Parallel Randomized Load Balancing
(Preliminary Version)

Micah Adler[*] Soumen Chakrabarti[†] Michael Mitzenmacher[‡] Lars Rasmussen[§]
Computer Science Division
University of California, Berkeley, CA 94720
{micah,soumen,mitzen,larsr}@cs.berkeley.edu

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. Recently, Azar *et al.* analyzed the following: randomly choose d bins for each ball, and then sequentially place each ball in the least full of its chosen bins [2]. They show that 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 in asynchronous environments.

[*]Supported by a Schlumberger Foundation graduate fellowship.
[†]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.
[‡]Supported by the Office of Naval Research.
[§]Supported by a fellowship from U.C. Berkeley.

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 [7, 10] with numerous applications in hashing [2, 5, 9], PRAM simulation [9, 11] and load balancing [2, 5].

Recently, an important extension of this result was proven by Azar *et al.* [2]. Suppose we adopt the following strategy: we place the balls sequentially, one at a time; for each ball, we choose two bins independenly and uniformly at random, and place the ball in the less full bin. When all the balls have been placed, the fullest bin contains only $\Theta(\log\log n)$ balls with high probability, an exponential improvement over the simple randomized approach.

Unfortunately, the new method 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 used in [9] and [11] 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 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. (For simplicity, we shall concentrate on the case $m = n$. Extension to general values will appear in the full paper.) 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, in parallel, messages to any prospective bin, and in the second stage each bin is able to send, in parallel, messages 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 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 include 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. Also, the obvious strategy of having the balls choose random I.D. numbers and applying standard sorting methods requires $\Omega(\log n)$ rounds in this model, as well as more sophisticated communication.

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. During some round at least $\Omega(\frac{n}{\log n})$ messages from balls that have not previously communicated must be handled. 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 Our results

In §2, we provide a general lower bound for non-adaptive and symmetric strategies that include parallel variations of GREEDY [2] and threshold methods [9, 11]. 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 $\Omega\left(\sqrt[r]{\frac{\log n}{\log\log n}}\right)$. The lower bounds are 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 tool, based on results of Gonnet [6]. In analyzing complex random processes, the use of heuristic approximations through normal or Poisson distributions is common. We apply this notion systematically to the scenario of a number of balls being thrown independently and uniformly at random into some number of bins. Apart from enabling us to prove our bounds, the tool may be of independent interest.

In §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 §5 we explore an entirely different paradigm based on thresholds, which were also used in [5, 9, 11].

We demonstrate algorithms based on thresholds that asymptotically match the lower bounds for any fixed number of rounds r up to a constant factor; 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.

2 Lower bounds using edge orientation

We first develop a general model for lower bounds that captures a class of non-adaptive, symmetric load balancing strategies. Recall that for non-adaptive, symmetric strategies, the destinations are chosen independently and uniformly at random before communication begins. Our lower bounds are based on the number of rounds of communication, r, and the number of choices available to each ball, d. In §2.1, we will focus on the case where $d = 2$ and $r = 2$, extending the results to arbitrary values of r and d in §2.2.

For our bounds, we will rephrase the balls and bins problem in terms of a graph orientation problem similar to that found in [1]. We temporarily restrict ourselves to the case of $d = 2$. Associate with each bin a vertex of a graph. 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[1]. Choosing a final destination is equivalent to choosing an orientation for the edge. The goal of the algorithm is to minimize the maximum indegree over all vertices of the graph. In the case where there are n balls and n bins, the corresponding graph is a random graph from $\mathcal{G}_{n,n}$, the set of all graphs with n vertices and n edges. Following standard terminology, we define the *neighbors* 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 neighbors of a vertex v, denoted by $N(v)$, is the set of all edges incident to v.

Definition 2.1 *The* r-neighborhood *of an edge e, denoted by $N_r(e)$, is defined inductively by: $N_1(e) = N(e)$, $N_r(e) = N(N_{r-1}(e))$.*

Definition 2.2 *The* (r,x)-neighborhood *of an edge $e = (x,y)$, denoted by $N_{r,x}(e)$, is defined inductively by: $N_{1,x}(e) = N(x) - \{e\}$, $N_{r,x}(e) = N(N_{r-1,x}(e)) - \{e\}$.*

[1]For convenience, we assume here that balls choose distinct bins; that is, the graph has no self-loops. The analysis is similar if self-loops are allowed.

Intuitively, for each round of communication, a ball discovers a little more about the graph. Specifically, since we are working towards lower bounds, we may assume that the bins transport all available information about the balls 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; this follows from a simple induction argument.

We now describe an assumption that we use to show that the final load is high with constant probability. The r-neighborhood of a ball $e = (x,y)$ splits into two subgraphs corresponding to $N_{r,x}(e)$ and $N_{r,y}(e)$; these are the parts of the neighborhood the ball discovers from each bin. Suppose that these two subgraphs of the ball's r-neighborhood are isomorphic rooted trees, with the roots being x and y. In this case we say the ball has a *symmetric r-neighborhood.* Then 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 1/2; we shall expand on this shortly.

Assumption 2.3 *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.*

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

We now show that, with constant probability, there exists a vertex with at least $T = \Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ incident edges such that each incident edge has a symmetric one-neighborhood. Thus, with at least constant probability, at least $T/2$ of these edges orient themselves to the vertex, and hence with constant probability, any two-round parallel algorithm for balancing balls and bins in this model must end with a final load at least $\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$. The vertex in question will be the root of a specific tree component in the graph.

Definition 2.4 *A* (T,r) tree *is a depth r tree, each of whose internal vertices has degree T and each of whose leaves is at depth r. A (T,r) tree is said to be* isolated *in a graph G if it is a connected component of G.*

Lemma 2.5 *If there is an isolated $(T,2)$ tree in the graph determined by randomly throwing balls into bins, then the probability that each ball incident to the root directs itself to the root is 1/2.*

PROOF. This follows since each edge incident to the root of the $(T,2)$ tree has a symmetric one-neighborhood. ■

Theorem 2.6 *There exists a $T = \Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ such that with constant probability, a random graph from $\mathcal{G}_{n,n}$ contains an isolated $(T,2)$ tree.*

We restrict ourselves to isolated trees in order to simplify the proof. Note that it would be sufficient for the graph to contain a (T, r) tree with further edges adjacent to the leaves; restricting ourselves to isolated trees, however, only affects lower order terms in the analysis.

PROOF. 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 variable that is 1 if v_0 is the root of an isolated (T, r) 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 [4], equation (3) of I.1):

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

The multinomial coefficient $\binom{n}{1;T;T-1;\ldots;T-1}$ represents 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 child. We now choose a specific $\vec{v}$ and determine the probability 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. Routine calculations give the probability of this event as:

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

Using linearity of expectation, we have

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

This unwieldy expression can be simplified by canceling appropriately and noting that we will choose T small enough so that many terms are $o(1)$. For example,

$$\frac{\binom{n-(T^2+1)}{2}^{n-T^2}}{\binom{n}{2}^{n-T^2}} \quad = \quad e^{-2(T^2+1)}(1 + o(1)).$$

We thus obtain:

$$\mathsf{E}[X] \quad = \quad \frac{n 2^{T^2}(1 + o(1))}{T! e^{2(T^2+1)}((T-1)!)^T}.$$

We now examine how to compute $\mathsf{E}[X^2]$. Note that, because we are considering only isolated (T, r) 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

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

it suffices then to compute the second term. The calculation is similar to that for $\mathsf{E}[X]$. Thus, with essentially the same argument as above, one finds that

$$\sum_{\vec{v} \neq \vec{w}} \mathsf{E}[X_v X_w] \quad = \quad \frac{n^2 2^{2T^2}(1 + o(1))}{(T!)^2 e^{4T^2+4}((T-1)!)^{2T}}.$$

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. One can thus check that there exists a $(T, 2)$ tree with $T = (\sqrt{2} - o(1))\sqrt{\frac{\log n}{\log\log n}}$ with constant probability. ■

Corollary 2.7 *Any non-adaptive, symmetric load distribution strategy for the balls and bins problem satisfying Assumption 2.3, where $d = 2$ and $r = 2$, has a final load at least $(\sqrt{2}/2 - o(1))\sqrt{\frac{\log n}{\log\log n}}$ with at least constant probability.*

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 easily be modified for the case where the balls are ranked at the bins by some fixed ordering as well 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, r) tree whose root has the given final load.

2.2 The general case

One can extend the proof to the case where $d > 2$ and $r > 2$; in fact, the extension applies if r and d grow sufficiently slowly with n as well.

When $r > 2$, the balls and bins scenario can again be reduced to a graph orientation problem; instead of showing the existence of a $(T, 2)$ tree, one needs to the existence of a (T, r) tree. The proof that such a tree exists is similar to that of Theorem 2.6.

When $d > 2$ we must consider hypergraphs instead of graphs. In this reduction, 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, and the depth of a tree is the the number of hyperedges in the longest path from the root to a leaf.

Definition 2.8 *A* (T, r, d) tree *is a depth r tree of hyperedges of size d, each of whose internal vertices has degree T and each of whose leaves is at depth r. A (T, r, d) tree is said to be* isolated *in a hypergraph G if G contains a subgraph that is a (T, r, d) tree and there are no other hyperedges incident to the (T, r, d) tree in the hypergraph.*

The r-neighborhood and (r, x)-neighborhood of a ball can be defined for hypergraphs similar to Definitions 2.1 and 2.2. As in Assumption 2.3, 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 2.3. Thus the root of an isolated (T, r, d) tree will end with T/d balls with at least constant probability. The important feature in our calculations is essentially the size of the (T, r, d) tree. As long as the tree size is approximately $\frac{\log n}{\log \log n}$, a suitable (T, r, d) tree will exist.

Theorem 2.9 *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 edges of size d contains an isolated (T, r, d) tree.*

Proof. The proof will appear in the full version of the paper; it requires a combinatorial calculation entirely similar to that of Theorem 2.6. ∎

Corollary 2.10 *Any non-adaptive, symmetric load distribution strategy for the balls and bins problem satisfying Assumption 2.3 where d and r are constants has a final load at least $\Omega(\sqrt[r]{\frac{\log n}{\log\log n}})$ with constant probability.*

The constants in the lower bound (for r and d fixed) are dependent on d. The theorem can also be used when d grows with n; with constant probability the final load is 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 $d^T = o(n)$.

3 The Poisson approximation

We now derive a tool that will be useful in developing upper bounds. 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 formalize this relationship by adapting an argument used by Gonnet [6] to determine the expected maximum number of balls in a bin. While useful tail bounds on the distributions of balls in bins can be found with other methods, most notably martingales [8, 9], our method appears to be more general, and in some cases easier to apply. Although tighter probability bounds for specific problems can often be obtained with more detailed analyses, as can be seen for example in [3], for our purposes this simple approach is quite effective. As mentioned in [4], similar ideas have been used in the study of random graphs to relate the setting where each edge is included independently with some probability and the setting where a graph with a certain number of edges is chosen randomly.

Theorem 3.1 *Suppose m balls are thrown into n bins independently and uniformly at random, and let X_i be the number of balls in the ith bin, where $1 \le i \le n$. Let $Y_1, \ldots, Y_n$ be independent Poisson random variables with mean $\frac{m}{n}$, and let $f(x_1, \ldots, x_n)$ be a non-negative function. Then*

$$\mathsf{E}[f(X_1, \ldots, X_n)] \quad\le\quad \sqrt{2\pi e m}\,\mathsf{E}[f(Y_1, \ldots, Y_n)]. \qquad (1)$$

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

$$\mathsf{E}[f(X_1, \ldots, X_n)] \quad\le\quad c\,\mathsf{E}[f(Y_1, \ldots, Y_n)] \qquad (2)$$

for some constant c.

Proof. We have that

$$\begin{aligned} &\mathsf{E}[f(Y_1, \ldots, Y_n)] \\ &\quad= \sum_{k=0}^{\infty} \mathsf{E}\left[f(Y_1, \ldots, Y_n)\middle|\sum Y_i = k\right] \mathsf{Pr}[\textstyle\sum Y_i = k] \\ &\quad\ge \mathsf{E}\left[f(Y_1, \ldots, Y_n)\middle|\sum Y_i = m\right] \mathsf{Pr}[\textstyle\sum Y_i = m] \\ &\quad= \mathsf{E}[f(X_1, \ldots, X_n)]\frac{m^m e^{-m}}{m!}, \end{aligned}$$

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, and that $\sum Y_i$ is Poisson distributed with mean m. Using Stirling's approximation now yields equation (1).

If $\mathsf{E}\left[f(X_1, \ldots, X_n)\right]$ increases with m, then by a similar argument we have

$$\begin{aligned} &\mathsf{E}[f(Y_1, \ldots, Y_n)] \\ &\quad\ge \mathsf{E}\left[f(Y_1, \ldots, Y_n)\middle|\sum Y_i = m\right] \mathsf{Pr}[\textstyle\sum Y_i \ge m] \\ &\quad= \mathsf{E}\left[f(X_1, \ldots, X_n)\right] \mathsf{Pr}\left[\textstyle\sum Y_i \ge m\right] \end{aligned}$$

Since $\mathsf{Pr}\left[\sum Y_i \ge m\right]$ can be bounded above by a constant, equation (2) 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.

Corollary 3.2 *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 cp in the exact case for some constant c.*

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 3.1. ∎

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

Lemma 3.3 *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 at least $\Omega(\frac{\log n}{\log \frac{n}{m}})$ and at most $O(\frac{\log n}{\log \frac{n}{m}})$.*

PROOF. By Corollary 3.2 it is sufficient to prove that the bounds hold in the Poisson case. Let p be the probability that any particular bin contains T or more balls.

For the lower bound, note that

$$p \geq \frac{(\frac{m}{n})^T e^{-m/n}}{T!},$$

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

$$\log T! + T\log(\tfrac{n}{m}) \quad \leq \quad \log n - O(\log\log n). \qquad (3)$$

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

For the upper bound, note that

$$p \leq \frac{2(\frac{m}{n})^T e^{-m/n}}{T!}, \qquad (4)$$

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

Corollary 3.2 will also 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 (such as those in [4], section I.3) in the Poisson case, and then transfer the result to the exact case.

4 Parallel GREEDY

The lower bounds in the previous section show that if the number of communication rounds and possible destinations for a ball are fixed, the $\log\log n/\log d + O(1)$ maximum load bound of [2] no longer applies. We therefore seek ways to parallelize the GREEDY strategy and gauge their performance. We first deal with the case of two rounds in §4.1, and then consider multiple rounds in §4.2.

4.1 A two-round parallelization of GREEDY

We note that in the GREEDY strategy, all balls can choose their random bins efficiently in parallel, but the rest of the protocol is sequential. In this section, we consider strategies that allow for the entire protocol to be performed efficiently in parallel. We first consider the case where a ball makes only two destination choices, *i.e.* $d = 2$. We begin with a description of GREEDY. Each ball a will at some point in the algorithm independently *choose* two destination bins $i_1(a)$ and $i_2(a)$. We may assume that these choices are made in parallel as the first step in the algorithm; this assumption clarifies that GREEDY is non-adaptive. Next, each ball a decides, solely by communicating with $i_1(a)$ and $i_2(a)$, to which of the two bins 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 unless stated otherwise.

```
CHOOSE(2):
  in parallel: each ball $a$
    chooses u.a.r. two bins $i_1(a)$ and $i_2(a)$
```

```
GREEDY:
  call CHOOSE(2)
  sequentially: each ball $a$
    queries bins $i_1(a)$ and $i_2(a)$ for current load
    commits to bin with smaller load
```

We first attempt to break the sequentiality of GREEDY by letting the balls choose between $i_1(a)$ and $i_2(a)$ according to the selections made by the other balls in the *initial* stage of the process. Let all the balls inform $i_1(a)$ and $i_2(a)$ of their choices by sending them both a *request*. We shall refer to the two requests as *siblings*.

Each bin then creates a list of the requests it has received. The bins may order the list 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 smaller height. This allows the entire process to finish in only two rounds:

```
PGREEDY:
  call CHOOSE(2)
  in parallel: each ball $a$
    sends requests to bins $i_1(a)$ and $i_2(a)$
  in parallel: each bin $i$
    creates list of received requests
    sends heights to requesting balls
  in parallel: each ball $a$
    commits to bin with smaller height
```

Note that Corollary 2.7 provides a lower bound for the PGREEDY strategy. We now prove an upper bound on the maximum load achieved by PGREEDY.

Theorem 4.1 *The maximum load achieved by* PGREEDY *is at most* $(4 + o(1))\sqrt{\frac{\log n}{\log\log n}}$ *with high probability.*

PROOF. We bound the probability that a specific bin i receives more than $2T$ balls, where T is to be determined. Consider a bin i with more than T requests. The probability that more than 3 balls sent both requests to i or that the total number of requests received by i is more than $\log n$ is at most $O(\frac{1}{n^2})$, so we condition on the event that neither is the case. The set R of requests sent to a bin other than i are distributed in the remaining $n-1$ bins independently and uniformly.

Consider a request in i of height at least T whose sibling lies outside i. Let $S \subset R$ be the set of siblings of such requests. We prove that, with sufficiently high probability, fewer than T requests in S have height T or more.

Consider the subprocess of requests R arriving to 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.

We now use an innovation from [2]. Let $N = \frac{n2^T}{eT!}$ and $\mathcal{E}_t$ be the event that, at time t, no more than N bins have received more than T requests from R. Also, let the random variable X_t equal 1 if the height of the t'th request is greater than T, and 0 otherwise. Finally, let the random variable Y_t equal 1 if $X_t = 1$ and $\mathcal{E}_t$ occurs, and 0 otherwise.

We define $\mathcal{E}$ to be the event that $\mathcal{E}_t$ is true for all t. Conditioned on $\mathcal{E}$, we have that $\sum_{t \in S} Y_t$ is an upper bound on the number of balls of height at least T that choose bin i as their final destination.

Note that $\Pr[Y_t = 1 \mid Y_1, \ldots, Y_{t-1}] < \frac{N}{n}$. It follows that the sum of a subset of the Y_i is stochastically dominated by the sum of the same number of independent Bernoulli variables with parameter $\frac{N}{n}$. Therefore, using the Chernoff bound, we have for $T = (2+o(1))\sqrt{\frac{\log n}{\log\log n}}$:

$$\Pr\left[\sum_{t \in S} Y_t \geq T\right] \leq \left(\frac{2^T \log n}{T \cdot T!}\right)^T \leq O(\frac{1}{n^2}).$$

We can bound $\Pr[\neg\mathcal{E}]$ since R consists of at most $2n - T$ requests uniformly distributed over $n-1$ bins. It is easy to show $\Pr[\neg\mathcal{E}] = O(\frac{1}{n^2})$ by Corollary 3.2 and Chernoff's bound. Thus

$$\Pr\left[\sum_{t \in S} Y_t \geq T \bigvee \neg\mathcal{E}\right] = O(\tfrac{1}{n^2}).$$

It follows that for each bin i, the probability that T balls of height greater than T choose bin i is $O(\frac{1}{n^2})$. Hence with high probability all bins must finish with at most $2T$ balls. ∎

The proof can be easily modified to the case where balls have more than two siblings as well; for fixed d, the final load will still be $O\left(\sqrt{\frac{\log n}{\log\log n}}\right)$, but the constant factor in the O-expression is $\frac{4}{\sqrt{d-1}} + o(1)$. In practice, however, for reasonable values of n, increasing d does not improve the final load. Informally, each ball receives more pieces of information, but each piece is less valuable since the height becomes a less accurate estimate of the final position. Also, the constant factor is dictated by our attempt to have the probability of failure be at most $O(\frac{1}{n})$; if one is willing to accept slightly larger error probabilities one can improve the constant factor slightly.

4.2 Multiple round strategies

Our lower bounds suggest that with more rounds of communication, one might achieve a better load balance. We thus suggest an alternative parallelization of GREEDY called MPGREEDY that makes use of more rounds. We first examine the case where $d = 2$.

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

```
MPGREEDY:
  call CHOOSE(2)
  in parallel: each ball $a$
    chooses a random I.D. number
    sends requests with I.D. to bins $i_1(a)$ and $i_2(a)$
  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 smaller current load
        tells bin holding other request to discard
```

One can imagine the algorithm by picturing a *scanline* moving level by level through 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.

One 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. However, a significant advantage is that all the communication paths required are determined by the initial choices of two bins made by the balls. This may be useful in practice in cases where there is a cost associated with modifying the communication pattern during the course of the algorithm, as in distributed networks.

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 4.2 *With high probability* MPGREEDY *finishes in at most* $\log\log n + O(1)$ *rounds.*

In order to prove the above statement, we consider the following variation of GREEDY (for any d): if 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 4.3 *The number of communication rounds used by* MPGREEDY *is one more than the maximum load given by* GREEDY WITH TIES *when the balls are thrown in the order given by the I.D. numbers and the bin choices made by the balls are the same for both trials.*

PROOF. Consider a modification of MPGREEDY where the ball commits to all bins from which it receives a message. The number of communication rounds used by this modified version of MPGREEDY is 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. Since the height of the scanline moves up one level each round, the number of communication rounds used by MPGREEDY is hence one more than the maximum load of GREEDY WITH TIES. ■

We now suggest a modification of the proof given in Azar *et al.* to handle the case where there may be ties. The following statement is sufficient:

Theorem 4.4 *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]. The main difference is that for each ball placed in the system up to d copies can be placed if ties remain. This problem can be handled by taking some care in the base cases. In the notation of Theorem 4 of [2], one can set $\beta_{6d^2} = n/2de$; for $d > 8$, one can show by Chernoff's bounds that setting $\beta_{2d} = n/2de$ works with sufficiently high probability for the argument to follow. ■

Theorem 4.2 follows immediately. Moreover, an extension to the case where d grows with n is interesting.

Corollary 4.5 *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 4.4 demonstrates that one can match the performance of GREEDY at the expense of $\frac{\log\log n}{\log d} + 2d + O(1)$ rounds of communication. As we shall see, Corollary 4.5 also implies that, in the case where $d = \frac{\log\log n}{\log\log\log n} + O(1)$, MPGREEDY performs better than the threshold strategy discussed in the next section.

It is open whether one can extend MPGREEDY to avoid the partial order on the balls or the synchronous rounds while achieving similar results.

5 Threshold strategy

We now examine another strategy, previously exploited in [5, 9, 11] 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. 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 the 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 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. Secondly, balls send and receive at most one message per round. Finally, we shall show that 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 up to $T$ requests from current round
      sends these balls acceptances
      sends other balls in this round 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 §5.1, we show how to set T for any constant number of rethrowing rounds. We then show in §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 the essential techniques to derive the relationship between T and r for any values of T and r.

A variation on this strategy would allow a bin to hold up to kT balls after k rounds for all k, instead of limiting the bin to T balls per round. We choose to analyze the latter approach because the proofs appear more straightforward. We also remark that we could show that the bounds we present hold 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, and the results will appear in the full version.

5.1 Thresholds with a fixed number of rounds

Theorem 5.1 *For $T = \sqrt[r]{\frac{(2r+o(1))\log n}{\log\log n}}$* THRESHOLD$(T)$ *terminates after r rounds with high probability.*

Proof. We begin with the proof when $r = 2$. We bound the number of rethrows after the first round by using the Poisson case. The probability that a bin contains more than T balls is at most $\frac{2}{eT!}$, as can be seen by bounding the probability by a suitable geometric series. Thus, with high probability, the number of bins with more than T balls is at most $\frac{2n}{T!}$. We also have that with probability exponentially close to 1 that no bin contains more than $\log n$ balls. We now assume that this is the case; formally, we can condition the event that no bin has more than $\log n$ balls, and all previous statements still hold. Thus the total number of rethrows in the Poisson case is at most $\frac{2n \log n}{T!}$ with high probability, and the same holds in the exact case by Corollary 3.2, as the expected number of rethrows is an increasing function in the number of balls thrown.

Now consider the second round. Using equation (4) from the proof of Lemma 3.3, we have the probability that a specific bin receives more than T balls in the second round is at most $2(2\log n)^T/(T!)^{T+1}$ in the Poisson case. This expression is $O(1/n^2)$ for the given value of T and as in Lemma 3.3 the result follows.

We now consider when $r > 2$. We have shown that after one round the number of balls that need to be rethrown is certainly less than $\frac{4n \log n}{T!}$. Let k_i be the number of balls that have to be rethrown after i rounds. Following entirely the same argument, one can show inductively that

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

with high probability for any fixed i and large enough n. Now consider the final round. By equation (4) of the proof of Lemma 3.3, the probability that a bin receives more than T balls on the rth round is at most $2(k_{r-1}/n)^T/(T!)$, which is $O(1/n^2)$ for the given value of T. The result follows. ■

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 directly show that for $T = \sqrt[r]{\frac{(r-o(1))\log n}{\log\log n}}$, THRESHOLD($T$) requires more than r rounds with high probability in a similar matter.

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. We consider the case where $T = 1$. The following results are similar to those in [9] and [11], but the simple proofs below are appealing.

Theorem 5.2 THRESHOLD(1) *terminates after at most* $\log\log n + O(1)$ *stages with high probability.*

Proof. Again let k_i be the number of balls to be thrown after stage i. We first claim that, as long as k_{i+1} is at least $4\sqrt{n \log n}$, $k_{i+1} \leq ek_i^2/n$ with probability $1 - O(1/n^2)$. For convenience we assume the balls arrive in some arbitrary order, with the first ball that arrives at a bin being accepted. Let X_j be the event that the jth ball falls into a non-empty bin, where $1 \leq j \leq k_i$. Note that $\Pr[X_j = 1 \mid X_1, \ldots, X_{j-1}] \leq k_i/n$. It follows 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 the claim follows. We thus have:

$$k_i \quad \leq \quad \frac{e^{(2^i-1)}}{n^{2^i-1}} {k_0}^{2^i}.$$

By picking $k_0 = n/2e$, $r = \log\log n$ 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. It is simple to show by Chernoff bounds that only a constant number of rounds are required before only k_0 balls remain to be thrown, and the result follows. ■

Theorem 5.3 *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 thrown in round i, but let $k_0 = n$. We can determine the number of balls thrown in the ith round by considering the number of bins that receive two or more balls in the ith round. Using the Poisson case and Chernoff bounds, we find that as long as $k_i > 10\sqrt{n \log n}$, then with probability at least $1 - O(1/n^2)$,

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

It is easy to check that we need $i = \Omega(\log\log n)$ before $k_i < 10\sqrt{n\log n}$. 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 $0, \ldots, i$, and let s_i be the number of bins that survive up to round i. Then

$$\begin{aligned} &\Pr\left[\text{bin survives up to } i+1 \,\middle|\, \text{it survives up to } i\right] \\ &\quad = \quad 1 - \left(1 - \frac{1}{n}\right)^{k_i} \geq \frac{k_i}{2n} \end{aligned}$$

where the last inequality holds since $k_i \leq n$. Applying Chernoff's bound 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 over $1 - O(\frac{1}{n^2})$ 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

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

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. ■

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.

Balls	Simple	GREEDY			PGREEDY			THRESHOLD(T)		
n	Random	$d = 2$	$d = 3$	$d = 5$	$d = 2$	$d = 3$	$d = 5$	2 rounds	3 rounds	5 rounds
10^6	8–11	4	3	2–3	5–6	5–6	5–6	5–6	4–5	4
$5 \cdot 10^6$	9–12	4	3	3	5–6	5–6	6–7	5–6	4–5	4
10^7	9–12	4	3	3	5–6	5–6	6–7	5–6	4–5	4
$5 \cdot 10^7$	9–12	4	3	3	5–6	5–6	6–7	6	5	4

Table 1: Simulation results.

6 Simulation Results

An important feature of these load balancing schemes is that the maximum load, even using the simplest randomization, 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. We thus briefly describe some simulation results.

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 ranges for the maximum load found after between fifty and one hundred trials for each strategy.

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 is the same as when $d = 2$, and that the maximum load increases when $d = 5$; this is not completely surprising given our previous analysis. Also, we note that the thresholds were not optimized for the threshold strategy; in practice one might want to take care to optimize the threshold for a given number of balls.

7 Conclusion

We have demonstrated lower bounds for simple parallel load distribution strategies in a distributed setting, and also found simple strategies that match the lower bounds within a constant factor. Our results show the tradeoff between the final load and the number of rounds of communication required. Directions for future work include looking at the case where each ball has an associated weight, and the goal is to minimize the maximum weight over all the bins after distribution. Also, it would be interesting to see how useful the general paradigms we employ are in the case where the underlying communication network is restricted, so that balls can only communicate with certain processors.

Acknowledgments

We would like to thank the numerous people at U.C. Berkeley and the International Computer Science Institute who offered suggestions and improvements to previous drafts of the paper. We thank Michael Luby and Alistair Sinclair for many helpful suggestions, and Claire Kenyon and Orli Waarts for leading us to examine the graph model.

References

[1] Miklos Atjai, James Aspnes, Moni Naor, Yuval Rabani, Leonard J. Schulman, and Orli Waarts. Fairness in scheduling. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algortihms*, pages 477–485, 1995.

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

[3] A.D Barbour, Lars Holst, and Svante Janson. *Poisson Approximation.* Oxford Science Publications, 1992.

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

[5] A. Broder and A. Karlin. Multi-level adaptive hashing. In *Proceedings of the 1st ACM/SIAM Symposium on Discrete Algortihms*, pages 43–53, 1990.

[6] G. Gonnet. Expected length of the longest probe sequence in hash code searching. *Journal of the ACM*, 28(2):289–304, April 1991.

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

[8] A. Kamath, R. Motwani, K. Palem, and P. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. In *Proceedings of the 24th IEEE Symposium on Foundations of Computer Science*, pages 592–603, 1994.

[9] R. 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 Theory of Computing*, pages 318–326, 1992.

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

[11] P.D. MacKenzie, C.G. Plaxton, and R. Rajaraman. On contention resolution protocols and associated probabilistic phenomena. Department of Computer Science Technial Report TR-94-06, University of Texas at Austin,, April 1994.