Accordingly, we apply a bootstrapping argument to reduce the asymptotic error bounds in the statistics for $M^{(h)}(T, I)$, for the three hashing models.

Theorem 4. Let $\beta = \frac{1}{2}(1 + 2\alpha - \alpha^2 + \frac{(2\alpha-\alpha^2)^{h+1}}{h^{3/2}(1-\alpha)}$. Let h be so large that $\beta < 1 - 1/h$. Let $\phi = 3\frac{\log n}{\log(1/\beta)}$, $\phi_0 = \frac{h^3}{2}[\phi \ln(\frac{1}{1-\alpha}) + 5 \ln n]$, $f(h) = e^{2(h+1/2)^{2/3}}$, $\omega = 3hf(h)\phi_0 + h$, and suppose that $\psi \geq \log n + 6\phi_0 + 5h^2\omega + h\omega$. Then

1) In UH, DH, and DH_ψ, with $|I| \leq \phi$:
   $q_\bullet^{(h)}(T, I) = (1 + O(\frac{|I|^3}{n}))q_{UH}^{(h)}(T, I)$.

2) Furthermore, $q_{UH}^{(h)}(T, I) = (1 + O(\frac{|I|^3}{n}))\prod_{i\in I} q_{UH}^{(h)}(T_i, 1)$.

3) Let $probe^{(h)}(j)$ be, in the three hashing models, the expected overestimate of the probe count to insert x_j. Let $q^{(h)}(j) = q_{UH}^{(h)}(j, 1)$.

   Then

$$\mathrm{E}_\bullet[probe_\bullet^{(h)}(t)] = \frac{1}{\sqrt{1 - \frac{2}{n}\sum_{0<j<t} q^{(h)}(j)}} + O(\frac{1}{n}).$$

Proof: A careful reading of the preceding proofs shows that (28) can be written as:

$$q_{UH}^{(h)}(T, I) = \sum_{WG \in \underline{W}_{empty}^{h}(T,I)} \left(1 + \frac{O(|WG|^2)}{n}\right) \mathrm{Prob}_{UH}^{pure}\{WG\}.$$

Similarly, (26) and (27) can be combined more carefully to read:

$$\sum_{\substack{WG \in W_{empty}^{h}(T,I) \\ |WG| < 3hf(h)I_0}} (1 - O(h^2\frac{|WG|^2}{n}))\mathrm{Prob}_\bullet^W\{WG\}$$

$$\leq q_\bullet^{(h)}(T, I) \leq \mathrm{Prob}\{|WG| > 3hf(h)I_0\} + \sum_{\substack{WG \in W_{empty}^{h}(T,I) \\ |WG| < 3hf(h)I_0}} (1 + O(h^2\frac{|WG|^2}{n}))\mathrm{Prob}_\bullet^W\{WG\}.$$

In the following, WG_i will be witness structures for the pair (T_i, I_i) and of size $O(\log n)$. Similarly, $\underline{W}_{empty}(i)$ will be the set of witness structures that declare I_i vacant at time T_i, and have a size bounded by ϕ_0. Notice that ϕ_0 is slightly larger than its counterpart in Lemma 6, so that $\text{Prob}_\bullet\{|\mathcal{L}| \geq 3hf(h)I_0\} < \frac{(1-\alpha)^{|I|}}{2n^3}$, and $\text{Prob}_\bullet\{|W| \geq 3hf(h)I_0\} < \frac{(1-\alpha)^{|I|}}{n^3}$, provided n is sufficiently large. Consequently,

$$\frac{\sum_{\substack{WG\in W^h_{empty}(T,I)\\ |WG|>3hf(h)I_0}} |WG|^2 \text{Prob}^W_\bullet\{WG\}}{\sum_{WG\in W^h_{empty}(T,I)} \text{Prob}\{WG\}} < \frac{n^2\text{Prob}_\bullet\{|WG| \geq 3hf(h)I_0\}}{(1-\alpha)^I} = O(1/n).$$

Define the (evaluated) conditional expectation for $|WG|^2$ given that $|WG|$ is of size $3hf(h)I_0$ or less, and declares I empty at respective times T:

$$\Gamma^{(2)}(T,I) = \frac{\sum_{WG\in \underline{W}^h_{empty}(T,I)} |WG|^2\text{Prob}\{WG\}}{\sum_{WG\in \underline{W}^h_{empty}(T,I)} \text{Prob}\{WG\}},$$

and

$$\Gamma^{(2)}_i(T,I) = \frac{\sum_{WG\in \underline{W}^h_{empty}(T,I)} |WG_i|^2\text{Prob}\{WG\}}{\sum_{WG\in \underline{W}^h_{empty}(T,I)} \text{Prob}\{WG\}}.$$

The model identifier has been suppressed, but we understand that $\Gamma^{(2)}(T,I)$ is evaluated in the same hashing model as $\text{Prob}\{WG\}$. If we can show that $\Gamma^{(2)}(T,I) = O(|I|^2)$, then all of the asymptotic error terms will have each $O(\log n)$ factor replaced by $O(|I|)$. This improvement would suffice to establish parts 1) and 2) of Theorem 4. As a consequence, the errors resulting from the use of (8) to achieve the computation of Corollary 2 will then establish part 3), since the resulting error bound computation is the same as that in (15), which results in an $O(\frac{1}{n})$ error.

We now show that

$$\Gamma^{(2)}(T,I) = O(|I|^2),$$

when $|I| = O(\log n)$. For convenience, we suppress the minutiae posed by probabilities that are only approximately $\frac{1}{n}$, and the fact that mild conditioning concerns suggest that we should replace the denominator n by a slightly smaller value.

There follows:

$$\begin{aligned}
\Gamma_i^{(2)}(T,I) &= \frac{\sum_{WG\in\underline{W}_{empty}(T,I)} |WG_i|^2 \text{Prob}\{WG\}}{\sum_{WG\in\underline{W}_{empty}(T,I)} \text{Prob}\{WG\}} \\
&= (1+O(|I|\frac{\log^2 n}{n})) \\
&\quad\times \frac{(\sum_{WG_i\in\underline{W}_{empty}(i)} |WG_i|^2 \text{Prob}\{WG_i\}) \prod_{j\neq i}(\sum_{WG_j\in\underline{W}_{empty}(j)} \text{Prob}\{WG_j\})}{\prod_j(\sum_{WG_j\in\underline{W}_{empty}(j)} \text{Prob}\{WG_j\})} \\
&= (1+O(\frac{\log^3 n}{n})) \times \frac{(\sum_{WG_i\in\underline{W}_{empty}(i)} |WG_i|^2 \text{Prob}\{WG_i\})}{\sum_{WG_i\in\underline{W}_{empty}(i)} \text{Prob}\{WG_i\}} \\
&< (1+O(\frac{\log^3 n}{n}))\frac{\text{E}[|WG_i|^2]}{\sum_{WG_i\in\underline{W}_{empty}(i)} \text{Prob}\{WG_i\}}, \text{ where E}[|WG_i|^2] \text{ is not conditioned} \\
&\le (1+O(\frac{\log^3 n}{n}))\frac{\text{E}[|WG_i|^2]}{1-\alpha} = O(E[|WG_i|^2]).
\end{aligned}$$

We bound $E[|WG_\ell|^2]$ as follows. Let $\mathcal{L}_\ell(k,t)$, for $k=1,2,\ldots,h-1$, be size of the eligible collision location sets reachable from I_ℓ at time t in the construction given in Definition 16.

The first step is to bound $\text{E}[(\sum_j \mathcal{L}_\ell(k,t))^2]$. The second is to do the same for the set of witness keys that collide with these locations.

The recurrence for $\mathcal{N}_\ell(k,t) \equiv \text{E}[\mathcal{L}_\ell(k,t)]$ reads:

$$\mathcal{N}_\ell(0,t)=1, \text{ and } \mathcal{N}_\ell(j,0)=0,\ j=1,2,\ldots,h-1;$$
$$\mathcal{N}_\ell(k,t)=\mathcal{N}_\ell(k,t-1)+\sum_{i<k}\mathcal{N}_\ell(i,t-1)\frac{k-i}{n},\ k>0;\ t>0.$$

A more explicit formulation, which characterizes the actual random variables as opposed to the expectations is momentarily necessary. It is:

$$\mathcal{L}_\ell(0,t)=1, \text{ and } \mathcal{L}_\ell(j,0)=0,\ j=1,2,\ldots,h-1;$$
$$\mathcal{L}_\ell(k,t)=\mathcal{L}_\ell(k,t-1)+\sum_{i<k}(k-i)B(i,t-1),\ k>0;\ t>0,$$

where $B(i,t-1)$ is a Bernoulli Trial with probability of success equal to $\frac{\mathcal{L}_\ell(i,t-1)}{n}$. Moreover, $B(1,t-1)$, $B(2,t-1)$, $B(3,t-1)$, $\ldots$ are mutually exclusive: no more that one can be a success.

The square satisfies:

$$\mathcal{L}_\ell^2(k,t)=\mathcal{L}_\ell^2(k,t-1)+2\sum_{i<k}\mathcal{L}_\ell(k,t-1)(k-i)B(i,t-1)+\sum_{i<k}(k-i)^2B(i,t-1),\ k>0;\ t>0.$$

Let $N_\ell^{(2)}(k,t) = \mathrm{E}[|\mathcal{L}(k,t)|^2]$. Evaluating the expectation gives:

$$N_\ell^{(2)}(k,t) = N_\ell^{(2)}(k,t-1) + 2\sum_{i<k}\mathrm{E}[\frac{\mathcal{L}_\ell(k,t-1)(k-i)\mathcal{L}_\ell(i,t-1)}{n}] + \mathrm{E}[\sum_{i<k}\frac{(k-i)^2}{n}\mathcal{L}_\ell(i,t-1)].$$

Applying the inequality $2abc \le a^2b + c^2b$ gives

$$N_\ell^{(2)}(k,t) \le N_\ell^{(2)}(k,t-1) + \sum_{i<k}\mathrm{E}[\frac{\mathcal{L}_\ell^2(k,t-1)(k-i) + \mathcal{L}_\ell^2(i,t-1)(k-i)}{n}] + \mathrm{E}[\sum_{i<k}\frac{(k-i)^2}{n}\mathcal{L}_\ell(i,t-1)].$$

So

$$N_\ell^{(2)}(k,t) \le N_\ell^{(2)}(k,t-1) + \sum_{i<k}\frac{N_\ell^{(2)}(k,t-1)(k-i)}{n} + \mathrm{E}[\sum_{i<k}\frac{k-i}{n}(\mathcal{L}_\ell(i,t-1)+h/2)^2],\ k>0;\ t>0.$$

Consequently, an overestimate for $N_\ell^{(2)}(k,t)$ can be formulated by:

$$N_\ell^{(2)}(0,t) = h^2, \text{ and } N_\ell^{(2)}(j,0) = h^2,\ j = 1,2,\ldots,h-1;$$
$$N_\ell^{(2)}(k,t) = N_\ell^{(2)}(k,t-1)(1+\frac{h^2}{n}) + \sum_{i<k}N_\ell^{(2)}(i,t-1)\frac{k-i}{n},$$

and even more so by

$$N_\ell^{(2)}(0,t) = h^2(1+\frac{h^2}{n})^t, \text{ and } N_\ell^{(2)}(j,0) = h^2,\ j = 1,2,\ldots,h-1;$$
$$N_\ell^{(2)}(k,t) = N_\ell^{(2)}(k,t-1)(1+\frac{h^2}{n}) + (1+\frac{h^2}{n})\sum_{i<k}N_\ell^{(2)}(i,t-1)\frac{k-i}{n}.$$

Set $N_\ell^{(2)}(k,t) = (1+\frac{h^2}{n})^t\mu_\ell^{(2)}(k,t)$. The resulting system for μ_ℓ reads:

$$\mu_\ell^{(2)}(0,t) = h^2, \text{ and } \mu_\ell^{(2)}(j,0) = h^2,\ j = 1,2,\ldots,h-1;$$
$$\mu_\ell^{(2)}(k,t) = \mu_\ell^{(2)}(k,t-1) + \sum_{i<k}\mu_\ell^{(2)}(i,t-1)\frac{k-i}{n},$$

which is of the same form as (21), and therefore

$$\mu_\ell^{(2)}(k,t) \le h^2 e^{2(k+1/2)^{2/3}},$$

whence

$$N_\ell^{(2)}(k,t) \le h^2 e^{\frac{h^2 t}{n}} e^{2(k+1/2)^{2/3}} = O(1).$$

Applying the inequality $(\sum_{i=1}^m x_i)^2 \le m\sum_{i=1}^m x_i^2$, gives:

$$\mathrm{E}[(\sum_{\ell,k}|\mathcal{L}_\ell(k,\alpha n)|)^2] \le h|I|\sum_{1\le\ell\le|I|}\ \sum_{0\le i<h} N_\ell^{(2)}(i,n) = O(|I|^2).$$

The proof for $\mathrm{E}[|W_\ell|^2]$ is similar, and can also be based on the bounds for $\Gamma^{(2)}$. It follows that $\mathrm{E}[|WG|^2] = O(|I|^2)$ as claimed. ∎

5. Conclusions

We have shown that in double hashing, a universal family of $\log n$-wise independent hash functions can give nearly optimal performance for any fixed load bounded below 1.

These results comprise a significant step toward understanding why extremely simple functions seem to perform so well when used to double hash arbitrary values into a partially filled table. Indeed, it is quite conceivable that real data, when hashed by such functions, might yield sequences that exhibit $O(\log n)$-wise independence.

Our proof analyzed local and global hashing interactions separately, and used analytic tools to measure complicated but weakly correlated events in terms of simpler independent processes.

6. Appendix

This section contains two technical Lemmas, which can simplify large deviation calculations in cases of full and limited independence. Lemma A.2 is a special case of Theorem 6 in [15].

Lemma A1. Let $X = \sum_{i=1}^{n} X_i$ be the sum of n mutually independent Bernoulli trials $X_1, \ldots, X_n$, where $\mathrm{Prob}\{X_i = 1\} = p_i$. Then

$$\text{for any } C > 0, \text{ and } 0 \le \epsilon \le 1, \ \mathrm{Prob}\{X \ge (1+\epsilon)\mathrm{E}[X] + C\} \le e^{-1.25\epsilon C}.$$

Proof: Let $p = \frac{\mathrm{E}[X]}{n}$. According to Hoeffding, [Ho-63]:

$$\mathrm{Prob}\{X \ge (1+\delta)\mathrm{E}[X]\} \le \left(\frac{1}{1+\delta}\right)^{(1+\delta)\mathrm{E}[X]} \left(\frac{1-p}{1-(1+\delta)p}\right)^{n-(1+\delta)\mathrm{E}[X]} \le \left(\frac{1}{1+\delta}\right)^{(1+\delta)\mathrm{E}[X]} e^{\delta \mathrm{E}[X]}.$$

Let $C = (\delta - \epsilon)\mathrm{E}[X]$. It suffices to show that for any $\delta > 0$, and $0 \le \epsilon \le 1$,

$$\left(\frac{1}{1+\delta}\right)^{(1+\delta)\mathrm{E}[X]} e^{\delta \mathrm{E}[X]} \le e^{-1.25(\epsilon)(\delta-\epsilon)\mathrm{E}[X]}.$$

(For $\delta \le 1$, a simple derivation gives the slightly better bound where the 1.25, is replaced by $\frac{3}{2}$.)

We therefore need only show that for all $0 \le \epsilon \le 1$,

$$\left(\frac{1}{1+\delta}\right)^{(1+\delta)} e^{\delta} \le e^{-1.25\epsilon(\delta-\epsilon)}.$$

Let $g(\delta,\epsilon) = \left(\frac{1}{1+\delta}\right)^{(1+\delta)} e^{\delta+1.25\epsilon(\delta-\epsilon)}$. For any fixed ϵ, $g(\delta,\epsilon)$ attains its maximum at $1+\delta = e^{1.25\epsilon}$. Therefore $\log g(\delta,\epsilon) \le \log g(e^{1.25\epsilon}-1,\epsilon) = e^{1.25\epsilon} - 1 - 1.25\epsilon - 1.25\epsilon^2$. For $\epsilon > 0$, the expression

$e^{1.25\epsilon} - 1 - 1.25\epsilon - 1.25\epsilon^2$ first decreases and then increases since it is initially decreasing and its derivative is the sum of a linear function and a function with a rapidly increasing derivative. Hence $f(\epsilon) = g(e^{1.25\epsilon} - 1, \epsilon)$ first decreases and then increases. It follows that $f(\epsilon) \leq \max\{f(0), f(1)\}$, for $\epsilon \in [0,1]$. Since $\log f(0) = 0$, and calculation shows that $\log f(1) < -.009$, we see that $g(\delta,\epsilon) < 1$, for all δ, and $\epsilon \in [0,1]$. ∎

In the following lemma, a collection of ψ-wise independent random variables is a collection where any subset of ψ of fewer random variables are independent.

Lemma A2. Let $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_n$ be Bernoulli trials with probabilities of success $\mathrm{E}[X_i] = \mathrm{E}[Y_i] = p_i$. Let $X = \sum_{i=1}^n X_i$. Suppose that the Y_i-s are mutually independent, and that the X_i-s are ψ-wise independent. Let $I = \{i_1, i_2, \ldots, i_k\}$, and let $\widehat{I} = [1,n] - I$. Let $p(I) = \mathrm{Prob}\{\left(\bigwedge_{j\in I}(Y_{i_j} = 1)\right) \wedge \left(\bigwedge_{j\in\widehat{I}}(Y_{i_j} = 0)\right)\}$, and let $p_\psi(I) = \mathrm{Prob}\{\left(\bigwedge_{j\in I}(X_{i_j} = 1)\right) \wedge \left(\bigwedge_{j\in\widehat{I}}(X_{i_j} = 0)\right)\}$, so that the subscript ψ indicates that the event is with respect to the fully ψ-wise independent trials $X_1, X_2, \ldots, X_n$.

(1) For $\psi \geq C + k + e\mathrm{E}[X] - \log\left(\prod_{i\in\widehat{I}}(1-p_i)\right)$ and some ϵ where $|\epsilon| \leq 1$,
$$p_\psi(I) = p(I)(1 + \epsilon e^{-C})$$

(2) If $\forall i$: $p_i \leq 1/2$, then for $\psi \geq C + k + 5\mathrm{E}[X]$ and some ϵ where $|\epsilon| \leq 1$,
$$p_\psi(I) = p(I)(1 + \epsilon e^{-C}).$$

Proof: The proof of Lemma A2 is a special case of Theorem 6 in [15]. It is given here for completeness.

We use standard inclusion-exclusion to estimate the probability $p_\psi(I)$ as follows.
$$\begin{aligned} p_\psi(I) &= \mathrm{Prob}_\psi\{\left(\bigwedge_{j\in I}(X_j = 1)\right) \wedge \left(\bigwedge_{\ell\notin I}(X_\ell = 0)\right)\} \\ &= \mathrm{Prob}_\psi\{\bigwedge_{j\in I}(X_j = 1)\}\sum_{\ell=0}^{n}\ \sum_{i_{k+1}<\ldots<i_{k+\ell}\notin I}(-1)^\ell\mathrm{Prob}_\psi\{\bigwedge_{j\in i_{k+1}<\ldots<i_{k+\ell}}(X_j = 1)\}. \end{aligned}$$

Truncating the outer summation at $\ell = \psi - k$ introduces an error that is bounded by the last term of the truncated sum. Let $p_\psi^T(k)$ and $p^T(k)$ denote these truncated sums, in the respective cases of ψ-wise and full independence. Since the first $\psi - k$ terms in the outer summation are the same for both fully and ψ-wise independent random variables, $p^T_\psi(k) = p^T(k)$. Furthermore, $\mathrm{Prob}_\psi\{\bigwedge_{j\in\{i_1,\ldots,i_{k+l}\}}(X_j = 1)\} = \prod_{j=1}^{k+l} p_{i_j}$. Hence

$$p_\psi(k) = p^T(k) - (-1)^{\psi-k}\delta_\psi[\prod_{j\in I} p_{i_j}] \sum_{\substack{i_1<\ldots<i_{\psi-k}\\ i_j\notin I}} \prod_{j=1}^{\psi-k} p_{i_j},$$

for some $\delta_\psi \in [0,1]$, and an identical inequality holds without the ψ subscripts. Hence

$$|p_\psi(I) - p(I)| \leq [\prod_{j\in I} p_{i_j}] \sum_{i_1<\ldots<i_{\psi-k}} \prod_{j=1}^{\psi-k} p_{i_j}.$$

Since $\sum_{i_1<\ldots<i_{\psi-k}} \prod_{j=1}^{\psi-k} p_{i_j}$ is maximized when all p_{i_j} are equal, the error $|p_\psi(k) - p(k)|$ is bounded by

$$[\prod_{j\in I} p_{i_j}]\binom{n}{\psi-k}\left(\frac{p_1+p_2+\ldots+p_n}{n}\right)^{\psi-k} \leq [\prod_{j\in I} p_{i_j}](\mathrm{E}[X])^{\psi-k}/(\psi-k)!,$$

To get multiplicative error bounds, we need that $(\mathrm{E}[X])^{\psi-k}/(\psi-k)! \leq e^{-C}\prod_{j\notin I}(1-p_j)$. Setting $\psi - k = e\mathrm{E}[X] - \log\left(\prod_{j\notin I}(1-p_j)\right) + C$, gives:

$$(\mathrm{E}[X])^{\psi-k}/(\psi-k)! \leq \left(\frac{e\mathrm{E}[X]}{\psi-k}\right)^{\psi-k} \leq \left(1 + \frac{\log\left(\prod_{j\notin I}(1-p_j)\right) - C}{\psi-k}\right)^{\psi-k} \leq e^{-C}\prod_{j\notin I}(1-p_j).$$

This proves (1). The second inequality follows immediately by observing that if, say, $\forall i$: $p_i \leq \frac{1}{2}$, then

$$-\log\left(\prod_{j\notin I}(1-p_j)\right) = \sum_{j\notin I}\sum_{k>0}\frac{p_j^k}{k} \leq \sum_{j\notin I} 2p_j \sum_{k>0}\frac{(1/2)^k}{k} = -\sum_j 2p_j \log\frac{1}{2} < (1.4)\mathrm{E}[X]. \quad \blacksquare$$

References

[**1**] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[**2**] M. Ajtai, J. Komlós, E. Szemerédi. There Is No Fast Single Hashing Algorithm, *IPL*, 7,6, 1978, pp. 270–273.

[**3**] B. Bollobás, A.Z. Broder, and I. Simon. The cost distribution of clustering in random probing, *JACM*, **37**(2), April, 1990, pp. 224–237.

[**4**] R Brent. Reducing the Retrieval Time of Scatter Storage Techniques, *CACM*, **16**(2), 1973, pp. 105–109.

[**5**] H. Chernoff. A measure of asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations, *Ann. Math. Statist.*, 23 (1952), pp. 493-507.

[**6**] J.L. Carter and M.N. Wegman. Universal Classes of Hash Functions, *JCSS*, 18, 1979, pp. 143–154.

[**7**] D.H. Greene and D.E. Knuth. *Mathematics for the analysis of algorithms*, Birkhauser, 1990.

[**8**] L. Guibas and E. Szemerédi. The Analysis of Double Hashing, *JCSS*, 16, 1978, pp. 226–274.

[**9**] W. Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *J. Am. Stat. Ass.*, 58(1963), 13–30.

[**10**] D.E. Knuth. *The Art of Computer Programming, Vol. 3: Sorting and Searching*, Addison-Wesley, Reading, Mass. 1973.

[**11**] G. Lueker and M. Molodowitch. More Analysis of Double Hashing, *Combinatorica*, 13, 1993, pp. 83–96.

[**12**] K. Mehlhorn. On the Program size of Perfect and Universal Hash functions, *Proc. 23rd Ann. Symp. on Foundations of Computer Science*, 1982, pp. 170–175.

[**13**] K. Mehlhorn. *Data Structures and Algorithms 1: Sorting and Searching*, Springer-Verlag, Berlin Heidelberg, 1984.

[**14**] J.P. Schmidt and A. Siegel. On aspects of universality and performance for closed hashing, *21*st *STOC*, May, 1989, pp. 355–366.

[**15**] J.P. Schmidt, A. Siegel, and A. Srinivasan. Chernoff-Hoeffding Bounds for Applications with Limited Independence, *SIAM J. on Discrete Mathematics*, 8(2), 1995, pp. 223–250.

[**16**] A. Siegel. On universal classes of fast hash functions, their time-space tradeoff, and their applications, *Proc. 30th Ann. Symp. on Foundations of Computer Science*, Oct., 1989, pp. 20–25.

[**17**] A. Siegel and J.P. Schmidt. Closed hashing is computable and optimally randomizable with universal hash functions, submitted.

[**18**] J.D. Ullman. A Note on the Efficiency of Hash Functions, *JACM*, Vol 19, 1972, pp. 569–575.

[**19**] M.N. Wegman and J.L. Carter. New Hash Functions and Their Use in Authentication and Set Equality, Journal of Comp. Syst. Sci. 22, 1981, pp. 265–279.

[**20**] A.C. Yao. Uniform Hashing Is Optimal, *JACM*, Vol 32, No. 3, July, 1985, pp. 687–693.