Balanced Allocation Through Random Walk

Alan Frieze* Samantha Petti†

March 4, 2018

Abstract

We consider the allocation problem in which $m \le (1-\epsilon)dn$ items are to be allocated to n bins with capacity d. The items $x_1, x_2, \ldots, x_m$ arrive sequentially and when item x_i arrives it is given two possible bin locations $p_i = h_1(x_i), q_i = h_2(x_i)$ via hash functions h_1, h_2. We consider a random walk procedure for inserting items and show that the expected time insertion time is constant provided $\epsilon = \Omega\left(\sqrt{\frac{\log d}{d}}\right)$.

1 Introduction

We consider the following allocation problem. We have m items that are to be allocated to n bins, where each bin has space for d items. The items $x_1, x_2, \ldots, x_m$ arrive sequentially and when item x_i arrives it is given two possible bin locations $p_i = h_1(x_i), q_i = h_2(x_i)$ via hash functions h_1, h_2. We shall for the purpose of this paper assume that $p_i \neq q_i$ for $i \in [m]$ and that (p_i, q_i) is otherwise chosen uniformly at random from $[n]^2$. This model is explicitly discussed in Dietzfelbinger and Weidling [2]. Probabilistic bounds on the number of items m so that all m items can be inserted have been found by Cain, Sanders and Wormald [1] and independently by Fernholtz and Ramachandran [3].

Algorithmically, if $m \le d(1-\varepsilon)n$ where m, n grow arbitrarily large and $\varepsilon > 0$ is small and independent of n, then [2] prove the following:

1. If $d \ge 1 + \frac{\log(1/\varepsilon)}{1-\log 2}$ then w.h.p.[1] all the items can placed into bins.
2. If $d > 90\log(1/\varepsilon)$ then the expected time for a Breadth First Search (BFS) procedure to insert an item is at most $(1/\varepsilon)^{O(\log d)}$.

This model is related to a d-ary version of Cuckoo Hashing (Pagh and Rodler [9]) that was discussed in Fotakis, Pagh, Sanders and Spirakis [4]. Here there are d hash functions and the bins are of size one. This latter paper also uses BFS to insert items.

Item insertion in both of these models can also be tackled via random walk. For d-ary Cuckoo Hashing, Frieze, Mitzenmacher and Melsted [8] and Fountoulakis, Panagiotou and Steger [5] gave

*Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh PA15213. Research supported in part by NSF grant DMS0753472

†School of Mathematics, Georgia Tech., Atlanta, GA30313. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1650044.

[1] A sequence of events $(\mathcal{E}_n, n \ge 0)$ is said to occur with high probability (w.h.p.) if $\lim_{n\to\infty} \mathsf{Pr}[\mathcal{E}_n] = 1$.

$O((\log n)^{O(1)})$ time bounds for random walk insertion and more recently Frieze and Johansson [6] proved an $O(1)$ time bound on random walk insertion, for d sufficently large.

The authors of [2] ask for an analysis of a random walk procedure for inserting an item. They ask for bounds of $O(\log 1/\varepsilon)$ insertion time while maintaining $d = O(\log 1/\varepsilon)$. While we cannot satisfy these demanding criteria, in this note we are able to establish constant expected time bounds with a larger value of d. We first describe the insertion algorithm. We say a bin is saturated if it contains d items.

Random Walk Insertion: RWI

for $i = 1$ to m **do**
begin
  Generate p_i, q_i randomly from $[n]$
  if either of bins p_i, q_i are not saturated, then assign item x_i arbitrarily to one of them.
  if both bins p_i, q_i are saturated then **do**
    begin
    Choose b randomly from $\{p_i, q_i\}$; $y \to x_i$.
A **repeat**
      Let x be a randomly chosen item from bin b.
      Remove x from bin b and replace it with item y.
      Let c be the other bin choice of item x.
      $y \leftarrow x$
      $b \leftarrow c$.
    until bin b is unsaturated.
  Place item x in bin b.
  end
end

Let r_i denote the number of steps in loop A of algorithm RWI. Then,

Theorem 1. *Let $m \le (1-\varepsilon)dn$. Then for some absolute constant $M > 0$,*

$$\mathsf{E}[r_i] \le \frac{4M}{\varepsilon^2} \text{ w.h.p. for } i \in [m] \text{ provided } \varepsilon \ge \sqrt{\frac{M\,(\log(4d)+1)}{d}}. \tag{1}$$

In the analysis below, we take $M = 96$. It goes without saying that we have not tried to optimze M here.

There are two sources of randomness here. The random choice of the hash functions and the random choices by the algorithm. The w.h.p. concerns the assignment of items to bins by the hash functions and the $\mathsf{E}[r_i]$ is then the conditional expectation given these choices.

2 Graphical Description

We use a digraph D' to represent the assignment of items to bins. Each bin is represented as a vertex in D' and item i is represented as a directed edge (p_i, q_i) or (q_i, p_i) that is oriented toward its assigned bin. We say a vertex is *saturated* if its in-degree in D' is d. As the algorithm is executed, we in fact build two digraphs D and D' simultaneously.

We now describe the insertion of an item in terms of D, D'. Let x and y denote the two randomly selected bins for an item. We place a randomly oriented edge between x and y in D. If x and y are unsaturated in D', then we place the edge in the same orientation as in D. If x or y is saturated in D', then we place the edge in D' according to the algorithm RWI, which may require flipping edges in D'. This is repeated for all items. Note that D is a random directed graph with $(1-\varepsilon)dn$ edges. The undirected degree of each vertex in D is the same as in D'. However, the directed degrees will vary. Let D_t and D'_t denote the respective graphs after t edges have been inserted.

We compute the expected insertion time after $(1-\varepsilon)dn$ items have been added by analyzing D'. The expected time to add the next item is equal to the expected length of the following random walk in D'. Select two vertices x and y. If either is unsaturated no walk is taken, so we say the walk has length zero. Otherwise, pick a vertex at random from $\{x, y\}$ and walk "backwards" along edges oriented into the current vertex until an unsaturated vertex is reached. We call this the "replacement walk." As usual in a random walk, vertices may be revisited during a replacement walk. On the other hand, edges are crossed in a direction opposite to their orientation, which is unusual. Note also that after a vertex is visited for the second time, two of its edges will have the opposite direction to what they had originally. This small observation is crucial for the analysis below.

Let G denote the common underlying graph of D, D' obtained by ignoring orientation. In order to compute the expected length of the replacement walk, we analyze the the structure of the subgraph G_S of G induced by a set S which contains all saturated vertices in G. In Section 3, we show that the expected number of connected components of size k among saturated vertices decays geometrically with k and that each component is a tree or contains precisely one cycle. In Section 4 we show that since the components of G_S are sparse and the number of components decays geometrically with size, the expected length of a replacement walk is constant.

3 Saturated Vertices

In this section we describe the structure induced by G on the set of saturated vertices. Throughout this section, our observations rely only on information about the digraph D, and therefore are independent of the RWI algorithm. First we define a set S that is a superset of all saturated vertices.

Definition 1. *Let A be the set of vertices of D with in-degree at least $d-1$ in D and $T_0 = \emptyset$. Given $A, T_0, \ldots T_k$, let T_{k+1} be all the vertices of $V \setminus (A \cup T_0 \cup T_1 \cup \cdots \cup T_k)$ with at least two neighbors in $A \cup T_0 \cup T_1 \cup \cdots \cup T_k$. Let $T = \bigcup T_i$ and $S = A \cup T$.*
Alternatively, let S be the smallest set of vertices that contains A and is closed under adding nodes that have two neighbors in S.

Lemma 2. *The set S defined above contains all saturated vertices.*

Proof. We prove the statement by induction on S_t the set of saturated vertices after the t^{th} edge is added. Since $S_0 = \emptyset$, $S_0 \subseteq S$ vacuously. Assume $S_t \subseteq S$. Note that the addition of a single edge can cause at most one vertex to become saturated. If $S_t = S_{t+1}$, $S_{t+1} \subseteq S$ trivially. For the other case, let v be the vertex that became saturated as a result of the addition of the $(t+1)^{st}$ edge. If v has in-degree at least $d-1$ in D then $v \in A \subseteq S$. Otherwise, there must exist two edges $\{u, v\}$ and $\{w, v\}$ that are oriented out of v in D and are oriented into v in D' at time $t+1$. Since

the orientation of an edge differs in D and D' only if one of its ends is saturated, it follows that u and w must be saturated at time t. By the inductive hypothesis, $u, w \in S$. Therefore, since v is adjacent to two vertices of S, $v \in S$. It follows $S_{t+1} = S_t \cup \{v\} \subseteq S$. □

Next we analyze the structure that G induces on S. Let G_S be the subgraph of G induced by S. The following lemma states that at least half of the vertices of each connected component of G_S are in A. This fact is crucial for proving Lemma 4, which gives an upper bound on the number of components of G_S of size k that decays geometrically with k.

Lemma 3. *Let $S = A \cup T$ as in Definition 1 and let G_S be the graph on S induced by G. Each spanning tree of a component in G_S has the following form:*

1. *A set K of k vertices.*
2. *A set $L = K \cap A$ of size ℓ. We suppose that L induces a forest with s components and $\ell - s$ edges.*
3. *A set $K \setminus L = K \cap T = \{v_1, v_2, \ldots, v_{k-\ell}\}$ where v_i is saturated by the algorithm after v_{i-1}, v_i has $s_i \geq 2$ neighbors in $L \cup \{v_1, v_2, \ldots, v_{i-1}\}$, and $s_1 + s_2 + \cdots + s_{k-\ell} = k - 1 - (\ell - s)$.*
4. *$\ell \geq \frac{k+2}{2}$.*

Proof. As described in the proof of Lemma 2 each vertex in T is adjacent to at least two vertices in S. Therefore $(1), (2), (3)$ hold. For (4), note

$$2(k - \ell) \leq s_1 + s_2 + \cdots + s_{k-\ell} = k - 1 - (\ell - s)$$

which implies

$$k - \ell \leq s - 1 \leq \ell - 2,$$

and (4) follows directly. □

Lemma 4. *Let S be as in Definition 1 and let $M = 96$. When*

$$\sqrt{\frac{M\,(\log(4d) + 1)}{d}} \leq \varepsilon,$$

the expected number of components of G_S of size k is bounded above by

$$\frac{n}{k^2} \exp\left(-\frac{\varepsilon^2 dk}{M}\right).$$

Proof. We use an upper bound on the expected number of spanning trees of size k in G_S as an upper bound on the number of components of G_S of size k.

Let T be an undirected tree on a specified labeled set of k vertices, and let L be a specified subset of vertices of $V(T)$ of size ℓ. Let $\mathcal{A}_{T,L}$ be the event that the T is present in G and all vertices in L are in S. Let $I(L)$ be the sum of the in-degrees of the vertices in L in D, and let $I^*(L)$ be $I(L)$ minus the number of edges in T oriented into a vertex in L in D. We compute

$$\begin{aligned}
\Pr[\mathcal{A}_{T,L}] &\leq \Pr[T \text{ appears in } G]\,\Pr[I(L) \geq \ell(d-1) \mid T \text{ appears in } G] \\
&\leq \Pr[T \text{ appears in } G]\,\Pr[I^*(L) \geq \ell(d-1) - (k-1) \mid T \text{ appears in } G] \qquad (2) \\
&\leq \left(\frac{2d}{n}\right)^{k-1} \exp\left(-\frac{\varepsilon^2 dk}{96}\right) \qquad (3)
\end{aligned}$$

Explanation of (2). If the tree T is present in G, then $I(L) - I^*(L)$ is at most $k-1$. Therefore $I^*(L)$ must be at least $\ell(d-1)-(k-1)$.

Explanation of (3). The term $\left(\frac{2d}{n}\right)^{k-1}$ can be explained as follows: Let $m = (1-\varepsilon)dn$ be the number of pairs offered to algorithm RWI. Each pair has probability $1/\binom{n}{2}$ of being a particular edge. The probability that G contains $k-1$ given edges is then at most

$$m^{k-1}\left(\frac{2}{n(n-1)}\right)^{k-1} = \left(\frac{2(1-\varepsilon)d}{n-1}\right)^{k-1} \le \left(\frac{2d}{n}\right)^{k-1}.$$

The term $\mathsf{exp}\left(-\frac{\varepsilon^2 dk}{96}\right)$ can be explained as follows: $I^*(L)$ is dominated by the sum of $\ell(n-\ell)$ independent Bernouilli random variables, $\xi_{u,v}$, each of which corresponds to an ordered pair (u,v) where $u \notin L, v \in L$ and $\{u,v\} \notin T$. There are at most $\ell(n-1)-(k-1)$ such pairs. Here $\mathsf{Pr}[\xi_{u,v}=1] \le \frac{m}{2\binom{n}{2}} = \frac{(1-\varepsilon)d}{n-1}$ bounds the probability that the edge $\{u,v\}$ exists in G and oriented from u to v in D. The events that edges exist are negatively correlated since the number of edges is bounded above. Therefore, $I^*(L)$ is dominated by $X_\varepsilon \sim Bin\left(\ell(n-1), \frac{(1-\varepsilon)d}{n-1}\right)$. See for example [7], Chapter 21.9 for the definition of dominance and for a background in the basic theory of random graphs.

Thus if

$$p^* = \mathsf{Pr}[I^*(L) \ge \ell(d-1)-(k-1) \mid T \text{ appears in } G]$$

then

$$\begin{aligned} p^* &\le \mathsf{Pr}[X_\varepsilon \ge \ell(d-1)-(k-1)] \\ &\le \mathsf{Pr}[X_\varepsilon \ge \ell\,(d-3)] \\ &\le \mathsf{Pr}[X_\varepsilon \ge (1+\theta)\,\mathsf{E}[X_\varepsilon]], \end{aligned}$$

where $\theta = \frac{\varepsilon d-3}{(1-\varepsilon)d}$. Note that $\theta \le 1$ if and only if $\varepsilon \le \varepsilon_0 = \frac{d+3}{2d}$.

The Chernoff bounds then imply that if $\varepsilon \le 1/2 \le \varepsilon_0$ then

$$p^* \le \mathsf{exp}\left(-\frac{1}{3}\cdot\left(\frac{\varepsilon d-3}{(1-\varepsilon)d}\right)^2 \ell d\,(1-\varepsilon)\right) \le e^{-\varepsilon^2 \ell d/12} \le e^{-\varepsilon^2 kd/24}$$

since $\varepsilon d \ge 6$. When $\varepsilon > 1/2$ we see that X_ε is dominated by $X_{1/2}$ and then replacing ε^2 by $(\varepsilon/2)^2$, we have (3).

We now compute the expected number of trees of size k by applying (3). Let Z_k denote the number of spanning trees in G_S with k vertices. Then

$$\begin{aligned} \mathsf{E}[Z_k] &\le \binom{n}{k} k^{k-2} \sum_{\ell \ge k/2} \binom{k}{\ell} \mathsf{Pr}[\mathcal{A}_{T,L}] \\ &\le \frac{(ne)^k}{k^2} 2^{k-1} \left(\frac{2d}{n}\right)^{k-1} \mathsf{exp}\left(-\frac{\varepsilon^2 dk}{96}\right) \\ &= \frac{n}{k^2}\,\mathsf{exp}\left(k\left(\log(4d)+1-\frac{\varepsilon^2 d}{96}\right)\right) \\ &\le \frac{n}{k^2}\,\mathsf{exp}\left(-\frac{\varepsilon^2 dk}{M}\right). \end{aligned}$$

□

Finally, we give a high probability bound on the size of the maximum component in G_S and show that with high probability each component of G_S contains at most one cycle.

Lemma 5. *Let $\varepsilon \geq \sqrt{\frac{M(\log(4d)+1)}{d}}$ and S be as in Definition 1. Then, w.h.p., the maximum size of a component of G_S is at most*

$$k_0 = \alpha \log n \text{ where } \alpha = \frac{M}{\varepsilon^2 d},$$

and each component contains at most one cycle.

Proof. Let Z_k be the expected number of components of G_S of size k. We use Lemma 4 to compute

$$\mathsf{Pr}[\exists \text{ a component of size} > k_0\ ] \leq \sum_{k_0<k<n} \mathsf{E}[Z_k] \leq n \sum_{k_0<k<n} k^{-2} \exp\left(-\frac{\varepsilon^2 dk}{M}\right) = o(1).$$

It follows that w.h.p. the maximum size of a component of G_S is at most k_0.

Next we show that w.h.p. each component of G_S contains at most one cycle. Let Y_k be the number of sets Q of size k with $e(Q) \geq |Q| + 1$. We bound the number of structures on k vertices consisting of a tree plus two edges by k^{k+2} and compute

$$\begin{aligned}
\mathsf{Pr}[\exists Q\colon |Q| \leq k_0,\, e(Q) \geq |Q|+1] &\leq \sum_{k\leq k_0} \mathsf{E}[Y_k] \\
&\leq \sum_{k\leq k_0} \binom{n}{k} k^{k+2} \left(\frac{2d}{n}\right)^{k+1} \\
&\leq \frac{2d}{n} \sum_{k\leq k_0} k^2 (2de)^k \\
&= o(1).
\end{aligned}$$

□

4 Expected insertion time

Recall from Section 2 the definition of a replacement walk. To bound the expected length of the replacement walk, we first show that a random walk on G' that starts in a size k component of G_S will remain in the component for at most $2k$ steps in expectation.

Lemma 6. *Let G satisfy the conditions of Lemma 5, and let S be as in Definition 1. Consider a random walk on G' begining in a size k component of G_S. Then expected number of steps this random walk takes before leaving G_S is bounded above by $2k$.*

Proof. Consider a random walk on G' beginning in a size k component C of G_S. Note the edges of G' are oriented, so the walk does not backtrack. Therefore, if C is a tree, a random walk can take at most k steps in C before leaving C.

Next suppose C has a cycle of length ℓ. Since C has only one cycle, in any walk in C all edges of the cycle must be visited consecutively (i.e. it is not possible to take a walk on the cycle, leave

the cycle, and then return to the cycle). Let v be the first vertex visited by the walk that is on the cycle. The expected number of times the walk completes the entire cycle at this point is

$$\sum_{i=1}^{\infty} \left(\frac{1}{d^\ell}\right)^i \left(1 - \frac{1}{d^\ell}\right) i = \frac{1}{d^\ell - 1}.$$

Since the walk cannot return to the cycle after the walk leaves the cycle, the expected length of the walk before it leaves C is at most

$$k + \frac{\ell}{d^\ell - 1} \leq 2k.$$

□

Finally, we prove Theorem 1.

Proof. (of Theorem 1). The expected insertion time is the probability that two randomly selected vertices x and y of G' are saturated times the expected length of a replacement walk in the graph G' starting a random saturated vertex. Since the replacement walk ends once an unsaturated vertex is reached, Lemma 6 implies that the expected length of a replacement walk starting at a vertex in a component of size k is bounded above by $2k$. We compute

$$\begin{aligned}
\mathsf{E}[\text{ insertion time }] &\leq 2\sum_{i=1}^{k_0} \mathsf{Pr}[x \in S]\, \mathsf{Pr}[y \text{ in component of size } k \text{ in } G_S\ ]k \\
&\leq 2\sum_{i=1}^{k_0} \frac{|S|}{n} \frac{k\frac{n}{k^2}\,\mathsf{exp}\left(\frac{-\varepsilon^2 dk}{M}\right)}{n} k \\
&\leq 2\sum_{i=1}^{\infty} \mathsf{exp}\left(-\frac{\varepsilon^2 dk}{M}\right) \\
&= \frac{2}{1 - e^{-\varepsilon^2/M}} \\
&\leq \frac{4M}{\varepsilon^2}.
\end{aligned}$$

□

5 Conclusion

We have proved an $O(1)$ bound on the expected insertion time of a natural random walk insertion algorithm. The next step in the analysis of this algorithm is to reduce the dependence of d on ε. It would also be of interest to see the effect of deletions of items on the algorithm.

References

[1] J. Cain, P. Sanders and N. Wormald, The random graph threshold for k-orientiability and a fast algorithm for optimal multiple-choice allocation, *Proceedings of 18th Annual ACM-SIAM SODA* (2007) 469–476.

[2] M. Dietzfelbinger and C. Weidling, Balanced Allocation and Dictionaries with Tightly Packed Constant Sized Bins, *Theoretical Computer Science* 380 (2007) 47–68.

[3] D. Fernholz and V. Ramachandran, The k-orientability thresholds for $G_{n,p}$, *Proceedings of 18th Annual ACM-SIAM SODA* Conference (2007) 459–468.

[4] D. Fotakis, R. Pagh, P. Sanders, and P. Spirakis, Space Efficient Hash Tables With Worst Case Constant Access Time, *Theory of Computing Systems* 8 (2005) 229–248.

[5] N. Fountoulakis, K. Panagiotou and A. Steger, On the Insertion Time of Cuckoo Hashing, *SIAM Journal on Computing* 42 (2013) 2156–2181.

[6] A.M. Frieze and T. Johansson, On the insertion time of random walk cuckoo hashing, *Proceedings of 28th Annual ACM-SIAM SODA Conference* (2017) 1497–1502.

[7] A.M. Frieze and M. Karoński, Introduction to Random Graphs, Cambridge University Press, 2015.

[8] A.M. Frieze, P. Melsted and M. Mitzenmacher, An Analysis of Random-Walk Cuckoo Hashing, *SIAM Journal on Computing* 40 (2011) 291–308.

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