Infinite Balanced Allocation via Finite Capacities

Petra Berenbrink
Universität Hamburg
Hamburg, Germany
petra.berenbrink@uni-hamburg.de

Tom Friedetzky
Durham University
Durham, U.K.
tom.friedetzky@durham.ac.uk

Christopher Hahn
Universität Hamburg
Hamburg, Germany
tim.christopher.hahn@uni-hamburg.de

Lukas Hintze
Universität Hamburg
Hamburg, Germany
lukas.hintze@uni-hamburg.de

Dominik Kaaser
Universität Hamburg
Hamburg, Germany
dominik.kaaser@uni-hamburg.de

Peter Kling
Universität Hamburg
Hamburg, Germany
peter.kling@uni-hamburg.de

Lars Nagel
Loughborough University
Loughborough, U.K.
l.nagel@lboro.ac.uk

***Abstract*—We analyze the following infinite load balancing process, modeled as a classical balls-into-bins game: There are n bins (servers) with a limited capacity (buffer) of size $c=c(n)\in\mathbb{N}$. Given a fixed arrival rate $\lambda=\lambda(n)\in(0,1)$, in every round λn new balls (requests) are generated. Together with possible leftovers from previous rounds, these balls compete to be allocated to the bins. To this end, every ball samples a bin independently and uniformly at random and tries to allocate itself to that bin. Each bin accepts as many balls as possible until its buffer is full, preferring balls of higher age. At the end of the round, every bin deletes the ball it allocated first.**

We study how the buffer size c affects the performance of this process. For this, we analyze both the number of balls competing each round (including the leftovers from previous rounds) as well as the worst-case waiting time of individual balls. We show that (i) the number of competing balls is at any (even exponentially large) time bounded with high probability by $4\cdot c^{-1}\cdot\ln\big(1/(1-\lambda)\big)\cdot n+\mathrm{O}\big(c\cdot n\big)$ and that (ii) the waiting time of a given ball is with high probability at most $\big(4\cdot\ln\big(1/(1-\lambda)\big)\big)/\big(c\cdot(1-1/e)\big)+\log\log n+\mathrm{O}\big(c\big)$. These results indicate a sweet spot for the choice of c around $c=\Theta\big(\sqrt{\log(1/(1-\lambda))}\big)$. Compared to a related process with infinite capacity [Berenbrink et al., PODC'16], for constant λ the waiting time is reduced from $\mathrm{O}(\log n)$ to $\mathrm{O}(\log\log n)$. Even for large $\lambda\approx 1-1/n$ we reduce the waiting time from $\mathrm{O}(\log n)$ to $\mathrm{O}(\sqrt{\log n})$.

***Index Terms*—load balancing, balls-into-bins, job allocation, capacity, infinite process, thresholds**

I. Introduction

The model of balls and bins is frequently used as primitive operation when modeling (probabilistic) load balancing, be it access to RAM or hard drives, processing tasks on CPUs or servers, or hashing. Generally the objective is to obtain an even workload distribution among resources, just minimizing the maximum workload of any resource, or in case of infinite processes, guaranteeing some notion of "stability". Maintaining a table of all (or a significant portion of all) servers' workloads may be too costly, and a centralized approach introduces a single point of failure and possible communication bottlenecks, which may be undesirable. Because of its simplicity, a common approach is to use a randomized load balancing strategy where each client is given a list of servers from which it chooses one random server to send its request to; often this list of servers is in fact the entire set. The authors of [19] show that this approach can balance the load fairly well; for n servers the maximum load is $\approx m/n+\sqrt{m\log(n)/n}$ for $m=\Omega(n\log n)$ many tasks and $O(\log n)$ for $m=n$ tasks.

The authors of [4] analysed a process allowing clients to send requests to $d\geq 2$ servers and committing to a server (of those queried) of lowest load. This leads to an exponentially better maximum load ($m/n+\log\log n+O(1)$), provided the clients operate *sequentially* [4, 9]. This effect is commonly referred to as "power of two choices" and it works very well in sequential settings. Not surprisingly, in parallel settings this d-choice process loses some of its powers [2, 7, 10]. To see this assume the tasks are allocated in batches of size n, one batch after another, but the balls within a batch simultaneously. If on the one hand the servers' answers to load requests do not include the requests of the current batch then the maximum load is more or less that of the one-choice case since the expected maximum number of tasks allocated to some server is $\Omega(\log n)$. If on the other hand the servers' answers do include the requests of the current batch then the load of a server can be far away from the actual number of tasks after the allocation. See [7] for details.

In this paper we consider a process where, instead of allowing multiple choices per tasks, the servers are equipped with buffers allowing the servers to accept multiple tasks from every batch. We consider the following parallel, infinite scenario: we have n servers which receive requests from clients. Every client is given a list of all servers, and it can query single randomly chosen servers. Every server has a limited-capacity buffer in which some requests are stored, and the server processes them one by one. While the clients can be neglected in our model, the requests are modeled as balls and the servers as bins. Our model of parallelism assumes synchronous rounds; in every round a batch of λn new balls arrives, and after the allocation stage every bin processes (i.e., deletes) the ball that it allocated first. Unallocated balls will simply be added to the batch of new balls arriving in the subsequent round.

On the result side of things, our main aim is to show long-term "stability" in terms of bounds on the size of the "pool" of balls needing to be allocated and the balls' waiting times, mainly as function of the generation coefficient λ and the bins' buffer size, at an arbitrary point of time. We however

also strive to keep the balancing protocols as simple and natural as possible, to use only fundamental mathematical tools and techniques in our analyses, and to have presentations of proofs as intuitive and streamlined as we see possible (the latter self-imposed requirement in particular made us resist the urge to optimize most constant coefficients in bounds etc.).

A. Related Work

There is a vast amount of literature on balls-into-bins games in different settings. In the static setting the number of balls is fixed whereas in infinite settings balls are allocated over an infinite period of time. In a parallel balls-into-bins game the balls are allocated concurrently, whereas balls are allocated one after the other in sequential settings. While there is a large amount of literature on sequential balls-into-bins games, both in the finite and the infinite settings, e.g., [4, 6, 9, 21], there is far less work on parallel balls-into-bins protocols. In this literature overview we focus on parallel and infinite games.

For $m=n$ it is known that placing the balls independently and uniformly at random (i.u.r.) leads to a maximum load of $(1-\mathrm{o}(1))\log n/\log\log n$ w.h.p.[1] A well-known paradigm called *power of two choices* works well in sequential settings. For example, Azar et al. [4] introduce the protocol GREEDY[d] and show that the load of the highest-loaded bin is at most $\log\log n/\log d+\mathrm{O}(1)$ w.h.p. by allocating each ball to the least loaded of $d\geq 2$ randomly chosen bins.

a) Static parallel processes: Adler et al. [2] consider parallel versions of GREEDY[d] and explore a trade-off between the maximum load and the number r of communication rounds between balls and the randomly chosen bins. Roughly speaking, their result shows that bounds which are close to the classical (sequential) processes can only be achieved if r is close to the maximum load [2]. The authors give a lower bound on the maximum load dependent on r, and Stemann [20] provides a matching upper bound via a collision-based protocol. In the same context, Adler et al. [2] also consider a protocol, called THRESHOLD[T], which bears resemblance to ours. In each round every ball chooses a random bin and every bin rejects all balls above this threshold. The authors prove that, w.h.p., THRESHOLD[1] terminates after at most $\ln\ln(n)+\mathrm{O}(1)$ steps which implies that the maximum load is also at most $\ln\ln(n)+\mathrm{O}(1)$. Lenzen et al. [15] consider the heavily-loaded case with $m>n$ balls. They present a simple parallel threshold algorithm that obtains a maximal load of $m/n+\mathrm{O}(1)$ w.h.p. using $\mathrm{O}(\log\log(m/n)+\log^* n)$ communication rounds.

b) Infinite Sequential Processes: Azar et al. [4] consider an infinite, sequential version of their GREEDY[d] starting with n balls arbitrarily assigned to n bins. In each round one random ball is reallocated using the d-choice process. For any $t>cn^2\log\log n$, the maximum load at time t is $\ln\ln(n)/\ln d+\mathrm{O}(1)$ w.h.p. Cole et al. [12] show that the maximum load is $\log\log n/\log d+O(1)$ for a polynomial number of steps, w.h.p., even if an adversary specifies the deletion sequence. They also show that at an arbitrary point of time the maximum load is at most $4\log\log n$ if the same n balls are re-inserted again and again and the random choices of each ball are fixed after its first insertion. Vöcking [21] extends the analysis of [4] by allowing arbitrary sequences of insertions and deletions. If the maximum number of balls never exceeds hn, the maximum load at any time is w.h.p. $\log_d\ln n+O(h)$ in the case of GREEDY[d] and $\ln\ln n/(d\ln\varphi_d)+O(h)$ in the case of ALWAYS-GO-LEFT[d].

c) Infinite Parallel Processes: Becchetti et al. [5] consider a parallel process where the number of balls is fixed to n. In each round one ball is chosen from every non-empty bin and reallocated to a randomly chosen bin (one choice per ball). The authors show that w.h.p. starting from an arbitrary configuration, it takes $\mathrm{O}(n)$ rounds to reach a configuration with maximum load $\mathrm{O}(\log n)$. Adler et al. [1] consider a system where in each round $m<n/(3de)$ balls are allocated. Bins have a FIFO queue, and each arriving ball is stored in the queue of d random bins. After each round, every non-empty bin deletes its first ball and initiates the deletion of that ball's copies from the other bins. It is shown that the expected waiting time is constant and the maximum waiting time is $\log\log(n)/\log d+\mathrm{O}(1)$ w.h.p. The restriction $m<n/(3de)$ is the major drawback of this process. Berenbrink et al. [8] conduct a further study, based on differential methods and experiments. The balls' arrival times are binomially distributed with parameters n and $\lambda=m/n$. Their results indicate a stable behavior for $\lambda\leq 0.86$. A similar model was considered by Mitzenmacher [16], who considers ball arrivals as a Poisson stream of rate λn for $\lambda<1$. It is shown (again using differential equations) that the 2-choice process reduces the waiting time exponentially compared to the 1-choice process. Berenbrink et al. [7] study the d-choice process in a scenario where m balls are allocated to n bins in batches of size n each. The authors show that the load of every bin is $m/n\pm\mathrm{O}(\log n)$ w.h.p.

Berenbrink et al. [10] study a round-based parallel version of the GREEDY[d] distribution scheme. $m=\lambda n$ new balls arrive per round and at the end of the round each non-empty bin deletes one of its balls. The measure used for the comparison is the number of balls in that bin at the beginning of the round; balls of the current batch are not taken into account. The authors show a strong self-stabilizing property: for *any* arrival rate $\lambda<1$, the system load is time-invariant. In particular, they show that the maximum bin load (or waiting time) at an arbitrary time is, w.h.p., bounded by $\mathrm{O}\left(\frac{1}{1-\lambda}\cdot\log\frac{n}{1-\lambda}\right)$ for the 1-choice process, and $\mathrm{O}\left(\log\frac{n}{1-\lambda}\right)$ for the 2-choice process. The load properties of the 1-choice process yield a super-linear maximum load for $\lambda=1-1/n$. The maximum load in the 2-choice process remains logarithmic for any $\lambda\leq 1-1/\mathrm{poly}(n)$.

B. New Results

The major aim of this paper, in contrast to [10], is to minimize the waiting time of the balls, rather than (just) the total number of balls in the system. The new process, which we call CAPPED(c,λ), works in parallel rounds. It limits the number of balls that a bin can store by a capacity c, and a *pool* contains all the balls from the previous round not accepted by any bin (which in a real scenario could stay with the clients). In every round λn new balls are added to the

[1] The expression *with high probability* refers to a probability of $1-n^{-\Omega(1)}$.

pool and each ball from the pool randomly chooses one bin. In turn, every bin accepts as many balls as possible, preferring the oldest balls among its requests. At the end of a round, every bin deletes a ball in a FIFO manner.

In this paper we present a simple analysis for this process which is based on simple combinatorial arguments and Chernoff bounds (compared to differential equation arguments [8, 16], potential function arguments [10] or infinite witness trees [1] which were used before to analyze infinite allocation processes where the number of balls is not fixed). To illustrate the main idea of our proof we first consider the case for $c=1$. We show that w.h.p. the total number of balls in the pool is at most $2\cdot\ln\big(1/(1-\lambda)\big)\cdot n+4n$ and that the waiting time of any ball generated in an arbitrary step t is at most $\big(2\ln\big(1/(1-\lambda)\big)+4\big)/(1-e^{-1})+\log\log n+\mathrm{O}(1)$. We then generalize this to $c\geq 1$ and show that w.h.p. the total number of balls in the pool is at most $4\cdot c^{-1}\cdot\ln\big(1/(1-\lambda)\big)\cdot n+\mathrm{O}\big(c\cdot n\big)$. The waiting time of any ball generated in any round t is w.h.p. at most $\big(4\cdot\ln\big(1/(1-\lambda)\big)\big)/\big(c\cdot(1-1/e)\big)+\log\log n+\mathrm{O}(c)$. Note that the result of [10] implies that both of the analyzed processes are *self-stabilizing*, i.e., *positive recurrent*, which means that the expected return time to a typical low-load situation is finite. Positive recurrence is a standard notion of stability and states that the system load is time-invariant. For ergodic Markov chains it implies the existence of a unique stationary distribution.

For large values of λ and c not too large, both the number of balls in the pool and the waiting time decrease by a factor of essentially c. Even for an injection rate of $\lambda=1-1/n$ the pool size is at most $\mathrm{O}\big(n\log n\big)$ and the waiting time is $\mathrm{O}\big(\log n\big)$. Note that, for constant λ, both results from [10] give a bound of $\Theta(\log n)$ on the waiting time, whereas our new results show a bound of $\log\log n+\mathrm{O}\big(1\big)$. The "long" waiting time in [10] is due to the fact that there is no coordination between the balls generated in the same round. As a result, up to $\mathrm{O}\big(\log n\big)$ of these balls can be allocated to one bin. Compared to our process, an advantage of the GREEDY[d] process from [10] is that it only needs d random choices to allocate a ball. However, our results imply that, for constant λ, the average number of random choices per ball is still constant.

II. Model and Definition of the Allocation Process

The number of bins is n and we assume that the system is initially empty. The process works in parallel rounds and is defined as follows. At the beginning of every round λn new balls are generated, where λ denotes the *injection rate*.[2] We assume that λn is an integer. A ball that is generated in round t is labeled with t, and the *age* of a ball in round t is defined as t minus the ball's label. The generated balls are added into a *pool*, which contains balls that could not be allocated in the previous round. We use $M(t)$ to denote the set of balls in the pool at the end of round $t\in\mathbb{N}$ and define $m(t):=|M(t)|$. Every bin has capacity c, i.e., it can store up to c balls. The load (number of balls) of bin i at the end of round t is denoted by $\ell_i(t)$. After the ball generation, in parallel every ball randomly chooses a bin and sends an allocation request to the bin. A bin i that receives ν_i request accepts the $\min\{c-\ell_i(t),\nu_i\}$ "oldest" balls (the ones with the smallest label), ties are broken arbitrarily. These balls are subsequently removed from the pool. At the end of round t, every bin deletes a ball following a FIFO-scheme.

The *waiting time* of a ball that is deleted in round t is defined as the ball's age in round t. For $a,b\in\mathbb{N}$ with $a\leq b$ we use the notation $[a;b]:=\{a,a+1,\ldots,b\}$ for integral intervals and the shorthands $[a]:=[1;a]$. Similar notation is used for open and half-open intervals, for example $(a;b]=\{a+1,a+2,\ldots,b\}$. We let our process start in round 1 and set $m(0)=0$ and $\ell_i(0)=0\,\forall i\in[n]$. In the following we will use CAPPED(c,λ) for the process with injection rate λ and capacity c. Note that $c=\infty$ implies no capacity limit and therefore CAPPED(∞,λ) is identical to GREEDY[1]. See Algorithm 1 for a formal definition of one round of CAPPED(c,λ).

Algorithm 1 Round t of CAPPED(c,λ)

Generate λn new balls and add them to the pool
Each ball b from the pool picks a bin independently and uniformly at random
Each bin i that receives ν_i requests accepts the oldest $\min\{c-\ell_i(t-1),\nu_i\}$ balls (ties broken arbitrarily); these balls are removed from the pool and inserted into a queue
Each non-empty bin deletes the first ball in its queue

III. Warm-up: Analysis for Unit Capacity

In this section we analyze the pool size and the waiting time of CAPPED$(1,\lambda)$. This special case allows us to convey the key ideas of the analysis while still hiding some of the more technical aspects we have to deal with for general capacities $c\in\mathbb{N}$. The major property that simplifies the analysis for unit capacity is that each round starts with empty bins.

Theorem 1. *Consider* CAPPED$(1,\lambda)$ *for* $0\leq\lambda\leq 1-1/n$ *with* $\lambda n\in\mathbb{N}$. *Let* $t\in\mathbb{N}$ *be an arbitrary round.*

1) *With probability at least* $1-2^{-2n}$, *the size of the pool at the end of round* t *is less than*

$$2\cdot\ln\left(\frac{1}{1-\lambda}\right)\cdot n+4n.$$

2) *With probability at least* $1-n^{-2}$, *the waiting time of any ball generated in round* t *is at most*

$$\frac{2\cdot\ln\left(\frac{1}{1-\lambda}\right)+4}{1-e^{-1}}+\log\log n+\mathrm{O}\big(1\big).$$

Note that the constants in the above result are not optimized but result from the urge to keep the analysis simple and clean.

The remainder of this section proves Theorem 1. The key part of the analysis is the proof of Statement 1, which is given in Section III-A. Afterward, we show in Section III-B how Statement 2 follows by standard arguments with the help of Statement 1.

[2] Our results can be adjusted to a probabilistic ball generation process with n generators and an expected injection rate of λ, but this would, for no real gain, quite unnecessarily complicate the presentation.

A. Analysis of the Pool Size

Our goal is to prove Statement 1 of Theorem 1, namely that for any $t \in \mathbb{N}$ it is extremely unlikely that the size $m(t)$ of the pool at the end of round t is larger or equal $2m^*$, where $m^* := \ln\big(1/(1-\lambda)\big) \cdot n + 2n$.

In the following, we say the *deletion attempt* of a bin i during round t is *successful* if bin i receives at least one ball (and, thus, deletes a ball) during round t. Otherwise, we say the deletion attempt of bin i during round t *failed*.

The idea of our proof is as follows. Fix a round t and assume $m(t) \geq 2m^*$. There must be a round $s \leq t$ such that $m(s-1) \leq m^*$ and $m(t') > m^*$ for all $t' \in [s;t]$. That is, during the $\Delta_{s,t} := t-s+1$ rounds from s to t the pool size grows from at most m^* to at least $2m^*$ without falling back to m^* in between. Such an increase is possible only if enough of the $\Delta_{s,t} \cdot n$ deletion attempts during these $\Delta_{s,t}$ rounds failed. More exactly, if $X_{s,t}$ denotes the total number of failed deletion attempts during $[s;t]$, we will show that an increase of at least m^* during $[s;t]$ implies $X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n$. Using that $m(t') \geq m^*$ for all $t' \in [s;t]$, we then bound the probability that a given deletion attempt in round t' fails by $(1-1/n)^{m(t')} \leq e^{-m(t')/n} \leq e^{-m^*/n}$. This yields $\mathbb{E}[X_{s,t}] \leq \Delta_{s,t} \cdot n \cdot e^{-m^*/n}$. Realizing that $X_{s,t}$ is the sum of $\Delta_{s,t} \cdot n$ indicator variables (indicating whether a given deletion attempt failed) suggests to bound the probability that $X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n$ via a simple Chernoff bound.

Unfortunately, the above argumentation relies crucially on our choice of s and t giving $m(t') > m^*$ for all $t' \in [s;t]$. Such a conditioning on future events might skew the probability distribution. We will use a coupling with a process that behaves as CAPPED$(1,\lambda)$ but always keeps the pool at size at least m^* to deal with this.

As outlined above, our analysis is based on a related process, MODCAPPED$(1,\lambda)$, which behaves like CAPPED$(1,\lambda)$ with only one exception: instead of generating λn balls at the beginning of any round $t \in \mathbb{N}$, it generates $\max\{\lambda n, m^* - m(t-1)\}$ balls. As a result, this modified process is guaranteed to throw in each round *at least* m^* balls into the bins. To differentiate between the pool sizes of both processes, in the following we use $m^C(t)$ and $m^M(t)$ for the pool size of CAPPED$(1,\lambda)$ and of MODCAPPED$(1,\lambda)$ at the end of round t, respectively. Using a simple coupling, the following lemma shows that at any time the pool size of CAPPED$(1,\lambda)$ is stochastically dominated by the pool size of MODCAPPED$(1,\lambda)$.

Lemma 1. *At the end of any round $t \in \mathbb{N}$, the pool size $m^C(t)$ of* CAPPED$(1,\lambda)$ *is stochastically dominated by the pool size $m^M(t)$ of* MODCAPPED$(1,\lambda)$.

Proof. We prove the stochastic dominance via induction. Initially, we have $m^C(0) = 0 = m^M(0)$. Now consider round $t \in \mathbb{N}$ and assume $m^C(t-1) \leq m^M(t-1)$. In round t, CAPPED$(1,\lambda)$ throws $\nu^C(t) := m^C(t-1) + \lambda n$ balls. Similarly, MODCAPPED$(1,\lambda)$ throws $\nu^M(t) := m^M(t-1) + \max\{\lambda n, m^* - m^M(t-1)\}$ balls. Combining both with the induction hypothesis for $t-1$ immediately yields $\nu^M(t) \geq \nu^C(t)$.

We now define the coupling for round t: number the balls thrown by CAPPED$(1,\lambda)$ from 1 to $\nu^C(t)$ and those thrown by MODCAPPED$(1,\lambda)$ from 1 to $\nu^M(t)$. W.l.o.g., we assume that, when allocating and deleting balls, both processes prefer balls with a smaller number. The first $\nu^C(t)$ balls of MODCAPPED$(1,\lambda)$ use the same random bin choices as their counterparts in CAPPED$(1,\lambda)$. The remaining $\nu^M(t) - \nu^C(t) \geq 0$ balls of MODCAPPED$(1,\lambda)$ uniformly and independently choose bins at random.

To finish the induction, it remains to prove $m^C(t) \leq m^M(t)$. By the above coupling, MODCAPPED$(1,\lambda)$ and CAPPED$(1,\lambda)$ delete the same balls among their first $\nu^C(t)$ balls. Additionally, MODCAPPED$(1,\lambda)$ may delete some of its $\nu^M(t) - \nu^C(t)$ extra balls. This yields $m^C(t) \leq m^M(t)$. □

With Lemma 1, it is sufficient to prove Statement 1 of Theorem 1 for MODCAPPED$(1,\lambda)$ instead of for CAPPED$(1,\lambda)$:

Lemma 2. *Consider* MODCAPPED$(1,\lambda)$ *with $0 \leq \lambda \leq 1 - 1/n$ and let $t \in \mathbb{N}$ be an arbitrary round. With probability at least $1 - 2^{-2n}$, the size of the pool at the end of round t is less than $2 \cdot \ln\big(1/(1-\lambda)\big) \cdot n + 4n$.*

Proof. For $t \in \mathbb{N}$ let $m^M(t)$ denote the pool size of MODCAPPED$(1,\lambda)$ at the end of round t. Remember that $m^* = \ln\big(1/(1-\lambda)\big) \cdot n + 2n$. Consider $s,t \in \mathbb{N}$ with $s \leq t$. We define the events

$$\mathcal{A}_t := \{m^M(t) \geq 2m^*\} \qquad \text{and}$$
$$\mathcal{B}_{s,t} := \{m^M(s-1) \leq m^* \wedge \forall t' \in [s;t] \colon m^M(t') > m^*\}.$$

Our goal is to prove $\Pr[\mathcal{A}_t] \leq 2^{-2n}$. For $\mathcal{A}_t$ to occur there must be exactly one $s \leq t$ for which $\mathcal{B}_{s,t}$ holds, and so we can partition $\mathcal{A}_t$ via $\mathcal{A}_t = \biguplus_{s \in [t]} \mathcal{A}_t \cap \mathcal{B}_{s,t}$. Define $\Delta_{s,t} := t-s+1$ as the number of rounds in the integral interval $[s;t]$. Let $X_{s,t}$ denote the number of failed deletion attempts during $[s;t]$. Our strategy is to first bound each $\Pr[\mathcal{A}_t \cap \mathcal{B}_{s,t}]$ by showing that the event $\mathcal{A}_t \cap \mathcal{B}_{s,t}$ occurring implies the event $X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n$ also occurring; in terms of event sets, $(\mathcal{A}_t \cap \mathcal{B}_{s,t}) \subseteq (X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n)$. This will yield $\Pr[\mathcal{A}_t \cap \mathcal{B}_{s,t}] \leq \Pr[X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n]$.

Fix $s,t \in \mathbb{N}$ with $s \leq t$ and assume event $\mathcal{A}_t \cap \mathcal{B}_{s,t}$ occurred. By definition of $\mathcal{B}_{s,t}$, MODCAPPED$(1,\lambda)$ generates exactly $\Delta_{s,t} \cdot \lambda n$ balls during $[s;t]$, and there are exactly $\Delta_{s,t} \cdot n$ deletion attempts (one per bin per round) which, if successful, each remove one of the generated balls. The pool size therefore increases by exactly $\Delta_{s,t} \cdot \lambda n - (\Delta_{s,t} \cdot n - X_{s,t})$ from round s to t. Consequently, for the pool size to increase from at most m^* to at least $2m^*$ during $[s;t]$, we must have $\Delta_{s,t} \cdot \lambda n - (\Delta_{s,t} \cdot n - X_{s,t}) \geq m^*$ or, equivalently, $X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n$. This gives

$$\Pr[\mathcal{A}_t \cap \mathcal{B}_{s,t}] \leq \Pr[X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n]. \qquad (1)$$

We now fix arbitrary rounds $s,t \in \mathbb{N}$ with $s \leq t$ and proceed to bound the probability $\Pr[X_{s,t} \geq m^* + (1-\lambda) \cdot \Delta_{s,t} \cdot n]$. Observe that $X_{s,t} \leq \sum_{t' \in [s,t]} \sum_{i \in [n]} X_{t'}(i)$, where $X_{t'}(i)$ are indicator variables with $X_{t'}(i) = 1$ if and only if none of the first m^* balls from round t' are thrown into bin i. Let $\tilde{X}_{s,t} := \sum_{t' \in [s,t]} \sum_{i \in [n]} X_{t'}(i)$. We have

$\Pr[X_{t'}(i)=1] \le (1-1/n)^{m^*} \le e^{-m^*/n} = e^{-2}\cdot(1-\lambda)$, which, by our choice of $m^* = \ln\big(1/(1-\lambda)\big)\cdot n + 2n$, yields $\mathbb{E}\Big[\tilde{X}_{s,t}\Big] \le \Delta_{s,t}\cdot n\cdot e^{-m^*/n} \le (1-\lambda)\cdot\Delta_{s,t}\cdot n/(2e)$. Note that the variables $(X_{t'}(i))_{t'\in[s,t],i\in[n]}$ are negatively associated. Indeed, this follows easily for fixed t', as $(X_{t'}(i))_{i\in[n]}$ correspond to the empty-bins indicator variables [13, Theorem 46] and can be extended to different rounds by the independence of the first m^* balls thrown in different rounds. Thus, we can define $R := m^* + (1-\lambda)\cdot\Delta_{s,t}\cdot n \ge 2e\cdot\mathbb{E}[X_{s,t}]$ and apply the Chernoff bound from Lemma 8 to get

$$\begin{aligned}\Pr[X_{s,t} \ge m^* + (1-\lambda)\cdot\Delta_{s,t}\cdot n] &= \Pr[X_{s,t}\ge R]\\ &\le \Pr\Big[\tilde{X}_{s,t}\ge R\Big] \le 2^{-R} = 2^{-m^*}\cdot 2^{-(1-\lambda)\cdot\Delta_{s,t}\cdot n}.\end{aligned} \tag{2}$$

We are now ready to prove $\Pr[\mathcal{A}_t] \le 2^{-2n}$. Using the partitioning $\mathcal{A}_t = \biguplus_{s=1}^{t}\mathcal{A}_t\cap\mathcal{B}_{s,t}$ and combining Equations (1) and (2) yields

$$\begin{aligned}&\Pr[\mathcal{A}_t]\\ &= \sum_{s\in[t]}\Pr[\mathcal{A}_t\cap\mathcal{B}_{s,t}] \le \sum_{s\in[t]}\Pr[X_{s,t}\ge m^*+(1-\lambda)\cdot\Delta_{s,t}\cdot n]\\ &\le \sum_{s\in[t]} 2^{-m^*}\cdot 2^{-(1-\lambda)\cdot\Delta_{s,t}\cdot n} \le 2^{-m^*}\cdot\sum_{i\in[t]}\big(2^{-1}\big)^i \le 2^{-m^*},\end{aligned}$$

where the penultimate inequality uses $\lambda \le 1-1/n$ and reverses the summation, and the last inequality bounds the sum via a geometric series. From this, the desired result follows by noting that $m^* = \ln\big(1/(1-\lambda)\big)\cdot n + 2n \ge 2n$. □

B. Analysis of the Waiting Time

In this section we prove Statement 2 of Theorem 1. Remember that $M(t)$ denotes the set of balls in the pool at the end of round t and $m(t) = |M(t)|$. For $t' \ge t$ we define $M(t,t')$ as the balls from $M(t)$ that are still in the pool at the end of round t' and let $m(t,t') := |M(t,t')|$. Note that $m(t,t')$ is monotonically decreasing in t'.

Consider a ball $b \in M(t)$ that was generated in round t. Note that when competing for allocation in a given bin during a round $t' \ge t$, this ball may be delayed by balls from $M(t,t')$ (which have been generated during some round $\le t$ and, thus, are at least as old as b) but it cannot be delayed by any ball generated after round t. Thus, to bound the waiting time of b we analyze how long it takes until all balls from $M(t)$ become allocated and, thus, deleted.

To this end, we first note that, by Statement 1 of Theorem 1, the set $M(t)$ has, with exponentially high probability, size $m(t) \le 2\cdot\ln\big(1/(1-\lambda)\big)\cdot n + 4n$. Now, given a round t with pool size $m(t) \le 2\cdot\ln\big(1/(1-\lambda)\big)\cdot n + 4n$, the next lemma states that, also with an exponentially high probability, the number of surviving balls from $M(t)$ decreases to $2n$ within $\Delta := m(t)/(n-n/e)$ rounds.

Lemma 3. *Consider a round $t \in \mathbb{N}$ with $m(t) \le n^2$. Let $\Delta := m(t)/(n-n/e)$. With probability at least $1-e^{-\Omega(n)}$, we have $m(t,t+\Delta) \le 2n$.*

Proof. Fix round $t\in\mathbb{N}$ with $m(t)\le n^2$. Consider a round $t' > t$ and assume $m(t,t'-1) > 2n$. For a bin $i\in[n]$ let X_i denote the indicator variable which is 1 if and only if bin i does not receive a ball in round t' and define $X := \sum_{i\in[n]} X_i$. Since during round t' at least $m(t,t'-1) > 2n$ balls are thrown, we have $\Pr[X_i=1] \le (1-1/n)^{m(t,t'-1)} \le e^{-m(t,t'-1)/n} < e^{-2}$, which in turn yields $\mu := \mathbb{E}[X] < n\cdot e^{-2}$. We apply a standard tail bound to the number of empty bins (Lemma 10, with $\lambda = n/e - n/e^2$) to get

$$\Pr[X\ge n/e] \le \Pr[X\ge\mu+\lambda] \le e^{-\frac{\lambda^2\cdot(n-1/2)}{n^2-\mu^2}} = e^{-\Omega(n)}. \tag{3}$$

In other words, with only exponentially small probability fewer than n/e bins do not receive a ball in round t', which implies that more than $n-n/e$ bins receive at least one ball and, thus, delete at least one ball at the end of round t'.

Note that $\Delta = m(t)/(n-n/e)$ is an upper bound on the number of rounds it takes to reduce $m(t)$ balls to $2n$ balls if in each round at least $n-n/e$ of them are deleted. Define the stopping times $T := \min\{t'\in\mathbb{N} \mid t' \ge t \wedge m(t,t') \le 2n\}$ and $T' := \min\{T, t+\Delta\}$. Using $m(t)\le n^2$ and a simple union bound over Equation (3) for the $T'-t \le \Delta = \mathrm{O}(n)$ rounds from $t+1$ to T' yields that, after T' rounds, either $T' < t+\Delta$ (which implies $T = T' < t+\Delta$ by definition of T') or that, with probability at least $1-\Delta\cdot e^{-\Omega(n)} = 1-e^{-\Omega(n)}$, each of the Δ rounds from t to $T' = t+\Delta$ deleted $n-n/e$ from the $m(t)$ balls, which (by our choice of Δ) implies $m(T') \le 2n$. □

Now given that there are $m(t,t+\Delta)\le 2n$ survivors from $M(t)$, the next lemma states that, again with an very high probability, the number of surviving balls from $M(t)$ decreases further to $n/(2e)$ within 19 additional rounds.

Lemma 4. *Consider two rounds $t_1 \le t_2$ with $m(t_1, t_2) \le 2n$. With probability at least $1 - e^{-\Omega(n)}$, we have $m(t_1, t_2+19) \le n/(2e)$.*

Proof. Fix a round $t' > t_2$ and assume $m(t_1,t'-1) > n/(2e)$. Let the indicator variable X_i be 1 if and only if bin $i \in [n]$ does not receive a ball in round t' and define $X := \sum_{i\in[n]} X_i$. Analogously to the proof of Lemma 3, we get $\mathbb{E}[X] < n\cdot e^{-1/(2e)} =: \mu$. Although the X_i are not independent they are negatively associated, and we can apply a standard Chernoff bound (Lemma 9) with $\delta := 1/20$ to get

$$\Pr[X\ge 9n/10] \le \Pr[X\ge(1+\delta)\cdot\mu] \le e^{-\Omega(n)}. \tag{4}$$

In other words, with only exponentially small probability fewer than $9n/10$ bins do not receive a ball in round t', which implies that more than $n/10$ bins receive at least one ball and, thus, delete at least one ball at the end of round t'.

Note that deleting $n/10$ balls for $\Delta := 19$ rounds is enough to reduce $m(t_1,t_2)$ from at most $2n$ to $2n - \Delta\cdot n/10 < n/(2e)$. Thus, using a stopping time argument analogously to the proof of Lemma 3, we get that with probability at least $1 - 19\cdot e^{-\Omega(n)} = 1-e^{-\Omega(n)}$ we have $m(t_1,t_2+\Delta) \le n/(2e)$. □

Finally, the next lemma shows that, with high probability, the remaining $n/(2e)$ survivors from $M(t)$ are deleted in an additional $\log\log n + \mathrm{O}(1)$ rounds.

Lemma 5. *Fix two rounds $t \le t'$ with $m(t,t') \le n/(2e)$. We have $m(t,t'+\log\log n+\mathrm{O}(1))=0$ with probability at least $1-n^{-2}$, that is, every ball from $M(t)$ is allocated by the end of round $t'+\log\log n+\mathrm{O}(1)$.*

Proof Sketch. Our proof uses layered induction and is basically the same as the analysis of GREEDY[2] (see [4, Theorem 4]). The only difference is that the induction in our proof goes over rounds instead of the balls' heights. This is merely a cosmetic difference and does not affect the idea and structure of the proof. In the following we briefly describe how to set up the induction, after which the proof then follows exactly that of [4, Theorem 4].

For $i \in \mathbb{N}_0$, let $B_i := M(t,t'+i)$ be the subset of balls from $M(t)$ that are not allocated at the end of round $t'+i$. Define $\beta_0 := n/(2e)$ and $\beta_{i+1} = e \cdot \beta_i^2/n$ for $i \in \mathbb{N}$. We let $i^* \in \mathbb{N}$ be minimal such that $\beta_{i^*}^2/n \le 4\ln(n)/n$. For $0 \le i \le i^*$ let ξ_i be the event that $|B_i| \le \beta_i$. The proof now proceeds to bound the probability of the event $\neg\xi_{i+1} \wedge \xi_i$, which can be achieved exactly as in [4, Theorem 4]. The result then follows using a simple union bound. □

The lemmas above now finally allow us to prove Statement 2 of Theorem 1.

Proof of Statement 2 from Theorem 1. The proof uses Lemmas 3 to 5 as well as Statement 1 of Theorem 1. First, Statement 1 of Theorem 1 shows that $m(t) \le 2\cdot\ln\big(1/(1-\lambda)\big)\cdot n + 4n$ with probability at least $1-2^{-2n}$. From Lemma 3 it follows that for $\Delta := m(t)/(n-n/e)$ we have $m(t,t+\Delta) \le 2n$ with probability at least $1-e^{-\Omega(n)}$. From Lemma 4 it follows that after 19 additional rounds we have $m(t,t+\Delta+19) \le n/(2e)$, also with probability at least $1-e^{-\Omega(n)}$. From Lemma 5 it follows that, with probability at least $1-n^{-2}$, all balls of $M(t)$ are deleted after $\log\log n+\mathrm{O}(1)$ additional rounds. Thus, with a probability of $1-n^{-\Omega(1)}$, the waiting time is bounded by

$$\Delta+19+\log\log n+\mathrm{O}(1) \le \frac{2\cdot\ln\left(\frac{1}{1-\lambda}\right)+4}{1-e^{-1}}+\log\log n+\mathrm{O}\big(1\big),$$

giving us the desired statement. □

IV. Analysis for Arbitrary Capacities

In this section we extend the analysis from Section III to arbitrary capacities $c \in \mathbb{N}$. Our main result for this section is Theorem 2, which closely resembles Theorem 1 except for some slightly weaker constants.

Theorem 2. *Consider* CAPPED(c,λ) *for* $c \in \mathbb{N}$ *and* $0 \le \lambda \le 1-1/n$ *with* $\lambda n \in \mathbb{N}$. *Let* $t \in \mathbb{N}$ *be an arbitrary round.*

1) *With probability at least $1-2^{-2n}$, the size of the pool at the end of round t is less than*
$$\frac{4}{c}\cdot\ln\left(\frac{1}{1-\lambda}\right)\cdot n+\mathrm{O}\big(c\cdot n\big).$$
2) *With probability at least $1-n^{-2}$, the waiting time of any ball generated in round t is at most*
$$\frac{4\cdot\ln\left(\frac{1}{1-\lambda}\right)}{c\cdot(1-e^{-1})}+\log\log n+\mathrm{O}\big(c\big).$$

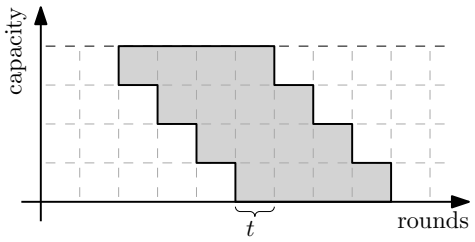

Figure 1. Region that affects or is affected by round t.

In Statement 1 of the theorem, bounding the pool size, we consider the $\mathrm{O}\big(c\cdot n\big)$ term to mainly be an artifact of our simple analysis using the coupling with MODCAPPED(c,λ). In Statement 2, bounding the waiting time, the term $4\cdot\ln\left(\frac{1}{1-\lambda}\right)/\big(c\cdot\big(1-\frac{1}{e}\big)\big)+\log\log n$ estimates the number of rounds until a ball is allocated into the buffer of a bin, whereas the $\mathrm{O}\big(c\big)$ is for the time until a ball is deleted from a buffer; in total the waiting time does not strictly decrease with increasing c. We believe this $\mathrm{O}\big(c\big)$ term not to be an artifact of our analysis but rather stemming largely from balls having a good chance to be allocated to a bin already having its buffer filled up to some constant fraction of its capacity. We observe this happening experimentally even if the pool size is relatively small.

a) Analysis Outline: The basic idea for the analysis of Theorem 2 is the same as for $c=1$ (see also the proof idea in Section III). However, while for $c=1$ each round starts with empty bins, the load situation for general c may vary widely from round to round and introduce additional dependencies.

To get some intuition about these dependencies, fix some round t. Note that every ball allocated in round t will be deleted during a round in $[t;t+c)$, and any ball deleted in round t was allocated during a round in $(t-c;t]$. We can imagine the region that affects and the region that is affected by round t as the marked area in Figure 1. We will refer to these regions as *buffers* and partition a bin over time into different buffers that overlap each other (see Figure 2).

Observe that overlapping buffers of a bin have certain dependencies. In particular, they compete for balls selecting that bin. To deal with these dependencies, we extend the coupling used for $c=1$. Our extended coupling will ensure that for each active buffer at least $m^*/2$ balls try to allocate themselves to that buffer, independently of any other buffer.

We define the modified process MODCAPPED(c,λ) (which, for $c=1$, simplifies to the modified process from Section III) in Section IV-A. The analysis in Section IV-B first proves that the pool size of CAPPED(c,λ) is stochastically dominated by the pool size of MODCAPPED(c,λ) (Lemma 6). Then it is sufficient to prove Statement 1 of Theorem 2 for the modified process (Lemma 7). Statement 2 of Theorem 2 in turn can be proved similarly as for $c=1$, using the bound on the pool size for arbitrary c instead of that for $c=1$. See Section IV-C for the details.

A. Definition of the Coupled Process

This section defines MODCAPPED(c,λ), a variant of CAPPED(c,λ) which avoids some of the dependencies that

plague CAPPED(c,λ). We partition time into *phases* of length c such that phase $j \in \mathbb{N}_0$ corresponds to the integral time interval $I_j := [c \cdot j; c \cdot (j+1) - 1]$. For any $j \in \mathbb{N}_0$, each bin has a *buffer* j with a time-dependent capacity $c_j(t) \in \{0,1,...,c\}$. We say buffer j is *active* in round t if $c_j(t) > 0$. The capacity is defined such that buffer j is active only during the phases $j-1$ and j. More precisely, it starts at capacity 0 in the first round of phase $j-1$ and increases its capacity by 1 per round until capacity c is reached in the first round of phase j. Subsequently, the buffer's capacity decreases again by 1 per round, reaching 0 after the end of phase j (see also Figure 2). Formally, we have

$$c_j(t) = \begin{cases} 0 & \text{if } t \notin (I_{j-1} \cup I_j), \\ t - (j-1) \cdot c & \text{if } t \in I_{j-1}, \text{ and} \\ (j+1) \cdot c - t & \text{if } t \in I_j. \end{cases} \tag{5}$$

Note that for any bin and any round t, at most two buffers are active (namely buffers $\lfloor t/c \rfloor$ and $\lceil t/c \rceil$), and the capacity of those active buffers sums up to the bin's capacity c. We refer to $\lceil t/c \rceil$ as the bin's *red* buffer in round t. If $\lfloor t/c \rfloor \neq \lceil t/c \rceil$, we refer to $\lfloor t/c \rfloor$ as the bin's *blue* buffer in round t.

We are now ready to define MODCAPPED(c, λ). It deviates from CAPPED(c, λ) in two ways: *ball generation* and *ball allocation/deletion*. For the ball generation, we proceed as in Section III. Instead of generating λn balls at the beginning of each round $t \in \mathbb{N}$, MODCAPPED(c, λ) generates $\max\{\lambda n, m^* - m(t-1)\}$ balls, where $m^* := 2c^{-1} \cdot \ln\big(1/(1-\lambda)\big) \cdot n + 6c \cdot n$.

The ball allocation/deletion is more involved. For a given round t let $\nu(t)$ denote the number of balls after that round's ball generation and note that $\nu(t) \geq m^*$. We partition the balls (arbitrarily) into $\lceil \nu(t)/2 \rceil$ balls with *preference red* and $\lfloor \nu(t)/2 \rfloor$ balls with *preference blue*. Then, every ball chooses one of the n bins independently and uniformly at random. Among all balls that choose a given bin, the bin greedily selects balls and distributes them between the currently active buffers, maximizing the number of satisfied (color) preferences (a ball's preference is satisfied if it is assigned to a buffer whose color matches the ball's preference) without exceeding the buffers' capacities. Note that this might result, e.g., in balls with a blue preference being assigned to the red buffer, but only if the blue buffer is already full. At the end of the round, any non-empty red buffer deletes a ball.

B. Analysis of the Pool Size

As in Section III, we first show that at any time the pool size of CAPPED(c,λ) is stochastically dominated by the pool size of MODCAPPED(c,λ). Again, to differentiate between the pool size of both processes, we use $m^C(t)$ and $m^M(t)$ to denote the pool size of CAPPED(c,λ) and of MODCAPPED(c,λ) at the end of round t, respectively. Similarly, for a bin i and round t, we use $\ell_i^C(t)$ and $\ell_i^M(t)$ to denote the number of balls allocated to bin i at the end of round t.

Lemma 6. *At the end of any round $t \in \mathbb{N}$, the pool size $m^C(t)$ of* CAPPED(c,λ) *is stochastically dominated by the pool size $m^M(t)$ of* MODCAPPED(c,λ).

Proof. Similar to the proof of Lemma 1, we show the stochastic dominance by inductively defining a coupling that maintains a suitable invariant. More exactly, given $t \in \mathbb{N}$ with $m^C(t-1) \leq m^M(t-1)$ as well as $\ell_i^C(t-1) \leq \ell_i^M(t-1)$ for each bin i, we define a coupling for the random choices in round t that ensures $m^C(t) \leq m^M(t)$ as well as $\ell_i^C(t) \leq \ell_i^M(t)$ for each bin i. Once this induction step is proven, the lemma's statement follows immediately by noticing that the base case holds trivially, since $m^C(0) = 0 = m^M(0)$ as well as $\ell_i^C(0) = 0 = \ell_i^M(0)$ for each bin i.

Before we prove the induction step, we introduce some useful notation. We use $\nu^C(t) := m^C(t-1) + \lambda n$ to denote the number of balls thrown by CAPPED(c, λ) in round t. Similarly, $\nu^M(t) := m^M(t-1) + \max\{\lambda n, m^* - m^M(t-1)\}$ is the number of balls thrown by MODCAPPED(c, λ) in round t. For a bin i, the values $\nu_i^C(t)$ and $\nu_i^M(t)$ are the number of balls thrown into bin i by CAPPED(c, λ) and MODCAPPED(c,λ), respectively. Note that $\nu^\bullet(t) = \sum_{i \in [n]} \nu_i^\bullet(t)$ for $\bullet \in \{C, M\}$. Finally, we use $\alpha_i^C(t)$ and $\alpha_i^M(t)$ to denote the number of balls accepted by bin i in round t by process CAPPED(c,λ) and MODCAPPED(c,λ), respectively, and define $\alpha^\bullet(t) := \sum_{i \in [n]} \alpha_i^\bullet(t)$ for $\bullet \in \{C, M\}$. Note that, since both processes accept as many balls as possible without exceeding the bin's capacity, $\alpha_i^\bullet(t) = \min\{\nu_i^\bullet(t), c - \ell_i^\bullet(t-1)\}$.

We are now ready for the induction step. So fix $t \in \mathbb{N}$ with $m^C(t-1) \leq m^M(t-1)$ as well as $\ell_i^C(t-1) \leq \ell_i^M(t-1)$ for each bin i. First, using the induction hypothesis $m^C(t-1) \leq m^M(t-1)$, we observe that

$$\begin{aligned} \nu^C(t) &= m^C(t-1) + \lambda n \\ &\leq m^M(t-1) + \max\{\lambda n, m^* - m^M(t-1)\} \\ &= \nu^M(t). \end{aligned} \tag{6}$$

This allows us to couple the processes' random choices in round t as follows: we number the balls thrown by CAPPED(c,λ) from 1 to $\nu^C(t)$ and those thrown by MODCAPPED(c, λ) from 1 to $\nu^M(t)$. The first $\nu^C(t)$ balls of MODCAPPED(c,λ) use the same random bin choices as their counterparts in CAPPED(c, λ). The remaining $\nu^M(t) - \nu^C(t) \geq 0$ balls of MODCAPPED(c,λ) choose independent, uniform bins.

Now we prove the first invariant of the induction step, namely $m^C(t) \leq m^M(t)$. To this end, note that $m^\bullet(t) = \nu^\bullet(t) - \alpha^\bullet(t)$ for $\bullet \in \{C, M\}$ (the number of thrown balls minus the number of accepted balls). To upper-bound $m^C(t)$, we lower-bound $\alpha^C(t) = \sum_{i \in [n]} \alpha_i^C(t)$:

$$\begin{aligned} \alpha_i^C(t) &= \min\{\nu_i^C(t) + \nu_i^M(t) - \nu_i^M(t), c - \ell_i^C(t-1)\} \\ &\geq \min\{\nu_i^M(t), c - \ell_i^M(t-1)\} - \big(\nu_i^M(t) - \nu_i^C(t)\big) \\ &= \alpha_i^M(t) - \big(\nu_i^M(t) - \nu_i^C(t)\big), \end{aligned} \tag{7}$$

where the inequality uses $\nu_i^M(t) - \nu_i^C(t) \geq 0$ (Equation (6)) as well as the induction hypothesis $\ell_i^C(t-1) \leq \ell_i^M(t-1)$. Summing over all bins $i \in [n]$ yields $\alpha^C(t) \geq \alpha^M(t) - \nu^M(t) + \nu^C(t)$. Thus, we get the first invariant via $m^C(t) = \nu^C(t) - \alpha^C(t) \leq \nu^C(t) - \alpha^M(t) + \nu^M(t) - \nu^C(t) = m^M(t)$.

To prove the second invariant of the induction step for a bin $i \in [n]$, namely $\ell_i^C(t) \leq \ell_i^M(t)$, note

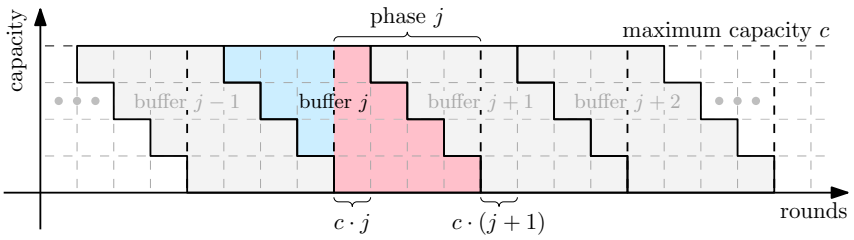

Figure 2. Buffers $j-1$ to $j+2$ for some bin and $c=4$.

that $\ell_i^C(t) = \max\{0, \ell_i^C(t-1)+\alpha_i^C(t)-1\}$ (old load plus accepted load minus a potential deletion) and $\ell_i^M(t) \geq \max\{0, \ell_i^M(t-1)+\alpha_i^M(t)-1\}$ (here we don't have equality, because the load after the allocation and before the deletion could be completely allocated to the bin's blue buffer, which performs no deletion). A calculation similar to Equation (7) yields $\alpha_i^C(t) \leq \alpha_i^M(t)+\ell_i^M(t-1)-\ell_i^C(t-1)$, which in turn allows us to calculate

$$\begin{aligned}\ell_i^C(t) &= \max\{0, \ell_i^C(t-1)+\alpha_i^C(t)-1\} \\ &\leq \max\{0, \ell_i^C(t-1)+\alpha_i^M(t)+\ell_i^M(t-1)-\ell_i^C(t-1)-1\} \\ &\leq \ell_i^M(t). \quad \square\end{aligned}$$

With Lemma 6, it is sufficient to prove Statement 1 of Theorem 2 for ModCapped(c,λ) instead of for Capped(c,λ). This is done in the following lemma.

Lemma 7. *Consider* ModCapped(c, λ) *for* $c \in \mathbb{N}$ *and* $0 \leq \lambda \leq 1-1/n$ *with* $\lambda n \in \mathbb{N}$. *Let* t *be an arbitrary round. With probability at least* $1-2^{-2n}$, *the size of the pool at the end of round* t *is less than* $4 \cdot c^{-1} \cdot \ln\big(1/(1-\lambda)\big) \cdot n + \mathrm{O}(c \cdot n)$.

Proof. For $t \in \mathbb{N}$ let $m^M(t)$ denote the pool size of ModCapped(c,λ) at the end of round t. Remember that $m^* = 2c^{-1} \cdot \ln\big(1/(1-\lambda)\big) \cdot n + 6c \cdot n$. Let $\tau = \tau(t) := \lfloor t/c \rfloor - 1$ (such that $t \in I_{\tau+1}$) and consider $\sigma \in \mathbb{N}$ with $\sigma \leq \tau+1$. We define the events

$$\mathcal{A}_t := \{m^M(t) \geq 2m^*\} \qquad \text{and}$$
$$\mathcal{B}_{\sigma,t} := \{\exists s \in I_{\sigma-1} \colon m^M(s) \leq m^* \wedge \forall t' \in (s;t] \colon m^M(t') > m^*\}.$$

Note that for $\mathcal{A}_t$ to occur, there must be exactly one $s < t$ as in the definition of $\mathcal{B}_{\sigma,t}$, and any round $t' \in (s;t]$ generates exactly λn balls at its beginning. Moreover, between s and t there must lie at least one complete phase (otherwise the pool size could grow by at most $2c \cdot \lambda n < m^*$ between s and t, not enough to increase from m^* to $2m^*$).

Our goal is to prove $\Pr[\mathcal{A}_t] \leq 2^{-2n}$. By the above observations, we can partition $\mathcal{A}_t$ via $\mathcal{A}_t = \biguplus_{\sigma \in [\tau(t)]} \mathcal{A}_t \cap \mathcal{B}_{\sigma,t}$. Thus, our strategy is to first bound each $\Pr[\mathcal{A}_t \cap \mathcal{B}_{\sigma,t}]$. So fix $t \in \mathbb{N}$ and $\sigma \in [\tau(t)]$ and assume event $\mathcal{A}_t \cap \mathcal{B}_{\sigma,t}$ occurs. Let $s \in I_{\sigma-1}$ be maximal with $m^M(s) \leq m^*$ (such an s exists because of $\mathcal{B}_{\sigma,t}$). Note that the pool size at the beginning of phase σ is at most $m^* + c \cdot \lambda n$ (each of the at most c rounds from s to the next phase generates λn balls by event $\mathcal{B}_{\sigma,t}$). Similarly, the pool size at the end of phase τ is at least $2m^* - c \cdot \lambda n$ (or the at most c rounds from the end to phase τ to round t could not generate enough balls to reach pool size $2m^*$). See Figure 3 for an illustration.

Define $\Delta_{\sigma,\tau} := \tau - \sigma + 1$ as the number of phases from σ to τ. During these phases, ModCapped(c,λ) generates exactly $\Delta_{\sigma,\tau} \cdot c \cdot \lambda n$ balls (since $\mathcal{B}_{\sigma,t}$ holds). For each of the n bins there are $\Delta_{\sigma,\tau}$ buffers that each delete up to c balls during the phases σ to τ. Let $X_{\sigma,\tau}$ denote the number of these $\Delta_{\sigma,\tau} \cdot n$ buffers that delete less than c balls. Then, $\Delta_{\sigma,\tau} \cdot n - X_{\sigma,\tau}$ is the number of those buffers that each delete exactly c balls, and we can upper-bound the increase of the pool during the phases σ to τ by $\Delta_{\sigma,\tau} \cdot c \cdot \lambda n - c \cdot (\Delta_{\sigma,\tau} \cdot n - X_{\sigma,\tau})$. Consequently, for the pool size to increase from at most $m^* + c \cdot \lambda n$ to at least $2m^* - c \cdot \lambda n$ during the phases σ to τ, we must have $\Delta_{\sigma,\tau} \cdot c \cdot \lambda n - c \cdot (\Delta_{\sigma,\tau} \cdot n - X_{\sigma,\tau}) \geq m^* - 2c \cdot \lambda n$ or, equivalently, $X_{\sigma,\tau} \geq m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n - 2 \cdot \lambda n$.

Thus, we have shown $\mathcal{A}_t \cap \mathcal{B}_{\sigma,t} \implies X_{\sigma,\tau} \geq m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n - 2 \cdot \lambda n$, where $\tau = \tau(t) = \lfloor t/c \rfloor - 1$. This gives

$$\Pr[\mathcal{A}_t \cap \mathcal{B}_{\sigma,t}] \leq \Pr[X_{\sigma,\tau} \geq m^*/c + (1-\lambda)\Delta_{\sigma,\tau} n - 2\lambda n]. \quad (8)$$

We now fix arbitrary phases $\sigma, \tau \in \mathbb{N}$ with $\sigma \leq \tau$ and bound $\Pr[X_{\sigma,\tau} \geq m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n - 2 \cdot \lambda n]$. Observe that the random variable $X_{\sigma,\tau}$ can be written as a sum $X_{\sigma,\tau} = \sum_{j \in [\sigma;\tau]} \sum_{i \in [n]} X_j(i)$ of indicator variables $X_j(i)$, where $X_j(i) = 1$ if buffer j of bin i deletes less than c balls and $X_{t'}(i) = 0$ otherwise.

Fix buffer $j \in \mathbb{N}_0$ of some bin $i \in [n]$. By construction, buffer j is active during the rounds $A_j := I_{j-1} \setminus \{(j-1) \cdot c\} \cup I_j$. Remember that buffer j deletes balls only during I_j and note that if j accepts a ball during a round $t \in A_j$, then that ball will be deleted by j during I_j (at any time $t \in I_j$, the maximum capacity equals the remaining steps in I_j, during each of which j can delete one ball). Thus, for j to not delete c balls ($X_j(i) = 1$), there must be at least c rounds in A_j during which j receives no balls (otherwise it accepts at least one ball in at least $|A_j| - (c-1) = c$ rounds, and each of these c balls is deleted during I_j). By definition of ModCapped(c,λ), in each round $t' \in A_j$ exactly $\nu^M(t') \geq m^*$ balls are thrown randomly into the n bins, and half of those balls prefer buffer j. Thus, the probability for j to not receive a ball in round t' is at most $p := (1-1/n)^{m^*/2} \leq e^{-m^*/(2n)}$. So the probability that there are at least c out of the $|A_j| = 2c-1$ rounds during which j receives no ball is at most $\binom{2c-1}{c} \cdot p^c \leq 2^{2c-1} \cdot p^c \leq 2^{2c-1} \cdot e^{-c \cdot m^*/(2n)}$, where we used $\binom{n}{k} \leq 2^n$.

Note that the reasoning above depends only on the random choices of the balls that prefer buffer j, of which we have at least $m^*/2$ each round during which j is active.

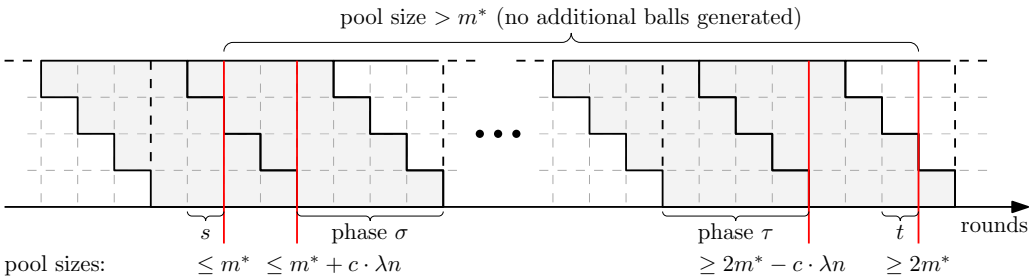

Figure 3. Situation in the proof of Lemma 7 if event $\mathcal{A}_t \cap \mathcal{B}_{\sigma,t}$ holds: the pool size after round t is at least $2m^*$ and $s \in I_{\sigma-1}$ is the last round before t after which the pool size was at most m^*.

Thus, we have shown that for a fixed j, $\Pr[X_j(i)=1] \leq 2^{2c-1} \cdot e^{-c \cdot m^*/(2n)}$.

Note that, by our choice of $m^* = 2c^{-1} \cdot \ln\big(1/(1-\lambda)\big) \cdot n + 6c \cdot n$, we can bound $\mathbb{E}[X_{\sigma,\tau}] \leq \Delta_{\sigma,\tau} \cdot n \cdot 2^{2c-1} \cdot e^{-c \cdot m^*/(2n)} = (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n/(2e) \cdot (4/e^{3-1/c})^c \leq (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n/(2e)$. Together with $m^*/c \geq 2 \cdot \lambda n$, this gives $R := m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n - 2 \cdot \lambda n \geq 2e \cdot \mathbb{E}[X_{\sigma,\tau}]$. Similar to Lemma 2, we note the negative association of $(X_j(i))_{j \in [\sigma,\tau], i \in [n]}$ and bound $X_j(i)$ by only considering the first $m^*/2$ balls. We can then apply the Chernoff bound from Lemma 8 to get

$$\begin{aligned} &\Pr[X_{\sigma,\tau} \geq m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n - 2 \cdot \lambda n] \\ &\quad = \Pr[X_{\sigma,\tau} \geq R] \leq 2^{-R} \\ &\quad = 2^{-m^*/c + 2 \cdot \lambda n} \cdot 2^{-(1-\lambda) \cdot \Delta_{\sigma,\tau} \cdot n}. \end{aligned} \tag{9}$$

We are now ready to prove $\Pr[\mathcal{A}_t] \leq 2^{-2n}$. Using the partitioning $\mathcal{A}_t = \biguplus_{\sigma \in [\tau(t)]} \mathcal{A}_t \cap \mathcal{B}_{\sigma,t}$ and combining Equations (8) and (9) yields

$$\begin{aligned} \Pr[\mathcal{A}_t] &= \sum_{\sigma \in [\tau(t)]} \Pr[\mathcal{A}_t \cap \mathcal{B}_{\sigma,t}] \\ &\leq \sum_{\sigma \in [\tau(t)]} \Pr\big[X_{\sigma,\tau(t)} \geq m^*/c + (1-\lambda) \cdot \Delta_{\sigma,\tau(t)} \cdot n - 2\lambda n\big] \\ &\leq \sum_{\sigma \in [\tau(t)]} 2^{-m^*/c + 2 \cdot \lambda n} \cdot 2^{-(1-\lambda) \cdot \Delta_{\sigma,\tau(t)} \cdot n} \\ &\leq 2^{-m^* + 2 \cdot \lambda n} \cdot \sum_{i \in [\tau(t)]} \big(2^{-1}\big)^i \leq 2^{-m^* + 2 \cdot \lambda n}, \end{aligned} \tag{10}$$

where the penultimate inequality uses $\lambda \leq 1 - 1/n$ and reverses the summation, and the last inequality bounds the sum via a geometric series. From this, the desired result follows by noting that $m^* = 2c^{-1} \cdot \ln\big(1/(1-\lambda)\big) \cdot n + 6c \cdot n - 2 \cdot \lambda n \geq 2n$. □

C. Analysis of the Waiting Time

The proof of Statement 2 of Theorem 2 directly follows from the following observation and the fact that any allocated balls wait for at most c rounds to be deleted.

Observation 1. *Lemmas 3 to 5 hold for $c \geq 1$.*

Proof. In Lemmas 3 to 5 we used that no ball in $M(t)$ can be delayed by younger balls. While this only holds for $c=1$, every bin has at least one empty slot at the beginning of a round. A bin will still never assign a ball created later than t while rejecting a ball of $M(t)$. Therefore, Lemmas 3 to 5 overestimate the number of rounds to allocate all balls of $M(t)$ for $c > 1$. □

Then we use Statement 1 of Theorem 2 to bound $m(t)$ for Lemma 3 to get the desired results in the same way as we did in the corresponding proof of Theorem 1.

V. Empirical Analysis

In this section we present our empirical data for Capped(c,λ). The goal of this section is two-fold. Firstly, we compare the theoretical results (for the pool size and the waiting time) with our simulations to gauge how much we lose by explicitly not optimizing constants in the analysis, and to show that the process works very well in practice. Secondly, our theoretical results indicate the existence of a sweet spot for the choice of c, and we wish to determine its value through the experiments.

Our simulator is implemented in C++ (https://github.com/t-c-hahn/balls-and-bins-simulation). The simulations are performed for an initially empty system of $n = 2^{15}$ bins. Extensive simulations have shown that the actual number of n has negligible impact on the (normalized) simulation results. Hence we only present the data for $n = 2^{15}$. All measurements refer to a stabilized system after a burn-in phase of suitable length. For each data point we average over 1000 rounds. In our figures we use the normalized pool size (actual value divided by n).

Pool Size: In the first set of experiments we consider varying values of λ and c. The left plot in Figure 4 shows the normalized pool size as a function of capacity $c \in [1,5]$. The right plot in Figure 4 shows it as a function of $\lambda = 1 - 1/2^i$ for $i \in [1,10]$. The data indicate that the number of jobs awaiting allocation is bounded by $n/c \cdot \ln(1/(1-\lambda)) + n$, which is essentially the bound of Theorem 2 omitting the factor of four. This suggests that our theoretically derived bound of $4n/c \cdot \ln(1/(1-\lambda)) + \mathrm{O}\big(c \cdot n\big)$ is rather pessimistic.

Waiting Times: In Figure 5, we consider the balls' average and maximum waiting times. The left plot shows the waiting time as a function of the capacity $c \in [1,5]$. The right plot shows it as a function of $\lambda = 1 - 2^{-i}$ for $i \in [1,10]$. Similarly to the pool size, our data indicate that the waiting time is bounded by $\ln(1/(1-\lambda)) + \log\log n + c$, which is again the bound of Theorem 2 omitting the constant factors. Furthermore, the data in Figure 5 show a minimum for both the average and the maximum waiting times around $c = 2$ and $c = 3$ for the specified values of λ.

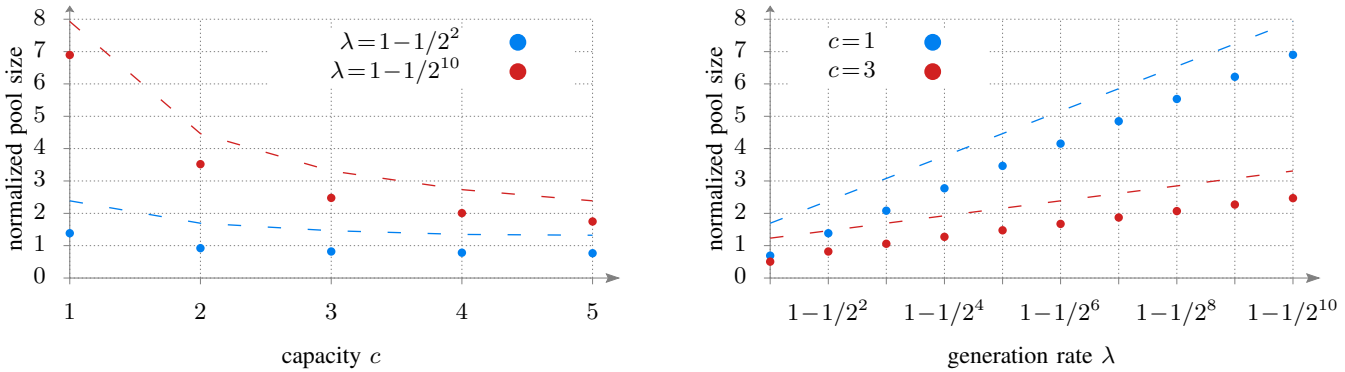

Figure 4. The left plot shows the normalized pool size for two fixed injection rates, $\lambda=1-1/2^2$ and $\lambda=1-1/2^{10}$. The right plot shows the normalized pool size for two fixed capacities, $c=1$ and $c=3$. The dashed lines correspond to the function $1/c\cdot\ln(1/(1-\lambda))+1$.

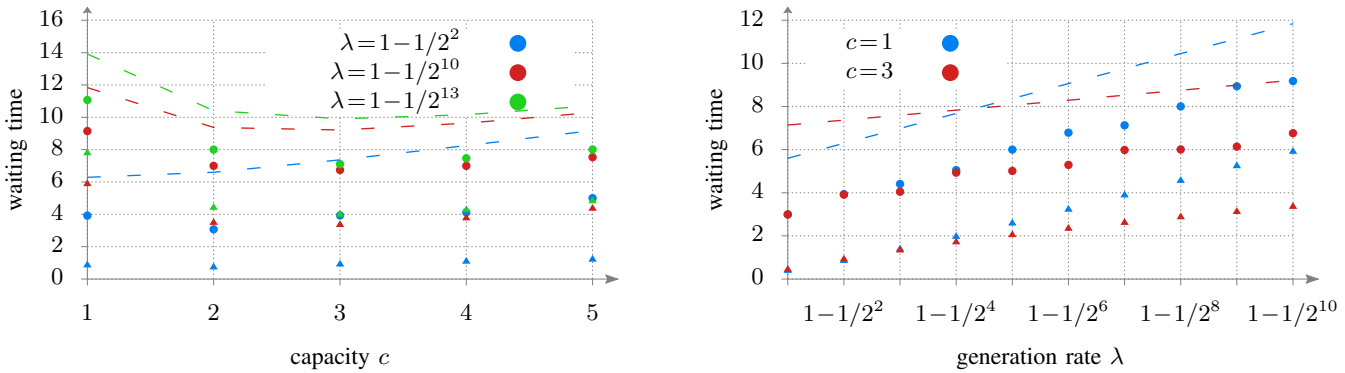

Figure 5. The left plot shows the average waiting time (in triangles) and the maximum waiting time (in points) for $\lambda=1-1/2^2$, $\lambda=1-1/2^{10}$, and $\lambda=1-1/2^{13}$. The right plot shows the average waiting time (in triangles) and the maximum waiting time (in points) for $c=1$ and $c=3$. The dashed lines correspond to the function $\ln(1/(1-\lambda))/c+\text{loglog}n+c$.

References

[1] M. Adler, P. Berenbrink, and K. Schröder. Analyzing an Infinite Parallel Job Allocation Process. In: *Proc. ESA*. 1998, pp. 417–428.

[2] M. Adler, S. Chakrabarti, M. Mitzenmacher, and L. Rasmussen. Parallel randomized load balancing. In: *Random Structures & Algorithms* 13.2 (1998), pp. 159–188.

[3] J. Aspnes. *Notes on Randomized Algorithms*. http://www.cs.yale.edu/homes/aspnes/classes/469/notes.pdf. 2019.

[4] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced Allocations. In: *SIAM Journal on Computing* 29.1 (1999), pp. 180–200.

[5] L. Becchetti, A. E. F. Clementi, E. Natale, F. Pasquale, and G. Posta. Self-Stabilizing Repeated Balls-into-Bins. In: *Proc. SPAA*. 2015, pp. 332–339.

[6] P. Berenbrink, A. Brinkmann, T. Friedetzky, and L. Nagel. Balls into Non-uniform Bins. In: *Journal of Parallel and Distributed Computing* 74.2 (2014), pp. 2065–2076.

[7] P. Berenbrink, A. Czumaj, M. Englert, T. Friedetzky, and L. Nagel. Multiple-Choice Balanced Allocation in (Almost) Parallel. In: *APPROX-RANDOM 2012*. 2012, pp. 411–422.

[8] P. Berenbrink, A. Czumaj, T. Friedetzky, and N. D. Vvedenskaya. Infinite Parallel Job Allocation (Extended Abstract). In: *Proc. SPAA*. 2000, pp. 99–108.

[9] P. Berenbrink, A. Czumaj, A. Steger, and B. Vöcking. Balanced Allocations: The Heavily Loaded Case. In: *SIAM Journal on Computing* 35.6 (2006), pp. 1350–1385.

[10] P. Berenbrink, T. Friedetzky, P. Kling, F. Mallmann-Trenn, L. Nagel, and C. Wastell. Self-Stabilizing Balls and Bins in Batches - The Power of Leaky Bins. In: *Algorithmica* 80.12 (2018), pp. 3673–3703.

[11] 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: *Proc. PODC*. 2016, pp. 83–92.

[12] R. Cole, A. M. Frieze, B. M. Maggs, M. Mitzenmacher, A. W. Richa, R. K. Sitaraman, and E. Upfal. On Balls and Bins with Deletions. In: *Proc. RANDOM*. 1998, pp. 145–158.

[13] D. P. Dubhashi and D. Ranjan. Balls and bins: A study in negative dependence. In: *BRICS Report Series* 3.25 (1996).

[14] M. Goemans. *Chernoff Bounds, and Some Applications*. http://math.mit.edu/~goemans/18310S15/chernoff-notes.pdf. 2015.

[15] C. Lenzen, M. Parter, and E. Yogev. Parallel Balanced Allocations: The Heavily Loaded Case. In: *Proc. SPAA*. 2019, pp. 313–322.

[16] M. Mitzenmacher. The Power of Two Choices in Randomized Load Balancing. In: *IEEE Trans. Parallel Distrib. Syst.* 12.10 (2001), pp. 1094–1104.

[17] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. 2017.

[18] R. Motwani and P. Raghavan. *Randomized Algorithms*. 1995.

[19] M. Raab and A. Steger. "Balls into bins"—A simple and tight analysis. In: *RANDOM'98*. 1998, pp. 159–170.

[20] V. Stemann. Parallel Balanced Allocations. In: *Proc. SPAA*. 1996, pp. 261–269.

[21] B. Vöcking. How asymmetry helps load balancing. In: *Journal of the ACM* 50.4 (2003), pp. 568–589.

APPENDIX

A. Tail Bounds

We use the following two variants of the Chernoff inequality. The former is stated and proved in [3] (and is based on Theorem 4.4 in [17]), and the latter may be found in [14].

Lemma 8. *Let $X_1,\ldots,X_n$ be independent Bernoulli trials such that $\Pr[X_i=1]=p_i$. Let $X=\sum_{i=1}^{n}X_i$. Then*

$$\Pr[X\geq R]\leq 2^{-R} \quad \text{if } R\geq 2e\mathbb{E}[X].$$

Lemma 9. *Let $X=\sum_{i=1}^{n}X_i$, where $X_i=1$ with probability p_i and $X_i=0$ with probability $1-p_i$, and all X_i are independent. Let $\mu=\mathbb{E}[X]=\sum_{i=1}^{n}p_i$. Then*

$$\Pr[X\geq(1+\delta)\mu]\leq e^{-\frac{\delta^2\mu}{2+\delta}} \quad \text{for all } \delta>0.$$

The next lemma gives tail bounds for the number of empty bins when throwing m balls into n bins.

Lemma 10 ([18, Theorem 4.18])**.** *Assume we allocate m balls into n bins. Let Z be the number of empty bins after the allocation. Then*

$$\Pr[|Z-\mathbb{E}[Z]|\geq\lambda]\leq 2\exp\left(-\frac{\lambda^2\cdot(n-1/2)}{n^2-(\mathbb{E}[Z])^2}\right).$$

The following useful inequality can be found in [4] (as Lemma 3.1) and relates to the Binomial distribution $B(n,p)$.

Lemma 11. *Let $X_1,X_2,\ldots,X_n$ be a sequence of random variables with values in an arbitrary domain. Let $Y_1,Y_2,\ldots,Y_n$ be a sequence of binary random variables with the property that $Y_i=Y_i(X_1,\ldots,X_i)$. If*

$$\Pr[Y_i=1 \mid X_1,\ldots,X_{i-1}]\leq p$$

then

$$\Pr\left[\sum_{i=1}^{n}Y_i\geq k\right]\leq\Pr[B(n,p)\geq k].$$