Algorithmica (2013) 65:60–94
DOI 10.1007/s00453-011-9574-6

Collecting Weighted Items from a Dynamic Queue

Marcin Bienkowski · Marek Chrobak · Christoph Dürr · Mathilde Hurand · Artur Jeż · Łukasz Jeż · Grzegorz Stachowiak

Received: 24 November 2010 / Accepted: 14 September 2011 / Published online: 4 October 2011

Abstract We consider online competitive algorithms for the problem of collecting weighted items from a dynamic queue S. The content of S varies over time. An update to S can occur between any two consecutive time steps, and it consists in deleting any number of items at the front of S and inserting other items into arbitrary locations in S. At each time step we are allowed to collect one item in S. The objective is to maximize the total weight of collected items. This is a generalization of bounded-delay packet scheduling (also known as buffer management). We present several upper and lower bounds on the competitive ratio for the general case and for some restricted variants of this problem.

Keywords Online algorithms · Competitive analysis · Packet scheduling · Buffer management

Supported by MNiSW grants N N206 368839, 2010–2013 and N N206 490638, 2010–2011, NSF grants OISE-0340752 and CCF-0729071, and ANR Alpage.

Most of the work was carried out when M. Hurand was at LIX Ecole Polytechnique.

A preliminary version of this paper appeared in the Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms (SODA'09).

M. Bienkowski · A. Jeż (✉) · Ł. Jeż · G. Stachowiak
Institute of Computer Science, University of Wrocław, 50-383 Wrocław, Poland
e-mail: aje@cs.uni.wroc.pl

M. Chrobak
Department of Computer Science, University of California, Riverside, CA 92521, USA

C. Dürr
CNRS, LIP6, Université Pierre et Marie Curie, 75252 Paris Cedex 05, France

M. Hurand
Google, 8002 Zürich, Switzerland

1 Introduction

We consider the problem of collecting weighted items from a dynamic queue S (an ordered list). The content of S varies over time. An update to S can occur between any two consecutive time steps, and it consists in deleting any number of items at the front of S (that is, a prefix of S) and inserting other items into arbitrary locations in S. An item, once deleted, cannot be re-inserted—in other words, it "expires". We are allowed to collect one item from S per step. Each item can be collected only once. The objective is to maximize the total weight of the collected items.

We focus on the online version of this dynamic queue problem. An online algorithm needs to make a decision about which item in S to collect in each step, before the next update of S is revealed.

The example in Table 1 illustrates the trade-off faced by such an online algorithm: collecting items near the front of S is likely to increase the number of collected items, but it could result in losing some heavy items later in the queue. In this example, for simplicity, we use integer values to represent both the items and their weights. Initially, items $3, 7, 4$ are inserted into the queue. Suppose that an online algorithm collects item 3 in step 1. Before step 2, items $3, 7$ are deleted and items $2, 9$ are inserted into the queue. In step 2, the online algorithm collects item 9. Before step 3, items $2, 4$ are deleted. In step 3, the algorithm cannot collect any items, since the only item 9 still in S has already been collected. After step 3, item 9 is deleted. So the overall gain of the algorithm is $3+9=12$. The optimum solution is to collect item 7 in step 1, item 4 in step 2, and item 9 in step 3, for the total gain of $7+4+9=20$.

To our knowledge, this dynamic queue problem has not been explicitly addressed in the literature, though it naturally generalizes the well-studied problem of bounded-delay buffer management. In the buffer management problem, packets with weights and deadlines arrive in a buffer of a network link; the weights represent various quality-of-service levels. At each step, we can send one packet along the link. The objective is to maximize the total weight of packets sent before their deadlines. This is a special case of our dynamic queue problem, where packets are represented by items ordered according to deadlines. The difference is crucial though: in packet scheduling, packet arrival times are unknown but their deadlines are revealed at their arrival, while in dynamic queues *both* the arrival and deadlines are not known.

Competitive algorithms for various versions of bounded-delay buffer management problem have been extensively studied [1, 3–6, 9–12]. In particular, it is known that no deterministic online algorithm has competitive ratio better than $\phi \approx 1.618$ [1, 4],

Table 1 An example of collecting items from a dynamic queue. Each row shows the state right after the queue has been updated and before the algorithm's move. Underlined numbers represent items collected by the algorithm

Step	Deleted items	Queue
1		3 7 4
2	$\underline{3}$ 7	2 4 9
3	$\underline{3}$ 7 2 4	$\underline{9}$
4	$\underline{3}$ 7 2 4 $\underline{9}$	

and an algorithm with competitive ratio ≈ 1.828 has been recently developed [5] (see also [12]). Closing the gap between these bounds remains an intriguing open problem. For agreeable deadlines (where the items are released in order of non-decreasing deadlines), an upper bound of ϕ has been established [11].

The buffer management model assumes that we can send a packet in each time step. This is not the case in networks when access to the link may be only intermittent. One example is that of a tiered QoS systems, where all traffic is divided into classes with different QoS guarantees. Packets from a lower tier are transmitted only if no higher-tier packets are pending. In this scenario, maximizing the total value of sent packets from the lower tier is equivalent to our item collection problem. The intuition is that we can simulate item deletions by blocking time slots using "tight" top-tier packets, which must be transmitted upon release. (We explain this in more detail in Sect. 2.)

Tiered systems are increasingly common—for example, the newly introduced WiMAX standard for the "last mile" connectivity comprises of five tiers with traffic ranging from the voice and video service with real-time guarantees to the lowest best-effort service for web-browsing and data transfer [15]. There are other scenarios where link access is intermittent or unpredictable, due to competition with other traffic streams, errors or interference in wireless channels, or link failures. One extreme example is that of meteor-burst communication, where a connection requires an entry of a meteor into the atmosphere at a desired location [14].

The dynamic queue problem is also loosely related to various versions of online bipartite matching (it can be thought of as computing a maximum weight matching between time steps and items) and to the adwords problem, both previously studied in the literature (see [2, 8, 13] and the references therein). Due to different focus and assumptions, however, algorithmic ideas developed for those problems do not seem to apply straightforwardly to dynamic queues.

1.1 Our Results

It is easy to see that algorithm GREEDY, which always collects the maximum-value item, is 2-competitive for general dynamic queues, exactly as for the buffer management problem [6, 9]. We improve this bound, by providing a 1.897-competitive algorithm PRUDENTMARK (see Sect. 4). Our ratio is larger than the ratio of ≈ 1.828 for the more restricted problem of buffer management [5], but the algorithm in [5] (as well as the one in [12]) uses information about packet deadlines and is not applicable to dynamic queues. We also show that our analysis of PRUDENTMARK is essentially tight, i.e., we give an example on which—with any choice of parameters—the competitive ratio of PRUDENTMARK is at least 1.894.

Next, in Sect. 5, we study the special case of *FIFO queues*, where items can be added only at the end of the queue. The FIFO case generalizes the variant of buffer management with agreeable deadlines. For this case we give an algorithm EFH, which is 1.737-competitive. Moreover, we show that its analysis is tight.

We then turn our attention to lower bounds. It is easy to establish a lower bound of ϕ for any deterministic algorithm for dynamic queues, using only two items (see Sect. 6). We improve this bound, by proving a lower bound of ≈ 1.63. This bound applies even to the *decremental case*, where all items are inserted at the beginning

Table 2 A summary of known results on the deterministic variant of the buffer management problem and our results on collecting items from a dynamic queue. (All values rounded to three decimal places.) Unreferenced results on buffer management are straightforward implications from other results in the table

	Buffer management		Collecting items	
	Lower bound	Upper bound	Lower bound	Upper bound
Unrestricted	1.618	1.828 [5]	1.637	1.897
Memoryless	1.618	1.893 [5]	2	2
FIFO	1.618 [1, 4, 6]	1.618 [11]	1.637	1.737

and no insertions are allowed afterwards. Note that such a scenario is trivial for buffer management, since an online algorithm can use the deadline values to pre-compute the optimal schedule of packet transmissions.

We also show two tight lower bounds for memoryless algorithms, which make decisions based only on the weights of the pending items. For deterministic algorithms we prove a lower bound of 2. This contrasts with a 1.893-competitive memoryless algorithm for buffer management [5]. Thus, for memoryless algorithms, knowing the exact deadlines helps. For randomized memoryless algorithms (against an adaptive adversary), we present a lower bound of $\mathrm{e}/(\mathrm{e}-1)$, matching an upper bound of RMIX [3] (see also [7]), which works, without any changes, in our model.

Table 2 summarizes our results and compares them to the corresponding results on buffer management.

2 Preliminaries

We refer to the items currently in the queue S as *active*. In other words, those are the items that have been already inserted but not yet deleted. (It is worth stressing here that whether an item is active or not is not related to it being collected or not by an algorithm under consideration; it only depends on the instance, that is on the sequence of queue updates.) We denote the weight of an item x as w_x and the total weight of a set X of items as $w(X)$. We use symbol "$\lhd$" to represent the ordering in S, i.e., $a \lhd b$ means that a is before b (or *a precedes b*) in the queue. This relation is well-defined, since items cannot be re-inserted into the queue. In fact, it can be extended to a linear order on all items in the instance by letting $a \lhd b$ if a was deleted from the queue before b was inserted.

If all queue updates are specified upfront, an optimal solution can be computed in polynomial time by reduction to maximum-weight matching: Represent the instance as a bipartite graph G whose partition classes are items and time steps; only $O(n)$ time steps need to be considered, where n is the number of items. An item a is connected to the time steps when a is active with edges of weight w_a. The maximum-weight matching in G represents an optimal collection sequence.

An algorithm A for dynamic queues is called *online* if at each step its decision as to what item from S to collect is made without knowledge of future queue updates. An item is called *pending* for A at a given step if it is active but not yet collected by A. Of course, only pending items can be collected.

Using a routine exchange argument, it is easy to show that, without loss of generality, optimal (adversary's) solutions satisfy the following property:

Earliest-Expiration-First (EEF) Property: Suppose that, at some time step, we have two active items $a \lhd b$, and that at this time step the adversary collects b. Then the adversary will not collect a in the future.

In other words, at each step, the adversary forfeits all active items in the queue that precede the item that he collects at this step. Motivated by this observation, throughout the paper we say that an item a is *pending for the adversary* at a certain time step if it can be collected later by the adversary satisfying the EEF property. Thus, to emphasize, the notion of "pending for the adversary" is slightly different from that of "pending for an online algorithm".

In FIFO queues, if two items $a \lhd b$ are active and a is pending for the adversary, then so is b. In general queues, however, this does not need to be true, since a may have been inserted into the queue after the adversary collected or forfeited b.

2.1 Competitive and Amortized Analysis

An online algorithm A is called *R-competitive* if its gain on any instance I is at least the optimum gain on I divided by R. An additive constant is sometimes allowed in this bound; in our upper bounds this constant is 0, and our lower bounds can be easily modified to work for this more general definition. The *competitive ratio* of A is the smallest R for which A is R-competitive.

All of our R-competitiveness proofs are based on *amortized analysis*. Ideally, in each step, we would like to bound the adversary's gain by the gain of the algorithm multiplied by R. However, this is not always possible; in some steps the former quantity is larger than the latter one. On the other hand, in other steps this bound may hold even with some slack. Thus, to obtain the bound on the total gains, over the whole request sequence, we use the following accounting framework.

When an algorithm collects an item i, we receive an allowance of Rw_i units. We want to show that the total allowance will cover the total adversary's gain. At any step, we may use the allowance for this step to pay for the item collected by the adversary. If there is any surplus left, we store it in the form of credits assigned to certain items pending for the adversary, according to the rules described in the analysis.

At each step, some items may cease to be pending for the adversary, and we can recover the credit stored on these items. Therefore, generally, we deal with three quantities: our allowance (R times the algorithm's gain), the adversary's gain, and the change of credits, that may be positive or not. The sum of the two latter quantities is called the *adversary's amortized gain*. The objective of the analysis is to show that at each step our allowance will cover (be at least as large as) the adversary's amortized gain.

Throughout the paper, in our proofs, we distinguish between actions of the algorithms and those related to their competitive analysis by writing "the algorithm does..." to describe the algorithm's steps, and "we do..." to describe manipulations on credits, etc., that are part of the analysis.

2.2 Our Model vs. Buffer Management in Tiered QoS Systems

In the introduction we discussed the relation between dynamic queues and buffer management in tiered QoS systems. Consider, for simplicity, a two-tier system, where

packets from tier 1 can be transmitted at any time, while packets from tier 2 can be transmitted only when there are no pending packets from tier 1. Then an instance of the tier-2 buffer management problem consists of a set of *blocked* time units and a collection of packets, with each packet i specified by a triple (r_i, d_i, w_i) where r_i is its release time, d_i its deadline and w_i its weight (value). We can transmit a packet i at time t only if t is not blocked, if i has not been sent earlier, and if $r_i \leq t < d_i$. The objective is to maximize the value of transmitted packets.

We now show that the item collection problem for dynamic queues is equivalent (with a minor caveat) to the tier-2 buffer management problem, in the sense that an online algorithm for one problem can be converted into an online algorithm for the other, without increasing its competitive ratio.

First, we show how we can transform an online algorithm A for the dynamic queue problem into an algorithm A' for scheduling tier-2 packets. The idea is that A will convert the given instance of tier-2 buffer management into an instance of dynamic queues and use A as a black-box oracle to determine which packets should be transmitted. The items in the instance of dynamic queues will be exactly the tier-2 packets, ordered in the queue by increasing deadlines. More specifically, at any step t, algorithm A' behaves as follows. If t is a blocked step, then A' does nothing. (In particular, in blocked steps the time of A does not advance.) If t is not blocked, then A' increments the clock of A and updates the queue by performing all updates (packet releases and expirations) that occurred since the last non-blocked step. If A collects item i in this step, A' transmits packet i. The gain of A' is equal to the gain of A and the optimal solutions of the two instances (packet scheduling and dynamic queue) are the same. Thus, the competitive ratios of A and A' are the same as well.

Next, we show how to convert an online algorithm B for scheduling 2nd-tier packets into an online algorithm B' for dynamic queues. Here, B' will convert, in an online manner, its instance into an instance of 2nd-tier buffer management. Each item i will be converted by B' into a packet i with the same weight. (The release times and deadlines are defined below.) B' also chooses which time slots are blocked. B' feeds this information into B, uses B as a black-box oracle to see which packet is transmitted, and then it collects the corresponding item. The idea is to simulate item expirations by creating blocked steps that will force expirations of the corresponding packets. In this reduction, we need to assume that there is a known upper bound, say m, on the total number of released items and on the number of steps (this is the caveat mentioned at the beginning of this sub-section).

Each step of B' will be replaced by one "real" step of B (where it can transmit a packet), plus a number of blocked steps. B' will maintain the following invariant. If t is the current step of B', then B is in some non-blocked step τ_t that corresponds to t. Suppose that the active items in the queue are $a_1 \lhd a_2 \lhd \cdots \lhd a_k$. Then the non-expired packets in the instance produced by B' will be the same, $a_1, a_2, \ldots, a_k$, where each item a_p is pending for B' if and only if packet a_p is pending for B. Further, for $p = 1, 2, \ldots, k$, the deadlines of these items satisfy the condition $d_{a_p} - d_{a_{p-1}} \geq (m+1)^{m-t+1}$, with d_{a_0} assumed to be equal to τ_t. This condition assures that there are sufficiently many time steps between packets' deadlines to simulate future insertions of items at arbitrary positions of the queue.

In this step t, B' now proceeds as follows: If B sends a packet a_q in step τ_t, B' will collect item a_q. It remains to show how to simulate queue updates. Suppose

that before step $t+1$ items $a_1, a_2, \ldots, a_s$ expire. Then B' will block for B all steps $\tau_t + 1, \tau_t + 2, \ldots, d_{a_s} - 1$ (that is, all steps until time d_{a_s}) and will set $\tau_{t+1} = d_{a_s}$. (We assume here that $s < k$, for otherwise the queue will be emptied and we can start the simulation from the beginning in the next time step.) Next, we need to show how to determine the deadlines of the newly released items. Consider some p, such that $s+1 \le p \le k-1$, and suppose that items $b_1 \lhd b_2 \lhd \cdots \lhd b_h$ are released between a_p and a_{p+1} (insertions before a_{s+1} and after a_k are handled in a similar manner). Then each corresponding packet b_g is assigned deadline $d_{b_g} = d_{a_p} + g(m+1)^{m-t}$. This way, the invariant at step $t+1$ will be preserved, and B' can continue its simulation of B.

The gain of B' is equal to the gain of B, and the optimum solution of the instance of 2nd-tier buffer management produced by B' is the same as the optimum of the original instance of dynamic queue. Thus the competitive ratios of B and B' are the same.

3 Intuitions

In this section we describe some basic intuitions behind online algorithms presented in this paper. We start by observing that GREEDY, which always collects the maximum value item, is 2-competitive. Indeed, to see why, partition the items collected by the adversary into two types: (1) those collected by GREEDY and (2) all other items. Obviously, the total weight of type-1 items does not exceed GREEDY's gain. If the adversary collects a type-2 item x at time t, then x is also pending for GREEDY at time t, so at this time GREEDY collects an item at least as heavy as x. Thus, the total weight of type-2 items also does not exceed GREEDY's gain.

The competitive ratio of GREEDY is in fact not better than 2: issue two items, one of weight $1-\epsilon$ and one of weight 1, in this order, and after the first step delete the first item. GREEDY will only collect the item of weight 1, while the optimum solution collects both items with total gain $2-\epsilon$. For $\epsilon \to 0$, the competitive ratio tends to 2.

To prevent the adversary from collecting twice as many items as the algorithm, one needs to give higher priority to urgent items, those that are near the front of the queue. Simply collecting the most urgent item at each step will not work; it is very easy to see that this strategy is not competitive at all. A natural compromise would be to collect the earliest item of sufficiently large weight, say weight at least βw_h, where h is the heaviest pending item and $\beta \in (0,1)$ is a some suitably chosen constant. This method, however, is not better than 2-competitive. To see this, choose some small positive $\epsilon < \beta$ and consider an adversary strategy where we issue n items of weight $\beta - \epsilon$, followed by n items of weight β, and then one item of weight 1. After n steps, we delete the first n items. The algorithm's gain is $n\beta + 1$, while the optimum gain is $n(2\beta - \epsilon) + 1$. For large n and $\epsilon \to 0$, the competitive ratio tends to 2. Note that this lower bound holds even for FIFO queues, as all items were inserted at the beginning.

The strategy above leads to the crucial insight: once an algorithm collects the earliest item of weight at least βw_h, it needs to remember that item h was already "used" (as an estimate of the gain for this step). Our two algorithms—PRUDENTMARK for general queues and EFH for FIFO queues, deal with this issue in two different ways.

PRUDENTMARK marks items used to estimate maximum gains. At each step it chooses the most urgent item whose weight is at least βw_m, where m is the heaviest unmarked item, and then it marks m. However, if m is too light, that is if $w_m < \alpha w_h$, for some constant α, then PRUDENTMARK simply collects h. (This option is needed to counteract an adversary strategy similar to the one described earlier in this section.) The analysis of PRUDENTMARK is presented in the next section.

The approach used in EFH for FIFO queues is different. The basic (although still not quite correct) idea is to simply alternate collecting the earliest item with weight at least βw_h and the heaviest item h itself. This formulation is ambiguous: due to an update, the heaviest item may change, so in the second option do we collect the heaviest item from the previous step or the current step? It is easy to show that neither of these two methods leads to an algorithm with ratio better than 2 (we leave this as an exercise). EFH is obtained by carefully refining this strategy. It works in stages consisting of up to three steps: it first collects the earliest item of weight at least βw_h, then the earliest item of weight at least ξw_h, for some $\xi > \beta$, and then h itself, with h referring to the item that had maximum weight when the stage started. However, if EFH detects a considerable change in the state of the queue, including a significant change in the weight of the heaviest item, it interrupts the stage and starts over. See Sect. 5 for a complete description of EFH and its analysis.

4 General Queues

In the previous section we showed that GREEDY achieves competitive ratio of 2. We now show how to improve this ratio, presenting an online algorithm PRUDENTMARK with competitive ratio ≈ 1.897. Later, we show that our analysis is nearly tight, i.e., that the competitive ratio of PRUDENTMARK is at least 1.894. For simplicity, we assume that there are always pending items, for otherwise we can insert any number of items of zero weight into the queue, without affecting the analysis.

Algorithm PRUDENTMARK Choose two parameters $\alpha, \beta \in (0, 1)$ whose values will be determined later. PRUDENTMARK follows the idea outlined in the previous section: it maintains marks on heavy items, using those items (roughly) to estimate the optimum. Specifically, at each step PRUDENTMARK proceeds as described below.

Algorithm PRUDENTMARK (one step)

```
/* update queue */
h ← the heaviest pending item
m ← the heaviest unmarked pending item
if w_m < α w_h then
    collect h
else
    mark m
    collect the earliest pending item e with w_e ≥ β w_m
```

4.1 Analysis of PRUDENTMARK

We choose β to be the unique root of $\beta^3 - 4\beta^2 + \beta + 1 = 0$ in the interval $[0, 1]$, i.e., $\beta \approx 0.7261$, and $\alpha = 2 - 1/\beta \approx 0.6228$. For these parameters, we prove that PRUDENTMARK is R-competitive, where $R = 1/\beta^2 \approx 1.8967$. The following relations are easy to verify and will be useful in the proof:

$$\beta + 1 \leq 1/\beta^2 \leq 2 = 1/\beta + \alpha \leq 2\beta + \alpha. \tag{1}$$

Types of Items Consider some arbitrary but fixed step of the computation. Recall that an item is called *active* if it is currently in the queue. When we say that an item is *pending* or *collected*, we mean that it is pending for or collected by PRUDENTMARK, respectively. In case of the adversary, we will always use complete phrases: "pending for the adversary" or "collected by the adversary".

We group some items into *protection pairs*. If (c, k) is a protection pair, then we say that k *protects* c; c and k are then called a *protected* and *protecting* item, respectively. Each protection pair (c, k) will satisfy the following *Pair Invariants* in each step:

(P1) c is collected by PRUDENTMARK and pending for the adversary;
(P2) k is marked, and it is either pending for PRUDENTMARK or is a protected item in some other pair;
(P3) $c \lhd k$ and $w_c \geq \beta w_k$;
(P4) there is no other protection pair where c is a protected item or k is a protecting item.

Pair Invariants imply that protection pairs form disjoint *protection chains*, where each item (except last) is protected by the next one. These chains have the following properties:

(CH1) All items in a chain except last are collected by PRUDENTMARK and pending for the adversary. The last item is pending for PRUDENTMARK and it may or may not be pending for the adversary.
(CH2) All items in a chain except first are marked. The first item may or may not be marked.

Items that are in protection pairs will be called *pair items* or *chain items*. Items that are pending for the adversary but are not in protection chains are called *solo items*. The current status of the item with respect to the algorithm (whether it is collected by or pending for PRUDENTMARK, marked or not) does not determine whether it is a chain item or a solo item. The protection chains will be determined in the process of the analysis.

Proof Idea The proof uses the amortized analysis framework described earlier in Sect. 2.1. Certain items will be assigned non-negative credits with value not exceeding their weight. Credits are used to decrease the adversary's amortized gain in steps when these items cease to be pending for him, in particular when he collects such an item. On the other hand, this increases the adversary's amortized gain in those steps when the credit is assigned to these items. The goal is to design a credit assignment strategy so as to minimize, over all types of steps, the ratio between the amortized

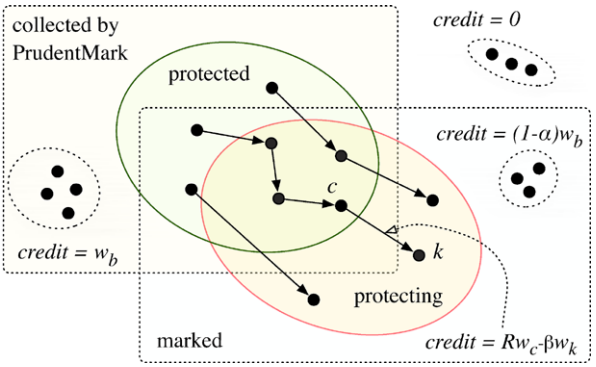

Fig. 1 Classification of items and their credits. Arrows represent protection pairs. With the possible exception of last items in protection chains, all items in the figure are pending for the adversary

adversary's gain and the gain of PRUDENTMARK. Below we explain the idea behind such a credit assignment.

To illustrate the main principle, consider a simple situation where we have an item b previously collected by PRUDENTMARK, pending for the adversary, and with weight much larger than the weight of the PRUDENTMARK's pending items. Then b already must have large credit on it, for otherwise, when the adversary collects b, we may not be able to afford to "pay" for b with the items that PRUDENTMARK still can collect. In fact, in our scheme, if b is pending for the adversary, collected by PRUDENTMARK, and with weight larger than w_h, then b will have full credit w_b. (For solo items, this will be explicit in the credit assignment rule (CR2) defined below; for chain items this fact will follow from the analysis.)

In general, if a solo item pending for the adversary is marked or collected, it will be assigned some credit, partial or full.

We will also assign credits to protection pairs. The basic idea is to create a pair (e, m) when PRUDENTMARK marks m and collects $e \neq m$ that remains pending for the adversary. When PRUDENTMARK's allowance of Rw_e is higher than the adversary's gain, the surplus is stored as a credit assigned to (e, m). If e belongs to a chain, this chain is now extended to m. Later, if the adversary collects a chain item, a starting section of the chain ending with this item will be destroyed and the credits associated with this section can be used to decrease the adversary's amortized gain or be redistributed among other items in this section.

We now discuss a step when a protection pair is created in a little more detail. Let z denote the item collected by the adversary (in this step), and h, e and m be defined as in the description of PRUDENTMARK, i.e., h is the heaviest pending item, m is the heaviest unmarked pending item, and e is the earliest pending item with $w_e \geq \beta w_m$. Assume that $w_m \geq \alpha w_h$, in which case PRUDENTMARK marks m and collects e.

As mentioned before, we will create protection pair (e, m) whenever e remains pending for the adversary; in particular, we have that $z \lhd e$. If z is either marked or collected, it already has a partial or full credit, which decreases the adversary's amortized gain. Let us then assume that z is unmarked and pending. This means that $w_z < \beta w_m$, for otherwise z would be collected instead of e. Then the adversary gains only $w_z < \beta w_m$, while PRUDENTMARK's allowance is $Rw_e \geq R\beta w_m \geq \beta w_m$. In this case the residual credit $Rw_e - \beta w_m$ will be assigned to the newly formed protection pair (e, m) for later use.

If $z \rhd m$, the adversary forfeits e and m (by the EEF property, they are not pending for the adversary anymore), so we do not need to assign any credit to them. Thus, his amortized gain is at most $w_z \le w_h \le w_m/\alpha \le w_e/(\alpha\beta)$, while PRUDENTMARK's gain is w_e. This immediately yields ratio $1/(\alpha\beta)$ in this case, which is, however, larger than 1.897. But we can improve it by dividing this case into two sub-cases: either $w_z \le w_m \le w_e/\beta$, which immediately gives a better ratio, or $w_m < w_z \le w_h$, in which case z is marked (by the choice of m) and has some credit, reducing the adversary's amortized gain and the competitive ratio.

Lastly, observe that in the intermediate case $e \trianglelefteq z \trianglelefteq m$ we do not need to assign any credit to e, because it is no longer pending for the adversary. Again, in this case we can argue that either z already had a partial credit assigned, if it had been marked or collected, or otherwise $w_z \le w_m \le w_e/\beta$.

Assignment of Credits We now formalize the idea above, by giving a complete specification of our credit assignment policy. We will maintain the following *Credit Invariants* at each step:

(CR1) A solo marked item b pending for PRUDENTMARK and for the adversary has credit at least $(1-\alpha)w_b$.
(CR2) A solo item b collected by PRUDENTMARK and pending for the adversary has full credit w_b (no matter whether it is marked or not).
(CR3) A protection pair (c,k) has credit at least $Rw_c - \beta w_k$. (Recall that c must be pending for the adversary.)

For all other items, their credits are non-negative. This assignment is illustrated in Fig. 1.

Credit Splitting Policy We emphasize that in (CR3) the credit is associated with the *pair* of items, not individual items. However, in some situations we will distribute their credits between their members; this could happen, for example, when a protection pair is destroyed. This credit distribution is done according to the following *Credit Splitting Policy*: Item k receives credit $(1-\alpha)w_k$ and c receives the remaining part of credit, i.e., $Rw_c - (1+\beta-\alpha)w_k$. Note also that if we destroy two consecutive pairs in a chain, then the middle item will receive double credit: as the protecting item in the first pair and as the protected item in the other.

Lemma 1 *The credits assigned by the Credit Splitting Policy are non-negative and satisfy* (CR1).

Proof Let us first check (CR1). Consider a protection pair (c,k), where k is pending for PRUDENTMARK. Then k's share of the credit, $(1-\alpha)w_k$, is exactly as specified by (CR1); thus, if k becomes a solo item, the Credit Splitting policy guarantees that it will receive a required amount of credit. Also, item c's credit share is non-negative, as, by Pair Invariant (P3), we have

$$Rw_c - (1+\beta-\alpha)w_k \ge (R\beta - 1 - \beta + \alpha)w_k = (1-\beta)w_k \ge 0,$$

which completes the proof. □

We remark that we do not always follow this Credit Splitting Policy: in some situations, we will destroy whole chains and distribute their total credit globally among its members.

Amortized Analysis From now on, fix one step, and let h, m and e be as described in PRUDENTMARK. Let z be the item collected by the adversary in this step. We divide each step into two phases called (Adv) and (Alg), with the first phase divided further into two sub-phases.

(Adv) In this phase, the adversary updates the queue and collects its item z. We divide it into two sub-phases, as follows:

  (Adv1) The adversary deletes some prefix of the queue, inserts new items into the queue, identifies z, and forfeits all items before z (these items are not considered pending for the adversary from now on).
  (Adv2) The adversary collects z.

(Alg) PRUDENTMARK executes its move, collecting an item.

In each phase, some items may change their status and their credits may be updated. The total change of credits associated with a phase or sub-phase named (S) will be called *(S)-credit*. We stress that these stand for the *change* of credits, and thus could be negative.

It is convenient to introduce a few more terms. When the adversary collects z, it gains w_z. The sum of w_z and the (Adv2)-credit is called *(Adv2)-gain*, while the sum of w_z and (Adv)-credit is called *(Adv)-gain*. Naturally, the adversary's amortized gain for this step is

$$\text{(adversary's amortized gain)} = \text{((Adv)-gain)} + \text{((Alg)-credit)}.$$

At this point, it is worth emphasizing that the minimum amount of credit assigned to an item, as required by Credit Invariants, is uniquely determined by its status with respect to PRUDENTMARK (whether it's collected or marked), with respect to the adversary's algorithm (whether it is pending for the adversary), as well as with respect to the analysis (whether it is a protected or protecting item). After each step, the status of the item changes according to the behavior of PRUDENTMARK and the adversary. These changes do not determine, however, the structure of protection chains after the move. As a matter of fact, the objective of the analysis below is to show how we can rearrange these chains so that the desired bounds hold.

We now start our analysis. First, we examine sub-phase (Adv1) and show that (Adv1)-credit is at most 0.

Lemma 2 *It is possible to modify the protection chains after sub-phase* (Adv1), *so that all Pair and Credit Invariants are preserved and* (*Adv*1)*-credit is at most* 0.

Proof First note that newly inserted items have zero credit, so they do not contribute to (Adv1)-credit. As for the items deleted or forfeited by the adversary in sub-phase (Adv1), we treat both in exactly the same manner, as they are all no longer pending

for the adversary. For each such solo item, we do not change its credit, thus keeping the Credit Invariants preserved. For any protection pair (c, k) in which c is deleted or forfeited in (Adv1), we destroy this pair and reassign credits according to the Credit Splitting Policy. This guarantees that (Adv1)-credit is at most 0. (Note that we could have reduced the credits of all deleted and forfeited items to 0; since all initial credits are non-negative, this would also imply the lemma.) □

We now deal with the whole phase (Adv). By the previous lemma, we can ignore the contribution of (Adv1)-credits. In fact, for simplicity, from now on we will simply assume that (Adv)-gain is equal to (Adv2)-gain. We now present two lemmas that estimate this quantity when z is pending and when z was collected by PRUDENTMARK respectively.

Lemma 3 *Assume that z is pending for* PRUDENTMARK. *It is possible to modify the protection chains after phase* (Adv), *so that all Pair and Credit Invariants are preserved and the following bounds hold*:

(i) *The* (*Adv*)*-gain is at most w_z.*
(ii) *If z is marked, then the* (*Adv*)*-gain is at most αw_z.*
(iii) *If $w_m < \alpha w_h$ (that is,* PRUDENTMARK *collects h in this step without marking m) and z is not marked, then the* (*Adv*)*-gain is at most αw_h.*

Proof We begin by observing that z is a solo item. Indeed, this follows from Pair Invariant (P1). Being pending for PRUDENTMARK, z cannot be a protected item. It also cannot be a protecting item, because, by the update in sub-phase (Adv1), it is the earliest item pending for the adversary, while each protecting item is preceded by the item that it protects, which is pending for the adversary.

We do not change any protection chains. Then the (Adv)-gain is equal to w_z plus the change of z's credit. Part (i) is then obvious, since z's credit does not increase. If z is marked, it already had credit at least $(1 - \alpha)w_z$, by (CR1), and we can now reduce it to 0 because z is not pending for the adversary anymore. So the (Adv)-gain is at most αw_z, yielding (ii). As for (iii), by (i), the (Adv)-gain is at most $w_z \leq w_m < \alpha w_h$, where the first inequality follows from the fact that z is pending and not marked. □

Lemma 4 *Assume that z was collected by* PRUDENTMARK. *It is possible to modify the protection chains after phase* (Adv), *so that all Pair and Credit Invariants are preserved and the following bounds hold*:

(i) *If z is unprotected then the* (*Adv*)*-gain is at most* 0.
(ii) *If z is protected by an item k then the* (*Adv*)*-gain is at most $(1 + \beta - \alpha)w_k - (R - 1)w_z$.*
(iii) *In either case, the* (*Adv*)*-gain can be upper-bounded by any of the following quantities: $\alpha\beta w_k$, αw_z, $\alpha\beta w_h$.*
(iv) *If* PRUDENTMARK *marks m in this step then* (*Adv*)*-gain is at most βw_m.*

Proof As in the previous lemma, we do not change any chains that do not contain z. So it is sufficient to consider only the contributions of z and the protection chain that contains z, if any.

First, assume that z is unprotected. As in the proof of Lemma 3, we observe that z cannot be a protecting item, since protecting items are preceded by at least one item pending for the adversary. Thus, z must be a solo item. As such, it has full credit, by invariant (CR1). So we can lower its credit to 0, and (i) is trivial.

By (i), claims (iii)–(iv) are trivial when z is unprotected. In the remainder of the proof we show that (ii)–(iv) hold when z is protected by some item k. As all items before z are forfeited, z must be the first item in its chain. We break the analysis into two cases: for $w_k \le w_h$ and $w_k > w_h$.

Case 1: $w_k \le w_h$. We destroy the pair (z, k) and distribute its credits according to the Credit Splitting Policy. If k is protected, its share of the credit can be discarded; if k is pending, then it becomes a solo marked item and its share satisfies invariant (CR1), by Lemma 1. As for z, its share of the pair's credit is $Rw_z - (1+\beta-\alpha)w_k$. This is non-negative by Lemma 1 and we can now reduce z's credit to zero because z is not pending for the adversary anymore. Therefore, overall, the (Adv)-gain is at most

$$w_z - [Rw_z - (1+\beta-\alpha)w_k] = (1+\beta-\alpha)w_k - (R-1)w_z,$$

and claim (ii) follows. Since $w_z \ge \beta w_k$, we can bound the last expression by $(1+\beta-\alpha-(R-1)\beta)w_k = \alpha\beta w_k$. The remaining bounds in (iii) follow from $w_k \le w_z/\beta$, $w_k \le w_h$. The bound in (iv) follows from (iii) and $w_h \le w_m/\alpha$.

Case 2: $w_k > w_h$. By the definition of h, item k cannot be pending, and thus by Pair Invariant (P2), k is also protected. Let the protection chain starting at z be $(z = k_0, k = k_1), (k_1, k_2), \ldots, (k_{s-1}, k_s)$. Using (CH1), k_s must be a pending marked item, so, by the definition of h, $w_{k_s} \le w_h$.

To get the desired bound, we change the status and credits of all items in the chain, by destroying the whole chain. Unlike before though, we do not apply the Credit Splitting Policy; instead, we distribute the credits globally among the chain items. All items from the chain become solo items, and we redistribute credits as follows: items $k_1, \ldots, k_{s-1}$ will be assigned full credit and item k_s will receive credit $(1-\alpha)w_{k_s}$. The adversary gain is $w_z = w_{k_0}$ and the new credit on z will be 0. This modification preserves Credit Invariants. Using the first inequality in (1) and $w_{k_s} \le w_h$, the (Adv)-gain is at most

$$\begin{aligned}
&w_{k_0} + \left[\sum_{i=1}^{s-1} w_{k_i} + (1-\alpha)w_{k_s} - \sum_{i=0}^{s-1}(Rw_{k_i} - \beta w_{k_{i+1}})\right] \\
&= (1+\beta-\alpha)w_{k_s} - (R-1)w_z - \sum_{i=1}^{s-1}(R-1-\beta)w_{k_i} \\
&\le (1+\beta-\alpha)w_{k_s} - (R-1)w_z \\
&\le (1+\beta-\alpha)w_h - (R-1)w_z.
\end{aligned}$$

Now, (ii) follows from $w_h \le w_k$.

Since $w_z \geq \beta w_k$ and $w_k \geq w_h$, using the derivation above, we can also bound the (Adv)-gain by

$$(1+\beta-\alpha)w_h - (R-1)w_z \leq (1+\beta-\alpha-(R-1)\beta)w_h = \alpha\beta w_h \leq \alpha\beta w_k.$$

The other bounds in (iii) and (iv) follow as in the previous case, completing the proof. □

Having established bounds on the (Adv)-gain, we now estimate the total amortized gain in a step and show that it is at most R times PRUDENTMARK's gain. When estimating the (Alg)-credit in phase (Alg), we assume that the protection chains have already been readjusted to take into account that z and all items before it are not pending for the adversary any more. Again, each Lemma covers a distinct case: when PRUDENTMARK marks m and collects e, and when PRUDENTMARK collects h without marking any item.

Lemma 5 *Suppose that $w_m \geq \alpha w_h$, that is* PRUDENTMARK *marks m. It is possible to modify the protection chains after the step, so that all Pair and Credit Invariants are preserved and the adversary's amortized gain is at most R times the gain of* PRUDENTMARK.

Proof According to the lemma's assumptions, PRUDENTMARK collects e, so we need to bound the adversary's amortized gain by Rw_e. Recall that the adversary's amortized gain is the sum of the (Adv)-gain, that we already estimated in Lemma 3 and Lemma 4, and (Alg)-credit that we still need to estimate. We break the analysis into two cases, depending on whether $z \lhd e$ or not.

Case 1: $z \lhd e$. We claim first that the (Adv)-gain is at most βw_m. Indeed, if z is pending for PRUDENTMARK, then $w_z < \beta w_m$, thus, by Lemma 3, the (Adv)-gain is at most βw_m. If z is a collected item, then, by Lemma 4, the (Adv)-gain is at most βw_m as well.

Next, we need to estimate the (Alg)-credit. As e is pending for PRUDENTMARK, it is either the last item in a protection chain or is a solo item. We consider two cases, depending on whether e is pending for the adversary or not.

Case 1.1: e is pending for the adversary. We have two further sub-cases.

Case 1.1.1: $e = m$. Then, e is not in any protection chain (because it was not marked). We do not create a protection pair in this case, and hence e becomes a solo collected item with full credit w_e, as required by Credit Invariant (CR2). Adding the βw_m bound on the (Adv)-gain, we get that the amortized adversary's gain is at most $w_e + \beta w_e \leq Rw_e$.

Case 1.1.2: $e \lhd m$. In this case we create a protection pair (e, m). If e was already a protecting (marked) item, then the chain ending at e will now be extended to m. As $w_e \geq \beta w_m$, Pair Invariant (P2) is preserved. The pair (e, m) receives credit $Rw_e - \beta w_m$, which satisfies (CR3). As the (Adv)-gain is bounded by βw_m, the adversary's amortized gain is at most Rw_e.

Case 1.2: e has been collected by the adversary. Intuitively, in this sub-case we should have some slack, since we do not need to assign any credit to e. Indeed, m becomes now a solo item and we only need to assign credit of at most $(1-\alpha)w_m$ to it. As explained earlier, the (Adv)-gain is at most βw_m. Thus, the adversary's amortized gain is at most

$$\beta w_m + (1-\alpha)w_m \le (1+\beta-\alpha)w_e/\beta = (R+1-1/\beta)w_e \le Rw_e,$$

as needed.

Case 2: $e \trianglelefteq z$. In this case, e is not pending for the adversary anymore. We begin with estimating the (Adv)-gain. Suppose first that z is pending. By Lemma 3, if $w_z \le w_m$, the (Adv)-gain is at most $w_z \le w_m$. If $w_z > w_m$, then z must be marked and $w_z \le w_h \le w_m/\alpha$, so, by Lemma 3, the (Adv)-gain is at most $\alpha w_z \le w_m$. On the other hand, if z is collected, then, by Lemma 4, the (Adv)-gain is at most $\beta w_m \le w_m$. Thus, in all cases, the (Adv)-gain is at most w_m.

Next, we estimate the (Alg)-credit. Since e is not pending for the adversary anymore, we do not need to give the adversary any credit for e. If m is still pending for the adversary (and thus $z \triangleleft m$), it becomes a solo marked item with credit $(1-\alpha)w_m$, and otherwise we do not give m any credit.

Overall, the adversary's amortized gain is at most

$$w_m + (1-\alpha)w_m = (2-\alpha)w_m \le (2-\alpha)w_e/\beta = Rw_e,$$

as needed. □

Lemma 6 *Suppose that $w_m < \alpha w_h$, that is* PRUDENTMARK *collects h without marking an item. It is possible to modify the protection chains after the step, so that all Pair and Credit Invariants are preserved and the adversary's amortized gain is at most R times the gain of* PRUDENTMARK.

Proof We need to show that the adversary's amortized gain is at most Rw_h.

We first estimate the (Adv)-gain. If z is a pending marked item, then the (Adv)-gain is at most $\alpha w_z \le \alpha w_h$, by Lemma 3(ii). If z is pending and not marked, then the (Adv)-gain is at most αw_h by Lemma 3(iii). If z is a collected item, then, by Lemma 4, the (Adv)-gain is at most $\alpha\beta w_h \le \alpha w_h$. We conclude that in all cases the (Adv)-gain is at most αw_h.

We now estimate the (Alg)-credit and the overall adversary's amortized gain. By the lemma assumption, h is pending and marked, and is therefore either a solo item or the last item in a chain. We distinguish these two cases.

Case 1: h is a solo item. Since h is marked, its credit is at least $(1-\alpha)w_h$, by (CR1). Thus, the credit increase for h, if any, will be bounded by αw_h. Adding the (Adv)-gain, we conclude that the amortized adversary's gain is at most $2\alpha w_h \le Rw_h$.

Case 2: h is the last item in a protection chain. Let the chain containing h be (b_0, b_1), $(b_1, b_2), \ldots, (b_{p-1} = b, b_p = h)$, where b_0 is a non-protecting item. In this case, we simply destroy the whole chain, turning items $b_0, b_1, \ldots, b_p$ into solo collected items

and giving each full credit. Using Pair Invariant (P3), we have $w_{b_i} \geq \beta^{p-i} w_h$ for all i. Thus, the (Alg)-credit will be at most

$$\begin{aligned}
&\sum_{i=0}^{p} w_{b_i} - \sum_{i=0}^{p-1} \left(R w_{b_i} - \beta w_{b_{i+1}}\right) \\
&\quad = (1+\beta) w_h - (R-1-\beta) \sum_{i=1}^{p-1} w_{b_i} - (R-1) w_{b_0} \\
&\quad \leq (1+\beta) w_h - (R-1-\beta) \sum_{i=1}^{p-1} \beta^i w_h - (R-1)\beta^p w_h \\
&\quad = w_h \left(1 + \beta - (R-1) \sum_{i=1}^{p} \beta^i + \sum_{i=2}^{p} \beta^i\right) \\
&\quad = w_h \left(1 + \beta - (R-1) \frac{\beta - \beta^{p+1}}{1-\beta} + \frac{\beta^2 - \beta^{p+1}}{1-\beta}\right) \\
&\quad = \frac{w_h}{1-\beta} \left(1 + \beta - \beta R - \beta^{p+1}(2-R)\right) \\
&\quad \leq \frac{1+\beta-\beta R}{1-\beta} w_h.
\end{aligned}$$

Adding the αw_h bound on the (Adv)-gain, substituting $\alpha = 2 - 1/\beta$ and $R = 1/\beta^2$, and using the definition of β, we conclude that the amortized adversary's gain in this case is at most

$$\begin{aligned}
\left(\alpha + \frac{1+\beta-\beta R}{1-\beta}\right) w_h &= \frac{-\beta^2 + 4\beta - 2}{\beta(1-\beta)} \cdot w_h \\
&= \frac{1}{\beta^2} \cdot w_h = R w_h,
\end{aligned}$$

completing the proof. □

By adding up the bounds from Lemma 5 and Lemma 6, over all steps, and amortizing, we obtain

$$\text{(adversary's total gain)} + \text{(final total credit)} \leq R \cdot (\text{PRUDENTMARK's total gain}),$$

and the R-competitiveness of PRUDENTMARK follows, because the credits are non-negative.

Recall that we chose β to be the unique root of $\beta^3 - 4\beta^2 + \beta + 1 = 0$ in the interval $[0, 1]$, i.e., $\beta \approx 0.7261$, and $\alpha = 2 - 1/\beta \approx 0.6228$. Summarizing, we obtain our main result:

Theorem 1 *For constants α and β defined above,* PRUDENTMARK *is R-competitive, where $R = 1/\beta^2 \approx 1.897$.*

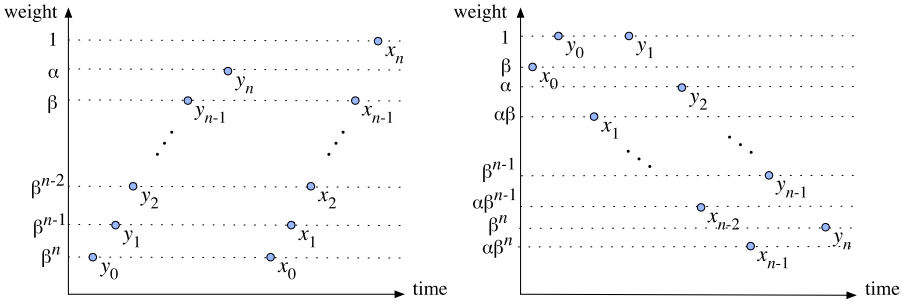

Fig. 2 The instances I_1 (*left*) and I_2 (*right*) used in the lower bound

4.2 Lower Bound for PRUDENTMARK

In this section, we show that the analysis of PRUDENTMARK is nearly tight: for any choice of $\alpha, \beta \in [0, 1]$, the competitive ratio of PRUDENTMARK is at least 1.894. We can assume that $\alpha, \beta \in (0, 1)$, since for $\alpha \in \{0, 1\}$ or $\beta \in \{0, 1\}$ it is easy to show that the competitive ratio is at least 2, using the same or similar instances to those in Sect. 3.

The proof is by presenting three instances, so that for any choice of $\alpha, \beta \in (0, 1)$, the PRUDENTMARK's ratio on at least one of these instances will exceed 1.894.

Lemma 7 *For any integer $n \geq 1$ there exists an instance I_1 with $2n + 2$ items on which the competitive ratio of* PRUDENTMARK *is at least $R_1(\alpha, \beta) = 1 + \alpha + \beta - \alpha\beta - O(\beta^n)$.*

Proof Fix some very small $\epsilon > 0$, say $\epsilon = (\alpha\beta)^n/n$. Instance I_1 has $2n + 2$ items $y_0 \lhd y_1 \lhd \cdots \lhd y_n \lhd x_0 \lhd x_1 \lhd \cdots \lhd x_n$, with weights $w_{x_i} = \beta^{n-i}$ for $i = 0, 1, \ldots, n$, $w_{y_i} = \beta^{n-i} - \epsilon$ for $i = 0, 1, \ldots, n-1$, and $w_{y_n} = \alpha - \epsilon$. The instance is depicted in Fig. 2. (In this figure we have $\alpha \geq \beta$, but the proof works for any choice of α and β.)

At step 1 the adversary releases items y_0, x_0 and x_1. At each step $i = 2, 3, \ldots, n$, the adversary releases items y_{i-1} and x_i. In each step $i = 1, 2, \ldots, n$, the PRUDENTMARK marks x_i and collects x_{i-1}, (indeed, at this time all items before x_{i-1} are lighter than $\beta w_{x_i} = \beta^{n-i+1}$), and the adversary collects y_{i-1}. After step n, PRUDENTMARK has collected x_0, marked and collected $x_1, x_2, \ldots, x_{n-1}$ and marked x_n. The adversary has collected $y_0, y_1, \ldots, y_{n-1}$. Before step $n + 1$ all items from y_0 to y_{n-1} are deleted and item y_n is released. Since $w_{y_n} < \alpha w_{x_n}$, PRUDENTMARK collects x_n. The adversary collects y_n and deletes it. After this step, PRUDENTMARK has no more items to collect, while the adversary can now collect all the items $x_0, x_1, \ldots, x_n$. The total gain of PRUDENTMARK is $\sum_{i=0}^{n} x_i = \sum_{i=0}^{n} \beta^i \leq 1/(1-\beta)$. The total gain of the adversary is $\sum_{i=0}^{n}(x_i + y_i) = \sum_{i=0}^{n} \beta^i + \sum_{i=1}^{n}(\beta^i - \epsilon) + \alpha - \epsilon = (1+\beta)/(1-\beta) + \alpha - O(\beta^n)$. Therefore, the competitive ratio is at least $R_1(\alpha, \beta)$. □

Lemma 8 *For any integer $n \geq 1$ there exists an instance I_2 with $2n + 1$ items on which the competitive ratio of* PRUDENTMARK *is at least $R_2(\alpha, \beta) = (2 - \alpha)/\beta - O(\alpha^n)$.*

Proof We choose ϵ as in the proof of Lemma 7. Instance I_2 has $2n+1$ items $x_0 \lhd y_0 \lhd x_1 \lhd y_1 \lhd \cdots \lhd x_{n-1} \lhd y_{n-1} \lhd y_n$, where the weights are $w_{x_i} = \alpha^i \beta$ for $i = 0, 1, \ldots, n-1$, $w_{y_i} = \alpha^{i-1}$ for $i = 1, 2, \ldots, n$, and $w_{y_0} = 1 - \epsilon$. Items $y_0, y_1, \ldots, y_n$ and x_0 are released at the beginning. The instance is depicted in Fig. 2. (In this figure we have $\alpha \leq \beta$, but the proof works for any choice of α and β.)

The adversary maintains the invariant that right before step $i = 1, 2, \ldots, n$ the active items are $x_{i-1}, y_{i-1}, y_i, \ldots, y_n$. To accomplish this, in each step $i < n$, he collects y_{i-1}, and right after step i, he deletes y_{i-1} and x_{i-1} and releases x_i. (After step n he deletes both y_{n-1} and y_n.) This way in each step $i = 1, 2, \ldots, n$ the algorithm will mark y_i (because $w_{y_i} \geq \alpha w_{y_{i-1}}$) and collect x_{i-1}. The PRUDENTMARK's total gain is $\sum_{i=0}^{n-1} w_{x_i} = \sum_{i=0}^{n-1} \beta\alpha^i \leq \beta/(1-\alpha)$. The total adversary gain is $\sum_{i=0}^{n-1} w_{y_i} = 1 - \epsilon + \sum_{i=0}^{n-2} \alpha^i = (2-\alpha)/(1-\alpha) - O(\alpha^n)$. Therefore the competitive ratio is at least $R_2(\alpha, \beta) = (2-\alpha)/\beta - O(\alpha^n)$. □

Theorem 2 *The competitive ratio of* PRUDENTMARK *is at least* 1.894.

Proof If $\beta \leq \frac{1}{2}$, the adversary can issue two items, $x \lhd y$, with weights $w_x = \beta$ and $w_y = 1$. PRUDENTMARK will collect x, while the adversary can collect y and delete both items. For this instance, the competitive ratio is $1/\beta \geq 2$.

Thus now we can assume that $\beta > \frac{1}{2}$. For any α and β, the competitive ratio of PRUDENTMARK is at least $R(\alpha, \beta) = \max\{R_1(\alpha, \beta), R_2(\alpha, \beta)\}$, by Lemma 7 and Lemma 8. For $n \to \infty$, $R(\alpha, \beta)$ converges to

$$R^*(\alpha, \beta) = \max\{1 + \alpha + \beta - \alpha\beta, (2-\alpha)/\beta\}.$$

For any fixed β, the first quantity increases with α whereas the second one decreases. Substituting $\alpha = 0$ into the two quantities, we have $1 + \beta < 2/\beta$, while for $\alpha = 1$ we get $2 > 1/\beta$ (because $\beta > \frac{1}{2}$). Therefore the minimum of $R^*(\alpha, \beta)$ will be attained when the quantities in the maximum are equal. Solving this equation, we get $\alpha = (2 - \beta - \beta^2)/(1 + \beta - \beta^2)$, and plugging it into the formula above, we get the competitive ratio $(3-\beta)/(1+\beta-\beta^2)$. By routine calculus, this expression is minimized for $\beta = 3 - \sqrt{5}$, yielding the lower bound of $1 + 2/\sqrt{5} \approx 1.8944$ on the competitive ratio. □

5 FIFO Queues

In this section, we present an online algorithm EFH that is 1.737-competitive for FIFO queues. Recall that in dynamic FIFO queues items can be inserted only at the end of the queue. EFH follows the basic idea described in Sect. 3.

Algorithm EFH The execution of EFH is divided into stages, where each stage is a single step, a pair of consecutive steps, or a triple of consecutive steps. EFH uses parameters $\alpha, \beta, \xi \in [0, 1]$, with $\beta \leq \xi$, whose values we specify later.

Algorithm EFH (one stage)
(E) $h \leftarrow$ the heaviest pending item
collect the earliest pending item e with $w_e \geq \beta w_h$
(U1) */* update queue */*
(F) $h' \leftarrow$ the heaviest pending item
if an item of weight $\geq \xi w_h$ pending in (E) was deleted in (U1)
or h is not pending **or** $\alpha w_{h'} > w_h$
then end this stage
collect the earliest pending item f with $w_f \geq \xi w_h$
(U2) */* update queue */*
(H) $h'' \leftarrow$ the heaviest pending item
if h is not pending **or** $\alpha w_{h''} > w_h$ **then** end this stage
collect h
(U3) */* update queue */*
end this stage

Two clarifications are necessary. First, we assume that a stage starts right after a queue update and before EFH (and, in the analysis, the adversary) makes its move. Thus the very first update of the computation, consisting only of insertions, is not part of any stage. Second, we assume that all items named in the algorithm actually exist; otherwise, as explained before, we can add a sufficient number of items of zero weight to the instance.

5.1 Analysis of EFH

We show that if we choose appropriate values of parameters, i.e., $\alpha = \frac{3}{4} = 0.75$, $\beta = (\sqrt{13}+1)/8 \approx 0.576$, and $\xi = \frac{4}{3}\beta = (\sqrt{13}+1)/6 \approx 0.768$, then the competitive ratio of EFH is $R = \frac{1}{\beta} = 2(\sqrt{13}-1)/3 \approx 1.737$.

Credits Our proof is based on amortized analysis, as outlined in Sect. 2. We associate non-negative credits with items. Items pending for the adversary will be assigned credit when they are collected by EFH. In most cases, each such item a will be given full credit w_a; this way, when the adversary collects a later, its amortized gain will be zero. There is one exception to this rule, which we describe shortly.

Suppose that a stage s just ended, and let e and h be the items from this stage, as defined in EFH. Item e is called *special* in stage $s+1$, if

(i) h is pending when stage $s+1$ starts, and
(ii) in stage s, either $\alpha w_{h'} > w_h$ was true in (F) or $\alpha w_{h''} > w_h$ was true in (H).

The two conditions say, in essence, that stage s ended (having lasted for one or two steps) because a very heavy item was released in (U1) or (U2).

Right after stage s, our credit assignment strategy will satisfy the following invariant:

(CR) Suppose that a is an item pending for the adversary and collected by EFH. If a is not special then a has *full credit* equal to w_a. If $a = e$ is special then a has *partial credit* equal to $w_e - \frac{1}{3}\beta w_h$. (Note that this credit is non-negative, as $w_e \geq \beta w_h$.) Other items collected by EFH have non-negative credits. Items pending for EFH have credit zero.

Observe that Invariant (CR) holds vacuously before the first stage, that is before the computation starts.

Amortized Gain Define the *adversary's amortized gain* in a stage as the sum of weights of the items he collected in this stage plus the total credit change in the stage. Our objective is to show that in each stage we have

$$\text{(adversary's amortized gain)} \leq R \cdot \text{(EFH's gain)}. \tag{2}$$

The initial credits are 0, while final credits are non-negative. Thus, by standard amortization argument, (2) will imply R-competitiveness of EFH.

Intuitively, one may view our analysis as the following process. When EFH collects an item a, we use the allowance of Rw_a to pay for the item collected by the adversary in this step, as well as pre-pay for a, in case the adversary collects a later. In some cases this pre-payment is not necessary, namely when a is not pending for the adversary anymore. Another unusual situation is when $a = e$ becomes the special item, as in that case we may not afford to fully pre-pay for a and it is assigned only a partial credit. Fortunately, as it turns out, in the next stage EFH is guaranteed to gain enough to increase this credit to full.

We take advantage of the FIFO property in the following way: After nominating h in the stage, we are guaranteed that no new items will appear before h. Thus, all the items collected by the adversary in this stage that are before h were either pending for EFH already at (E) or had credit on them. On the other hand, all new items, possibly including h' and h'', are after h, and thus if the adversary collects them, then the items collected by EFH are no longer pending for him (by EEF property), and we do not need to give him credit for these items.

Adversary's Adjusted Gains By x, y, z we will denote, respectively, the first, the second, and the third item collected by the adversary in the stage, if defined. By the FIFO assumption and the EEF property, the relations $x \lhd y \lhd z$ hold (for the items that are defined). When the adversary collects an item $a \in \{x, y, z\}$, he gains the difference between w_a and the credit on a, and this quantity will be called the *a-gain*. Obviously, the a-gain is at most w_a.

In the proof, to facilitate analysis, it will be convenient to use a modified definition of the a-gain. Intuitively, we will always include the cost of increasing the partial credit on the special item in the x-gain, even if this special item is one of y or z. We will modify the a-gains for $a \in \{y, z\}$ accordingly, i.e., excluding the cost of increasing their partial credit from the respective gain. This modified quantity will be called the *adjusted a-gain*, for $a \in \{x, y, z\}$.

Denote by $\hat{e}$ the item collected by EFH in step (E) of the *previous* stage and by $\hat{h}$ the corresponding heaviest item. (For the first stage, assume that $\hat{e}$ and $\hat{h}$ are

items of weight 0.) According to invariant (CR), when the stage starts $\hat{e}$ may have an associated partial credit $w_{\hat{e}} - \frac{1}{3}\beta w_{\hat{h}}$. However, after the stage $\hat{e}$ is not special anymore, so, if $\hat{e}$ is still pending for the adversary, its credit should be increased to $w_{\hat{e}}$ to preserve (CR). Another possibility is that $\hat{e} \in \{x, y, z\}$, in which case the missing part of $\hat{e}$'s credit, equal to $\frac{1}{3}\beta w_{\hat{h}}$, will also count towards the amortized gain.

Formally, we define the *adversary's adjusted x-gain* as follows:

- If $\hat{e} \trianglelefteq x$ then the adjusted x-gain is equal to the x-gain. (This is because the adversary collects or forfeits $\hat{e}$, so we don't need to increase its credit.)
- If $\hat{e}$ has full credit then the adjusted x-gain is equal to the x-gain.
- Otherwise, we have that $\hat{e} \triangleright x$ and $\hat{e}$ must be a special item with partial credit (because $\hat{e} \triangleright x$ implies that $\hat{e}$ is pending for the adversary when the stage starts). In this case, the adjusted x-gain is equal to the x-gain plus $\frac{1}{3}\beta w_{\hat{h}}$.

The quantity $\frac{1}{3}\beta w_{\hat{h}}$ in the last case is the increase of $\hat{e}$'s credit to full. Note that this guarantees that $\hat{e}$ will have full credit if it's still pending after this stage, as required by (CR). (In some cases this increase might be unnecessarily generous; for example when $x \triangleleft \hat{e} \triangleleft y$ we will increase $\hat{e}$'s credit to full even though $\hat{e}$ will not be pending for the adversary after this stage.)

For $a \in \{y, z\}$, the adjusted a-gain is defined as follows: if a is already collected by EFH (before this stage), then the adjusted a-gain is 0, otherwise it is w_a. Thus for $a \in \{y, z\}$ the adjusted a-gain is almost always the same as the a-gain; the only exception is when $a = \hat{e}$ and $\hat{e}$ has partial credit, in which case we do not include the remaining part of a's credit in the adjusted a-gain (since it is already accounted for in the adjusted x-gain).

In this terminology, the adversary's amortized gain in the stage can be bounded from above by the sum of the following quantities:

- adjusted x-gain,
- adjusted y-gain and adjusted z-gain, if they are defined,
- credit on e, either partial or full, depending on whether e becomes special or not, if e is pending for the adversary after the stage,
- full credits on f and h, if they are collected and remain pending for the adversary after the stage.

Lemma 9

(a) *For $a \in \{y, z\}$, the adjusted a-gain is* 0, *if a was collected before the phase started, otherwise it is w_a.*
(b) *The adjusted x-gain is at most* $\max(\beta w_h, \min(w_x, w_h))$.

Proof Part (a) is trivial, being simply a restatement of the definition. To prove (b), we consider three cases, according to the definition of the adjusted x-gain.

Case 1: $\hat{e} \trianglelefteq x$. In this case the adjusted x-gain is equal to the x-gain. Suppose first that $x \neq \hat{e}$. Then either $w_x > w_h$, in which case x has full credit (because it cannot be pending) and the x-gain is 0; or else $w_x \leq w_h$, in which case the x-gain is at most $w_x = \min(w_x, w_h)$.

The second sub-case is when $x = \hat{e}$. Since x is not pending for EFH, x has either full or partial credit. If x has full credit, then the x-gain is 0. If x has partial credit, then the x-gain is $\frac{1}{3}\beta w_{\hat{h}}$ and $x = \hat{e}$ is special. The definition of special items implies that the previous stage ended after an item of weight at least $w_{\hat{h}}/\alpha$ appeared, so we have $\alpha w_h \geq w_{\hat{h}}$. Thus the x-gain is at most $\frac{1}{3}\beta w_{\hat{h}} \leq \frac{1}{3}\alpha\beta w_h \leq \beta w_h$.

Case 2: $\hat{e}$ has full credit when the stage starts. The argument is similar to the first subcase of Case 1. The adjusted x-gain is equal to the x-gain. Also, either $w_x > w_h$, in which case x has full credit (because it cannot be pending) and the x-gain is 0; or else $w_x \leq w_h$, in which case the x-gain is at most $w_x = \min(w_x, w_h)$.

Case 3: $\hat{e} \rhd x$ and $\hat{e}$ is special with partial credit. In this case, the adjusted x-gain is equal to the x-gain plus $\frac{1}{3}\beta w_{\hat{h}}$. The definition of special items implies that $\alpha w_h \geq w_{\hat{h}}$. As $x \lhd \hat{e}$, either x has full credit or it was pending in (E) of the previous stage, by the FIFO property. If x has full credit, then the adjusted x-gain is $\frac{1}{3}\beta w_{\hat{h}}$. On the other hand, if x was pending in (E) in the previous stage, then, by the code of EFH, we have $w_x < \beta w_{\hat{h}}$ and the adjusted x-gain is at most $\beta w_{\hat{h}} + \frac{1}{3}\beta w_{\hat{h}} = \frac{4}{3}\beta w_{\hat{h}}$. Thus, in both cases, the adjusted x-gain is at most $\frac{4}{3}\beta w_{\hat{h}} \leq \frac{4}{3}\alpha\beta w_h = \beta w_h$. □

Amortized Analysis We examine three disjoint cases, depending on the number of steps in a stage. Within each case, we examine the reason for the stage to end (h' or h'' is very heavy, h was collected or deleted, or some other item of weight at least ξw_h was deleted) and the possible behavior of the adversary—namely the position of his $\rhd$-maximal item collected in this stage relative to the items collected by EFH (as this determines the credit on items collected by EFH). In each of those sub-cases, we show that the amortized gain of the adversary does not exceed R times the gain of EFH.

To simplify the proofs, in the rest of this section we will assume that $w_h > 0$, in which case the gain of EFH is non-zero, so we can rewrite (2) in the form

$$\frac{\text{adversary's amortized gain}}{\text{EFH's gain}} \leq R. \tag{3}$$

All arguments below easily extend to $w_h = 0$, but then one needs to work directly with (2) to avoid division by zero.

In many cases of the analysis we encounter a situation, in which the bounds on both the adversary's amortized gain and EFH's gain contain w_e (with coefficient 1). Then, as $w_e \geq \beta w_h$, replacing w_e in both gains by βw_h can only increase the estimate on the competitive ratio, because

$$\frac{w_e + A}{w_e + B} \leq \frac{\beta w_h + A}{\beta w_h + B},$$

for $A \geq B$—which will be the case in our analysis. A similar argument applies to w_f, which can be replaced by ξw_h.

We introduce now yet another trick that will allow us to reduce the number of cases in the analysis. We make the following claim:

(NC) Without loss of generality, all items collected by the adversary in a stage have not been collected by EFH prior to this stage.

The claim follows from the way we perform our amortized analysis. Intuitively, if an item has been collected by EFH in an earlier stage and has full credit then collecting it now does not change the adversary's amortized gain. For special items the argument is a bit more subtle.

We now make this argument formal. Fix some adversary strategy and consider $a \in \{x, y, z\}$ that is collected by EFH when the stage starts. We introduce a dummy item $\tilde{a}$ of weight 0, inserted into the queue at the same time as a, located immediately before a in the queue, and not collected by EFH before this stage. Make the adversary collect $\tilde{a}$ instead of a. We claim that the adjusted a-gain remains the same.

For $a \in \{y, z\}$, the argument is simple: By the definition of adjusted gains, since a is collected by EFH, the adjusted a-gain is 0, and so is the adjusted $\tilde{a}$-gain.

Suppose now that $a = x$. If $\hat{e}$ is not a special item with partial credit then the argument is the same as for y and z. So assume now that $\hat{e}$ is a special item with partial credit. The argument is based on the definition of the adjusted x-gain. If $\hat{e} \lhd x$ then $\hat{e} \lhd \tilde{x}$ as well, and both the adjusted x-gain and adjusted $\tilde{x}$-gain are 0. If $\hat{e} \rhd x$ then $\hat{e} \rhd \tilde{x}$ as well, and both the adjusted x-gain and adjusted $\tilde{x}$-gain are $\frac{1}{3}\beta w_{\hat{h}}$. If $\hat{e} = x$ then the adjusted x-gain is the same as the x-gain which is equal to $\frac{1}{3}\beta w_{\hat{h}}$. But in this case we have $\hat{e} \rhd \tilde{x}$, so the adjusted $\tilde{x}$-gain is $\frac{1}{3}\beta w_{\hat{h}}$ too, completing the justification for (NC).

Lemma 10 *If EFH collects exactly one item in a stage, then* (3) *holds.*

Proof In this scenario, EFH collects e while the adversary collects x. By (NC), we can assume that x has not been yet collected by EFH. We show that the ratio on the left-hand side of (3) is bounded by R.

To streamline the argument, we assume first that $\hat{e}$ is not special. Since $\hat{e}$ is not special, the adjusted x-gain is the same as x-gain and is equal to w_x. Thus the adversary's amortized gain is the x-gain of w_x and possibly the new credit for e. We break the analysis into three (non-disjoint) cases, depending on the reason why the stage was terminated in (F).

Case 1: $e = h$. If $x \lhd h$, then $w_x \le \beta w_h$ (because we assumed that x was pending) and we give the adversary full credit for h. Hence, the competitive ratio is at most

$$\frac{w_h + \beta w_h}{w_h} = 1 + \beta \approx 1.576 < R.$$

On the other hand, if $x \trianglerighteq h$, then $w_x \le w_h$ and the adversary gains no new credit, in which case the competitive ratio is at most 1.

Case 2: There was an item q pending in (E) and deleted in (U1) such that $w_q \ge \xi w_h$ (this includes the case that h was deleted in (U1)). Since $w_q \ge \xi w_h > \beta w_h$ and q was pending in (E), we have $q \trianglerighteq e$; thus e was deleted in (U1) as well. Since $w_x \le w_h$ and as we do not give the adversary any new credit, the competitive ratio is at most

$$\frac{w_x}{w_e} \le \frac{w_h}{\beta w_h} = \frac{1}{\beta} = R.$$

Case 3: h is pending and $\alpha w_{h'} > w_h$ in (F). Suppose first that $x \lhd e$, in which case we have $w_x < \beta w_h$. If e remains pending, then it becomes a special item; thus we only need to give it $w_e - \frac{1}{3}\beta w_h$ of credit. So the competitive ratio is at most:

$$\frac{\beta w_h + w_e - \frac{1}{3}\beta w_h}{w_e} = \frac{\frac{2}{3}\beta w_h + w_e}{w_e} \leq \frac{\frac{5}{3}\beta w_h}{\beta w_h} = \frac{5}{3} < R.$$

If $x \trianglerighteq e$, then we not need to give the adversary any credit. Since also $w_x \leq w_h$, the competitive ratio is at most $\frac{1}{\beta} = R$.

This completes the proof with the assumption that $\hat{e}$ is not special. To extend the argument to the general case, we need to show that the above analysis remains valid for the case when $\hat{e}$ is special. The basic idea is that we can replace the x-gain of w_x by the estimate on the adjusted x-gain derived in Lemma 9. The justification is quite simple: The case analysis above and all estimates on gains and credits, except those for x, are independent of $\hat{e}$ being special or not. Regarding x, in each case above we estimated w_x by some quantity σw_h, where $\sigma \in \{\beta, 1\}$. But for $\sigma \geq \beta$, $w_x \leq \sigma w_h$ implies that $\max(\beta w_h, \min(w_x, w_h)) \leq \sigma w_h$, and thus the adjusted x-gain is bounded by σw_h as well, by Lemma 9. □

In the following, we consider only the situation in which EFH collected at least two items in a stage. To simplify the analysis, we begin with giving a general upper bound on the weight of items preceding f that are collected by the adversary.

Lemma 11 *Consider a stage where EFH collects at least two items. Let $a \in \{x, y, z\}$ be an item collected by the adversary in this stage that was pending for EFH at the beginning of the stage, in* (E)*. If $a \neq e$ and $a \lhd f$, then $w_a < \xi w_h$.*

Proof It is sufficient to show that a is pending for EFH in (F), because then $a \lhd f$ implies $w_a < \xi w_h$, by the choice of f.

If $a \in \{y, z\}$, then a is pending in (F), because $a \neq e$ and a is not deleted. So we can assume that $a = x$. Since EFH collected at least two items, by the code of EFH, no item pending for EFH in (E) of weight at least ξw_h was deleted in (U1). Thus, if x was deleted in (U1), then $w_x < \xi w_h$. If x was not deleted in (U1), then $x \neq e$ implies that x is pending for EFH in (F). □

Lemma 12 *If EFH collects exactly two items in a stage, then* (3) *holds.*

Proof By the assumption of the lemma, EFH collects e and f, with $e \lhd f \trianglelefteq h$, while the adversary collects x and y, with $x \lhd y$. Obviously, $e \neq h$ and $w_{h'} \geq w_h$. By (NC), we can assume that x and y have not been yet collected by EFH (this means that x is pending and y is either pending or released in (U1)). We show that the ratio on the left-hand side of (3) is bounded by R.

As in the proof of Lemma 10, we assume first that $\hat{e}$ is not special. Thus, the adversary's amortized gain consists of the x-gain of w_x, the y-gain of w_y, as well as possible new credits for e and f.

We consider three cases, depending on the reason why the stage was aborted in (H). For each case, we have sub-cases depending on the position of y relative to e and f.

Case 1: $f = h$. EFH collects e and h. There are several sub-cases.

Case 1.1: $y \lhd e$. Then $w_x \le \beta w_h$, $w_y \le \beta w_h$, and we give the adversary full credit for e and h. So the competitive ratio is at most

$$\frac{2\beta w_h + w_e + w_h}{w_e + w_h} \le \frac{3\beta w_h + w_h}{\beta w_h + w_h} \approx 1.731 < R.$$

Case 1.2: $e \trianglelefteq y \lhd h$. If none of items x, y is e, then $w_x \le \xi w_h$ and $w_y \le \xi w_h$, by Lemma 11. If one of x, y is e, then the other one weighs at most ξw_h, again by Lemma 11. In either case, we have $w_x + w_y \le \xi w_h + \max(w_e, \xi w_h)$. We also give the adversary full credit for h. Thus the competitive ratio is at most

$$\frac{\xi w_h + \max(w_e, \xi w_h) + w_h}{w_e + w_h} \le \frac{\xi w_h + \xi w_h + w_h}{\beta w_h + w_h} \approx 1.609 < R.$$

Case 1.3: $h \trianglelefteq y$. Then $w_x \le w_h$ and $w_y \le w_{h'} \le w_h/\alpha$ and we do not need to give the adversary any credit. Thus the competitive ratio is at most

$$\frac{w_h + \frac{1}{\alpha} w_h}{w_e + w_h} \le \frac{w_h + \frac{1}{\alpha} w_h}{\beta w_h + w_h} \approx 1.481 < R.$$

Case 2: h was deleted in (U2). Then $w_x \le w_h$, $w_y \le w_{h'} \le w_h/\alpha$, and we do not need to give the adversary any credit. Thus the competitive ratio is at most

$$\frac{w_h + \frac{1}{\alpha} w_h}{w_e + w_f} \le \frac{w_h + \frac{1}{\alpha} w_h}{\beta w_h + \xi w_h} = \frac{1}{\beta} = R.$$

Case 3: h is pending in (H) and $\alpha w_{h''} > w_h$. Again, there are several sub-cases.

Case 3.1: $y \lhd e$. Then $w_x \le \beta w_h$ and $w_y \le \beta w_h$. By the case assumption, if e is not deleted before (H), then it will become a special item, so we only need to give the adversary at most $w_e - \frac{1}{3}\beta w_h$ credit for e, as well as at most w_f credit for f. Then the competitive ratio is at most

$$\frac{2\beta w_h + w_e - \frac{1}{3}\beta w_h + w_f}{w_e + w_f} \le \frac{2\beta w_h + \beta w_h - \frac{1}{3}\beta w_h + \xi w_h}{\beta w_h + \xi w_h} = \frac{12}{7} \approx 1.714 < R.$$

Case 3.2: $e \trianglelefteq y \lhd f$. Here, the argument is similar to that in Case 1.2. Using Lemma 11, if none of items x, y is e, then $w_x \le \xi w_h$ and $w_y \le \xi w_h$. If one of x, y is e, then the other one weighs at most ξw_h. Thus $w_x + w_y \le \xi w_h + \max(w_e, \xi w_h)$. We give the adversary full credit for f. Hence, the competitive ratio is at most

$$\frac{\xi w_h + \max(w_e, \xi w_h) + w_f}{w_e + w_f} \le \frac{\xi w_h + \xi w_h + \xi w_h}{\beta w_h + \xi w_h} = \frac{12}{7} \approx 1.714 < R.$$

Case 3.3: $f \lhd y$. Then $w_x \le w_h$, $w_y \le w_{h'} \le w_h/\alpha$, and we do not have to give the adversary any credit. Hence, the competitive ratio is at most

$$\frac{w_h + \frac{1}{\alpha} w_h}{w_e + w_f} \le \frac{w_h + \frac{1}{\alpha} w_h}{\beta w_h + \xi w_h} = \frac{1}{\beta} = R.$$

This completes the proof with the assumption that $\hat{e}$ is not special. The extension to the general case is done as in the proof of Lemma 10, by showing that in each case the bounds shown for w_x and w_y hold for the adjusted x- and y-gain, respectively. Observe first that the case analysis, the bounds on the EFH's gain, and the new credits do not depend on whether $\hat{e}$ is special or not.

For the adjusted x-gain, we use Lemma 9. Note that in each case above we estimated w_x by σw_h, where $\sigma \in \{\beta, \xi, 1\}$, and for $\sigma \ge \beta$, $w_x \le \sigma w_h$ implies that $\max(\beta w_h, \min(w_x, w_h)) \le \sigma w_h$ too. The formula for the adjusted y-gain (no matter whether $\hat{e}$ is special or not) is the same as for the y-gain with $\hat{e}$ being not special. Thus all the bounds above hold if we replace the x-gain by the adjusted x-gain and the y-gain by the adjusted y-gain. □

Lemma 13 *If EFH collects exactly three items in a stage, then* (3) *holds.*

Proof By the lemma's assumption, EFH collects e, f and h, with $e \lhd f \lhd h$, while the adversary collects x, y, and z, with $x \lhd y \lhd z$. Obviously, $e, f \ne h$ and $w_h \le w_{h'} \le w_{h''}$. By (NC), we can assume that x, y and z have not been yet collected by EFH (this means that x is pending and y, z are either pending or released in this stage).

As in the proofs of the previous two lemmas, we assume first that $\hat{e}$ is not special. Thus, the adversary's amortized gain consists of the x-gain of w_x, the y-gain of w_y, the z-gain of w_z, as well as possible new credits for e, f and h.

The location of the adversary's last item z determines which of e, f and h should receive full credit (note that e will not become special in this case). We thus consider four cases, depending on the location of z.

Case 1: $z \lhd e$. Then w_x, w_y, $w_z \le \beta w_h$, and we give the adversary full credit for e, f and h. Thus, the competitive ratio is at most

$$\frac{3\beta w_h + w_e + w_f + w_h}{w_e + w_f + w_h} \le \frac{3\beta w_h + \beta w_h + \xi w_h + w_h}{\beta w_h + \xi w_h + w_h} = \frac{4\beta + \xi + 1}{\beta + \xi + 1} = R.$$

Case 2: $e \trianglelefteq z \lhd f$. Using Lemma 11, if none of x, y, z is e, then $w_x \le \xi w_h$, $w_y \le \xi w_h$ and $w_z \le \xi w_h$. If one of x, y, z is e, then each of the other two items weighs at most ξw_h. In both cases, $w_x + w_y + w_z \le 2\xi w_h + \max(w_e, \xi w_h)$. We also give the adversary full credit for f and h. Thus, the competitive ratio is at most

$$\frac{2\xi w_h + \max(w_e, \xi w_h) + w_f + w_h}{w_e + w_f + w_h} \le \frac{2\xi w_h + \xi w_h + \xi w_h + w_h}{\beta w_h + \xi w_h + w_h} = \frac{4\xi + 1}{\beta + \xi + 1} = R.$$

Case 3: $f \trianglelefteq z \triangleleft h$. Then $w_x, w_y, w_z \leq w_h$. We give the adversary full credit for h. Thus, the competitive ratio is at most

$$\frac{3w_h + w_h}{w_e + w_f + w_h} \leq \frac{4w_h}{\beta w_h + \xi w_h + w_h} \approx 1.707 < R.$$

Case 4: $h \trianglelefteq z$. Then $w_x \leq w_h$, $w_y \leq w_{h'} \leq w_h/\alpha$ and $w_z \leq w_{h''} \leq w_h/\alpha$. In this case, we do not need to give the adversary any credit. Hence, the competitive ratio is at most

$$\frac{w_h + \frac{2}{\alpha} w_h}{w_e + w_f + w_h} \leq \frac{w_h + \frac{2}{\alpha} w_h}{\beta w_h + \xi w_h + w_h} \approx 1.565 < R.$$

This completes the proof with the assumption that $\hat{e}$ is not special. To extend it to the general case, we proceed as in the proofs of Lemma 10 and Lemma 12. Our case analysis, the bounds on EFH's gain and on new credits were independent of whether $\hat{e}$ is special or not. For the x-gain, we again note that in each case above we estimated w_x by σw_h, where $\sigma \in \{\beta, \xi, 1\}$ and, for $\sigma \geq \beta$, $w_x \leq \sigma w_h$ implies that $\max(\beta w_h, \min(w_x, w_h)) \leq \sigma w_h$ too. Thus, all the bounds on the x-gain above hold for the adjusted x-gain as well. The formula for the adjusted y-gain and z-gain (no matter whether $\hat{e}$ is special or not) is the same as for the y-gain and z-gain, respectively, with $\hat{e}$ being not special. Thus all the bounds above hold if we replace the x-gain, y-gain and z-gain, by the corresponding adjusted gains. □

Note that all our credit updates ensure that (CR) is preserved from stage to stage. By summing up the adversary's amortized gain in all steps, Lemmas 10, 12, and 13 imply that

$$\text{(adversary's total gain)} + \text{(final total credit)} \leq R \cdot \text{(EFH's total gain)},$$

and R-competitiveness of EFH follows, because the credits are non-negative.

Recall that in the proof above we chose $\alpha = \frac{3}{4}$, $\beta = (\sqrt{13} + 1)/8$, and $\xi = (\sqrt{13} + 1)/6$. Summarizing this section, we obtain:

Theorem 3 *For constants α, β and ξ given above, EFH is R-competitive, where $R = 2(\sqrt{13} - 1)/3 \approx 1.737$.*

5.2 Lower Bound for EFH

In this section we prove that the analysis of EFH is tight, even for the case of *decremental queues*, that is when all items are inserted at the beginning, before the first step, and all subsequent updates are deletions. Note that for decremental queues the choice of α is irrelevant, because in this case we will always have $w_{h'}, w_{h''} \leq w_h$.

Theorem 4 *For any choice of parameters $\beta \leq \xi$ in EFH, its competitive ratio is at least $2(\sqrt{13} - 1)/3 \approx 1.737$, even in the case of decremental queues.*

Proof We consider three instances. In each instance, to simplify notation, we will identify items with their weights, that is we will say "item w" instead of "item of weight w".

In the first instance we have two items, β and 1, in this order. EFH collects β, while the adversary collects 1 and deletes both items. The ratio is $R_1(\beta,\xi) = R_1(\beta) = \frac{1}{\beta}$.

Now, choose a small ϵ, $0 < \epsilon < \beta$. (Since $R_1(\beta)$ is unbounded for $\beta = 0$, we can assume that $\beta > 0$.)

In the second instance we have six items, $\beta-\epsilon$, $\beta-\epsilon$, $\beta-\epsilon$, β, ξ, and 1, in this order. In the first three steps EFH collects β, ξ and 1, while the adversary collects the three items $\beta-\epsilon$. Right after the third step the adversary deletes these first three items and in the remaining steps he collects β, ξ and 1. For $\epsilon \to 0$, the ratio is arbitrarily close to $R_2(\beta,\xi) = \frac{4\beta+\xi+1}{\beta+\xi+1}$.

In the third instance we have six items, $\beta, \xi-\epsilon, \xi-\epsilon, \xi-\epsilon, \xi$, and 1, in this order. In the first three steps EFH collects β, ξ and 1, while the adversary collects the three items $\xi-\epsilon$. Right after the third step the adversary deletes the first four items and later he collects ξ and 1. For $\epsilon \to 0$, the ratio is arbitrarily close to $R_3(\beta,\xi) = \frac{4\xi+1}{\beta+\xi+1}$.

It remains to show that $\max_{i=1,2,3} R_i(\beta,\xi) \geq 2(\sqrt{13}-1)/3$ for any valid choice of parameters β and ξ. Actually, we show that this condition holds even if we allow β and ξ to be greater than 1 (which is not possible in our algorithm), i.e., we only require that $0 \leq \beta \leq \xi$.

To this end, note that for $\beta \leq (\sqrt{13}+1)/8 \approx 0.576$ it holds that $R_1(\beta,\xi) \geq 2(\sqrt{13}-1)/3$, so the claim holds.

Hence, it is sufficient to show that $\max_{i=2,3} R_i(\beta,\xi) \geq 2(\sqrt{13}-1)/3$ for any choice of β and ξ such that $(\sqrt{13}+1)/8 \leq \beta \leq \xi$. Observe that for any fixed β, $R_2(\beta,\xi)$ is a decreasing function of ξ and $R_3(\beta,\xi)$ is an increasing function of ξ. Further, $R_2(\beta,\beta) > R_3(\beta,\beta)$, $R_2(\beta,2\beta) \leq R_3(\beta,2\beta)$, and hence $\max_{i=2,3} R_i(\beta,\xi)$ is minimized when $\xi = \frac{4}{3}\beta$, for which

$$R_2(\beta,\xi) = R_3(\beta,\xi) = \frac{16\beta+3}{7\beta+3}.$$

The latter fraction is an increasing function of β. Hence, $\max_{i=2,3} R_i(\beta,\xi) \geq (16\beta+3)/(7\beta+3)$, which for $\beta \geq (\sqrt{13}+1)/8$ is at least $2(\sqrt{13}-1)/3$. □

Note that adding further steps with thresholds larger than ξ would not improve EFH's competitive ratio, as such an algorithm behaves as EFH on all the instances presented in the above proof.

6 Lower Bounds

As the item collection problem is a generalization of the buffer management problem, we immediately get a lower bound of ϕ [1, 4] for the competitive ratio of any deterministic algorithm. However, for our problem the proof can be substantially simplified: Start with two items, $a \lhd b$, with weights $w_a = 1$ and $w_b = \phi$. If the algorithm

chooses b, the adversary chooses a, deletes it, and collects b in the next step, so the competitive ratio is $(w_a + w_b)/w_b = (1 + \phi)/\phi = \phi$. If the algorithm chooses a, the adversary chooses b and deletes both items, so the competitive ratio is $w_b/w_a = \phi$ again.

6.1 Lower Bound for the General Case

In the remainder of this section, we show how to improve the lower bound to 1.63. Our lower bound works even in the decremental case, however, for simplicity reasons, we show it first for the general case. The proof is by presenting an adversary's strategy that forces any deterministic online algorithm A to gain less than $1/1.63$ times the adversary's gain.

Adversary's Strategy We assume that the items appear gradually, so that at each step the algorithm has at most three items to choose from. Fix some $n \geq 2$. To simplify notation, in this section we refer to items simply by their weight, thus "z_i" denotes both an item and its weight. The instance consists of a sequence of $2n$ items $1, z_1, z_2, \ldots, z_{2n-1}$ such that

$$z_2 \lhd z_4 \lhd \cdots \lhd z_{2n-2} \lhd z_{2n-1} \lhd z_{2n-3} \lhd \cdots \lhd z_3 \lhd z_1 \lhd 1 \quad \text{and}$$

$$1 > z_1 > z_2 > \cdots > z_{2n-3} > z_{2n-2} > z_{2n-1} > 0.$$

The even- and odd-numbered items in this sequence form two roughly geometric sequences. In fact, z_{2i} is only slightly smaller than z_{2i-1}, for all $i = 1, \ldots, n-1$.

Initially, items $z_2 \lhd z_1 \lhd 1$ are present. In step $i = 1, 2, \ldots, n-1$, the adversary maintains the invariant that the active items are $z_{2i} \lhd z_{2i-1} \lhd z_{2i-3} \lhd \cdots \lhd z_1 \lhd 1$, of which only three items z_{2i}, z_{2i-1} and 1 are pending for A. In other words, items $z_2, z_4, \ldots, z_{2i-2}$ are already deleted, while items $z_1, z_3, \ldots, z_{2i-3}$ are collected by A.

The adversary's move in step i depends now on what A collects in this step:

Case (i): A collects z_{2i}. Then the adversary ends the game by deleting all active items. In this case the adversary collects i heaviest items: $1, z_1, z_2, \ldots, z_{i-1}$.
Case (ii): A collects 1. The adversary ends the game by deleting z_{2i} and z_{2i-1}. This leaves A with no pending items, and the adversary can now collect A's items one by one. Overall, in this case the adversary collects $2i$ heaviest items: $1, z_1, z_2, \ldots, z_{2i-1}$.
Case (iii): A collects z_{2i-1}. In this case the game continues. If $i < n-1$, the adversary deletes z_{2i}, inserts z_{2i+2} and z_{2i+1} into the current list (according to the order defined earlier), and the game proceeds to step $i+1$. The case $i = n-1$ is slightly different: here the adversary deletes z_{2n-2} and inserts only the last item z_{2n-1} before proceeding to step n (described below).

If the game reaches step n, all odd-numbered items and 1 are active, whereas A has two pending items, z_{2n-1} and 1. In this step, the adversary behavior is similar to previous steps: if A collects z_{2n-1}, then the adversary deletes the whole sequence and collects n heaviest items: $1, z_1, z_2, \ldots, z_{n-1}$. If A collects 1, the adversary deletes z_{2n-1}, leaving A without pending items, and allowing the adversary to collect the whole sequence.

Note that the adversary's strategy is feasible, in the sense that he can indeed collect items as described above. This follows from the fact that all steps (when A collects an item) except the last one are of type (iii), where only the leftmost active item is deleted, and the adversary needs only one item from among the items deleted in the last step. Thus, if the adversary collects his items from left to right (in the queue order), each item will be pending when it is about to be collected.

Lemma 14 *Suppose that there is a sequence* $1 > z_1 > \cdots > z_{2n-1} > 0$, *and a constant* R, *such that for all* $j = 0, 1, \ldots, n-1$ *it holds that*

$$R \cdot \left(1 + \sum_{i=1}^{j} z_{2i-1}\right) \leq 1 + \sum_{i=1}^{2j+1} z_i \quad \textit{and}$$

$$R \cdot \left(z_{2j+2} + \sum_{i=1}^{j} z_{2i-1}\right) \leq 1 + \sum_{i=1}^{j} z_i,$$

where, in the second inequality, for $j = n-1$ *we assume that* $z_{2n} = z_{2n-1}$. *Then there is no* R'*-competitive deterministic online algorithm such that* $R' < R$.

The lemma is clear from the description of the strategy given earlier, since the sides of the inequalities above represent the gains of the adversary and the algorithm in various steps. Additionally, the lemma holds even if all the items are inserted at the beginning. To achieve this, we slightly modify the adversary's strategy: all the items are present at the beginning, and whenever A deviates from the choices (i), (ii), (iii), it collects an item lighter than z_{2i}, and thus the adversary can finish the game as in Case (i). Lemma 14 and straightforward calculations for $n = 3$ (6 items) yield the following.

Theorem 5 *There is no deterministic online algorithm for dynamic queues* (*even for the decremental case*) *with competitive ratio smaller than* 1.6329.

Proof We follow the adversary strategy explained earlier for $n = 3$. To simplify notation, we rename the items: z_2, z_4, z_5, z_3, z_1 as x, y, z, u, v, respectively. Using Lemma 14, we want to find numbers x, y, z, u, v such that $0 < z < y < u < x < v < 1$ and a number R for which:

$$\begin{aligned}
R \cdot 1 &\leq 1 + v, \\
R \cdot x &\leq 1, \\
R \cdot (1 + v) &\leq 1 + v + x + u, \\
R \cdot (v + y) &\leq 1 + v, \\
R \cdot (1 + v + u) &\leq 1 + v + x + u + y + z, \\
R \cdot (v + u + z) &\leq 1 + v + x.
\end{aligned}$$

(According to the statement of Lemma 14, we have $z_6 = z_5 = z$ as well.)

It is straightforward to check that the following numbers satisfy the inequalities above: $R = 1.6329$, $z = 0.3205$, $y = 0.3671$, $u = 0.4211$, $x = 0.6124$, and $v = 0.6329$, implying the lemma. These numbers were obtained by replacing inequalities by equalities, some substitutions, and then computing an approximate solution of the resulting polynomial equation $R^5 + R^3 - 5R^2 - R - 1 = 0$. This polynomial has one real root, and its exact value $R = 1.63297\ldots$ is only slightly bigger that the value given above. □

A natural question arises how much this bound can be improved with sequences $\{z_i\}$ of arbitrary length. For $n = 5$ (10 items), one can obtain $R = 1.6367\ldots$, and our numerical experiments indicate that the corresponding ratios tend to ≈ 1.6378458, so the improvement is minor.

6.2 Lower Bound for Memoryless Algorithms

An algorithm A for dynamic queues is called *memoryless* if at each step its decision as to which item to collect is based only on its pending items, i.e., their weights and order in the queue. We now give some lower bounds on the competitive ratios of such memoryless algorithms, both deterministic and randomized ones.

Theorem 6 *The competitive ratio of any deterministic memoryless algorithm is at least* 2.

Proof Fix a memoryless algorithm A. We give an adversary's strategy where the adversary's gain is $2 - o(1)$ times A's gain.

Pick large integers n and $T \gg n$, and let $X = \{x_0, \ldots, x_n\}$ be a set of items where $w_{x_i} = 1 + \frac{i}{n}$ for $i = 0, 1, \ldots, n$. The adversary maintains the invariant that at each step A's pending set is X (or, more precisely, the weights of the pending items are the same as those in X), with the items ordered by increasing value. Suppose that for this pending set, A collects some item x_k.

If $k = 0$, the adversary collects item x_n, deletes all items, inserts copies of all items from X again into the queue, and repeats the process T times. A's total gain is $T w_{x_0} = T$ while the optimum gain is $T w_{x_n} = 2T$, so the competitive ratio is 2. Suppose now that $k \geq 1$. In this case, the adversary collects x_{k-1}, deletes all items $x_0, \ldots, x_{k-1}$, and inserts new copies of items $x_0, \ldots, x_k$. This process is repeated T times. After T steps, the adversary collects the remaining uncollected items, in particular, all T copies of item x_k. A can of course collect the remaining pending items. The value collected by A is at most

$$T w_{x_k} + 2(n+1) = T(1 + k/n) + 2(n+1),$$

while the value collected by the adversary is at least

$$T(w_{x_{k-1}} + w_{x_k}) = T(2 + (2k-1)/n).$$

So with $T = n^3$ and $n \to \infty$, the ratio approaches 2. □

In lower bound proofs based on an adversary argument, the adversary is assumed to know the online algorithm A under consideration. Thus, if A is deterministic, the adversary can predict A's decision on each input sequence. This is not true if A is randomized; in this case the adversary can only predict the probability distribution of A's decisions on each input. That type of an adversary is referred to as an *oblivious adversary*—oblivious in the sense that at each step he does not know what random choices were taken by A in the past.

Another, stronger type of an adversary considered in the literature is called an *adaptive-online adversary*. At each step, an adaptive-online adversary has access to A's random choices from previous steps, so it knows the exact state of A at this step. We now prove a lower bound of $\mathrm{e}/(\mathrm{e}-1)$ for randomized memoryless algorithms against such an adversary. This matches an upper bound achieved by RMIX [3], which, although originally designed for packet scheduling, can be easily adapted to our dynamic queue model. (The competitive analysis detailed in [3] works for the oblivious adversary, but the proof can be modified to work even against the adaptive-online adversary—[7].)

Theorem 7 *The competitive ratio of any randomized memoryless algorithm against an adaptive-online adversary is at least* $\frac{\mathrm{e}}{\mathrm{e}-1}$.

Proof Fix some online memoryless randomized algorithm A. We consider the following scheme. Let $a > 1$ be a constant, which we specify later, and n be a fixed integer. At the beginning, the adversary inserts items $a^0, a^1, \ldots, a^n$ into the queue, in this order. (To simplify notation, in this proof we identify items with their weights.) In our construction we maintain the invariant that in each step, the list of items pending for A is equal to $a^0, a^1, \ldots, a^n$. Since A is memoryless, in each step it uses the same probability distribution $(q_j)_{j=0}^n$, where q_j is the probability of collecting item a^j. Moreover, $\sum_{i=0}^n q_i = 1$, as without loss of generality the algorithm always makes a move.

We consider $n+1$ strategies for an adversary, numbered $0, 1, \ldots, n$. The k-th strategy is as follows: in each step collect a^k, delete items $a^0, a^1, \ldots, a^k$, and then issue new copies of these items. Additionally, if A collected a^j for some $j > k$, then the adversary issues a new copy of a^j as well. This way, in each step exactly one copy of each a^j is pending for A, while the adversary accumulates in its pending set copies of the items a^j, for $j > k$, that were collected by A.

This step is repeated $T \gg n$ times, and after the last step both the adversary and the algorithm collect all their pending items. Since $T \gg n$, we only need to focus on the expected amortized profits (defined below) in a single step.

We look at the gains of A and the adversary in a single step. If the adversary chooses strategy k, then it gains a^k. Additionally, at the end it collects the item collected by the algorithm if this item is greater than a^k. Thus, its *amortized expected gain* in a single step is $a^k + \sum_{i>k} q_i a^i$. The expected gain of A is $\sum_i q_i a^i$.

For any probability distribution $(q_j)_{j=0}^n$ of the algorithm, the adversary chooses a strategy k which maximizes the competitive ratio. Thus, the competitive ratio of A is

at least

$$R = \max_k \left\{ \frac{a^k + \sum_{j>k} q_j a^j}{\sum_j q_j a^j} \right\} \geq \sum_k v_k \frac{a^k + \sum_{j>k} q_j a^j}{\sum_j q_j a^j}, \tag{4}$$

for any coefficients $v_0, \ldots, v_n \geq 0$ such that $\sum_k v_k = 1$. Let $M = a^{n+1} - n(a-1)$. For $k = 0, 1, \ldots, n$, we choose

$$v_k = \begin{cases} \frac{1}{M} a^{n-k}(a-1), & \text{if } k < n, \\ \frac{1}{M}\left(a - n(a-1)\right), & \text{if } k = n. \end{cases}$$

The choice of these values may seem somewhat mysterious, but it's in fact quite simple—it is obtained by considering A's distributions where $q_j = 1$ for some j (and thus when A is deterministic), assuming that the resulting lower bounds on the right-hand side of (4) are equal, and solving the resulting system of equations.

For these values of v_k we obtain

$$\begin{aligned} MR \sum_{j=0}^{n} q_j a^j &\geq \sum_{k=0}^{n} M v_k a^k + \sum_{k=0}^{n} M v_k \sum_{j>k} q_j a^j \\ &= \sum_{k=0}^{n-1} M v_k a^k + M v_n a^n + \sum_{j=0}^{n} q_j a^j \sum_{k<j} M v_k \\ &= n(a-1)a^n + [a - n(a-1)]a^n + \sum_{j=0}^{n} q_j (a^j - 1) a^{n+1} \\ &= a^{n+1} + a^{n+1} \sum_{j=0}^{n} q_j a^j - a^{n+1} \sum_{j=0}^{n} q_j \\ &= a^{n+1} + a^{n+1} \sum_{j=0}^{n} q_j a^j - a^{n+1} \\ &= a^{n+1} \sum_{j=0}^{n} q_j a^j. \end{aligned}$$

Therefore, $R \geq a^{n+1}/M$. This bound is maximized for $a = 1 + 1/n$, in which case we get

$$R \geq \frac{(1 + \frac{1}{n})^{n+1}}{(1 + \frac{1}{n})^{n+1} - 1} \xrightarrow{n \to \infty} \frac{\mathrm{e}}{\mathrm{e} - 1},$$

completing the proof. □

The $\mathrm{e}/(\mathrm{e}-1)$ bound in the proof of Theorem 7 follows from considering an infinite series of adversary's strategies, such that the n-th strategy forces ratio $r_n =$

$(1 + 1/n)^{n+1}/((1 + 1/n)^{n+1} - 1)$. Furthermore, for all n, the n-th strategy satisfies the invariant that there are at most $n + 1$ items pending for the algorithm in every step. We note that the algorithm provided in [7] matches the bounds provided by all these strategies simultaneously, in the sense that its competitive ratio is at most r_n, whenever there are at most $n + 1$ items pending for it in every step.

Acknowledgements We would like to thank Mart Molle for pointing out a connection between our model and networks with tiered QoS traffic or with intermittent access. We are also grateful to anonymous referees for numerous useful suggestions that helped us improve the presentation and simplify some arguments.

References

1. Andelman, N., Mansour, Y., Zhu, A.: Competitive queueing policies in QoS switches. In: Proceedings of the 14th Symposium on Discrete Algorithms (SODA), pp. 761–770. ACM/SIAM, New York (2003)
2. Birnbaum, B., Mathieu, C.: On-line bipartite matching made simple. SIGACT News **39**(1), 80–87 (2008)
3. Chin, F.Y.L., Chrobak, M., Fung, S.P.Y., Jawor, W., Sgall, J., Tichý, T.: Online competitive algorithms for maximizing weighted throughput of unit jobs. J. Discrete Algorithms **4**, 255–276 (2006)
4. Chin, F.Y.L., Fung, S.P.Y.: Online scheduling for partial job values: Does timesharing or randomization help? Algorithmica **37**, 149–164 (2003)
5. Englert, M., Westermann, M.: Considering suppressed packets improves buffer management in QoS switches. In: Proceedings of the 18th Symposium on Discrete Algorithms (SODA), pp. 209–218. ACM/SIAM, New York (2007)
6. Hajek, B.: On the competitiveness of online scheduling of unit-length packets with hard deadlines in slotted time. In: Conference on Information Sciences and Systems, pp. 434–438 (2001)
7. Jeż, Ł.: One to rule them all: A general randomized algorithm for buffer management with bounded delay. In: Proceedings of the 19th Annual European Symposium on Algorithms (ESA), pp. 239–250 (2011)
8. Kalyanasundaram, B., Pruhs, K.: An optimal deterministic algorithm for online b-matching. Theor. Comput. Sci. **233**(1–2), 319–325 (2000)
9. Kesselman, A., Lotker, Z., Mansour, Y., Patt-Shamir, B., Schieber, B., Sviridenko, M.: Buffer overflow management in QoS switches. SIAM J. Comput. **33**, 563–583 (2004)
10. Kesselman, A., Mansour, Y., van Stee, R.: Improved competitive guarantees for QoS buffering. Algorithmica **43**, 63–80 (2005)
11. Li, F., Sethuraman, J., Stein, C.: An optimal online algorithm for packet scheduling with agreeable deadlines. In: Proceedings of the 16th Symposium on Discrete Algorithms (SODA), pp. 801–802. ACM/SIAM, New York (2005)
12. Li, F., Sethuraman, J., Stein, C.: Better online buffer management. In: Proceedings of the 18th Symposium on Discrete Algorithms (SODA), pp. 199–208. ACM/SIAM, New York (2007)
13. Mehta, A., Saberi, A., Vazirani, U.V., Vazirani, V.V.: Adwords and generalized online matching. J. ACM **54**(5), 22 (2007)
14. Meteor burst communications. http://en.wikipedia.org/wiki/Meteor_burst
15. WiMAX. http://en.wikipedia.org/wiki/WiMAX