Multiple-choice Balanced Allocation in (almost) Parallel*

Petra Berenbrink[1], Artur Czumaj[2], Matthias Englert[2],
Tom Friedetzky[3], and Lars Nagel[4]

[1] School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada.
petra@sfu.ca
[2] DIMAP and Department of Computer Science, University of Warwick, UK.
{A.Czumaj, M.Englert}@warwick.ac.uk
[3] School of Engineering and Computing Sciences, Durham University, Durham, UK.
tom.friedetzky@dur.ac.uk
[4] Zentrum für Datenverarbeitung, Johannes Gutenberg Universität Mainz, Germany.
nagell@uni-mainz.de

Abstract. We consider the problem of resource allocation in a parallel environment where new incoming resources are arriving online in groups or *batches*.
We study this scenario in an abstract framework of allocating balls into bins. We revisit the allocation algorithm GREEDY[2] due to Azar, Broder, Karlin, and Upfal (*SIAM J. Comput.* 1999), in which, for sequentially arriving balls, each ball chooses two bins at random, and gets placed into one of those two bins with minimum load. The maximum load of any bin after the last ball is allocated by GREEDY[2] is well understood, as is, indeed, the entire load distribution, for a wide range of settings. The main goal of our paper is to study balls and bins allocation processes in a parallel environment with the balls arriving in *batches*. In our model, m balls arrive in batches of size n each (with n being also equal to the number of bins), and the balls in each batch are to be distributed among the bins simultaneously. In this setting, we consider an algorithm that uses GREEDY[2] for all balls within a given batch, the answers to those balls' load queries are with respect to the bin loads at the end of the previous batch, and do not in any way depend on decisions made by other balls from the same batch.
Our main contribution is a tight analysis of the new process allocating balls in batches: we show that after the allocation of *any* number of batches, the gap between maximum and minimum load is $O(\log n)$ with high probability, and is therefore independent of the number of batches used.

1 Introduction

One of the central challenges in modern distributed systems is to cope with the problem of allocating their resources effectively in a balanced way. In this paper we consider a general scenario of resource allocation in the case when new incoming resources are arriving as a *stream of batches*, and the resources from each incoming batch are to be

* Research supported by the Centre for Discrete Mathematics and its Applications (DIMAP), by EPSRC awards EP/D063191/1, EP/G069034/1, EP/F043333/1 and EP/F043333/1, and by .

allocated instantly. Although any analysis of the resource allocation protocols depends heavily on various properties of the underlying system, such as, for instance, the underlying network, service times and processing times, our focus is to study resource allocation schemes in an abstract framework of balls and bins, which is known to be able to provide important insights into more complex systems.

The framework of balls and bins is a powerful model with various applications in computer science, e.g., the analysis of hashing, the modeling of load balancing strategies, or the analysis of distributed processes. The typical aim is to find strategies that balance the balls evenly among the bins and produce small maximum loads. While traditionally mostly processes allocating balls into random bins (the single-choice scheme) have been studied (cf. [8,10]), more recently the main focus has been on the analysis of extensions of processes that let each ball choose multiple random bins instead of one, and then allocate itself to one of the chosen bins by considering their loads (see, e.g., [3,4,7,9,14,15]). As many of these papers have demonstrated, multiple-choice protocols maintain the simplicity of the original single-choice scheme, while at the same they have superior performance in many natural settings. For example, if one allocates sequentially n balls into n bins by choosing $d \geq 2$ random bins for each ball and then places the ball into the lesser loaded of the chosen bins (such a scheme will be denoted by GREEDY$[d]$), then no bin will have load greater than $\frac{\ln \ln n}{\ln d} + O(1)$ with high probability (w.h.p.[5]) [3], which compares favorably to the allocation of the balls performed i.u.r. (*independently and uniformly at random*), where the maximum load is $\Theta(\frac{\ln n}{\ln \ln n})$, w.h.p.

One major disadvantage of the multiple-choice strategies described above is that they unfold their full potential only in a sequential setting. For example, to prove the bounds for the standard multiple-choice schemes [3,4,15], it is assumed that the m balls are allocated online, one after another, and that the information about bin loads is immediately updated after allocation of each ball. These assumptions are unrealistic in various load balancing applications, e.g., where the balls model the jobs in some parallel or distributed setting and the choices of the balls must be performed independently and in parallel, or in scenarios where the balls cannot easily access the current load of the bins, for example, because of the delay in receiving this information. To cope with this, various multiple-choice strategies have been developed for parallel environments to deal with concurrent requests [1,2,11,13] and communication delays [6,7,12]. They base their decisions on the number of parallel requests, allow extra rounds of communication, and in some cases let balls re-choose.

We investigate how multi-choice schemes perform in a semi-parallel environment. In our model, a stream of m balls arrives in batches of size n each (with n being equal to the number of bins), and the loads of the bins are updated only between batches, that is, any decisions made by balls belonging to the same batch are strictly concurrent. In this setting, the algorithm uses GREEDY$[d]$, but for all balls within a given batch, the answers to those balls' load queries are with respect to the bin loads at the end of the previous batch, and do not in any way depend on decisions made by balls belonging to the same batch. We show that, for $d = 2$, after the allocation of the m^{th} ball, the gap

[5] Event $\mathcal{E}$ holds *with high probability* if $P(\mathcal{E}) \geq 1 - n^{-c}$ for any constant $c > 0$; it is important to notice that throughout the paper this bound is independent of m, the number of balls.

between the maximum and the minimum load is $O(\log n)$ with high probability, with probability at least $1 - 1/n^{O(1)}$, and is therefore independent of the number of batches.

1.1 Our model

In this paper we investigate how the bare GREEDY$[2]$ protocol performs in a semi-parallel environment in which $m \geq n$ balls are allocated into n bins. Concurrent requests to the same bin are answered with the same current load (load here means the number of balls allocated to the bin) and no additional information, like the number of new requests. We model this by updating the bins only after every n^{th} ball and show that the gap between maximum and minimum load is independent of the number of balls (and batches). With high probability, the gap is $O(\log n)$, similar to the bounds in the sequential setting [4]. Our process follows GREEDY$[2]$ [3,4], but we introduce explicit *batches* of size n, and will assume that all *balls within one batch will be allocated concurrently*. Our protocol (which we shall refer to as BGREEDY$[2]$, short for *batch greedy*) is, therefore, in some sense a mix between sequential and parallel.

When a new batch of n balls arrives, each ball has two random choices and goes into a bin of lower load. If both loads are equal, the bin with smaller ID is selected. (The use of the bins' IDs is only to break the ties; there are no restrictions for the IDs other than that they are all distinct and there is a total order defined on them.) Note that due to the batch structure of our model, bin loads are updated only after having allocated all n balls belonging to a batch.

BGREEDY$[d]$**:**

- Repeat $\frac{m}{n}$ times:
  - ▷ for each of the n new balls in a new batch, *independently in parallel* do the following:
    - ◇ choose d bins i.u.r.
    - ◇ allocate the ball into the chosen bin with the minimum load[a]; in case of tie, allocate the ball into the chosen minimum-load bin with the smallest ID

[a] The load of each bin remains unchanged for the allocation of all balls from the same batch.

Our goal is to show that after throwing m balls into n bins, the gap between the maximum load and the minimum load is at most $O(\log n)$, with high probability, independent of the number of balls. We restrict our analysis to the case $d = 2$. Indeed, experiments with larger values of d suggest that the resulting load distribution does not improve but gets slightly worse, though still being $O(\log n)$ for constant d.

1.2 Related work

There is a vast amount of literature studying the resource allocation problem modeled using the balls into bins framework. The classical processes allocating balls into random bins (the single-choice schemes) have been surveyed, e.g., in [8,10], and used in many

areas of mathematics, computer science, and engineering. The multiple-choice schemes have been used in these areas and in various settings, e.g., in adaptive load sharing [7], PRAM simulations [9], load balancing [3], and numerous follow-up papers, e.g., [1,4,5,14].

Although the multiple-choice schemes have been originally studied in the context of sequential allocation, there has also been a significant interest in its use in a parallel setting, see, e.g., [1,2,11,13]. Most known strategies involve additional rounds of communication, some are also adaptive and allow for re-choosing bins. In a typical parallel multiple-choice scheme, one aims at allocating n balls into n bins by allocating the balls with very limited coordination and using as few as possible extra communication rounds. For example, Lenzen and Wattenhofer [11] show that one can attain a maximum load of 2 using $\log^* n + O(1)$ rounds of communication, w.h.p.

The main difference between the parallel multiple-choice schemes and our model is that in our setting, the allocation of the balls from a single batch must be done instantly, without any coordination between the allocation of balls in the same batch.

Our model shares some similarities with the *bulletin board model with periodic updates*, as proposed by Mitzenmacher [12], to deal with systems with "outdated information." The model deals with the continuous process of allocating balls into bins: the balls are arriving as a Poisson stream of rate λn, $\lambda < 1$, and each bin "serves" (removes) its balls with exponential distribution with mean 1. The novel feature of the model is the access to the information about the load of the bins, which is available through a *bulletin board*, and which can contain outdated information about the load of the bins. The main variant of the model proposed by Mitzenmacher [12], *the bulletin board model with periodic updates*, assumes that the information about the load of each bin is updated periodically every T seconds, that is, for every $k \in \mathbb{N}$, to allocate the balls arriving in time interval $[kT, (k+1)T)$, the process will use the load of the bins at time kT. Mitzenmacher [12] considers three allocation mechanisms in this setting: (i) each ball chooses a bin i.u.r., (ii) each ball chooses a bin with the smallest load in the bulletin board, and (iii) each ball chooses d bins i.u.r. and is then allocated to the chosen bin with the smallest load in the bulletin board. Mitzenmacher [12] provided an analytical study for this model for the limiting case as $n \to \infty$ and supported the analytical results by simulations. The third model studied by Mitzenmacher [12] is very related to the model considered in our paper, though with several key differences. Firstly, it assumes stochastic arrivals of the balls and stochastic ball removals. Secondly, the paper only provides an analytical study in the limiting case which is supported by simulations, whereas our paper gives a rigorous probabilistic analysis.

1.3 Contributions of this paper

We analyze BGREEDY[2] in which the balls are allocated in batches of size n. We consider the scenario in which m balls are allocated into n bins, and we assume that the bins are initially empty. The allocation at *time* t is described by the load vector directly after the t^{th} batch. Our main goal is to understand the load of the bins after allocating m balls in $\frac{m}{n}$ batches for arbitrary values of m.

The main result of the paper, Theorem 3, is that after the last batch has been allocated, the load of any bin is $\frac{m}{n} \pm O(\log n)$ w.h.p. (with probability at least $1 - n^{-c}$ for any constant c). This follows from our two main technical results, Theorems 1 and 2.

We begin with Theorem 1 which studies the process under the assumption that the number of allocated balls is (relatively) small, at most polynomially large in n.

Theorem 1. *Let $\delta \geq 1$ be an arbitrary constant. Suppose that we run* BGREEDY$[2]$ *for $\tau \leq n^{\delta-1}$ batches, allocating $m \leq n^{\delta}$ many balls.*

1. For all $i \geq 0$ simultaneously, the number of bins with load at least $\frac{m}{n} + i + \gamma$ is upper bounded by ne^{-i}, w.h.p., where $\gamma = \gamma(\delta)$ denotes a suitable constant.

2. No bin has fewer than $\frac{m}{n} - O(\log n)$ balls, w.h.p.

Theorem 1 directly implies Corollary 1.

Corollary 1. *For any constant $\delta \geq 1$, if $m \leq n^{\delta}$ then the maximum load is $\frac{m}{n} + O(\log n)$ w.h.p. and the minimum load is $\frac{m}{n} - O(\log n)$ w.h.p.*

Our proof of Theorem 1 crucially relies on the assumption that m is at most polynomial in n. To deal with arbitrarily large values of m we prove Theorem 2 which removes the restriction of having to have only polynomially many balls, and reduces the problem to the case $m = \text{poly}(n)$.

Theorem 2. *Let c be a sufficiently large constant. Suppose that we run* BGREEDY$[2]$ *for $\tau \geq n^{c}$ batches. Further suppose that the maximum load is at most* MAX *and that the minimum load is at least* MIN *with probability at least $\mathfrak{p}$. Then, for any positive constant δ and any $\tau^* > \tau$, the process after running τ^* batches will have maximum load at most* MAX *and minimum load at least* MIN *with probability at least $\mathfrak{p} - n^{-\delta}$.*

By combining Theorem 2 with Corollary 1 we immediately obtain the following main theorem, which holds for any number m of allocated balls.

Theorem 3 (Main). *Fix n and m to be arbitrary integers and let c be any constant. If one allocates m balls into n bins using* BGREEDY$[2]$ *then with probability at least $1 - n^{-c}$ the maximum load is $\frac{m}{n} + O(\log n)$ and the minimum load is $\frac{m}{n} - O(\log n)$.*

Remark 1. Let us emphasize that Theorem 3 ensures that the gap between the maximum and the minimum load is $O(\log n)$ w.h.p. *at the end of the process*. It is easy to see that for large enough m no such bound can be ensured after every single batch.

The approach. On a high level, our analysis follows the approach proposed by Berenbrink *et al.* [4] (see also [14]), but there are differences when applying the line of attack from [4] to the parallel setting considered in this paper. Our analysis uses new ideas and needs to be significantly tighter in several places.

The first part of our analysis (Theorem 1 and Corollary 1, proven in Section 2) deals with the process after allocating a polynomial number of balls in the system, or equivalently, after a polynomial number of batches. That part forms the basic block of this paper, as the analysis of the general case can be reduced to it. Many ideas from [4] do not work any more once decisions have to be made based upon outdated information.

We split this (batch-wise) analysis into two sub-parts: The first provides bounds on the distribution of the underloaded bins (with load below the average), the second bounds on the distribution of the overloaded bins (with load above the average). Whereas the analysis of the underloaded bins follows the one in [4] rather closely, the analysis of the overloaded bins requires several new ideas. The basic approach used in [4], the layered induction, cannot (easily) be applied because of the large number of new balls allocated in parallel in each single round. Instead, using the fact that the probability for a bin to receive a ball does not change within a batch, we base our analysis on an appropriate bound on the expected number of new balls for each bin.

The second part of the analysis is related to the infinite process (the number of batches is arbitrarily large) and is formalized by the so-called Short Memory Theorem. It states that, informally, if we run the process for a long time, then the behavior of the load of the bins is essentially determined only by a small number of the most recent batches. With that, one can reduce the analysis for an arbitrary number of batches to the case in the first part, that is, to the case when the number of batches is only polynomially small. The proof of the Short Memory Theorem uses similar coupling arguments as the approach initiated in [4] (cf. also [14]), but the need to cope with parallel allocations for the same batch makes the arguments more involved.

Further discussion. Our analysis shows that even in a parallel environment, where the tasks from the same batch are to be allocated concurrently, the idea of using multiple-choices for the allocation leads to a significant improvement in the performance of system. Indeed, if we used BGREEDY$[1]$ instead of BGREEDY$[2]$, that is, if all balls were allocated at random, then it is a folklore result that for $m \geq n \log n$ the gap between the maximum and the minimum load is $\Theta(\sqrt{m \log n/n})$, w.h.p. Thus, our result shows that despite the lack of any coordination between the allocation of balls in a single batch, the use of a multiple-choice allocation scheme can improve the performance of system as compared to the naive approach of fully random allocations (BGREEDY$[1]$).

Our result in Theorem 3 provides further evidence that even in systems with outdated information, by carefully choosing the allocation rules (multiple-choice allocation scheme), one can obtain a very balanced load allocation.

Let us also mention that our analysis is tight in the sense that for large enough m the gap between the maximum and the minimum load in BGREEDY$[2]$ is $\Omega(\log n)$, w.h.p.

2 Polynomially many balls (Theorem 1)

Our analysis for the case of polynomially many balls follows the outline of the proof of [4]. We will show two invariants, one for the underloaded and one for the overloaded bins. The underloaded bins are analyzed in Section 2.2, the overloaded bins in Section 2.3. Together the invariants shown in both sections imply Theorem 1.

2.1 Preliminaries

The *load* of a bin is the number of balls it contains. Assuming that balls are allocated sequentially, a ball's *height*, or *level*, is the load of the selected bin right after the allocation. Thus, one can picture the bin as a stack of balls and every new ball is simply

pushed on top of the stack. If balls arrive at the same time, then we nevertheless assume that they are added to the stack one after the other (in an arbitrary order) so that each ball has a unique height.

Fix a time step t and let m be the number of balls allocated until time step t (that is, in t batches of size n each). The average number of balls per bin at time t is $\frac{m}{n} = t$.

We call bins with fewer than t balls *underloaded* and bins with more than t balls *overloaded*. We will frequently refer to *holes* in the distribution. For a given bin, the number of holes is defined to be the number of balls it is short of the average load at that point of time.

Key invariants. Our analysis relies on the following invariants that we will prove to hold w.h.p. (for $t \leq \text{poly}(n)$):

- $L(t)$: At time t, there are at most $0.7 \cdot n$ holes.
- $H(t)$: At time t, there are at most $0.47 \cdot n$ balls of height at least $t + 5$.

Observe that since the total number of holes equals the total number of balls with height above average, invariant $L(t)$ immediately implies that there are at most $0.7 \cdot n$ balls with height $t + 1$ or larger at time t.

We will use induction on t to prove the invariants $L(t)$ and $H(t)$: we will show that if $L(0), \ldots, L(t-1)$ and $H(0), \ldots, H(t-1)$ hold, then $L(t)$ and $H(t)$ are fulfilled w.h.p. (Observe that unlike in [4], we do not need $L(t)$ to prove $H(t)$.) We will analyze the underloaded and overloaded bins separately; the corresponding analyses communicate only through the two invariants above. We will finally use invariant $H(t)$ to derive Theorem 1. Throughout the analysis, we use the following notation:

Definition 1. *For i, $t \geq 0$, we let $\alpha_i^{(t)}$ denote the fraction of bins with load at most $t - i$ at time t, and $\beta_i^{(t)}$ denote the fraction of bins with load at least $t + i$ at the same time t.*

2.2 Analysis of underloaded bins

We begin with the analysis of the load in the underloaded bins, that is, in the bins with the load below the average load. Our goal is to prove that for any t, if the invariants $L(0), \ldots, L(t-1)$ and $H(0), \ldots, H(t-1)$ hold, then $L(t)$ is fulfilled, that is, there are at most $0.7n$ holes at time t w.h.p. Our analysis follows the analysis for the underloaded bins from [4]. The details are omitted here.

Let c_1 and c_2 be suitable constants with $c_1 \leq c_2$. The idea is to prove the following two invariants (implying $L(t)$) for time $t \in [0, \text{poly}(n)]$:

- $L_1(t)$: For $1 \leq i \leq c_1 \cdot \ln n$, we have $\alpha_i^{(t)} \leq 1.6 \cdot 0.3^i$.
- $L_2(t)$: For $i \geq c_2 \cdot \ln n$, we have $\alpha_i^{(t)} = 0$.

The proofs of $L_1(t)$ and $L_2(t)$ use an "outer" induction on t and an "inner" (layered) induction on i. Note that the second invariant establishes the bound on the minimum load of Theorem 1.

2.3 Analysis of overloaded bins

In this section, we analyze the load in overloaded bins and we will prove invariant $H(t)$: there are not more than $0.47 \cdot n$ balls with height at least $t+5$ w.h.p. The proof assumes that invariant $L(t-1)$ holds, and hence that at time $t-1$ there are at most $0.7 \cdot n$ balls above the average $t-1$. Unlike our analysis in Section 2.2, this section is new and the analysis requires many new ideas compared to [4].

We will analyze invariants $H_1(t)$ and $H_2(t)$ that imply both $H(t)$ and Theorem 1. To formulate the invariants $H_1(t)$ and $H_2(t)$, we first define two auxiliary functions h and f:

Definition 2. *For any $i \geq 0$, define $h(i) = 67 \cdot 0.34^i$.*

Let ℓ denote the smallest integer i such that $h(i) \leq n^{-0.9}$ and let $\sigma \geq 1$ denote a suitable constant (that will be specified later). For $i \geq 4$, we define:

$$f(i) = \begin{cases} h(i) & \text{for } 4 \leq i < \ell \ , \\ \max\{h(i), \frac{1}{3} \cdot n^{-0.9}\} & \text{for } i = \ell \ , \\ \sigma \cdot n^{-1} & \text{for } i = \ell + 1 \ . \end{cases}$$

We use Definition 2 to set up our main invariants, $H_1(t)$ and $H_2(t)$. (Let us recall that $\beta_i^{(t)}$ denotes the fraction of bins with load at least $t+i$ at time t; see Definition 1.)

- $H_1(t)$: For $5 \leq i \leq \ell$, we have $\beta_i^{(t)} \leq f(i)$,
- $H_2(t)$: $\sum_{i>\ell} \beta_i^{(t)} \leq \sigma \cdot n^{-1}$.

$H_1(t)$ tells us that the number of balls decrease exponentially with each level. On level ℓ the fraction of balls is upper-bounded by $n^{-0.9}$. The number of balls above level ℓ can be bounded by a constant σ. The proof of the following observation follows easily from the properties of the function f.

Observation 1 *$H_1(t)$ and $H_2(t)$ imply $H(t)$.*

Observation 2 *If $L(t)$, $H_1(t)$ and $H_2(t)$ hold w.h.p. for all t, then Theorem 1 holds.*

Proof. First we show that the number of bins with load at least $\frac{m}{n} + i + 5$ is upper bounded by $n \cdot e^{-i}$: using Definition 2 and basic properties of functions f and h, we can show that for $i \geq 5$, the fraction β_i of balls on level i is upper-bounded by $h(i)$. Thus, it suffices to show that $e^{-k} \geq h(k+4)$ for $k \geq 1$:

$$1.08^k \geq 0.9 \Rightarrow e^{-k} \cdot 0.34^{-k} \geq 67 \cdot 0.34^4 \Leftrightarrow e^{-k} \geq h(k+4) = 67 \cdot 0.34^{k+4} \ .$$

It remains to prove that this upper bound holds w.h.p. for all $t \leq \frac{n^\delta}{n} = n^{\delta-1}$. This follows directly from the statement that $L(t)$, $H_1(t)$ and $H_2(t)$ hold w.h.p. for all t. □

Further details are omitted here, but the invariants H_1 and H_2 are proven by induction on t. Our induction assumptions are $H_1(0), \ldots, H_1(t-1)$, $H_2(t-1)$ and $L(t-1)$. These assumptions provide a distribution of the balls over the bins at time

$t-1$. The induction step is proven by bounding the number of additional balls for each bin w.h.p. Counting the number of additional balls is somewhat simplified by the fact that the probability for a bin to receive a ball from batch t does not depend on how many balls of batch t have been allocated before. This is because the protocol defines that the allocation of a ball depends only on the loads of the bins (immediately) before batch t.

3 Reducing to polynomially many batches (Theorem 2)

In this section we sketch the arguments used to prove Theorem 2, which shows that in order to analyze the maximum and/or minimum load after allocating m balls it is sufficient to consider the scenario when the number of balls is polynomial with respect to the number of bins, that is, $m = \mathrm{poly}(n)$.

The proof of Theorem 2 follows the approach proposed by [4] (see also [14]). The main idea (stated formally in Theorem 4 in Section 3.3) is to prove that in BGREEDY[2], if we start the process with K balls already allocated in the bins, and we then allocate another $K \cdot \mathrm{poly}(n)$ batches using BGREEDY[2], the obtained load distribution will be (in a stochastic sense) almost independent of the initial allocation of the K balls in the system. Therefore, without loss of generality, we could assume that the initial allocation started with the same number of balls in every bin, in which case the process would be identical to the one which ignored the initial K balls. This allows us to reduce the analysis of BGREEDY[2] with m balls to the analysis of BGREEDY[2] with $m' \ll m$ balls, and by applying this recursively, we can reduce the analysis of BGREEDY[2] to the case when m is not too big, namely $m = \mathrm{poly}(n)$.

3.1 Basic definitions and notation

We use the standard notation $[M] = \{1, 2, \ldots, M\}$ for any natural number M.

Load vectors and normalized load vectors. We model the allocation of balls in the bins using *load vectors*. A load vector $\mathbf{x} = (x_1, \ldots, x_n)$ specifies that the *load* of the i^{th} bin is x_i. We will consider *normalized* load vectors; a load vector $\mathbf{x}$ is *normalized* if the entries in $\mathbf{x}$ are sorted in non-increasing order, that is, $x_i \geq x_{i+1}$ for every $1 \leq i < n$. In that case, x_i denotes the number of balls in the i^{th} fullest bin. We observe that since in our analysis the order among the bins is irrelevant (apart from tie breaking according to bin IDs, which themselves are essentially arbitrary), we can restrict the state space to normalized load vectors.

Let us mention an important feature of our analysis: while the normalized load vectors are n-vectors with integer values, BGREEDY[2] resolves the ties in the load of the two chosen bin by taking the one with the smallest ID, and so the outcome of BGREEDY[2] depends on more than just the vector. However, one can always see any normalized load vector as the one in which we order the bins of the same load according to their IDs, from the largest ID to the smallest one. In view of that, the process of selecting two bins to allocate a ball according to BGREEDY[2] for a normalized load vector $\mathbf{x} = (x_1, \ldots, x_n)$ is equivalent to one of choosing two indices i^t, j^t i.u.r. and then allocating the ball into the bin corresponding to $x_{\max\{i^t, j^t\}}$.

3.2 Allocation process and Markov chains

We will model the allocation process (one step of BGREEDY[2]) by a Markov chain: if $\mathbf{X}_t$ denotes the (normalized) load vector at time t (after inserting t batches) then the stochastic process $(\mathbf{X}_t)_{t\in\mathbb{N}}$ corresponds to a Markov chain $\mathbb{MC} = (\mathbf{X}_t)_{t\in\mathbb{N}}$ whose transition probabilities are defined by our allocation process. In particular, $\mathbf{X}_t$ is a random variable obeying a probability distribution $\mathcal{L}(\mathbf{X}_t)$ defined by t steps of BGREEDY[2]. (Throughout the paper we use the standard notation to denote the probability distribution of a random variable U by $\mathcal{L}(U)$.)

Measuring similarity of distributions. We use a standard measure of discrepancy between two probability distributions ϑ and ν on a space Ω, the *variation distance*, defined as $\|\vartheta - \nu\| = \frac{1}{2}\sum_{\omega\in\Omega} |\vartheta(\omega) - \nu(\omega)| = \max_{A\subseteq\Omega}(\vartheta(A) - \nu(A))$.

3.3 Short Memory Theorem

Now we are ready to state our key result: Short Memory Theorem 4. Let us begin with some further useful terminology. For any $n, K \in \mathbb{N}$, let $\Psi_{n,K}$ be the set of all normalized load vectors $\mathbf{x} = (x_1, \ldots, x_n)$ with $\sum_{i=1}^{n} x_i = K$. That is, $\Psi_{n,K}$ is the set of all normalized load vectors that describe the system with K balls allocated to n bins.

Theorem 4 (Short Memory Theorem). *Let $K \in \mathbb{N}$ and let $\mathbf{x}$ and $\mathbf{y}$ be any two normalized load vectors in $\Psi_{n,K}$. For any t, let $\mathbf{X}_t$ ($\mathbf{Y}_t$) be the random variable describing the normalized load vector after allocating t further batches on top of $\mathbf{x}$ ($\mathbf{y}$, respectively) using* BGREEDY[2].

Then, for any $\varepsilon > 0$ there is some $\tau = O(K \cdot n + n^6 \cdot \log^2(Kn/\varepsilon))$, such that for every $T \geq \tau$, $\|\mathcal{L}(\mathbf{X}_T) - \mathcal{L}(\mathbf{Y}_T)\| \leq \varepsilon$.

One should read the claim in Theorem 4 so that if we start with any two arbitrary allocations of K balls into n bins, then after adding $T = K(n\log(1/\varepsilon))^{O(1)}$ batches to each of them, the normalized load vectors of these two systems are almost indistinguishable; they will be stochastically identical with probability at least $1 - \varepsilon$, for an arbitrary small, positive ε.

Sketch of the proof of Theorem 4. The proof of Theorem 4 uses the neighboring coupling approach initiated in [4]. We consider two normalized load vectors after allocating τ batches, $\mathbf{x}^\tau = (x_1^\tau, \ldots, x_n^\tau)$ and $\mathbf{y}^\tau = (y_1^\tau, \ldots, y_n^\tau)$ that differ by a single ball, that is, $\mathbf{x}^\tau = \mathbf{y}^\tau + \mathbf{e}_i - \mathbf{e}_j$ for $i \neq j$. (Here, for any $s \in [n]$, $\mathbf{e}_s$ will denote an n-vector consisting of a single element 1 in the coordinate s and of 0 in all other coordinates. With this notation, if $\mathbf{x}^\tau = \mathbf{y}^\tau + \mathbf{e}_i - \mathbf{e}_j$ for $i \neq j$ then $x_i = y_i + 1$, $x_j = y_j - 1$, and $x_s = y_s$ for all $s \in [n] \setminus \{i, j\}$.) For any two normalized load vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ with $\mathbf{x} = \mathbf{y} + \mathbf{e}_i - \mathbf{e}_j$ for any i, j, define $\mathbf{\Delta}(\mathbf{x}, \mathbf{y}) = \max\{|x_i - x_j|, |y_i - y_j|\}$. Note that if $\mathbf{x} = \mathbf{y}$ then $i = j$ and $\mathbf{\Delta}(\mathbf{x}, \mathbf{y}) = 0$.

We will analyze a *coupling* for the Markov chains $(\mathbf{X}_t)$ and $(\mathbf{Y}_t)$ starting with $\mathbf{x}^0$ and $\mathbf{y}^0$ differing by a single ball, where all random choices performed by $\mathbf{x}^\tau$ are identical to those performed by $\mathbf{y}^\tau$. More formally, let us first consider state $\mathbf{x}^\tau$. For

each ball in a given batch, we first choose two random numbers $i, j \in [n]$ i.u.r., then take the larger of them, say i, and then allocate the ball to the i^{th} bin in the vector $\mathbf{x}^\tau = (x_1^\tau, \ldots, x_n^\tau)$. Then, the same choice of i is used for the same ball for the vector $\mathbf{y}^\tau$. Observe that this construction uses the fact that in the normalized vector, the bins with the same load are sorted in the decreasing order of their IDs.

It is not difficult to see that *(i)* the coupling $(\mathbf{x}^\tau, \mathbf{y}^\tau) \mapsto (\mathbf{x}^{\tau+1}, \mathbf{y}^{\tau+1})$ is a proper coupling (i.e., transitions $\mathbf{x}^\tau \mapsto \mathbf{x}^{\tau+1}$ and $\mathbf{y}^\tau \mapsto \mathbf{y}^{\tau+1}$ are faithful copies of one step of BGREEDY[2]), *(ii)* if $\mathbf{x}^\tau$ and $\mathbf{y}^\tau$ differ by a single ball then either $\mathbf{x}^{\tau+1}$ and $\mathbf{y}^{\tau+1}$ differ by a single ball or $\mathbf{x}^{\tau+1} = \mathbf{y}^{\tau+1}$, and *(iii)* if $\mathbf{x}^\tau = \mathbf{y}^\tau$ then our coupling ensures that $\mathbf{x}^{\tau+1} = \mathbf{y}^{\tau+1}$. In view of these properties, our interest is in the analysis of the number of steps required until $\mathbf{x}^\tau = \mathbf{y}^\tau$. Our central result about the coupling is as follows.

Lemma 1. *Let t be any time step of the process with $\mathbf{\Delta}(\mathbf{x}^t, \mathbf{y}^t) > 0$. Then, either*

- *for some constant $c > 0$: $\mathbf{Pr}\left[\, \mathbf{\Delta}(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) = 0 \mid \mathbf{x}^t, \mathbf{y}^t \,\right] \geq \frac{c}{n^3}$, or*
- *$\mathbf{E}\big[\mathbf{\Delta}(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) \mid \mathbf{x}^t, \mathbf{y}^t, \mathbf{x}^t \neq \mathbf{y}^t\big] \leq \mathbf{\Delta}(\mathbf{x}^t, \mathbf{y}^t) - \frac{1}{n}$.*

By combining Lemma 1 with some basic analysis of random walks on a line, we can prove the following.

Lemma 2. *Let ε be any positive real. If $\mathbf{\Delta}(\mathbf{x}^0, \mathbf{y}^0) = \Delta$ then the coupling satisfies $\mathbf{Pr}\left[\, \mathbf{x}^\tau = \mathbf{y}^\tau \mid \mathbf{x}^0, \mathbf{y}^0 \,\right] \geq 1 - \varepsilon$ for some $\tau = O(\Delta \cdot n + n^6 \cdot \log^2(n/\varepsilon))$.*

As the final step, we can combine Lemma 2 with the neighboring coupling approach from [4] to conclude the proof of Theorem 4.

3.4 Using Short Memory Theorem 4 to prove Theorem 2

We are now ready to prove our key result, Theorem 2. Our approach follows the approach used in [4, Section 4] (see the discussion in [4, Remark 2, p. 1376]), and below we will briefly present the main ideas of the reduction.

Suppose that we have m batches to be allocated into n bins. We first allocate a smaller number of batches, say $m' \ll m$ batches with $m' \cdot n$ balls. Then, suppose we can show that the maximum load in any bin is at most $m' + \vartheta$ and the minimum load in any bin is at least $m' - \vartheta$, with sufficiently high probability $1 - p$, and for an appropriate value ϑ (majorization by the process of allocating all balls in random gives $\vartheta = O(\sqrt{m' \cdot \log n/p})$, see, e.g., [4]). Since the difference between the maximum and minimum load is at most 2ϑ, the distance between the load vector after allocating $m' \cdot n$ balls and the load vector in which every bin has identical load m' is at most $2\vartheta n$. Therefore, if we apply the Short Memory Theorem 4, after allocating a further ϑn^c batches for an appropriate constant c, we will have a system with $m' \cdot n + \vartheta n^{c+1}$ balls for which the distributions of the bins loads in these two processes are almost indistinguishable (w.h.p.). Hence, instead of analyzing the original process, it is sufficient to analyze the process in which we first allocate m' balls to each bin, and then allocate a further ϑn^c batches using BGREEDY[2] – but this process can completely ignore the first m' batches, because they are allocated deterministically. Therefore, we have shown that in order to analyze the process for $m = m' + \vartheta n^c$ batches, it is sufficient to analyze the same process for a smaller number of batches, for $m^* = \vartheta n^c$. As it has been

shown in detail in [4], by applying the reduction recursively with an appropriate choice of parameters, the arguments above can be easily formalized to prove Theorem 2.

References

1. M. Adler, P. Berenbrink, and K. Schröder. Analyzing an infinite parallel job allocation process. In *Proceedings of the 6th Annual European Symposium on Algorithms (ESA)*, pages 417–428, 1998.
2. M. Adler, S. Chakrabarti, M. Mitzenmacher, and L. Rasmussen. Parallel randomized load balancing. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing (STOC)*, pages 238–247, USA, 1995.
3. Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999.
4. P. Berenbrink, A. Czumaj, A. Steger, and B. Vöcking. Balanced allocations: The heavily loaded case. *SIAM Journal on Computing*, 35(6):1350–1385, 2006.
5. A. Czumaj and V. Stemann. Randomized allocation processes. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 194–203, 1997.
6. M. Dahlin. Interpreting stale load information. *IEEE Transactions on Parallel and Distributed Systems*, 11(10):1033–1047, 2000.
7. D. L. Eager, E. D. Lazowska, and J. Zahorjan. Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, 12:662–675, May 1986.
8. N. L. Johnson and S. Kotz. *Urn Models and Their Application: An Approach to Modern Discrete Probability Theory*. John Wiley & Sons, New York, 1977.
9. R.M. Karp, M. Luby, and F. Meyer auf der Heide. Efficient PRAM simulation on a distributed memory machine. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC)*, pages 318–326, 1992.
10. V. F. Kolchin, B. A. Sevast'yanov, and V. P. Chistyakov. *Random Allocations*. V. H. Winston and Sons, Washington, D.C., 1978.
11. C. Lenzen and R. Wattenhofer. Tight bounds for parallel randomized load balancing. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 11–20, 2011.
12. M. Mitzenmacher. How useful is old information? *IEEE Transactions on Parallel and Distributed Systems*, 11(1): 6–20, January 2000.
13. V. Stemann. Parallel balanced allocations. In *Proceedings of the 8th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 261–269, 1996.
14. K. Talwar and U. Wieder. Balanced allocations: The weighted case. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 256–265, 2007.
15. B. Vöcking. How asymmetry helps load balancing. *Journal of the ACM*, 50(4): 568–589, 2003.