Serving in the Dark should be done Non-Uniformly

Yossi Azar and Ilan Reuven Cohen [⋆]

Blavatnik School of Computer Science, Tel-Aviv University, Israel.

Abstract. We study the following balls and bins stochastic game between a player and an adversary: there are B bins and a sequence of ball arrival and extraction events. In an arrival event a ball is stored in an empty bin chosen by the adversary and discarded if no bin is empty. In an extraction event, an algorithm selects a bin, clears it, and gains its content. We are interested in analyzing the gain of an algorithm which serves in the dark without any feedback at all, i.e., does not see the sequence, the content of the bins, and even the content of the cleared bins (i.e. an oblivious algorithm). We compare that gain to the gain of an optimal, open eyes, strategy that gets the same online sequence. We name this gain ratio the *"loss of serving in the dark"*.
The randomized algorithm that was previously analyzed is choosing a bin independently and uniformly at random, which resulted in a competitive ratio of about 1.69. We show that although no information is ever provided to the algorithm, using non-uniform probability distribution reduces the competitive ratio. Specifically, we design a 1.55-competitive algorithm and establish a lower bound of 1.5. We also prove a lower bound of 2 against any deterministic algorithm. This matches the performance of the round robin 2-competitive strategy. Finally, we present an application relating to a prompt mechanism for bounded capacity auctions.

1 Introduction

The behavior of an algorithm inherently depends on its input. In some cases the input is only partially known to the algorithm (e.g. online algorithms, distributed algorithms and incentive compatible algorithms) and it may still perform well. In extreme cases the input is virtually unknown to the algorithm. In these cases the algorithm needs to act (almost) independently of the input. Such algorithms are called *oblivious algorithms*. Typically, oblivious algorithms act uniformly at random over their choices. For example, consider a case where there are m weighted balls and n bins. The algorithm needs to assign the balls to the bins as to minimize the maximum load over all bins (where the load of a bin is the sum of weights of balls which are assigned to it). Consider a simple case where $m = n^2$, n of them are of weight 1 and the others are of weight 0. Clearly, the optimal solution is 1. An oblivious algorithm does not know the

[⋆] Supported in part by the Israel Science Foundation (grant No. 1404/10) and by the Israeli Centers of Research Excellence (I-CORE) program (Center No. 4/11).

weights (it only knows n and m). Clearly any deterministic oblivious algorithm may encounter a maximum of load of n. Fortunately, using randomization an algorithm which assigns each ball uniform at random achieves an expected maximum load of $\log n / \log\log n$. In this paper, we consider a problem where the best previous known oblivious algorithm is to select uniformly at random. Interestingly, we show that using a non-uniform distribution improves the performance. This problem is called *serving in the dark* and has an application in prompt mechanism design for packet scheduling.

The Serving in the dark Game. In this game, there is an arbitrary sequence of ball arrival events and ball extraction events. On the arrival of a new ball, the adversary assigns the ball to an unoccupied bin of its choice. The ball is discarded only if all bins are occupied. On an extraction event the algorithm chooses one of the bins, clears it, and gains its contents. Once the sequence ends, all the bins that contain balls are cleared and their content is added to the total gain. The goal of the algorithm is to maximize the number of cleared balls for the sequence. If the algorithm can see the content of the bins, at any extraction step it would choose a bin with a ball, if one exists, thereby maximizing the total gain. This gain is defined as the **optimal gain** (note that in such a case, the adversary's choices of which bin to assign the ball to are irrelevant). We consider an algorithm which *serves in the dark*. Specifically, the algorithm is not aware of the arrival events and of the content of the bins. Moreover, when the algorithm clears a bin, it does not see the bin content. Equivalently, the algorithm does not get any feedback during the sequence (as such, it can also be called an *oblivious algorithm*). We can describe any sequence which contains N extractions as a sequence of N time units $X = \langle X_1, \ldots, X_N \rangle$, where at time j, $X_j \geq 0$ balls arrive and then one extraction event takes place. In this paper we compare the gain achieved by an algorithm that serves in the dark to the **optimal gain** on the worst possible sequence.

The most natural algorithm is the round robin on the bins, which is $(2-1/B)$-competitive. We show in this paper that this is the best possible deterministic algorithm. Hence, in order to improve this bound one needs to use randomization. The most natural randomized algorithm is to choose a bin independently and uniformly at random. For such an algorithm, the choices the adversary makes for the assignment of the balls become irrelevant and the game becomes somewhat degenerate. For the uniform algorithm, the exact competitive ratio for the worst sequence has been determined in [4] to be approximately 1.69.

On one hand, it may seem that the best possible strategy for an algorithm is to choose a bin uniformly at random, since the algorithm does not get any feedback during the sequence. Hence, if some bin is chosen with a smaller probability, then the adversary is more likely to put the next ball in that bin. On the other hand, although no information is provided, the algorithm might want to choose a bin that has not been examined recently. Here we show that by using a **non-uniform** distribution we can substantially improve the competitive ratio to 1.55 and get relatively close to the lower bound of 1.5 that we establish.

Application: prompt mechanism design for packet scheduling. Consider the basic packet scheduling mechanism in which an online sequence of packets with arbitrary private values arrives to a network device that can accommodate up to B packets. The device can transmit one packet in each time step. The goal is to maximize the overall value of the transmitted packets. A trivial greedy mechanism keeps the B packets with the highest values at any moment in time, and transmits the packet with the highest value when possible. This mechanism is optimal, truthful, but not prompt, i.e., the price cannot be determined at the time of transmission (see [7]). A prompt mechanism can be designed by using a *value-oblivious* algorithm. Such algorithms have the property that during transmission no preference is given to a packet with a higher value. We note that value-oblivious algorithms may inspect the values of packets on their arrival. Therefore, one can assume, without loss of generality, that any value-oblivious algorithm keeps the B packets with the highest values at any moment in time. One example of a value-oblivious algorithm is the FIFO algorithm, which transmits the earliest packet in the buffer. This algorithm is known to be $2 - 1/B$-competitive against the absolute optimum [10]. An algorithm which transmits a packet independently and uniformly at random is approximately 1.69-competitive [4]. A natural question is whether one can gain from using a non-uniform distribution. This question can be reduced, by the zero-one principle [5], to the *the Serving in the dark Game* described above.

1.1 Our results

In this paper, we provide a *time-order based* algorithm that uses a non-uniform distribution over the bins. This algorithm is approximately 1.55-competitive for the *serving in the dark game*, which improves the previously known results. Recall that the competitive ratio of a randomized algorithm is the worst ratio over all sequences between the **optimal gain** and the *expected gain* of the algorithm.

Theorem 1. *There exists a randomized algorithm for serving in the dark which is* $(1.55 + o(1))$*-competitive, where* $o(1)$ *is a function of* B*.*

We also show a relatively close lower bound for **any** randomized algorithm for serving in the dark.

Theorem 2. *Any randomized algorithm for serving in the dark is at least* 1.5*-competitive.*

The lower bound for Theorem 2, and all other missing details and proofs are in the appendix. In order to prove Theorem 1, we actually prove a more general theorem, for any *time-order based* algorithm. A *time-order based* algorithm is described by a probability distribution on B ordered bins, where the order is determined by the last time a bin has been cleared. A probability distribution is called monotone non-decreasing, if for any two bins, the most recent bin in the order does not have a higher probability than the least recent bin. We analyze any monotone *time-order based* algorithm as follows:

Theorem 3. *For any* p *a monotone non-decreasing and bounded probability distribution on* $[0, 1]$ *, let* $H(x) = \int_x^1 p(y)dy$*. Let* f *be the solution to the differential* *equation $f'(x) = -H(f(x))$, with $f(0) = 1$. The competitive ratio of a block time-order based algorithm which uses p is* $\max_{x\geq 1}\left\{\frac{x}{x-f(x)+f(x-1)-1}\right\}(1+o(1))$, *where $o(1)$ is a function of B.*

In the above theorem $f(x)$ corresponds to the fraction of balls in the bins starting with B balls followed by xB extractions step with no arrivals.

Application: prompt mechanisms for bounded capacity auctions. We can use the a serving in the dark algorithm to establish a truthful and prompt selection mechanism for bounded capacity auctions. A bounded capacity auction is a single-item periodic auction for bidders that arrive online, in which the number of participating bidders is bounded, e.g., when the auction room has a limited size. We can apply the serving in the dark algorithm for designing a mechanism for packet scheduling. Specifically, we design a truthful prompt mechanism that is approximately $(1.55 + o(1))$-competitive.

1.2 Our approach and techniques

An essential component in our approach is to utilize a deterministic fractional algorithm, which describes in vector form the 'expected' content of the bins, since we do not know how to analyze directly the randomized algorithm. The *deterministic* fractional algorithm will be used as a proxy for the analysis. We analyze the gain of this fractional algorithm compared with the gain of the optimal gain-maximizing strategy. This fractional algorithm is designed in a natural way to correspond to the randomized algorithm and depends on its probability density function.

It is important to note that our analysis is significantly more complicated than the analysis of the uniform distribution case. Specifically, when using the uniform distribution the state of all the bins can be described by a single number, the number of balls in the bins. For arbitrary distributions, presenting the state as a vector is crucial in analyzing the behavior of the algorithm, since different bins are chosen with different probabilities and the probability of choosing one specific bin changes over time. We carefully examine the arrival events and the extraction events for this vector. Our techniques enable us to consider any monotone probability density function for a time-order based algorithm, and characterize up to one parameter the worst input sequence for a fractional algorithm that uses this distribution.

Next, we compare the randomized algorithm with the fractional one. We observe that the gain of this algorithm is not the expected gain of the randomized algorithm, but rather dominates it. Nevertheless, we still establish that it is within a $1 + o(1)$ factor away from the expected gain of the randomized algorithm. For the uniform distribution this was previously done by defining a simple supermartingale on the Markov process of the difference between the number of balls in the fractional algorithm and that in the randomized algorithm. Here, we have to define a chain of separate Markov processes for groups of consecutive bins. In addition, each Markov process in the chain influences the next one. In order to deal with this complex process, we design sequence of hybrid algorithms, each has some randomized part followed by some fractional part. We compare the fractional algorithm to the randomized algorithm by performing a sequence of comparisons between a consecutive hybrid algorithms. Finally by combining the result comparing the fractional algorithm and the optimum algorithm with the result comparing randomized algorithm to the fractional one enables us to prove the upper bound.

1.3 Further related work

In the example that we previously considered there are m weighted balls and n bins. The algorithm needs to assign the balls to the bins to minimize the maximum load. Sanders [17] considered the case where the size of the balls are unknown to the algorithm. He analyzed an oblivious algorithm which assigns each ball uniformly at random to a bin and proved that the worst ratio of the maximum load to the optimal maximum load is achieved when a subset of balls have equal size and the rest are 0. Obliviously, this ratio is bounded by $\log n / \log \log n$. As mentioned before any deterministic oblivious algorithm will achieve a ratio of at least n. Another version of the balls and bins problem is assigning B balls to B bins, where the goal is to maximize the number of non empty bins, where the bins may be permuted by an adversary. A simple result states that if the balls are placed independently and uniformly at random in the bins, then the expected fraction of full bins is $1 - 1/e$. If this procedure could have been performed under light i.e. the permutation at each step was known, then one could deterministically place each ball in a different bin, and hence the fraction of full bins would have been 1.

There are other randomized balls and bins stochastic processes that have been analyzed using various techniques such as martingales and Azuma's inequality. We refer the reader to the papers [9,11,12,2,1,13,14,8] and to the references therein for a more comprehensive review of the literature.

Another example of an algorithm that behaves oblivious to the input is an algorithm for scheduling jobs with release times on identical machines in order to minimize the weighted completion time, introduced by Schulz and Skutella [18]. Their 2-competitive algorithm assigns each job uniformly to a random machine, independently of the assignment of other jobs, while the order of processing the jobs on each machine depends on the input. For the flow time Chekuri et al. [6] showed a constant competitive ratio using the same random oblivious dispatching with extra resources.

Another problem that can be viewed as serving requests independently of the input is the well studied oblivious routing [16,15,3]. Here, a graph is given together with a set of requests to connect pairs of vertices with arbitrary demands. At the preprocessing stage, the graph is given without the requested pairs. The oblivious routing algorithm determines a route (a flow) between any two vertices, independently of their demands and the existence of other pairs. For each request pair, the service is performed using the predetermined flow for this pair scaled by their demand. This achieves logarithmic approximation with respect to the optimal solution for the specific pairs and demands.

2 The Model

Given B bins, consider an arbitrary sequence of arrival events and extraction events.

– Arrival event: a new ball is stored in an unoccupied bin determined by the adversary. If all the bins are occupied, then the ball is discarded.
– Extraction event: the algorithm chooses one of the bins, clears it, and gains its content.

The goal of the algorithm is to maximize the number of extracted balls for the sequence. We assume that all the balls that remain in the bins at the end of the sequence are extracted (this is not required if the optimal gain is large enough).

We consider an algorithm which serves in the dark, i.e., without any input during the whole process (except for the value of B). The algorithm can be viewed as a probability distribution over all infinite sequences of numbers in the set $\{1, \ldots, B\}$. Note that the input sequence is arbitrary and the algorithm does not know the sequence or does not know when the sequence ends. We assume that the adversary knows the algorithm, and that at any moment of time it sees the contents of all bins (even if the algorithm is randomized).

We can describe any sequence which contains N extractions as a sequence of N time steps $X = \langle X_1, \ldots, X_N \rangle$, where at time step j, $X_j \geq 0$ balls arrive and then one extraction event takes place. For a given algorithm ALG, we set $G_i = 1$ if a ball is extracted in extraction step i, and $G_i = 0$ otherwise. Let L_i be the number of balls in the bins before extraction step i: $L_i = \min\{L_{i-1} + X_i - G_{i-1}, B\}$. Denote by O_i the number of overflown (discarded) balls at arrival time i. We have $O_i = \max\{L_{i-1} + X_i - G_{i-1} - B, 0\}$. Using these notations, let $G(X)$ bet the total gain of ALG on a sequence X. By definition, $G(X) = \sum_{i=1}^{N} G_i + L_{N+1} = \sum_{i=1}^{N} X_i - \sum_{i=1}^{N} O_i$. The gain of a randomized algorithm is its expected gain over all algorithm's coins tosses.

The open eye optimal gain: We compare the gain achieved by an algorithm that serves in the dark with the optimal gain on the worst possible sequence. The optimal algorithm can see the contents of the bins at any extraction step and would always choose a bin with a ball if one exists. By that it would maximize the total gain (defined as **optimal gain**). Since the optimal algorithm (called OPT) sees the content of the bins, the choices of the adversary are irrelevant. The gain of OPT in each step i is $G_i^{\mathrm{OPT}} = \min\{L_i^{\mathrm{OPT}}, 1\}$. We use the standard measure to compare a general algorithm with the optimal one (denoted as ϱ): $\rho(X) = \frac{G^{\mathrm{OPT}}(X)}{G(X)}$ and $\varrho = \max_X \rho(X)$.

3 Deterministic Serving Algorithms

One simple deterministic serving algorithm is to perform a round robin over the bins, i.e., on an extraction event the algorithm chooses the least recent bin that it had cleared.

Theorem 4. *The round robin serving algorithm is* $(2 - 1/B)$*-competitive.*

The proof is a simpler version of the proof given for FIFO packet scheduling [10]. Next, we give a bound for the competitiveness of any deterministic serving algorithm.

Theorem 5. *Any deterministic serving algorithm is at least* $(2-1/B)$*-competitive.*

4 Randomized Algorithms and their Analysis

In this section, we design and analyze randomized serving in the dark algorithms.

4.1 Time-Order Based Randomized Algorithms

Let $p : [0, 1] \to \mathbb{R}$ be a monotone non-decreasing probability density function (i.e., $\int_0^1 p(x)dx = 1$). We define a *time-order based* algorithm, denoted by TOB_p, as follows:

```
On each extraction event:
  – Order the bins according to their last extraction step (latest is first).
  – Clear one bin, where the probability to clear the $j$'th ordered bin is $\int_{(j-1)/B}^{j/B} p(x)dx$.
```

Algorithm 1: Time-Order Based Algorithm TOB_p

The algorithm may be described as follows: the bins are ordered in a line of length B. At each step a position in the line is chosen with a fixed monotone non-decreasing probability distribution function p on the positions. The ball (if exists) is extracted from the corresponding bin and then the bin is moved to the beginning of the line.

4.2 Grouping bins together

In order to use a concentration result for the bins, we generalize the algorithm so that instead of B ordered bins, we keep $b_1, \ldots, b_{B/K}$ ordered blocks of volume K (i.e., with K bins each), for some constant $K \geq 1$. We impose no internal order inside a block of bins. The algorithm, denoted as TOB_p^K, uses a data structure of list of B/K blocks, where each block contains K bin indices. On an extraction event, it chooses a block r with probability $q^r = \int_{K(r-1)/B}^{Kr/B} p(x)dx$. Afterwards, it chooses one of the bins in the block uniformly at random and clear it. Finally, for each block $r' < r$ a bin is chosen uniformly at random from it and associate it to the next ordered block, where the extracted bin is associated to the first block in the order.

```
Algorithm $\mathrm{TOB}_p^K$ on extraction event $i$:

  – Choose block $c_i$ with probability $Pr[c_i = r] = q^r = \int_{K(r-1)/B}^{Kr/B} p(x)dx$.
  – Choose a bin $j_r \in b_r$ from each block $r \leq c_i$ uniformly at random.
  – Clear bin $j_{c_i}$ (from block $c_i$).
  – Associate bin $j_r$ with block $b_{r+1}$ (for $r < c_i$), associate bin $j_{c_i}$ with block $b_1$.
```

Algorithm 2: The Block Time-Order Based Algorithm - TOB_p^K

Note that the block time-order based algorithm with $K = 1$ is exactly the time-order based algorithm introduced above. We introduce the following notation with respect to the K-block time-order based algorithm with monotone distribution p ,called TOB_p^K, (we omit K and p if they are clear from the context). Let c_i be the block chosen in step i. Let E_i^r be the indicator of whether in extraction step i a ball is extracted from block r, i.e., $E_i^r = 1$ if $r \leq c_i$ and j_r contains a ball, 0 otherwise. The gain in step i is $G_i = E_i^{c_i}$. Clearly, the gain is equal to 1 if we extracted a ball from the chosen block. Let L_i^r be the number of balls in the r'th ordered block before extraction step i. By the definition of the algorithm, the load of block r after the i'th extraction is $L_i^r + E_i^{r-1} - E_i^r$ if $r \leq c_i$, otherwise it remains L_i^r.

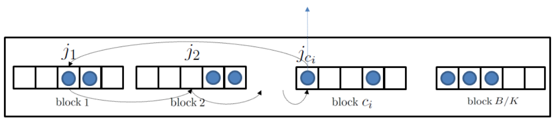

Fig. 1. The algorithm's selection in some step i: the selected block is c_i, j_{c_i} contains a ball therefore $G_i = E_i^{c_i} = 1$. Note also that the algorithm choses a bin from each block before c_i and associates this bin with the next block, and that the algorithm associates the extracted bin j_{c_i} with the first block. Specifically, in the above example, $E_i^1 = 1, E_i^2 = 0$, therefore, the load in the first block decreased by one and the load in the second block increased by one.

Next, let us consider arrival events. Since the algorithm uses a fixed monotone non-decreasing distribution over the ordered blocks, it is easy to determine the optimal strategy of the adversary.

Observation 6 *For the block time-order based algorithm, on an arrival event the adversary assigns a ball in the block with the smallest index that has an empty bin.*

By the above observation, on an arrival event the number of balls in the minimum index block block whose load is smaller than K increases by one. Note that for a given sequence X, the load in each block is a random variable. Since a new ball is stored in the first vacant bin, the block index of this new ball is also a random variable, which makes the analysis of the algorithm complicated. In order to circumvent this difficulty, we next introduce a deterministic fractional algorithm that is close to the randomized one.

4.3 Fractional deterministic algorithms

We define a deterministic algorithm that 'behaves like' the expectation of the TOB_p^K algorithm. Given an input sequence X, the gain and the current loads of the blocks in TOB_p^K are (integer) random variables, since there is randomization in the extraction events. Alternatively, we define FRC_p^K algorithm as a deterministic fractional algorithm, where a fractional of a ball is the deterministically extracted. In each step, the fraction of the balls that is extracted in FRC_p^K corresponds to the the probability that a ball is extracted in TOB_p^K given the current state. Specifically, the load of a block after an extraction event is defined as (we omit FRC, K, p, X if those are clear from the context)

$$L_{i-1}^r + \left(E_i^{r-1} - E_i^r\right) \sum_{j=r}^{B/K} q^j, \tag{1}$$

where $E_i^r = L_i^r / K$. The gain in each step is $G_i = \sum_{r=1}^{B/K} q^r E_i^r$. The arrival of balls is defined as for the randomized algorithm. Note that since the load is fractional, a ball can be split into parts lying in several different blocks.

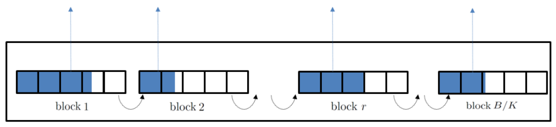

Fig. 2. The fractional block time-order based algorithm, FRC in which L_i^j, E_i^j, G_i are fractional numbers. In the example above $E_i^r = 3/5$.

4.4 Analyzing the fractional algorithm versus the optimal algorithm

The analysis consists of two parts. In the first part we characterize the worst sequence for any distribution p. In the second part we analyze the worst gain ratio of that sequence. By combining the two parts we bound the maximum gain ratio using p. The proof is in the appendix.

Theorem 7. *Given an arbitrary monotone non-decreasing and bounded probability density function p, let $H^p(x) = \int_x^1 p(y)dy$. Let f be a function that satisfies $f(0) = 1$ and $f'(x) = -H^p(f(x))$. The competitive ratio ϱ of the fractional algorithm that uses the function p:* $\varrho^{\mathrm{FRC}} \leq \max_{x \geq 1} \left\{ \frac{x}{x - f(x) + f(x-1) - 1} \right\} (1 + o(1))$.

4.5 Analysis of the randomized algorithm versus the fractional one

In order to compare the fractional algorithm with the randomized one, it is sufficient to analyze input instances in which the fractional algorithm does not overflow. The reason is that removing balls which overflow in the fractional algorithm from the sequence, does not decrease the gain of the fractional algorithm and does not increase the gain of the randomized algorithm. We prove that with high probability a randomized algorithm with slightly larger volume does not overflow on such sequences. First, we compare a single fractional block to a single randomized block:

We define the **extraction probability** of a block with index i as the probability that a block with index at least i will be chosen. We define the **input sequence** of a block as the sum of: (A) the volume overflown from the previous block and (B) the volume extracted from the previous block that was not added to the gain. Additionally, we define the **output sequence** of a block is its extracted volume plus its overflown volume. Note that, the load of a block depends on the block's input sequence and on its extraction probability.

We prove that any input sequence that does not overflow a fractional block, does not overflow a 'slightly larger' randomized block with high probability. A slightly larger means that we increase the randomized block size as well as increase its extraction probability . We prove that this implies that their output sequences are close for any input sequence. Finally, we introduce a hybrid algorithm HYB_m, in HYB_m the first m blocks are randomized and the rest are fractional. Note that in the HYB_m algorithm the input sequence for the block $m+1$ is a random sequence. We compare a HYB_m algorithm to a HYB_{m+1} algorithm by replacing block $m+1$ (a fractional block) with a randomized block. Specifically, using coupling on the randomized choices in the first m blocks we get that the input sequences for the block $m+1$ are the same. Next, we compare the output sequence of the block $m+1$ in HYB_{m+1} with the deterministic output (after the coupling) of block $m+1$ in HYB_m. Specifically, given a sequence and a coupling for which HYB_m does not overflow then HYB_{m+1} with a slightly larger fractional block does not overflow with high probability. By applying this iteratively for $m = 0$ to B/K, we prove that with high probability the randomized algorithm will not overflow and deduce the following theorem:

Theorem 8. *For any fractional block algorithm* FRC *there exists a time order base* TOB *algorithm such that* $G^{\mathrm{TOB}}(X) \geq G^{\mathrm{FRC}}(X)(1 - o(1))$*, for any input sequence* X*.*

Single fractional block versus randomized block Recall that for a fractional or a randomized block, the load in each step depends only on its **input sequence**, and its **extraction probability** as defined above. First, we bound (with high probability) the difference in the load between a fractional block and a randomized block for sequences where the fractional block does not overflow. Let $\epsilon_{N,\Delta K} = N^3 \exp\left(-(\Delta K)^2/8N\right)$ (we omit $N, \Delta K$).

Lemma 1. *Let* $\mathrm{B_{FRC}}$ *be a fractional block of size* K *and extraction probability* $Q^{\mathrm{B_{FRC}}}$ *and* $\mathrm{B_{TOB}}$ *be a randomized block of size* $K+\Delta K$ *and extraction probability* $Q^{\mathrm{B_{TOB}}} = Q^{\mathrm{B_{FRC}}}\frac{K+\Delta K}{K}$. *Then for any input sequence* X *that* $\mathrm{B_{FRC}}$ *does not overflow,*
$\Pr(\exists i \le N : |L_i^{\mathrm{B_{FRC}}}(X) - L_i^{\mathrm{B_{TOB}}}(X)| \ge \Delta K) \le N^3 \exp\left(-(\Delta K)^2/8N\right) = \epsilon.$

Next, we examine the output sequence Y of the FRC block compared to output sequence of the TOB block for any input sequence X. The output sequence is defined as the extracted volume plus the overflow volume, i.e., $Y_i = E_i + O_i$.

Lemma 2. *Let* $\mathrm{B_{FRC}}$ *be a fractional block of size* K *and extraction probability* $Q^{\mathrm{B_{FRC}}}$, *and let* TOB *be a randomized block of size* $K + \Delta K$ *and extraction probability* $Q^{\mathrm{B_{TOB}}} = Q^{\mathrm{B_{FRC}}}\frac{K+\Delta K}{K}$. *For any input sequence* X *we have with probability of at least* $1-\epsilon$ *that* $-\Delta K \le \sum_{j=1}^{i}\left(Y_j^{\mathrm{B_{TOB}}}(X) - Y_j^{\mathrm{B_{FRC}}}(X)\right) \le 3\Delta K$.

The hybrid algorithm We define the hybrid algorithm HYB_m in which the first m blocks are randomized and the rest are fractional. On extraction step i a block c_i is chosen. A randomized block r ($r \le m$) will extract from one of its bins if $r \le c_i$. The extraction from the fractional block is done as in the fractional algorithm independent of the choice c_i. Note that HYB_0 is a fractional algorithm and that $\mathrm{HYB}_{B/K}$ is a randomized algorithm. We design HYB_{m+1} such that all the blocks except block $m+1$ and block B/K are with the same size and extraction probability as in HYB_m. In HYB_{m+1} we set block $m+1$ to be of size $K + \Delta K$ and extraction probability $Q \cdot (K + \Delta K)/K$, where K and Q are the size and extraction probability of block $m+1$ in HYB_m. In addition, we set the last block of HYB_{m+1} to be of size $\tilde{K} + 4\Delta K$ and set its extraction probability to $\tilde{Q} \cdot (\tilde{K} + 4\Delta K)/\tilde{K}$, where $\tilde{K}$ and $\tilde{Q}$ are the size and the extraction probability of the last block in HYB_m. Denote X^m as the (random) input sequence to the block m. The following observation follows immediately from the above construction of HYB_{m+1}.

Observation 9 *For any input sequence* X^m *such that* $\mathrm{HYB}_{m-1}(X^m)$ *does not overflow then* $\mathrm{HYB}_m(X^m)$ *has at least* $4\Delta K$ *vacant volume in each step.*

Lemma 3. *If a sequence* X *does not overflow* HYB_{m-1} *with probability of at least* $(1-\epsilon)^{m-1}$ *then* X *does not overflow* HYB_m *with probability of at least* $(1-\epsilon)^m$.

By applying Lemma 3 B/K times we obtain the following

Corollary 1. *If a sequence* X *does not overflow* HYB_0 *then* X *does not overflow* $\mathrm{HYB}_{B/K}$ *with probability of at least* $(1-\epsilon)^{B/K}$.

Putting everything together The summary of the proof of Theorem 8 is in the appendix. By combining Theorem 7 and Theorem 8, we conclude that $\rho^{\mathrm{TOB}} \le \max_{x\ge 1}\left\{\frac{x}{x-f(x)-1+f(x-1)}\right\}(1+o(1))$, which completes the proof for Theorem 3. The specific distribution function to prove Theorem 1 is in the appendix.

References

1. Noga Alon and Joel H. Spencer. *The Probabilistic Method.* Wiley, New York, second edition, 2000.
2. Yossi Azar, Andrei Z. Broder, Anna R. Karlin, and Eli Upfal. Balanced allocations. *SIAM J. Comput.*, 29(1):180–200, 1999.
3. Yossi Azar, Edith Cohen, Amos Fiat, Haim Kaplan, and Harald Räcke. Optimal oblivious routing in polynomial time. *J. Comput. Syst. Sci.*, 69(3):383–394, 2004.
4. Yossi Azar, Ilan Reuven Cohen, and Iftah Gamzu. The loss of serving in the dark. In *Proceedings 45th Annual ACM Symposium on Theory of Computing*, pages 951–960, 2013.
5. Yossi Azar and Yossi Richter. The zero-one principle for switching networks. In *Proceedings 36th Annual ACM Symposium on Theory of Computing*, pages 64–71, 2004.
6. Chandra Chekuri, Ashish Goel, Sanjeev Khanna, and Amit Kumar. Multi-processor scheduling to minimize flow time with epsilon resource augmentation. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 363–372, 2004.
7. Richard Cole, Shahar Dobzinski, and Lisa Fleischer. Prompt mechanisms for on-line auctions. In *Proceedings 1st International Symposium on Algorithmic Game Theory*, pages 170–181, 2008.
8. Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 2009.
9. Norman L. Johnson and Samuel Kotz. *Urn Models and Their Applications.* John Wiley & Sons, 1977.
10. Alexander Kesselman, Zvi Lotker, Yishay Mansour, Boaz Patt-Shamir, Baruch Schieber, and Maxim Sviridenko. Buffer overflow management in qos switches. *SIAM J. Comput.*, 33(3):563–583, 2004.
11. Valentin F. Kolchin, Boris A. Sevastyanov, and Vladimir P. Chistyakov. *Random Allocations.* John Wiley & Sons, 1978.
12. Colin McDiarmid. Concentration. In *Probabilistic Methods for Algorithmic Discrete Mathematics.* Springer, 1998.
13. Michael Mitzenmacher, Andréa W. Richa, and Ramesh Sitaraman. The power of two random choices: A survey of techniques and results. In *Handbook of Randomized Computing.* Springer.
14. Michael Mitzenmacher and Eli Upfal. *Probability and computing - randomized algorithms and probabilistic analysis.* Cambridge University Press, 2005.
15. Harald Räcke. Minimizing congestion in general networks. In *43rd Symposium on Foundations of Computer Science*, pages 43–52. IEEE Computer Society, 2002.
16. Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings 40th Annual ACM Symposium on Theory of Computing*, pages 255–264, 2008.
17. Peter Sanders. On the competitive analysis of randomized static load balancing. *Proceedings of the first Workshop on Randomized Parallel Algorithms, RANDOM*, 1996.
18. Andreas S. Schulz and Martin Skutella. Scheduling unrelated machines by randomized rounding. *SIAM J. Discrete Math.*, 15(4):450–469, 2002.

Appendix

A Proof of Theorem 7

As mentioned, the analysis consists of two parts. In the first part we characterize the worst sequence for any distribution p. In the second part we analyze the worst gain ratio of that sequence. The proofs are in Appendix B

Characterization of the worst gain ratio sequence In the following subsection, we characterize the worst gain ratio sequence for the fractional algorithm FRC. Although the worst gain ratio sequence depends on p, we found some properties that are independent of p for such sequences which allow us to characterize the sequences in terms of only one parameter, namely, the number of steps between two consecutive overflow steps. The characterization has three parts. First, we show that it is sufficient to consider *compact* sequences, defined by the property that the optimal algorithm clears a ball in each step and does not have an overflow. Second, we prove that in the steps where the fractional algorithm overflows, the optimal algorithm bins are full. Finally, we describe exactly the sequence between two overflow steps in terms of the number of steps between them.

Compact sequences of balls We will prove that it is sufficient to consider *compact* sequences of balls, for which the optimal strategy has no overflow, nor underflow. An *overflow* is a situation in which OPT cannot store all arriving balls in the bins and therefore has to discard some of them, while an *underflow* is an extraction step in which there are no balls in the OPT's bins to be cleared.

Lemma 4. *Given any sequence of balls $X = \langle X_1, \ldots, X_N \rangle$, there is a compact sequence of balls $X' = \langle X'_1, \ldots, X'_{N'} \rangle$ for which the optimal strategy does not have an overflow or an a underflow, and for any* TOB *it holds that $\rho(X') \geq \rho(X)$.*

Overflow steps Next, we prove that it is sufficient to consider sequences where in the steps where the fractional algorithm overflows, the optimal algorithm bins are full.

Lemma 5. *Given a compact sequence of balls X with a maximum gain ratio ρ, we can assume that if $L_i(X) = B$ then $L_i^{\mathrm{OPT}}(X) = B$.*

Corollary 2. *Given a compact sequence of balls X with a maximum gain ratio $\rho(X)$, we can assume for all i that if $L_i^{\mathrm{OPT}}(X) < B$, then $L_i(X) < B$ and thus $O_i(X) = 0$.*

Let $T = \{T_1, \ldots, T_k\}$ be the steps at which the optimal load is full. From Corollary 2 it follows that these are the only possivle overflow steps in the sequence. Note that the number of balls that arrive between two overflow steps is exactly the number of steps between them.

The sequence between two consecutive overflows We next characterize the steps between two consecutive overflows (called inner steps). Obviously, if we change the order of the balls arrival between T_t and T_{t+1} while keeping the sequence compact, and not decreasing the overflow at step T_{t+1}, then the gain ratio between OPT and FRC may only increase.

The main lemma asserts that if we postpone a ball from one step to the next, then the load of the fractional algorithm can only increase. Intuitively, the fractional algorithm has less opportunities to extract this volume, therefore it retains more volume. In fact, we prove that the load will be higher for any prefix of blocks.

Lemma 6. *Let X, X' be input sequences s.t. $X_\ell \geq v$, $X'_\ell = X_\ell - v$, $X'_{\ell+1} = X_{\ell+1} + v$, $X'_i = X_i$ for $i \neq \ell, \ell+1$. Then, for any k and any $i \geq \ell+1$, we have*

$$\sum_{r=1}^{k} L_i^r(X') \geq \sum_{r=1}^{k} L_i^r(X) \tag{2}$$

Since there is no overflow in an inner step, by Lemma 6 it follows that delaying a ball may only increase the total overflow in FRC. Therefore, delaying a ball in an inner step without decreasing the optimal gain (i.e., without causing an underflow in OPT) can only increase the gain ratio. We may delay a ball in an inner step i if $X_i > 0$ and $L_i^{\mathrm{OPT}} = X_i + L_{i-1}^{\mathrm{OPT}} - 1 \geq 2$ (recall that there is no underflow if $L_j^{\mathrm{OPT}} \geq 1$ for any j). Therefore,

Corollary 3. *Given a compact sequence of balls X with a maximum gain ratio $\rho(X)$, and T the set of overflow steps for X, we may assume that for any inner step $i \notin T$ we have*

$$X_i = \begin{cases} 0, & \text{if } L_{i-1}^{\mathrm{OPT}} \geq 2, \\ 1, & \text{if } L_{i-1}^{\mathrm{OPT}} = 1. \end{cases}$$

Corollary 3 allow us to characterize the maximum gain ratio sequence given the overflow steps T.

Lemma 7. *For a maximum gain ratio sequence X, where T is the set of overflow steps, we have*

- $\langle X_1, \ldots, X_{T_1} \rangle = \langle 1, \ldots, 1, B \rangle$.
- *If* $T_{t+1} - T_t = d \leq B$, *then* $\langle X_{T_t+1}, \ldots, X_{T_{t+1}} \rangle = \langle 0, \ldots, 0, d \rangle$.
- *If* $T_{t+1} - T_t = d > B$, *then* $\langle X_{T_t+1}, \ldots, X_{T_{t+1}} \rangle = \langle 0, \ldots, 0, 1, \ldots, 1, B \rangle$, *where the number of* 0*'s is* $B - 1$.

Analysis of the worst gain ratio of the fractional algorithm We bound the worst gain ratio using the characterization of the worst gain ratio sequence. Let $V(d)$ be the load in FRC after d steps, assuming that FRC starts with a full load (i.e., a ball in each bin), and in the subsequent d steps no other balls arrive.

Lemma 8. *The competitive ratio of the fractional algorithm* FRC *is* $\rho \leq \max_{d \geq B} \frac{d}{d - V(d) - B + V(d-B)}$.

Analysis of the worst gain ratio for bounded probability functions We conclude the bounding of the fractional algorithm by analyzing $V(d)$ for bounded probability functions. Let H be the cumulative distribution function of p. Define f as $f(0) = 1$, $f'(x) = -H^p(f(x))$, let $p^{\text{fac}} = \max_x p(x + K/B)/p(x)$. We claim that $V(d) \approx B \cdot f(d/B)$. Since, we observe that after d steps most of the volume $V(d)$ is at blocks with higher indices. So the rate of change, i.e., the extracted volume, is approximately $-H(V(d)/B)$. Formally:

Lemma 9. *Let f be the solution for the differential equation $f'(x) = -H^p(f(x))$, with $f(0) = 1$. The load in FRC_p^K after d steps, starting with a full load, $V(d)$, satisfies $B \cdot f(d/B) \le V(d) \le B \cdot f(d/B) + d(p^{\text{fac}} - 1)$.*

For any p we can compute f (analytically or numerically) and bound the competitive ratio (under the assumption $p^{\text{fac}} - 1 = o(1)$):
$\rho^{\text{FRC}} \le \max_{x \ge 1} \left\{ \frac{x}{x - f(x) - 1 + f(x-1)} \right\} (1 + o(1))$.
This concludes the proof of Theorem 7.

B Omitted Proofs

Proof of Lemma 4. First, if the sequence X has on overflow at step j, we can omit the overflown balls from that step. The optimal gain and the TOB gain do not change, since the optimal load is never higher than the TOB load. Second, if the sequence X has an underflow at step j, then we can omit that step. The optimal gain does not change, while the TOB gain does not increase since p is monotone. We can repeat this procedure, and eventually we get a sequence without any underflow or overflow.

■

Proof of Lemma 5. Consider a sequence where there exists a step such that FRC is fully loaded at step i, but OPT is not. As a consequence, we may modify the sequence by shifting balls arrival from the remainder of the sequence to this step. Since OPT was not full, the ball sequence stays compact. Furthermore, the gain of OPT does not change, while the gain of FRC cannot increase since the shifted balls overflown, and hence, the gain ratio may only increase. Note that, there must be other balls in the rest of the sequence, since otherwise we may increase the gain ratio by adding an additional ball to step i. Notice that this modification does not change the gain of FRC, and increases the gain of OPT.
■

Lemma 10. *For any $\beta_1 \ge \beta_2 \ge \cdots \ge \beta_k \ge 0$ and for any $\alpha_1, \ldots \alpha_k$ s.t for all $j \le k$, $\sum_{r=1}^{j} \alpha_r \ge 0$, we have $\sum_{i=1}^{k} \beta_i \alpha_i \ge 0$.*

Proof. The proof follows since $\sum_{i=1}^{k} \beta_i \alpha_i$ is a positive linear combination of $\sum_{r=1}^{j} \alpha_r$.

Proof of Lemma 6. We separate the arrival at step $\ell+1$ of $X'_{\ell+1} = v + X_{\ell+1}$ into the arrival of v followed by the arrival of $X_{\ell+1}$. We first prove that after the arrival of v, the claimed inequalities hold for any k. If for the input X, v was added to block r (we can assume w.l.o.g. that v is small enough for one block), then after the extraction step, the total difference in the loads is $v'(\leq v)$ and is in the blocks r and $r+1$. If for the input X' the v volume is added to blocks preceding r then obviously all the relations 2 hold, otherwise there is a vacant volume at blocks r and $r+1$, so the volume fills these gaps before filling any other higher-index blocks. Therefore, relations 2 hold.

Next, we prove that if the inequalities hold at some moment of time, they will continue to hold after an arrival event and after an extraction event hence, by induction the inequalities hold for any $i \geq \ell+1$, since the two sequences have precisely the same arrival and extraction events (after the arrival of v). We first consider an extraction event, by summing relations (1). The load after an extraction event at the first k blocks is

$$\sum_{r=1}^{k-1} L^r \left(1 - \frac{q^r}{K}\right) + L^k \left(1 - \sum_{r \geq k} \frac{q^r}{K}\right)$$

For any k, let $\alpha_r = L^r(X') - L^r(X)$, $\beta_r = 1 - \frac{q^r}{K}$, $\beta_k = 1 - \sum_{r \geq k} \frac{q^r}{K}$. The induction assumption is equivalent to: for any j, $\sum_{i=1}^{j} \alpha_i \geq 0$. Moreover, since q is monotone, for any r it holds that $\beta_r \geq \beta_{r+1}$. By Lemma 10, $\sum_{r=1}^{k} \alpha_r \beta_r \geq 0$, which means that relations 2 hold after the extraction event. Next, we deal with the arrival of a new volume u. Assume w.l.o.g. that the volume is added in X' to block r, therefore all the blocks $1, \ldots, r-1$ are full. Then the relations hold for these prefixes. For the remaining blocks relations 2 hold since we have added at most the volume u to them. ■

Proof of Lemma 7. First, we prove that $X_r = 1$, for $r < T_1$, by Corollary 3, for each $r \notin T$ we have $X_r \in \{0,1\}$. Therefore, for any $r < T_1$ we have $L^{\text{OPT}}_{r-1} \leq 1$, hence $X_r = 1$. Next, we prove that for any T_t, T_{t+1} and $0 < r < T_{t+1} - T_t$ then $X_{T_t+r} = 0$ if $r < B$ and $X_{T_t+r} = 1$ otherwise. If $r < B$ then $L^{\text{OPT}}_{T_t+r-1} \geq B - (r-1) \geq 2$ hence, by Corollary 3, $X_{T_t+r} = 0$. Otherwise, since $X_{T_t+r'} = 0$ for any $r' < B$ and $X_{T_t+r} \in \{0,1\}$, we have $L^{\text{OPT}}_{r-1} \leq 1$, therefore $X_r = 1$. Finally, $X_{T_{t+1}} = \min\{d, B\}$ since $L^{\text{OPT}}_{T_{t+1}} = B$.
■

Proof of Lemma 8. In order to bound ϱ we proceed as follows. It is easy to verify that after T_k there is no more balls arrival (otherwise we omit those balls and subtract the same gain from OPT and FRC). Therefore, let $d_0 = T_1$, and $d_t = T_{t+1} - T_t$, then

$$G^{\text{OPT}}(X) = d_0 + B + \sum_{t=1}^{T-1} d_i \ ,$$

$$G^{\text{FRC}}(X) = d_0 - O_{d_0} + B + \sum_{t=1}^{T-1} \left(d_i - O_{T_{t+1}}\right),$$

and then

$$\rho(X) = \frac{G^{\mathrm{OPT}}(X)}{G^{\mathrm{FRC}}(X)} \le \max\left\{\frac{d_0 + B}{d_0 - O_{d_0} + B}, \max_t \left\{\frac{d_t}{d_t - O_{T_{t+1}}}\right\}\right\}.$$

Let $\bar{V}(d)$ be the load in FRC after d steps, assuming that FRC started with an empty load, and in each of the subsequent d steps a single ball arrives. Using these notations and Lemma 7, we can compute the overflow as follows:

– $O_{T_1} = \bar{V}(T_1)$.
– $O_{T_{t+1}} = V(d_t) + d_t - B$, if $d_t < B$.
– $O_{T_{t+1}} = V(d_t) + \bar{V}(d_t - B)$, if $d_t \ge B$.

We thus get

$$\varrho^{\mathrm{FRC}} \le \max_d \left\{\frac{d+B}{d - \bar{V}(d) + B}, I_{d<B}\frac{d}{B - V(d)}, I_{d\ge B}\frac{d}{d - V(d) - \bar{V}(d-B)}\right\},$$

where $I_{\mathrm{condition}} = 1$ if the condition holds, and 0 otherwise.

We observe that $V(d) + \bar{V}(d) = B$, since if we start with a ball in each bin after an extraction and arrival of a ball, then each extraction is partially from V, with the complement from $\bar{V}$. At each step the vacant volume in a block in $V(d)$ is exactly the volume taken in $\bar{V}(d)$. Therefore,

$$\varrho^{\mathrm{FRC}} \le \max_d \left\{\frac{d+B}{d + V(d)}, I_{d<B}\frac{d}{B - V(d)}, I_{d\ge B}\frac{d}{d - V(d) - B + V(d-B)}\right\}$$

Note that by putting $d' = d + B$ in the third expression, the result dominates the first expression. Note also that $V(d)$ is concave, since if there are more balls, then the extracted volume is larger. Therefore ($V(0) = B$),

$$\frac{d}{B - V(d)} \le \frac{B}{V(0) - V(B)}.$$

Hence, the third expression for $d = B$ dominates the second expression in the max formula. We conclude that

$$\rho^{\mathrm{FRC}} \le \max_{d\ge B} \frac{d}{d - V(d) - B + V(d-B)}.$$

■

Proof of Lemma 9. Let $W(r,d)$ be the volume left after d steps when r balls are in the blocks with higher indices. We show that $W(r,d) \approx W(r - H(r/B), d-1)$ for $p^{\mathrm{fac}} \to 1$ (since $K/B \to 0$). First, we bound the extracted volume after one extraction step, given that the r balls are in the blocks with higher indices. Let $k = B/K - \lceil r/B \rceil$, be the first block with balls and $b = r - K \cdot \lfloor r/K \rfloor$, the volume in this block. The volume extracted at the next step is

$$\begin{aligned}
r - W(r,1) &= \sum_{j=k+1}^{B/K} q^j + q^k \frac{b}{K} \\
&= H\left(\frac{(k+1)K}{B}\right) + \left(H\left(\frac{kK}{B}\right) - H\left(\frac{(k+1)K}{B}\right)\right)\frac{b}{K} \\
&\geq H(r/B) - (p^{\text{fac}} - 1).
\end{aligned}$$

Since $W(r,0) = r$, we have $W(r,1) \leq W(r - H(r/B), 0) + (p^{fac} - 1)$. We want to write $W(r,d)$ as a recursive function. Hence, except for the extracted volume at the next step, $W(r, d-1)$ assumes that all the volume is in the blocks with higher indices. For that we note that if a block and its predecessor are full, then after an extraction event this block remains full. Therefore, we may need to push at most one ball volume from a block to the successor block, which might reduce the volume after d steps by at most $p^{\text{fac}} - 1$. Therefore, we have

$$W(r,d) \leq W(r - H(r/B), d-1) + 2(p^{\text{fac}} - 1).$$

Defining f by $f(0) = 1$ and $f(x + 1/B) = f(x) - H(f(x))/B$, we get $V(d) = W(B,d) \leq Bf(d/B) + 2d(p^{\text{fac}} - 1)$. The bound on $W(r,d)$ is tight, since $W(r,1) \geq W(r - H(r/B), 0)$, and $W(r,d) \geq W(r - H(r/B), d-1)$. We have $V(d) \geq Bf(d/B)$. Since we assume $B \gg 1$, we get the following differential equation for f:

$$f'(x) = -H(f(x)).$$

■

Proof of Lemma 1. Observe that $L_i^{\text{B}_{\text{TOB}}}$ is a random variable, and that the loads in question form a Markov chain. Specifically, the conditional expectation of the load of B_{TOB} at any step $i+1$ depends only on the preceding load $L_i^{\text{B}_{\text{TOB}}}$. Formally,

$$\mathbb{E}\left(L_{i+1}^{\text{B}_{\text{TOB}}} | L_1^{\text{B}_{\text{TOB}}}, \ldots, L_i^{\text{B}_{\text{TOB}}}\right) = \mathbb{E}\left(L_{i+1}^{\text{B}_{\text{TOB}}} | L_i^{\text{B}_{\text{TOB}}}\right).$$

The conditional expectation of the load satisfies

$$\mathbb{E}\left(L_{i+1}^{\text{B}_{\text{TOB}}} | L_i^{\text{B}_{\text{TOB}}}\right) \leq \min\left\{L_i^{\text{B}_{\text{TOB}}} \cdot \left(1 - \frac{Q^{\text{B}_{\text{TOB}}}}{K + \mathit{\Delta} K}\right) + X_{i+1}, K + \mathit{\Delta} K\right\}.$$

Note that $L_{i+1}^{\text{B}_{\text{FRC}}} = \min\left\{L_i^{\text{B}_{\text{FRC}}} \cdot \left(1 - \frac{Q^{\text{B}_{\text{FRC}}}}{K}\right) + X_{i+1}, K\right\}$. We examine only the cases where B_{FRC} will not overflow, and B_{TOB} will also not overflow, since we enlarge it by $\mathit{\Delta} K$. Therefore, w.l.o.g., we may assume that

$$\mathbb{E}\left(L_{i+1}^{\text{B}_{\text{TOB}}} | L_i^{\text{B}_{\text{TOB}}}\right) = L_i^{\text{B}_{\text{TOB}}} \cdot \left(1 - \frac{Q^{\text{B}_{\text{TOB}}}}{K + \mathit{\Delta} K}\right) + X_{i+1} = L_i^{\text{B}_{\text{TOB}}} \cdot \left(1 - \frac{Q^{\text{B}_{\text{FRC}}}}{K}\right) + X_{i+1}, \tag{3}$$

$$L_{i+1}^{\mathrm{B_{FRC}}} = L_i^{\mathrm{B_{FRC}}} \cdot \left(1 - \frac{Q^{\mathrm{B_{FRC}}}}{K}\right) + X_{i+1}. \tag{4}$$

We now define the process Z_i as the difference between $L_i^{\mathrm{B_{TOB}}}$ and $L_i^{\mathrm{B_{FRC}}}$. We note that given the difference at a step, the expected difference at the next step is closer to 0. Since $L_i^{\mathrm{B_{TOB}}} = L_i^{\mathrm{B_{FRC}}} + Z_i$, relations (3) and (4) yield

$$\begin{aligned}
\mathbb{E}[L_{i+1}^{\mathrm{B_{TOB}}}|Z_i, L_i^{\mathrm{B_{TOB}}}] &= L_i^{\mathrm{B_{TOB}}} \cdot \left(1 - \frac{Q^{\mathrm{B_{FRC}}}}{K}\right) + X_{i+1} \\
&= L_i^{\mathrm{B_{FRC}}} \cdot \left(1 - \frac{Q^{\mathrm{B_{FRC}}}}{K}\right) + Z_i \cdot \left(1 - \frac{Q^{\mathrm{B_{FRC}}}}{K}\right) + X_{i+1} \\
&= L_{i+1}^{\mathrm{B_{FRC}}} + Z_i \cdot \left(1 - \frac{Q^{\mathrm{B_{FRC}}}}{K}\right).
\end{aligned}$$

Therefore,

$$|\mathbb{E}[Z_{i+1}|Z_i]| - |Z_i| \leq 0. \tag{5}$$

Note that from (5) it follows that

$$E[Z_{i+1}|Z_i, Z_i > 1] \leq Z_i,$$

$$E[Z_{i+1}|Z_i, Z_i < -1] \geq Z_i.$$

Moreover, $Z_{i+1} - Z_i \leq 2$. Therefore, we can use Azuma's inequality for any interval (there are N^2 intervals of length at most N). ■

Proof of Lemma 2. For the sake of our analysis, we split the input sequence X, with respect to the fractional block, into two volume types: green and red, i.e., $X = X^{\mathrm{g}} + X^{\mathrm{r}}$. The green volume is the part that is added to the fractional block, while the red volume is the part that has overflown in the fractional block. The output sequence splits accordingly as $Y = Y^{\mathrm{g}} + Y^{\mathrm{r}}$. By definition, $Y^{\mathrm{r,B_{FRC}}} = X^r$. Now let $Y^{\mathrm{B_{TOB}}} = Y^{\mathrm{g,B_{TOB}}} + Y^{\mathrm{r,B_{TOB}}}$, For the analyze, the green volume is not aware of the red volume in the block, i.e. it may overflown it if the block is full. Note that the green and the red are with respect to the fractional block.

By definition, X^{g} do not overflow on $\mathrm{B_{FRC}}$, hence, by Lemma 1, at any step it holds that $|L^{g,\mathrm{B_{FRC}}} - L^{g,\mathrm{B_{TOB}}}| \leq \Delta K$ with probability of at least ϵ_N. Therefore, by volume preservation,

$$-\Delta K \leq \sum_{j=1}^{i} \left(Y_j^{\mathrm{g,B_{TOB}}} - Y_j^{\mathrm{g,B_{FRC}}}\right) \leq \Delta K. \tag{6}$$

Next we deal with the red volume. By definition, $L^{\mathrm{r,B_{FRC}}} = 0$ and $X_i^r > 0$ only when $L_i^{g,\mathrm{B_{FRC}}} = K$, by Lemma 1, $L_i^{\mathrm{g,B_{TOB}}} \geq K - \Delta K$. Hence, at any step $L^{\mathrm{r,B_{TOB}}} \leq (K + \Delta K) - (K - \Delta K) = 2\Delta K$. Therefore, by volume preservation

$$0 \leq \sum_{j=1}^{i} \left(Y_j^{\mathrm{r,B_{TOB}}} - Y_j^{\mathrm{r,B_{FRC}}}\right) \leq 2\Delta K. \tag{7}$$

Finally, Lemma 2 follows from adding relation 6 and relation 7.

■

Proof of Lemma 3. Let X be such a sequence. For the sequence X the probability that X^m (an input sequence for the blocks $m, \ldots, B/K$) does not cause an overflow to HYB_{m-1} is at least $(1-\epsilon)^{m-1}$. HYB_{m-1} and HYB_m have the same first $m-1$ randomized blocks. By coupling, we assume that the input instance X^m is the same for both algorithms. Let Y^m be the output sequence of block m in HYB_m for the input X^m. Let $\bar{Y}^m$ be the (random) output sequence of block m in HYB_m for the input X^m. By applying Lemma 2 for the given X^m we get that with probability of at least $(1-\epsilon)$ we have, $-\Delta K \leq \sum_{j=1}^{i} \left(\bar{Y}_j^m - Y_j{}^m\right) \leq 3\Delta K$.

The probability that X^m does not overflow HYB_m as well as $\bar{Y}^m$ satisfies the above relation is at least $(1-\epsilon)^{m-1}(1-\epsilon) = (1-\epsilon)^m$. Accordingly, for such sequences let $\bar{X}^{m+1}$ and X^{m+1} be a pair of input sequences of $\mathrm{HYB}_m, \mathrm{HYB}_{m-1}$ (for the $m+1$ block) respectively. One can verify that also the input sequence satisfies,

$$-\Delta K \leq \sum_{j=1}^{i} \left(\bar{X}_j^{m+1} - X_j^{m+1}\right) \leq 3\Delta K. \tag{8}$$

Assume, by contradiction, that $\bar{X}^{m+1}$ overflows HYB_m, let $i \leq N$ be the first step at which overflow takes place. Let $j < i$ be a step where $\bar{X}_j^{m+1} > X_j^{m+1}$. We modify the sequence $\bar{X}^{m+1}$ by pushing $v = \bar{X}_j^{m+1} - X_j^{m+1}$ volume from step j to the next step $j+1$, by Lemma 6 the overflow on step i for the modified sequence can only increase. Note that after the shift relation 8 still holds. We can do this repeatedly, eventually we get the sequence $\hat{X}^{m+1}$ such that for any step $j < i$ that $\hat{X}_j^{m+1} \leq X_j^{m+1}$. Since X^{m+1} does not overflow for HYB_{m-1} by monotonicity and by Observation 9, we conclude that on $\langle \hat{X}_1^{m+1} \ldots, \hat{X}_{i-1}^{m+1}, X_i^{m+1} \rangle$ algorithm HYB_m would have at least $4\Delta K$ vacant volume. Note that, $\hat{X}_i^{m+1} - X_i^{m+1} \leq 4\Delta K$ since relation 8 holds. Hence, HYB_m does not overflow in step i for sequence $\hat{X}^{m+1}$ and hence it does not overflow in step i for sequence $\bar{X}^{m+1}$. ■

Proof of Theorem 8. We construct a randomized algorithm according to $\mathrm{HYB}_{B/K}$. Therefore, we design a slightly smaller fractional algorithm $\widetilde{\mathrm{FRC}} = \mathrm{HYB}_0$. The fractional algorithm $\widetilde{\mathrm{FRC}}$ uses fewer bins $\widetilde{B} < B$ than the randomized algorithm. In addition, it reduces the total fractional probability, so that the total sum to choose blocks is less than 1. These two modifications result in just a slightly bigger overflow in the fractional algorithm. We choose $K = \widetilde{B}^{7/8}$ and $\Delta K = B^{5/8}$. Then the number of blocks is $\widetilde{B}^{1/8}$. For this choice of parameters, application of Corollary 1 shows that for any sequence of length at most $N \leq B^{9/8}$ which does not overflow $\widetilde{\mathrm{FRC}}$, this sequence will not overflow $\mathrm{TOB} = \mathrm{HYB}_{B/K}$ with probability

$$(1-\epsilon)^{B/K} = \left(1 - N^3 \exp\left(\frac{-\Delta K^2}{8N}\right)\right)^{B/K} = \left(1 - B^{27/24} \exp\left(\frac{-B^{10/8}}{8B^{9/8}}\right)\right)^{B^{1/8}} = 1 - o(1).$$

We conclude that for sequences of length at most $B^{9/8}$ the expected gain of the randomized algorithm is at least the gain of $\widetilde{\text{FRC}}$ times $1 - o(1)$.

For an input sequences longer than $B^{9/8}$ we split the input sequence into parts, each of length $B^{9/8}$. After each part, we may assume that the current load of the randomized algorithm is overflown, and the current fractional gain is cleared and added to its gain. Since we consider a compact sequence, the total gain of FRC on each part is $\Theta(B^{9/8})$. Therefore, the difference in the gain of the after the modification is at least a factor of $1 - \frac{B}{B^{9/8}} = 1 - o(1)$ from the original gain.

To complete the proof of Theorem 8, we relate $\widetilde{\text{FRC}}$ to FRC. Indeed, $B = \widetilde{B} + 5\Delta K \widetilde{B}^{1/8} = \widetilde{B} + 5\widetilde{B}^{6/8}$, and therefore $\widetilde{B}/B = 1 + o(1)$. We also bound the reduction of the total probability in $\widetilde{\text{FRC}}$. Note that we increase the probability by at most a factor of $((K + 4\Delta K)/K)^{B^{1/8}} = 1 + o(1)$. By scaling we conclude that $G^{\widetilde{\text{FRC}}}(X) \geq G^{\text{FRC}}(X)(1 - o(1))$ for any input sequence X.

Additionally, for any input X sequence, let X' a truncated version of X with respect of $\widetilde{\text{FRC}}$ to X i.e. $X'_i \leq X_i$, $G^{\widetilde{\text{FRC}}}(X') = G^{\widetilde{\text{FRC}}}(X)$ and $O^{\widetilde{\text{FRC}}}(X') = 0$. By Corollary 1,

$$G^{\text{HYB}_{B/K}}(X) \geq G^{\text{HYB}_{B/K}}(X') \geq G^{\widetilde{\text{FRC}}}(X')(1 - o(1)) = G^{\widetilde{\text{FRC}}}(X)(1 - o(1)).$$

This completes the proof of Theorem 8. ■

B.1 Analyzing a specific randomized algorithm against the optimal one

A natural choice for H would be

$$H^p(x) = (1 - x)^\alpha.$$

This yields a competitive ratio of 1.59, for $\alpha = 0.6$. However, H is not admissible, because its derivative is not bounded near $x = 1$. To fix this, we replace H^p, keeping the notation, by

$$H^p(x) = \frac{(1 + \gamma - x)^\alpha - \gamma^\alpha}{(1 + \gamma)^\alpha - \gamma^\alpha}$$

where $\gamma \geq 0$ is small enough (e.g., $\gamma = 0.001$), which hardly affects the competitive ratio. The competitive ratio can be reduced to 1.55 by choosing

$$H^p(x) = \frac{(1 + \gamma - x)^\alpha + \beta x - \gamma^\alpha}{(1 + \gamma)^\alpha + \beta - \gamma^\alpha},$$

where $\alpha = 0.13$, $\beta = 2.5$, and $\gamma = 0.001$. This concludes the proof of Theorem 1.

C Lower bounds proofs

Proof of Theorem 5. For any deterministic algorithm we describe a sequence and an assignment to the bins. The sequence is fixed for all algorithms and it is $X = \langle 1, \ldots, 1, B \rangle$, where the number of 1's is $B - 1$. The assignment to the bins depends on the algorithm. Specifically, let $b_1, \ldots, b_{B-1}$ be the indices of the first $B - 1$ bins that the algorithm clears. There exists a bin $j_1 \in \{1, \ldots, B\}$ s.t. $j_1 \neq b_i$ for any $i \geq 1$, since there are at most $B - 1$ different numbers in $b_1, \ldots, b_{B-1}$. Hence, the algorithm does not clear bin j_1 in the first $B - 1$ steps. Next, there exists $j_2 \in \{1, \ldots, B\} - \{j_1\}$ s.t. $j_2 \neq b_i$ for any $i \geq 2$. Similarly, we define j_i for $1 \leq i \leq B - 1$. Eventually, we have $B - 1$ different indices $j_1, \ldots, j_{B-1}$. The adversary assigns the first ball to bin j_1, the second ball to bin j_2, and so on. This assignment is feasible since the indices are different. Moreover, the algorithm does not clear any ball in the first $B - 1$ extractions, while the optimal algorithm clears $B - 1$ balls, and all its bins are vacant. After the first $B - 1$ balls, as mentioned, B balls arrive at once. Those balls are stored in the optimal algorithm run; however, $B - 1$ balls out of them are discarded by the algorithm. Therefore, the optimal gain is $2B - 1$, while the algorithm gain is B. Note that we can repeat this sequence after additional B extraction events with no balls arriving. ■

Proof of Theorem 2. For any algorithm ALG, let $E_h^{\text{ALG}}(k)$ denote the expected number of balls extracted, in extractions steps $k+1, \ldots, k+B-1$, given that in ALG at step k all the bins contain a ball, and no more balls arrive. In addition, let $E_t^{\text{ALG}}(k)$ denote the expected number of balls extracted, in steps $k, \ldots, k+B-1$, starting with empty bins, and so that at steps $k+1, \ldots, k+B-1$ a single ball arrives, and the adversary assigns each new ball to the last examined bin.

Lemma 11. *For any algorithm and for any k, we have $E_h^{\text{ALG}}(k) + E_t^{\text{ALG}}(k) = B$.*

Proof. For each sequence of bins we define an indicator $I_k(j)$, where $I_k(j)$ indicates whether the bin examined in step $k + j$ is different from all bins examined in steps $k, \ldots, k + j - 1$. Therefore, we have $E_h^{\text{ALG}}(k) = \mathbb{E}[\sum_{j=0}^{B-1} I_k(j)]$ and $E_t^{\text{ALG}}(k) = \mathbb{E}[\sum_{j=0}^{B-1} (1 - I_k(j))]$

We examine $E_h^{\text{ALG}}(k)$ for $k \in \{0, B, 2B, \ldots, dB\}$ for some d.

- Case 1: there exists i s.t. $E_h^{\text{ALG}}(iB) \geq 2B/3$. We consider the sequence $\langle 0, \ldots 0, 1, \ldots, 1, B \rangle$, where the number of 0's is iB and the number of 1's is B. The adversary assigns each ball to the most recently examined bin (except in the last step). By Lemma 11, $E_t^{\text{ALG}}(iB) \leq B/3$, hence the expected gain for this sequence is at most $4B/3$, while the optimal gain is $2B$, and the ratio is $3/2$.
- Case 2: for all i, $E_h^{\text{ALG}}(iB) < 2B/3$. Set $\widetilde{X} = \langle B, 0, \ldots, 0 \rangle$ where the number of 0's is $B-1$ and the input sequence is $X =< \widetilde{X}, \ldots, \widetilde{X} >$, where the number

of $\widetilde{X}$'s is d, the optimal gain is dB, while ALG gain is at most $2dB/3 + B$, and the ratio $1/(2/3 + 1/d)$.

For any ALG, case 1 or 2 happens, therefore we can choose d large enough and conclude that the competitive ratio of ALG is at least $1.5 - \epsilon$, for any $\epsilon \geq 0$. Note that we can repeat this process for any dB steps. ■

D Application: Prompt Mechanism for Packet Scheduling

We consider the problem of developing *prompt truthful mechanisms* for *periodic bounded capacity auctions.* One practical motivation for studying the above problem relates to buffer management issues arising in context of network devices such as switches and routers.

In this application domain, one deals with incoming sequences of (strategic) packets with private values for being served. There is a buffer that can accommodate a bounded number of packets at any give time. One packet can be transmitted in each time slot. The goal is to design a truthful prompt mechanism maximizing the total value of the transmitted packets (packets are charged only if transmitted). In a prompt mechanism [7] the price for a packet should be determined at its transmission.

One possible prompt mechanism for this problem is FIFO transmission with greedy admission control (i.e., one keeps the packets with the highest values). The FIFO mechanism is truthful, 2-competitive [10], and supports prompt payments. Specifically, it is easy to prove that the payment of each transmitted packet depends only on the bids of buyers that arrived before its transmission. In fact, computing these prices is simple. Essentially, the price for a transmitted packet is the maximum over all the bids of rejected packets between the arrival of that packet and its transmission time.

A mechanism with greedy admission control and which transmits a packet independently and uniformly at random, was analyzed in [4]. Specifically, therein it was shown how to reduce the competitive ratio of such mechanisms to the value of loss of serving in the dark. By reducing the loss of serving in the dark using a non-uniform distribution one arrives at the ratio 1.55. By using Theorem 1, we achieve a 1.55 competitive prompt mechanism. Since the technical details of the reduction appear in [4], we omit them.