Balanced Allocations: The Heavily Loaded Case with Deletions

Nikhil Bansal
University of Michigan

William Kuszmaul
MIT

***Abstract*—In the 2-choice allocation problem, m balls are placed into n bins, and each ball must choose between two random bins $i, j \in [n]$ that it has been assigned to. It has been known for more than two decades, that if each ball follows the GREEDY strategy (i.e., always pick the less-full bin), then the maximum load will be $m/n + O(\log\log n)$ with high probability in n (and $m/n + O(\log m)$ with high probability in m). It has remained an open question whether the same bounds hold in the *dynamic* version of the same game, where balls are inserted/deleted with no more than m balls present at a time.**

We show that, somewhat surprisingly, these bounds *do not* hold in the dynamic setting: already on 4 bins, there exists a sequence of insertions/deletions that cause the GREEDY strategy to incur a maximum load of $m/4 + \Omega(\sqrt{m})$ with probability $\Omega(1)$—this is the same bound that one gets in the single-choice allocation model where each ball is assigned to a random bin!

This raises the question of whether *any* 2-choice allocation strategy can offer a strong bound in the dynamic setting. Our second result answers this question in the affirmative: we present a new strategy, called MODULATEDGREEDY, that guarantees a maximum load of $m/n + O(\log m)$, at any given moment, with high probability in m. We also show how to generalize MODULATEDGREEDY to obtain dynamic guarantees for the $(1+\beta)$-choice setting, and for the setting of balls-and-bins on a graph.

Finally, we consider an extension of the dynamic setting in which balls can be *reinserted* after they are deleted, and where the pair i, j that a given ball uses is consistent across insertions. This seemingly small modification renders tight load balancing impossible: on 4 bins, any balls-and-bins strategy that is oblivious to the specific identities of balls being inserted/deleted *must* allow for a maximum load of $m/4 + \text{poly}(m)$ at some point in the first $\text{poly}(m)$ insertions/deletions, with high probability in m. This is a remarkable departure from the $m = n$ case where the maximum load of $O(\log\log n)$ holds independently of whether reinsertions are allowed or not.

I. INTRODUCTION

Randomized balls-into-bins processes [1], [2] serve as a useful abstraction for studying load-balancing problems, with applications such as scheduling, distributed systems, and data structures. The goal is to assign balls (e.g., tasks) to bins (e.g., machines) such that the balls are balanced as evenly as possible across the bins, where each individual ball may have only a few available random options for bins that it can be placed in.

It is well known that, if n balls are placed into n bins using the classical SINGLECHOICE rule, where each ball is placed independently in a uniformly random bin, then the maximum load is $\Theta(\log n / \log\log n)$ with probability $1 - 1/\text{poly}(n)$.

The power of 2-choices. In a seminal 1994 paper, Azar, Broder, Karlin and Upfal [3] showed that under a seemingly minor modification, where for each ball *two* bins are chosen independently and uniformly at random, and the ball is placed *greedily* in the least loaded of the two bins, the maximum load reduces to $\log\log n + O(1)$ with high probability in n. In the decades since, this *power of 2-choices* paradigm has been extremely influential, with both theoretical (e.g., [4]–[8]) and empirical (e.g., [9]–[13]) applications, and with a large literature on generalizations; see e.g., [1], [2] for some excellent surveys.

The heavily-loaded case. Azar et al.'s result [3] prompted researchers to consider the *heavily-loaded case*, where $m \gg n$ balls are inserted into n bins. The early techniques that were developed for the lightly-loaded setting (i.e., layered induction [3], witness trees [14], [15], and differential-equation approaches [16], [17]) struggled to deliver strong bounds in the heavily-loaded setting, and for several years the best known bound stood at $m/n + \log\log n + O(m/n)$ [14], [15]. If we define the *overload* to be the amount by which the maximum load exceeds m/n, then this bound allows for an overload as large as $\log\log n + O(m/n)$—such a bound is useful if $m \approx n$, but when $m \gg n \log n$, the bound becomes worse even than the standard bound offered by SINGLECHOICE (i.e., an overload of $O(\sqrt{(m/n)\log n})$).

In a breakthrough result, Berenbrink, Czumaj, Steger and Vöcking [18] showed how to use Markov-chain techniques to obtain a much stronger bound of $\log\log n + O(1)$ on the overload, with probability $1 - 1/\text{poly}(n)$.

2575-8454/22/$31.00 ©2022 IEEE
DOI 10.1109/FOCS54457.2022.00081

Thus, somewhat remarkably, the gap between the maximum and average loads in the heavily-loaded case *is the same* as in the lightly-loaded case, with high probability in n.

When $m \gg n$, the $O(\log\log n)$ overload bound does not, in general, extend to hold with probability $1 - 1/\text{poly}(m)$ (i.e., w.h.p. in the number of *balls*). However, the known techniques can be used to achieve a quite strong (and, when $n = O(1)$, optimal) bound of $O(\log m)$ on the overload in this case.

The dynamic setting. In typical load-balancing and data-structures applications, however, the items can be both inserted and deleted dynamically over time. Here two natural models have been studied: (i) the *insertion/deletion* model in which each insertion involves a new ball with independent random bin choices, and (ii) the *reinsertion/deletion* model in which a ball can be *reinserted* after being deleted, and has the same two random bin choices each time it is reinserted. In both models, deletions simply specify which previously inserted ball should be deleted. Although these two models may seem quite similar at first glance, we shall see later that the distinction is significant.

Note that, whereas in the insertion-only setting, m is set to be the total number of insertions, in the dynamic setting, m is set to be an *upper bound* on the number of balls that are present at any given moment (and the sequence of insertions/deletions may be infinite). The objective is to minimize the *overload*, which is now defined as the amount by which the maximum load exceeds m/n at any given moment.[1]

Azar et al. [3] considered the insertion/deletion model with $m = n$ and with *random deletions*: that is, n balls are inserted initially, and then there is an infinite sequence of alternating insertions/deletions, where each deletion removes a *random* ball. They showed that, at any given moment, the GREEDY strategy achieves a maximum load of $\log\log n + O(1)$, with high probability in n.

Subsequent work has considered the more general setting where the insertions/deletions are determined by an *oblivious-adversary* (i.e., an adversary that does not know the random choices of the algorithm), and where the only constraint on the adversary is that the number of balls in the system can never exceed m. Using the witness tree technique, first introduced by [19], Cole et al. [15] analyzed the reinsertion/deletion model with $m = n$, and established that the GREEDY strategy guarantees a maximum load of $O(\log\log n)$ with high probability in n. Later, Vöcking [14] improved this to $\log\log n + O(1)$, which remarkably, matches the bound in the non-dynamic (insertion-only) case up to an additive $O(1)$ term.

What about the dynamic heavily-loaded case? For more than two decades, it has remained an open question what the optimal bounds are in the *heavily-loaded case* if we wish to support both insertions and deletions performed by an oblivious adversary. Besides obvious theoretical interest, the question also arises naturally in practice—for example, as a scheduling problem in which jobs arrive and depart over time, the number of jobs (balls) at any moment is much larger than the number n of machines (bins), and the only guarantee on the arrivals/departures of jobs is an upperbound m/n on the average load at any moment.

The dynamic heavily-loaded setting was studied by Cole et al. [15] and Vöcking [14], [20], who showed that GREEDY has overload $\log\log n + O(m/n)$ with high probability in n. But again this bound is already worse for $m \gg n\log n$ than the $O(\sqrt{(m/n)\log n})$ overload bound for SINGLECHOICE (which also holds in the dynamic setting).

However, it is widely believed that GREEDY should also achieve similar bounds in the dynamic heavily-loaded case as in the non-dynamic heavily-loaded case (i.e., an overload of $O(\log\log n)$ and $O(\log m)$, w.h.p. in n and m, respectively). The current limitation would seem to be a technical one: the witness-tree techniques that allow for us to analyze dynamic games with oblivious adversaries [15], [20] are incompatible with the techniques (i.e., Markov-chain [18] and potential-function [21]–[23] arguments) that achieve strong bounds in the heavily-loaded case.

In this work we prove new upper and lower bounds for the dynamic heavily-loaded case. We split our results into two parts, the first of which considers the insertion/deletion model, and the second of which considers the reinsertion/deletion model.

A. Results in the Insertion/Deletion Model

We begin by considering the insertion/deletion model, that is, an oblivious adversary performs an arbitrary

[1] It is tempting to define the overload to be the amount by which the maximum load exceeds $m(t)/n$, where $m(t)$ is the number of balls present at time t. However, the following (folklore) example demonstrates the flaw with such a definition: Suppose we insert m balls (using an arbitrary insertion strategy), and then we delete a random $m/2$ of those balls. Since the $m/2$ deletions are random, even if the system was perfectly balanced after the initial m insertions, the bin loads will typically be $m/2n \pm \sqrt{m/2n}$, and the maximum load will be $m(t)/n + \tilde{\Theta}(\sqrt{m/n})$, which is no better than the bound trivially achieved by SINGLECHOICE.

sequence of insertions/deletions subject only to the constraint that no more than m balls are present at a time.

A lower bound for GREEDY. We show that, somewhat surprisingly, the GREEDY strategy actually *does not* offer strong bounds in the dynamic heavily-loaded setting. In particular, already for $n = 4$ bins, there exists an oblivious sequence of insertions/deletions after which there is a maximum load of

$$m/n + \Omega(\sqrt{m})$$

with probability $\Omega(1)$. In other words, the GREEDY strategy is no better than SINGLECHOICE in this setting!

Our result represents a remarkable departure from the lightly-loaded $m = n$ case, where GREEDY achieves an optimal bound of $O(\log\log n)$ (even in the *re*insertion/deletion model). The result also offers an explanation for why all previous attempts [15], [20] to analyze GREEDY for large m have yielded only relatively weak bounds.

The high-level intuition behind our lower bound is as follows. Using GREEDY, if some bin i contains far fewer balls than the other bins, then there will be a contiguous time window during which all of the insertions are maximally biased towards bin i. But this means that, later on, the adversary can perform a sequence of deletions in which the balls being *deleted* exhibit a strong bias towards being from bin i. In other words, the biases that GREEDY exhibits during insertions can be thrown back at it by future deletions.

We present the full construction in Section III. As a warmup, we first show a simpler (but already non-trivial) lower bound of $m/n + \Omega(m^{1/4})$ for $n = 4$ bins in Section III-A, and then give the full lower bound of $m/n + \Omega(m^{1/2})$ in Section III-B. For ease of exposition we mostly focus on the case of $n = 4$ — however, we also show how to use our techniques to obtain a lower bound of $m/n + m^{1/4}/\operatorname{poly}(n)$ for general n.

The MODULATEDGREEDY algorithm. Of course, the above phenomenon is not isolated to the GREEDY strategy. Any strategy that exhibits biases between bins is at risk of having those biases thrown back at it via future deletions. This raises a natural question: is it possible for *any 2-choice allocation strategy* to beat the bounds trivially achieved in the single-choice model?

Our second result is a new algorithm called MODULATEDGREEDY, in the insertion/deletion model, that at any time, with high probability in m, achieves a maximum load of

$$m/n + O(\log m).$$

This bound is optimal for any strategy that achieves high-probability bounds in m (see Section II-C).

Given the choice between two bins i and j, the MODULATEDGREEDY algorithm chooses between the bins probabilistically, based on how their loads compare. In particular, it carefully modulates its biases between bins so that the adversary is unable to find any non-trivial correlations between how balls are inserted. Interestingly, the structure of MODULATEDGREEDY also allows for a direct combinatorial analysis, which proceeds by coupling the behavior of MODULATEDGREEDY to a seemingly different (and much simpler) randomized process that we call the *stone game*.

Generalizations. Our analysis of MODULATEDGREEDY extends to support a number of generalizations and applications. This includes a tight bound of $m/n + O(\beta^{-1}\log m)$ for the $(1+\beta)$-*choice* version of the game [21], where a $(1-\beta)$-fraction of the balls are inserted using SINGLECHOICE and only a β-fraction of the balls get two choices; a bound of $m/n + \operatorname{polylog} m$ for the dynamic balls-and-bins game on an undirected well-connected regular graphs [24], [25]; and a bound of $m/n + O(\log M)$ for the setting in which m is permitted to increase over time, subject only to the constraint that $m \le M$. In all of these settings, the previous states of the art were restricted to the insertion-only model.

For brevity, we describe these results in two parts. In Section II we consider a simpler version of MODULATEDGREEDY that guarantees the $m/n + O(\log m)$ bound for insertion/deletion sequences of $\operatorname{poly}(m)$ length. Then, in the full version of the paper [26], we consider the general setting with unbounded request sequences and where m can increase over time, as well as extensions to the $(1+\beta)$-choice and the graphical 2-choice processes.

B. An Impossibility Result for the Reinsertion/Deletion Model

Finally, in the full version of the paper [26], we also consider the reinsertion/deletion model. That is, the adversary can perform an arbitrary sequence of insertions, deletions, and reinsertions (as long as the ball being reinserted is not currently present) subject only to the constraint that no more than m balls are present at a time.

Here we establish an impossibility result. Consider any 2-choice bin-allocation strategy that is *oblivious* to the specific *identities* of balls (i.e., when a ball is inserted, all that the strategy gets to see is the pair i, j of bins that the ball is assigned to). We show that, against any such strategy, it is possible for an oblivious adversary

on $n = 4$ bins to force a maximum load of $m/4 + \text{poly}(m)$ at some point in the first $\text{poly}(m)$ insertions/deletions, with high probability in m.

This result reveals a fundamental (and perhaps unexpected) gap between the insertion/deletion model and the reinsertion/deletion model. In particular, in the lightly-loaded setting with deletions where $m \leq n$, both models yield the same $O(\log\log n)$ bounds even for infinite sequences of reinsertions/deletions [15], [20]. But, in the heavily-loaded setting, the cyclic dependencies that are introduced by reinsertions (i.e., a ball x being reinserted is being placed into a system whose state has *already* been affected by x's bin choices in the past) end up being lethal to any ID-oblivious allocation strategy.

C. Other Related Work

Beyond research on the heavily-loaded and dynamic settings, there has been a large body of work on other ways to extend the 2-choice allocation framework—because the literature on this subject is so extensive, we give only a brief overview here. These extensions have included work on restricted classes of insertion strategies (e.g., $(1+\beta)$-choice strategies [21], thinning strategies [22], [27]–[29], strategies with limited information [28], etc.), on balls with nonuniform sizes [21], [23], [30], [31], on parallel settings in which balls arrive in batches [32]–[36], on settings in which bins correspond to vertices on a graph [24], [25], on settings where balls can be relocated after insertion [37], [38], etc. Another notable extension is Vöcking's asymmetric d-choice paradigm [20] which, in the lightly-loaded setting, chooses between d bins on each insertion to achieve a maximum load of $O((\log\log n)/d)$.

Another line of work, related to the current work on the dynamic setting, is on queuing models [16], [39]–[45], where insertions and deletions are *stochastic*. Many of these focus on the so-called supermarket model, introduced by [16], [39], in which customers (i.e., balls) arrive in a Poisson stream of rate λn, $\lambda < 1$, and are processed within each queue (i.e., bin) in FIFO order, where each customers requires processing time that is exponentially distributed with mean 1. In the case where λ is allowed to go to 1 (see, e.g., [43], [44]), the number of balls in the system can become $\omega(n)$ (this is analogous to the heavy case in standard balls and bins). However, because insertions/deletions are assumed to be stochastic, the analyses (and the flavors of the results) take a very different form than those in this paper (where deletions are performed by an oblivious adversary, and the number of balls in the system is deterministically bounded by a parameter m).

In addition to the past work described above, there have also been recent efforts within the succinct-data-structure literature to obtain stronger bounds for the reinsertion/deletion model in specialized regimes, resulting in a 3-choice allocation scheme that achieves a bound of $m/n + O(\log\log n) + O(\sqrt{m/n} \cdot \sqrt{\log(m/n)})$ on the maximum load at any given moment [46], [47]. This bound is useful when $m \leq O(n \log n)$, but does not improve significantly on SINGLECHOICE when $m \gg n$.

Finally, there are a number of works [29], [48]–[54] that study load-balancing problems in which slightly non-greedy behavior can out-perform more greedy approaches (either because the less-greedy approach relies less on stale information [29], [48], [49], or because the less-greedy approach benefits from randomization [50]–[54]). Our work reveals that this same theme appears somewhat unexpectedly even in the classical setting of power-of-two-choice with deletions (but, of course, for different reasons).

D. Preliminaries

In the *dynamic 2-choice allocation problem*, an oblivious adversary performs a sequence of ball insertions and deletions subject to the constraint that the number of balls in the system can never exceed m. Whenever a ball x is inserted, a uniformly random pair $h(x) = (h_1(x), h_2(x)) \in [n] \times [n]$ of distinct bins is selected, and the *insertion strategy* must choose which of the bins $h_1(x)$ or $h_2(x)$ the ball will be placed in. The pair $h(x)$ is sometimes referred to as the *hash* of the ball x.

There are two models that we will consider for insertions and deletions. In the *insertion/deletion model*, each insertion INSERT(x) places a new ball x into the system that has never been present before. In the *reinsertion/deletion model*, each insertion INSERT(x) places a ball x into the system that is not *currently* present, but that may have been present in the past (each time x is inserted, its bin pair $h(x)$ stays the same). In both models, the DELETE(x) operation selects a ball x that is currently present and removes it.

We are interested in bounding the maximum *load* (i.e., the number of balls) of any bin. Our algorithms will offer guarantees with high probability (w.h.p.) in m, meaning that the failure probability is $1/\text{poly}(m)$ for a polynomial of our choice. Two basic insertion strategies that we will discuss frequently are GREEDY, which always selects the least full of the bins $h_1(x), h_2(x)$, and SINGLECHOICE, which always selects bin $h_1(x)$.

Finally, although $h(x) = (h_1(x), h_2(x))$ is a uniformly random pair of *distinct* bins, any strategy in the insertion/deletion model can choose to view $h(x)$ as a pair of

independent bins by artificially resetting $h_2(x) = h_1(x)$ with probability $1/n$. The strategies that we design in this paper will assume (without loss of generality) that they are given a uniformly random pair of (not necessarily distinct) bins for each insertion.

II. ModulatedGreedy: Handling poly(m) Insertions/Deletions

In this section, we consider the insertion/deletion model, with n bins and up to m balls present at a time, and we describe an insertion strategy, called ModulatedGreedy, that achieves a strong bound on maximum load. Here, we describe the simplest possible version of the strategy, which supports any sequence of poly(m) insertions/deletions while guaranteeing a maximum load of $m/n + O(\log m)$ with high probability in m. In the full version of the paper [26], we further extend ModulatedGreedy in various ways, such as supporting an infinite sequence of insertions/deletions, allowing m to increase over time, etc.

The main result of the section is the following:

Theorem 1. *Let $m \geq n$. Consider the insertion/deletion model with n bins and an upper bound of at most m balls present at a time. Consider a sequence of* poly(m) *insertions/deletions, where insertions are implemented using* ModulatedGreedy*. With high probability in m,* ModulatedGreedy *does not halt during any of the insertions/deletions, and no bin ever has load more than $m/n + O(\log m)$.*

When we describe the lower bound for Greedy in Section III, we will see that the main problem with Greedy is that it is too aggressive. Given the choice between two bins i, j, as Greedy always chooses the less loaded of the two—this creates correlations between balls that can be exploited to construct a bad sequence of insertions/deletions. In contrast, ModulatedGreedy will try to be as *unaggressive* as possible, while still guaranteeing an upper gap of $O(\log m)$. In particular, it carefully modulates its behavior and only exhibits a strong bias between two bins i and j if (1) the two bins i and j have significantly different loads; and (2) the system is nearly saturated (i.e., there are nearly m balls present).

As we shall see, this modulated behavior also allows for a simple (but clever) combinatorial analysis, marking a departure from the (typically quite involved) potential-function and Markov-chain arguments used in past analyses of the heavily-loaded case.

A. The Algorithm

The ModulatedGreedy algorithm for allocating a bin to a ball is given below. We assume without loss of generality that m is a multiple of n.

Algorithm 1 The ModulatedGreedy insertion strategy. Here, ℓ_k is the number of balls in bin k prior to the insertion, and c is a large positive constant.

procedure ModulatedGreedy
  Select two bins $i, j \in [n]$ independently and uniformly at random.
  Set $T = m/n + c\log m - \sum_r \ell_r/n$.
  if $(\max_k \ell_k) - (\min_k \ell_k) \leq T$ **then**
    Assign the ball to bin i with probability $1/2 + \frac{\ell_j - \ell_i}{2T}$, and otherwise assign it to bin j.
  else
    Halt.

For $k \in [n]$, let ℓ_k denote the load on bin k prior to the insertion, let $\overline{\ell} = \sum_k \ell_k/n$ be the average bin load, and c be a (sufficiently large) fixed constant. When choosing between two bins i, j, the algorithm exhibits bias

$$(\ell_j - \ell_i)/2T$$

towards bin i, where $T = m/n + c\log m - \overline{\ell}$.

Note that the algorithm is well-defined as long as $|\ell_j - \ell_i| \leq T$ for all $i, j \in [n]$. One should think of T as representing the average amount of leftover space that each bin would have if each bin had a total capacity of $m/n + c\log m$ balls. This means that the bias is proportional to the difference $\ell_j - \ell_i$ between the loads of the bins, and is inversely proportional to the average amount T of space left in each bin.

The following lemma gives a closed-form solution for the probability of a given bin k being selected by ModulatedGreedy.

Lemma 2. *Suppose that $|\ell_i - \ell_j| \leq T$ for all bins i, j. For each bin k, set $T_k = m/n + c\log m - \ell_k$. Upon insertion, bin k is selected with probability $T_k/(nT) = T_k/(\sum_i T_i)$.*

Proof. Let i, j denote the random bin choices for the ball being inserted. The probability that a given bin k is

selected is given by

$$\Pr[i=k,j=k]+\sum_{s\neq k}\Pr[i=k,j=s]\left(\frac{1}{2}+\frac{\ell_s-\ell_k}{2T}\right)$$
$$+\sum_{s\neq k}\Pr[i=s,j=k]\left(\frac{1}{2}+\frac{\ell_s-\ell_k}{2T}\right)$$
$$=\frac{1}{n^2}+\frac{2}{n^2}\sum_{s\neq k}\left(\frac{1}{2}+\frac{\ell_s-\ell_k}{2T}\right)=\frac{2}{n^2}\sum_{s=1}^{n}\left(\frac{1}{2}+\frac{\ell_s-\ell_k}{2T}\right)$$
$$=\frac{2}{n}\left(\frac{1}{2}+\frac{\overline{\ell}-\ell_k}{2T}\right)=\frac{T+\overline{\ell}-\ell_k}{nT}=\frac{T_k}{nT}.$$

Finally we note that $\sum_{i=1}^{n}T_i=\sum_{i=1}^{n}(m/n+c\log m-\ell_i)=m+nc\log m-n\overline{\ell}=nT$. □

B. Analysis

To analyze MODULATEDGREEDY, we begin by describing a seemingly different process (which we call the stone game) that, by design, yields to a simple combinatorial analysis. We then show that the MODULATEDGREEDY algorithm and the stone game can be *coupled together* so that bounds on the behavior of the stone game directly imply bounds on the behavior of MODULATEDGREEDY.

Stone Game. In the (Q,n)-*stone game*, parameterized by Q and n, there are Qn stones which are distributed among two bags; an *inactive bag* and an *active bag*. Initially the active bag is empty, and all the stones are in the inactive bag.

The game supports two types of operations: the ACTIVATE() operation moves a *random* stone from the inactive bag to the active bag; and the DEACTIVATE(r) operation examines the stones in the active bag, selects the stone that was added the r-th most recently, and moves it back to the inactive bag. (ACTIVATE() can only be called if the inactive bag is non-empty, and DEACTIVATE(r) can only be called if the active bag contains r or more balls). The sequence of operations is generated by an oblivious adversary, independent of the random bits used by the game.

The stones are labeled $x_{k,q}$ for $k\in[n],q\in[Q]$. We call k the *color* of the stone, so that there are Q stones of each color. However, the labels of the stone should be thought of as *hidden*, since the behaviors of ACTIVATE() and DEACTIVATE(r) do not depend on the labels of the stones.

We will now prove some lemmas establishing that the stone game is, by design, very well behaved. Our first lemma shows that, even though the adversary gets to perform activations/deactivations, it has no control over which specific stones are in the active bag.

Lemma 3. *At any given moment, if the active/inactive bag contains s stones, then these stones are a uniformly random subset of size s of the stones $\{x_{k,q}\}_{k\in[n],q\in[Q]}$.*

Proof. The point is that the activation/deactivation operations do not depend on the labels of the balls.

Formally, fix any sequence of activations/deactivations and the random choices of the ACTIVATE() operations, and let S be set of stones currently in the inactive bag (the argument for the active bag is identical). Then for any run of the game with a random permutation π applied to the Qn labels $\{x_{k,q}\}_{k\in[n],q\in[Q]}$, the set stones in the active bag will be $\pi(S)$. Thus, if the inactive bag contains s stones, every s-element subset of the nQ stones is equally likely. □

This implies that as long as the inactive bag contains a reasonably large number of stones (namely, $\Omega(n\log(nQ))$), each color is guaranteed to have roughly equal representation in the bag.

Lemma 4. *Suppose at some given moment, the inactive bag contains $s\geq cn\log(nQ)$ stones, for some large enough constant c. Let s_k be the number of these stones with color k. Then $s_k\in[s/2n,3s/2n]$ for each $k\in[n]$, with probability at least $1-1/(Qn)^{\Omega(c)}$.*

Proof. By Lemma 3, the balls S in the inactive bag are a random subset of size s of the Qn balls $\{x_{k,q}\}$. Let $X_k=\{x_{k,1},\ldots,x_{k,Q}\}$ be the set of all color-k balls. Then $s_k=|X_k\cap S|$, the number s_k of balls of color k in S, has the hypergeometric distribution $H(Qn,Q,s)$.

As the standard tail bounds on sampling without replacement at least as sharp as those given by Chernoff bounds for sampling with replacement [55] (Section 23.5), and as $\mathbb{E}[s_i]=s/n$, we get that

$$\Pr[|s_k-s/n|\geq\varepsilon s/n]\leq 2\exp(-\varepsilon^2 s/3n). \qquad (1)$$

Setting $\varepsilon=1/2$, and taking a union bound over the n colors, gives that $s_k\in[s/2n,3s/2n]$ for each $k\in[n]$ with probability $1-2n\exp(-\Omega(c\log Qn))$ which is $1-1/(Qn)^{\Omega(c)}$ for large enough c. □

1) Relating the stone game to the balls-and-bins game: One can think of the stones in the stone game as being similar to balls in the balls-and-bins game—the active bag represents the set of balls that are present, the color of a stone dictates which "bin" a given ball is in, and activations/deactivations correspond to insertions/deletions.

However, there are several significant differences between the games. Notably, the whole point of the balls-and-bins game is to ensure that no single bin contains

too many balls, but in the stone game, the active bag trivially (and deterministically) has at most Q stones of any given color. Nonetheless, we shall now see how to couple the two games together in such a way that our analysis of the stone game yields a bound for the balls-and-bins game.

Mapping between instances. We first giving a mapping between the sequence of insertions/deletions for balls-and-bins game and the input sequence for the stone game. For any sequence $\mathcal{S}$ of insertions/deletions in balls-and-bins game, define $\phi(\mathcal{S})$ to be a corresponding sequence of activations/deactivations, where each INSERT operation is replaced with an ALLOCATE operation, and where each DELETE(x) operation on a ball x is replaced with a DEACTIVATE(r) operation, where $r-1$ is the number of balls in the system that were inserted after x.

The following key lemma shows that the random choices in the two games can be coupled.

Lemma 5 (Coupling)**.** *Let $n \leq m$ and let $\Delta = c\log m$, where c is the positive constant used by* MODULATED-GREEDY*. Consider a sequence $\mathcal{S}$ of insertions/deletions in a balls-and-bins game on n bins, where there are never more than m balls present at a time. Let G_1 be a balls-and-bins game with operation-sequence $\mathcal{S}$ and let G_2 be (Q,n)-stone game with $Q = m/n+\Delta$ with operation sequence $\phi(\mathcal{S})$.*

If G_1 is implemented using MODULATEDGREEDY*, then there exists a coupling between G_1 and G_2 with the following property: Up until* MODULATEDGREEDY *halts, the number of balls in a given bin k (in the balls-and-bins game) always equals the number of stones in the active bag with color k (in the stone game).*

Proof. Let $\ell_1, \ell_2, \ldots, \ell_n$ denote the loads of the bins at any given moment. By Lemma 2, we know that, on any given insertion in which MODULATEDGREEDY does not halt, each bin k is selected with probability

$$\frac{T_k}{nT} = \frac{T_k}{\sum_{i=1}^n T_i}. \tag{2}$$

Now suppose that, for each color k there are ℓ_k stones with color k in the active bag (and hence $Q-\ell_k$ such stones in the inactive bag) of the stone game. Then on any given activation, the probability of a ball with color k being moved into the active bag is

$$\frac{Q-\ell_k}{nQ-\sum_{i=1}^n \ell_i} = \frac{m/n+\Delta-\ell_k}{m+n\Delta-\sum_i \ell_i} = \frac{T_k}{\sum_{i=1}^n T_i}, \tag{3}$$

where the first equality uses that $Q = m/n+\Delta$. The two probabilities (2) and (3) are precisely equal. Thus, we can couple the games so that the bin selected by the insertion in the balls-and-bins game is the same as the stone color selected by the activation in the stone game.

If we implement the insertions/activations in this way, then the deletions/deactivations also become coupled: whenever a ball is deleted from a bin k, a stone with color k is removed from the active bag (in particular, the ball and stone were assigned to have the same bin/color when they were inserted/activated previously). Thus the proof of the lemma is complete. □

Proof of Theorem 1. Finally, we can use the coupling in Lemma 5 to bound the probability of MODULATED-GREEDY halting and prove Theorem 1.

Proof. (Theorem 1) Observe that, if MODULATED-GREEDY does not halt, then deterministically there are at most $m/n+O(\log m)$ balls in any given bin. In particular, the condition $\max_k \ell_k - \min_k \ell_k \leq T$ implies that $\max_k \ell_k - \overline{\ell} \leq T$. Plugging $T = m/n + c\log m - \overline{\ell}$, this gives that $\max_k \ell_k \leq m/n + c\log m$.

Thus, it suffices to analyze the probability of halting.

By Lemma 5, up until MODULATEDGREEDY halts, it can be coupled to a stone game on $nQ = m+nc\log m$ balls, where the number of balls in the active bag never exceeds m. Under this coupling, the number of balls ℓ_k in bin k satisfies $\ell_k = Q - s_k$, where s_k is the number of color-k stones in the inactive bag.

The MODULATEDGREEDY algorithm halts only if

$$|\ell_i - \ell_j| > T = m/n + c\log m - \overline{\ell} = Q - \overline{\ell} \tag{4}$$

for some pair i, j of bins. For the stone game, denoting $s = \sum_k s_k = \sum_k (Q-\ell_k) = n(Q-\overline{\ell})$, and as $|s_i - s_j| = |\ell_i - \ell_j|$, condition (4) is equivalent to

$$|s_i - s_j| > s/n.$$

But we know by Lemma 4 that, w.h.p. in m, we have $|s_i - s_j| \leq s/n$ at all times during the stone game (since the number of balls in the inactive bag is always at least $nc\log m$). Thus, we have w.h.p. in m that MODULATED-GREEDY never halts. □

C. Tightness of the Bound

Clearly, the bound of $m/n+O(\log m)$ is not optimal for all parameter regimes, since it is known that GREEDY achieves maximum load $O(\log\log n)$ in the regime of $n = m$. We remark, however, that for parameter regimes where m is much larger than n, or when n is fixed, this bound is essentially optimal.

Proposition 6. *Let c be a sufficiently large positive constant. Consider 4 bins using any sequential 2-choice insertion strategy. Then there exists a sequence of $\mathrm{poly}(m)$ operations such that, with high probability in m, there is some point in time at which some bin contains at least $m/4+\Omega(\log m)$ balls.*

For brevity, we defer the proof of Proposition 6 to the full version of the paper [26].

III. A Lower Bound for Greedy with Deletions

This section gives a lower bound for the GREEDY algorithm in the insertion/deletion model against an oblivious adversary, with up to m balls present at a time. Recall that the trivial SINGLECHOICE strategy achieves an overload of $O(\sqrt{(m/n)\log n})$ (w.h.p. in m) in this setting, so the natural question is whether GREEDY does any better. We show that, even for $n=4$ (meaning that SINGLECHOICE has an overload of $O(\sqrt{n})$), it does not.

Theorem 7. *Consider the insertion/deletion model on $n=4$ bins, with the restriction that at most m balls can be present at any time, and suppose that insertions are implemented using* GREEDY*. There exists an oblivious sequence of $\mathrm{poly}(m)$ insertions/deletions such that, after the sequence is complete, we have with probability $\Omega(1)$ that some bin contains $m/4+\Omega(\sqrt{m})$ balls.*

For ease of exposition, and to keep the main ideas as clear as possible, we focus our lower bound on $n=4$ bins. We will also see, however, that for general n and m, a similar construction gives an $m/n+\Omega(\sqrt{m}/\mathrm{poly}(n))$ lower bound on the maximum load.

A. A Simpler $\Omega(m^{1/4})$ Bound

Before proving Theorem 7, we first describe a simpler (but already surprisingly) lower bound of $m/4+\Omega(m^{1/4})$. Later in Section III-B we build on these ideas to prove Theorem 7.

We first describe a construction with the property that if we ever reach a state where one of the bins (say, bin 1) contains significantly fewer balls (say, k fewer balls) than the other bins, then we can subsequently reach a state in which (with probability $\Omega(1)$), some bin contains at least $m/4+\Omega(\sqrt{k})$ balls. As we shall see later in the subsection, this can be used to directly obtain the $m/4+\Omega(m^{1/4})$ bound.

Proposition 8 (Gap to overload)**.** *Consider the* GREEDY *algorithm on 4 bins, on instances where at most m balls can be present at a time. Suppose we begin in a state that contains at most $m-k$ balls, and where bin 1 contains $k+1$ fewer balls than each of bins $2,3,4$. Then there is an oblivious sequence of $O(m)$ insertions/deletions such that, after the sequence is complete, we have the following property with probability $\Omega(1)$: some bin contains $m/4+\Omega(\sqrt{k})$ balls.*

Proof. Let X_0 denote the initial state of the game. Consider the sequence with the following three steps.

1) Insert k balls $x_1,x_2,\ldots,x_k$ to get to a state X_1.
2) Then insert $m-j$ balls $y_1,y_2,\ldots,y_{m-j}$, where j is the number of balls in state X_1—this brings us to a state X_2 with m balls in total.
3) Finally, delete the balls $x_1,x_2,\ldots,x_k$, and insert new balls $z_1,z_2,\ldots,z_k$ to reach a state X_3.

We claim that, for at least one of the two states X_2 and X_3, we have with probability $\Omega(1)$ that some bin contains $m/4+\Omega(\sqrt{k})$ balls. (This means that, if we terminate the sequence of operations randomly at one of X_2 or X_3, then after the sequence is complete, we have with probability $\Omega(1)$ that some bin contains $m/4+\Omega(\sqrt{k})$ balls.)

During the insertions of $x_1,x_2,\ldots,x_k$, we are always in a state where bin 1 contains fewer balls than bins $2,3,4$. Thus, each insertion x_i will go into bin 1 if and only if $1\in\{h_1(x_i),h_2(x_i)\}$ (this is where we are exploiting that the GREEDY algorithm is too aggressive). The number A of balls $x_1,x_2,\ldots,x_k$ that are placed in bin 1 is therefore given by $A=|\{i\mid 1\in\{h_1(x_i),h_2(x_i)\}\}|$.

Let $\mu=\mathbb{E}[A]$. As A is a binomial random variable with mean $\mu=\Theta(k)$, with probability $\Omega(1)$ we have $A\geq\mu+\Omega(\sqrt{k})$. Now consider the number B of balls $z_1,z_2,\ldots,z_k$ that are placed into bin 1. We deterministically have that

$$B\leq|\{i\mid 1\in\{h_1(z_i),h_2(z_i)\}\}|. \tag{5}$$

Since the right side of (5) is a binomial random variable with mean μ, we have with probability $\Omega(1)$ that $B\leq\mu$. Moreover, since A and B are independent, the bounds on A and B hold simultaneously with probability $\Omega(1)$.

Finally, let us consider the number of balls in bins $2,3,4$ once we reach state X_3. Assume that state X_2 has maximum load $m/4+o(\sqrt{k})$, otherwise we are already done. Then, since X_2 contains m balls in total, bins $2,3,4$ must contain a total of at least $3m/4-o(\sqrt{k})$ balls. By step 3 of the input sequence above, it follows that, in state X_3, the total number of balls in bins $2,3,4$ is at least

$$3m/4-o(\sqrt{k})-(k-A)+(k-B).$$

Conditioning on the event above, and plugging in our bounds for A and B, we see that (with probability $\Omega(1)$) this is at least $3m/4+\Omega(\sqrt{k})$. Thus, at least one of bins $2,3,4$ must contain $m/4+\Omega(\sqrt{k})$ balls, as desired. □

The lower bound. Using Proposition 8, the claimed lower bound follows quite easily. Consider the following input sequence, starting from an empty system. (1) Insert m balls into the system; (2) delete each ball independently and randomly with probability $1/2$; and (3) apply the sequence in Proposition 8 with $k = \sqrt{m}$.

As the deletions are random in step (2), the precondition for Proposition 8 (i.e., the least loaded bin contain at least $k = \sqrt{m}$ fewer balls than every other bins) holds with probability $\Omega(1)$. So by Proposition 8, we can achieve $m/4 + \Omega(\sqrt{k}) = m/4 + \Omega(m^{1/4})$ balls in some bin, with probability $\Omega(1)$.

We remark that, for n bins, where n is arbitrary, the same approach gives a lower bound of

$$m/n + \Omega(m^{1/4}/\sqrt{n^3 \log n}). \tag{6}$$

with probability $\Omega(1)$. For a more detailed discussion of this bound, we refer the reader to the full version of this paper [26].

B. The Stronger $\Omega(m^{1/2})$ Lower Bound

We now show how to achieve the stronger bound of $m/4 + \Omega(\sqrt{m})$ balls in some bin. Given Proposition 8, to prove Theorem 7 it suffices to show how to achieve a gap of $k = \Omega(m)$ between bin 1 and bins $2,3,4$. This is accomplished in the following proposition.

Proposition 9. *Consider the* GREEDY *algorithm on 4 bins, with the restriction that at most m balls can be present at a time. There exists an oblivious sequence of* $\mathrm{poly}(m)$ *insertions/deletions such that, after the sequence is complete, we have the following property with probability $\Omega(1)$: Bin 1 contains $\Omega(m)$ fewer balls than each of bins $2,3,4$.*

The rest of the section proves Proposition 9.

Let $0 < \varepsilon_1, \varepsilon_2, \varepsilon_3 < 1$ be constants, where ε_2 is sufficiently small as a function of ε_1, and let ε_3 is sufficiently small as a function of ε_2. Sometimes we will write $\varepsilon_1, \varepsilon_2, \varepsilon_3$ inside the $O(\cdot)$ notation, to make the dependence on them explicit, while hiding fixed constants that do not depend on $\varepsilon_1, \varepsilon_2, \varepsilon_3$.

Some basic gadgets. We begin with a basic technical lemma establishing that GREEDY has a tendency of eliminating imbalances over time. For brevity (and since the proof follows from standard arguments), we defer the proof of Lemma 10 to the extended paper [26].

Lemma 10. *Consider the* GREEDY *algorithm on 4 bins, and fix an arbitrary initial state in which the bins have loads within $\varepsilon_2 m$ of each other. If $\varepsilon_1 m$ insertions are performed, then after the sequence is complete, all of the bins have loads within $O(\log m)$ of each other with high probability in m. Furthermore, with high probability in m, there is a point in time prior to the final insertion at which all of the bins have equal loads.*

Using Lemma 10, we construct a simple strategy for forcing GREEDY to add a ball to a random bin.

Lemma 11 (Uniform ball placement gadget)**.** *Consider the* GREEDY *algorithm on 4 bins, and fix an arbitrary initial state in which the bins have loads within $\varepsilon_2 m$ of each other. Suppose we insert balls $x_1, \ldots, x_{\varepsilon_1 m}$, and then we delete balls $x_1, \ldots, x_{\varepsilon_1 m - 1}$ (all except the last insertion). With high probability in m, this is equivalent to placing the ball $x_{\varepsilon_1 m}$ uniformly at random into one of the bins $1,2,3,4$.*

Proof. We have by Lemma 10 that, with high probability in m, there is some insertion x_i, $i \in [\varepsilon_1 m - 1]$, after which the bins have equal loads. It follows that, from the perspectives of insertions $x_{i+1}, \ldots, x_{\varepsilon_1 m}$, the four bins are symmetric. Thus the last insertion $x_{\varepsilon_1 m}$ is equally likely to be placed into each of the bins, which establishes the lemma. □

Lemma 11 allows for us to place a ball into a random bin, but we can only do this $O(m)$ times before there are too many balls ($> m$) in the system. But for the purposes of Proposition 9, we will need to do this $\Omega(m^2)$ times. Our next lemma provides a mechanism for reducing the number of balls that are present while having only a small effect on the relative loads of the bins.

Lemma 12 (Almost equal load reduction gadget)**.** *Consider the* GREEDY *algorithm on 4 bins, and fix an arbitrary initial state in which the bins $1,2,3,4$ have loads $\ell_1, \ell_2, \ell_3, \ell_4$ within $\varepsilon_2 m$ of each other. We can construct an oblivous sequence of $O(\varepsilon_1 m)$ insertions/deletions such that, after this sequence, the total number of balls in the system is at most $\varepsilon_1 m$; and such that, with high probability in m, the new bin loads ℓ_i' for $i \in [4]$ satisfy*

$$\ell_i' = \ell_i - r + Y^{(i)}, \tag{7}$$

where $r \in \mathbb{N}$, $|Y^{(i)}| \le O(\log m)$, and $\mathbb{E}[Y^{(i)}] = 0$.

Proof. Let us begin by describing is a sequence of $O(\varepsilon_1 m)$ insertions/deletions after which (1) the total number of balls in the system is at most $\varepsilon_1 m$; and (2) the new loads ℓ_i' of the bins satisfy (w.h.p. in m)

$$\ell_i' = r - \ell_i + Y^{(i)}, \tag{8}$$

where $r \in \mathbb{N}$, $|Y^{(i)}| \le O(\log m)$, and $\mathbb{E}[Y^{(i)}] = 0$. (Note that (8) is the same as (7) but with r and ℓ_i flipped).

The lemma would then follow by applying the above construction twice. That is, first we obtain $\ell'_1,\ell'_2,\ell'_3,\ell_4$ satisfying (8), and then apply it again to obtain $\ell''_1,\ell''_2,\ell''_3,\ell''_4$ satisfying

$$\ell_i'' = r' - \ell'_i + Y'^{(i)}, \tag{9}$$

where $r' \in \mathbb{N}$, $|Y'^{(i)}| \le O(\log m)$, and $\mathbb{E}[Y'^{(i)}] = 0$. Chaining together (8) and (9), we get relationship between $\ell_1,\ell_2,\ell_3,\ell_3$ and $\ell''_1,\ell''_2,\ell''_3,\ell''_4$ as desired by (7).

Our construction for achieving (8) is very simple: we perform $\varepsilon_1 m$ insertions $x_1,x_2,\ldots,x_{\varepsilon_1 m}$, and then we delete all of the *other elements* besides $x_1,x_2,\ldots,x_{\varepsilon_1 m}$. Let ℓ_i be the load of bin i before these insertions/deletions, let q_i be the load of bin i after the insertions are completed (but the deletions have not yet begun), and let ℓ'_i be the load of bin i after the deletions have completed.

By Lemma 12, the quantities q_1,q_2,q_3,q_4 are within $O(\log m)$ of each other (w.h.p. in m). Moreover, w.h.p. in m, there is some point during the insertions at which all of the bins have equal loads—if we condition on this, then we have $\mathbb{E}[q_1] = \mathbb{E}[q_2] = \mathbb{E}[q_3] = \mathbb{E}[q_4]$ by symmetry. Defining $Y^{(i)} = q_i - \mathbb{E}[q_i]$, we have $|Y^{(i)}| \le O(\log m)$, and $\mathbb{E}[Y^{(i)}] = 0$.

As $\ell'_i = q_i - \ell_i$, we have $\ell'_i = \mathbb{E}[q_i] + Y^{(i)} - \ell_i$. Setting $r = \mathbb{E}[q_i]$, it follows that (8) holds w.h.p. in m. □

Applying the gadgets. We say that an application of Lemma 11 or of Lemma 12 *fails* if either: the precondition of $\ell_1,\ell_2,\ell_2,\ell_4$ being within $\varepsilon_2 m$ of each other fails (this is a *precondition failure)*; or the high-probability guarantee offered by the lemma fails (this is a *probabilistic failure*).

We now describe the sequence of insertions/deletions that we use to achieve Proposition 9. We perform $\varepsilon_3 m$ phases, where phase $a \in [\varepsilon_3 m]$ proceeds as follows:

- Apply Lemma 11 m times, one after another. For $b \in [m]$, use $Z_{m\cdot(a-1)+b}$ to denote the bin that the b-th application of the lemma adds a ball to. If the lemma fails, then for the sake of analysis, we redefine $Z_{m\cdot(a-1)+b}$ to be uniformly random in $[4]$. Thus, regardless of whether the lemma fails, the Z_i's are independently and uniformly random in $[4]$.
- Apply Lemma 12 once to reduce the loads almost equally. Let $Y_a^{(1)},Y_a^{(2)},Y_a^{(3)},Y_a^{(4)}$ denote the outcomes of $Y^{(1)},Y^{(2)},Y^{(3)},Y^{(4)}$ in that application of the lemma. If the lemma fails, then for the sake of analysis, we redefine $Y_a^{(1)},Y_a^{(2)},Y_a^{(3)},Y_a^{(4)}$ to be 0.

To analyze the sequence of insertions/deletions, we first argue that the $Y_i^{(s)}$s have a negligible effect on the loads of the bins at any given moment.

Lemma 13. *Let $s \in [4]$ and $k \in [\varepsilon_3 m]$. Then w.h.p. in m, it holds that for each k, $|\sum_{a=1}^{k} Y_a^{(s)}| \le \tilde{O}(\sqrt{m})$, where $\tilde{O}(\cdot)$ hides polylogarithmic factors in m.*

Proof. The sequence of partial sums $P_r = \sum_{a=1}^{r} Y_a^{(s)}$ for $r = 0,\ldots,k$ forms a martingale satisfying $|P_r - P_{r-1}| = O(\log m)$ deterministically for each $r \in [k]$. The lemma follows from Azuma's inequality. □

Next we consider the effect of the $\varepsilon_3 m^2$ insertions Z_i over the εm phases, and show that with probability at least $1-\varepsilon_2$, there is no point in time at which the Z_i's cause an imbalance of more $\varepsilon_2 m/2$.

For $k \in [\varepsilon_3 m^2]$ and $s \in [4]$, let

$$S(k,s) = |\{i \in [k] \mid Z_i = s\}|$$

denote the number insertions in bin s during the first k applications of Lemma 11.

Lemma 14. *Let $s \in [4]$ and $\varepsilon_2 = (2\varepsilon_3)^{1/3}$. With probability at least $1-\varepsilon_2$, it holds (simultaneously) for all $k \in [\varepsilon_3 m^2]$ that $|S(k,s) - k/4| \le \varepsilon_2 m/2$.*

Proof. As Z_i is equal to s independently with probability $1/4$, the sequence $S(k,s) - k/4$ for $k = 0,1,\ldots,\varepsilon_3 m^2$ forms a martingale with increments $\{-1/4, 3/4\}$ (and hence variance at most 1). By the maximal inequality for martingales, for any $\lambda > 0$,

$$\Pr\Big[\max_{k\in[\varepsilon_3 m^2]} |S(k,s)| > \lambda\Big] \le 2\frac{\mathrm{Var}[S(\varepsilon_3 m^2, s)]}{\lambda^2} \le 2\frac{\varepsilon_3 m^2}{\lambda^2}.$$

Setting $\lambda = m(2\varepsilon_3/\varepsilon_2)^{1/2}$ so that the right hand side above is ε_2, and choosing $\varepsilon_2^3 \le 2\varepsilon_3$ so that $\lambda \ge \varepsilon_2 m$ gives the claimed result. □

Combining Lemmas 13 and 14, we can bound the probability of any failures occurring during our construction.

Lemma 15. *With probability at least $1-\varepsilon_2 - 1/\mathrm{poly}(m)$, no failures (either precondition failures or probabilistic failures) occur during the construction.*

Proof. Probabilistic failures occur with probability only $1/\mathrm{poly}(m)$ per application of Lemma 11 or Lemma 12. Across the $O(m^2)$ applications of the lemmas, the probability of a probabilistic failure ever occurring is at most $1/\mathrm{poly}(m)$. For the rest of the proof, we condition on no probabilistic failures occurring.

We now bound the probability of any precondition failure. Before any particular application of Lemma 11 or Lemma 12 (during the input sequence of insertions/deletions), for bin $s \in [4]$, the amount by which

its load differs from the mean can be expressed as $|\sum_{i=1}^{k_1} Y_i^{(s)} + S(k_2,s) - k_2/4|$ for some k_1, k_2. By Lemmas 13 and 14, the probability that this quantity ever exceeds $\varepsilon_2 m$ (and hence any precondition failure occurring) is at most $\varepsilon_2 + 1/\operatorname{poly}(m)$, which completes the proof. □

Finally, we argue that with probability at least ε_1, the Z_i's *do cause* an imbalance of $\Omega(m)$. In particular, bin 1 contains $\Omega(m)$ fewer balls than bins $2,3,4$.

Lemma 16. *With probability at least ε_1, we have that*

$$|S(\varepsilon_3 m^2, 1)\}| < \max_{s\in\{2,3,4\}} |S(\varepsilon_3 m^2, s)| - \Omega(\sqrt{\varepsilon_3} m).$$

Proof. Let X_s denote the number of such balls inserted in bin s. Then X_1 is a binomial random variable with mean $\mu = \Theta(\varepsilon_3 m^2)$. Thus, with probability at least $2\varepsilon_1$, we have that, $X_1 \le \mu - 10\sqrt{\mu}$. On the other hand, if we condition on some value $\le \mu - 10\sqrt{\mu}$ for X_1, then the variables X_2, X_2, X_4 become binomial random variables with means $\mu' > \mu$. Each X_i has probability at least 0.9 of satisfying $X_i > \mu' - 5\sqrt{\mu'} \ge \mu - 5\sqrt{\mu}$. Thus, if we condition on $X_1 \le \mu - 10\sqrt{\mu}$, then the probability at least 0.7, we have $X_2, X_3, X_4 > \mu - 5\sqrt{\mu}$. Putting these together, the probability that $\max\{X_2, X_3, X_4\} - X_1 > 5\sqrt{\mu}$ is at least $\Pr[X_1 \le \mu - 10\sqrt{\mu}] \cdot \Pr[X_2, X_3, X_4 > \mu - 5\sqrt{\mu} \mid X_1 \le \mu - 10\sqrt{\mu}] \ge 2\varepsilon_1 \cdot 0.7 > \varepsilon_1$. □

Proof of Proposition 9. We prove the proposition using the construction described in this section. Note that, by design, there are never more than m balls present at a time, as Lemma 12 brings the number of balls back down to $\varepsilon_1 m$ every $O(\varepsilon_1 m)$ operations.

By Lemma 15, with probability at least $1 - \varepsilon_2 - 1/\operatorname{poly}(n)$, all of the applications of Lemma 11 and Lemma 12 succeed. Conditioned on this, at the end of the construction, the gap of each bin $s \in [4]$ is

$$\sum_{i=1}^{\varepsilon_3 m} Y_i^{(s)} + S(\varepsilon_3 m^2, s) - \varepsilon_3 m^2/4.$$

By Lemma 13, we have $\left|\sum_{i=1}^{\varepsilon_3 m} Y_i^{(s)}\right| \le \tilde{O}(\sqrt{m})$ with high probability in m. On the other hand, by Lemma 16,

$$|S(\varepsilon_3 m^2, 1)| < \max_{s\in\{2,3,4\}} |S(\varepsilon_3 m^2, s)| - \Omega(\sqrt{\varepsilon_3} m)$$

with probability at least ε_1. It follows that, with probability at least $\varepsilon_1 - \varepsilon_2 - 1/\operatorname{poly}(n)$, the load of bin 1 at the end of the construction is $\Omega(\sqrt{\varepsilon_3} m)$ smaller than the loads of bins $2,3,4$. □

IV. Acknowledgements

Nikhil Bansal was supported in part by the NWO VICI grant 639.023.812. William Kuszmaul was partially sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. William Kuszmaul was also partially supported by a Hertz Fellowship and an NSF GRFP Fellowship.

References

[1] M. Mitzenmacher, A. W. Richa, and R. Sitaraman, "The power of two random choices: A survey of techniques and results," *Combinatorial Optimization*, vol. 9, pp. 255–304, 2001.

[2] U. Wieder *et al.*, "Hashing, load balancing and multiple choice," *Foundations and Trends® in Theoretical Computer Science*, vol. 12, no. 3–4, pp. 275–379, 2017.

[3] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," in *Symposium on theory of computing (STOC)*, 1994, pp. 593–602.

[4] R. Pagh and F. F. Rodler, "Cuckoo hashing," *Journal of Algorithms*, vol. 51, no. 2, pp. 122–144, 2004.

[5] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 1–19.

[6] F. Bonomi, M. Mitzenmacher, R. Panigrahy, S. Singh, and G. Varghese, "An improved construction for counting bloom filters," in *European Symposium on Algorithms (ESA)*. Springer, 2006, pp. 684–695.

[7] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan, "Online stochastic matching: Beating 1-1/e," in *Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2009, pp. 117–126.

[8] B. Haeupler, V. S. Mirrokni, and M. Zadimoghaddam, "Online stochastic weighted matching: Improved approximation algorithms," in *International Workshop on Internet and Network Economics*. Springer, 2011, pp. 170–181.

[9] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.

[10] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, 2008.

[11] K. Ousterhout, P. Wendell, M. Zaharia, and I. Stoica, "Sparrow: distributed, low latency scheduling," in *Symposium on Operating Systems Principles (SOSP)*, 2013, pp. 69–84.

[12] D. Ongaro, S. M. Rumble, R. Stutsman, J. Ousterhout, and M. Rosenblum, "Fast crash recovery in ramcloud," in *Symposium on Operating Systems Principles (SOSP)*, 2011, pp. 29–41.

[13] A. Broder and M. Mitzenmacher, "Using multiple hash functions to improve ip lookups," in *Conference on Computer Communications (INFOCOM)*, vol. 3. IEEE, 2001, pp. 1454–1463.

[14] B. Vöcking, "How asymmetry helps load balancing," in *Foundations of Computer Science (FOCS)*, 1999, p. 131.

[15] R. Cole, A. Frieze, B. M. Maggs, M. Mitzenmacher, A. W. Richa, R. Sitaraman, and E. Upfal, "On balls and bins with deletions," in *International Workshop on Randomization and Approximation Techniques in Computer Science*. Springer, 1998, pp. 145–158.

[16] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1094–1104, 2001.

[17] ——, "Studying balanced allocations with differential equations," *Combinatorics, Probability and Computing*, vol. 8, no. 5, pp. 473–482, 1999.

[18] P. Berenbrink, A. Czumaj, A. Steger, and B. Vöcking, "Balanced allocations: the heavily loaded case," in *Symposium on Theory of Computing (STOC)*, 2000, pp. 745–754.

[19] R. Cole, B. M. Maggs, F. M. auf der Heide, M. Mitzenmacher, A. W. Richa, K. Schröder, R. K. Sitaraman, and B. Vöcking, "Randomized protocols for low congestion circuit routing in multistage interconnection networks," in *Symposium on Theory of Computing (STOC)*. ACM, 1998, pp. 378–388.

[20] B. Vöcking, "How asymmetry helps load balancing," *Journal of the ACM (JACM)*, vol. 50, no. 4, pp. 568–589, 2003.

[21] Y. Peres, K. Talwar, and U. Wieder, "The (1+ β)-choice process and weighted balls-into-bins," in *Symposium on Discrete Algorithms (SODA)*. SIAM, 2010, pp. 1613–1619.

[22] D. Los, T. Sauerwald, and J. Sylvester, "Balanced allocations: Caching and packing, twinning and thinning," in *Symposium on Discrete Algorithms (SODA)*. SIAM, 2022, pp. 1847–1874.

[23] K. Talwar and U. Wieder, "Balanced allocations: A simple proof for the heavily loaded case," in *International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer, 2014, pp. 979–990.

[24] N. Bansal and O. Feldheim, "Well-balanced allocation on general graphs," in *Symposium on Theory of Computing (STOC)*, 2022.

[25] K. Kenthapadi and R. Panigrahy, "Balanced allocation on graphs," in *Symposium on Discrete Algorithms (SODA)*, vol. 6, 2006, pp. 434–443.

[26] N. Bansal and W. Kuszmaul, "Balanced allocations: The heavily loaded case with deletions," *arXiv preprint arXiv:2205.06558*, 2022.

[27] O. N. Feldheim and O. Gurel-Gurevich, "The power of thinning in balanced allocation," *Electronic Communications in Probability*, vol. 26, pp. 1–8, 2021.

[28] D. Los and T. Sauerwald, "Balanced allocations with incomplete information: The power of two queries," in *Innovations in Theoretical Computer Science Conference (ITCS)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[29] ——, "Balanced allocations in batches: Simplified and generalized," *arXiv preprint arXiv:2203.13902*, 2022.

[30] P. Berenbrink, T. Friedetzky, Z. Hu, and R. Martin, "On weighted balls-into-bins games," *Theoretical Computer Science*, vol. 409, no. 3, pp. 511–520, 2008.

[31] K. Talwar and U. Wieder, "Balanced allocations: the weighted case," in *Symposium on Theory of Computing (STOC)*, 2007, pp. 256–265.

[32] V. Stemann, "Parallel balanced allocations," in *Symposium on Parallel algorithms and Architectures (SPAA)*, 1996, pp. 261–269.

[33] C. Lenzen, M. Parter, and E. Yogev, "Parallel balanced allocations: The heavily loaded case," in *Symposium on Parallelism in Algorithms and Architectures (SPAA)*, 2019, pp. 313–322.

[34] P. Berenbrink, A. Czumaj, M. Englert, T. Friedetzky, and L. Nagel, "Multiple-choice balanced allocation in (almost) parallel," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2012, pp. 411–422.

[35] P. Berenbrink, T. Friedetzky, P. Kling, F. Mallmann-Trenn, L. Nagel, and C. Wastell, "Self-stabilizing balls & bins in batches: The power of leaky bins," in *Symposium on Principles of Distributed Computing (PODC)*, 2016, pp. 83–92.

[36] ——, "Self-stabilizing balls and bins in batches," *Algorithmica*, vol. 80, no. 12, pp. 3673–3703, 2018.

[37] A. Aamand, J. B. T. Knudsen, and M. Thorup, "Load balancing with dynamic set of balls and bins," in *Symposium on Theory of Computing (STOC)*, 2021, pp. 1262–1275.

[38] M. A. Bender, M. Farach-Colton, J. Kuszmaul, and W. Kuszmaul, "On the optimal time/space tradeoff for hash tables," in *Symposium on Theory of Computing (STOC)*, 2022.

[39] N. D. Vvedenskaya, R. L. Dobrushin, and F. I. Karpelevich, "Queueing system with selection of the shortest of two queues: An asymptotic approach," *Problemy Peredachi Informatsii*, vol. 32, no. 1, pp. 20–34, 1996.

[40] D. Mukherjee, S. C. Borst, J. S. Van Leeuwaarden, and P. A. Whiting, "Universality of power-of-d load balancing in many-server systems," *Stochastic Systems*, vol. 8, no. 4, pp. 265–292, 2018.

[41] M. Bramson, Y. Lu, and B. Prabhakar, "Randomized load balancing with general service time distributions," *ACM SIGMETRICS performance evaluation review*, vol. 38, no. 1, pp. 275–286, 2010.

[42] M. J. Luczak and C. McDiarmid, "On the maximum queue length in the supermarket model," *The Annals of Probability*, vol. 34, no. 2, pp. 493–527, 2006.

[43] G. Brightwell and M. Luczak, "The supermarket model with arrival rate tending to one," *arXiv preprint arXiv:1201.5523*, 2012.

[44] P. Eschenfeldt and D. Gamarnik, "Supermarket queueing system in the heavy traffic regime. short queue dynamics," *arXiv preprint arXiv:1610.03522*, 2016.

[45] M. J. Luczak and J. Norris, "Strong approximation for the supermarket model," *The Annals of Applied Probability*, vol. 15, no. 3, pp. 2038–2061, 2005.

[46] M. A. Bender, A. Conway, M. Farach-Colton, W. Kuszmaul, and G. Tagliavini, "Tiny pointers," *arXiv preprint arXiv:2111.12800*, 2021.

[47] ——, "All-purpose hashing," *arXiv preprint arXiv:2109.04548*, 2021.

[48] M. Mitzenmacher, "How useful is old information (extended abstract)?" in *Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing*, 1997, pp. 83–91.

[49] M. Dahlin, "Interpreting stale load information," *IEEE Transactions on parallel and distributed systems*, vol. 11, no. 10, pp. 1033–1047, 2000.

[50] W. Kuszmaul, "How asymmetry helps buffer management: achieving optimal tail size in cup games," in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, 2021, pp. 1248–1261.

[51] M. A. Bender, M. Farach-Colton, and W. Kuszmaul, "Achieving optimal backlog in multi-processor cup games," in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 2019, pp. 1148–1157.

[52] M. A. Bender and W. Kuszmaul, "Randomized cup game algorithms against strong adversaries," in *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2021, pp. 2059–2077.

[53] W. Kuszmaul, "Achieving optimal backlog in the vanilla multiprocessor cup game," in *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2020, pp. 1558–1577.

[54] M. A. Bender, R. Das, M. Farach-Colton, R. Johnson, and W. Kuszmaul, "Flushing without cascades," in *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2020, pp. 650–669.

[55] A. Frieze and M. Karonski, *Introduction to Random Graphs*. Cambridge University Press, 2015.