The Annals of Applied Probability
2010, Vol. 20, No. 4, 1470–1511
DOI: 10.1214/09-AAP656

CHOICE-MEMORY TRADEOFF IN ALLOCATIONS

BY NOGA ALON[1], ORI GUREL-GUREVICH AND EYAL LUBETZKY

Tel Aviv University, Microsoft Research and Microsoft Research

In the classical balls-and-bins paradigm, where n balls are placed independently and uniformly in n bins, typically the number of bins with at least two balls in them is $\Theta(n)$ and the maximum number of balls in a bin is $\Theta(\frac{\log n}{\log\log n})$. It is well known that when each round offers k independent uniform options for bins, it is possible to typically achieve a constant maximal load if and only if $k = \Omega(\log n)$. Moreover, it is possible w.h.p. to avoid any collisions between $n/2$ balls if $k > \log_2 n$.

In this work, we extend this into the setting where only m bits of memory are available. We establish a tradeoff between the number of choices k and the memory m, dictated by the quantity km/n. Roughly put, we show that for $km \gg n$ one can achieve a constant maximal load, while for $km \ll n$ no substantial improvement can be gained over the case $k = 1$ (i.e., a random allocation).

For any $k = \Omega(\log n)$ and $m = \Omega(\log^2 n)$, one can achieve a constant load w.h.p. if $km = \Omega(n)$, yet the load is unbounded if $km = o(n)$. Similarly, if $km > Cn$ then $n/2$ balls can be allocated without any collisions w.h.p., whereas for $km < \varepsilon n$ there are typically $\Omega(n)$ collisions. Furthermore, we show that the load is w.h.p. at least $\frac{\log(n/m)}{\log k + \log\log(n/m)}$. In particular, for $k \le \text{polylog}(n)$, if $m = n^{1-\delta}$ the optimal maximal load is $\Theta(\frac{\log n}{\log\log n})$ (the same as in the case $k = 1$), while $m = 2n$ suffices to ensure a constant load. Finally, we analyze nonadaptive allocation algorithms and give tight upper and lower bounds for their performance.

1. Introduction. The balls-and-bins paradigm (see, e.g., [11, 17]) describes the process where b balls are placed independently and uniformly at random in n bins. Many variants of this classical occupancy problem were intensively studied, having a wide range of applications in computer science.

It is well known that when $b = \lambda n$ for λ fixed and $n \to \infty$, the load of each bin tends to Poisson with mean λ and the bins are asymptotically independent. In particular, for $b = n$, the typical number of empty bins at the end of the process is $(1/\mathrm{e} + o(1))n$. The typical maximal load in that case is $(1 + o(1))\frac{\log n}{\log\log n}$ (cf. [15]).

Received October 2009.

[1]Supported in part by a USA Israeli BSF grant, by a grant from the Israel Science Foundation, by an ERC Advanced Grant and by the Hermann Minkowski Minerva Center for Geometry at Tel Aviv University.

AMS 2000 subject classifications. 60C05, 60G50, 68Q25.

Key words and phrases. Space/performance tradeoffs, balls and bins paradigm, lower bounds on memory, balanced allocations, online perfect matching.

In what follows, we say that an event holds with high probability (w.h.p.) if its probability tends to 1 as $n \to \infty$.

The extensive study of this model in the context of load balancing was pioneered by the celebrated paper of Azar et al. [3] (see the survey [21]) that analyzed the effect of a choice between k independent uniform bins on the maximal load, in an online allocation of n balls to n bins. It was shown in [3] that the GREEDY algorithm (choose the least loaded bin of the k) is optimal and achieves a maximal-load of $\log_k \log n$ w.h.p., compared to a load of $\frac{\log n}{\log \log n}$ for the original case $k = 1$. Thus, $k = 2$ random choices already significantly reduce the maximal load, and as k further increases, the maximal load drops until it becomes constant at $k = \Omega(\log n)$.

In the context of online bipartite matchings, the process of dynamically matching each client in a group A of size $n/2$ with one of k independent uniform resources in a group B of size n precisely corresponds to the above generalization of the balls-and-bins paradigm: Each ball has k options for a bin, and is assigned to one of them by an online algorithm that should avoid collisions (no two balls can share a bin). It is well known that the threshold for achieving a perfect matching in this case is $k = \log_2 n$: For $k \geq (1+\varepsilon)\log_2 n$, w.h.p. every client can be exclusively matched to a target resource, and if $k \leq (1-\varepsilon)\log_2 n$ then $\Omega(n)$ requests cannot be satisfied.

In this work, we study the above models in the presence of a constraint on the memory that the online algorithm has at its disposal. We find that a tradeoff between the choice and the memory governs the ability to achieve a perfect allocation as well as a constant maximal load. Surprisingly, the threshold separating the subcritical regime from the supercritical regime takes a simple form, in terms of the product of the number of choices k, and the size of the memory in bits m:

- If $km \gg n$, then one can allocate $(1-\varepsilon)n$ balls in n bins without any collisions w.h.p., and consequently achieve a load of 2 for n balls.
- If $km \ll n$, then *any* algorithm for allocating εn balls w.h.p. creates $\Omega(n)$ collisions and an unbounded maximal load.

Roughly put, when $km \gg n$ the amount of choice and memory at hand suffices to guarantee an essentially best-possible performance. On the other hand, when $km \ll n$, the memory is too limited to enable the algorithm to make use of the extra choice it has, and no substantial improvement can be gained over the case $k = 1$, where no choice is offered whatsoever.

Note that rigorous lower bounds for space, and in particular tradeoffs between space and performance (time, communication, etc.), have been studied intensively in the literature of Algorithm Analysis, and are usually highly nontrivial. See, for example, [1, 4–6, 8, 9, 12, 13] for some notable examples.

Our first main result establishes the exact threshold of the choice-memory tradeoff for achieving a constant maximal-load. As mentioned above, one can verify that when there is unlimited memory, the maximal load is w.h.p. uniformly bounded iff

$k = \Omega(\log n)$. Thus, assuming that $k = \Omega(\log n)$ is a prerequisite for discussing the effect of limited memory on this threshold.

THEOREM 1. *Consider n balls and n bins, where each ball has $k = \Omega(\log n)$ uniform choices for bins, and $m = \Omega(\log^2 n)$ bits of memory are available. If $km = \Omega(n)$, one can achieve a maximal-load of $O(1)$ w.h.p. Conversely, if $km = o(n)$, any algorithm w.h.p. creates a load that exceeds any constant.*

Consider the case $k = \Theta(\log n)$. The naïve algorithm for achieving a constant maximal-load in this setting requires roughly n bits of memory ($2n$ bits of memory always suffice; see Section 1.3). Surprisingly, the above theorem implies that $O(n/\log n)$ bits of memory already suffice, and this is tight.

As we later show, one can extend the upper bound on the load, given in Theorem 1, to $O(\frac{n}{km})$ (useful when $\frac{n}{km} \le \frac{\log n}{\log\log n}$), whereas the lower bound tends to ∞ with $\frac{n}{km}$. This further demonstrates how the quantity $\frac{n}{km}$ governs the value of the optimal maximal load. Indeed, Theorem 1 will follow from Theorems 3 and 4 below, which determine that the threshold for a perfect matching is $km = \Theta(n)$.

Again consider the case of $k = \Theta(\log n)$, where an online algorithm with unlimited memory can achieve an $O(1)$ load w.h.p. While the above theorem settles the memory threshold for achieving a constant load in this case, one can ask what the optimal maximal load would be below the threshold. This is answered by the next theorem, which shows that in this case, for example, $m = n^{1-\delta}$ bits of memory yield no significant improvement over an algorithm which makes random allocations.

THEOREM 2. *Consider n/k balls and n bins, where each ball has k uniform choices for bins, and $m \ge \log n$ bits of memory are available. Then for any algorithm, the maximal load is at least $(1 + o(1))\frac{\log(n/m)}{\log\log(n/m)+\log k}$ w.h.p.*

In particular, if $m = n^{1-\delta}$ for some $\delta > 0$ fixed and $2 \le k \le \mathrm{polylog}(n)$, then the maximal load is $\Theta(\frac{\log n}{\log\log n})$ w.h.p.

Recall that a load of order $\frac{\log n}{\log\log n}$ is what one would obtain using a random allocation of n balls in n bins. The above theorem states that, when $m = n^{1-\delta}$ and $k \le \mathrm{polylog}(n)$, any algorithm would create such a load already after n/k rounds.

Before describing our other results, we note that the lower bounds in our theorems in fact apply to a more general setting. In the original model, in each round the online algorithm chooses one of k uniformly chosen bins, thus inducing a distribution on the location of the next ball. Clearly, this distribution has the property that no bin has a probability larger than k/n.

Our theorems apply to a relaxation of the model, where the algorithm is allowed to dynamically choose a distribution Q_t for each round t, which is required to

satisfy the above property (i.e., $\|Q_t\|_\infty \le k/n$). We refer to these distributions as *strategies*.

Observe that indeed this model gives more power to the online algorithm. For instance, if $k = 2$ (and the memory is unlimited), an algorithm in the relaxed model can allocate $n/2$ balls perfectly (by assigning 0 probability to the occupied bins), whereas in the original model collisions occur already with $n^{2/3}w(n)$ balls w.h.p., for any $w(n)$ tending to ∞ with n.

Furthermore, we also relax the memory constraint on the model. Instead of treating the algorithm as an automaton with 2^m states, we only impose the restriction that there are at most 2^m different strategies to choose from. In other words, at time t, the algorithm knows the entire history (the exact location of each ball so far), and needs to choose one of its 2^m strategies for the next round. In this sense, our lower bounds are for the case of limited communication complexity rather than limited space complexity.

We note that all our bounds remain valid when each round offers k choices with repetitions.

1.1. *Tradeoff for perfect matching.* The next two theorems address the threshold for achieving a perfect matching when allocating $(1-\delta)n$ balls in n bins for some fixed $0 < \delta < 1$ [note that for $\delta = 0$, even with unlimited memory, one needs $k = \Omega(n)$ choices to avoid collisions w.h.p.]. The upper and lower bounds obtained for this threshold are tight up to a multiplicative constant, and again pinpoint its location at $km = \Theta(n)$. The constants below were chosen to simplify the proofs and could be optimized.

THEOREM 3. *For $\delta > 0$ fixed, consider $(1-\delta)n$ balls and n bins: Each ball has k uniform choices for bins, and there are $m \ge \log n$ bits of memory. If*

$$km \le \varepsilon n \qquad \text{for some small constant } \varepsilon > 0,$$

then any algorithm has $\Omega(n)$ collisions w.h.p.

Furthermore, the maximal load is w.h.p. $\Omega(\log\log(\frac{n}{km}))$.

THEOREM 4. *For $\delta > 0$ fixed, consider $(1-\delta)n$ balls and n bins, where each ball has k uniform choices for bins, and m bits of memory are available. The following holds for any $k \ge (3/\delta)\log n$ and $m \ge \log n \cdot \log_2 \log n$. If*

$$km \ge Cn \qquad \text{for some } C = C(\delta) > 0,$$

then a perfect allocation (no collisions) can be achieved w.h.p.

In light of the above, for any value of k, the online allocation algorithm given by Theorem 4 is optimal with respect to its memory requirements.

1.2. *Nonadaptive algorithms*. In the nonadaptive case, the algorithm is again allowed to choose a fixed (possibly randomized) strategy for selecting the placement of ball number t in one of the k possible randomly chosen bins given in step t. Therefore, each such algorithm consists of a sequence $Q_1, Q_2, \ldots, Q_n$ of n predetermined strategies, where Q_t is the strategy for selecting the bin in step number t.

Here, we show that even if $k = n\frac{\log\log n}{\log n}$, the maximum load is w.h.p. at least $(1-o(1))\frac{\log n}{\log\log n}$, that is, it is essentially as large as in the case $k = 1$. It is also possible to obtain tight bounds for larger values of k. We illustrate this by considering the case $k = \Theta(n)$.

THEOREM 5. *Consider the problem of allocating n balls into n bins, where each ball has k uniform choices for bins, using a nonadaptive algorithm.*

(i) *The maximum load in any nonadaptive algorithm with $k \le n\frac{\log\log n}{\log n}$ is w.h.p. at least $(1-o(1))\frac{\log n}{\log\log n}$.*

(ii) *Fix $0 < \alpha < 1$. The maximum load in any nonadaptive algorithm with $k = \alpha n$ is w.h.p. $\Omega(\sqrt{\log n})$. This is tight, that is, there exists a nonadaptive algorithm with $k = \alpha n$ so that the maximum load in it is $O(\sqrt{\log n})$ w.h.p.*

1.3. *Range of parameters*. In the above theorems and throughout the paper, the parameter k may assume values up to n. As for the memory, one may naïvely use $n\log_2 L$ bits to store the status of n bins, each containing at most L balls. The next observation shows that the $\log_2 L$ factor is redundant.

OBSERVATION. At most $n + b - 1$ bits of memory suffice to keep track of the number of balls in each bin when allocating b balls in n bins.

Indeed, one can maintain the number of balls in each bin using a vector in $\{0,1\}^{n+b-1}$, where 1-bits stand for separators between the bins. In light of this, the original case of unlimited memory corresponds to the case $m = 2n$.

1.4. *Main techniques*. The key argument in the lower bound on the performance of the algorithm with limited memory is analyzing the expected number of new collisions that a given step introduces. We wish to estimate this value with an error probability smaller than 2^{-m}, so it would hold w.h.p. for all of the 2^m possible strategies for this step.

To this end, we apply a large deviation inequality, which relates the sum of a sequence of dependent random variables (X_i) with the sum of their "predictions" (Y_i), where Y_i is the expectation of X_i given the history up to time i. Proposition 2.1 essentially shows that if the sum of the predictions Y_i is large (exceeds some ℓ), then so is the sum of the actual random variables X_i, except with probability $\exp(-c\ell)$. In the application, the variable X_i measures the number of new

collisions introduced by the ith ball, and Y_i is determined by the strategy Q_i and the history so far.

The key ingredient in proving this proposition is a Bernstein–Kolmogorov type inequality for martingales, which appears in a paper of Freedman [14] from 1975, and bounds the probability of deviation of a martingale in terms of its cumulative variance. We reproduce its elegant proof for completeness. Crucially, that theorem does not require a uniform bound on individual variances (such as the one that appears in standard versions of Azuma–Hoeffding), and rather treats them as random variables. Consequently, the quality of our estimate in Proposition 2.1 is unaffected by the number of random variables involved.

For the upper bounds, the algorithm essentially partitions the bins into blocks, where for different blocks it maintains an accounting of the occupied bins with varying resolution. Once a block exceeds a certain threshold of occupied bins, it is discarded and a new block takes its place.

1.5. *Related work.* The problem of balanced allocations with limited memory is due to Itai Benjamini. In a recent independent work, Benjamini and Makarychev [7] studied the special case of the problem for $k = 2$ (i.e., when there are two choices for bins at each round). While our focus was mainly the regime $k = \Omega(\log n)$ (where one can readily achieve a constant maximal load when there is unlimited memory), our results also apply for smaller values of k. Namely, as a by-product, we improve the lower bound of [7] by a factor of 2, as well as extend it from $k = 2$ to any $k \leq \text{polylog}(n)$.

A different notion of memory was introduced to load balancing balls into bins in [20], where one has the option of placing the current ball in the least loaded bin offered in the previous round. In that setting, one could indeed improve the asymptotics (yet not the order) of the maximal load. Note that in our case we consider the original balls and bins model (as studied in [3]) and just impose restrictions on the space complexity of the algorithm.

See, for example, [22], Chapter 5, for more on the vast literature of load balancing balls into bins and its applications in computer science.

A modern application for the classical online perfect matching problem has advertisers (or *bidders*) play the role of the bins and internet search queries (or *keywords*) play the role of the balls. Upon receiving a search query, the search engine generates the list of related advertisements (revealing the choices for this ball) and must decide which of them to present in response to the query (where to allocate the ball). Note that in the classical papers that analyze online perfect matching one assumes a worst-case graph rather than a random bipartite graph, and the requests are randomly permuted; see [18] for a fundamental paper in this area.

1.6. *Organization.* This paper is organized as follows. In Section 2, we prove the large deviation inequality (Proposition 2.1). Section 3 contains the lower

bounds on the collisions and load, thus proving Theorem 3. Section 4 provides algorithms for achieving a perfect-matching and for achieving a constant load, respectively proving Theorem 4 and completing the proof of Theorem 1. In Section 5, we extend the analysis of the lower bound to prove Theorem 2. Section 6 discusses nonadaptive allocations, and contains the proof of Theorem 5. Finally, Section 7 is devoted to concluding remarks.

2. A large deviation inequality. This section contains a large deviation result, which will later be one of the key ingredients in proving our lower bounds for the load. Our proof will rely on a Bernstein–Kolmogorov-type inequality of Freedman [14], which extends the standard Azuma–Hoeffding martingale concentration inequality. Given a sequence of bounded (possibly dependent) random variables (X_i) adapted to some filter $(\mathcal{F}_i)$, one can consider the sequence (Y_i) where $Y_i = \mathbb{E}[X_i \mid \mathcal{F}_{i-1}]$, which can be viewed as predictions for the (X_i)'s. The following proposition essentially says that, if the sum of the predictions Y_i is large, so is the sum of the actual variables X_i.

PROPOSITION 2.1. *Let (X_i) be a sequence of random variables adapted to the filter $(\mathcal{F}_i)$ so that $0 \le X_i \le M$ for all i, and let $Y_i = \mathbb{E}[X_i \mid \mathcal{F}_{i-1}]$. Then*

$$\mathbb{P}\left(\left\{\left|\frac{\sum_{i\le t} X_i}{\sum_{i\le t} Y_i} - 1\right| \ge \frac{1}{2} \text{ and } \sum_{i\le t} Y_i \ge h\right\} \text{ for some } t\right) \le \exp\left(-\frac{h}{20M} + 2\right).$$

PROOF. As mentioned above, the proof hinges on a tail-inequality for sums of random variables, which appears in the work of Freedman [14] from 1975 (see also [23]), and extends such inequalities of Bernstein and Kolmogorov to the setting of martingales. See [14] and the references therein for more background on these inequalities, as well as [10] for similar martingale estimates. We include the short proof of Theorem 2.2 for completeness.

THEOREM 2.2 ([14], Theorem 1.6). *Let $(S_0, S_1, \ldots)$ be a martingale with respect to the filter $(\mathcal{F}_i)$. Suppose that $S_{i+1} - S_i \le M$ for all i, and write $V_t = \sum_{i=1}^{t} \mathrm{Var}(S_i \mid \mathcal{F}_{i-1})$. Then for any $s, v > 0$ we have*

$$\mathbb{P}(S_n \ge S_0 + s, V_n \le v \text{ for some } n) \le \exp\left[-\frac{s^2}{2(v + Ms)}\right].$$

PROOF. Without loss of generality, suppose $S_0 = 0$, and put $X_i \stackrel{\triangle}{=} S_i - S_{i-1}$. Re-scaling S_n by M, it clearly suffices to treat the case $X_i \le 1$. Set

$$V_t \stackrel{\triangle}{=} \sum_{i=1}^{t} \mathrm{Var}(S_i \mid \mathcal{F}_{i-1}) = \sum_{i=1}^{t} \mathbb{E}(X_i^2 \mid \mathcal{F}_{i-1}),$$

and for some $\lambda > 0$ to be specified later, define

$$Z_t \stackrel{\triangle}{=} \exp\bigl(\lambda S_t - (e^{\lambda} - 1 - \lambda) V_t\bigr).$$

The next calculation will show that (Z_t) is a super-martingale with respect to the filter $(\mathcal{F}_t)$. First, notice that the function

$$f(z) \stackrel{\triangle}{=} \frac{e^z - 1 - z}{z^2} \qquad \text{for } z \neq 0,\ f(0) \stackrel{\triangle}{=} \frac{1}{2},$$

is monotone increasing [as $f'(z) > 0$ for all $z \neq 0$], and in particular, $f(\lambda z) \leq f(\lambda)$ for all $z \leq 1$. Rearranging,

$$\exp(\lambda z) \leq 1 + \lambda z + (e^{\lambda} - 1 - \lambda) z^2 \qquad \text{for all } z \leq 1.$$

Now, since $X_i \leq 1$ and $\mathbb{E}[X_i \mid \mathcal{F}_{i-1}] = 0$ for all i, it follows that

$$\begin{aligned}\mathbb{E}[\exp(\lambda X_i) \mid \mathcal{F}_{i-1}] &\leq 1 + (e^{\lambda} - 1 - \lambda)\mathbb{E}[X_i^2 \mid \mathcal{F}_{i-1}] \\ &\leq \exp\bigl((e^{\lambda} - 1 - \lambda)\mathbb{E}[X_i^2 \mid \mathcal{F}_{i-1}]\bigr).\end{aligned}$$

By definition, this precisely says that $\mathbb{E}[Z_i \mid \mathcal{F}_{i-1}] \leq Z_{i-1}$. That is, (Z_t) is a super-martingale, and hence by the Optional Stopping Theorem so is $(Z_{\tau \wedge n})$, where n is some integer and $\tau = \min\{t : S_t \geq s\}$. In particular,

$$\mathbb{E} Z_{\tau \wedge n} \leq Z_0 = 1,$$

and (noticing that $V_{t+1} \geq V_t$ for all t) Markov's inequality next implies that

$$\mathbb{P}\Biggl(\bigcup_{t \leq n} (S_t \geq s,\ V_t \leq v)\Biggr) \leq \exp[-\lambda s + (e^{\lambda} - 1 - \lambda) v].$$

A choice of $\lambda = \log(\frac{s+v}{v}) \geq \frac{s}{s+v} + \frac{1}{2}(\frac{s}{s+v})^2$ therefore yields

$$\begin{aligned}\mathbb{P}\Biggl(\bigcup_{t \leq n} (S_t \geq s,\ V_t \leq v)\Biggr) &\leq \exp\left[s - (s+v)\log\left(\frac{s+v}{v}\right)\right] \\ &\leq \exp\left[-\frac{s^2}{2(s+v)}\right],\end{aligned}$$

and taking a limit over n concludes the proof. □

REMARK. Note that Theorem 2.2 generalizes the well-known version of the Azuma–Hoeffding inequality, where each of the terms $\mathrm{Var}(X_i \mid \mathcal{F}_{i-1})$ is bounded by some *constant* σ_i^2 (cf., e.g., [19]).

We now wish to infer Proposition 2.1 from Theorem 2.2. To this end, define

$$Z_t \stackrel{\triangle}{=} \sum_{i=1}^{t} Y_i - X_i, \qquad V_t \stackrel{\triangle}{=} \sum_{i=1}^{t} \mathrm{Var}(Z_i \mid \mathcal{F}_{i-1}),$$

and observe that (Z_t) is a martingale by the definition $Y_i = \mathbb{E}[X_i \mid \mathcal{F}_{i-1}]$. Moreover, as the X_i's are uniformly bounded, so are the increments of Z_t:

$$|Z_i - Z_{i-1}| = |Y_i - X_i| \le M.$$

Furthermore, crucially, the variances of the increments are bounded as well in terms of the conditional expectations:

$$\mathrm{Var}(Y_i - X_i \mid \mathcal{F}_{i-1}) = \mathrm{Var}(X_i \mid \mathcal{F}_{i-1}) \le M \cdot \mathbb{E}[X_i \mid \mathcal{F}_{i-1}] = M \cdot Y_i,$$

giving that $V_t \le M \sum_{i=1}^{t} Y_i$.

Finally, for any integer $j \ge 1$ let A_j denote the event

$$A_j = \left(\left\{ \sum_{i \le t} X_i \le \frac{1}{2} \sum_{i \le t} Y_i \text{ and } jh \le \sum_{i \le t} Y_i \le (j+1)h \right\} \text{ for some } t \right).$$

Note that the event A_j implies that $Z_t \ge jh/2$. Hence, applying Theorem 2.2 to the martingale (Z_t) along with its cumulative variances (V_t) we now get

$$\begin{aligned} \mathbb{P}(A_j) &\le \mathbb{P}\left(Z_t \ge \frac{j}{2}h, V_t \le (j+1)hM \right) \\ &\le \exp\left[-\frac{((j/2)h)^2}{2((j+1)hM + M((j/2)h))} \right] \\ &= \exp\left[-\frac{j^2}{4(3j+2)}(h/M) \right] \le \exp\left(-\frac{h}{20M} j \right). \end{aligned}$$

Summing over the values of j, we obtain that if $h \ge 20M$ then

$$\mathbb{P}\left(\bigcup_{j \ge 1} A_j \right) \le \frac{\mathrm{e}}{\mathrm{e}-1} \exp\left(-\frac{h}{20M} \right) \le \exp\left(-\frac{h}{20M} j + 1 \right),$$

while for $h \le 20M$ the above inequality holds trivially. Hence, for all $h > 0$,

$$\mathbb{P}\left(\exists t : \left\{ \sum_{i \le t} X_i \le \frac{1}{2} \sum_{i \le t} Y_i \text{ and } \sum_{i \le t} Y_i \ge h \right\} \right) \le \exp\left(-\frac{h}{20M} + 1 \right). \tag{2.1}$$

To complete the proof of the proposition, we repeat the above analysis for

$$Z'_t \stackrel{\triangle}{=} -Z_t = \sum_{i=1}^{t} X_i - Y_i, \qquad V'_t \stackrel{\triangle}{=} \sum_{i=1}^{t} \mathrm{Var}(Z'_i \mid \mathcal{F}_{i-1}) = V_t.$$

Clearly, we again have $|Z'_i - Z'_{i-1}| \le M$ and $V'_t \le M \sum_{i=1}^{t} Y_i$. Defining

$$A'_j = \left(\left\{ \sum_{i \le t} Y_i \le \frac{2}{3} \sum_{i \le t} X_i \text{ and } jh \le \sum_{i \le t} Y_i \le (j+1)h \right\} \text{ for some } t \right),$$

it follows that the event A'_j implies that $Z'_t \geq \frac{1}{2}\sum_i Y_i \geq jh/2$. Therefore, as before, we have that

$$\mathbb{P}(A'_j) \leq \mathbb{P}\left(Z'_t \geq \frac{j}{2}h,\, V'_t \leq (j+1)hM\right) \leq \exp\left(-\frac{h}{20M}j\right),$$

and thus for all $h > 0$

$$\mathbb{P}\left(\exists t : \left\{\sum_{i\leq t} Y_i \leq \frac{2}{3}\sum_{i\leq t} X_i \text{ and } \sum_{i\leq t} Y_i \geq h\right\}\right) \leq \exp\left(-\frac{h}{20M}+1\right). \tag{2.2}$$

Summing the probabilities in (2.1) and (2.2) yields the desired result. □

We note that essentially the same proof yields the following generalization of Proposition 2.1. As before, the constants can be optimized.

PROPOSITION 2.3. *Let (X_i) and (Y_i) be as given in Proposition 2.1. Then for any $0 < \varepsilon \leq \frac{1}{2}$,*

$$\mathbb{P}\left(\left\{\left|\frac{\sum_{i\leq t} X_i}{\sum_{i\leq t} Y_i} - 1\right| \geq \varepsilon \text{ and } \sum_{i\leq t} Y_i \geq h\right\} \text{ for some } t\right)$$
$$\leq \exp\left(-\frac{h\varepsilon^2}{5M}+2\right).$$

REMARK 2.4. The statements of Propositions 2.1 and 2.3 hold also in conjunction with any stopping time τ adapted to the filter $(\mathcal{F}_i)$. That is, we get the same bound on the probability of the mentioned event happening at any time $t < \tau$. This follows easily, for instance, by altering the sequence of increments to be identically 0 after τ. Such statements become useful when the uniform bound on the increments is only valid before τ.

3. Lower bounds on the collisions and load. In this section, we prove Theorem 3 as well as the lower bound in Theorem 1, by showing that if the quantity km/n is suitably small, then any allocation would necessarily produce nearly linearly many bins with arbitrarily large load.

The main ingredient in the proof is a bound for the number of collisions, that is, pairs of balls that share a bin, defined next. Let $N_t(i)$ denote the number of balls in bin i after performing t rounds; the number of collisions at time t is then

$$\mathrm{Col}_2(t) \triangleq \sum_{i=1}^{n} \binom{N_t(i)}{2}.$$

The following theorem provides a lower bound on $\mathrm{Col}_2(t)$ for $t \geq c \cdot km$ for some absolute $c > 0$.

THEOREM 3.1. *Consider n balls and n bins, where each ball has k uniform choices for bins, and $m \geq \log n$ bits of memory are available.*

(i) *For all $t \geq 500 \cdot km$, we have*

$$\mathbb{E}\,\mathrm{Col}_2(t) \geq t^2/(9n).$$

(ii) *Furthermore, with probability $1 - O(n^{-4})$, for all $L = L(n)$ and any $t \geq (500km \vee 30\sqrt{Ln\log n})$, either the maximal load is at least L or*

$$\mathrm{Col}_2(t) \geq t^2/(16n).$$

Note that the main statement of Theorem 3 immediately follows from the above theorem, by choosing $t = (1-\delta)n$ and $L = \sqrt{n}$. Indeed, recalling the assumption in Theorem 3 that $m \geq \log n$, we obtain that, except with probability $O(n^{-4})$, either the algorithm creates a load of $\sqrt{n}$, or it has $\mathrm{Col}_2(n) \geq \frac{(1-\delta)^2}{16}n$. Observing that a load of L immediately induces $\binom{L}{2}$ collisions, we deduce that either way there are at least $\Omega(n)$ collisions w.h.p.

We next prove Theorem 3.1; the statement of Theorem 3 on unbounded maximal load will follow from an iterative application of a more general form of this theorem (namely, Theorem 3.4), which appears in Section 3.1.

PROOF OF THEOREM 3.1. As noted in the Introduction, we relax the model by allowing the algorithm to choose any distribution $\mu = (\mu(1), \ldots, \mu(n))$ for the location of the next ball, as long as it satisfies $\|\mu\|_\infty \leq k/n$.

We also relax the memory constraint as follows. The algorithm has a pool of at most 2^m different strategies, and may choose any of them at a given step without any restriction (basing its dynamic decision on the entire history).

To summarize, the algorithm has a pool of at most 2^m strategies, all of which have an L^∞-norm of at most k/n. In each given round, it adaptively chooses a strategy μ from this pool based on the entire history, and a ball then falls to a bin distributed according to μ.

The outline of the proof is as follows: consider the sequence $Q_1, \ldots, Q_n$, chosen adaptively out of the pool of 2^m of strategies. The large deviation inequality of Section 2 (Proposition 2.1) will enable us to show the following. The expected number of collisions encountered in the above process is well approximated by the expected number of collisions between n independent balls, placed according to $Q_1, \ldots, Q_n$ (i.e., equivalent to the result of the nonadaptive algorithm with strategies $Q_1, \ldots, Q_n$).

Having reduced the problem to the analysis of a nonadaptive algorithm, we may then derive a lower bound on $\mathbb{E}\,\mathrm{Col}_2(t)$ by analyzing the structure of the above strategies. This bound is then translated to a bound on $\mathrm{Col}_2(t)$ using another application of the large deviation inequality of Proposition 2.1.

Let $\nu = (\nu(1), \ldots, \nu(n))$ be an arbitrary probability distribution on $[n]$ satisfying $\|\nu\|_\infty \leq k/n$, and denote by $Q_s = (Q_s(1), \ldots, Q_s(n))$ the strategy of the

algorithm at time s. It will be convenient from time to time to treat these distributions as vectors in $\mathbb{R}^n$.

By the above discussion, Q_s is a random variable whose values belong to some a priori set $\{\mu_1, \ldots, \mu_{2^m}\}$. We further let J_s denote the actual position of the ball at time s (drawn according to the distribution Q_s).

Given the strategy at time s, let x_s denote the probability of a collision between ν and Q_s given J_s, that is, that the ball that is distributed according to ν will collide with the one that arrived in time s. We let v_s be the inner product of Q_s and ν, which measures the expectation of these collisions:

$$x_s^\nu \triangleq \nu(J_s),$$

$$v_s^\nu \triangleq \langle Q_s, \nu\rangle = \sum_{i=1}^n Q_s(i)\nu(i) = \mathbb{E}[x_s^\nu \mid \mathcal{F}_{s-1}].$$

Further define the cumulative sums of v_s^ν and x_s^ν as follows:

$$X_t^\nu \triangleq \sum_{s=1}^t x_s^\nu,$$

$$V_t^\nu \triangleq \sum_{s=1}^t v_s^\nu.$$

To motivate these definitions, notice that given the history up to time $s-1$ and any possible strategy for the next round, ν, we have

$$X_{s-1}^\nu = \sum_{i=1}^{s-1} \nu(J_i) = \sum_{i=1}^n \nu(i)|\{r < s : J_r = i\}| = \sum_{i=1}^n \nu(i)N_{s-1}(i),$$

and so $X_{s-1}^{Q_s}$ is the expected number of collisions that will be contributed by the ball $J_s \sim Q_s$ given the entire history $\mathcal{F}_{s-1}$. Summing over s, we have that

$$\mathbb{E}\,\mathrm{Col}_2(t) = \mathbb{E}\left[\sum_{s=1}^t X_{s-1}^{Q_s}\right],$$

thus estimating the quantities $X_{s-1}^{Q_s}$ will provide a bound on the expected number of collisions. Our aim in the next lemma is to show that w.h.p., whenever $V_{s-1}^{Q_s}$ is large, so is $X_{s-1}^{Q_s}$. This will reduce the problem to the analysis of the quantities $V_{s-1}^{Q_s}$, which are deterministic functions of $Q_1, \ldots, Q_n$. This is the main conceptual ingredient in the lower bound, and its proof will follow directly from the large deviation estimate given in Proposition 2.1.

LEMMA 3.2. *Let $Q_1, \ldots, Q_n$ be a sequence of strategies adapted to the filter $(\mathcal{F}_i)$, and let X_s^ν and V_s^ν be defined as above. Then with probability at*

least $1 - O(e^{-4m})$, *for every* $\nu \in \{\mu_1, \ldots, \mu_{2^m}\}$ *and every* s *we have that* $V_s^\nu \geq 100\|\nu\|_\infty m$ *implies* $X_s^\nu \geq V_s^\nu/2$.

PROOF. Before describing the proof, we wish to emphasize a delicate point. The lemma holds for any sequence of strategies $Q_1, Q_2, \ldots, Q_n$ (each Q_i is an arbitrary function of $\mathcal{F}_{i-1}$). No restrictions are made here on the way each such Q_i is produced (e.g., it does not even need to belong to the pool of 2^m strategies), as long as it satisfies $\|Q_i\|_\infty \leq k/n$. The reason that such a general statement is possible is the following: Once we specify how each Q_i is determined from $\mathcal{F}_{i-1}$ (this can involve extra random bits, in case the adaptive algorithm is randomized), the process of exposing the positions of the balls, $J_i \sim Q_i$, defines a martingale. Hence, for each fixed ν, we would be able to show that the desired event occurs except with probability $O(e^{-5m})$. A union bound over the strategies ν (which, crucially, do belong to the pool of size 2^m) will then complete the proof.

Fix a strategy ν out of the pool of 2^m possible strategies, and recall the definitions of x_s^ν and v_s^ν, according to which

$$0 \leq x_s^\nu \leq \|\nu\|_\infty, \qquad v_s^\nu = \mathbb{E}[x_s^\nu \mid \mathcal{F}_{s-1}].$$

By applying Proposition 2.1 to the sequence (x_s^ν) (with the cumulative sums X_s^ν and cumulative conditional expectations V_s^ν), we obtain that for all h,

$$\mathbb{P}(X_s^\nu \leq V_s^\nu/2, V_s^\nu \geq h \text{ for some } s) \leq \exp\left(-\frac{h}{20\|\nu\|_\infty} + 2\right).$$

Thus, taking $h = 100\|\nu\|_\infty m$ we obtain that

$$\mathbb{P}(X_s^\nu \leq V_s^\nu/2, V_s^\nu \geq 100\|\nu\|_\infty m \text{ for some } s) \leq \exp(-5m + 2).$$

Summing over the pool of at most 2^m predetermined strategies, ν completes the proof. □

Having shown that X_t^ν is well approximated by V_t^ν, and recalling that we are interested in estimating $X_{s-1}^{Q_s}$, we now turn our attention to the possible values of $V_{s-1}^{Q_s}$.

CLAIM 3.3. *For any sequence of strategies* $Q_1, \ldots, Q_t$, *we have that*

$$\sum_{s=1}^{t} V_{s-1}^{Q_s} \geq \frac{t(t-k)}{2n}.$$

PROOF. By our definitions, for the strategies $Q_1, \ldots, Q_t$ we have

$$\begin{aligned}\sum_{s=1}^{t} V_{s-1}^{Q_s} &= \sum_{s=1}^{t}\sum_{r=1}^{s-1}\langle Q_r, Q_s\rangle = \sum_{i=1}^{n}\sum_{r<s\leq t} Q_r(i)Q_s(i) \\ &= \frac{1}{2}\sum_{i=1}^{n}\left[\left(\sum_{s=1}^{t} Q_s(i)\right)^2 - \sum_{s=1}^{t} Q_s(i)^2\right].\end{aligned} \tag{3.1}$$

Recalling the definition of the strategies Q_i, we have that

$$\begin{cases} 0 \le Q_s(i) \le k/n, & \text{for all } i \text{ and } s, \\ \sum_{i=1}^{n} Q_s(i) = 1, & \text{for all } s. \end{cases}$$

Therefore,

$$\sum_{i=1}^{n}\sum_{s=1}^{t} Q_s(i)^2 \le \frac{k}{n}\sum_{i=1}^{n}\sum_{s=1}^{t} Q_s(i) = \frac{kt}{n}.$$

On the other hand, by Cauchy–Schwarz,

$$\sum_{i=1}^{n}\left(\sum_{s=1}^{t} Q_s(i)\right)^2 \ge \frac{1}{n}\left(\sum_{i=1}^{n}\sum_{s=1}^{t} Q_s(i)\right)^2 = \frac{t^2}{n}.$$

Plugging these two estimates in (3.1), we deduce that

$$\sum_{s=1}^{t} V_{s-1}^{Q_s} \ge \frac{t(t-k)}{2n},$$

as required. □

While the above claim tells us that the average size of $V_{s-1}^{Q_s}$ is fairly large [has order at least $(t-k)/n$], we wish to obtain bounds corresponding to individual distributions Q_s. As we next show, this sum indeed enjoys a significant contribution from indices s where $V_{s-1}^{Q_s} = \Omega(km/n)$. More precisely, setting $h = 100km/n$, we claim that for large enough n,

$$\sum_{s=1}^{t} V_{s-1}^{Q_s}\mathbf{1}_{\{V_{s-1}^{Q_s} > h\}} \ge \frac{t^2}{4n}. \tag{3.2}$$

To see this, observe that if

$$t \ge t_0 \stackrel{\triangle}{=} 5hn = 500km,$$

then

$$\sum_{s=1}^{t} V_{s-1}^{Q_s}\mathbf{1}_{\{V_{s-1}^{Q_s} \le h\}} \le th \le \frac{t^2}{5n}.$$

Combining this with Claim 3.3 [while noting that $\frac{t(t-k)}{2n} = (1-o(1))\frac{t^2}{2n}$] yields (3.2) for any sufficiently large n.

We may now apply Lemma 3.2, and obtain that, except with probability $O(\mathrm{e}^{-4m})$, whenever $V_{s-1}^{Q_s} > h$ we have $X_{s-1}^{Q_s} \ge \frac{1}{2}V_{s-1}^{Q_s}$, and so

$$\sum_{s=1}^{t} X_{s-1}^{Q_s} \ge \frac{1}{2}\sum_{s=1}^{t} V_{s-1}^{Q_s}\mathbf{1}_{\{V_{s-1}^{Q_s} > h\}} \ge \frac{t^2}{8n} \qquad \text{for all } t \ge t_0. \tag{3.3}$$

Altogether, since $X_{s-1}^{Q_s} \geq 0$, we infer that

$$\mathbb{E}\,\mathrm{Col}_2(t) = \mathbb{E}\left[\sum_{s=1}^{t} X_{s-1}^{Q_s}\right] \geq \frac{t^2}{8n}\big(1 - O(n^{-4})\big) \geq \frac{t^2}{9n} \qquad \text{for all } t \geq t_0, \tag{3.4}$$

where the last inequality holds for large enough n. This proves part (i) of Theorem 3.1.

It remains to establish concentration for $\mathrm{Col}_2(t)$ under the additional assumption that $t \geq 30\sqrt{Ln\log n}$ for some $L = L(n)$. First, set the following stopping-time for reaching a maximal-load of L:

$$\tau_L \stackrel{\triangle}{=} \min\Big\{t : \max_j N_t(j) \geq L\Big\}.$$

Next, recall that

$$\mathrm{Col}_2(t) = \sum_{s=1}^{t} N_{s-1}(J_s),$$

and notice that

$$\mathbb{E}[N_{s-1}(J_s) \mid \mathcal{F}_{s-1}] = \sum_{i=1}^{n} Q_s(i) N_{s-1}(i) = X_{s-1}^{Q_s}.$$

Therefore, we may apply our large deviation estimate given in Section 2 (Proposition 2.1), combined with the stopping-time τ_L (see Remark 2.4):

- The sequence of increments is $(N_{s-1}(J_s))$.
- The sequence of conditional expectations is $(X_{s-1}^{Q_s})$.
- The bound on the increments is L, as $N_{s-1}(J_s) \leq \max_i N_{s-1}(i) \leq L$ for all $s < \tau_L$.

It follows that

$$\mathbb{P}\left(\left\{\mathrm{Col}_2(t) \leq \frac{1}{2}\sum_{s\leq t} X_{s-1}^{Q_s} \text{ and } \sum_{s\leq t} X_{s-1}^{Q_s} \geq \frac{t^2}{8n}\right\} \text{ for some } t < \tau_L\right)$$

$$\leq \exp\left(-\frac{t^2/8n}{20L} + 2\right) \leq O(n^{-5}),$$

where the last inequality is by the assumption $t \geq 30\sqrt{Ln\log n}$. Finally, by (3.3), we also have that $\sum_{s\leq t} X_{s-1}^{Q_s} \geq t^2/(8n)$ for all $t \geq t_0$, except with probability $O(n^{-4})$. Combining these two statements, we deduce that for any $t \geq (t_0 \vee 30\sqrt{Ln\log n})$,

$$\mathbb{P}\left(\mathrm{Col}_2(t) < \frac{t^2}{16n},\ \tau_L > t\right) = O(n^{-4}),$$

concluding the proof of Theorem 3.1. □

3.1. *Boosting the subcritical regime to unbounded maximal load.* While Theorem 3.1 given above provides a careful analysis for the number of 2-collisions, that is, pairs of balls sharing a bin, one can iteratively apply this theorem, with very few modifications, in order to obtain that the number of q-collisions (a set of q balls sharing a bin) has order $\Omega(n^{1-o(1)})$ w.h.p. The proof of this result hinges on Theorem 3.4 below, which is a generalization of Theorem 3.1.

Recall that in the relaxed model studied so far, at any given time t the algorithm adaptively selects a strategy Q_t (based on the entire history $\mathcal{F}_{t-1}$), after which a ball is positioned in a bin $J_t \sim Q_t$. We now introduce an extra set of random variables, in the form of a sequence of increasing subsets, $A_1 \subset \cdots \subset A_n \subset [n]$. The set A_t is determined by $\mathcal{F}_{t-1}$, and has the following effect: If $J_t \in A_t$, we add a ball to this bin as usual, whereas if $J_t \notin A_t$, we ignore this ball (all bins remain unchanged). That is, the number of balls in bin i at time t is now given by

$$N_t(i) \stackrel{\triangle}{=} \sum_{s=1}^{t} \mathbf{1}_{\{J_s=i\}} \mathbf{1}_{A_s}(i),$$

and as before we are interested in a lower bound for the number of collisions:

$$\mathrm{Col}_2(t) \stackrel{\triangle}{=} \sum_{i=1}^{n} \binom{N_t(i)}{2}.$$

The idea here is that, in the application, the set A_t will consists of the bins that already contain ℓ balls at time t. As such, they indeed form an increasing sequence of subsets determined by $(\mathcal{F}_i)$. In this case, any collision corresponds to 2 balls placed in some bin which already has ℓ other balls, and thus immediately implies a load of $\ell + 2$.

THEOREM 3.4. *Consider the following balls and bins setting*:

(1) *The online adaptive algorithm has a pool of* 2^m *possible strategies, where each strategy* μ *satisfies* $\|\mu\|_\infty \le k/n$. *The algorithm selects a* (*random*) *sequence of strategies* $Q_1, \ldots, Q_n$ *adapted to the filter* $(\mathcal{F}_i)$.
(2) *Let* $A_1 \subset \cdots \subset A_n \subset [n]$ *denote a random increasing sequence of subsets adapted to the filter* $(\mathcal{F}_i)$, *that is,* A_i *is determined by* $\mathcal{F}_{i-1}$.
(3) *There are n rounds, where in round t a new potential location for a ball is chosen according to* Q_t. *If this location belongs to* A_t, *a ball is positioned there* (*otherwise, nothing happens*).

Define $T = \sum_{s=1}^{n} Q_s(A_s)$. *Then for any* $L = L(n)$,

$$\mathbb{P}\left(T \ge 30\left(\sqrt{kmn} \vee \sqrt{Ln \log n}\right), \mathrm{Col}_2(n) < \frac{T^2}{16n}, \max_j N_n(j) \le L \right) \le O(n^{-4}).$$

PROOF. As the proof follows the same arguments of Theorem 3.1, we restrict our attention to describing the modifications that are required for the new statement to hold.

Define the following subdistribution of Q_s with respect to A_s:

$$Q'_s \triangleq Q_s \mathbf{1}_{A_s}.$$

As before, given Q_s, the strategy at time s, define the following parameters:

$$x_s^\nu \triangleq \nu(J_s), \qquad v_s^\nu \triangleq \sum_{i=1}^{n} Q'_s(i)\nu(i),$$

and let the cumulative sums of v_s^ν and x_s^ν be denoted by

$$X_t^\nu \triangleq \sum_{s=1}^{t} x_s^\nu, \qquad V_t^\nu \triangleq \sum_{s=1}^{t} v_s^\nu.$$

We claim that a statement analogous to that of Lemma 3.2 holds as is with respect to the above definitions, for any choice of increasing subsets $A_1 \subset \cdots \subset A_n$ [adapted to the filter $(\mathcal{F}_i)$]. As we soon argue, the martingale concentration argument is valid without any changes, and the only delicate point is the identity of the target strategy ν.

LEMMA 3.5. *Let $Q_1, \ldots, Q_n$ and $A_1 \subset \cdots \subset A_n$ be strategies and subsets respectively, adapted to the filter $(\mathcal{F}_i)$, and let X_s^ν and V_s^ν be defined as above. Then with probability at least $1 - O(\mathrm{e}^{-4m})$, for every $\nu \in \{\mu_1, \ldots, \mu_{2^m}\}$ and every s we have that $V_s^{\nu'} \geq 100\|\nu\|_\infty m$ implies $X_s^{\nu'} \geq V_s^{\nu'}/2$, where $\nu' = \nu \mathbf{1}_{A_{s+1}}$.*

PROOF. Let ν be a strategy. Previously (in the proof of Lemma 3.2), we compared X_s^ν to V_s^ν using the large deviation inequality of Section 2. Now, for each s, our designated ν' is a function of ν and A_{s+1}, and hence depends on $\mathcal{F}_s$. In particular, there are potentially more than 2^m different strategies to consider as ν', destroying our union bound. The crucial observation that resolves this issue is the following.

OBSERVATION 3.6. Let $r > s$ and let ν be a strategy. Then $V_s^\nu = V_s^{\nu'}$ and $X_s^\nu = X_s^{\nu'}$ for any increasing sequence $A_1, \ldots, A_r$, where $\nu' = \nu \mathbf{1}_{A_r}$.

To see this, first consider X_s^ν and $X_s^{\nu'}$. If x_i^ν for some $1 \leq i \leq s$ had a nonzero contribution to X_s^ν, then by definition $J_i \in A_i$. Since $A_i \subset A_r$, we also have $J_i \in A_r$, and so $x_i^{\nu'} = \nu(J_i)\mathbf{1}_{A_r}(J_i) = x_i^\nu$. The statement $V_s^{\nu'} = V_s^\nu$ now follows from the fact that V_s^ν is the sum of $v_i^\nu = \mathbb{E}[x_i^\nu \mid \mathcal{F}_{i-1}]$.

Using the above observation, it now suffices to prove the statement of Lemma 3.5 directly on the strategies ν (rather than on ν'). Hence, the only difference between this setting and that of Lemma 3.2 is that here some of the rounds are

forfeited (as reflected in the new definition of the v_s^v's). The proof of Lemma 3.2 therefore holds unchanged for this case. □

Similarly, the following claim is the analogue of Claim 3.3, with t (the number of balls in the original version) replaced by $T = \sum_s \sum_i Q'_s(i)$ (the expected number of balls actually positioned).

CLAIM 3.7. *For any $Q_1, \ldots, Q_n$ and $A_1 \subset \cdots \subset A_n$, we have that*

$$\sum_{s=1}^{n} V_{s-1}^{Q'_s} \geq \frac{T(T-k)}{2n}.$$

The proof of the above claim follows from the exact same argument as in Claim 3.3. Notice that the bound there, given as a function of t, was actually a bound in terms of $\sum_{s=1}^{t} \sum_i Q_s(i)$, and so replacing Q_s by Q'_s yields the desired bound as a function of T.

With this in mind, set $h = 100km/n$ and note that, clearly,

$$\sum_{s=1}^{n} V_{s-1}^{Q'_s} \mathbf{1}_{\{V_{s-1}^{Q'_s} \leq h\}} \leq hn.$$

Therefore, if

$$t_0 \triangleq n\sqrt{5h} \leq 25\sqrt{kmn},$$

then

$$hn \leq \frac{T^2}{5n} \qquad \text{for any } T \geq t_0,$$

and so for such T and any large enough n

$$\sum_{s=1}^{n} V_{s-1}^{Q'_s} \mathbf{1}_{\{V_{s-1}^{Q'_s} > h\}} \geq \frac{T^2}{4n}.$$

By following the next arguments from the proof of Theorem 3.1, it now follows that, as long as $T \geq t_0$,

$$\mathbb{E}\,\mathrm{Col}_2(n) = \mathbb{E}\left[\sum_{s=1}^{t} X_{s-1}^{Q'_s}\right] \geq \frac{T^2}{8n}\left(1 - O(n^{-4})\right) \geq \frac{T^2}{9n}.$$

Similarly, using the argument as in the proof of Theorem 3.1, which defines the stopping-time τ_L and applies Proposition 2.1 on the sequence of increments given by

$$\mathrm{Col}_2(t) - \mathrm{Col}_2(t-1) = N_{s-1}(J_s)\mathbf{1}_{A_s}(J_s),$$

we deduce that, if $T \geq (t_0 \vee 30\sqrt{Ln\log n})$ then

$$\mathbb{P}\left(\mathrm{Col}_2(n) < \frac{T^2}{16n}, \tau_L > n\right) = O(n^{-4}),$$

as required. □

We next show how to infer the results regarding an unbounded maximal load from Theorem 3.4. For each integer $\ell = 0, 1, 2, \ldots$, we define the increasing sequence (A_t) by

$$A_t^\ell \triangleq \{i \in [n] : N_t(i) \geq \ell\}.$$

Further define

$$T_\ell \triangleq \sum_{s=1}^{n} Q_s(A_s^\ell),$$

which is the expected number of balls that are placed in bins which already hold at least ℓ balls. The proof will follow from an inductive argument, which bounds the value of $T_{\ell+1}$ in terms of T_ℓ.

For some $L = L(n)$ to be specified later, our bounds will be meaningful as long as the maximal load is at most L, and

$$T_\ell \geq 30(\sqrt{kmn} \vee \sqrt{Ln\log n}). \tag{3.5}$$

Using Theorem 3.4, we will show that, if (3.5) holds then

$$T_{\ell+1} \geq \frac{T_\ell^2}{20nL}. \tag{3.6}$$

To this end, define

$$R_\ell \triangleq \sum_{i=1}^{n} \binom{N_n(i) - \ell}{2},$$

that is, R_ℓ denotes the number of collisions between all pairs of balls that were placed in a bin, that already held at least ℓ balls.

To infer (3.6), apply Theorem 3.4 with respect to the subsets (A_s^ℓ). The assumption (3.5) implies that, except with probability $O(n^{-4})$, either the load is at least L, or

$$R_\ell \geq \frac{T_\ell^2}{16n}.$$

Notice that any ball that is placed in a bin, which contains at most L balls, can contribute at most L collisions to the count of R_ℓ. Therefore, if the maximal load

is less than L, the following holds: The number of balls placed in bins that already contain at least ℓ balls, is at least

$$R_\ell/L \geq \frac{T_\ell^2}{16nL} \qquad \text{with probability } 1 - O(n^{-4}). \tag{3.7}$$

Recalling that $T_{\ell+1}$ is the expected number of such balls, we infer that

$$T_{\ell+1} \geq \big(1 - O(n^{-4})\big)\frac{R_\ell}{L} \geq \frac{T_\ell^2}{20nL},$$

where the last inequality holds for large enough n (with room to spare).

This establishes that (3.5) implies (3.6). Since by definition $T_0 = n$, we deduce that the decreasing series $(T_0, T_1, \ldots)$ satisfies

$$T_{\ell+1} \geq \frac{n}{(20L)^{2^{\ell+1}-1}} \qquad \text{if } T_\ell \text{ satisfies (3.5).}$$

Rearranging, it follows that, in particular, (3.5) is satisfied if

$$30 \cdot (20L)^{2^\ell - 1} \leq \sqrt{\frac{n}{km}} \wedge \sqrt{\frac{n}{L \log n}}. \tag{3.8}$$

It is now easy to verify that, for any fixed $\varepsilon > 0$, choosing

$$L = \ell = (1 - \varepsilon) \log_2 \log\left(\frac{n}{km}\right)$$

satisfies (3.8) for large enough n. By (3.7), we can then infer that $R_\ell > 0$ with probability $1 - O(n^{-4})$, hence the maximal load is at least ℓ. This concludes the proof of Theorem 3. □

4. Algorithms for perfect matching and constant load. In this section, we prove Theorem 4 by providing an algorithm that avoids collisions w.h.p. using only $O(n/k)$ bits of memory, which is the minimum possible by Theorem 3. The case $km = \Omega(n)$ of Theorem 1 will then follow from repeated applications of this algorithm.

PERFECT ALLOCATION ALGORITHM FOR $(1 - \delta)n$ BALLS.

1. For $\ell = \lfloor \frac{n}{\lfloor m/2 \rfloor} \rfloor$, partition the bins into contiguous blocks $B_1, \ldots, B_\ell$ each comprising $\lfloor m/2 \rfloor$ bins. Ignore any remaining unused bins.
2. Set $d = \lceil \log_2(\frac{5}{C\delta} \log n) \rceil$, and define the arrays $A_0, \ldots, A_{d-1}$:
   - A_j comprises 2^j contiguous blocks (a total of $\sim 2^{j-1}m$ bins).
   - For each contiguous (nonoverlapping) 4^j-tuple of bins in A_j, we keep a single bit that holds whether any of its bins is occupied.
   - All blocks currently or previously used are contiguous.

3. Repeat the following procedure until exhausting all rounds:
   - Let j be the minimal integer so that a bin of A_j, marked as empty, appears in the current selection of k bins. If no such j exists, the algorithm announces failure.
   - Allocate the ball into this bin, and mark its 4^j-tuple as occupied.
   - If the fraction of empty 4^j-tuples remaining in A_j just dropped below $\delta/2$, relocate the array A_j to a fresh contiguous set of empty 2^j blocks (immediately beyond the last allocated block). If there are less than 2^j available new blocks, the algorithm fails.
4. Once $(1-\delta)n$ rounds are performed, the algorithm stops.

We proceed to verify the validity of the algorithm in stages: First, we discuss a more basic version of the algorithm suited for the case where $km = \Omega(n \log n)$; then, we examine an intermediate version which extends the range of the parameters to $km \log m = \Omega(n \log n)$; finally, we study the actual algorithm, which features the tight requirement $km = \Omega(n)$.

Throughout the proof of the algorithm, assume that in each round we are presented with k independent uniform indices of bins, possibly with repetitions. Clearly, an upper bound for the maximal load in this relaxed model translates into one for the original model (k choices without repetitions).

4.1. *Basic version of the algorithm.* We begin with a description and a proof of a simpler version of the above algorithm, suited for the case where

$$km \geq (3/\delta)n \log n. \tag{4.1}$$

This version will serve as the base for the analysis. For simplicity, assume first that $m \mid n$.

BASIC VERSION OF ALLOCATION ALGORITHM FOR $(1-\delta)n$ BALLS.

1. Let $B_1, \ldots, B_\ell$ be an arbitrary partition of the n bins into $\ell \stackrel{\triangle}{=} n/m$ blocks, each containing m bins. Put $r \stackrel{\triangle}{=} \lfloor (1-\delta)m \rfloor$.
2. Throughout stage $j \in [\ell]$, only the m bins belonging to B_j are tracked. At the beginning of the stage, all bins in the block are marked empty.
3. Stage j comprises r rounds, in each of which:
   - The algorithm attempts to place a ball in an arbitrary empty bin of B_j if possible.
   - If no empty bin of B_j is offered, the algorithm declares failure.
4. Once $(1-\delta)n$ rounds are performed, the algorithm stops.

To verify that this algorithm indeed produces a perfect allocation w.h.p., examine a specific round of stage j, and condition on the event that so far the algorithm did not fail. In particular, its accounting of which bins are occupied in B_j is accurate, and at least $m - r = (\delta - o(1))m$ bins in B_j are still empty [notice that by our assumption $m = \Omega(\log n)$, and so $m \to \infty$ with n].

Let Miss_j denote the event that the next ball precludes all of the empty bins of B_j in its k choices, we have

$$\mathbb{P}(\mathsf{Miss}_j) \leq \left(1 - \frac{m-r}{n}\right)^k \leq \mathrm{e}^{-(\delta - o(1))(km/n)} \leq n^{-3+o(1)}, \tag{4.2}$$

by assumption (4.1). A union bound over the n rounds now yields (with room to spare) that the algorithm succeeds w.h.p.

The case where m does not divide n is treated similarly: Set $\ell = \lfloor \frac{n}{\lfloor m/2 \rfloor} \rfloor$, and partition the bins into blocks that now hold $\lfloor m/2 \rfloor$ bins each, except for the final block B_ℓ which would have between $\lfloor m/2 \rfloor$ and $m - 1$ bins. As before, in stage j we attempt to allocate $\lfloor (1-\delta)|B_j| \rfloor$ balls into B_j, while relying on the property that B_j has at least $(\delta - o(1))|B_j| \geq (\delta - o(1))m/2$ empty bins. This gives

$$\mathbb{P}(\mathsf{Miss}_j) \leq \mathrm{e}^{-(\delta - o(1))((km/2)/n)} \leq n^{-3/2+o(1)},$$

as required.

4.2. *Intermediate version of the algorithm.* We now wish to adapt the above algorithm to the following case:

$$km \log_2 m \geq (20/\delta) \log(5/\delta) n \log n, \qquad \log^3 n \leq m \leq \frac{n}{\log n}. \tag{4.3}$$

Notice that if $m \geq n^\varepsilon$, the above requirement is essentially that

$$km = \Omega(n/\varepsilon).$$

The full version of the algorithm will eliminate this dependency on ε.

INTERMEDIATE VERSION OF ALLOCATION ALGORITHM FOR $(1-\delta)n$ BALLS.

1. For $\ell = \lfloor \frac{n}{\lfloor m/2 \rfloor} \rfloor$, partition the bins into contiguous blocks $B_1, \ldots, B_\ell$ each comprising $\lfloor m/2 \rfloor$ bins. Ignore any remaining unused bins.
2. Set $d = \lfloor \frac{1}{4} \log_2 m \rfloor$, and define the arrays $A_0, \ldots, A_{d-1}$:
   - A_j is one of the blocks $B_1, \ldots, B_\ell$.
   - For each contiguous (non-overlapping) 2^j-tuple of bins in A_j, we keep a single bit that holds whether any of its bins is occupied.
3. Repeat the following procedure until exhausting all rounds:
   - Let j be the minimal integer so that a bin of A_j, marked as empty, appears in the current selection of k bins. If no such j exists, the algorithm announces failure.
   - Allocate the ball into this bin, and mark its 2^j-tuple as occupied.

- If the fraction of empty 2^j-tuples remaining in A_j just dropped below $\delta/2$, relocate the array A_j to a fresh block (immediately beyond the last allocated block). If no such block is found, the algorithm fails.

4. Once $(1-\delta)n$ rounds are performed, the algorithm stops.

Since the array A_j contains $2^{-j}(m/2)$ different 2^j-tuples, the amount of memory required to maintain the status of all tuples is

$$\frac{m}{2}\sum_{j=0}^{d-1} 2^{-j} = (1-2^{-d})m \le m - m^{3/4}.$$

In addition, we keep an index for each A_j, holding its position among the ℓ blocks. By definition of d and ℓ, this amounts to at most

$$d\log_2 \ell \le (\log_2 n)^2 < m^{3/4}$$

bits of memory, where the last inequality holds for any large n by (4.3).

We first show that the algorithm does not fail to find a bin of A_j marked as empty. At any given point, each A_j has a fraction of at least $\delta/2$ bins marked as empty. Hence, recalling (4.2), the probability of missing all the bins marked as empty in $A_0, \dots, A_{d-1}$ is at most

$$\begin{aligned} &\exp\left[-\left(\frac{\delta}{2}-o(1)\right)\frac{km}{2n}d\right] \\ &\le \exp\left[-\left(\frac{\delta}{2}-o(1)\right)\frac{10\log n}{\delta\log_2 m}\log\left(\frac{20}{\delta}\right)\frac{1}{4}\log_2 m\right] \\ &\le n^{-\log(5/\delta)5/4-o(1)} < n^{-5/4}, \end{aligned} \tag{4.4}$$

where the last inequality holds for large n. Therefore, w.h.p. the algorithm never fails to find an array A_j with an empty bin among the k choices.

It remains to show that, whenever the algorithm relocates an array A_j, there is always a fresh block available.

By the above analysis, the probability that a ball is allocated in A_j for $j \ge 1$ at a given round is at most

$$\begin{aligned} \exp\left[-\left(\frac{\delta}{2}-o(1)\right)\frac{km/2}{n}j\right] &\le \exp\left[-\left(\frac{\delta}{2}-o(1)\right)\frac{10\log n}{\delta\log_2 m}\log\left(\frac{20}{\delta}\right)j\right] \\ &\le \exp\bigl(-3\log(5/\delta)\,j\bigr) \stackrel{\triangle}{=} p_j, \end{aligned}$$

where the last inequality holds for any sufficiently large n.

Let N_j denote the number of balls that were allocated in blocks of type j throughout the run of the algorithm. Clearly, N_j is stochastically dominated by

a binomial random variable $\mathrm{Bin}(n, p_j)$. Hence, known estimates for the binomial distribution (see, e.g., [2]) imply that for all j,

$$\mathbb{P}\big(N_j > np_j + C\sqrt{n}\log n\big) \le n^{-C}.$$

The total number of blocks needed for A_j is at most

$$\left\lceil \frac{2^j N_j}{(1-\delta/2)(m/2)} \right\rceil,$$

and hence the total number of blocks needed is w.h.p. at most

$$\left\lceil \sum_{j=0}^{d-1} \frac{2^j(1-\delta)np_j + C2^j\sqrt{n}\log n}{(1-\delta/2)(m/2)} \right\rceil$$

$$\le \sum_{j=0}^{d-1} \frac{2^j(1-\delta)np_j}{(1-\delta/2)(m/2)} + O\left(\frac{n^{3/4}\log n}{m}\right).$$

Since

$$\sum_{j=1}^{d-1} 2^j p_j = \sum_{j=1}^{d-1} \exp\big(j\big(\log 2 - 3\log(5/\delta)\big)\big) < 2\cdot 2(\delta/5)^3 < \delta/5$$

(with room to spare), the total number of blocks needed is w.h.p. at most

$$\frac{(1+\delta/5)(1-\delta)n}{(1-\delta/2)(m/2)} + O\left(\frac{n^{3/4}\log n}{m}\right) < \left\lfloor \frac{n}{\lfloor m/2\rfloor} \right\rfloor = \ell$$

for any sufficiently large n.

4.3. *Final version of the algorithm.* The main disadvantage in the intermediate version of the algorithm is that the size of each A_j was fixed at $m/2$ bins. Since the resolution of each A_j is in 2^j-tuples, we are limited to at most $\log_2 m$ arrays. However, the probability of missing all the arrays $A_0, \ldots, A_{d-1}$ has to compete with n, hence the requirement that m would be polynomial in n.

To remedy this, the algorithm uses arrays with increasing sizes, namely 2^j blocks for A_j. The resolution of each array is now in 4^j-tuples, that is, tracking the status of A_j now requires at most $2^j\lfloor m/2\rfloor/4^j \le m/2^{j+1}$ bits. Recalling that $d = \lceil \log_2(\frac{5}{C\delta}\log n)\rceil$, the number of memory bits required for all arrays is at most

$$\frac{m}{2}\sum_{j=0}^{d-1} 2^{-j} = (1-2^{-d})m \le m - O(m/\log n). \tag{4.5}$$

The following calculation shows that indeed there are sufficiently many blocks to initially accommodate all the arrays:

$$(2^d-1)\lfloor m/2\rfloor \le \frac{5}{2C\delta} m\log n \le \frac{5km}{6C} = \frac{5}{6}n,$$

where we used the assumptions $k \geq (3/\delta)\log n$ and $km = Cn$.

Each of the arrays comes along with a pointer to its starting block, and the total number of memory bits required for this is at most

$$d\log_2(2n/m) \leq \big(\log_2\log n + O(1)\big)\log_2 n = \big(1+o(1)\big)\log_2 n \cdot \log_2\log n.$$

When $m = \Omega(\log^3 n)$, the space for these pointers clearly fits among the $O(m/\log n)$ bits remaining according to (4.5). For smaller values of m, as before we can apply the algorithm for, say, $m' = m/3$ (after tripling the constant C_δ to reflect this change), thus earning $2m/3$ bits for the pointers (recall the requirement that $m \geq \log_2\log n \cdot \log_2 n$).

As final evidence that the choice of parameters for the algorithm is valid, note that each A_j indeed contains many 4^j-tuples. It suffices to check A_{d-1}, which indeed comprises about

$$(1+o(1))\frac{m}{2}\frac{2^{d-1}}{4^{d-1}} = (1+o(1))m/2^d = \left(\frac{C\delta}{5}+o(1)\right)\frac{m}{\log n} = \Omega(\log\log n)$$

4^{d-1}-tuples, where the last equality is by the assumption on the order of m.

It remains to verify that the algorithm succeeds w.h.p. This will follow from the same argument as in the intermediate version of the algorithm. In that version, each A_j contained at least a fraction of $(\delta/2)$ empty bins, and $|A_j|$ was about $m/2$ for all j. In the final version of the algorithm, each A_j again contains at least a fraction of $(\delta/2)$ empty bins, but crucially, now A_j contains 2^j bins. Thus, recalling (4.4), the probability to miss $A_0, \ldots, A_{d-1}$ in a given round is now at most

$$\exp\left[-\left(\frac{\delta}{2}-o(1)\right)\frac{km}{2n}\sum_{j=0}^{d-1}2^j\right] \leq \exp\left(-(1-o(1))\frac{C\delta}{4}(2^d-1)\right)$$
$$= n^{-5/4-o(1)},$$

where the last inequality is by the definition of d. A union bound over the n rounds gives that, w.h.p., an array A_j with an empty bin is found for every ball.

To see that w.h.p. there are always sufficiently many available fresh blocks to relocate an array, one essentially repeats the argument from the intermediate version of the algorithm. That is, we again examine the probability that a ball is allocated in A_j, to obtain that this time

$$p_j = \exp\left(-(1-o(1))\frac{C\delta}{4}(2^j-1)\right).$$

A choice of $C \geq K(1/\delta)\log(1/\delta)$ with some suitably large $K > 0$ would give

$$\sum_{j=1}^{d-1} 4^j p_j < \delta/5,$$

and the rest of that argument unchanged now implies that the algorithm never runs out of fresh blocks w.h.p.

This completes the proof of Theorem 4.

4.4. *Proof of upper bound in Theorem 1.* We now wish to apply the algorithm from Theorem 4 in order to obtain a constant load in the case where $km \geq cn$ for some $c > 0$. To achieve this, consider the perfect matching algorithm for, say, $\delta = \frac{1}{2}$, and let C_δ be the constant that appears in Theorem 4. Next, join every consecutive $\lceil C_\delta/c \rceil$-tuple of bins together and write n' for the number of such tuples. As $km \geq Cn'$, we may apply the perfect-matching algorithm for $n'/2$ balls with respect to the n' tuples of bins, keeping in mind that the algorithm is valid also for the model of repetitions. This gives a perfect matching w.h.p., and repeating this process gives a total load of at most $2C_\delta/c = O(1)$ for all n balls. □

5. Improved lower bounds for poly-logarithmic choices.

5.1. *Proof of Theorem 2.* Our proof of this case is an extension of the proof of Theorem 3. We now wish to estimate the number of q-collisions for general q:

$$\mathrm{Col}_q(t) \triangleq \sum_{i=1}^{n} \binom{N_t(i)}{q}.$$

The analysis hinges on a recursion on q, for which we need to achieve bounds on a generalized quantity, a linear function of the q-collisions vector:

$$X_t^{f;q} \triangleq \sum_{s_1<\cdots<s_q\leq t} \sum_i f(i)\mathbf{1}_{\{J_{s_1}=i\}}\cdots\mathbf{1}_{\{J_{s_q}=i\}} = \sum_i f(i)\binom{N_t(i)}{q}, \tag{5.1}$$

$$V_t^{f;q} \triangleq \sum_{s_1<\cdots<s_q\leq t} \sum_i f(i)Q_{s_1}(i)\cdots Q_{s_q}(i). \tag{5.2}$$

Our objective is to obtain lower bounds for $X_t^{f;q}$ with $f \equiv 1$, as clearly $\mathrm{Col}_q(t) = X_t^{\mathbf{1};q}$. Notice that the parameters X_t^ν, V_t^ν from Section 3 are exactly $X_t^{\nu;1}$, $V_t^{\nu;1}$ defined above. There, ν was a strategy, whereas now our f will be the product of different strategies. This fact will allow us to formulate a recursion relation between the $V_t^{f;q}$'s and an approximate recursion for the $X_t^{f;q}$. We achieve this using the next lemma, where here in and throughout the proof we let

$$L \triangleq \log(n/m) \tag{5.3}$$

denote a maximal load we do not expect to reach (except if the algorithm is far from optimal). We further define

$$\Gamma \triangleq \left\{ \prod_{i=1}^{L} f_i : f_i \in \{\mathbf{1}, \mu_1, \ldots, \mu_{2^m}\} \text{ for all } i \right\}$$

to be the set of all point-wise products of at most L strategies from the pool.

LEMMA 5.1. *Either the maximal load exceeds L, or the following holds for all $q < L$, every $t \le n/k$ and every $f \in \Gamma$, except with probability e^{-3mL}:*

$$\text{If } V_t^{f;q} \ge 100\frac{(3L)^{q+1}}{q!}m\|f\|_\infty \qquad \text{then } X_t^{f;q} \ge 3^{-q}V_t^{f;q}. \tag{5.4}$$

PROOF. The key property of the quantities $V_t^{f;q}$, which justified the inclusion of the inner products with f, is the following recursion relation, whose validity readily follows from definition (5.2):

$$V_t^{f;q+1} = \sum_{s<t} V_s^{(Q_{s+1}\cdot f);q} \qquad \text{for any } q \ge 1 \text{ and any } t. \tag{5.5}$$

We now wish to write a similar recursion for the variables $X_t^{f;q}$. As opposed to the variables $V_t^{f;q}$, which satisfied the above recursion combinatorially, here the recursion will only be stochastic. Notice that

$$\begin{aligned} X_{t+1}^{f;q+1} - X_t^{f;q+1} &= f(J_{t+1})\left(\binom{N_t(J_{t+1})+1}{q+1} - \binom{N_t(J_{t+1})}{q+1}\right) \\ &= f(J_{t+1})\binom{N_t(J_{t+1})}{q}, \end{aligned}$$

and hence

$$\mathbb{E}[X_{t+1}^{f;q+1} - X_t^{f;q+1} \mid \mathcal{F}_t] = \sum_i Q_{t+1}(i)f(i)\binom{N_t(i)}{q} = X_t^{(Q_{t+1}\cdot f);q}.$$

We may therefore apply Proposition 2.1 as follows:

- The sequence of increments we consider is $(X_{t+1}^{f;q+1} - X_t^{f;q+1})$ (that results in a telescopic sum).
- The sequence of conditional expectations is $(X_t^{(Q_{t+1}\cdot f);q})$.
- The bound on the increment is $M = \|f\|_\infty\binom{L}{q}$, where L is an upper bound for the maximal load (if we encounter a load of L, we stop the process).

This implies that

$$\begin{aligned} &\mathbb{P}\left(\exists t : \left\{X_t^{f;q+1} \le \frac{1}{2}\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q}, \sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} \ge 100mL\|f\|_\infty\binom{L}{q}\right\}\right) \\ &\quad \le \exp\left(-\frac{100mL\|f\|_\infty\binom{L}{q}}{20\|f\|_\infty\binom{L}{q}} + 2\right) = O(\exp(-5mL)). \end{aligned}$$

As a result, the above event does not occur for any $f \in \Gamma$ (since there are at most 2^{mL} such functions) except with probability e^{-4mL}. Therefore, setting

$$h_{f;q} \triangleq 100\frac{(3L)^{q+1}}{q!}m\|f\|_\infty,$$

we have that, except with probability e^{-4mL},

$$\text{if } \sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} > 3^{-q} h_{f;q} \qquad \text{then } X_t^{f;q+1} \geq \frac{1}{2}\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q}. \tag{5.6}$$

We now proceed to prove (5.4) by induction on q. For $q = 1$, notice that

$$X_t^{f;1} = \sum_{s\leq t}\sum_i f(i)\mathbf{1}_{\{J_s=i\}} = \sum_i f(i)N_t(i),$$
$$V_t^{f;1} = \sum_{s\leq t}\sum_i f(i)Q_s(i).$$

Furthermore, as the definition of $X_t^{f;q}$ also applies to the case $q = 0$, we obtain that

$$X_t^{f;0} = \sum_i f(i) \qquad \text{and so}$$
$$V_t^{f;1} = \sum_{s<t}\sum_i Q_{s+1}(i)f(i) = \sum_{s<t} X_s^{(Q_{s+1}\cdot f);0}.$$

Hence, combining the assumption $V_t^{f;1} \geq 100(3L)^2 m\|f\|_\infty = h_{f;1}$ with statement (5.6) yields that $X_t^{f;1} \geq \frac{1}{2}V_t^{f;1} \geq \frac{1}{3}V_t^{f;1}$, except with probability e^{-4mL}.

It remains to establish the induction step. The induction hypothesis for q states that whenever $V_t^{f;q} \geq h_{f;q}$ we also have $X_t^{f;q} \geq 3^{-q}V_t^{f;q}$ except with probability e^{-4mL}. Therefore,

$$\begin{aligned}\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} &\geq 3^{-q}\sum_{s<t} V_s^{(Q_{s+1}\cdot f);q}\cdot \mathbf{1}_{\{V_s^{(Q_{s+1}\cdot f);q} > h_{(Q_{s+1}\cdot f);q}\}} \\ &\geq 3^{-q}\left(\sum_{s<t} V_s^{(Q_{s+1}\cdot f);q} - t\cdot h_{(Q_{s+1}\cdot f);q}\right) \\ &\geq 3^{-q}\left(V_t^{f;q+1} - t\cdot 100\frac{(3L)^{q+1}}{q!}m\|Q_{s+1}\cdot f\|_\infty\right),\end{aligned} \tag{5.7}$$

where in the last inequality we applied the recursion relation (5.5). Recalling that Q_{s+1} is a strategy, the following holds for all $t \leq n/k$:

$$t\|Q_{s+1}\cdot f\|_\infty \leq t\|Q_{s+1}\|_\infty\|f\|_\infty \leq t\frac{k}{n}\|f\|_\infty \leq \|f\|_\infty.$$

Plugging this into (5.7), we obtain that for all $t \leq n/k$,

$$\begin{aligned}\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} &\geq 3^{-q}\left(V_t^{f;q+1} - 100\frac{(3L)^{q+1}}{q!}m\|f\|_\infty\right) \\ &= 3^{-q}(V_t^{f;q+1} - h_{f;q}).\end{aligned} \tag{5.8}$$

Now, if $V_t^{f;q+1} \geq h_{f;q+1} = 100\frac{(3L)^{q+2}}{(q+1)!}m\|f\|_\infty$, then in particular

$$V_t^{f;q+1} \geq 3 \cdot 100\frac{(3L)^{q+1}}{q!}m\|f\|_\infty = 3h_{f;q} \qquad \text{for all } q \leq L-1.$$

Thus, under this assumption, (5.8) takes the following form:

$$\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} \geq 3^{-q} \cdot \frac{2}{3} V_t^{f;q+1} \geq 3^{-q} \cdot 2h_{f;q}.$$

Since this satisfies the condition of (5.6) (where we actually only needed a lower bound of $3^{-q}h_{f;q}$), we obtain that except with probability e^{-4mL},

$$X_t^{f;q+1} \geq \frac{1}{2}\sum_{s<t} X_s^{(Q_{s+1}\cdot f);q} \geq 3^{-(q+1)} V_t^{f;q+1},$$

completing the induction step.

Summing the error probabilities over the induction steps for every $q < L$ concludes the proof of the lemma. □

It remains to apply the above lemma to deduce the maximal load of $\Omega(\frac{\log n}{\log\log n})$ for $k = \mathrm{polylog}(n)$. Recalling that $m \leq n^{1-\delta}$ for some fixed $\delta > 0$, let $0 < \varepsilon < \delta/2$ and choose the following parameters:

$$q = (1-\varepsilon)\frac{\log(n/m)}{\log k + \log\log(n/m)}, \qquad f = \mathbf{1}, t = n/k.$$

Lemma 5.1 now gives that, either the maximal load exceeds $L = \log(n/m)$, or w.h.p. the following statements holds:

$$\text{If } V_{n/k}^{\mathbf{1};q} \geq 100\frac{(3L)^{q+1}}{q!}m \qquad \text{then } X_{n/k}^{\mathbf{1};q} \geq 3^{-q} V_{n/k}^{\mathbf{1};q}. \tag{5.9}$$

Notice that for the above value of q, we have $3^{-q} = (n/m)^{o(1)}$, and therefore, showing that the condition of (5.9) is satisfied and that $V_{n/k}^{\mathbf{1};q} \geq (n/m)^{\varepsilon/2}$ would immediately imply that the maximal load exceeds q w.h.p.

The following lemma, which provides a lower bound on $V_t^{\mathbf{1};q}$, is thus the final ingredient required for the proof of the theorem:

LEMMA 5.2. *For all t, k and q, all $Q_1, \ldots, Q_t$ and any fixed $\alpha > 0$ we have*

$$V_t^{\mathbf{1};q} \geq \frac{(t-(1+\alpha)kq)^q}{\mathrm{e}^{q/(2\alpha)}n^{q-1}q!}.$$

PROOF. Recall that

$$V_t^{\mathbf{1};q} = \sum_{s_1<\cdots<s_q\le t} \sum_{i=1}^{n} (Q_{s_1}\cdots Q_{s_q})(i)$$

$$= \frac{1}{q!}\sum_{i=1}^{n}\sum_{s_1\le t} Q_{s_1}(i) \sum_{\substack{s_2\le t\\ s_2\neq s_1}} Q_{s_2}(i)\cdots \sum_{\substack{s_q\le t\\ s_q\notin\{s_1,\ldots,s_{q-1}\}}} Q_{s_q}(i).$$

Defining

$$r_i \stackrel{\triangle}{=} \sum_{s\le t} Q_s(i),$$

and recalling that $\|Q_s\|_\infty \le k/n$ for all s, it follows that for all i and $j \ge 1$,

$$\sum_{\substack{s_j\le t\\ s_j\neq\{s_1,\ldots,s_{j-1}\}}} Q_{s_j}(i) \ge r_i - (j-1)k/n.$$

Consequently,

$$\begin{aligned} V_t^{\mathbf{1};q} &\ge \frac{1}{q!}\sum_{i=1}^{n}\prod_{j=1}^{q}\big(r_i-(j-1)k/n\big) \\ &\ge \frac{1}{q!}\sum_{i=1}^{n}\prod_{j=1}^{q}\big(r_i-(j-1)k/n\big)\mathbf{1}_{\{r_i>(1+\alpha)(k(q-1)/n)\}}. \end{aligned} \tag{5.10}$$

Next, notice that for all $1 \le j \le q-1$,

$$\begin{aligned} 1-\frac{j}{(1+\alpha)(q-1)} &\ge \exp\left[-\frac{j}{(1+\alpha)(q-1)}\Big/\left(1-\frac{j}{(1+\alpha)(q-1)}\right)\right] \\ &\ge \exp\left[-\frac{j}{\alpha(q-1)}\right]. \end{aligned}$$

Thus, in case $r_i > (1+\alpha)\frac{k(q-1)}{n}$ we have the following for all $1 \le j \le q$:

$$r_i - \frac{(j-1)k}{n} > r_i\left(1-\frac{j-1}{(1+\alpha)(q-1)}\right) \ge r_i\exp\left[-\frac{j-1}{\alpha(q-1)}\right].$$

Combining this with (5.10), we deduce that

$$\begin{aligned} V_t^{\mathbf{1};q} &\ge \frac{1}{q!}\sum_{i=1}^{n}\prod_{j=1}^{q} r_i\exp\left[-\frac{j-1}{\alpha(q-1)}\right]\mathbf{1}_{\{r_i>(1+\alpha)(k(q-1)/n)\}} \\ &= \frac{1}{q!}\sum_{i=1}^{n}\big(\mathrm{e}^{-1/(2\alpha)} r_i \mathbf{1}_{\{r_i>(1+\alpha)(k(q-1)/n)\}}\big)^q. \end{aligned}$$

Applying Cauchy–Schwarz, we infer that

$$V_t^{\mathbf{1};q} \geq \frac{n}{q!}\left(\frac{\mathrm{e}^{-1/(2\alpha)}\sum_i r_i \mathbf{1}_{\{r_i>(1+\alpha)k(q-1)/n\}}}{n}\right)^q$$
$$= \frac{(\sum_i r_i \mathbf{1}_{\{r_i>(1+\alpha)(k(q-1)/n)\}})^q}{\mathrm{e}^{q/(2\alpha)} n^{q-1} q!}.$$

The proof of the lemma now follows from noticing that

$$\sum_i r_i \mathbf{1}_{\{r_i\leq(1+\alpha)(k(q-1)/n)\}} \leq (1+\alpha)k(q-1) < (1+\alpha)kq,$$

whereas $\sum_i r_i = \sum_{s\leq t}\sum_i Q_s(i) = t$. $\square$

To complete the proof using Lemma 5.2, apply this lemma for $\alpha = 1$, $t = n/k$, and $kq = n^{o(1)}$, giving that

$$V_{n/k}^{1;q} \geq \frac{n}{((\mathrm{e}^{1/2} - o(1))k)^q q!} \geq (2k)^{-q} n/q!,$$

where the last inequality holds for any sufficiently large n. Consequently,

$$\frac{100(3L)^{q+1} m/q!}{V_{n/k}^{\mathbf{1};q}} \leq 100(3L)^{q+1}(2k)^q \frac{m}{n} \leq 100(6kL)^{q+1}\frac{m}{n}.$$

Since our choice of q is such that

$$(6kL)^{q+1} = \mathrm{e}^{(1+o(1))q(\log L+\log k)} = (n/m)^{1-\varepsilon-o(1)},$$

we have that

$$\frac{100(3L)^{q+1} m/q!}{V_{n/k}^{\mathbf{1};q}} \leq (n/m)^{-\varepsilon+o(1)}.$$

This implies both that $V_{n/k}^{\mathbf{1};q} \geq (n/m)^{\varepsilon/2}$ for any large n (recall that $L > q$), and that the condition of (5.9) is satisfied for any large n. Altogether, the maximal load is w.h.p. at least q, concluding the proof of Theorem 2. $\square$

5.2. *A corollary for nonadaptive algorithms.* We end this section with a corollary of Theorem 2 for the case of non-adaptive algorithms, that is, the strategies $Q_1, \ldots, Q_n$ are fixed ahead of time. Namely, we show that for $k = O(n\frac{\log\log n}{\log n})$ the optimal maximal load is w.h.p. $\Theta(\frac{\log n}{\log\log n})$, that is, of the same order as the one for $k = 1$. Theorem 5, whose proof appears in Section 6, includes a different approach that proves this result more directly.

COROLLARY 5.3. *Consider the allocation problem of n balls into n bins, where each ball has k independent uniform choices. If $k \le Cn\frac{\log\log n}{\log n}$, then any nonadaptive algorithm w.h.p. has a maximal-load of at least $\frac{1-o(1)}{C\vee 1}\cdot\frac{\log n}{\log\log n}$. In particular, if $k \le n\frac{\log\log n}{\log n}$ then the load is at least $(1-o(1))\frac{\log n}{\log\log n}$ w.h.p.*

PROOF. Let $Q_1,\dots,Q_n$ be the optimal sequence of strategies for the problem. Using definitions (5.1) and (5.2) with $f\equiv 1$, we have the following for all q:

$$X_t^{\mathbf{1};q} = \sum_i \binom{N_t(i)}{q} = \mathrm{Col}_q(t) \quad \text{and}$$

$$V_t^{\mathbf{1};q} = \mathbb{E}X_t^{\mathbf{1};q}.$$

Fix $0<\varepsilon<\frac{1}{2}$. Applying Lemma 5.2 with $t=n$ and $\alpha=\varepsilon/[2(1-\varepsilon)]$,

$$V_n^{\mathbf{1};q} \ge \frac{(n-(1+\alpha)kq)^q}{\mathrm{e}^{q/(2\alpha)}n^{q-1}q!} = \left(1-\frac{(2-\varepsilon)kq}{2(1-\varepsilon)n}\right)^q\cdot\frac{n}{\mathrm{e}^{(1-\varepsilon)q/\varepsilon}q!}. \tag{5.11}$$

Recalling that $k\le Cn\frac{\log\log n}{\log n}$ for some fixed $C>0$, set

$$q = \frac{1-\varepsilon}{C\vee 1}\cdot\frac{\log n}{\log\log n}. \tag{5.12}$$

This choice has $kq/n\le 1-\varepsilon$ and $q!\le n^{1-\varepsilon+o(1)}$. Combined with (5.11),

$$\begin{aligned}\mathbb{E}\,\mathrm{Col}_q(n) &= V_n^{\mathbf{1};q}\\ &\ge \exp\left(-\frac{(2-\varepsilon)kq^2}{2(1-\varepsilon)n}\Big/\left(1-\frac{(2-\varepsilon)kq}{2(1-\varepsilon)n}\right)\right)\frac{n}{\mathrm{e}^{(1-\varepsilon)q/\varepsilon}q!}\\ &\ge \exp\left(-\frac{2-\varepsilon}{\varepsilon}q\right)\frac{n}{n^{1-\varepsilon+o(1)}} = n^{\varepsilon-o(1)}.\end{aligned} \tag{5.13}$$

To translate the number of q-collisions to the number of bins with load q, consider the case where for some bin j we have $\sum_{s=1}^n Q_s(j)\ge 100\log n$. Proposition 2.1 (applied to the Bernoulli variables $\mathbf{1}_{\{J_s=j\}}$) then implies that $N_n(j)\ge 50\log n$ except with probability $O(n^{-5})$, and in particular the maximal load exceeds q w.h.p. We may therefore assume from this point on that $\sum_{s=1}^n Q_s(j)\le 100\log n$ for all j.

Set $L\stackrel{\triangle}{=}150\log n$. Clearly, upon increasing $Q_s(j)$ for some $1\le s\le n$, the load in bin j will stochastically dominate the original one. Thus, for any integer $r\ge 1$ we may increase $\sum_{s=1}^n Q_s(j)$ to $\frac{2}{3}rL$, and by Proposition 2.1 obtain that $N_n(j)\le rL$ except with probability $O(\exp(-rL/30))$. Defining

$$A_r \stackrel{\triangle}{=} \left\{rL\le \max_{1\le j\le n} N_n(j) < (r+1)L\right\} \qquad (\text{for } r=0,1,\dots),$$

we in particular get $\mathbb{P}(A_r) = O(n\exp(-rL/30))$. However, clearly on this event $\mathrm{Col}_q(n) \le n\binom{(r+1)L}{q}$, and since $n^2\binom{(r+1)L}{q} \le O(\exp(rL/50))$, we have

$$\mathbb{E}[\mathrm{Col}_q(n) \mid \overline{A_0}]\mathbb{P}(\overline{A_0}) \le \sum_{r\ge 1} O\big(\exp(-rL/100)\big) = O(n^{-3/2}) = o(1).$$

Thus, by (5.13), we have $\mathbb{E}[\mathrm{Col}_q(n) \mid A_0] \ge n^{\varepsilon - o(1)}$. Finally, since any given bin can contribute at most $\binom{L}{q} = n^{o(1)}$ collisions to $\mathrm{Col}_q(n)$ given A_0,

$$\mathbb{E}\left[\sum_{j=1}^{n} \mathbf{1}_{\{N_n(j)\ge q\}}\right] \ge \mathbb{E}\left[\mathrm{Col}_q(n)\Big/\binom{L}{q}\,\Big|A_0\right]\mathbb{P}(A_0) = n^{\varepsilon - o(1)}.$$

As demonstrated in the next section (see Lemma 6.2), one can now use the fact that the events $\{N_n(j) \ge q\}$ are negatively correlated to establish concentration for the variable $\sum_{j=1}^{n} \mathbf{1}_{\{N_n(j)\ge q\}}$. Altogether, we deduce that the maximal load w.h.p. exceeds q, as required. □

6. Tight bounds for nonadaptive allocations. In this section, we present the proof of Theorem 5. Throughout the proof we assume, whenever this is needed, that n is sufficiently large. To simplify the presentation, we omit all floor and ceiling signs whenever these are not crucial. We need the following lemma.

LEMMA 6.1. *Let $p_1, p_2, \ldots, p_n$ be reals satisfying $0 \le p_i \le \frac{\log\log n}{\log n}$ for all i, such that $\sum_{i=1}^{n} p_i \ge 1-\varepsilon$, where $\varepsilon = \varepsilon(n) \in [0,1]$. Let $X_1, X_2, \ldots, X_n$ be independent indicator random variables, where $\mathbb{P}(X_i = 1) = p_i$ for all i, and put $X = \sum_{i=1}^{n} X_i$. Then*

$$\mathbb{P}\left(X \ge (1-\varepsilon)\frac{\log n}{\log\log n}\right) \ge \frac{1}{n^{1-\varepsilon}}.$$

PROOF. Without loss of generality, assume that $p_1 \ge p_2 \ge \cdots \ge p_n$. Define a family of k pairwise disjoint blocks $B_1, B_2, \ldots, B_k \subset \{1, 2, \ldots, n\}$, where $k \ge (1-\varepsilon)\frac{\log n}{\log\log n}$ so that for each i, $1 \le i \le k$,

$$\frac{2}{\log n} \le \sum_{j\in B_i} p_j \le \frac{\log\log n}{\log n}.$$

This can be easily done greedily; the first block consists of the indices $1, 2, \ldots, r$ where r is the smallest integer so that $\sum_{j=1}^{r} p_j \ge \frac{2}{\log n}$. Note that it is possible that $r = 1$, and that since the sequence p_j is monotone decreasing, $\sum_{j=1}^{r} p_j \le \frac{\log\log n}{\log n}$. Assuming we have already partitioned the indices $\{1, \ldots, r\}$ into blocks, and assuming we still do not have $(1-\varepsilon)\frac{\log n}{\log\log n}$ blocks, let the next block be $\{r+1, \ldots, s\}$ with s being the smallest integer exceeding r so that $\sum_{j=r+1}^{s} p_j \ge \frac{2}{\log n}$.

Note that if $p_{r+1} \geq \frac{2}{\log n}$ then $s = r+1$, that is, the block consists of a single element, and otherwise $\sum_{j=r+1}^{s} p_j < \frac{4}{\log n} < \frac{\log\log n}{\log n}$. Thus, in any case the sum above is at least $\frac{2}{\log n}$ and at most $\frac{\log\log n}{\log n}$. Since the total sum of the reals p_j is at least $1-\varepsilon$ this process does not terminate before generating $k \geq (1-\varepsilon)\frac{\log n}{\log\log n}$ blocks, as needed.

Fix a family of $k = (1-\varepsilon)\frac{\log n}{\log\log n}$ blocks as above. Note that for each fixed block B_i in the family, the probability that $\sum_{j\in B_i} X_j \geq 1$ is at least

$$\sum_{j\in B_i} p_j - \sum_{j,q\in B_i, j<q} p_j p_q \geq \sum_{j\in B_i} p_j - \frac{1}{2}\Big(\sum_{j\in B_i} p_j\Big)^2 \geq \frac{2}{\log n} - \frac{2}{\log^2 n} > \frac{1}{\log n}.$$

It thus follows that the probability that for each of the k blocks B_i in the family $\sum_{j\in B_i} X_j \geq 1$ is at least $(\frac{1}{\log n})^k = \frac{1}{n^{1-\varepsilon}}$, completing the proof of the lemma. □

PROOF OF THEOREM 5 [Part (i)]. As before, our framework is the relaxed model where there are strategies $Q_1, Q_2, \ldots, Q_n$, where Q_t is the distribution of the bin to be selected for ball number t, satisfying $\|Q_t\|_\infty \leq k/n$. However, since now we consider nonadaptive algorithms, the strategies are no longer random variables, but rather a predetermined sequence. We therefore let $P = (p_{it})$ denote the $n \times n$ matrix of probabilities, where p_{it} is the probability that the ball at time t would be placed in bin i. Clearly,

$$0 \leq p_{it} \leq k/n = \frac{\log\log n}{\log n} \qquad \text{for all } i \text{ and } t$$

and

$$\sum_{1\leq i\leq n} p_{it} = 1 \qquad \text{for all } t.$$

The sum of entries of each column of the n by n matrix p_{it} is 1, and hence the total sum of its entries is n. If it contains a row i so that the sum of entries in this row is at least, say, $\log n$, then the expected number of balls in bin number i by the end of the process is $\sum_{t=1}^{n} p_{it} \geq \log n$. As the variance is

$$\sum_{t+1}^{n} p_{it}(1-p_{it}) \leq \sum_{t=1}^{n} p_{it},$$

it follows by Chebyshev's inequality (or by Hoeffding's inequality) that with high probability the actual number of balls placed in bin number i exceeds $\frac{\log n}{2} > \frac{\log n}{\log\log n}$, showing that in this case the desired result holds.

We thus assume that the sum of entries in each row is at most $\log n$. As the average sum in a row is 1, there is a row whose total sum is at least 1. Omit this row, and note that since its total sum is at most $\log n$, the sum of all remaining

entries of the matrix is still at least $n - \log n$, and hence the average sum of a row in it is at least $\frac{n-\log n}{n-1} > 1 - \frac{\log n}{n}$. Therefore, there is another row of total sum at least this quantity. Omitting this row and proceeding in this manner we can define a set of rows so that the sum in each of them is large. Note that as long as we defined at most $\frac{n}{\log^2 n}$ rows, the total sum of the remaining elements of the matrix is still at least $n - \frac{n}{\log n}$, and hence there is another row of total sum at least $1 - \frac{1}{\log n}$. We have thus shown that there is a set I of $\frac{n}{\log^2 n}$ rows such that

$$\sum_{t=1}^{n} p_{it} \geq 1 - \frac{1}{\log n} \qquad \text{for each } i \in I.$$

For each $i \in I$, let A_i denote the event

$$A_i \stackrel{\triangle}{=} \left(\text{There are at most } \left(\frac{\log n}{\log\log n} - 4 \right) \text{ balls in bin } i \right).$$

Applying Lemma 6.1 with $\varepsilon = \frac{4\log\log n}{\log n}$, we get

$$\mathbb{P}(A_i) \leq 1 - \frac{\log^4 n}{n} \qquad \text{for each } i \in I.$$

We will next show that, as the events A_i are negatively correlated, the probability that all of these events occurs is at most the product of these probabilities (which is negligible).

LEMMA 6.2. *Define the events $\{A_i : i \in I\}$ as above. Then*

$$\mathbb{P}\left(\bigcap_{i \in S} A_i \right) \leq \prod_{i \in S} \mathbb{P}(A_i) \qquad \textit{for any subset } S \subset I.$$

PROOF. The proof proceeds by induction on $|S|$. For the empty set this is trivial, and we will prove that for every set S and $j \in I \setminus S$

$$\mathbb{P}\left(\bigcap_{i \in S} A_i \cap A_j \right) \leq \mathbb{P}\left(\bigcap_{i \in S} A_i \right) \mathbb{P}(A_j).$$

Define the following independent random variables for every time t:

$$\mathbb{P}(B_t = 1) = p_{tj}, \qquad \mathbb{P}(B_t = 0) = 1 - p_{tj},$$
$$\mathbb{P}(H_t = i) = \frac{p_{ti}}{1 - p_{tj}} \qquad \text{for each } i \neq j.$$

We may now define J_t, the position of the ball at time t, as a function of B_t and H_t, such that indeed $\mathbb{P}(J_t = i) = p_{ti}$ for all i:

$$J_t = \begin{cases} j & B_t = 1, \\ H_t & B_t = 0. \end{cases}$$

Crucially, the event A_j depends only on the values of $\{B_t\}$, and is a monotone decreasing in them. Further notice that the function

$$f(b_1,\ldots,b_n) \triangleq \mathbb{P}\Big(\bigcap_{i\in S} A_i \mid B_1 = b_1,\ldots,B_n = b_n\Big)$$

is monotone increasing in the b_i's. Therefore, applying the FKG-inequality (see, e.g., [2], Chapter 6, and also [16], Chapter 2) on $Y = f(B_1,\ldots,B_n)$ and $\mathbf{1}_{A_j}$ gives

$$\mathbb{P}\Big(\bigcap_{i\in S} A_i \cap A_j\Big) = \mathbb{E}[Y\mathbf{1}_{A_j}] \le \mathbb{E}[Y]\mathbb{P}(A_j) = \mathbb{P}\Big(\bigcap_{i\in S} A_i\Big)\mathbb{P}(A_j),$$

as required. □

Altogether, we obtain that the probability that all of the bins with indices in I have at most $(\frac{\log n}{\log\log n} - 4)$ balls is

$$\mathbb{P}\Big(\bigcap_{i\in I} A_i\Big) \le \left(1 - \frac{\log^4 n}{n}\right)^{n/\log^2 n} \le \mathrm{e}^{-\log^2 n}.$$

This completes the proof of part (i). □

PROOF OF THEOREM 5 [Part (ii)]. It is convenient to describe the proof of this part for a slightly different model instead of the one considered in the previous sections. Namely, in the variant model, in every round each bin among the n bins is chosen randomly and independently as one of the options with probability α. By Chernoff's bounds, our results in this model will carry into the original one, since obtaining αn uniform bins is dominated by getting each bin independently with probability $(1+\varepsilon)\alpha$, and dominates a probability of $(1-\varepsilon)\alpha$ for each bin.

For the simplicity of the notations, we will henceforth consider the case $k = n/2$, noting that our proofs hold for $k = \alpha n$ with any $0 < \alpha < 1$ fixed.

As noted in the Introduction, the relaxed model of strategies Q_t such that $\|Q_t\|_\infty \le k/n$ is stronger than the model where there are k uniform options for bins. In fact, the results of this part (an optimal maximal load of order $\sqrt{\log n}$) do not hold for the relaxed model. For instance, if Q_t assigns probability $k/n = \frac{1}{2}$ to $i = t$ and $i = (t+1)$ (with the indices reduced modulo n), the maximum load will be at most 2.

However, it is easy to see that in fact each strategy Q_t is more restricted. Indeed, the total probability that Q_t can assign to any r bins does not exceed $1 - 2^{-r}$, as for each fixed set I of r bins, the probability that none of the members of I is an optional choice for ball number t is 2^{-r}.

We start with the simple proof of the upper bound, obtained by the natural algorithm which places the ball in round t in the first possible bin (among the k given choices) that follows bin number t in the cyclic order of the bins.

LEMMA 6.3. *There exists a nonadaptive strategy ensuring that, w.h.p., the maximum load in the above model is at most* $O(\sqrt{\log n})$.

PROOF. Order the bins cyclically $b_1, b_2, \ldots, b_n, b_{n+1} = b_1$. For each round t, $1 \le t \le n$, place the ball number t in the first possible bin b_i that follows b_t in our cyclic order and is one of the given options for this round. Note, first, that the probability that the ball in round t is placed in a bin whose distance from b_t exceeds $2\log n$, is precisely the probability that none of the $2\log n$ bins following b_t is chosen in round t, which is

$$2^{-2\log n} < n^{-5/4}.$$

Therefore, with high probability, this does not happen for any t. In addition, the probability that a fixed bin b_i gets a load of $\sqrt{\log n}$ from balls placed in the $2\log n - 2\sqrt{\log n}$ rounds $\{i - 2\log n + 1, i - 2\log n + 2, \ldots, i - 2\sqrt{\log n}\}$, does not exceed

$$\binom{2\log n - 2\sqrt{\log n}}{\sqrt{\log n}} \left(\frac{1}{2^{2\sqrt{\log n}}}\right)^{\sqrt{\log n}} \le \frac{1}{n^{2-o(1)}}.$$

Indeed, for each fixed value of $t \in [i - 2\log n + 1, i - 2\sqrt{\log n}]$, if the ball placed in round number t ends in bin number i, then none of the $2\sqrt{\log n}$ bins preceding b_i is chosen as an optional bin for ball number t, and the probability of this event is $2^{-2\sqrt{\log n}}$. There are $\binom{2\log n - 2\sqrt{\log n}}{\sqrt{\log n}}$ possibilities to select $\sqrt{\log n}$ rounds in the set $\{i - 2\log n + 1, i - 2\log n + 2, \ldots, i - 2\sqrt{\log n}\}$, and as the choices of options for each round are independent, the desired estimate follows.

We conclude that with high probability no bin b_i gets any balls from round t with $t \le i - 2\log n$, and no bin b_i gets more than $\sqrt{\log n}$ balls from rounds t with $i - 2\log n < t \le i - 2\sqrt{\log n}$. As b_i can get at most $2\sqrt{\log n}$ balls from all other rounds t, (as there are only $2\sqrt{\log n}$ such rounds), it follows that with high probability the maximum load does not exceed $3\sqrt{\log n}$, completing the proof of the lemma. Note that it is easy to improve the constant factor 3 in the estimate proved here, but we make no attempt to optimize it. □

We proceed with the proof of the lower bound. As in the proof of part (i) of the theorem, let $P = (p_{it})$ be the $n \times n$ matrix of probabilities corresponding to our nonadaptive strategy, where p_{it} is the probability that the ball in round t will be placed in bin number i. Recall that for each fixed round t, the sum of the largest r numbers p_{it} cannot exceed $1 - 2^{-r}$. This fact will be the only property of the distribution p_{it} used in the proof.

Call an entry p_{it} of the matrix P *large* if $p_{it} \ge 2^{-\sqrt{\log n}}$, otherwise, p_{it} is *small*. Call a column t of P *concentrated* if it has at least $\frac{\sqrt{\log n}}{2}$ large elements. We consider two possible cases.

- *Case 1*: There are at least $n/2$ concentrated columns.

In this case, there are at least $\frac{n\sqrt{\log n}}{4}$ large entries in P. If there is a row, say row number i of P, containing at least, say, $2^{2\sqrt{\log n}}$ large entries, then the expected number of balls in the corresponding bin is $\sum_{t=1}^{n} p_{it} > 2^{\sqrt{\log n}}$, and, as the variance of this quantity is smaller than the expectation, it follows that in this case with high probability this bin will have a load that exceeds $\Omega(2^{\sqrt{\log n}}) > \sqrt{\log n}$. We thus assume that no row contains more than $2^{2\sqrt{\log n}}$ large elements. Therefore, there are at least $\frac{n}{2^{2\sqrt{\log n}}} = n^{1-o(1)}$ rows, each containing at least, say, $\frac{\sqrt{\log n}}{8}$ large elements. Indeed, we can select such rows one by one. As long as the number of selected rows does not exceed $\frac{n}{2^{2\sqrt{\log}}}$, the total number of large elements in them is at most n, and hence the remaining rows still contain at least $\frac{n\sqrt{\log n}}{4} - n > \frac{n\sqrt{\log n}}{8}$ large elements, implying that there is still another row containing at least $\frac{\sqrt{\log n}}{8}$ large elements.

Fix a bin corresponding to a row with at least $\frac{\sqrt{\log n}}{8}$ large elements, and fix $\frac{\sqrt{\log n}}{8}$ of them. The probability that all balls corresponding to these large elements will be placed in this bin is at least

$$\left(\frac{1}{2\sqrt{\log n}}\right)^{\sqrt{\log n}/8} = \frac{1}{n^{1/8}}.$$

As there are at least $n^{1-o(1)}$ such bins, and the events of no large load in distinct bins are negatively correlated (see Lemma 6.2), we conclude that the probability that none of these bins has a load at least $\frac{\sqrt{\log n}}{8}$ is at most

$$\left(1 - \frac{1}{n^{1/8}}\right)^{n^{1-o(1)}} = o(1),$$

showing that in this case the maximum load is indeed $\Omega(\sqrt{\log n})$ with high probability.

- *Case 2*: There are less than $n/2$ concentrated columns.

In this case, the sum of all small entries of the matrix P is at least

$$\frac{n}{2 \cdot 2^{0.5\sqrt{\log n}}},$$

since each of the $n/2$ nonconcentrated columns has less than $0.5\sqrt{\log n}$ large elements, and hence in each such column the sum of all small elements is at least $2^{-0.5\sqrt{\log n}}$.

Call a small entry $p = p_{ij}$ of P an entry of *type* r (where $\sqrt{\log n} \le r \le 2\log n$), if $\frac{1}{2^{r+1}} \le p < \frac{1}{2^r}$. Since the sum of all entries of P that are smaller than

$2^{-2\log n} = 1/n^2$ is at most 1, there is a value of r in the above range, so that the sum of all entries of P of type r is at least

$$\frac{n}{4\log n \cdot 2^{0.5\sqrt{\log n}}} > \frac{n}{2^{0.75\sqrt{\log n}}}.$$

Put

$$x \triangleq 2^{0.75\sqrt{\log n}},$$

and note that there are at least $\frac{n2^r}{x}$ entries of type r in P (since otherwise their total sum cannot be at least n/x). We now restrict our attention to these entries.

We can assume that there is no row containing more than $2^{r+1}\log n$ of these entries. Indeed, otherwise the expected number of balls in the corresponding bin is at least $\log n$, the variance is smaller, and hence by Chebyshev with high probability the load in this bin will exceed $\Omega(\log n) > \sqrt{\log n}$. We can now apply again our greedy procedure and conclude that there are at least $\frac{n}{4x\log n} = n^{1-o(1)}$ rows, each containing at least $\frac{2^r}{2x}$ entries of type r; indeed, a set of less than $\frac{n}{4x\log n}$ rows contains a total of at most

$$\frac{n}{4x\log n} 2^{r+1}\log n = \frac{n2^r}{2x}$$

elements of type r, leaving at least $\frac{n2^r}{2x}$ such elements in the remaining rows, and hence ensuring the existence of an additional row with at least $\frac{2^r}{2x}$ such entries.

Fix a bin corresponding to a row with at least $\frac{2^r}{2x}$ entries of type r. The probability that exactly t balls corresponding to these entries will be placed in this bin is at least

$$\binom{\frac{2^r}{2x}}{t}\left(\frac{1}{2^{r+1}}\right)^t\left(1-\frac{1}{2^r}\right)^{2^r/(2x)} \geq \left(\frac{2^r}{2xt}\right)^t\left(\frac{1}{2^{r+1}}\right)^t\left(1-\frac{1}{2x}\right) > \frac{1}{2}(4xt)^{-t}.$$

for $t = \sqrt{\log n}$ the last quantity is at least

$$\tfrac{1}{2}\big(4\cdot 2^{0.75\sqrt{\log n}}\sqrt{\log n}\big)^{-\sqrt{\log n}} = n^{-3/4-o(1)}.$$

This, the fact that there are $n^{1-o(1)}$ such rows, and the negative correlation implies that in this case, too, with high probability there is a bin with load at least $\Omega(\sqrt{\log n})$. This completes the proof of part (ii) of the theorem. □

7. Concluding remarks and open problems.

- We have established a sharp choice-memory tradeoff for achieving a constant maximal load in the balls-and-bins experiment, where there are n balls and n bins, each ball has k uniformly chosen options for bins, and there are m bits of memory available. Namely:

1. If $km = \Omega(n)$ for $k = \Omega(\log n)$ and $m = \Omega(\log n \log\log n)$, then there exists an algorithm that achieves an $O(1)$ maximal load w.h.p.
2. If $km = o(n)$ for $m = \Omega(\log n)$, then any algorithm w.h.p. creates an unbounded maximal load. For this case we provide two lower bounds on the load: $\Omega(\log\log(\frac{n}{km}))$ and $(1+o(1))\frac{\log(n/m)}{\log\log(n/m)+\log k}$.

- In particular, if $m = n^{1-\delta}$ for some $\delta > 0$ fixed and $2 \le k \le \text{polylog}(n)$, we obtain a lower bound of $\Theta(\frac{\log n}{\log\log n})$ on the maximal load. That is, the typical maximal load in *any* algorithm has the same order as the typical maximal load in a *random* allocation of n balls in n bins.
- Given our methods, it seems plausible and interesting to improve the above lower bounds to $(1+o(1))\frac{\log(n/(km))}{\log\log(n/(km))}$, analogous to the load of $(1+o(1))\frac{\log n}{\log\log n}$ in a completely random allocation.
- Note that, when $km = n^{1-\delta}$ for some fixed $\delta > 0$, even the above conjectured lower bound is still a factor of δ away from the upper bound given by a random allocation. It would be interesting to close the gap between these two bounds. Concretely, suppose that $km = \sqrt{n}$; can one outperform the typical maximal load in a random allocation?
- To prove our main results, we study the problem of achieving a perfect allocation (one that avoids collisions, i.e., a matching) of $(1-\delta)n$ balls into n bins. We show that there exist constants $C > c > 0$ such that:

  1. If $km > Cn$ for $k = \Omega(\log n)$ and $m = \Omega(\log n \cdot \log\log n)$, then there exists an algorithm that achieves a perfect allocation w.h.p.
  2. If $km < cn$ for $m = \Omega(\log n)$, then any algorithm creates $\Omega(n)$ collisions w.h.p.

- In light of the above, it would be interesting to show that there exists a critical $c > 0$ such that, say for $k, m \ge \log^2 n$, the following holds: If $km \ge (c+o(1))n$ then there is an algorithm that achieves a perfect allocation w.h.p., whereas if $km \le (c-o(1))n$ then any algorithm has $\Omega(n)$ collisions w.h.p.
- The key to proving the above results is a combination of martingale analysis and a Bernstein–Kolmogorov type large deviation inequality. The latter, Proposition 2.1, relates a sum of a sequence of random variables to the sum of its conditional expectations, and crucially does *not* involve the length of the sequence. We believe that this inequality may have other applications in combinatorics and the analysis of algorithms.
- We also analyzed the case of nonadaptive algorithms, where we showed that for every $k = O(n\frac{\log\log n}{\log n})$, the best possible maximal load w.h.p. is $\Theta(\frac{\log n}{\log\log n})$, that is, the same as in a random allocation. For $k = \alpha n$ with $0 < \alpha < 1$, we proved that the best possible maximal load is $\Theta(\sqrt{\log n})$. Hence, one can ask what the minimal order of k is, where an algorithm can outperform the order of the maximal load in the random allocation.

Acknowledgments. We thank Yossi Azar and Allan Borodin for helpful discussions, as well as Yuval Peres for pointing us to the reference for Theorem 2.2. We also thank Itai Benjamini for proposing the problem of balanced allocations with limited memory.

REFERENCES

[1] AJTAI, M. (2002). Determinism versus nondeterminism for linear time RAMs with memory restrictions. *J. Comput. System Sci.* **65** 2–37. MR1946206

[2] ALON, N. and SPENCER, J. H. (2008). *The Probabilistic Method*, 3rd ed. Wiley, Hoboken, NJ. MR2437651

[3] AZAR, Y., BRODER, A. Z., KARLIN, A. R. and UPFAL, E. (1999). Balanced allocations. *SIAM J. Comput.* **29** 180–200 (electronic). MR1710347

[4] BEAME, P. (1991). A general sequential time–space tradeoff for finding unique elements. *SIAM J. Comput.* **20** 270–277. MR1087749

[5] BEAME, P., JAYRAM, T. S. and SAKS, M. (2001). Time–space tradeoffs for branching programs. *J. Comput. System Sci.* **63** 542–572. MR1894521

[6] BEAME, P., SAKS, M., SUN, X. and VEE, E. (2003). Time–space trade-off lower bounds for randomized computation of decision problems. *J. ACM* **50** 154–195 (electronic). MR2147528

[7] BENJAMINI, I. and MAKARYCHEV, Y. (2009). Balanced allocation: Memory performance trade-offs. Preprint. Available at arXiv:0901.1155v1.

[8] BORODIN, A. and COOK, S. (1982). A time–space tradeoff for sorting on a general sequential model of computation. *SIAM J. Comput.* **11** 287–297. MR652903

[9] BORODIN, A., FICH, F., MEYER AUF DER HEIDE, F., UPFAL, E. and WIGDERSON, A. (1987). A time–space tradeoff for element distinctness. *SIAM J. Comput.* **16** 97–99. MR873252

[10] BURKHOLDER, D. L. (1988). Sharp inequalities for martingales and stochastic integrals. *Astérisque* **157–158** 75–94. MR976214

[11] FELLER, W. (1968). *An Introduction to Probability Theory and Its Applications* **I**, 3rd ed. Wiley, New York. MR0228020

[12] FORTNOW, L. (1997). Nondeterministic polynomial time versus nondeterministic logarithmic space: Time–space tradeoffs for satisfiability. In *Twelfth Annual IEEE Conference on Computational Complexity* (*Ulm*, *1997*) 52–60. IEEE Computer Society, Los Alamitos, CA. MR1758122

[13] FORTNOW, L., LIPTON, R., VAN MELKEBEEK, D. and VIGLAS, A. (2005). Time–space lower bounds for satisfiability. *J. ACM* **52** 835–865 (electronic). MR2179549

[14] FREEDMAN, D. A. (1975). On tail probabilities for martingales. *Ann. Probab.* **3** 100–118. MR0380971

[15] GONNET, G. H. (1981). Expected length of the longest probe sequence in hash code searching. *J. Assoc. Comput. Mach.* **28** 289–304. MR612082

[16] GRIMMETT, G. (1999). *Percolation*, 2nd ed. *Grundlehren der Mathematischen Wissenschaften* [*Fundamental Principles of Mathematical Sciences*] **321**. Springer, Berlin. MR1707339

[17] JOHNSON, N. L. and KOTZ, S. (1977). *Urn Models and Their Application*. Wiley, New York. MR0488211

[18] KARP, R. M., VAZIRANI, U. V. and VAZIRANI, V. V. (1990). An optimal algorithm for online bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing* (*Baltimore*, *MD*, *1990*) 352–358. ACM, New York.

[19] MCDIARMID, C. (1998). Concentration. In *Probabilistic Methods for Algorithmic Discrete Mathematics*. *Algorithms and Combinatorics* **16** 195–248. Springer, Berlin. MR1678578

[20] MITZENMACHER, M., PRABHAKAR, B. and SHAH, D. (2002). Load balancing with memory. In *Proceedings of the* 43*rd Annual IEEE Symposium on Foundations of Computer Science* 799–808. IEEE Computer Society, Los Alamitos, CA.

[21] MITZENMACHER, M., RICHA, A. W. and SITARAMAN, R. (2001). The power of two random choices: A survey of techniques and results. In *Handbook of Randomized Computing* **I**, **II**. *Combinatorial Optimization* **9** 255–312. Kluwer Academic, Dordrecht. MR1966907

[22] MITZENMACHER, M. and UPFAL, E. (2005). *Probability and Computing*: *Randomized Algorithms and Probabilistic Analysis*. Cambridge Univ. Press, Cambridge. MR2144605

[23] STEIGER, W. L. (1969). A best possible Kolmogoroff-type inequality for martingales and a characteristic property. *Ann. Math. Statist.* **40** 764–769. MR0240843

N. ALON
SCHOOL OF MATHEMATICS
TEL AVIV UNIVERSITY
TEL AVIV, 69978
ISRAEL
AND
MICROSOFT-ISRAEL R&D CENTER
HERZELIYA, 46725
ISRAEL
E-MAIL: nogaa@tau.ac.il

O. GUREL-GUREVICH
E. LUBETZKY
MICROSOFT RESEARCH
ONE MICROSOFT WAY
REDMOND, WASHINGTON 98052-6399
USA
E-MAIL: origurel@microsoft.com
eyal@microsoft.com