$O(1)$ Insertion for Random Walk d-ary Cuckoo Hashing up to the Load Threshold*

Tolson Bell† and Alan Frieze‡
Department of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213
U.S.A.

December 3, 2025

Abstract

The random walk d-ary cuckoo hashing algorithm was defined by Fotakis, Pagh, Sanders, and Spirakis to generalize and improve upon the standard cuckoo hashing algorithm of Pagh and Rodler. Random walk d-ary cuckoo hashing has low space overhead, guaranteed fast access, and fast in practice insertion time. In this paper, we give a theoretical insertion time bound for this algorithm. More precisely, for every $d \geq 3$ random hashes, let c_d^* be the sharp threshold for the load factor at which a valid assignment of cm objects to a hash table of size m exists with high probability. We show that for any $d \geq 3$ hashes and load factor $c < c_d^*$, the expectation of the random walk insertion time is $O(1)$, that is, a constant depending only on d and c but not m.

*A preliminary version of this paper appeared in the proceedings of the Foundations of Computer Science (FOCS) 2024 Conference.

†thbell@cmu.edu. Research supported in part by NSF Graduate Research Fellowship grant DGE 2140739.

‡frieze@cmu.edu. Research supported in part by NSF grant DMS 1952285

1 Introduction

1.1 Problem Statement and Theorem

In random walk d-ary cuckoo hashing, the goal is to store a set of objects X where $X \subseteq U$ for a universe of elements U in a hash table with table slots Y given d hash functions $h_1, \ldots, h_d : U \to Y$. Following previous literature, we will take each hash function to be chosen independently and uniformly at random from all functions from X to Y. When a new object x_1 is inserted, a uniformly random $i_1 \in [d]$ is chosen, and x_1 is placed into position $h_{i_1}(x_1)$. If $h_{i_1}(x_1)$ was already occupied, we remove its previous occupant, x_2, and reinsert x_2 by the same algorithm (choosing a new $i_2 \in [d]$ and putting x_2 into $h_{i_2}(x_2)$). This iterative algorithm terminates when we insert an object into an empty slot.

An object x is queried by checking $h_1(x), \ldots, h_d(x)$, which takes constant time for constant d. If we want to remove x, we simply delete it from its slot in the hash table. Thus access and deletion are both guaranteed to be fast.

Let $n = |X|$ and $m = |Y|$. We represent the hash functions as a bipartite graph with vertex set (X, Y), and for each $x \in X$, edges from x to $h_1(x), \ldots, h_d(x)$. For a set $W \subseteq X$, we let $N(W)$ denote its set of neighbors in Y. An analogous definition is assumed for $Z \subseteq Y$. Finally, we replace $N(\{u\})$ by $N(u)$ for singleton sets.

For the insertion process to terminate, it must be true that there is an assignment of every object to a slot such that no slot has more than one object assigned to it and every object x is assigned to $h_i(x)$ for some $1 \leq i \leq d$. This can be represented as a matching of size n in the bipartite graph. We know by Hall's Theorem that such a matching exists if and only if $|N(W)| \geq |W|$ for every $W \subseteq X$.

Unless explicitly noted otherwise, all asymptotics in this paper are written for $n \to \infty$ (or equivalently, $m \to \infty$) with $n = cm$ for fixed $d \in \mathbb{N}$ and fixed constant load factor $c \in (0, 1)$. For instance, one could say that access and deletion need to query $\Theta(1)$ slots in the worst case, as our asymptotics suppress all factors depending on c and d. We use "with high probability" to mean with probability $1 - o(1)$ as $m, n \to \infty$ for fixed d and c. When we refer to "with high probability" events, we are always talking about things that happen with high probability over the choice of the random hash function on the already inserted objects. That is, we do not use this phrase to refer to the likelihood of events dependent on the initial hash of the newly inserted object or on the progression of the random walk.

There is a sharp threshold c_d^*, called the load threshold, for a matching of size n to exist in the bipartite graph; that is, there is a constant c_d^* such that if $c < c_d^*$ then there exists a matching with high probability and if $c > c_d^*$ then there with high probability does not exist a matching [DGM$^+$10, FP10, FM12].

Our result, which will stated more precisely in Section 2, is the following:

Theorem 1.1. *Assume that we have $d \geq 3$, $c < c_d^*$, and $n = cm$. Then with high probability over the random hash functions, we have that the expected insertion time for the random walk insertion process is $O(1)$.*
Additionally, under the same conditions, for any constant $C \geq 0$, there is a constant $C' = C'(C, c, d) = \Theta(1)$ such that for sufficiently large n and all $\ell \in \mathbb{N}$, the probability of the random walk insertion process taking more than ℓ steps is at most $C'\ell^{-C}$.

In other words, our main result is that the expected insertion time is a constant depending only on d and c but not n or m. Throughout the paper, we will use $\Theta(1)$ to denote constants that may depend on d or c but do not depend on n or m. We did not try to optimize the constant

in Theorem 1.1. By insertion time, we mean the number of reassignments, that is, the number of times we move an object to a different one of its hash functions during the insertion .

We do not explicitly consider deletions in this paper (consider building the hash table only), but, as explained in Section 11, our results are robust to any sequence of n^β oblivious (not adaptive to the hash values) deletions and insertions of new elements (excluding re-insertions of deleted elements) for some small $\beta = \Theta(1)$.

Note that we are required to take our statement to only hold with high probability over the choices of hash functions, as there is a non-zero chance that the hash functions will not have any valid assignment of objects to slots (will fail Hall's condition) and thus will have infinite insertion time. Our "with high probability" statements are true with high probability over not just one element's insertion but over the entire process of building a cuckoo hash table of cn elements for $c < c_d^*$. Therefore, we do get that with high probability the expected time to build a cuckoo hash table of cn elements for $c < c_d^*$ is $O(n)$.

1.2 Applications and Relation to Previous Literature

Standard cuckoo hashing was invented by Pagh and Rodler in 2001 [PR01] and has been widely used in both theory and practice. Their formulation, though originally phrased with two hash tables, is essentially equivalent to the case $d = 2$ of the algorithm described here. They showed that for all $c < c_2^* = 0.5$, one can get $O(1)$ expected insertion time, an analysis that was extended by Devroye and Morin [PR01, DM03]. Thus, cuckoo hashing is a data structure with $O(1)$ average-case insertion, $O(1)$ worst-case access and deletion, and only twice the amount of space that the elements themselves take up.

Cuckoo hashing can be seen as the "average-case" or "random graph" version of the "online bipartite matching with replacements" problem, with BFS insertion corresponding to the "shortest augmenting path" algorithm. Take any bipartite graph with $V = (X, Y)$ that contains a matching of size $|X|$. If elements of X and their incident edges arrive online, the amortized BFS insertion time was proven to be $O(\log^2(n))$ [BHR18]. The lower bound is $\Omega(\log(n))$ [GKKV95], which is matched if the vertex arrival order is randomized [CDKL09]. The previous paragraph shows that if the graph itself is random rather than worst-case, this $\Omega(\log(n))$ amortized insertion time bound is with high probability reduced to $\Theta(1)$.

d-ary cuckoo hashing was invented by Fotakis, Pagh, Sanders, and Spirakis in 2003 [FPSS03]. The main advantage of increasing d above 2 is that the load threshold increases. Even going from $d = 2$ to $d = 3$, the threshold c_d^* goes from 0.5 to ≈ 0.918, that is, with just one more hash function, we can utilize 91% of the hash table instead of 49%. The corresponding tradeoff is that the access time increases linearly with d. d-ary cuckoo hashing, also called generalized cuckoo hashing or improved cuckoo hashing, "has been widely used in real-world applications" [SHF$^+$17].

The exact value for c_d^* for all $d \geq 3$ was discovered via independent works by a number of authors [DGM$^+$10, FP10, FM12]. This combinatorial problem of finding the matching threshold in these random bipartite graphs (which can also be viewed as random d-uniform hypergraphs) is directly related to other problems like d-XORSAT [DGM$^+$10] and load balancing [GW10, FKP11].

The primary insertion algorithm analyzed by Fotakis, Pagh, Sanders, and Spirakis was not random walk insertion, but rather was BFS insertion. In BFS insertion, instead of selecting a random $i_1 \in [d]$ and hashing x_1 to $h_{i_1}(x_1)$, the algorithm finds the insertion path minimizing the number of reassignments. In other words, $i_1, \ldots, i_\ell \in [d]$ are chosen such that ℓ is minimized, where x_1 is to be hashed to $h_{i_1}(x_1)$, the removed object x_2 is to be hashed to $h_{i_2}(x_2)$, and so on until $h_{i_\ell}(x_\ell)$ is an empty slot. While BFS insertion requires more overhead to compute in practice, it is easier to analyze theoretically than random walk insertion. Fotakis, Pagh, Sanders,

and Spirakis proved that BFS insertion only requires $O(1)$ expected reassignments for load factor c when $d \geq 5 + 3\log(c/(1-c))$ [FPSS03]. Our Corollary 10.2 (which on its own has a shorter proof than Theorem 1.1) shows that this result extends to all $d \geq 3$ and $c < c_d^*$.

Fotakis, Pagh, Sanders, and Spirakis also introduced the insertion algorithm we study, random walk insertion, describing it as "a variant that looks promising in practice", since they did not theoretically bound its insertion time but saw its strong performance in experiments [FPSS03]. Random walk insertion requires no extra space overhead or precomputation. In a 2009 survey on cuckoo hashing, Mitzenmacher's first open question was to prove theoretical bounds for random walk insertion, calling random walk insertion "much more amenable to practical implementation" and "usually much faster" than BFS insertion [Mit09]. Insertion algorithms other than random walk or BFS have also been proposed, which have proven $O(n)$ total insertion time for $O(n)$ elements with high probability [KA19] or more evenly distributed memory usage [EGMP14]. However, random walk insertion "is currently the state-of-art method" [KA19].

For load factors somewhat below the load threshold and $d \geq 8$, the random walk expected insertion time was proven to with high probability be polylogarithmic by Frieze, Melsted and Mitzenmacher in 2009 [FMM09]. Fountoulakis, Panagiotou, and Steger then extended this result to show polylogarithmic expected insertion time holds with high probability for all $d \geq 3$ and $c < c_d^*$. The exponent of their logarithm is anything greater than $1 + b_d$, where $b_d = \frac{d+\log(d-1)}{(d-1)\log(d-1)}$ [FPS13]. Our proof uses techniques and lemmas from these two papers.

The average-case insertion time for hash tables is expected to be $O(1)$, however, not polylogarithmic. The first $O(1)$ random walk insertion bound was proven by Frieze and Johansson, who showed that for any load factor c, there exists some d such that there is $O(1)$ expected insertion time with high probability for d hashes at load factor c [FJ17]. However, their bounds only hold for large d and load factors significantly less than the load threshold, $c = 1 - O_{d\to\infty}(\log(d)/d)$, while it had been shown that $c_d^* = 1 - (1 + o_{d\to\infty}(1))(e^{-d})$ [DGM$^+$10, FP10, FM12].

To obtain a result that works for lower d, Walzer used entirely different techniques to prove $O(1)$ expected random walk insertion time with high probability up to the "peeling threshold", a load factor that is a lower number than the load threshold for any $d \geq 3$. The strongest result here is in the case $d = 3$, where Walzer gets $O(1)$ expected insertion up to load factor $c = .818$, compared to the optimal value $c_3^* = .918$. Walzer pointed out that there was no $d \geq 3$ for which $O(1)$ insertion was known up to the load threshold, saying, "Given the widespread use of cuckoo hashing to implement compact dictionaries and Bloom filter alternatives, closing this gap is an important open problem for theoreticians" [Wal22].

Theorem 1.1 is the first result to get $O(1)$ expected random walk insertion with high probability up to the load threshold for any $d \geq 3$, and works for all $d \geq 3$. The state of the art results are summarized in the tables below:

d	c_d^*	$O(1)$ expected insertion up to load factor...	Insertion time at $c = (1-\epsilon)c_d^* \; \forall \; \epsilon > 0$
2	[1]0.5	[1]0.5	[1]$O(1)$
3	[2]0.918	[3]0.818	[5]$O(\log^{3.664}(n))$
4	[2]0.977	[3]0.772	[5]$O(\log^{2.547}(n))$
5	[2]0.992	[3]0.702	[5]$O(\log^{2.152}(n))$
6	[2]0.997	[3]0.637	[5]$O(\log^{1.946}(n))$
Large	[2]$1 - (1 + o_{d\to\infty}(1))(e^{-d})$	[4]$1 - O_{d\to\infty}(\frac{\log d}{d})$	[5]$O(\log^{1+(\log d)^{-1}+O_{d\to\infty}(1/d)}(n))$

Prior work: [1][PR01, DM03] [2][DGM$^+$10, FP10, FM12] [3][Wal22] [4][FJ17] [5][FPS13]

d	c_d^*	$O(1)$ expected insertion up to load factor...	Insertion time at $c = (1-\epsilon)c_d^* \; \forall \; \epsilon > 0$
2	[1]0.5	[1]0.5	[1]$O(1)$
3	[2]0.918	[6]0.918	[6]$O(1)$
4	[2]0.977	[6]0.977	[6]$O(1)$
5	[2]0.992	[6]0.992	[6]$O(1)$
6	[2]0.997	[6]0.997	[6]$O(1)$
Large	[2]$1-(1+o_{d\to\infty}(1))(e^{-d})$	[6]$1-(1+o_{d\to\infty}(1))(e^{-d})$	[6]$O(1)$

Bounds after our work: [6]Theorem 1.1

2 Preliminaries

Our techniques to prove Theorem 1.1 build off the techniques of Fountoulakis, Panagiotou, and Steger [FPS13], who showed expansion-like properties of the bipartite hashing graph that hold with high probability. The main new ingredient is the introduction of recursively defined "bad" sets B_i for $i \in \mathbb{N}$. In this section, we will give some intuition for the overall proof structure and will more precisely state our results.

2.1 The Bipartite Graph and Matchings

We will study the form of the random walk where at each object removal, we choose a random one of the $d-1$ other hashes for the object that was just evicted (not returning it to the spot it was just evicted from). In Section 10, we will show that proving the expected run time of this non-backtracking random walk is $O(1)$ also proves the same of the random walk that chooses any one of the d hashes each time (including the one it was just removed from). Section 10 will also show an $O(1)$ expected insertion time for the BFS insertion for all $d \geq 3$ and all $c < c_d^*$.

Let $\mathcal{M}$ be the starting matching of size $n-1$ just before we insert the nth element. We can think of $\mathcal{M}$ as turning the bipartite graph into a directed graph, where an edge between object x and slot y is oriented $y \to x$ if x is matched to slot y, while it is oriented $x \to y$ if x is not matched to slot y. The cuckoo hashing procedure can be thought of as a random walk on this directed graph (with the random walk also changing the directions of some edges as it progresses).

Let $U \subseteq Y$ be the set of open spots in the hash table, which stays the same at each time step while the algorithm is running (as the algorithm terminates when it hits an open slot).

Our proof only relies on expansion-like properties of the bipartite graph on (X, Y) that hold with high probability. In particular, given the random bipartite graph, our result holds for any arbitrary starting matching $\mathcal{M}$ of objects to slots. As our expectation is over the hash values of the object being inserted, one fact we do need is that the hash values of this object being inserted are random among all slots in Y, after $\mathcal{M}$ is determined.

In other words, our theorem could be stated in more detail as follows:

Theorem 1.1**.** *Assume that we have $d \geq 3$, $c < c_d^*$, and $n = cm$. There exists an event $\mathcal{A}$ related to the hashes of the n objects that occurs with probability $1 - o(1)$ over uniformly random hash functions. Let $i \leq n$ and let $x \in X$ be the i-th element being inserted. If $\mathcal{A}$ occurs, then for any matching $\mathcal{M}$ of the first $i-1$ elements to slots of the hash table that is independent of the hash values of x, we have that the expectation (over the hash values of x and the choices of the random walk) of the insertion time for the random walk insertion process on x is $O(1)$.*

Corollary 9.1 tells us that furthermore, if $\mathcal{A}$ occurs, then for any constant $C_6 \geq 0$, there is a constant $C_7 = C_7(C_6, c, d)$ such that for sufficiently large n and all $\ell \in \mathbb{N}$, the probability (over

the hash values of x) of the random walk insertion process taking more than ℓ steps is at most $C_7\ell^{-C_6}$.

For convenience, we will consider inserting the n-th element throughout the paper, which we imagine inserting into a random slot in Y before determining the rest of its hash values. $\mathcal{A}$ can be taken to be one event over all n insertions; or, in other words, every "with high probability" statement in our proof is about bipartite graph structures that persist when elements are removed from X (in Lemmas 2.1, 3.1, 4.3, 5.1, and 7.1). Thus, our result implies an $O(n)$ expected time to build the hash table of n elements online.

Starting from some $x \in X$, we will use the convention that a walk of length i means that we do i reassignments, which corresponds to a walk of length $2i$ in the bipartite graph (X, Y). Let $W'_{+i}(x) \subseteq X$ be the set of all possible endpoints of a walk of length i starting from x under a particular matching $\mathcal{M}$. Therefore, $|W'_{+i}(x)| \leq (d-1)^i$, as we have $d-1$ choices of assignment at each step (at the first step, x is banned from choosing the slot that it is matched to under $\mathcal{M}$).

The B_i will be defined based on counting the number of "good" elements in $W'_{+i}(x)$. If we are considering a walk of length i from x, and there is some walk from x that lands on an unoccupied slot ($u \in U$) on the jth reassignment for some $1 \leq j \leq i$, that is extremely good, so we want to properly account for this. Intuitively, we want to imagine that the walk continues for $i-j$ more steps after it hits u, so u should count $(d-1)^{i-j}$ times as a good element of $W'_{+i}(x)$. For instance, if x has one neighbor $u \in U$, we want u to contribute $(d-1)^{i-1}$ dummy elements to $W'_{+i}(x)$. If there were also a different walk from x that hit that same u on the jth reassignment for some $1 \leq j \leq i$, then the same u would also contribute $(d-1)^{i-j}$ additional dummy elements, and so on.

Formally, we accomplish this as follows: for every $i \in \mathbb{N}$ and $x \in X$, let the set $\mathcal{U}_i(x)$ be a set of dummy elements (newly-introduced elements that are not in X), with

$$|\mathcal{U}_i(x)| = \sum_{j=1}^{i} (\#\text{walks from } x \text{ that hit } U \text{ on the } j\text{th reassignment})(d-1)^{i-j}.$$

Then we define $W_{+i}(x) = W'_{+i}(x) \sqcup \mathcal{U}_i(x)$. For $S \subseteq X$, we can similarly define $W_{+i}(S) = \bigcup_{x \in S} W_{+i}(x)$. We also define $W_{+\leq i}(x) = \bigcup_{j=0}^{i} W_{+j}(x)$ and $W_{+\leq i}(S)$ analogously for $S \subseteq X$.

Similarly, for $x \in X$ and $j \in \mathbb{N}$, let $W_{-j}(x)$ be defined to equal $\{w \in X : x \in \cup_{k=0}^{j} W_{+k}(w)\}$, that is, the set of elements that could reach x in at most j steps.

The BFS distance, or distance, of an object $w \in X$ from an object $x \in X$ under $\mathcal{M}$ is the minimal i such that $w \in W_{+i}(x)$. We can define BFS distances involving sets in the natural way, by minimizing over elements of those sets. We can similarly define the BFS distance of a slot $y \in Y$ from an object $x \in X$ as 1 plus the BFS distance from x to $N(y)$. For example, $\{h_1(x), \ldots, h_d(x)\}$ is exactly the set of slots at BFS distance 1 from x. Slots with no hash functions to them (isolated vertices in the bipartite graph) can be assumed to have infinite distance.

Lemma 2.1 (Corollary 2.3 of [FPS13])**.** *Let $d \geq 3$ and assume $n = cm$ for $c < c^*_d$. Then with high probability, we have that for any matching $\mathcal{M}$ and any $\alpha = \Theta(1) > 0$, there exists $M = \Theta(1)$ such that for the unoccupied vertices U of Y, we have that at most αn of the vertices of X have BFS distance $> M$ to U.*

Lemma 2.1 was first proven by the inventors of d-ary cuckoo hashing, but only under the weaker condition $d \geq 5 + 3\log(c/(1-c))$ for $n = cm$ [FPSS03]. (Note logarithms are natural unless denoted otherwise.) Corollary 2.3 of [FPS13] extended this lemma to all $d \geq 3$ and $c < c^*_d$. Some intuition for Lemma 2.1 will be given in Section 4.

Let $\alpha > 0$ be sufficiently small (but still $\Theta(1)$, to be set later) and take the corresponding $M = \Theta(1)$ as in Lemma 2.1. For our starting matching $\mathcal{M}$, let G be all vertices of X of BFS distance at most M from U. When we start at a vertex $g \in G$, we have at least a $(d-1)^{-M}$ chance that our random walk will finish in at most M more steps. (That is, there is at least a $(d-1)^{-M}$ chance that our random walk will be the BFS path, which has length $\leq M$.) Intuitively, this gives that the expected length on a random walk that stays inside G at every time t is at most $M(d-1)^M + M = \Theta(1)$ (though some technicalities arise due to the changing matching as the walk progresses). This shows intuitively that it suffices to only focus on the "worst" αn vertices for some small $\alpha = \Theta(1) > 0$.

2.2 Paper Outline

In Section 3, we will show an upper bound on the number of hashes that any set of slots receives. Iterating this will prove in Lemma 3.2 that, with high probability, for any $j \in \mathbb{N}$ and any $S \subseteq X$ with $|S| \leq n/12$, we have

$$|W_{-j}(S)| \leq \left(3d \log\left(\frac{n}{|S|}\right)\right)^j |S|.$$

As the random walk progresses, the matching of objects to slots changes as we perform evictions. Section 4 explains why this does not present a problem for our analysis.

Section 5 is our longest and most technical section. It begins by giving a lower bound on the number of distinct slots hashed to by a set of objects in Lemma 5.1. Applying Lemma 5.1 gives lower bounds on $|W_{+j}(S)|$ that hold for any $S \subseteq X$. This lemma proves to be a keystone of our proof, as the lower bounds on $|W_{+j}(S)|$ can be iteratively built up to give bounds on the likelihood of ending up in one set when starting from another. Intuitively, if $|W_{+j}(S)|$ is near its upper bound of $(d-1)^j|S|$, then the random walks starting in S do not concentrate on any small set of vertices, which helps our analysis.

In Section 6, we define the sets B_i, where we iteratively define

$$B_i = \{x \in X : \text{at least } 2(d-1)^i i^{-1.5} \text{ paths of length } i \text{ from } x \text{ end in } B_{i-1}\}.$$

In other words, if we are outside of B_i, we have at least $1 - 2i^{-1.5}$ probability that in i steps we will be outside of B_{i-1}. So, if the random walk begins outside of B_i, it is likely to iteratively progress from $X \setminus B_i$ to $X \setminus B_{i-1}$, and so on, to eventually reach the G of Lemma 2.1. The fact that $\sum_{i=1}^{\infty} 2i^{-1.5}$ converges means that we can achieve an arbitrarily small constant probability of this progression failing on any step.

Lemma 5.7 from Section 5 quickly implies that $|B_i| \leq (d-1)^{-i^2/4} n$. Section 6 continues on to show that walks starting "sufficiently far" from B_i have probability at least 0.97 of finishing in $O(i^2)$ steps.

Section 7 directly improves Lemma 3.2, with a more technical proof giving a stronger bound on $|W_{-j}(S)|$. This was not needed for anything before Section 7, but is needed in Section 8, which finishes the proof of $O(1)$ insertion by improving the "with probability at least 0.97" statement to an expected insertion time. Roughly, we can show that if a walk starting in $X \setminus B_i$ fails to finish in $O(i^2)$ steps, then we are still likely to be outside of $X \setminus B_{9i}$ and can attempt another run.

Section 9 proves stronger tail bounds on the insertion time of the random walk, that is, an upper bound on the probability that the random walk will take at least ℓ steps.

Section 10 extends our work to show $O(1)$ insertion for BFS insertion, as well as the random walk procedure that chooses a random one of the d hashes each time rather than excluding the one hash from which the object was just evicted.

Finally, Section 11 discusses possible future improvements on our results.

3 Bounding the Number of Paths to any Set

To show that reaching some bad set is unlikely, we want to upper bound the probability of reaching some small set, which we can later combine with a proof of bad sets being small. To accomplish this, we need to bound the number of neighbors that a small set can have.

Lemma 3.1. *For any $d \geq 3$ and $c < c_d^*$, we have with high probability that there is not a set $Z \subseteq Y$ with $|Z| \leq n/12$ such that $|N(Z)| \geq 3d \log\left(\frac{n}{|Z|}\right)|Z|$.*

Proof. First, imagine fixing $Z \subseteq Y$, then randomly choosing the edges of our graph. Let $e(Z)$ be the number of edges incident to Z. Our bipartite graph has dn edges, and each has an independent $|Z|/m \leq |Z|/n$ chance of landing in $|Z|$. Thus, $e(Z)$ is stochastically dominated by the binomial random variable $Bin(dn, |Z|/n)$, and so
$\mathbb{E}(e(Z)) \leq d|Z|$. By standard Chernoff bounds,

$$\mathbb{P}\left(e(Z) \geq 3d \log\left(\frac{n}{|Z|}\right)|Z|\right) \leq \left(\frac{e}{3\log(n/|Z|)}\right)^{3d|Z|\log(n/|Z|)} \leq e^{-3d|Z|\log(n/|Z|)} = \left(\frac{|Z|}{n}\right)^{3d|Z|}.$$

Then

$$\begin{aligned}
&\mathbb{P}\left(\exists\ Z \subseteq Y \text{ s.t. } |N(Z)| \geq 3d \log\left(\frac{n}{|Z|}\right)|Z|\right) \\
&\leq \sum_{i=1}^{n/12} \binom{m}{i}\left(\frac{i}{n}\right)^{3di} \leq \sum_{i=1}^{n/12}\left(\frac{2en}{i}\right)^{i}\left(\frac{i}{n}\right)^{3di} = \sum_{i=1}^{n/12}\left(2e\left(\frac{i}{n}\right)^{3d-1}\right)^{i} \\
&\leq \sum_{i=1}^{\log^2(n)} 2e\left(\frac{\log^2(n)}{n}\right)^{2} + \sum_{i=\log^2(n)}^{n/12}\left(2e\left(\frac{1}{12}\right)^{2}\right)^{\log^2(n)} = o(1/n).
\end{aligned}$$

□

Now, for $x \in X$ and $j \in \mathbb{N}$, let $W_{-j}(x)$ be defined to equal $\{w \in X : x \in \cup_{k=0}^{j} W_{+k}(w)\}$, that is, the set of elements that could reach x in at most j steps.

Lemma 3.2. *For any $d \geq 3$ and $c < c_d^*$, we have with high probability that for any matching $\mathcal{M}$ for any $j \in \mathbb{N}$ and any $S \subseteq X$ with $|S| \leq n/12$, we have $|W_{-j}(S)| \leq \left(3d \log\left(\frac{n}{|S|}\right)\right)^{j}|S|$.*

Proof. We will assume that the conclusion of Lemma 3.1 holds, as it does with high probability. We can then prove this lemma inductively as a corollary of Lemma 3.1.

We see that Lemma 3.2 is true for $j = 0$. Then note that $W_{-j}(S) = W_{-1}(W_{-(j-1)}(S)) = N(Z)$ where $Z \subseteq Y$ is the spots occupied by $W_{-(j-1)}(S)$, which thus has the same cardinality of $W_{-(j-1)}(S)$.

So using Lemma 3.1, we have

$$\begin{aligned}
|W_{-j}(S)| \leq 3d \log\left(\frac{n}{|W_{-(j-1)}(S)|}\right)|W_{-(j-1)}(S)| \leq 3d \log\left(\frac{n}{|S|}\right)|W_{-(j-1)}(S)| \\
\leq \left(3d \log\left(\frac{n}{|S|}\right)\right)^{j}|S|,
\end{aligned}$$

as desired.

Note that if we ever have $|W_{-(j-1)}(S)| \geq n/12$ (so Lemma 3.1 can't be applied), then we have $|W_{-j}(S)| \leq 3d \log\left(\frac{n}{|S|}\right) |W_{-(j-1)}(S)|$ anyway, as the right side of the equation is then more than n. □

Lemma 3.2 will be strong enough for our work in the next few sections, including the technical Section 5. After Section 5, in Section 7 we will prove Lemma 7.1, which is a more technical improvement on Lemma 3.2 needed for our final proof.

4 The Changing Matching

As we noted above, Fountoulakis, Panagiotou, and Steger proved the following lemma, which we will use as a black box:

Lemma 2.1 (Corollary 2.3 of [FPS13]). *Let $d \geq 3$ and assume $n = cm$ for $c < c_d^*$. Then with high probability, we have that for any matching $\mathcal{M}$ and any $\alpha = \Theta(1) > 0$, there exists $M = \Theta(1)$ such that for the unoccupied vertices U of Y, we have that at most αn of the vertices of X have BFS distance $> M$ to U.*

The above lemma comes as a corollary of their following theorem:

Lemma 4.1 (Theorem 2.2 of [FPS13]). *Let $d \geq 3$ and assume $n = cm$ for $c < c_d^*$. Then with high probability, there exists a $\delta > 0$ such that for every $R \subseteq Y$, we have $|\{x \in X : N(x) \subseteq R\}| < (1-\delta)|R|$.*

Note for comparison that $|\{x \in X : N(x) \subseteq R\}| \leq |R|$ for every $R \subseteq Y$ is exactly the requirement for a matching to exist. This is essentially saying that for $c < c_d^*$, we beat Hall's bound by a constant factor for all sets as $n \to \infty$.

In other words, you could consider the parameter of the graph $\xi = \max_{R \subseteq Y} \frac{|\{x \in X : N(X) \subseteq R\}|}{|R|}$. The definition of c_d^* tells us that for any $c > c_d^*$, we with high probability have $\xi > 1$, while for any $c < c_d^*$, we with high probability have $\xi \leq 1$. The theorem above says that for any $c < c_d^*$, there exists an $\epsilon' = \epsilon'(c)$ such that we with high probability have $\xi \leq 1 - \epsilon'$.

When we first start our random walk by inserting the n-th element, it is inserted into a random slot in Y, independent of any previous hashes or actions taken by the cuckoo hashing process when inserting the previous elements. This fact is critical to our proof. Interestingly, the paper of [FPS13] does not use this fact; their result would hold true even if the initial hash of the element were adversarially chosen:

Theorem 4.2 (Theorem 1.2; Lemma 2.7 of [FPS13]). *Assume that we have $d \geq 3$, $c < c_d^*$, and $n = cm$. Then with high probability over the random hash functions, we have that the expected insertion time for the random walk insertion process is $O(\log^{1+b_d}(n))$, where $b_d = \frac{d+\log(d-1)}{(d-1)\log(d-1)}$.*

It is useful for us to use as a black-box that we can have $O(poly \log n)$ insertion time even when starting from an arbitrary starting hash in the graph. In other words, at any point in the insertion process, conditioned on any prior events in the insertion process (and still assuming the "with high probability" facts about the underlying graph), the expected time from that time until the random walk finishes is $O(\log^{1+b_d}(n))$. $b_d \leq 3$ for all $d \geq 3$, so this is $O(\log^4(n))$.

This proof that $O(\log n)$ expected insertion time from a given step holds conditioned on any prior events also can be shown to follow from our work here: Lemma 6.1 will show that with high probability it is true that for any matching (even one that might have been changed over the

course of the walk) that our bad set B_i has that $B_i = \emptyset$ when $i = C'\sqrt{\log n}$ with a sufficiently high constant C'. It is more notationally convenient to explain away the changes to the matching now, so we can hereafter treat the matching as fixed.

As the random walk progresses, the matching changes from $\mathcal{M}$, as some elements are moved to different spots. Again picturing $\mathcal{M}$ as assigning directions to the edges of the bipartite graph, as we only change the direction of edges that we move along, we see that the only time that the change in the matching may affect our random walk might need to worry about the changing matching is if the walk cycles; that is, if an object $x \in X$ is reached twice by the random walk.

For this purpose, we will define a special set $\mathcal{C} \subseteq X$, which we can think of as the vertices near cycles significantly shorter than $\log(n)$. Formally, let $z = (10\log(n))^{0.9}$ and let $S_{Cyc} \subseteq X$ be the set of vertices who are on a cycle of length z or less. Then we define $\mathcal{C} = W_{-z}(S_{Cyc})$.

Lemma 4.3. *For any $d \geq 3$ and $c < c^*_d$, we have with high probability over the choice of random hashes that $|\mathcal{C}| < n^{0.3}$.*

Before proving this lemma, we will explain how it allows us to deal with changes to $\mathcal{M}$. If we start inside $\mathcal{C}$, we will simply use the $O(poly\log n)$ bound. Since the probability of starting in $\mathcal{C}$ is at most $n^{-0.7}$, this adds an $O(1)$ factor to our expected run time.

Similarly, we will show in Corollary 9.1 that, conditioned on staying outside of S_{Cyc}, the probability of taking more than z steps is $O(z^{-5}) \leq O((\log n)^{-4})$ as well. So, if the random walk starting outside of $\mathcal{C}$ reaches z steps in length, then we can again revert to the $O(\log^4 n)$ bound while only adding an $O(1)$ factor to the expected run time.

Therefore, this subsection shows that we do not need to worry about any changes to the matching from $\mathcal{M}$, as the cases that remain to be proven only include cases that do not involve any cycling in the random walk. So, for the rest of this paper, we can consider the cuckoo hashing insertion procedure to be a random walk on the fixed directed graph given by $\mathcal{M}$.

Proof of Lemma 4.3. Fix $\ell \in \mathbb{N}$ and consider the cycles of length 2ℓ in the bipartite graph. Each has the form $(x_1, y_1, x_2, y_2, \ldots, x_\ell, y_\ell)$ for some $x_1, \ldots, x_\ell \in X$ and $y_1, \ldots, y_\ell \in Y$, where x_i hashes to both y_i and y_{i-1} (with x_1 also hashing to y_ℓ). There are at most $n^\ell m^\ell$ ordered sets of vertices $(x_1, y_1, x_2, y_2, \ldots, x_\ell, y_\ell)$. The probability that all required hashes will be chosen is at most $\left(\frac{d(d-1)}{m^2}\right)^\ell \leq d^{2\ell} m^{-2\ell}$. Thus, the expected number of cycles of length 2ℓ in the bipartite graph is at most $n^\ell m^\ell d^{2\ell} m^{-2\ell} < d^{2\ell}$.

Then the number of cycles of length at most z is at most $\sum_{\ell=1}^{z/2} d^{2\ell} \leq d^{z+1} = o(d^{\log(n)/(100d)}) = o(n^{0.1})$. Markov's inequality gives that with high probability there are less than $n^{0.1}$ cycles of length at most z.

Each of these cycles has at most z vertices on it, so $|S_{Cyc}| < n^{0.1} z < n^{0.2}$ for sufficiently large n.

Then we apply Lemma 3.2 to say that

$$|\mathcal{C}| \leq \left(3d\log\left(\frac{n}{|S_{Cyc}|}\right)\right)^z |S_{Cyc}| < (3d\log(n))^z\, n^{0.2} < n^{0.3}.$$

□

5 Expansion from any vertex set

5.1 Lower bounds on $|W_{+j}(S)|$

The following lemma was proven by Fountoulakis, Panagiotou, and Steger:

Lemma 5.1 (Proposition 2.4 of [FPS13]). *Let $d \geq 3$ and $c < c_d^*$. For any $1 \leq s < |X|/d$, define*

$$p_s = \begin{cases} 0 & \text{if } s \leq \log\log(n) \\ \frac{\log_d((d-1)e^d)}{\log_d(|X|/(ds))} & \text{if } \log\log(n) \leq s \leq |X|/d \end{cases}$$

With high probability, we have that for all $S \subseteq X$ with $|S| < |X|/d$ that

$$|N(S)| \geq (d - 1 - p_{|S|})|S|.$$

Some facts to note here are that $p_s \geq 0$ and p_s is monotonically increasing with s. Also, for $s \leq |X|/(e^{1000d})$, which we will generally be able to assume by Lemma 2.1, we have $p_s < 0.01$.

Fountoulakis, Panagiotou, and Steger applied Lemma 5.1 iteratively, and the term in the numerator of Lemma 5.1 ended up becoming the exponent of the logarithm in their $O(poly \log n)$ run-time bound.

Because it is so critical for our paper, we provide a proof of Lemma 5.1 here, reproducing the proof of Fountoulakis, Panagiotou, and Steger [FPS13].

Proof. For a given $S \subseteq X$ and $T \subseteq Y$, we have that

$$\mathbb{P}(N(S) \subseteq T) = \left(\frac{|T|}{m}\right)^{d|S|}.$$

Fix an s such that $1 \leq s \leq n$. For there to be an $S \subseteq X$ with $|S| = s$ that fails the lemma, it must have $N(S) \subseteq T$ for a T with $|T| = (d - 1 - p_s)s$. So, for $s \geq \log\log(n)$, we have

$$\begin{aligned} \mathbb{P}(\exists\ S \subseteq X \text{ with } |S| = s \text{ failing the lemma}) &\leq \binom{n}{s}\binom{m}{(d-1-p_s)s}\left(\frac{(d-1-p_s)s}{m}\right)^{ds} \\ &\leq \left(\frac{ne}{s}\right)^s \left(\frac{em}{(d-1-p_s)s}\right)^{(d-1-p_s)s} \left(\frac{(d-1-p_s)s}{m}\right)^{ds} \\ &\leq \left(\frac{cs^{p_s}e^{(d-p_s)}(d-1-p_s)^{(1+p_s)}}{m^{p_s}}\right)^s \\ &\leq \left(\left(\frac{s(d-1)}{em}\right)^{p_s}(ce^d(d-1))\right)^s \\ &\leq c^s, \end{aligned}$$

where we recall that the load factor c satisfies $c < c_d^* < 1$. For $s \leq \log(m)/(d^2)$, we can note that for S to fail the lemma we must in fact have $|T| < (d-1)s$, so as $(d-1)s$ is an integer we need $|T| \leq (d-1)s - 1 = (d - 1 - 1/s)s$. So, applying the above process with $1/s$ in the place of p_s, we have for $s \leq \log(m)/(d^2)$ that

$$\begin{aligned} \mathbb{P}(\exists\ S \subseteq X \text{ with } |S| = s \text{ failing the lemma}) &\leq \binom{n}{s}\binom{m}{(d-1-1/s)s}\left(\frac{(d-1-1/s)s}{m}\right)^{ds} \\ &\leq \left(\left(\frac{s(d-1)}{em}\right)^{1/s}(ce^d(d-1))\right)^s \\ &\leq \left(\frac{\log(m)}{edm}\right)\left(ce^d(d-1)\right)^{\log(m)/(d^2)} \end{aligned}$$

$$\leq \left(\frac{\log(m)(d-1)}{em}\right) 2^{\log(m)}$$
$$\leq o(m^{-0.2})$$

Then summing over all s, the probability that there exists some $S \subseteq X$ that fails our lemma is at most

$$\sum_{s=1}^{\log(m)/(d^2)} o(m^{-0.2}) + \sum_{s=\log(m)/(d^2)}^{cm} c^s \leq o(m^{-0.19}) + O(m^{log(c)/(d^2)}).$$

So, with high probability there is no such S, as desired. □

For our purposes, we want to give a lower bound on $|W_{+j}(S)|$ using this lemma. Note that we have a natural upper bound of $|W_{+j}(S)| \leq (d-1)|W_{+(j-1)}(S)| \leq (d-1)^j|S|$.

Lemma 5.2. *Let $d \geq 3$, $c < c_d^*$, and p_s be defined as in Lemma 5.1. With high probability, for every matching $\mathcal{M}$ and any $S \subseteq X$ and $j \in \mathbb{N}$ with $|S| < (d-1)^{-j}n$, we have that $|W_{+j}(S)| \geq (d-1-p_{(d-1)^j|S|})^{j-1}(d-2-p_{(d-1)^j|S|})|S|$*

Proof. Note that $|W_{+\leq j}(S)| = |N(W_{+\leq j-1}(S))|$, as $W_{+\leq j}(S) \subseteq X$ is exactly the set of elements that fill the slots in Y that are neighbors of $W_{+\leq j-1}(S) \subseteq X$. Therefore, we can apply Lemma 5.1 to say that

$$|W_{+\leq j}(S)| = |N(W_{+\leq j-1}(S))| \geq (d-1-p_{|W_{+\leq j-1}(S)|})|W_{+\leq j-1}(S)|.$$

Applying this inductively gives

$$|W_{+\leq j}(S)| \geq |S| \prod_{k=0}^{j-1}(d-1-p_{(d-1)^{k+1}|S|}) \geq (d-1-p_{(d-1)^j|S|})^j|S|,$$

using that p_s is monotonically increasing with s. Then,

$$\begin{aligned}
|W_{+j}(S)| &\geq |W_{+\leq j}(S)| - |W_{+\leq j-1}(S)| \\
&\geq (d-1-p_{(d-1)^j|S|})|W_{+\leq j-1}(S)| - |W_{+\leq j-1}(S)| \\
&\geq (d-2-p_{(d-1)^j|S|})|W_{+\leq j-1}(S)| \qquad (1) \\
&\geq (d-2-p_{(d-1)^j|S|})(d-1-p_{(d-1)^{j-1}|S|})^{j-1}|S| \\
&\geq (d-2-p_{(d-1)^j|S|})(d-1-p_{(d-1)^j|S|})^{j-1}|S|
\end{aligned}$$

as desired. □

Lemma 5.2 essentially gives that $|W_{+j}(S)|$ is within a constant factor, $\frac{d-2}{d-1}$, of the upper bound of $|W_{+j}(S)| \leq (d-1)^j|S|$. Lemma 5.3 essentially shows that relatively little of this loss appears at higher j.

Lemma 5.3. *Let p_s be defined as in Lemma 5.1. For any $S \subseteq X$ and any $j \in \mathbb{N}$ with $|S| < (d-1)^{-j}n/e^{1000d}$, we have that*

$$|W_{+j}(S)| \geq (d-1)|W_{+(j-1)}(S)| - |S| - 2.1p_{(d-1)^j|S|}|W_{+(j-1)}(S)|.$$

Proof. As in the proof of Lemma 5.2, we start from

$$|W_{+\leq j}(S)| = |N(W_{+\leq j-1}(S))| \geq (d - 1 - p_{|W_{+\leq j-1}(S)|})|W_{+\leq j-1}(S)|.$$

In other words, this means that if you take any ordering of the $d|W_{+\leq j-1}(S)|$ hashes leaving $W_{+\leq j-1}(S)$, there are at most $(1 + p_{|W_{+\leq j-1}(S)|})|W_{+\leq j-1}(S)|$ hashes that hit a table slot already hit by another hash in $W_{+\leq j-1}(S)$, which we could call repeat hashes.

Considering ordering the hashes where the ones from S come first, then the ones from $W_{+1}(S)$, and so on until $W_{+(j-1)}(S)$. We see that for every $1 \leq k < j-1$, by the definition of $W_{+k}(S)$, every object in $W_{+k}(S)$ sends at least one of its d hashes into a slot occupied by an element of $W_{+(k-1)}(S)$, giving a repeat at every element in $W_{+k}(S)$. That shows that from $W_{+1}(S)$ to $W_{+(j-2)}(S)$ we have

$$\geq \bigcup_{k=1}^{j-2} |W_{+k}(S)| \geq |W_{+\leq j-1}(S)| - |W_{+(j-1)}(S)| - |S|$$

repeats.

Therefore, the number of the $d|W_{+(j-1)}(S)|$ hash functions from $W_{+(j-1)}(S)$ that can go to a slot already hashed to is at most the number of repeats remaining, which is at most

$$\begin{aligned}(1 + p_{|W_{+\leq j-1}(S)|})|W_{+\leq j-1}(S)| - (|W_{+\leq j-1}(S)| - |W_{+(j-1)}(S)| - |S|) \\ = |W_{+(j-1)}(S)| + |S| + p_{|W_{+\leq j-1}(S)|}|W_{+\leq j-1}(S)|.\end{aligned}$$

Each new slot that is hashed to gives a corresponding element of $W_{+j}(S)$, so

$$\begin{aligned}|W_{+j}(S)| &\geq d|W_{+(j-1)}(S)| - (|W_{+(j-1)}(S)| + |S| + p_{|W_{+\leq j-1}(S)|}|W_{+\leq j-1}(S)|) \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - p_{|W_{+\leq j-1}(S)|}|W_{+\leq j-1}(S)| \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - p_{(d-1)^j|S|}|W_{+\leq j-1}(S)| \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - p_{(d-1)^j|S|}(|W_{+(j-1)}(S)| + |W_{+\leq j-2}(S)|) \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - p_{(d-1)^j|S|}\left(|W_{+(j-1)}(S)| + \frac{|W_{+(j-1)}(S)|}{d - 2 - p_{(d-1)^{j-1}|S|}}\right), \text{ using (1)} \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - p_{(d-1)^j|S|}\left(|W_{+(j-1)}(S)| + \frac{|W_{+(j-1)}(S)|}{d - 2.01}\right), \\ &\qquad \text{using } |W_{+(j-1)}(S)| \leq (d-1)^{j-1}|S| < |X|/(e^{1000d}) \\ &\geq (d-1)|W_{+(j-1)}(S)| - |S| - 2.1p_{(d-1)^j|S|}|W_{+(j-1)}(S)|\end{aligned}$$

as desired. □

5.2 Avoiding small sets through expansion

Now that we have proven lower bounds on $|W_{+j}(S)|$ for any set, we will now use these bounds to go in a different direction and show that for a set S, relatively few elements will have many paths to S. In other words, we create sets $B_j(S)$, which consist of elements $x \in X$ from which we have a high likelihood of being in S after j steps.

Unlike bounding $|W_{-j}(S)|$, which includes all objects that have at least one path of length j to S, we will only include objects which have at least some fraction of their paths of length j

reaching S. Correspondingly, while $|W_{-j}(S)|$ must grow with j, when we set the parameters here, we will see that $|B_j(S)|$ will actually shrink with j; if j is large, there are very few objects that, as a start point, have a high likelihood of being in S after a j steps. You can imagine this as saying something about the mixing of our random walk, as it shows the random walk is unlikely to concentrate on a small set after a while.

A sketch of the basic argument goes like this: if you take a set Q, the lemmas in the previous section show that $|W_{+j}(Q)|$ is large. In particular, this means that there are many distinct endpoints for a walk of length j starting in Q. Only $|S|$ of the distinct endpoints are in S, so if $|W_{+j}(Q)| \gg |S|$, then it is unlikely for a walk of length j starting in Q to end up in S. For this to work, it is also necessary to know how many of the walks could concentrate on the same endpoints.

The next four lemmas will iteratively bootstrap off each other to get better bounds on the likelihood of ending in S, with new definitions for $B_j^{(1)}(S)$, $B_j^{(2)}(S)$, $B_j^{(3)}(S)$, and $B_j^{(4)}(S)$.

Formally now: for any set $S \subseteq X$ and a given matching $\mathcal{M}$, define $B_i^{(1)}(S) \subseteq X$ as follows. An object x is in $B_i^{(1)}(S)$ if and only if at least $1/4$ of the $(d-1)^{\log^2(i)}$ paths of length $\log^2(i)$ from x end in an object $z \in W_{+\log^2(i)}(x)$ such that at least $\frac{150}{i}$ proportion of the $(d-1)^{i-\log^2(i)}$ paths of length $i - \log^2(i)$ starting at z end in S.

In other words, we define

$$S_1' = \{z \in X :\geq 150(d-1)^{i-\log^2(i)}/i \text{ paths of length } i - \log^2(i) \text{ from } x \text{ end in } S\}$$

and then

$$B_i^{(1)}(S) = \{x \in X :\geq (d-1)^{\log^2(i)}/4 \text{ paths of length } \log^2(i) \text{ from } x \text{ end in } S_1'\}.$$

Lemma 5.4. *For any $d \geq 3$ and $c < c_d^*$, we have the following with high probability that for any matching $\mathcal{M}$: for any $S \subseteq X$ and $i \in \mathbb{N}$ with $i \geq C_1$ for some $C_1 = \Theta(1)$ and $|S| \leq (d-1)^{-i^2/5}n$, we have that $|B_i^{(1)}(S)| < (d-1)^{-.9i}|S|$.*

Proof. Let $Q \subseteq X$ such that $|Q| = (d-1)^{-.9i}|S|$. We will prove that there must exist some $x \in Q$ such that $x \notin B_i^{(1)}(S)$, therefore proving that no such Q can equal $B_i^{(1)}(S)$ and thus $|B_i^{(1)}(S)| < (d-1)^{-.9i}|S|$.

In fact, we will show this by showing that, starting from a uniformly random point in $x \in Q$, there is at least probability $\geq \frac{1}{4}$ that after $\log^2(i)$ steps, we are at a point z such that more than $1 - \frac{150}{i}$ proportion of the $(d-1)^{i-\log^2(i)}$ paths of length $i - \log^2(i)$ do not end in S.

First, we note that

$$\begin{aligned}
|W_{+\log^2(i)}(Q)| &\geq (d - 1 - p_{(d-1)^{\log^2(i)}|Q|})^{\log^2(i)-1}(d - 2 - p_{(d-1)^{\log(i)}|Q|})|Q| && \text{by Lemma 5.2} \\
&\geq \left(d - 1 - \frac{\log_d((d-1)e^d)}{\log_d((d-1)^{i^2/5}/d)}\right)^{\log^2(i)-1}\left(d - 2 - \frac{\log_d((d-1)e^d)}{\log_d((d-1)^{i^2/5}/d)}\right)|Q|, \\
&\qquad \text{as } \log^2(i)|Q| < |S| \leq (d-1)^{-i^2/5}n \\
&\geq \left(d - 1 - \frac{15(d-1)}{i^2}\right)^{\log^2(i)-1}(d - 2 - 0.1)\,|Q|, \\
&\qquad \text{using } 15(d-1) \geq \frac{5\log_d((d-1)e^d)}{\log_d(d-1)} \text{ for all } d \geq 3 \\
&\geq (d-1)^{\log^2(i)}\left(\frac{d-2.1}{d-1}\right)\left(1 - \frac{15(\log^2(i)-1)}{i^2}\right)|Q|
\end{aligned}$$

$$\geq (d-1)^{\log^2(i)}|Q|/3.$$

Then, from here we note that for every $\log^2(i) \leq j \leq i$, we have that

$$\begin{aligned}
|W_{+j}(Q)| &\geq (d-1)|W_{+(j-1)}(Q)| - |Q| - 2.1p_{(d-1)^j|Q|}|W_{+(j-1)}(Q)| && \text{By Lemma 5.3}\\
&\geq (d-1)|W_{+(j-1)}(Q)| - \frac{3|W_{+\log^2(i)}(Q)|}{(d-1)^{\log^2(i)}} - 2.1p_{(d-1)^{i-i^2/5}n}|W_{+(j-1)}(Q)|\\
&\geq (d-1)|W_{+(j-1)}(Q)| - \frac{|W_{+(j-1)}(Q)|}{i^2} - 2.1\frac{\log_d((d-1)e^d)}{(i^2/5-i)\log_d(d-1)-1}|W_{+(j-1)}(Q)|\\
&\geq \left(d-1-\frac{2.1(15(d-1))}{i^2}\right)|W_{+(j-1)}(Q)|,
\end{aligned}$$

again using that $15(d-1) > \frac{5\log_d((d-1)e^d)}{\log_d(d-1)}$ for all $d \geq 3$.

Applying this iteratively, we get that

$$\begin{aligned}
|W_{+i}(Q)| &\geq \left(d-1-\frac{32(d-1)}{i^2}\right)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|\\
&\geq (d-1)^{i-\log^2(i)}\left(1-\frac{32}{i^2}\right)^i|W_{+\log^2(i)}(Q)| \geq (d-1)^{i-\log^2(i)}\left(1-\frac{32}{i}\right)|W_{+\log^2(i)}(Q)|
\end{aligned}$$

This tells us that for any ordering of the $(d-1)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|$ walks of length $i-\log^2(i)$ leaving Q, at most $32/i$ proportion of them end at an object that was also the endpoint of a previous path, that is, there are at most $(d-1)^{i-\log^2(i)}\left(\frac{32}{i}\right)|W_{+\log^2(i)}(Q)|$ repeats. Now,

$$|S| = (d-1)^{0.9i}|Q| \leq (d-1)^{-.05i}(d-1)^{i-\log^2(i)}|Q|.$$

This implies that of the $(d-1)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|$ paths of length $i-\log^2(i)$ from $W_{+\log^2(i)}(Q)$, at most a $\left(\frac{32}{i}+(d-1)^{-0.5i}\right) \leq \frac{33}{i}$ proportion end in S, as if we order these paths we can have at most $|S|$ ones hit S for the first time, plus the repeats. Then, the Markov inequality tells us that less than $\frac{1}{4}$ of the elements in $W_{+\log^2(i)}(Q)$ have at least a $\frac{150}{i}$ proportion of their paths ending in S, and thus (recalling the definition of S_1' before the start of Lemma 5.4),

$$|S_1' \cap W_{+\log^2(i)}(Q)| < |W_{+\log^2(i)}(Q)|/4$$

so then

$$|W_{+\log^2(i)}(Q) \cap (X \setminus S_1')| > (3/4)|W_{+\log^2(i)}(Q)| \geq (d-1)^{\log^2(i)}|Q|/4.$$

This finishes the proof, as it then must be true that more than $1/4$ of the $(d-1)^{\log^2(i)}|Q|$ paths of length $\log^2(i)$ leaving Q must not end up in S_1', meaning that Q cannot be $B_i^{(1)}(S)$. □

Now, we proceed to the second of four lemmas, where we can replace the two step "constant probability of having $\frac{150}{i}$ probability of being in S" with a pure $\frac{200}{i}$ probability of being in S.

For any set $S \subseteq X$, define $B_i^{(2)}(S) \subseteq X$ under a given matching $\mathcal{M}$ as follows. An object x is in $B_i^{(2)}(S)$ if and only if at least $\frac{200}{i}$ of the $(d-1)^i$ paths of length i starting at x end in S.

In other words, you could define

$$B_i^{(2)}(S) = \{x \in X :\geq 200(d-1)^i/i \text{ paths of length } i \text{ from } x \text{ end in } S\}.$$

Lemma 5.5. *For any $d \geq 3$ and $c < c_d^*$, we have the following with high probability that for any matching $\mathcal{M}$: for any $S \subseteq X$ and $i \in \mathbb{N}$ with $i \geq C_2$ for some $C_2 = \Theta(1)$ and $(d-1)^{-i^{10}}n < |S| < (d-1)^{-i^2/4.5}n$, we have that $|B_i^{(2)}(S)| < (d-1)^{-.8i}|S|$.*

Proof. We claim that $B_i^{(2)}(S) \subseteq W_{-\log^4(i)}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$. We first show that this suffices to complete the proof, as we show $|W_{-\log^4(i)}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))| < (d-1)^{-.8i}|S|$. First note that

$$|W_{-\log^4(i)}(S)| \leq \left(3d\log\left(\frac{n}{|S|}\right)\right)^{\log^4(i)}|S| \leq \left(3di^{10}\right)^{\log^4(i)}(d-1)^{-i^2/4.5}n \leq (d-1)^{-i^2/5}n$$

for $i \geq C_2$, by Lemma 3.2. So

$$\begin{aligned}
|B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))| &\leq (d-1)^{-0.9(i-\log^4(i))}|W_{-\log^4(i)}(S)| && \text{by Lemma 5.4}\\
&\leq (d-1)^{-0.85i}\left(3d\log\left(\frac{n}{|S|}\right)\right)^{\log^4(i)}|S| && \text{by Lemma 3.2}\\
&\leq (d-1)^{-0.85i}\left(3di^{10}\log(d-1)\right)^{\log^4(i)}|S| && \text{as } (d-1)^{-i^{10}}n < |S|\\
&\leq (d-1)^{-0.82i}|S|
\end{aligned}$$

and thus

$$\begin{aligned}
&|W_{-\log(i)^4}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))|\\
&\leq \left(3d\log\left(\frac{n}{|B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))|}\right)\right)^{\log^4(i)}|B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))|, \quad \text{by Lemma 3.2}\\
&\leq \left(3d\log\left(\frac{n}{(d-1)^{-0.82i}|S|}\right)\right)^{\log^4(i)}(d-1)^{-0.82i}|S|\\
&\leq \left(3d(i^{10}+0.82i)\log(d-1)\right)^{\log^4(i)}(d-1)^{-0.82i}|S|\\
&\leq (d-1)^{-0.8i}|S|
\end{aligned}$$

as desired.

Now, let $x \notin W_{-\log^4(i)}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$, and we will prove that $x \notin B_i^{(2)}(S)$.

Let x_2 be the position that we reach after $\log^2(i-\log^4(i))$ steps. Because $x \notin B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))$, there is at least a $1/4$ chance that x has the property that there is at least $1-150/i$ chance that we will be outside of $W_{-\log^4(i)}(S)$ in a further $(i-\log^4(i))-\log^2(i-\log^4(i))$ steps.

Also, because $x \notin W_{-\log^4(i)}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$, we know for sure that $x_2 \notin W_{-(\log^4(i)-\log^2(i-\log^4(i)))}(B^{(1)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$. Then if the $1/4$ probability event does not occur, we still have that after a further $\log^2(i-\log^4(i))$ steps from x_2, there is again at least a $1/4$ chance that we are at a point x_3 such that with probability $1-150/i$ we will be outside of $W_{-\log^4(i)}(S)$ in a further $(i-\log^4(i))-\log^2(i-\log^4(i))$ steps from x_3.

In this way, we see that we can iterate, and then for any $k \in \mathbb{N}$ such that $k \log^2(i - \log^4(i)) < \log^4(i)$, it is true that after $k \log^2(i - \log^4(i))$ steps, we have probability at least $1 - \left(\frac{3}{4}\right)^k$ to reach a point x' such that at least a $1 - \frac{150}{i}$ fraction of paths from x' are outside $W_{-\log^4(i)}(S)$ in a further $(i - \log^4(i)) - \log^2(i - \log^4(i))$ steps from x'.

We plug in $k = \log^2(i)$. Then with probability $\geq 1 - \left(\frac{3}{4}\right)^{\log^2(i)} \geq 1 - \frac{1}{i}$, in the first $\log^4(i)$ steps we have that we reach a point x' such that at least a $1 - \frac{150}{i}$ fraction of paths from x' are outside $W_{-\log^4(i)}(S)$ in a further $(i - \log^4(i)) - \log^2(i - \log^4(i))$ steps from x'. This shows that if we do reach such an x', then conditioned on reaching that x' we have at least a $1 - \frac{150}{i}$ of being outside of S after exactly i steps from our initial x (as we reach x' after a number of steps between $\log^2(i - \log^4(i))$ and $\log^4(i)$; and then a further $(i - \log^4(i)) - \log^2(i - \log^4(i))$ steps later we are likely to be outside of $W_{-\log^4(i)}(S)$, meaning outside of S after $i - k \log^2(i - \log^4(i))$ steps from x' for every $1 \leq k \leq \log^2(i)$ as desired).

Since the probability of reaching such an x' is at least $1 - \frac{1}{i}$, we have probability at least $\left(1 - \frac{150}{i}\right)\left(1 - \frac{1}{i}\right) \geq 1 - \frac{200}{i}$ of being outside of S after exactly i steps from our initial x. Therefore, $x \notin B_i^{(2)}(S)$, as desired. □

We defined $B_i^{(2)}(S)$ to have $200/i$ probability of hitting S. Now, we want to strengthen this result by reducing this probability to $2/i^{1.5}$. We will now bootstrap Lemma 5.5 to a stronger failure probability, but first using the same definition as in Lemma 5.4 except with the $150/i$ probability replaced with $i^{-1.5}$.

For any set $S \subseteq X$, define $B_i^{(3)}(S) \subseteq X$ under a given matching $\mathcal{M}$ as follows. An object x is in $B_i^{(3)}(S)$ if and only if at least $1/4$ of the $(d-1)^{\log^2(i)}$ paths of length $\log^2(i)$ from x end in an object $z \in W_{+\log^2(i)}(x)$ such that at least $i^{-1.5}$ proportion of the $(d-1)^{i-\log^2(i)}$ paths of length $i - \log^2(i)$ starting at z end in S.

In other words, you could define

$$S_3' = \{z \in X :\geq (d-1)^{i-\log^2(i)}/i^{1.5} \text{ paths of length } i - \log^2(i) \text{ from } x \text{ end in } S\}$$

and then

$$B_i^{(3)}(S) = \{x \in X :\geq (d-1)^{\log^2(i)}/4 \text{ paths of length } \log^2(i) \text{ from } x \text{ end in } S_3'\}.$$

Lemma 5.6. *For any $d \geq 3$ and $c < c_d^*$, we have the following with high probability that for any matching $\mathcal{M}$: for any $S \subseteq X$ and $i \in \mathbb{N}$ with $i \geq C_3$ for some $C_3 = \Theta(1)$ and $(d-1)^{-i^3} n < |S| < (d-1)^{-i^2/4.5} n$, we have that $|B_i^{(3)}(S)| < (d-1)^{-.7i}|S|$.*

Proof. This proof will follow much of the same structure of Lemma 5.4, and will also use the result of Lemma 5.5.

Let $Q \subseteq X$ such that $|Q| = (d-1)^{-.7i}|S|$. As in Lemma 5.4, we will prove that, starting from a uniformly random point $x \in Q$, there is at least probability $\geq \frac{1}{4}$ that after $\log^2(i)$ steps, we are at a point z such that more than a $1 - i^{-1.5}$ proportion of the $(d-1)^{i-\log^2(i)}$ paths of length $i - \log^2(i)$ do not end in S. This shows that there must be some $x \in Q$ such that $x \notin B_i^{(3)}(S)$, proving that $|B_i^{(3)}(S)| < (d-1)^{-.7i}|S|$.

In the same way as in the proof of Lemma 5.4, we see that

$$|W_{+\log^2(i)}(Q)| \geq (d-1)^{\log^2(i)}|Q|/3$$

and

$$|W_{+j}(Q)| \geq \left(d - 1 - \frac{32(d-1)}{i^2}\right)|W_{+(j-1)}(Q)| \tag{2}$$

for every $\log^2(i) \leq j \leq i$. Applying these iteratively, we get that

$$\begin{aligned}|W_{+j}(Q)| &\geq (d-1)^{j-\log^2(i)}\left(1 - \frac{32}{i^2}\right)|W_{+\log^2(i)}(Q)| \geq (d-1)^j\left(1 - \frac{32}{i^2}\right)|Q|/3 \\ &\geq (d-1)^j|Q|/4 \geq (d-1)^{j-.7i}|S|/4\end{aligned}$$

for every $\log^2(i) \leq j \leq i$. We also have that, for every $i^{0.3} \leq k \leq i$,

$$\begin{aligned}|W_{-1}(B_k^{(2)}(S))| &\leq 3d\log\left(\frac{n}{|B_k^{(2)}(S)|}\right)|B_k^{(2)}(S)| \qquad \text{by Lemma 3.2} \\ &\leq 3d\log\left(\frac{n}{(d-1)^{-.8k}|S|}\right)(d-1)^{-.8k}|S| \\ &\qquad \text{by Lemma 5.5, as } k \geq C_2, \text{ and } |S| > (d-1)^{-i^3}n \geq (d-1)^{-k^{10}}n, \\ &\qquad\qquad \text{and } |S| < (d-1)^{-i^2/4.5}n \leq (d-1)^{-k^2/4.5}n \\ &\leq 3d\log\left(\frac{n}{(d-1)^{-.8i}(d-1)^{-i^3}n}\right)(d-1)^{-.8k}|S| \\ &\leq 3d(\log(d-1))(i^3 + .8i)(d-1)^{-.8k}|S|\end{aligned}$$

Comparing these two bounds, we get that for all $\log^2(i) \leq j \leq i - i^{0.3}$,

$$\begin{aligned}|W_{-1}(B_{i-j}^{(2)}(S))| &\leq 3d(\log(d-1))(i^3 + .8i)(d-1)^{-.8(i-j)}|S| \\ &\leq (d-1)^{.8j-.8i+.05i}|S| \qquad \text{for all } i \geq C_3 \\ &\leq (d-1)^{(j-1)-.7i}|S|/(4i^3) \qquad \text{for all } i \geq C_3 \\ &\leq |W_{+(j-1)}(Q)|/(i^3)\end{aligned}$$

Now, consider the $(d-1)|W_{+(j-1)}(Q)|$ hashes leaving $W_{+(j-1)}(Q)$. If $j \leq i - i^{0.3}$, at most an i^{-3} proportion of them end inside $B_{i-j}^{(2)}(S)$.

Additionally, by (2), at most an $\frac{32}{i^2}$ proportion of them under any ordering land on a slot already hashed to by a previous hash. If we use R to denote the set of hashes that land on a slot that another hash from $W_{+(j-1)}(Q)$ also lands on, we have that R is at most a $\frac{64}{i^2}$ proportion of the total hashes. Counting separately the i^{-3} proportion ending up inside $B_{i-j}^{(2)}(S)$, the at most $\frac{64}{i^2}$ proportion outside of $B_{i-j}^{(2)}(S)$ corresponding to R thus has at least probability $1 - \frac{200}{i-j}$ of landing outside of S after $i-j$ more steps (i total steps).

Of the remaining hashes that go to unique locations at each step, at most $|S|$ end up in S.

Therefore, of the $(d-1)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|$ paths of length $i - \log^2(i)$ from $W_{+\log^2(i)}(Q)$, every path that lands in S falls into one of the following categories:

- At the first j where its position is the same as the position of another one of the paths, it falls into $B_{i-j}^{(2)}(S)$
  - At most an i^{-3} proportion of paths for a given $\log^2(i) \leq j \leq i - i^{0.3}$

  - At most an $64i^{-2}$ proportion of paths for a given $i - i^{0.3} \leq j \leq i$

- At the first j where its position is the same as the position of another one of the paths, it does not fall into $B^{(2)}_{i-j}$
  - At most an $\frac{64}{i^2}$ proportion of paths for a given j
  - At most a $\frac{200}{i-j}$ proportion of the paths that fall into this category for this j end up in S, if $i - j \geq i^{0.3}$

- Has no j such that its position at step j is the same as the position of another one of the paths
  - At most $|S|$ total paths

Putting this together, the proportion that land in S out of the of the $(d-1)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|$ paths of length $i - \log^2(i)$ from $W_{+\log^2(i)}(Q)$ is at most

$$\begin{aligned}
&\frac{i-\log^2(i)}{i^3} + \frac{64}{i^2}\sum_{j=\log^2(i)}^{i-i^{0.3}} \frac{200}{i-j} + \frac{64}{i^2}\sum_{j=i-i^{0.3}}^{i} 1 + \frac{|S|}{(d-1)^{i-\log^2(i)}|W_{+\log^2(i)}(Q)|} \\
&\qquad \leq i^{-2} + \frac{12800\log(i)}{i^2} + \frac{64i^{0.3}}{i^2} + \frac{|S|}{(d-1)^i|Q|/3} \\
&\qquad < i^{-1.5}/4. \qquad\qquad \text{for } i \geq C_3
\end{aligned}$$

From here, we finish the proof as in Lemma 5.4: The Markov inequality tells us that less than $\frac{1}{4}$ of the elements in $W_{+\log^2(i)}(Q)$ have at least a $i^{-1.5}$ proportion of their paths ending in S, and thus (recalling the definition of S'_3 before the start of Lemma 5.6),

$$|S'_3 \cap W_{+\log^2(i)}(Q)| < |W_{+\log^2(i)}(Q)|/4$$

so then

$$|W_{+\log^2(i)}(Q) \cap (X \setminus S')| > (3/4)|W_{+\log^2(i)}(Q)| \geq (d-1)^{\log^2(i)}|Q|/4.$$

This finishes the proof, as it then must be true that more than $1/4$ of the $(d-1)^{\log^2(i)}|Q|$ paths of length $\log^2(i)$ leaving Q must not end up in S'_3, meaning that Q cannot be $B^{(3)}_i(S)$. □

Finally, we complete the analogy with $B^{(4)}_i(S)$, which will be defined for $B^{(3)}_i(S)$ in the way that $B^{(2)}_i(S)$ was for $B^{(1)}_i(S)$.

For any set $S \subseteq X$, define $B^{(4)}_i(S) \subseteq X$ under a given matching $\mathcal{M}$ as follows. An object x is in $B^{(4)}_i(S)$ if and only if at least $2i^{-1.5}$ of the $(d-1)^i$ paths of length i starting at x end in S.

In other words, you could define

$$B^{(4)}_i(S) = \{x \in X :\geq 2(d-1)^i i^{-1.5} \text{ paths of length } i \text{ from } x \text{ end in } S\}.$$

Lemma 5.7. *For any $d \geq 3$ and $c < c^*_d$, we have the following with high probability that for any matching $\mathcal{M}$: for any $S \subseteq X$ and $i \in \mathbb{N}$ with $i \geq C_4$ for some $C_4 = \Theta(1)$ and $(d-1)^{-i^3}n < |S| < (d-1)^{-i^2/4}n$, we have that $|B^{(4)}_i(S)| < (d-1)^{-.6i}|S|$.*

Proof. This proof follows in exactly the same way as the proof of Lemma 5.5. We will still present the same proof here for completeness.

We claim that $B_i^{(4)}(S) \subseteq W_{-\log^4(i)}(B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$. Then

$$|W_{-\log^4(i)}(S)| \leq \left(3d\log\left(\frac{n}{|S|}\right)\right)^{\log^4(i)}|S| \leq \left(3di^3\right)^{\log^4(i)}(d-1)^{-i^2/4}n \leq (d-1)^{-i^2/4.5}n$$

for $i \geq C_4$, by Lemma 3.2. So

$$\begin{aligned}
|B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))| &\leq (d-1)^{-0.7(i-\log^4(i))}|W_{-\log^4(i)}(S)| && \text{by Lemma 5.6}\\
&\leq (d-1)^{-0.65i}\left(3d\log\left(\frac{n}{|S|}\right)\right)^{\log^4(i)}|S| && \text{by Lemma 3.2}\\
&\leq (d-1)^{-0.65i}\left(3di^3\log(d-1)\right)^{\log^4(i)}|S| && \text{as } (d-1)^{-i^3}n < |S|\\
&\leq (d-1)^{-0.62i}|S|
\end{aligned}$$

and thus

$$\begin{aligned}
&|W_{-\log(i)^4}(B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))|\\
&\leq \left(3d\log\left(\frac{n}{|B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))|}\right)\right)^{\log^4(i)}|B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))|, \quad \text{by Lemma 3.2}\\
&\leq \left(3d\log\left(\frac{n}{(d-1)^{-0.62i}|S|}\right)\right)^{\log^4(i)}(d-1)^{-0.62i}|S|\\
&\leq \left(3d(i^{10}+0.62i)\log(d-1)\right)^{\log^4(i)}(d-1)^{-0.62i}|S|\\
&\leq (d-1)^{-0.6i}|S|
\end{aligned}$$

as desired.

Now, let $x \notin W_{-\log^4(i)}(B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$, and we will prove that $x \notin B_i^{(4)}(S)$.

Let x_2 be the position that we reach after $\log^2(i-\log^4(i))$ steps. Because $x \notin B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S))$, there is at least a $1/4$ chance that x_2 has the property that there is at least $1-i^{-1.5}$ chance that we will be outside of $W_{-\log^4(i)}(S)$ in a further $(i-\log^4(i))-\log^2(i-\log^4(i))$ steps.

Also, because $x \notin W_{-\log^4(i)}(B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$, we know for sure that $x_2 \notin W_{-(\log^4(i)-\log^2(i-\log^4(i)))}(B^{(3)}_{i-\log^4(i)}(W_{-\log^4(i)}(S)))$. Then if the $1/4$ chance does not occur, we still have that after a further $\log^2(i-\log^4(i))$ steps from x_2, there is again at least a $1/4$ chance that we are at a point x_3 such that with probability $1-i^{-1.5}$ we will be outside of $W_{-\log^4(i)}(S)$ in a further $(i-\log^4(i))-\log^2(i-\log^4(i))$ steps from x_3.

In this way, we see that we can iterate, and then for any $k \in \mathbb{N}$ such that $k\log^2(i-\log^4(i)) < \log^4(i)$, it is true that after $k\log^2(i-\log^4(i))$ steps, we have probability at least $1-\left(\frac{3}{4}\right)^k$ to reach a point x' such that at least a $1-i^{-1.5}$ fraction of paths from x' are outside $W_{-\log^4(i)}(S)$ in a further $(i-\log^4(i))-\log^2(i-\log^4(i))$ steps from x'.

We plug in $k = \log^2(i)$. Then with probability $\geq 1 - \left(\frac{3}{4}\right)^{\log^2(i)} \geq 1 - i^{-1.5}$, in the first $\log^4(i)$ steps we have that we reach a point x' such that at least a $1 - i^{-1.5}$ fraction of paths from x' are outside $W_{-\log^4(i)}(S)$ in a further $(i - \log^4(i)) - \log^2(i - \log^4(i))$ steps from x'. This shows that if we do reach such an x', then conditioned on reaching that x' we have at least a $1 - i^{1.5}$ of being outside of S after exactly i steps from our initial x (as we reach x' after a number of steps between $\log^2(i - \log^4(i))$ and $\log^4(i)$; and then a further $(i - \log^4(i)) - \log^2(i - \log^4(i))$ steps later we are likely to be outside of $W_{-\log^4(i)}(S)$, meaning outside of S after $i - k\log^2(i - \log^4(i))$ steps from x' for every $1 \leq k \leq \log^2(i)$ as desired).

Since the probability of reaching such an x' is at least $1 - i^{-1.5}$, we have probability at least $\left(1 - i^{-1.5}\right)\left(1 - i^{-1.5}\right) \geq 1 - 2i^{-1.5}$ of being outside of S after exactly i steps from our initial x. Therefore, $x \notin B_i^{(4)}(S)$, as desired. □

Note that if we adjust the constants, we could repeat this process an arbitrary constant number of times, define a B_i with i^{-c} probability of hitting S for any $c \in \mathbb{R}$, and still show this is smaller than S by an exponential factor in i. But for our overall proof, the $2i^{-1.5}$ probability we have now obtained suffices.

6 Chaining together the B_i sets

Take α sufficiently small; in fact, what we need is that

$$\alpha < (d-1)^{-(C_4)^2/4}$$

using the C_4 from Lemma 5.7. Let G be the corresponding set given by Lemma 2.1 under the starting matching $\mathcal{M}$, where G consists of all elements of BFS distance at most M for some appropriate M, and let $B_{C_4} = X \setminus G$.

For all $i > C_4$, recursively define $B_i = B_i^{(4)}(B_{i-1})$.

Lemma 6.1. *For any $d \geq 3$ and $c < c_d^*$, we have with high probability that for any matching $\mathcal{M}$, $|B_i| \leq (d-1)^{-i^2/4}n$ for all $i \in \mathbb{N}$.*

Note that, in particular, this implies that there is a $C' = \Theta(1)$ such that $B_i = \emptyset$ for all $i \geq C'\sqrt{\log n}$.

Proof. We will prove this by induction on i, noting that it is true for $i = C_4$.

If $|B_{i-1}| < (d-1)^{-i^3}n$, then we note that $B_i^{(4)}(B_{i-1}) \subseteq W_{-i}(B_{i-1})$, so

$$\begin{aligned}
|B_i| &\leq |W_{-i}(B_{i-1})| \leq 3d\log\left(\frac{n}{|B_{i-1}|}\right)|B_{i-1}| && \text{by Lemma 3.2} \\
&\leq 3d\log\left(\frac{n}{(d-1)^{-i^3}n}\right)(d-1)^{-i^3}n \\
&\leq 3d(i^3\log(d-1))(d-1)^{-i^3}n \\
&\leq d^{-i^2/4}n
\end{aligned}$$

as desired. Otherwise, we have that

$$(d-1)^{-i^3}n \leq |B_{i-1}| \leq (d-1)^{-(i-1)^2/4}n,$$

and thus we can apply Lemma 5.7 to say that

$$\begin{aligned}|B_i| &\le (d-1)^{-.6i}|B_{i-1}| \qquad \text{by Lemma 5.7}\\ &\le (d-1)^{-.6i}(d-1)^{-(i-1)^2/4}n\\ &\le (d-1)^{-.6i-i^2/4+i/2-1/4}n\\ &\le (d-1)^{-i^2/4}n\end{aligned}$$

as desired. □

Lemma 6.2. *Conditioned on starting at any vertex outside of B_i, the random walk has probability ≥ 0.99 of being in G (or having finished) in exactly $i(i+1)/2 - C_4(C_4+1)/2$ steps.*

Proof. By the definition of B_j, if we are at a vertex outside of B_j, then we have probability $\ge 1 - \frac{2}{j^{1.5}}$ of being outside of B_{j-1} after j steps. Iterating this, we see that the probability of being in G or finished (that is, outside of B_{C_4}) after $\sum_{j=C_4}^{i} j = i(i+1)/2 - C_4(C_4+1)/2$ steps is

$$\ge \prod_{j=C_4}^{i}\left(1-\frac{2}{j^{1.5}}\right) \ge 1 - \sum_{j=C_4}^{i}\frac{2}{j^{1.5}} \ge 1 - \sum_{j=C_4}^{\infty}\frac{2}{j^{1.5}} \ge 0.99$$

as desired (using $C_4 \ge 5000$). □

Lemma 6.3. *For any $i \ge C_4$, conditioned on starting at any vertex outside of $W_{-10(M+1)(d-1)^M}(B_i)$, the random walk has probability ≥ 0.97 of terminating (finishing in U) within $10(M+1)(d-1)^M + (i(i+1)/2 - C_4(C_4+1)/2)$ steps.*

Proof. For any random walk, denote by x_t the location of the random walk after t steps, with x_0 being the initial hash location. Recalling the constant M and set G from Lemma 2.1, define an M-separated sequence in G to be a list of steps $t_1, \dots, t_q$ such that $x_{t_1}, \dots, x_{t_q} \in G$ and $t_{r+1} - t_r > M$ for every $1 \le r < q$. That is, an M-separated sequence in G is a list of times that the random walk is in G where each time is at least M steps after the previous.

Given a random walk that goes until finishing at an empty slot, let ζ be the maximum length of an M-separated sequence in G. Then we have

$$\mathbb{P}(\zeta \ge s) \le (1-(d-1)^{-M})^s,$$

as for every time t where $x_t \in G$, we have that the random walk will be finished in at most M more steps with probability at least $(d-1)^{-M}$.

Furthermore, note that this inequality $\mathbb{P}(\zeta \ge s) \le (1-(d-1)^{-M})^s$ still holds conditioned on any given starting location for the random walk.

Let $\mathcal{G}_1$ denote the event that $\zeta \ge 5(d-1)^M$. Then $\mathbb{P}(\mathcal{G}_1) \le (1-(d-1)^{-M})^{5(d-1)^M} \le e^{-5} < 0.01$. This is still true conditioned on starting at any vertex outside of $W_{-10(M+1)(d-1)^M}(B_i)$.

Now, for any $k \in \mathbb{N}$, let E_k denote the event that step $k(M+1) + (i(i+1)/2 - C_4(C_4+1)/2)$ of the random walk is in G or finished. The fact that the walk starts outside of $W_{-10(M+1)(d-1)^M}(B_i)$ means that for $k \le 10(d-1)^M$, step $k(M+1)$ of the random walk is not in B_i, and thus by Lemma 6.2, there is probability at least 0.99 that step $k(M+1) + (i(i+1)/2 - C_4(C_4+1)/2)$ of the random walk will either be in G or finished.

Therefore, $\mathbb{P}(E_k) \ge 0.99$ for all $k \le 10(d-1)^M$. (We do not claim that these events are independent.) Then the expected number of $k \in \{0, \dots, 10(d-1)^M - 1\}$ such that E_k does not occur is at most $0.1(d-1)^M$.

Let $\mathcal{G}_2$ be the event that there are at least $5(d-1)^M$ values $k \in \{0, \ldots, 10(d-1)^M - 1\}$ such that E_k does not occur. By Markov's inequality, $\mathbb{P}(\mathcal{G}_2) \leq \frac{0.1(d-1)^M}{5(d-1)^M} = 0.02$.

Together, we get $\mathbb{P}(\mathcal{G}_1 \cup \mathcal{G}_2) \leq \mathbb{P}(\mathcal{G}_1) + \mathbb{P}(\mathcal{G}_2) \leq 0.01 + 0.02 = 0.03$.

Finally, we claim that if neither $\mathcal{G}_1$ nor $\mathcal{G}_2$ happen, then the random walk finishes within $10(M+1)(d-1)^M + (i(i+1)/2 - C_4(C_4+1)/2)$ steps, which will complete the proof.

If $\mathcal{G}_2$ does not happen, then there are more than $5(d-1)^M$ values of k such that E_k does occur. This means that there are more than $5(d-1)^M$ values of k for which $k(M+1) + (i(i+1)/2 - C_4(C_4+1)/2)$ is either in G or finished. If all of those k were in G (and not finished), then that would produce a M-separated sequence in G of length $\geq 5(d-1)^M$. However, $\mathcal{G}_1$ not occuring means that no such sequence exists. Therefore, there must be some $k \in \{0, \ldots, 10(d-1)^M - 1\}$ such that step $k(M+1) + (i(i+1)/2 - C_4(C_4+1)/2)$ is finished, completing the proof. □

7 Improved Bounds on the Number of Paths to any Set

We now seem to be very close to proving the theorem, as we have shown that there are sets B_i such that $|B_i|$ declines exponentially with i, and there is .97 probability in finishing in $O(i^2)$ steps when starting outside of B_i. However, we do still need to deal with what happens in the 0.03 probability case. To complete our proof of Theorem 1.1, we need to improve the bounds on Lemma 3.2.

Lemma 3.2 showed that $|W_{-j}(S)| \leq (O(\log(n/|S|))^j |S|$. Intuitively, as an average slot has in expectation d hashes to it, you should expect $|W_{-j}(S)|$ to grow like $d^j|S|$ for an average S. Rather than doing a new union bound over all $S \subseteq X$ of a given size at each of the j steps as Lemma 3.2 implicitly did, we can get a stronger result by overcoming a smaller union bound.

Lemma 7.1. *For any $d \geq 3$ and $c < c_d^*$, we have with high probability that for any matching $\mathcal{M}$ any $S \subseteq X$ with $|S| < |X|/(10d)$, and $0 \leq j \leq \log^2(n)$, we have that*

$$|W_{-j}(S)| \leq 10\,(2d + \log(d))^j\, e^{\left(\log^2\left(\log\left(\frac{n}{|S|}\right)\right)\right)} |S|.$$

Proof. First, imagine fixing some $S \subseteq X$ before any hashes are revealed. Then, we generate the hashes of the objects in S. We then perform a union bound over the $\leq d^{|S|}$ choices for which out of d slots each object in S is occupying under $\mathcal{M}$. Next, we reveal which other hashes land in the slots that are occupied by S, thus determining $W_{-1}(S)$.

Then, we again continue iteratively, next generating the hashes of the objects in $W_{-1}(S)$ and union bounding over the $\leq d^{|W_{-1}(S)|}$ choices of where they occupy.

Note that for any fixed choice of the slots occupied by $W_{-k}(S)$, we have that $|W_{-(k+1)}(S)|$ is stochastically dominated by the binomial random variable $Bin(dn, |W_{-k}(S)|/n)$, and so $\mathbb{E}(|W_{-(k+1)}(S))| \leq d|W_{-k}(S)|$. By standard Chernoff bounds, for any $\lambda > 0$ and for any particular choice of the slots occupied by $W_{-k}(S)$, which we denote by the "conditioning on $\mathcal{M}$", or "$\mid \mathcal{M}$" symbol, we have

$$\mathbb{P}\left(|W_{-(k+1)}(S)| \geq (1+\lambda)d|W_{-k}(S)| \mid \mathcal{M}\right) \leq e^{-\lambda^2 d(|W_{-k}(S)|)/(\lambda+2)}.$$

Intuitively, this means that $|W_{-j}(S)|$ should on average be upper bounded by $d^j|S|$. We will use these Chernoff bounds to get a result that holds even in our worst case. Let

$$\lambda_k = \frac{\log(d)}{d} + \frac{4\log(en/|S|)}{d^{k+1}} + 1.$$

We will induct on k to bound $\mathbb{P}\left(|W_{-(k+1)}(S)| \geq \left(\prod_{\ell=0}^{k}(1+\lambda_\ell)\right) d^k|S|\right)$.

We claim that

$$\frac{\lambda_k^2}{\lambda_k+2} \geq \frac{\log(d)}{d} + \frac{2\log(en/|S|)}{d^{k+1}}$$

To show this claim, we set $C_1 = \frac{\log(d)}{d} \in (0,1)$ and $C_2 = \frac{4|S|\log(en/|S|)}{d^{k+1}} \geq 0$. Then

$$\begin{aligned}
&\frac{(C_1+C_2+1)^2}{C_1+C_2+3} \geq C_1 + 0.5C_2 \\
\iff\ & C_1^2 + C_2^2 + 2C_1C_2 + 2C_1 + 2C_2 + 1 \geq C_1^2 + 0.5C_2^2 + 1.5C_1C_2 + 3C_1 + 1.5C_2 \\
\iff\ & 0.5C_2^2 + 0.5C_1C_2 + 0.5C_2 \geq C_1 - 1,
\end{aligned}$$

which is true whenever $C_1 \in (0,1)$ and $C_2 > 0$, as the left side will then be positive while the right side is negative. So, assuming (inductively) that $|W_{-k}(S)| \leq \left(\prod_{\ell=0}^{k-1}(1+\lambda_\ell)\right) d^{k-1}|S|$, we have for any particular choice of the slots occupied by $W_{-k}(S)$ that

$$\begin{aligned}
\mathbb{P}\left(|W_{-(k+1)}(S)| \geq \left(\prod_{\ell=0}^{k}(1+\lambda_\ell)\right) d^k|S| \,\middle|\, \mathcal{M}\right) &\leq \mathbb{P}\left(|W_{-(k+1)}(S)| \geq (1+\lambda_k)d|W_{-k}(S)| \mid \mathcal{M}\right) \\
&\qquad \text{by the inductive hypothesis} \\
&\leq e^{-\lambda_k^2 d(|W_{-k}(S)|)/(\lambda_k+2)} \\
&\leq e^{-\left(\log(d)|W_{-k}(S)| + 2|W_{-k}(S)|\log(en/|S|)/d^k\right)} \\
&\leq e^{-\left(\log(d)|W_{-k}(S)| + 2\left(\prod_{\ell=0}^{k-1}(1+\lambda_k)\right)|S|\log(en/|S|)\right)} \\
&\leq e^{-(\log(d)|W_{-k}(S)| + 2|S|\log(en/|S|))} \\
&\leq d^{-|W_{-k}(S)|}\left(\frac{en}{|S|}\right)^{-2|S|}.
\end{aligned}$$

Now, we union bound over the $\leq d^{|W_{-k}(S)|}$ choices we faced to choose which slots the objects in $W_{-k}(S)$ were matched to. Therefore,

$$\mathbb{P}\left(|W_{-(k+1)}(S)| \geq \left(\prod_{\ell=0}^{k}(1+\lambda_\ell)\right) d^{k+1}|S|\right) \leq \left(\frac{en}{|S|}\right)^{-2|S|}.$$

Summing over all $1 \leq k \leq j$ gives us that

$$\mathbb{P}\left(|W_{-j}(S)| \geq \left(\prod_{k=0}^{j-1}(1+\lambda_k)\right) d^j|S|\right) \leq k\left(\frac{en}{|S|}\right)^{-2|S|}.$$

Now, we union bound over the $\binom{n}{s} \leq \left(\frac{en}{s}\right)^s$ choices for S with $|S| = s$ to say that

$$\begin{aligned}
\mathbb{P}\left(\exists\ S \subseteq X \text{ s.t. } |W_{-j}(S)| \geq \left(\prod_{\ell=0}^{j-1}(1+\lambda_k)\right) d^j|S|\right) &\leq \sum_{s=1}^{n} j\left(\frac{en}{s}\right)^{-s} \\
&\leq \frac{j}{n}\sum_{s=1}^{n} se^{-s}\left(\frac{s}{n}\right)^{s-1} \leq \frac{j}{n}\sum_{s=1}^{n} se^{-s} \\
&\leq \frac{j}{n}\frac{e}{(e-1)^2} \leq o(1)
\end{aligned}$$

as long as $j = o(n)$, which is true as it is $O(\log^2(n))$.

Therefore, we have proven that with high probability, for every $S \subseteq X$ and $0 \leq j \leq \log^2(n)$, we have that

$$|W_{-j}(S)| \leq \left(\prod_{k=0}^{j-1}(1+\lambda_k)\right) d^j|S|.$$

Now, what remains is to upper bound the product $\prod_{\ell=0}^{j-1}(1+\lambda_k)$.

$$\begin{aligned}
&\left(\prod_{k=0}^{j-1}(1+\lambda_k)\right) d^j|S| \\
&= \left(\prod_{k=0}^{j-1}\left(1+\frac{\log(d)}{d}+\frac{4\log(en/|S|)}{d^{k+1}}+1\right)\right) d^j|S| \\
&= \left(\prod_{k=0}^{j-1}\left(1+\frac{4\log(en/|S|)}{(2+\log(d)/d)d^{k+1}}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq \left(\prod_{k=0}^{\infty}\left(1+\frac{\log(en/|S|)}{d^k}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq \left(\prod_{k=0}^{\infty}\left(1+\frac{\log(en/|S|)}{3^k}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq \left(\prod_{k=0}^{\log_3\left(\log\left(\frac{en}{|S|}\right)\right)}\left(1+\frac{\log(en/|S|)}{3^k}\right)\right)\left(\prod_{k=\log_3\left(\log\left(\frac{en}{|S|}\right)\right)}^{\infty}\left(1+\frac{\log(en/|S|)}{3^k}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq \left(\prod_{k=0}^{\log_3\left(\log\left(\frac{en}{|S|}\right)\right)}\left(1+\frac{\log(en/|S|)}{3^k}\right)\right)\left(\prod_{k=0}^{\infty}\left(1+\frac{1}{3^k}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq 4\left(\prod_{k=0}^{\log_3\left(\log\left(\frac{en}{|S|}\right)\right)}\left(1+\frac{\log(en/|S|)}{3^k}\right)\right) (2d+\log(d))^j\,|S| \\
&\leq 4\left(\prod_{k=0}^{\log_3\left(\log\left(\frac{en}{|S|}\right)\right)}(1+\log(en/|S|))\right) (2d+\log(d))^j\,|S| \\
&\leq 4\left((1+\log(en/|S|))^{\log_3(\log(en/|S|))}\right) (2d+\log(d))^j\,|S| \\
&\leq 4e^{(\log(1+\log(en/|S|))\log_3(\log(en/|S|)))}\,(2d+\log(d))^j\,|S| \\
&\leq 10e^{\left(\log^2(\log(en/|S|))\right)}\,(2d+\log(d))^j\,|S|
\end{aligned}$$

as desired. □

8 Proof of Theorem 1.1

Theorem 8.1. *For any $d \geq 3$ and $c < c_d^*$, we have with high probability that the expected insertion time is $O(1)$.*

Proof. Essentially, the idea here is that if we fail to finish in $\leq i^2$ on the run starting outside B_i (which happens with probability at most 0.03), then we are probably still outside B_{3i}, so we try again with a run starting there, then B_{9i}, and so on.

Note that $|B_{3^k i}| \leq (d-1)^{-9^k i^2/4} n$ by Lemma 6.1. Furthermore, if the walk is currently outside of $B_{3^k i}$, then Lemma 6.3 says that we have probability at least 0.97 of finishing in $10(M+1)(d-1)^M + (3^k i(3^k i+1)/2 - C_4(C_4+1)/2)$ steps. For sufficiently large i, we have that $10(M+1)(d-1)^M + (3^k i(3^k i+1)/2 - C_4(C_4+1)/2) \leq (0.51)9^k i^2$.

Formally, for all $i \geq C_5$ for a sufficiently large constant C_5, let $\mathcal{E}_i$ be the event that the starting hash of the object we are inserting is outside of $W_{-9^k i^2/15}(B_{3^k i})$ for every $k \in \mathbb{Z}_{\geq 0}$. Note that if C_5 is sufficiently large (specifically, if $C_5^2/15 \geq 10(M+1)(d-1)^M$), then the $k=0$ case of this hypothesis includes being outside of $W_{-(10(M+1)(d-1)^M)}(B_i)$, satisfying the hypothesis of Lemma 6.3.

We now bound $\sum_k |W_{-9^k i^2/15}(B_{3^k i})|$. For all $i \geq C_5$, we have

$$
\begin{aligned}
|W_{-9^k i^2/15}(B_{3^k i})| &\leq O\left((2d+\log(d))^{9^k i^2/15} e^{\left(\log\left(\log\left(\frac{n}{|B_{3^k i}|}\right)\right)^2\right)} |B_{3^k i}| n \right) \\
&\qquad \text{by Lemma 7.1} \\
&\leq O\left((2d+\log(d))^{9^k i^2/15} e^{\left(\log\left((3^k i)^2 \log(d-1)/2\right)^2\right)} (d-1)^{-(3^k i)^2/4} n \right) \\
&\qquad \text{by Lemma 6.1} \\
&\leq O\left((2d+\log(d))^{9^k i^2/15} e^{\left((\log(3^k i))^3\right)} (d-1)^{-9^k i^2/4} n \right) \\
&\leq O\left(e^{\left((\log(3^k i))^3 + 9^k i^2 (\log(2d+\log(d))/15 - \log(d-1)/4)\right)} n \right) \\
&\leq O\left(e^{-9^k i^2/50} n \right) \qquad \text{for } i \geq C_5 \text{ and } d \geq 3
\end{aligned}
$$

Then

$$
\begin{aligned}
\left| \bigcup_{k=0}^{\infty} W_{-9^k i^2/15}(B_{3^k i}) \right| &\leq \sum_{k=0}^{\infty} O\left(e^{-9^k i^2/50} n \right) \\
&\leq O(e^{-i^2/50} n)
\end{aligned}
$$

In particular, this means that $\mathcal{E}_i$ happens with probability at least $1 - O(e^{-i^2/50})$.

Conditioned on starting on any specific vertex under which $\mathcal{E}_i$ happens, we claim the expected run-time is $O(i^2)$. By Lemma 6.3, since we started outside of $W_{-(10(M+1)(d-1)^M)}(B_i)$, there is a ≥ 0.97 probability of finishing in $\leq 0.51 i^2$ steps (again using $i \geq C_5$). If we do not finish after $0.51 i^2$ steps (the ≤ 0.03 event occurs), then because we started outside of $W_{-9i^2/15}(B_{3i})$ and thus outside of $W_{-((0.51)8i^2/15 + 10(M+1)(d-1)^M)}(B_{3i})$, we are still outside of $W_{-(10(M+1)(d-1)^M)}(B_{3i})$. Then by Lemma 6.3, there is now a 0.97 probability of finishing in $0.51(3i)^2$ more steps.

In general, after k iterations we have taken

$$\sum_{q=0}^{k} 0.51(3^q i)^2 \leq (0.58)9^k i^2 < (9^{k+1})i^2/15 - 10(M+1)(d-1)^M$$

steps, so we are still outside of $W_{-(10(M+1)(d-1)^M)}(B_{3^k i})$. The chance of reaching the k-th stage without finishing is 0.03^k, and the number of steps taken through the k-th stage is $(0.58)9^k i^2$. Therefore, the total expected number of steps taken is at most

$$\sum_{k=0}^{\infty} (0.03^k)((0.58)9^k i^2) \leq i^2$$

So, conditioned on starting at a given vertex under which $\mathcal{E}_i$ does not happen, the expected run time of the random walk is at most i^2.

For any $i \geq C_5$, let $\mathcal{F}_i$ be the event that $\mathcal{E}_i$ happens but $\mathcal{E}_j$ does not happen for every $C_5 \leq j < i$. This is a partition of where our starting hash lands. Let T be the run time of our random walk. Then by the law of total probability,

$$\begin{aligned}
\mathbb{E}(T) &= \sum_{i=C_5}^{\infty} \mathbb{E}(T|\mathcal{F}_i)\mathbb{P}(\mathcal{F}_i) \\
&\leq (C_5)^2 + \sum_{i=C_5+1}^{\infty} \mathbb{E}(T|\mathcal{F}_i)\mathbb{P}(\mathcal{F}_i) && \text{as we start at a vertex where } \mathcal{E}_{C_5} \text{ happens} \\
&\leq O(1) + \sum_{i=C_5+1}^{\infty} (i^2)\mathbb{P}(\mathcal{F}_i) && \text{as we start at a vertex where } \mathcal{E}_i \text{ happens} \\
&\leq O(1) + \sum_{i=C_5+1}^{\infty} (i^2)(O(e^{-(i-1)^2/50})) && \text{as } \mathcal{E}_{i-1} \text{ does not happen} \\
&\leq O(1)
\end{aligned}$$

as desired. □

9 Note on Tail Bounds

Corollary 9.1. *Let $C_6 > 1$. There exists a constant $C_7 = C_7(C_6, c, d) = \Theta(1)$ such that for all $\ell \in \mathbb{N}$, the probability that the random walk takes more than ℓ steps is at most $C_7 \ell^{-C_6}$.*

Proof. We can assume that ℓ is sufficiently large in terms of C_6, d, and ϵ by increasing C_7 accordingly.

First, note that for any $\epsilon_1 > 0$, the value 0.99 in Lemma 6.2 can be replaced with $1 - \epsilon_1$ by requiring C_4 to be large enough such that $\sum_{j=C_4}^{\infty} \frac{2}{j^{1.5}} \leq \epsilon_1$.

Correspondingly, the 0.97 in Lemma 6.3 can be replaced with $1 - \epsilon_2$ any $\epsilon_2 \geq 0$ as well. This is because we can take $\epsilon_1 = \epsilon_2/3$, replace $10(d-1)^M$ with $2\log(3/\epsilon_2)(d-1)^M$ and $5(d-1)$ with $\log(3/\epsilon_2)(d-1)^M$.

Then, letting $\mathcal{G}_1$ in Lemma 6.3 denote the event that $\zeta \geq \log(3/\epsilon_2)(d-1)^M$, we get $\mathbb{P}(\mathcal{G}_1) \leq (1-(d-1)^{-M})^{\log(3/\epsilon_2)(d-1)^M} \leq e^{-\log(3/\epsilon_2)} = \epsilon_2/3$.

Similarly, we use the same definition of E_k and let $\mathcal{G}_2$ in Lemma 6.3 denote the event that there are at least $\log(3/\epsilon_2)(d-1)^M$ values $k \in \{0, \ldots, 2\log(3/\epsilon_2)(d-1)^M - 1\}$ such that E_k does not occur. we get $\mathbb{P}(\mathcal{G}_2) \leq \frac{\epsilon_1(2\log(3/\epsilon_2)(d-1)^M)}{\log(3/\epsilon_2)(d-1)^M} = 2\epsilon_1 = 2\epsilon_2/3$.

This gives a total failure probability of at most ϵ_2.

Now, when considering the probability that the random walk takes at least ℓ steps, we partition on whether $\mathcal{E}_{\log(\ell)}$ happens, where the definition of $\mathcal{E}_i$ is taken from Section 8.

The probability that $\mathcal{E}_{\log(\ell)}$ does not happen is $O(e^{-\log^2(\ell)/50}) = O(\ell^{-\log(\ell)/50}) \leq O(\ell^{-C_6})$ for sufficiently large ℓ.

If $\mathcal{E}_{\log(\ell)}$ does happen, then as in Section 8, the probability of being finished after k iterations is $(\epsilon_2)^k$, and the total number of steps taken up to iteration k is $\leq (0.51)9^k(\log(\ell))^2$. Taking k to be $\log_9(\sqrt{\ell})$, we see that the probability of having taking more than

$$(0.51)9^{\log_9(\sqrt{\ell})}(\log(\ell))^2 = 0.51\sqrt{\ell}(\log(\ell))^2 < \ell$$

steps is at most

$$(\epsilon_2)^{\log_9(\sqrt{\ell})} = \ell^{\log_9(\sqrt{\epsilon_2})} \leq \ell^{-C_6}$$

as long as we have made ϵ_2 sufficiently small in terms of C_6. □

This shows that the tail bounds on the random walk decline faster than any polynomial. This does not show that these tail bounds are exponential: for instance, we have not excluded the possibility that the probability of the random walk taking ℓ steps is $\Theta(e^{-(\log^2(\ell))})$. We believe that the true tail bounds should be exponentially decreasing (for some base of the exponent):

Conjecture 9.2. *There exists constants C_8 and C_9 such that for all $\ell \in \mathbb{N}$, the probability of the random walk taking at least ℓ steps is at most $C_8((C_9)^{-\ell})$.*

10 Modified Insertion Algorithms

Throughout this paper, we have studied a form of random walk insertion where an evicted object chooses uniformly at random one of its other $d-1$ hash values to insert at next. This seems critical, as we are looking at $(d-1)^i$ possibilities of length i. However, some implementations of random walk insertion may simply choose to insert an object at any one of its d hash values, including the one it was just evicted from.

Corollary 10.1. *Theorem 1.1 still holds for the form of random walk insertion where each object chooses uniformly among its d hash functions for re-insertion at each step.*

Proof. In order to prove this, we will give a coupling from the random walk with backtracking, to the random walk without backtracking but with some delays. Essentially, every time the walk backtracks, we can imagine that it just stayed in the same spot for the same amount of time that the backtracking took. We will show that the expected time "wasted" by this backtracking simply multiplies the expected random walk time by at most a $O(1)$ factor. Therefore, if the non-backtracking walk had $O(1)$ expected time, then the backtracking walk also has $O(1)$ expected time.

Every time we are at an object x_{i+1} on step $i+1$ of the random walk and choose the hash that x_{i+1} was just evicted from, that means that we return again to the previous object x_i. Essentially, we will charge this time backtracked to x_i. So, at every step x_i, we unveil how many steps will be

"wasted" only to eventually end up back at x_i through backtracking, and charge those to x_i right then before going to the new object x_{i+1}.

To do this, we want to upper bound the probability of backtracking to return to x_i. This does not include the probability that we cycle around on new hashes to return to x_i, so we are only thinking of returning through the exact same hashes we leave from x_i on.

To return to x_i by backtracking in exactly $2t$ steps (at time $i+2t$), we need to choose t of those steps to be backtracks. Each of those t steps has an independent $\frac{1}{d}$ probability of indeed being a backtrack, while the other t cannot be a backtrack, which has an independent $\frac{d-1}{d}$ probability for each. Therefore, the probability that we return to x_i by backtracking in $2t$ steps is upper bounded by $\binom{2t}{t} d^{-t} \left(\frac{d-1}{d}\right)^t \leq \left(\frac{4(d-1)}{d^2}\right)^t$. Note that this does already include the situation where we backtrack to x_i in fewer than $2t$ steps, and then backtrack again to reach x_i again exactly $2t$ steps after time i.

Then for all $d \geq 3$, the expected delay at x_i from backtracking is at most $\sum_{t=1}^{\infty} 2t \left(\frac{4(d-1)}{d^2}\right)^t = O(1)$ as desired, as the sum is convergent. □

An insertion algorithm that differs significantly from random walk insertion is BFS insertion. Recall that BFS insertion refers to the insertion algorithm where we compute the shortest augmenting path and reassign objects along that. There are $d(d-1)^{i-1}$ possibilities for paths of length i. We will now note that $O(1)$ expected time for BFS insertion comes as a corollary of Lemmas 2.1 and 5.1.

Corollary 10.2. *Let $d \geq 3$ and $c < c_d^*$. With high probability, BFS Insertion takes $O(1)$ expected time.*

Proof. For all $i \in \mathbb{N}$, let D_i be the set of all elements at BFS distance of at least i from U, the set of unoccupied slots. Note that every slot in $N(D_{i+1})$ must be occupied by an object in D_i, so $|N(D_{i+1})| \leq |D_i|$.

By Lemma 5.1, we note that there is a constant $\alpha = \Theta(1)$ such that if $1 \leq |S| \leq |X|/\alpha$, then $|N(S)| \geq (d-1.5)|S|$, as making $|X|/|S|$ a sufficiently large constant makes $p_{|S|} < 0.5$. Apply Lemma 2.1 with this α, and let M be the constant that results. The previous paragraph then implies that for every $i \geq M$, we have $|D_{i+1}|(d-1.5) \leq |N(D_{i+1})| \leq |D_i|$. Applying this iteratively, we get that $|D_{i+M}| \leq (d-1.5)^i |D_M| \leq (d-1.5)^i n$ for every $i \geq M$. Then there exists a $C = \Theta(1)$ (in particular, $C = (d-1.5)^M$) such that for every $i \in \mathbb{N}$, $|D_i| \leq C(d-1.5)^{-i} n$.

The run time of BFS insertion on an object x that is at BFS distance i from U can be bounded by $O((d-1)^i)$, as noted in [FPSS03]. Then using a similar argument to [FPSS03], we find that

$$\begin{aligned}
\mathbb{E}(\text{BFS Insertion Time}) &= O\left(\sum_{i \in \mathbb{N}} (d-1)^i \mathbb{P}(x \text{ at BFS distance } \geq i)\right) \\
&= O\left(\sum_{i \in \mathbb{N}} (d-1)^i \mathbb{P}(h_j(x) \in D_i \ \forall\ 1 \leq j \leq d)\right) \\
&= O\left(\sum_{i \in \mathbb{N}} (d-1)^i \left(\frac{|D_i|}{n}\right)^d\right) \\
&= O\left(\sum_{i \in \mathbb{N}} (d-1)^i (d-1.5)^{-id}\right) = O(1)
\end{aligned}$$

as $(d-1)(d-1.5)^{-d} < 1$ for all $d \geq 3$. □

11 Future Work

One line of improvement would be to improve the tail bounds on the number of steps in the random walk beyond what was proven in Corollary 9.1. Proving (or disproving) Conjecture 9.2 would be a good goal, though weaker improvements would also be worthwhile.

It would also be interesting to give a stronger bound on the $o(1)$ term in our "with high probability" statements. A careful analysis of our and previous works ([FP10, FPS13]) shows that this probability (originating from Lemmas 2.1, 3.1, 4.3, 5.1, and 7.1) could currently be taken to be $O(n^{-\beta})$ for some small $\beta = \Theta(1)$. By a union bound, the failure probability also implies that the $O(1)$ expected insertion time is robust to a sequence of $O(n^{\beta})$ non-hash-dependent deletions and insertions of new elements (not allowing re-insertions of previously deleted elements). Note that $\beta < 1$, so the load factor will remain below c_d^*.

Now that we have an insertion time independent of n, another avenue for future study is to optimize the insertion time in terms of d, c, and absolute constants. Subsequent to the initial version of this paper, Kuszmaul and Mitzenmacher have done work along this line [KM25].

It has been shown under some previous models of cuckoo hashing that the assumption of uniformly random hash functions can be relaxed to families of efficiently computable hash functions while retaining the theoretical insertion time guarantees [CK09, ADW14]. As our proof relies on similar "expansion-like" properties of the bipartite graph to previous work, we believe that Theorem 1.1 should still hold under practically computable hash families.

A different model for generalizing cuckoo hashing, proposed in 2007, gives a capacity greater than one to each hash table slot (element of Y), instead of (or in addition to) additional hash functions [DW07]. The load thresholds for this model are known for both two hashes [CSW07, FR07] and $d \geq 3$ hashes [FKP11]. As in our model, $O(1)$ expected time for random walk insertion has been shown for some values below the load threshold [FP18, Wal22], but it remains open for any capacities greater than one to prove $O(1)$ insertion up to the load thresholds.

In general, it would be nice to extend our random walk insertion time guarantees to other modifications of cuckoo hashing, such as those schemes that increase the probability of a valid matching [KMW09, MP23, Yeo23].

Acknowledgment

We thank Stefan Walzer and the anonymous referees for their helpful comments and discovering issues with a previous version.

References

[ADW14] Martin Aumüller, Martin Dietzfelbinger, and Philipp Woelfel. Explicit and efficient hash families suffice for cuckoo hashing with a stash. *Algorithmica*, 70:428–456, 2014.

[BHR18] Aaron Bernstein, Jacob Holm, and Eva Rotenberg. Online bipartite matching with amortized $O(\log^2(n))$ replacements. *Proceedings of the 2018 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 947–959, 2018.

[CDKL09] Kamalika Chaudhuri, Constantinos Daskalakis, Robert Kleinberg, and Henry Lin. Online bipartite perfect matching with augmentations. *Proceedings of the 28th IEEE Conference on Computer Communications (IEEE INFOCOM)*, pages 1044–1052, 2009.

[CK09] Jeffrey S. Cohen and Daniel M. Kane. Bounds on the independence required for cuckoo hashing. 2009.

[CSW07] Julie Anne Cain, Peter Sanders, and Nick Wormald. The random graph threshold for k-orientiability and a fast algorithm for optimal multiple-choice allocation. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page 469–476, 2007.

[DGM+10] Martin Dietzfelbinger, Andreas Goerdt, Michael Mitzenmacher, Andrea Montanari, Rasmus Pagh, and Michael Rink. Tight thresholds for cuckoo hashing via xorsat. *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming (ICALP)*, pages 213–225, 2010.

[DM03] Luc Devroye and Pat Morin. Cuckoo hashing: Further analysis. *Information Processing Letters*, 86(4):215–219, 2003.

[DW07] Martin Dietzfelbinger and Christoph Weidling. Balanced allocation and dictionaries with tightly packed constant size bins. *Theoretical Computer Science*, 380(1):47–68, 2007.

[EGMP14] David Eppstein, Michael T. Goodrich, Michael Mitzenmacher, and Paweł Pszona. Wear minimization for cuckoo hashing: How not to throw a lot of eggs into one basket. *Proceedings of the International Symposium on Experimental Algorithms (SEA)*, pages 162–173, 2014.

[FJ17] Alan Frieze and Tony Johansson. On the insertion time of random walk cuckoo hashing. *Proceedings of the 2017 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1497–1502, 2017.

[FKP11] Nikolaos Fountoulakis, Megha Khosla, and Konstantinos Panagiotou. The multiple-orientability thresholds for random hypergraphs. *Proceedings of the 2017 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1222–1236, 2011.

[FM12] Alan Frieze and Páll Melsted. Maximum matchings in random bipartite graphs and the space utilization of cuckoo hash tables. *Random Structures & Algorithms*, 41(3):334–364, 2012.

[FMM09] Alan Frieze, Páll Melsted, and Michael Mitzenmacher. An analysis of random-walk cuckoo hashing. *Proceedings of the 2009 International Conference on Randomization and Computation (RANDOM)*, 2009.

[FP10] Nikolaos Fountoulakis and Konstantinos Panagiotou. Orientability of random hypergraphs and the power of multiple choices. *Proceedings of the 37th International Colloquium Conference on Automata, Languages and Programming (ICALP)*, pages 348–359, 2010.

[FP18] Alan Frieze and Samantha Petti. Balanced allocation through random walk. *Information Processing Letters*, 131:39–43, 2018.

[FPS13] Nikolaos Fountoulakis, Konstantinos Panagiotou, and Angelika Steger. On the insertion time of cuckoo hashing. *SIAM Journal on Computing*, 42(6):2156–2181, 2013.

[FPSS03] Dimitris Fotakis, Rasmus Pagh, Peter Sanders, and Paul G. Spirakis. Space efficient hash tables with worst case constant access time. *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, page 271–282, 2003.

[FR07] Daniel Fernholz and Vijaya Ramachandran. The k-orientability thresholds for $G_{n,p}$. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 459–468, 2007.

[GKKV95] Edward Grove, Ming-Yang Kao, P. Krishnan, and Jeffrey Scott Vitter. Online perfect matching and mobile computing. *Proceedings of the 4th International Workshop on Algorithms and Data Structures (WADS)*, 955:194–205, 1995.

[GW10] Pu Gao and Nicholas C. Wormald. Load balancing and orientability thresholds for random hypergraphs. *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 97–104, 2010.

[KA19] Megha Khosla and Avishek Anand. A faster algorithm for cuckoo insertion and bipartite matching in large graphs. *Algorithmica*, 81(9):3707–3724, 2019.

[KM25] William Kuszmaul and Michael Mitzenmacher. Efficient d-ary cuckoo hashing at high load factors by bubbling up. *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3931–3952, 2025.

[KMW09] Adam Kirsch, Michael Mitzenmacher, and Udi Wieder. More robust hashing: Cuckoo hashing with a stash. *SIAM Journal on Computing*, 39(4):1543–1561, 2009.

[Mit09] Michael Mitzenmacher. Some open questions related to cuckoo hashing. *Proceedings of the 17th Annual European Symposium on Algorithms (ESA)*, pages 1–10, 2009.

[MP23] Brice Minaud and Charalampos Papamanthou. Generalized cuckoo hashing with a stash, revisited. *Information Processing Letters*, 181(106356), 2023.

[PR01] Rasmus Pagh and Flemming Friche Rodler. Cuckoo hashing. *Proceedings of the 9th Annual European Symposium on Algorithms (ESA)*, pages 121–133, 2001.

[SHF+17] Yuanyuan Sun, Yu Hua, Dan Feng, Ling Yang, Pengfei Zuo, Shunde Cao, and Yuncheng Guo. A collision-mitigation cuckoo hashing scheme for large-scale storage systems. *IEEE Transactions on Parallel and Distributed Systems*, 28(3):619–632, 2017.

[Wal22] Stefan Walzer. Insertion time of random walk cuckoo hashing below the peeling threshold. *Proceedings of the 30th Annual European Symposium on Algorithms (ESA)*, 244(87):1–11, 2022.

[Yeo23] Kevin Yeo. Cuckoo hashing in cryptography: Optimal parameters, robustness and applications. *43rd Annual International Cryptology Conference (CRYPTO)*, page 197–230, 2023.