Parallel Randomized Load Balancing [*]

Micah Adler[†] Soumen Chakrabarti[‡] Michael Mitzenmacher[§] Lars Rasmussen[¶]

Abstract

It is well known that after placing n balls independently and uniformly at random into n bins, the fullest bin holds $\Theta(\log n/\log\log n)$ balls with high probability. More recently, Azar *et al.* analyzed the following process: randomly choose d bins for each ball, and then place the balls, one by one, into the least full bin from its d choices [2]. They show that after all n balls have been placed, the fullest bin contains only $\log\log n/\log d + \Theta(1)$ balls with high probability. We explore extensions of this result to parallel and distributed settings.

Our results focus on the tradeoff between the amount of communication and the final load. Given r rounds of communication, we provide lower bounds on the maximum load of $\Omega(\sqrt[r]{\log n/\log\log n})$ for a wide class of strategies. Our results extend to the case where the number of rounds is allowed to grow with n.

We then demonstrate parallelizations of the sequential strategy presented in Azar *et al.* that achieve loads within a constant factor of the lower bound for two communication rounds and almost match the sequential strategy given $\log\log n/\log d + O(d)$ rounds of communication. We also examine a parallel threshold strategy based on rethrowing balls placed in heavily loaded bins. This strategy achieves loads within a constant factor of the lower bound for a constant number of rounds, and it achieves a final load of at most $O(\log\log n)$ given $\Omega(\log\log n)$ rounds of communication. The algorithm also works well in asynchronous environments.

1 Introduction

When n balls are thrown independently and uniformly at random into n bins, it is known that with high probability (by which we shall mean $1 - O(1/n)$) the maximum number of balls received by any bin is $\Theta(\frac{\log n}{\log\log n})$. (In this paper log is used for $\log_2$.) Occupancy results such as this have a long history in the mathematical literature [12, 15] with numerous applications in hashing [2, 7, 14], PRAM simulation [17, 9, 14, 16, 20] and load balancing [2, 7].

[*] A preliminary version of this work appeared in the *Proceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing*, May 1995, pp. 238-247.

[†] U.C. Berkeley. Supported by a Schlumberger Foundation graduate fellowship.

[‡] IBM Almanden Research Center. This work was primarily done while attending U.C. Berkeley, and was supported in part by ARPA under contract DABT63-92-C-0026, by NSF (numbers CCR-9210260 and CDA-8722788), and by Lawrence Livermore National Laboratory.

[§] Digital Systems Research Center. This work was primarily done while attending U.C. Berkeley, and was supported by the Office of Naval Research and by NSF grant CCR-9505448.

[¶] U.C. Berkeley. Supported by a fellowship from U.C. Berkeley.

Azar *et al.* proved an important extension of this result [2]. Suppose we place the balls sequentially, one at a time. For each ball, we choose d bins independently and uniformly at random, and place the ball in the least full bin (ties are broken arbitrarily). When all the balls have been placed, the fullest bin contains only $\frac{\log\log n}{\log d} + O(1)$ balls with high probability [2]. A similar result (in a slightly different model) was also proven in [14].

Unfortunately, this extension requires the resting place of the balls to be determined sequentially. This limits its applicability in parallel and distributed settings, a major drawback when compared to the simple randomized approach. In this paper, we examine the potential of parallelizing the above procedure, as well as other possible strategies for reaching a small maximum load in a distributed environment. We focus on the tradeoff between the number of communication rounds and the final load one can achieve using simple, randomized strategies.

We first show lower bounds that hold for a wide class of load balancing strategies, including natural parallelizations of the method of Azar *et al.* (Following [2], we shall hereafter refer to their algorithm as GREEDY.) We demonstrate a parallelization of GREEDY for two communication rounds that matches the lower bounds to within a constant factor, and we examine alternative parallelizations of GREEDY that are effective when the number of communication rounds is approximately equal to the maximum load. We also examine an idea similar to those used in [14] and [16] based on setting a threshold at each bin: balls that attempt to enter a bin that is already above its threshold for that round must be rethrown. This strategy matches the lower bounds up to a constant factor for any constant number of rounds. Our results show that thresholding strategies can achieve a useful tradeoff between communication cost and the maximum load achieved.

1.1 The balls and bins model

We first describe our model in terms of balls and bins. Each of m balls is to be placed in one of n bins. Each ball begins by choosing d bins as prospective destinations, each choice being made independently and uniformly at random (with replacement) from all possible bins. The balls decide on their final destinations using r rounds of communication, where each round consists of two stages. In the first stage each ball is able to send messages in parallel to any prospective bin, and in the second stage each bin is able to send messages in parallel to any ball from which it has ever received a message. In the final round, the balls commit to one of the prospective bins and the process terminates. Messages are assumed to be of size $\text{polylog}(n, m)$. The goal is to minimize the maximum load, which is defined to be the maximum number of balls in any bin upon completion.

This model is motivated by the following realistic scenario: modern computer networks often have decentralized compute-servers (bins) and client workstations issuing jobs (balls). A distributed load-balancing strategy has to assign jobs to servers. Clients are ignorant of the intention of other clients to submit jobs; contention is known only from server load. Servers are ignorant of jobs from clients that have not communicated with them. It is also prohibitively expensive for clients to globally coordinate job submissions. The primary objectives are to minimize the maximum load achieved as well as the number of communication rounds required. Reducing the number of rounds is an important goal since, in a network setting, the time to complete a round is determined by network latency, which is generally orders of magnitude higher than CPU cycle times.

We examine a class of simple strategies that includes many of the standard algorithms presented in the literature. The strategies we restrict our attention to are *non-adaptive*, in that the possible destinations are chosen before any communication takes place. We will also restrict our discussion to strategies that are *symmetric*, in the sense that all balls and bins perform the same underlying algorithm and all possible destinations are chosen independently and uniformly at random. We believe that these restrictions have practical merit, as an algorithm with these properties would be easier to implement and modify even as the underlying system changes.

Informally, we shall say that an algorithm functions *asynchronously* if a ball (or bin) has to wait only for messages addressed to it (as opposed to messages destined elsewhere). That is, balls and bins are not required to wait for a round to complete before continuing. An algorithm requires *synchronous* rounds if there must exist a synchronization barrier between some pair of rounds; that is, a ball or bin must explicitly wait for an entire previous round to complete before sending a message. In many distributed settings, the ability of an algorithm to function asynchronously can be a significant advantage; an algorithm with synchronous rounds needs some notion of global time to maintain coordination. Note that the algorithm of Azar *et al.* achieves final load no worse than $O(\log\log n)$, but requires $\Omega(n)$ synchronous rounds.

We remark that many of our algorithms can perform asynchronously. In these versions of our algorithms any ball sends or receives at most d messages per round, whereas a bin may receive or send up to $O(\frac{\log n}{\log\log n})$ messages per round. It seems unlikely that this latter number could be made smaller while insisting on a small ($O(\log n)$) number of rounds, for the following reason. During some round there are at least $\Omega(\frac{n}{\log n})$ messages from balls that have not previously communicated. If these messages are distributed randomly, some bin will receive at least $\Omega(\frac{\log n}{\log\log n})$ of them. We can avoid this complication if we modify the algorithms to use synchronous rounds and assume a time limit for each round. In most of our algorithms, a bin must explicitly acknowledge each message and send a negative response to all but a constant number of balls in each round. If these negative responses need not be sent explicitly, and instead a lack of response is interpreted as a negative reply, then the bins need only acknowledge and respond to a constant number of messages per round. The remaining messages can be discarded.

1.2 Summary of our results

In Section 2, we provide a lower bound for a general class non-adaptive and symmetric strategies that include parallel variations of GREEDY [2] and threshold methods [14, 16]. For any fixed number r of rounds of communication and any fixed number d of choices for each ball, we show that with constant probability the maximum load is at least $\Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$. This lower bound is proved by reducing the balls and bins scenario to an edge orientation problem on random graphs.

The rest of the paper deals with upper bounds. Our analysis exploits a basic, general tool that we derive, based on results of Gonnet [10], relating the distribution of the number of balls that land in a bin when balls are thrown independently and uniformly at random and the distribution of Poisson random variables. We note that the close relationship between these two models has been observed and made use of previously, and tighter bounds on specific problems can often be obtained with more detailed analyses; see, for example [3, Chapter 6],[6], or [13]. Apart from enabling us to prove our bounds, the tool may be of independent interest.

In Section 4 we describe an asynchronous parallelization of GREEDY for two rounds that matches the lower bound to within a constant factor for any fixed d. We also describe a more complicated extension of GREEDY in which the number of rounds is allowed to grow with n. We show that this extension achieves a final load no worse than $\frac{\log\log n}{\log d} + 2d + O(1)$ with high probability if we allow $\frac{\log\log n}{\log d} + 2d + O(1)$ synchronous rounds.

In Section 5 we explore an entirely different paradigm based on thresholds, which were also used in [7, 14, 16]. We demonstrate algorithms based on thresholds that asymptotically match the lower bounds for any fixed number of rounds r; that is, the final load is $O\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$ with high probability. However, if r and d are allowed to grow with n, we show that the thresholding method (with threshold one) is inferior to the parallel GREEDY approach: while the latter achieves a maximum load of $O(\frac{\log\log n}{\log\log\log n})$ with $O(\frac{\log\log n}{\log\log\log n})$ rounds, thresholding achieves a maximum load of $\Omega(\log\log n)$ with $\log\log n + O(1)$ rounds. Nevertheless, thresholding has the advantage of functioning asynchronously and offering a continuous trade-off between rounds used and final load achieved.

Finally, we also present results obtained by simulating our algorithms. As one might expect, our parallel strategies lead to a final load close to that obtained by GREEDY, and much better than that achieved by choosing one bin randomly for each ball.

We note that since the writing of this paper a great deal of further work has been done in this area. Stemann [21] extends our work by analyzing an algorithm that asymptotically matches our lower bound in the case of n balls and n bins for *any* r (although his algorithm does not function asynchronously); he also analyzes the case where the number of balls m differs from the number of bins n for his algorithm, as well as considers other similar load balancing problems. Mitzenmacher [18, 19] studies load balancing using multiple choices in dynamic settings related to queueing networks. Czumaj and Stemann [8] provide a general framework that extends the model and the results of Azar, Broder, Karlin and Upfal [2]; for example, they also consider the average allocation time in a threshold-based scheme similar in spirit to the one we analyze here. Berenbrink, Meyer auf der Heide, and Schröder develop algorithms for dealing with weighted balls [4]. We note that these papers also contain many other results for further variations on the original model; because of the many interesting and useful variations of the basic balls and bins problem, we expect it will continue to be a fruitful area of research.

1.3 Basic Lemmas

In preparation for our proofs, we make note of two useful lemmas. Let $B(n, p)$ is a Bernoulli random variable with parameters n and p. The first statement can be proven by standard coupling methods:

Lemma 1 *Let $X_1, X_2, \ldots, X_n$ be a sequence of random variables in an arbitrary domain, and let $Y_1, Y_2, \ldots, Y_n$ be a sequence of binary random variables, with the property that $Y_i = Y_i(X_1, \ldots, X_{i-1})$. If*

$$\mathsf{Pr}(Y_i = 1 \mid X_1, \ldots, X_{i-1}) \leq p,$$

then

$$\mathsf{Pr}(\sum_{i=1}^{n} Y_i \geq k) \leq \mathsf{Pr}(B(n,p) \geq k);$$

and similarly, if

$$\mathsf{Pr}(Y_i = 1 \mid X_1, \ldots, X_{i-1}) \geq p,$$

then

$$\mathsf{Pr}(\sum_{i=1}^{n} Y_i \leq k) \leq \mathsf{Pr}(B(n,p) \leq k).$$

■

The second lemma presents some useful Chernoff-type bounds that will be used frequently throughout the paper; proofs may be found in [11].

Lemma 2 *If X_i $(1 \leq i \leq n)$ are independent binary random variables such that $\mathsf{Pr}[X_i = 1] = p$, then the following hold:*

$$\text{For } t \geq np, \qquad \mathsf{Pr}\left(\sum_{i=1}^{n} X_i \geq t\right) \leq \left(\frac{np}{t}\right)^t \mathrm{e}^{t-np}. \tag{1}$$

$$\text{For } t \leq np, \qquad \mathsf{Pr}\left(\sum_{i=1}^{n} X_i \leq t\right) \leq \left(\frac{np}{t}\right)^t \mathrm{e}^{t-np}. \tag{2}$$

In particular, we have

$$\mathsf{Pr}\left(\sum_{i} X_{i=1}^{n} \geq \mathrm{e}np\right) \leq \mathrm{e}^{-np}, \text{ and} \tag{3}$$

$$\mathsf{Pr}\left(\sum_{i} X_{i=1}^{n} \leq np/\mathrm{e}\right) \leq \mathrm{e}^{(2/\mathrm{e}-1)np}. \tag{4}$$

■

2 Lower bounds

2.1 The random graph model

We first develop a general model for lower bounds that captures a class of non-adaptive, symmetric load balancing strategies. Our lower bounds are expressed in terms of the number of rounds of communication, r, and the number of choices available to each ball, d. In Section 2.2, we will focus on the case where $d = 2$ and $r = 2$, extending the results to arbitrary values of r and d in Section 2.3.

For our bounds, we will rephrase the balls and bins problem in terms of a random graph orientation problem. The relationship between balls and bins problems and random graphs has been noted previously [1, 14, 16]. Here, we show that proving a lower bound for the balls and bins problem is equivalent to showing that, with sufficiently high probability, a specific subgraph appears in a random graph. These results on random graphs may be of independent interest.

We temporarily restrict ourselves to the case of $d = 2$. Associate with each bin a vertex of a graph with n vertices. Each ball can be represented by an undirected edge in this graph, where the vertices of the edge correspond to the two bins chosen by the ball. (For convenience, in this section, we assume that each ball chooses two bins *without* replacement. This has the effect of ensuring that no self-loops arise in the graph. Multiple edges, however, may arise: these correspond to two balls that have chosen the same pair of bins. Our proofs may be modified to allow self-loops, and our restriction does not change our asymptotic results.) With each edge we shall associate an orientation. An edge, or ball, shall be oriented toward the vertex, or bin, that it chooses as its final destination. The goal of the algorithm is thus to minimize the maximum indegree over all vertices of the graph. In the case where there are m balls and n bins, the corresponding graph is a random graph chosen uniformly from the set of all graphs with n vertices and m edges, where an edge may occur with multiplicity greater than one. Abusing standard notation slightly, we call this class of graphs $\mathcal{G}_{m,n}$. We shall focus on the case $m = n$, since this is the most interesting case in terms of behavior.

We now characterize communication in this model. For each round of communication, every ball and bin will determine a larger portion of the graph around it. Following standard terminology, we define the *neighborhood* of an edge e, denoted by $N(e)$, to be the set of all edges incident to an endpoint of e. For a set S of edges, we write $N(S)$ for $\cup_{e \in S} N(e)$. The neighborhood of a vertex v, denoted by $N(v)$, is the set of all edges incident to v. To model the increasing knowledge gained by communication, we introduce more general related definitions.

Definition 3 *The* l-neighborhood *of an edge e, denoted by $N_l(e)$, is defined inductively by: $N_1(e) = N(e)$, $N_l(e) = N(N_{l-1}(e))$.*

Definition 4 *The* (l, x)-neighborhood *of an edge $e = (x, y)$, denoted by $N_{l,x}(e)$, is defined inductively by: $N_{1,x}(e) = N(x) - \{e\}$, $N_{l,x}(e) = N(N_{l-1,x}(e)) - \{e\}$.*

Intuitively, for each round of communication, a ball learns about a larger neighborhood in the graph.

Specifically, since we are working towards lower bounds, we may assume that the messages sent by the bins contain all available information whenever possible. Consider an r round protocol for the balls and bins problem where balls commit to their final choice in the rth round. In this case, we may assume a ball knows everything about the balls in its $(r-1)$-neighborhood, and no more, before it must commit to a bin in the rth round; this may be verified formally by a simple induction argument. (Note that our model does not allow for "pointer jumping"; the neighborhood expands by just one level each round.)

We now describe an assumption that we use to show that the final load is high with constant probability; that is, with probability greater than some fixed constant $\gamma > 0$. The l-neighborhood of a ball $e = (x, y)$ splits into two subgraphs corresponding to $N_{l,x}(e)$ and $N_{l,y}(e)$; these are the parts of the neighborhood the ball learns about from each bin. Suppose that these two subgraphs of the ball's l-neighborhood are isomorphic rooted trees, with the roots being x and y. In this case we say that the ball has a *symmetric l-neighborhood*, or, more graphically, we say that the ball is *confused* (see Figure 1). The ball has no reason to prefer one bin over another, and must essentially choose randomly. For the moment, we explicitly assume that in this situation the ball chooses a bin randomly with probability $\frac{1}{2}$; we shall expand on this shortly.

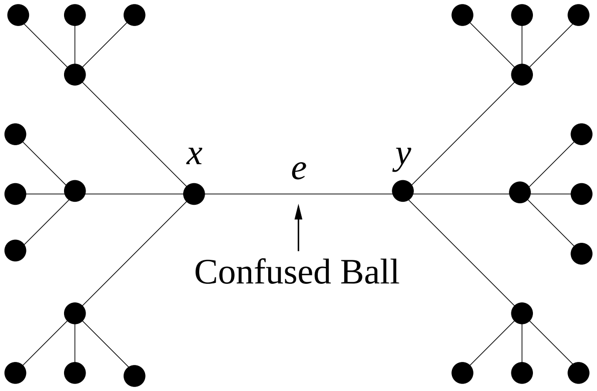

Figure 1: The central edge e corresponds to a confused ball: its left and right neighborhoods ($N_{2,x}(e)$ and $N_{2,y}(e)$) appear the same.

Assumption 5 *If a ball has a symmetric $(r-1)$-neighborhood, then in any protocol of r rounds it chooses a destination bin with a fair coin flip.*

We further justify this assumption at the end of Section 2.2.

2.2 The $d = 2$, $r = 2$ case

By Assumption 5, if many confused balls are incident on one bin, then with constant probability over half of them will opt for the bin, which will become overloaded. We show that for r and T suitably related to n, a random graph G with n vertices and n edges has, with high probability, an *isolated* (T, r)-tree, defined as follows:

Definition 6 *A* (T, r) tree *is a rooted, balanced tree of depth r, such that the root has degree T and each internal node has $T - 1$ children. A* (T, r) *tree is* isolated *in a graph G if it is a connected component of G with no edges of multiplicity greater than one.*

Note that a (T, r) tree is slightly different from a $(T - 1)$-ary tree of depth r, in that the root has degree T. (See Figure 2.)

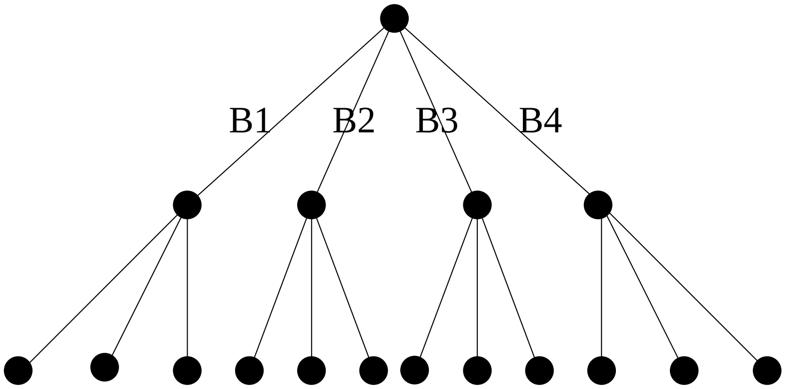

Figure 2: A (4,2) tree. Each vertex has degree 4, and the depth of the tree is 2. Balls B1-B4 will be confused after one round of communication, and hence each orients itself to the root with probability 1/2.

We shall show that a random graph from $\mathcal{G}_{n,n}$ contains a (T, r) tree of a suitable size. For convenience, we begin with the simple case of $d = 2$ and $r = 2$.

Theorem 7 *With constant probability, a random graph from $\mathcal{G}_{n,n}$ contains an isolated $(T, 2)$ tree with $T = (\sqrt{2} - o(1))\sqrt{\frac{\log n}{\log \log n}}$.*

Since, with constant probability, half of the confused edges in an isolated $(T, 2)$ tree adjacent to the root will orient themselves toward it (by Assumption 5), the above theorem immediately yields the following corollary:

Corollary 8 *Any non-adaptive, symmetric load distribution strategy for the balls and bins problem with n balls and n bins satisfying Assumption 5, where $d = 2$ and $r = 2$, has a final load at least $\left(\frac{\sqrt{2}}{2} - o(1)\right)\sqrt{\frac{\log n}{\log \log n}}$ with at least constant probability.*

Corollary 8 demonstrates that the $O(\log \log n)$ bounds achieved by Azar *et al.* using the GREEDY strategy cannot be achieved by any two round strategy where each ball has two choices.

Although Theorem 7 is stated for the case where m, the number of edges, and n, the number of bins, are equal, it will be useful to write the number of edges as m throughout the proof. Besides making the proof clearer, this will allow us to extend the theorem easily to a broader range of m; this is discussed after the proof.

Proof of Theorem 7: Let $\vec{v} = (v_0, v_1, \ldots, v_{T^2})$ be a vector of T^2+1 vertices. Let $X_{\vec{v}}$ be an indicator random variable that is 1 if v_0 is the root of an isolated $(T,2)$ tree, $v_1, \ldots, v_T$ are the nodes of depth 1, $v_{T+1}, \ldots, v_{2T-1}$ are the children of v_1, and so on, and let $X = \sum_{\vec{v}} X_{\vec{v}}$. We show that $X > 0$ with at least constant probability by determining the expectation and variance of X and applying the simple bound (from [5], equation (3) of I.1):

$$\mathsf{Pr}(X=0) \quad \leq \quad 1 - \frac{\mathsf{E}[X]^2}{\mathsf{E}[X^2]}.$$

We first compute $\mathsf{E}[X]$. The multinomial coefficient $\binom{n}{1;T;T-1;\ldots;T-1}$, where $(T-1)$ occurs T times and the sum of the terms on the second level is $1+T+T(T-1) = T^2+1$, gives the number of possible choices for $\vec{v}$; we must first choose the root, and then the T children of the root, and then the $T-1$ children for each of T children. We now choose a specific $\vec{v}$ and determine the probability p that $X_{\vec{v}}$ is 1. If $X_{\vec{v}}$ is 1, there must be T^2 edges corresponding to the (T,r) tree connecting the vertices of $\vec{v}$ and no other edges incident to these vertices. We must first choose T^2 of the m edges to make up the tree: there are $\binom{m}{T^2}(T^2)!$ ways of doing this. The remaining edges must lie on the remaining $n-(T^2+1)$ vertices so that the tree is isolated. Hence

$$p = \frac{\binom{n-(T^2+1)}{2}^{m-T^2}\binom{m}{T^2}(T^2)!}{\binom{n}{2}^m}.$$

Using the linearity of expectations, we have

$$\mathsf{E}[X] \quad = \quad \frac{\binom{n}{1;T;T-1;\ldots;T-1}\binom{n-(T^2+1)}{2}^{m-T^2}\binom{m}{T^2}(T^2)!}{\binom{n}{2}^m}.$$

This unwieldy expression can be simplified by canceling appropriately and noting that we will choose T small enough so that many terms are $o(1)$ (in the case $m=n$). For instance, if $T = o(\log n)$, then we have

$$\frac{\binom{n-(T^2+1)}{2}^{m-T^2}}{\binom{n}{2}^{m-T^2}} \quad = \quad \mathrm{e}^{-2(T^2+1)m/n}(1+o(1))\,; \tag{5}$$

$$\frac{\binom{m}{T^2}(T^2)!}{n^{T^2}} \quad = \quad \left(\frac{m}{n}\right)^{T^2}(1+o(1))\,; \tag{6}$$

$$\frac{\binom{n}{1;T;T-1;\ldots;T-1}}{(n-1)^{T^2}} \quad = \quad \frac{n}{((T-1)!)^{T+1}T}(1+o(1))\,;. \tag{7}$$

We thus obtain

$$\mathsf{E}[X] \quad = \quad \frac{n\left(\frac{2m}{n}\right)^{T^2}}{\mathrm{e}^{2(T^2+1)m/n}((T-1)!)^{T+1}T}(1+o(1)). \tag{8}$$

We now examine how to compute $\mathsf{E}[X^2]$. Note that, because we are considering only isolated $(T, 2)$ trees, if $\vec{v} \neq \vec{w}$ then $X_{\vec{v}}$ and $X_{\vec{w}}$ can both equal 1 if and only if $\vec{v}$ and $\vec{w}$ consist of disjoint sets of vertices or are equal. This simplifies the calculation of $\mathsf{E}[X^2]$ considerably. Since

$$\mathsf{E}[X^2] \quad = \quad \mathsf{E}[X] + \sum_{\vec{v} \neq \vec{w}} \mathsf{E}[X_{\vec{v}} X_{\vec{w}}],$$

it suffices to compute the second term. The calculation is similar to that for $\mathsf{E}[X]$ and proceeds as follows.

The number of possible disjoint pairs $\vec{v}$ and $\vec{w}$ is $\binom{n}{1;T;T-1;\ldots;T-1;1;T;T-1;\ldots;T-1}$, and the probability q that a given disjoint pair $\vec{v}$ and $\vec{w}$ yields two isolated $(T, 2)$ trees is

$$q = \frac{\binom{n-(2T^2+2)}{2}^{m-2T^2} \binom{m}{2T^2} (2T^2)!}{\binom{n}{2}^m}.$$

From these terms we derive $\sum_{\vec{v} \neq \vec{w}} \mathsf{E}[X_v X_w] = \mathsf{E}[X]^2(1 + o(1))$, by simplifying with equations entirely similar to equations (5), (6), and (7).

$$\begin{aligned}
\frac{\binom{n-(2T^2+2)}{2}^{m-2T^2}}{\binom{n}{2}^{m-2T^2}} \quad &= \quad \mathrm{e}^{-2(2T^2+2)m/n}(1 + o(1))\,; \\
\frac{\binom{m}{2T^2}(2T^2)!}{n^{2T^2}} \quad &= \quad \left(\frac{m}{n}\right)^{2T^2} (1 + o(1))\,; \\
\frac{\binom{n}{1;T;T-1;\ldots;T-1;1;T;T-1;\ldots;T-1}}{(n-1)^{2T^2}} \quad &= \quad \frac{n}{((T-1)!)^{2T+2}T^2}(1 + o(1))\,; .
\end{aligned}$$

We thus have that $\mathsf{E}[X^2] = \mathsf{E}[X] + \mathsf{E}[X]^2(1 + o(1))$. It now suffices to choose a T such that $\mathsf{E}[X]$ is bounded below by a constant. Taking the logarithm of both sides of Equation 8, we find this will hold as long as

$$T^2 \log T + \frac{2T^2 m}{n} - T^2 \log \frac{2m}{n} \leq \log n + o(\log n). \tag{9}$$

Hence for $m = n$ there exists a $(T, 2)$ tree with $T = (\sqrt{2} - o(1))\sqrt{\frac{\log n}{\log \log n}}$ with constant probability. ■

Remark: Although we have stated Theorem 7 only for the case $m = n$, it is clear that nearly the same proof, and hence equation (9), applies for a range of m. For example, if $m = \frac{n}{\log^k n}$ for some fixed k, then we again have $\Omega\left(\sqrt{\frac{\log n}{\log \log n}}\right)$ bounds on the maximum load with at least constant probability. The important points to check are where we have stated that some terms are $o(1)$, as in equations (5), (6), and (7), which places restrictions on the possible values of m and T. It is also worth noting that equation (9) can be improved somewhat. We have insisted up to this point that our trees be isolated, even though there is no reason that the leaf nodes cannot be adjacent to nodes outside the tree. Taking advantage of this fact would reduce the $\frac{2T^2 m}{n}$ terms of equation (9); this does affect the bound in the case where $m = n$. ■

We now justify Assumption 5. Although it may at first seem unreasonable to insist that balls with symmetric r-neighborhoods choose a bin randomly, obvious tie-breaking schemes do not affect the lower bound. For instance, if the balls are ordered at the bins, either by random I.D. numbers or by a random permutation, and then choose a bin according to their rank, the balls are essentially choosing a bin at random. The proof can also easily be modified for the case where the balls are ranked at the bins by some fixed ordering by using the symmetry of the destination choices of the balls. Similarly, if bins are numbered and given a preferred ordering in case of ties, then with constant probability there is still a $(T, 2)$ tree whose root has the given final load.

2.3 The general case

One can extend Theorem 7 to the case where $d > 2$ and/or $r > 2$; in fact, the extension also applies if r and d grow sufficiently slowly with n.

When $r > 2$ and $d = 2$, the balls and bins scenario can again be reduced to a random graph problem; instead of showing the existence of a $(T, 2)$ tree, one needs to demonstrate the existence of a (T, r) tree. When $d > 2$ we must consider hypergraphs instead of graphs. In this model, balls correspond to hyperedges of d distinct vertices in the hypergraph. The degree of a vertex is the number of incident hyperedges. A tree of hyperedges is simply a connected acyclic hypergraph; that is, there is no path from a vertex back to itself along adjacent hyperedges. The depth of a tree is the number of hyperedges in a longest path from the root to a leaf.

Figure 3 gives an example of a $(3, 2, 3)$ tree.

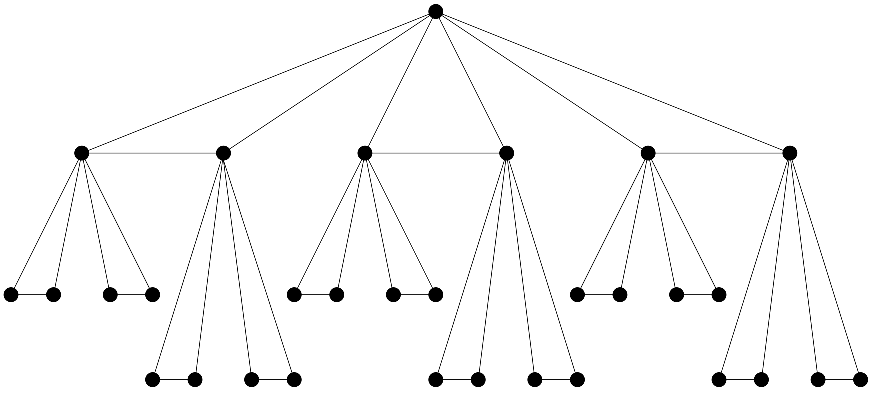

Figure 3: A (3,2,3) tree. Each triangle corresponds to a hyperedge of three vertices.

The l-neighborhood and (l, x)-neighborhood of a ball can be defined for hypergraphs similar to Definitions 3 and 4. As in Assumption 5, we will assume that if a ball has a symmetric $(r-1)$-neighborhood, it chooses one of the d bins uniformly at random at the end of an r round algorithm; for convenience, we still call this Assumption 5. Thus the root of an isolated (T, r, d) tree will end with $\frac{T}{d}$ balls with at least constant probability. As we shall see, whether an isolated (T, r, d) tree exists is essentially a matter of its size, in terms of the number of edges it contains.

In the remainder of this section, we shall prove the following theorem:

Theorem 9 *For any fixed r and d, there exists a $T = \Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$ such that with constant probability, a random graph with n vertices and n hyperedges of size d contains an isolated (T, r, d) tree.*

The theorem immediately yields the following corollary:

Corollary 10 *Any non-adaptive, symmetric load distribution strategy for the balls and bins problem satisfying Assumption 5 where d and r are constants and $m = n$ has a final load at least $\Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$ with constant probability.*

Corollary 10 generalizes Corollary 8 by demonstrating that the $O(\log\log n)$ bounds achieved by Azar *et al.* using the GREEDY strategy cannot be achieved by any strategy in which each ball has a fixed constant number of choices and in which only a constant number of rounds are used.

The constant factor in the lower bound (for r and d fixed) is dependent on d. The proof of the theorem also yields results when d grows with n. With constant probability the final load is $\frac{T}{d}$ if there is a (T, r, d) tree in the corresponding hypergraph. Similarly, if there is a (T, r, d) tree in the corresponding hypergraph, then with probability d^{-T} the final load is T; this can be used to give negative results by showing that no non-adaptive, symmetric load distribution strategy achieves load T with high probability when $d^T = o(n)$. For example, we have the following corollary:

Corollary 11 *Any non-adaptive, symmetric load distribution strategy for the balls and bins problem satisfying Assumption 5 with $m = n$ where $d = O\left(\frac{\log\log n}{\log\log\log n}\right)$ and $r = O\left(\frac{\log\log n}{\log\log\log n}\right)$ has a final load at least $\Omega\left(\frac{\log\log n}{\log\log\log n}\right)$ with probability at least $O\left(1/\log^c n\right)$ for some constant c (dependent on d and r but independent of n).*

Proof of Theorem 9: As in Theorem 7, although we are considering only the case $m = n$, we continue to distinguish m and n when writing the proof, in the interests of enhancing clarity and allowing generalizations to other values of m where suitable.

We begin by finding the expected number of isolated (T, r, d) trees. Let V denote the total number of vertices in the desired (T, r, d) tree and let E denote the total number of hyperedges of the tree. A given list of V vertices corresponds to a unique (T, r, d) tree in a canonical way. For a given list of V vertices, the probability p that the corresponding tree component exists is

$$p = \frac{\binom{n-V}{d}^{m-E}\binom{m}{E}E!}{\binom{n}{d}^m}.$$

That is, we must choose from the m edges the E edges of the tree, and all other edges cannot be incident to any vertex of the tree.

To compute the number of possible trees, we consider all lists of V vertices, where the first vertex corresponds to the root, the next $T(d-1)$ vertices correspond to the first T edges, and so on. Each possible re-ordering of the vertices that make up an edge leads to the same tree; also, the subtrees at each level can be permuted without changing the tree. Keeping these facts in mind, we find the total possible number of vectors corresponding to distinct isolated trees, which we denote by N, is

$$\begin{aligned} N &= \frac{\binom{n}{1;T(d-1);(T-1)(d-1);\ldots;(T-1)(d-1)}\binom{T(d-1)}{d-1;\ldots;d-1}\binom{(T-1)(d-1)}{d-1;\ldots;d-1}^{(E-T)/(T-1)}}{T![(T-1)!]^{(E-T)/(T-1)}} \\ &= \frac{n!}{(n-V)![(d-1)!]^E T[(T-1)!]^{(E-1)/(T-1)}}, \end{aligned}$$

where in the first multinomial coefficient, the sum of the terms on the second level, $1+T(d-1)+(T-1)(d-1)+\ldots+(T-1)(d-1)$, is V.

Routine calculations yield that $V = 1 + T(d-1)\frac{[(T-1)(d-1)]^r-1}{(T-1)(d-1)-1}$ and $E = \frac{V-1}{d-1}$. For suitable values of T (and hence E and V), after absorbing lower order terms the product Np reduces to:

$$Np = \frac{n\left(\frac{m}{n}\right)^E d^E(1+o(1))}{\mathrm{e}^{Vdm/n}T((T-1)!)^{(E-1)/(T-1)}}.$$

Hence the expected number of (T, r, d) trees can be made at least a constant for suitable choices of T, r, and d; this requires

$$\log n + E\log\frac{m}{n} + E\log d \geq \frac{Vdm}{n} + (E-1)\log(T-1) + o((E-1)\log(T-1)).$$

Noting that $E \approx (Td)^r$, we find that (up to lower order terms) we require $\log n \geq (Td)^r \log T$ when $m = n$ and d is a fixed constant. In particular, we can find a T of size at least $\Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$ such that the expected number of (T, r, d) trees is at least a constant when $m = n$.

We must now also show that the variance is not too large. Recall that

$$\mathsf{E}[X^2] = \mathsf{E}[X] + \sum_{\vec{v}\neq\vec{w}} \mathsf{E}[X_{\vec{v}}X_{\vec{w}}].$$

Again, we make use of the fact that the trees are isolated. If $\vec{v}$ and $\vec{w}$ share no vertices, then the probability q that $X_{\vec{v}}$ and $X_{\vec{w}}$ are both 1 is

$$q = \frac{\binom{n-2V}{d}^{m-2E}\binom{m}{2E}2E!}{\binom{n}{d}^m}.$$

The number of pairs of disjoint $\vec{v}$ and $\vec{w}$ is

$$\frac{n!}{(n-2V)![(d-1)!]^{2E}T^2[(T-1)!]^{2(E-1)/(T-1)}}.$$

Again it is simple to check that

$$\mathsf{E}[X^2] = \mathsf{E}[X] + \mathsf{E}[X]^2(1+o(1)),$$

which is sufficient to prove the theorem. ■

3 The Poisson approximation

We now switch from proving lower bounds to examining parallel algorithms for load balancing based on the GREEDY idea. Before developing any particular algorithms, we derive a useful general tool. The main difficulty in analyzing balls and bins problems is that it is often hard to handle the dependencies that naturally arise in such systems. For example, if one bin is empty, then it is less likely that another bin is empty; the loads of the various bins are correlated. It will be useful to have a general way to circumvent these sorts of dependencies. We show here how to do so when we are examining the probability of a rare event.

It is well known that after throwing m balls independently and uniformly at random into n bins, the distribution of the number of balls in a given bin is approximately Poisson with mean $\frac{m}{n}$. We would like to say that the joint distribution of the number of balls in *all* the bins is well approximated by assuming the load at *each* bin is an *independent* Poisson random variable with mean $\frac{m}{n}$. This would allow us to treat bin loads as independent random variables, and hence use standard techniques such as Chernoff bounds.

Suppose m balls are thrown into n bins independently and uniformly at random, and let $X_i^{(m)}$ be the number of balls in the i-th bin, where $1 \leq i \leq n$. Let $Y_1^{(m)}, \ldots, Y_n^{(m)}$ be independent Poisson random variables with mean $\frac{m}{n}$. We will omit the superscript when it is clear from the context. We derive a relation between these two sets of random variables, adapting an argument used by Gonnet [10] to determine the expected maximum number of balls in a bin.

Theorem 12 *Let $f(x_1, \ldots, x_n)$ be any non-negative function. Then*

$$\mathsf{E}[f(X_1, \ldots, X_n)] \quad \leq \quad \sqrt{2\pi e m}\,\mathsf{E}[f(Y_1, \ldots, Y_n)]. \tag{10}$$

Further, if $\mathsf{E}[f(X_1, \ldots, X_n)]$ is either monotonically increasing or monotonically decreasing with m, then

$$\mathsf{E}[f(X_1, \ldots, X_n)] \quad \leq \quad 4\,\mathsf{E}[f(Y_1, \ldots, Y_n)]. \tag{11}$$

Proof: We have that

$$\begin{aligned}
\mathsf{E}[f(Y_1, \ldots, Y_n)] &= \sum_{k=0}^{\infty} \mathsf{E}\Big[f(Y_1, \ldots, Y_n)\Big|\textstyle\sum Y_i = k\Big]\,\mathsf{Pr}(\textstyle\sum Y_i = k) \\
&\geq \mathsf{E}\Big[f(Y_1, \ldots, Y_n)\Big|\textstyle\sum Y_i = m\Big]\,\mathsf{Pr}(\textstyle\sum Y_i = m) \\
&= \mathsf{E}[f(X_1, \ldots, X_n)]\,\mathsf{Pr}(\textstyle\sum Y_i = m)
\end{aligned}$$

where the last equality follows from the fact that the joint distribution of the Y_i given $\sum Y_i = m$ is exactly that of the X_i, as can be checked by comparing the probabilities of any given set of bin loads in both cases. As $\sum Y_i$ is Poisson distributed with mean m, we now have

$$\mathsf{E}[f(Y_1, \ldots, Y_n)] \quad \geq \quad \mathsf{E}[f(X_1, \ldots, X_n)]\frac{m^m e^{-m}}{m!},$$

and using Stirling's approximation now yields equation (10).

If $\mathsf{E}[f(X_1,\ldots,X_n)]$ increases with m, then similarly

$$\begin{aligned}
\mathsf{E}[f(Y_1,\ldots,Y_n)] &\geq \sum_{k=m}^{\infty} \mathsf{E}\Big[f(Y_1,\ldots,Y_n)\Big|\sum Y_i = k\Big]\,\mathsf{Pr}(\sum Y_i = k) \\
&= \sum_{k=m}^{\infty} \mathsf{E}\Big[f(X_1^{(k)},\ldots,X_n^{(k)})\Big]\,\mathsf{Pr}(\sum Y_i = k) \\
&\geq \sum_{k=m}^{\infty} \mathsf{E}\Big[f(X_1^{(m)},\ldots,X_n^{(m)})\Big]\,\mathsf{Pr}(\sum Y_i = k) \\
&= \mathsf{E}\left[f(X_1,\ldots,X_n)\right]\mathsf{Pr}(\sum Y_i \geq m)
\end{aligned}$$

It is easy to check that $\mathsf{Pr}(\sum Y_i \geq m)$ can be bounded below by 1/4, and equation (11) follows. The case where $\mathsf{E}\left[f(X_1,\ldots,X_n)\right]$ decreases with m is similar. ■

From this theorem, we derive a corollary that will be central to most of our proofs. Let us call the scenario in which bin loads are taken to be independent Poisson random variables with mean $\frac{m}{n}$ the *Poisson case*, and the scenario where m balls are thrown into n bins independently and uniformly at random the *exact case*. Also, let a *load based event* be an event that depends solely on the loads of the bins. More precisely, a load based event is determined by a 0/1 indicator function on the variables $X_1,\ldots,X_n$.

Corollary 13 *A load based event that takes place with probability p in the Poisson case takes place with probability at most $p\sqrt{2\pi e m}$ in the exact case. If the probability of the event is monotonically increasing or decreasing with the total number of balls, then the probability of the event is at most $4p$ in the exact case.*

Proof: Let f be the indicator function of the load based event. In this case $\mathsf{E}[f]$ is just the probability that the event occurs, and the result follows immediately from Theorem 12. ■

To demonstrate the utility of this corollary, we provide a simple representative example that will prove useful later.

Lemma 14 *Suppose $m < \frac{n}{\log n}$, and suppose m balls are thrown independently and uniformly at random into n bins. Then, with high probability, the maximum load is $\Theta(\frac{\log n}{\log \frac{n}{m}})$.*

Proof: By Corollary 13, since the maximum load is monotonically increasing with the number of balls, it is sufficient to prove that the bounds hold in the Poisson case. Let p_k be the probability that any particular bin contains k or more balls.

For the lower bound, note that

$$p_k \quad \geq \quad \frac{(\frac{m}{n})^k \mathrm{e}^{-m/n}}{k!},$$

as the right hand side is simply the probability that a bin has exactly k balls. The probability that no bin has k or more balls is thus at most $(1-p_k)^n \leq \mathrm{e}^{-p_k n}$, and we need to show that $\mathrm{e}^{-p_k n} \leq \frac{1}{n}$ when $k = \Omega(\frac{\log n}{\log \frac{n}{m}})$. Taking logarithms twice yields the following sufficient condition:

$$\log k! + k \log(\tfrac{n}{m}) \quad \leq \quad \log n - O(\log \log n). \tag{12}$$

It is now simple to check that choosing $k = \frac{a \log n}{\log \frac{n}{m}}$ for any constant $a < \frac{1}{2}$ suffices as long as $m < \frac{n}{\log n}$.

For the upper bound, note that

$$p_k \quad \leq \quad \frac{2(\frac{m}{n})^k \mathrm{e}^{-m/n}}{k!}, \tag{13}$$

as can be found by bounding the probability that a bin has k or more balls by a geometric series (and using just that $m < n$). It is easy to show that when $k \geq \frac{3 \log n}{\log \frac{n}{m}}$, this probability is less than $\frac{1}{n^2}$, and thus no bin contains $\frac{3 \log n}{\log \frac{n}{m}}$ or more balls with probability at least $1 - O(1/n)$ in the exact case. ■

We emphasize that Corollary 13 will prove useful to us because in the Poisson case all bin loads are independent. This independence allows us to use various forms of Chernoff bounds in the Poisson case, and then transfer the result to the exact case, greatly simplifying the analysis.

4 Parallel GREEDY

The lower bounds in Section 2 show that if the number of communication rounds and possible destinations for a ball are fixed, the $\frac{\log \log n}{\log d} + O(1)$ maximum load bound of for GREEDY shown in [2] no longer applies. We therefore seek ways to parallelize the GREEDY strategy, with the aim of matching the lower bounds. We first deal with the case of two rounds in Section 4.1, and then consider multiple rounds in Section 4.2. For these sections, we restrict ourselves to the case $d \geq 2$.

4.1 A two round parallelization of GREEDY

We begin with a more formal description of GREEDY. Each ball a will at some point in the algorithm independently and uniformly at random *choose* d destination bins $i_1(a), i_2(a), \ldots i_d(a)$ (with replacement). We may assume that these choices are made in parallel as the first step in the algorithm; this assumption makes it clear that GREEDY is non-adaptive. Next, each ball a decides, solely by communicating with $i_1(a), \ldots, i_d(a)$, to which bin it shall *commit*. Once a ball has committed to a bin, its decision cannot be reversed. We note that ties in this and other algorithms are broken arbitrarily and the d bin choices are made with replacement unless stated otherwise.

```
CHOOSE(d):
  in parallel: each ball $a$
    chooses u.a.r. $d$ bins $i_1(a), \dots, i_d(a)$
```

```
GREEDY($d$):
  call CHOOSE(d)
  sequentially: each ball $a$
    queries bins $i_1(a), \dots, i_d(a)$ for current load
    commits to bin with smallest load
```

To parallelize the GREEDY algorithm, we let the balls choose between $i_1(a), \dots, i_d(a)$ according to the load on a bin as computed using the number of balls that sent a *request* to it in **CHOOSE(d)**, not the number that has actually *committed* to occupy it. Let all the balls inform their d choices that they have been chosen by sending them each a *request*. We shall refer to the d requests as *siblings*.

Each bin then creates a list of the requests it has received. The bins may order their lists arbitrarily. However, if they handle requests in the order they arrive, the algorithm may function asynchronously. Notice that we make no claim that the requests arrive at the bins in any particular order.

The *height* of a request is its position in the request list it belongs to. The bins now send back the heights of their requests to the balls. Finally, each ball commits to the bin in which its request had the smallest height. This allows the entire process, which we call PGREEDY, to finish in only two rounds:

```
PGREEDY($d$):
  call CHOOSE(d)
  in parallel: each ball $a$
    sends requests to bins $i_1(a), \dots, i_d(a)$
  in parallel: each bin $i$
    creates list of received requests
    sends heights to requesting balls
  in parallel: each ball $a$
    commits to bin with smallest height
```

Note that Corollary 9 provides a lower bound for the PGREEDY strategy, showing that the parallelization cannot possibly maintain the load balance of GREEDY. We now prove a matching upper bound on the maximum load achieved by PGREEDY.

Theorem 15 *For fixed d and $m = n$, the maximum load achieved by* PGREEDY(d) *is at most* $O\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ *with high probability.*

Proof: As in Theorems 7 and 9, although we are considering only the case $m = n$, we maintain the distinction between m and n when writing the proof.

The outline of the proof is as follows: consider any bin i, and consider all balls with requests of height larger than some T_1 in bin i. For such a ball to choose bin i, all of its siblings' requests must have height at least T_1, and hence they must all have landed in bins that received at least T_1 requests. By choosing T_1 large enough, we can make the probability that a request at bin i of height at least T_1 chooses bin i very small, and thereby bound the number of balls that choose bin i.

We begin with some technical details. First, there may be balls that have one or more siblings choose bin i. The expected number of such balls is $O\left(\frac{md^2}{n^2}\right)$; as $m = n$ and d is fixed, with high probability the number of such balls is bounded by a constant. We can therefore absorb these balls in the $O(1)$ term and ignore them in the remainder of the proof. Second, we choose a bound M such that with high probability, bin i receives no more than M requests. (For example, in this case, we may take $M = O(\log n)$.) Conditioned on both these high probability events occurring, the set R of requests sent to a bin other than i are distributed in the remaining $n - 1$ bins independently and uniformly.

Consider all requests in i of height at least T_1, all of whose siblings lie outside i; call this set of requests I. We now find a suitable T_2 such that, with sufficiently high probability, fewer than T_2 elements of I have siblings whose heights are all T_1 or more. This will show that (with high probability) at most T_2 elements of I commit to bin i. Choosing T_1 and T_2 to be $\Theta\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ will then complete the proof.

Consider the subprocess of requests R arriving at the bins other than i. We can imagine these requests arriving sequentially at the bins according to some arbitrary ordering. Let time t be the instant immediately after the t'th such request arrives. Let $\mathcal{E}_t$ be the event that, at time t, no more than N bins have received more than T_1 requests from R, for some N to be determined later. Also, let the random variable X_t equal 1 if the height of the t'th request is greater than T_1, and 0 otherwise. Now, for $r \in I$, let $S(r)$ be the set of arrival times for the siblings of r, and let the random variable Y_r equal 1 if, for every $t \in S(r)$, $X_t = 1$ and $\mathcal{E}_t$ occurs; Y_r is 0 otherwise. That is, $Y_r = 1$ if and only if all the siblings of r are of height at least T_1, but the number of bins of height T_1 has not become higher than N before all siblings of r arrive.

We define $\mathcal{E}$ to be the event that $\mathcal{E}_t$ is true for all t. Conditioned on $\mathcal{E}$, we have that $\sum_{r \in R} Y_r$ is an upper bound on the number of balls with requests of height at least T_1 at bin i that choose bin i as their final destination. Note that $\mathsf{Pr}(Y_r = 1) \le \left(\frac{N}{n}\right)^{d-1}$. It follows that the sum of any subset of the Y_i is stochastically dominated by the sum of the same number of independent Bernoulli variables with parameter $\left(\frac{N}{n}\right)^{d-1}$ by Lemma 1. Now we choose an N so that $\mathcal{E}$ is true with high probability. In the Poisson case, the probability that a bin has T_1 requests is $\frac{\mathrm{e}^{-md/n}(md/n)^{T_1}}{T_1!}$. As long as $T_1 > 2md/n$, then the probability that a bin has at least T_1 requests is at most $\frac{2\mathrm{e}^{-md/n}(md/n)^{T_1}}{T_1!}$. Applying Chernoff bounds (Lemma 2, equation (1)), with high probability the number of bins with at least T_1 requests is at most $N = \frac{4n\mathrm{e}^{-md/n}(md/n)^{T_1}}{T_1!}$ in the Poisson case. Since the number of bins with at least T_1 requests is monotonically increasing in the number of requests, the same is true in the exact case as well by Corollary 13.

We use the bound on N to bound the number of balls with requests of height at least T_1 in i whose siblings all have height at least T_1. Again, using Chernoff bounds (Lemma 2, equation (1)), we have

$$\mathsf{Pr}\left[\sum_{r\in R} Y_r \geq T_2\right] \quad \leq \quad \left(\mathrm{e}M\left(\frac{N}{n}\right)^{d-1}\right)^{T_2}.$$

We want the probability on the left to be at most, say, $O(\frac{1}{n^c})$ for some constant $c \geq 1$. Hence we require

$$\left[\mathrm{e}M\left(\frac{N}{n}\right)^{d-1}\right]^{T_2} \quad \leq \quad \frac{1}{n^c}.$$

We now take logarithms of both sides and remove lower order terms. Note that as $M = O(\log n)$ its contribution is only a lower order term. Simplifying yields:

$$T_2\left(T_1\log T_1 - T_1\log\frac{md}{n}\right) \quad \geq \quad \frac{c\log n}{d-1}. \qquad (14)$$

For $m = n$, we may choose $T_1 = T_2 = \Theta\left(\sqrt{\frac{\log n}{\log\log n}}\right)$, and the result follows. ■

One would be inclined to believe that increasing d would decrease the final load. The equation (14) indicates that this is true when $m = n$ for very large n, as in this case the effect of d is to reduce the required values of T_1 and T_2 by a constant factor. In practice, however, for reasonable values of n, increasing d yields no improvement, and in fact increasing d can increase the final load. This can be explained by the term $-T_1 \log\frac{md}{n}$ in equation (14), which has a disproportionate effect when T_1 and T_2 are small. Also, the constant factor is dictated by our attempt to have the probability of failure be at most $O\left(\frac{1}{n}\right)$; if one is willing to accept slightly larger error probabilities one can improve it slightly.

4.2 Multiple round strategies

Our lower bounds suggest that with more rounds of communication, one might achieve a better load distribution. We thus suggest an alternative parallelization of GREEDY called MPGREEDY that makes use of more rounds. Although this algorithm may not be useful in practical settings, its connection to the GREEDY scheme appears interesting in its own right.

The algorithm proceeds in a number of rounds, until every ball has committed. In each round, each bin will allow at most one of its requesting balls to commit to it. If a ball receives that privilege from more than one bin, the ball commits to the bin with the smallest current load. Once a ball has committed, the bins holding the other requests are informed that they may discard those requests:

```
**MPGREEDY($d$)**:
  call CHOOSE(d)
  in parallel: each ball $a$
    chooses a random I.D. number
    sends requests with I.D. to bins $i_1(a), \ldots, i_d(a)$
  in parallel: each bin $i$
    sorts requests by I.D. number
  sequentially: repeat until all balls have committed
    in parallel: each bin $i$
      sends current load to first uncommitted ball on request list
    in parallel: each ball $a$
      if received at least one message
        commits to the bin with smallest current load
        tells bins holding other requests to discard
```

One can imagine the algorithm by picturing a *scanline* moving level by level up the request lists of the bins. When the scanline moves up to a new level, bins send messages to all the balls that the scanline has just passed through. When bins receive responses, they delete the corresponding balls in the request list above the scanline. The algorithm terminates when every request has either been passed through or deleted.

A practical disadvantage of this algorithm is that it requires synchronous rounds; the discards for each round must complete before the next round can begin. We also require a partial order on the balls, given in this case by randomly chosen I.D. numbers (chosen from a suitably large set to ensure uniqueness with high probability), to instill some notion of sequentiality.

Clearly, the maximum number of balls in any bin upon completion is bounded above by the number of rounds taken to finish. We analyze the latter.

Theorem 16 *With high probability* MPGREEDY(d) *finishes in at most* $\frac{\log\log n}{\log d} + 2d + O(1)$ *rounds, and hence the maximum load is also* $\frac{\log\log n}{\log d} + 2d + O(1)$.

In order to prove the above statement, we relate MPGREEDY to the following variant of GREEDY (for any $d \geq 2$): if, when placing a ball, there is a tie for the least loaded bin, then a copy of the ball is placed in each bin with the minimal load. We call this scheme GREEDY WITH TIES.

Lemma 17 *The number of communication rounds used by* MPGREEDY *is one more than the maximum load produced by* GREEDY WITH TIES *if the balls are thrown in the order given by the I.D. numbers and if the bin choices made by the balls are the same for both trials.*

Proof: Consider a modification of MPGREEDY where a ball commits to all bins from which it receives a message. The number of communication rounds used by this modified version of MPGREEDY is

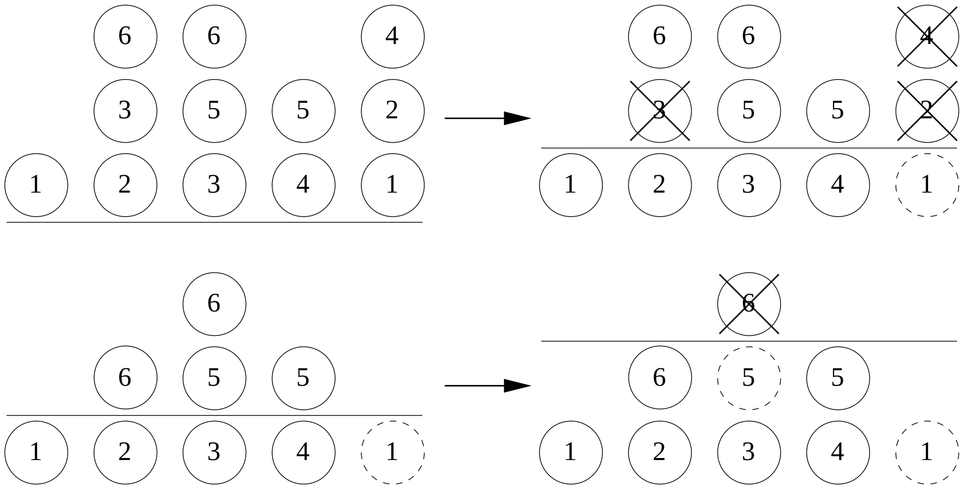

Figure 4: Comparing GREEDY WITH TIES and MPGREEDY. Each level is one round of communication. The crossed and dashed balls are discarded by the MPGREEDY process. The GREEDY WITH TIES process includes the dashed balls.

the same as for the original. With a little thought one can see that this scheme exactly mimics the GREEDY WITH TIES scheme, and hence the two methods give the same final distribution of the balls. (See Figure 4.) Since the height of the scanline moves up one level each round, the number of communication rounds used by MPGREEDY is one more than the maximum load of GREEDY WITH TIES. ■

Remark: Using ties as we have done may seem unnecessary, but it allows the scanline to be at the same height for all bins after each round. It may appear that it would improve the final maximum load if, after ties are deleted, heights in the request lists are updated to reflect the deletions. This is difficult to prove, because once heights are updated in this way, the connection between the scanline scheme and the GREEDY scheme is not readily apparent. ■

We now suggest a modification of the proof of by Azar *et. al.* (Theorem 4 of [2]) to handle the case where there may be ties. The following statement is sufficient:

Theorem 18 *The maximum load achieved by* GREEDY WITH TIES *when n balls are thrown into n bins is at most $\frac{\log \log n}{\log d} + 2d + O(1)$ with high probability. In particular, for any fixed d the maximum load is $\frac{\log \log n}{\log d} + O(1)$.*

Proof: The proof is almost entirely the same as Theorem 4 of [2]; we review the differences here, using their notation. We let h_t be the height of the tth ball, and let $\nu_i(t)$ and $\mu_i(t)$ refer to the number of bins with load at least i and the number of balls with height at least i at time t, respectively. The main consideration is that for each ball placed in the system up to d copies can be placed if ties remain.

We let $\mathcal{E}_i$ be the event that $\nu_i(n) \leq \beta_i$, but we must use a different inductive definition of β_i, As our base case, we may take $\beta_{6d^2} = n/2de$; then $\mathcal{E}_{6d^2}$ holds with certainty. We set $\beta_{i+1} = \mathrm{e}d\beta_i^d/n^{d-1}$.

For a fixed i consider a sequence of random variables Y_t where

$$Y_t \quad = \quad \begin{cases} d & \text{iff } h_t \geq i+1 \text{ and } \nu_i(t-1) \leq \beta_i, \\ 0 & \text{otherwise.} \end{cases}$$

Note that over any given set of choices for the balls before time t, $\mathsf{Pr}(Y_t = d) \leq (\beta_i/n)^d = p_i$; hence by Lemma 1

$$\mathsf{Pr}(\textstyle\sum Y_t \geq k) \quad \leq \quad \mathsf{Pr}\left[B(n, p_i) \geq \frac{k}{d}\right],$$

where $B(n,p)$ is the sum of n independent Bernoulli random variables. Conditioned on $\mathcal{E}_i$ we have $\nu_{i+1} \leq \mu_{i+1} \leq \sum Y_t$, so

$$\begin{aligned} \mathsf{Pr}(\nu_{i+1} \geq k \mid \mathcal{E}_i) \quad &\leq \quad \mathsf{Pr}(\textstyle\sum Y_t \geq k \mid \mathcal{E}_i) \\ &\leq \quad \mathsf{Pr}(B(n,p_i) \geq \frac{k}{d} \mid \mathcal{E}_i) \\ &\leq \quad \frac{\mathsf{Pr}(B(n,p_i) \geq k/d)}{\mathsf{Pr}[\mathcal{E}_i]} \end{aligned}$$

The proof now proceeds along the same lines as that of Theorem 4 of [2]. This shows that the maximum load $\frac{\log\log n}{\log d} + 6d^2 + O(1)$. We can improve this by taking a different base case: for $d > 8$, $\beta_{2d} = n/2de$ holds with high probability even if all balls are simply placed into d random bins, and hence we can start the induction from this point instead. ■

Theorem 16 follows immediately from Lemma 17 and Theorem 18. Moreover, an extension to the case where d grows with n is interesting.

Corollary 19 *When* MPGREEDY *is run with $d = \frac{\log\log n}{\log\log\log n} + O(1)$, the number of rounds and maximum load are at most $O(\frac{\log\log n}{\log\log\log n})$ with high probability.* ■

Theorem 18 demonstrates that one can match the performance of GREEDY using only $\frac{\log\log n}{\log d} + 2d + O(1)$ rounds of communication, rather than the obvious n rounds. Corollary 19 also matches the lower bound of Corollary 11, up to constant factors.

It is an open question whether one can extend MPGREEDY to avoid the need for the partial order on the balls or the synchronous rounds while achieving a similar maximum load. Stemann, using a different algorithm, also achieves a maximum load as good as the MPGREEDY algorithm [21]. This algorithm is also not completely asynchronous, although it seems to require weaker synchrony than MPGREEDY.

5 The threshold strategy

We now examine another strategy, previously exploited in [7] and [14] in similar contexts, to achieve good load balancing. Given a threshold T, we imagine throwing the balls over r rounds. If more than T balls enter a bin during a round, the excess balls are rethrown in the next round. We wish to set T as small as possible while ensuring that, with high probability, at most T balls are thrown into any bin in the rth round. Then, after all r rounds, the fullest bin will contain at most rT balls. Note that a ball can choose its bins for all r rounds before any messages are sent, so this scheme again falls into the general model of Section 2 for which our lower bounds apply.

There are several advantages this method has over the PGREEDY strategy already presented. First, this method can work in completely asynchronous environments: as long as a request includes the number of its current round as part of the message, messages from distinct rounds can be handled simultaneously at the bins. Secondly, balls send and receive at most one message per round. Finally, as we shall show, this method demonstrates a potentially useful tradeoff between the maximum load and the number of rounds.

```
THRESHOLD($T$):
  while there exists a ball that has not been accepted
    in parallel: each unaccepted ball $a$
      chooses u.a.r. a bin $i(a)$
      sends a request to $i(a)$
    in parallel: each bin $i$
      chooses $\min\{T, \#\text{received}\}$ requests from current round
      sends these balls acceptances
      sends other requesting balls rejections
```

The question is how to set the parameter T so that the procedure terminates with high probability within some specified number of rounds. In Section 5.1, we show how to set T for any constant number of rounds. We then show in Section 5.2 that, when $T = 1$, THRESHOLD(T) takes at most $O(\log\log n)$ rounds and has maximum load $\Omega(\log\log n)$ with high probability. Our proofs demonstrate essential techniques that can be used for a wide range of T and r values to emonstrate that a pair (T,r) leads to an algorithm that terminates wtih high probability.

5.1 Thresholds with a fixed number of rounds

Theorem 20 *If r is fixed independent of n, then* THRESHOLD(T) *terminates after r rounds with high probability, where $T = O\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$.*

Proof: Let k_i be the number of balls to be (re)thrown after i rounds ($k_0 = n$). We will show by induction that

$$k_i \;\; \leq \;\; n\left(\frac{4\log n}{T!}\right)^{\frac{T^i-1}{T-1}} \tag{15}$$

for all $i \leq r-1$ with high probability. From this statement one may verify that for constants r and $0 < \epsilon < 1$, and suitably large n, $T = O\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$ suffices to reduce k_{r-1} to less than $n^{1-\epsilon}$. We may then conclude that only one more round is necessary by applying Lemma 14 with $m = n^{1-\epsilon}$.

We now inductively prove equation (15). The case $i = 0$ is readily verified. Now consider the situation when k_i balls are thrown into n bins in the $(i+1)$-st round. It can be verified from equation (15) that for large enough n, $k_i/n < T$ for all $i \in \{0, \ldots, r\}$. We can thus apply the Poisson approximation and Corollary 13 to obtain that, with high probability, in the $(i+1)$-st round,

$$\mathsf{Pr}(\text{a given bin receives} > T \text{ requests}) \;\; \leq \;\; \frac{2e^{-k_i/n}(k_i/n)^T}{T!}. \tag{16}$$

Therefore (via the Chernoff bounds of Lemma 2) with high probability the number of bins with more than T requests is at most $\frac{4ne^{-k_i/n}(k_i/n)^T}{T!}$. We can make the conservative upper bound assumption that with probability exponentially close to one, none of these over-full bins has more than $\log n$ requests, so that with high probability,

$$k_{i+1} \;\; \leq \;\; n\left(\frac{4\log n}{T!}\right)^{\frac{T(T^i-1)}{T-1}+1}. \tag{17}$$

Equation (15) now follows by induction, as long as the number of rethrows is large enough so that the Chernoff bound holds. This immediately implies a maximum load of $O(rT)$, which, for fixed r, is $O\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$. ■

The theorem suggests that using the threshold strategy, one can successfully trade load balance for communication time in a well-defined manner. We note that one can also show that for $T = \Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$, THRESHOLD$(T)$ requires more than r rounds with high probability. We also remark that we could show that Theorem 20 holds with very high probability; that is, the probability of failure is bounded above by $1/f(n)$ where $f(n)$ is a superpolynomial function. This requires more attention to the Chernoff bounds.

5.2 The case of $T = 1$

We can extend our argument to the case where r grows with n with a bit more care. As an illustrative example, we consider the case where $T = 1$. We note that more powerful results are given in [14] and [16], but the simple proofs below are appealing.

Theorem 21 THRESHOLD(1) *terminates after at most* $\log\log n + O(1)$ *stages with high probability.*

Proof: As in the proof of Theorem 20, let k_i be the number of balls to be thrown after round i. It is simple to show by Chernoff bounds (equation (1)) that, with high probability, after only two rounds at most $n/2\mathrm{e}$ balls remain to be thrown. We claim that, as long as k_{i+1} is at least $4\sqrt{n \log n}$, $k_{i+1} \leq \mathrm{e}k_i^2/n$ with probability $1 - O(1/n^2)$. For convenience we assume that in each round the balls arrive in some arbitrary order, with the first ball that arrives at a bin being accepted. Let X_j be the indicator random variable of the event that the jth ball falls into a non-empty bin, where $1 \leq j \leq k_i$. Note that $\mathsf{Pr}(X_j = 1 \mid X_1, \ldots, X_{j-1}) \leq k_i/n$. It follows from Lemma 1 that the sum of the k_i random variables X_j is stochastically dominated by the sum of k_i independent Bernoulli random variables with parameter k_i/n. Using Chernoff bounds (equation (1)) the above claim follows; the restriction that k_{i+1} is at least $4\sqrt{n \log n}$ is required for the Chernoff bound to hold with sufficient probability. We thus have, if $i \geq 2$ and $k_i \geq 4\sqrt{n \log n}$, that

$$k_i \quad \leq \quad \frac{n}{\mathrm{e}2^{2^{i-2}}}.$$

Hence $r = \log\log n + O(1)$ rounds will suffice to cut down k_r to below $4\sqrt{n \log n}$ with high probability. By using the Poisson case to bound the number of bins that receive more than one ball, one can show that only $O(1)$ more rounds will be needed after this point, and the result follows. ■

The strategy THRESHOLD(1) achieves a maximum load that is essentially the same as GREEDY, but uses only $O(\log\log n)$ asynchronous rounds instead of $O(n)$ synchronous rounds. Moreover, because of its simplicity, we expect that this strategy may be the best choice when the GREEDY strategy does not apply. Because the bound of Theorem 21 does not match the lower bound of Corollary 11. one might hope that THRESHOLD(1) actually performs even better than the given upper bound. This could happen in one of two ways: either THRESHOLD(1) might terminate in fewer than $\Omega(\log\log n)$ rounds, or even if $\Omega(\log\log n)$ rounds are required, perhaps no bin actually receives $\Omega(\log\log n)$ balls. We will now show, however, that the bound of Theorem 21 is tight, up to constant factors.

Theorem 22 *The maximum load of* THRESHOLD(1) *is at least* $\Omega(\log\log n)$ *with high probability.*

Proof: As before, let k_i be the number of balls to be thrown in round i, with $k_0 = n$. We can determine k_{i+1} by considering the number of bins that receive two or more balls in the ith round. In the Poisson case, the probability that a bin receives two balls in round i is $\mathrm{e}^{-k_i/n}\frac{k_i^2}{2n^2} \geq \frac{k_i^2}{2\mathrm{e}n^2}$. By equation (2) of Lemma 2 and Corollary 13, as long as $k_i > 10\sqrt{n \log n}$, then with probability at least $1 - O(1/n^2)$, $k_{i+1} \geq \frac{k_i^2}{4\mathrm{e}n}$. Hence, for all $i \leq n$ with $k_i > 10\sqrt{n \log n}$,

$$k_{i+1} \quad \geq \quad \left(\frac{1}{4\mathrm{e}n}\right)^{2^{i+1}-1} k_0^{2^{i+1}} \quad = \quad \frac{4\mathrm{e}n}{(4\mathrm{e})^{2^{i+1}}}. \tag{18}$$

It is easy to check from equation (18) that we need $i = \Omega(\log\log n)$ before $k_i \leq 10\sqrt{n \log n}$. We now show that with high probability, there will be at least one bin that receives a ball in each of the first

$\Omega(\log\log n)$ rounds. Say that a bin *survives* up to round i if it gets a ball in each of rounds $1,\ldots,i$, and let s_i be the number of bins that survive up to round i. Then

$$\mathsf{Pr}\Big[\text{bin survives up to } i+1 \;\Big|\; \text{it survives up to } i\Big] = 1-\left(1-\frac{1}{n}\right)^{k_i} \geq \frac{k_i}{2n},$$

where the last inequality holds since $k_i \leq n$. Applying Chernoff's bound (equation (2)) tells us that the fraction of bins that survived round i that also survive round $i+1$ is at least $\frac{k_i}{4n}$ with probability at least $1-O(\frac{1}{n^2})$ as long as s_i is sufficiently large. Therefore, after the $(i+1)$-st round, with high probability the number of surviving bins is at least

$$\begin{aligned} s_{i+1} &\geq n \times \frac{k_0}{4n} \times \cdots \times \frac{k_i}{4n} \\ &> \frac{n\mathrm{e}^{i+1}}{(4\mathrm{e})^{2^{i+1}}}. \end{aligned}$$

It remains to be checked that for $i = \Omega(\log\log n)$ all the Chernoff bounds will hold, and thus with high probability there is still a surviving bin. ■

6 Simulation results

It is important to note that in the balls and bins scenario, even if each ball just chooses one bin independently and uniformly at random, the maximum load is very small compared to the total number of bins. Thus, even though one may be able to show that asymptotically one strategy performs better than another, it is worthwhile to test actual performance. For example, it is not clear from the results we have described that GREEDY performs better than straightforward random selection unless n is exceedingly large! (In fact, for all values of n, the expected maximum load of GREEDY is less than that of simple random selection; see [2] for more details.) Even if one can guarantee better performance, however, a system designer interested in using a load balancing scheme must balance the tradeoff between the maximum load and the complexity of the underlying algorithm. Asymptotic notation proves less helpful than specific numbers in understanding this tradeoff. We therefore examine actual performance through some simulation results.

For simplicity, we here consider only the case where the numbers of balls and bins are equal. As usual, d represents the number of bins to which each ball sends requests. The numbers given in the table represent the maximum load found for one hundred trials of each strategy.

The first thing worth noting is that GREEDY performs noticeably better than simple one-choice randomization: even at just one million balls, the difference is at least a factor of two in all of our trials. A second interesting feature of GREEDY is that the maximum load appears to vary little across trials, suggesting that the maximum load is sharply concentrated on a single value. This can in fact be shown using straightforward martingale arguments. (See, for example, [19, Section 4.2].)

As expected, both PGREEDY and THRESHOLD(T) perform somewhere between simple random selection and GREEDY. Notice that for PGREEDY when $d = 3$ the maximum load tends to be smaller than when

$d = 2$, but that the maximum load tends to increase when $d = 4$. This is not completely surprising given our previous analysis in Section 4.

In the threshold schemes, the thresholds used were as follows: 3 balls per round per bin in the 2 round scheme, and 2 balls per bin per round in the 3 round scheme. These choices were made to ensure that the algorithm terminated with all balls having a final destination in the correct number of rounds with high probability: in all trials, the algorithm terminated in the correct number of rounds. Our simulations suggest that threshold schemes are the best practical choice when one wishes to achieve a better load balance, but cannot meet the sequentiality requirement of GREEDY.

Balls	One	GREEDY			PGREEDY			THRESHOLD(T)	
n	Choice	$d = 2$	$d = 3$	$d = 4$	$d = 2$	$d = 3$	$d = 4$	2 rnds.	3 rnds.
1 m.	**8** . . 28 **9** . . 57 **10** . . 13 **11** . . 2	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 92 **6** . . 8	**5** . . 95 **6** . . 5	**5** . . 77 **6** . . 23	**5** . . 88 **6** . . 12	**4** . . 77 **5** . . 23
2 m.	**8** . . 7 **9** . . 72 **10** . . 18 **11** . . 3	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 90 **6** . . 10	**5** . . 96 **6** . . 4	**5** . . 68 **6** . . 32	**5** . . 74 **6** . . 26	**4** . . 69 **5** . . 31
4 m.	**8** . . 1 **9** . . 63 **10** . . 35 **12** . . 1	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 71 **6** . . 29	**5** . . 87 **6** . . 13	**5** . . 36 **6** . . 64	**5** . . 54 **6** . . 46	**4** . . 47 **5** . . 53
8 m.	**9** . . 40 **10** . . 58 **11** . . 1 **12** . . 1	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 55 **6** . . 45	**5** . . 71 **6** . . 29	**5** . . 6 **6** . . 94	**5** . . 20 **6** . . 80	**4** . . 19 **5** . . 81
16 m.	**9** . . 21 **10** . . 62 **11** . . 16 **16** . . 1	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 31 **6** . . 69	**5** . . 48 **6** . . 52	**5** . . 1 **6** . . 99	**5** . . 5 **6** . . 95	**4** . . 1 **5** . . 99
32 m.	**9** . . 5 **10** . . 65 **11** . . 25 **12** . . 5	**4** . . 100	**3** . . 100	**3** . . 100	**5** . . 8 **6** . . 92	**5** . . 20 **6** . . 80	**6** . . 100	**6** . . 100	**5** . . 100

Table 1: Simulation results for GREEDY and other strategies. The number of balls ranges from one million to thirty-two million. The results from 100 trials are presented: the load is given in bold on the left, and the number of times (out of 100) that load occurred is given on the right (e.g., "...35").

7 Conclusion

In this paper we examined various mechanisms to balance load in a decentralized manner, using the abstraction of a balls and bins occupancy problem. We specifically explored the trade-off between the number of communication rounds invested and the remaining load imbalance. Existing work defined the two extremes in this trade-off, namely, $O(\log n/\log\log n)$ imbalance after one round, and $O(\log\log n)$ after n sequential rounds.

We have demonstrated that, in our models, the sequentiality requirement of the GREEDY algorithm is in fact necessary for the strong $O(\log\log n)$ result they obtained. In light of our lower bounds, we looked for related algorithms that perform well, both in theory and in practice. Both our lower and upper bounds are strikingly uniform for all of them, despite their apparent differences. Our analysis together with our simulations suggest that the THRESHOLD(T) algorithm is the best choice when sequentiality cannot be maintained and imbalance smaller than a single random choice is desired.

Acknowledgements. We thank Claire Kenyon and Orli Waarts for suggesting the random graph model, pointing us to [1], and several helpful related ideas. We thank Michael Luby and Alistair Sinclair for many comments and suggestions.

References

[1] M. Atjai, J. Aspnes, M. Naor, Y. Rabani, L. Schulman, and O. Waarts. Fairness in scheduling. In *Proceedings of the Sixth Annual ACM/SIAM Symposium on Discrete Algorithms*, pages 477–485, 1995.

[2] Y. Azar, A. Broder, A. Karlin, and E. Upfal. Balanced allocations. In *Proceedings of the 26th ACM Symposium on the Theory of Computing*, pages 593–602, 1994.

[3] A. D. Barbour, L. Holst, and S. Janson. *Poisson Approximation.* Oxford University Press, 1992.

[4] P. Berenbrink, F. Meyer auf der Heide, and K. Schröder. Allocating weighted balls in parallel. In *Proceedings of the Ninth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 302–310, 1997.

[5] B. Bollobás. *Random Graphs.* Academic Press, London, 1985.

[6] A. Broder, A. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proceedings of the 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 322–330, 1993.

[7] A. Broder and A. Karlin. Multi-level adaptive hashing. In *Proceedings of the 1st ACM/SIAM Symposium on Discrete Algorithms*, 1990.

[8] A. Czumaj and V. Stemann. Randomized allocation processes. In *Proceedings of the Thirty-eighth Annual IEEE Symposium on Foundations of Computer Science*, pages 194–203, 1997.

[9] M. Dietzfelbinger and F. Meyer auf der Heide. Simple, efficient shared memory simulations. In *Proceedings of the Fifth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 110–119, 1993.

[10] G. Gonnet. Expected length of the longest probe sequence in hash code searching. *Journal of the ACM*, 28(2):289–304, April 1981.

[11] T. Hagerup and C. Rüb. A guided tour of Chernoff bounds. *Information Processing Letters*, 33:305–308, February 1990.

[12] N. Johnson and S. Kotz. *Urn Models and Their Application.* John Wiley and Sons, 1977.

[13] A. Kamath, R. Motwani, K. Palem, and P. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, 1994.

[14] R. M. Karp, M. Luby, and F. Meyer auf der Heide. Efficient PRAM simulation on a distributed memory machine. In *Proceedings of the 24th ACM Symposium on the Theory of Computing*, pages 318–326, 1992.

[15] V. F. Kolchin, B. A. Sevsat'yanov, and V. P. Chistyakov. *Random Allocations.* V. H. Winston & Sons, 1978.

[16] P. D. MacKenzie, C. G. Plaxton, and R. Rajaraman. On contention resolution protocols and associated probabilistic phenomena. Department of Computer Science TR-94-06, University of Texas at Austin, April 1994. An extended abstract appears in STOC 1994.

[17] F. Meyer auf der Heide, C. Scheideler, and V. Stemann. Exploiting storage redundancy to speed up randomized shared memory simulations. In *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science*, pages 267–278, 1995.

[18] M. Mitzenmacher. Density dependent jump Markov processes and applications to load balancing. In *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science*, 1996.

[19] M. Mitzenmacher. *The Power of Two Choices in Randomized Load Balancing.* PhD thesis, University of California, Berkeley, September 1996.

[20] V. Stemann. *Contention Resolution Protocols in Hashing Based Shared Memory Simulations.* PhD thesis, University of Paderborn, 1995.

[21] V. Stemann. Parallel balanced allocations. In *Proceedings of the Eighth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 261–269, 1996.