An improved algorithm for online rectangle filling*

Rob van Stee†

November 7, 2011

Abstract

We consider the problem of scheduling resource allocation where a change in allocation results in a changeover penalty of one time slot. We assume that we are sending packets over a wireless channel of uncertain and varying capacity. In each time slot, a bandwidth of at most the current capacity can be allocated, but changing the capacity has a cost, which is modeled as an empty time slot. Only the current bandwidth and the bandwidth of the immediately following slot are known. We give an online algorithm with competitive ratio 1.753 for this problem, improving over the previous upper bound of 1.848. The main new idea of our algorithm is that it attempts to avoid cases where a single time slot with a nonzero allocation is immediately followed by an empty time slot. Additionally, we improve the lower bound for this problem to 1.6959, and give a better randomized lower bound. Our results significantly narrow the gap between the best known upper and lower bound.

Keywords: online algorithms, resource management and awareness, wireless networks

1 Introduction

In wireless networks, channel conditions can change frequently, which affects the bit error rate and therefore the channel transmission capacity [5]. We consider the problem of setting data transmission rates over such a channel in order to maximize the throughput. Naturally, at any time the transmission rate cannot be higher than the current transmission capacity, but there is also typically a nonzero cost involved in changing the transmission rate, because the transmitter and receiver will have to coordinate and reset to a new transmission rate. We model this cost as the loss of a single time slot. That is, whenever we want to change the transmission rate (or if we are forced to change it, because the current capacity is below the rate that we set earlier), we will have one time slot in which nothing can be transmitted.

Formally, we are given an online sequence of nonnegative real numbers $h(1), h(2), \ldots$, which represent the maximum transmission capacities of the wireless channel at each time step, and we need to determine the transmission rate $u(i)$ at each time step. Our goal is to maximize $\sum_i u(i)$, and due to the changeover cost we have for any i that $u(i) = u(i+1)$, $u(i) = 0$, or $u(i+1) = 0$. It can be seen that if only the current bandwidth is known, no competitive online algorithm exists [2]. We therefore focus on the case where some information about future bandwidth is given; in particular, for our results we assume that we have a lookahead of a single time slot.

*An extended abstract of this paper appeared in the 8th Workshop on Approximation and Online Algorithms (WAOA 2010), LNCS 6534, p.249–260. Springer, 2011.

†Research supported by German Research Foundation (DFG). Max-Planck-Institut für Informatik, Saarbrücken, Germany. vanstee@mpi-inf.mpg.de

Figure 1: An overview of the deterministic upper and lower bounds for this problem. A = Arora et al. [2], W = Wang et al. [8].

The name rectangle filling comes from a geometrical interpretation of the problem, where each time slot i is represented by a rectangle of unit width and height $h(i)$ (also called a *column*). An algorithm needs to decide how much of each rectangle to fill, i.e., what the transmission rate $\in [0, h(i)]$ should be. Any feasible solution for this problem is a set of rectangles (of varying width) where the transmission rate is constant; all these rectangles are separated by one or more zero columns.

Probabilistic analysis for this problem was given by Tsibonis et al. [6] and Borst [3]. Arora and Choi were the first ones to study this problem from a worst-case perspective [1]. They gave a dynamic program for the offline version with a running time of $O(n^3)$, and a 4-competitive online algorithm called Wait Dominate Hold (using a lookahead of 1). Arora et al. [2] soon afterwards showed that this algorithm is actually $8/3$-competitive, and gave a lower bound of $8/5$. For the version of the problem with k-lookahead, they gave an online algorithm with competitive ratio 2 for any k, and a lower bound of $(k+2)/(k+1)$.

Wang et al. [8] presented a faster offline algorithm with a running time of $O(n^2)$, and new lower and upper bounds of 1.6358 and 1.848 respectively. Later, the same authors [7] considered the version with k-lookahead. They gave a deterministic algorithm with a competitive ratio of $1 + 2/(k-1)$, as well as a randomized algorithm with competitive ratio $1 + 1/(k+1)$. They also gave a randomized lower bound of $1 + 1/(\sqrt{k+2} + \sqrt{k+1})^2$, which is more than $1 + 1/(4k+8)$ and tends to $1 + 1/(4k+4)$ for large k.

Generally, despite the seeming simplicity of the model, the gap between the upper and lower bounds has so far remained relatively large, particularly for the most basic version with a lookahead of 1, and it appears to be very hard to give tight bounds for this problem. See also the conclusions.

Our results We give an improved algorithm which achieves a competitive ratio of less than 1.753. The main new idea of our algorithm is to try and limit the amount of changes in the used bandwidth. Therefore, as soon as a nonzero transmission rate is allocated, we temporarily relax the condition for changing the rate, and allow a *single* time slot with relatively high capacity to appear without resetting the transmission rate. The idea behind this is that as long as there is only a single slot with high capacity, we do not lose too much compared to the optimal solution, because the optimal solution needs to allocate zero columns before and after the slot in order to be able to use the full capacity of the slot. That is, it is not worth the trouble of paying a penalty to serve a slot completely, especially if we have very recently paid another penalty to start transmitting.

The analysis uses the natural block partitioning from Wang et al. [8] as a starting point, but is significantly more involved. Apart from their partitioning rules, we will need a number of additional assignment rules to deal with the columns that are left unassigned in their scheme. These assignment rules depend on the optimal solution, and sometimes assign part of the optimal profit to preceding blocks and part to following blocks in order to allow us to analyze these blocks independently.

Moreover, in some cases we analyze the competitive ratio by splitting the instance into two parts, replacing one column by a sequence of columns, and showing that our algorithm gives exactly the same allocation as before to all columns where it does not allocate 0, and the optimal profit is split according to another set of rules between the first part and the second part of the input.

Finally, we improve the lower bound from 1.6358 to 1.6959. The construction is similar to the one from Wang et al. [8], but we use an extra threat in every step of the input. We also present a randomized lower bound of $(k+2)\ln\frac{k+2}{k+1} > 1 + 1/(2k+3)$ due to Epstein and Levin [4].

2 Algorithm MoreFilling

We first give an informal description of our algorithm. MoreFilling creates blocks between zero columns. If it ever encounters a column with less height than the currently chosen allocation, it must set $u(t) = 0$ (Rule 1). The nonzero part of a block starts by trying to guess a good allocation for the first two nonzero columns. If the first height is much smaller than the second one (by a factor of at least $\gamma > 2$), then the nonzero part of the block is simply postponed till later (Rule 2, last entry of table). Otherwise, if the first height is much larger (by a factor of at least $\frac{1}{\beta} > 1.4$), then the first column is allocated its full height, and the block will contain a single nonzero column (Rule 2, first entry of table). If the two heights are relatively close, the minimum between the two heights is allocated (Rule 2, middle entries).

These ideas were also used before [1, 8]. The main enhancement of our algorithm is to try and avoid having zero columns if the nonzero part of a block has just started. Hence there is one case where the first nonzero column is fully used, where we allow the column height to grow by a factor of $\gamma > 2$ for just one step, while still keeping other (later) column heights bounded by a smaller factor. This is done by setting q_3 in line 2 of Rule 3. Hence we ensure that there is a zero column only if the new column height is significantly larger, or if the nonzero column is allocated its full height. The third column in a block is almost always limited to a maximum allowed height of $\delta < 1.6$ times the allocation of MoreFilling on columns of that block. The only exception is if the second column has height between 1 and δ times the first column, i.e., the block does not contain a column of height (almost) γ yet. We never let the heights grow by a factor of γ if the block already contains at least two nonzero columns.

We now formally define our algorithm. Define the following values.

$$\begin{array}{lll@{\qquad}lll}
\mathcal{R} = & & = 1.75214 & \beta = & \mathcal{R}/(2\mathcal{R}-1) & = 0.69966 \\
\varepsilon = & (2\mathcal{R}-3)\beta & = 0.35282 & \gamma = & \mathcal{R}/(4-2\mathcal{R}+\varepsilon) & = 2.06489 \\
\delta = & (2\mathcal{R}\beta-\varepsilon-1)/\beta & = 1.57074 & \eta = & 1-\mathcal{R}/\gamma+\varepsilon & = 0.50414
\end{array}$$

We have $(5-\mathcal{R}/\gamma+\varepsilon)/(1+\delta) = \mathcal{R}$. Our algorithm is defined in Figure 2.

Theorem 1 *The competitive ratio of MoreFilling is at most* $\mathcal{R} = 1.75214$.

It should be pointed out that δ could be set to any value between roughly 1.55 and $5/3$, and MoreFilling would still be 1.753-competitive. The other variables, β, ε and γ are decisive.

3 Upper bound analysis

Fix an optimal solution OPT. We denote its allocation to column i by $a(i)$. For a set of columns S, let ALG(S) denote the total profit of algorithm ALG on the columns in S. We abbreviate MoreFilling by MF.

We begin our analysis by introducing the block partitioning and giving some properties of the optimal solution in Section 3.1. We use this to analyze the most basic type of block in Theorem 2. The remaining cases are treated in Sections 3.2–3.5. A more detailed overview of the cases is given at the end of Section 3.1.

The first column arrives at time $t = 1$. We take $u(0) = 0$.

1. If $u(t-1) > h(t)$, set $u(t) = 0$.
2. If $u(t-1) = 0$, set $u(t)$ depending on $h(t+1)/h(t)$, as follows:

$h(t+1)/h(t)$	$[0, \beta)$	$[\beta, 1)$	$[1, \gamma)$	$[\gamma, \infty)$
$u(t)$	$h(t)$	$h(t+1)$	$h(t)$	0

3. If $u(t-1) > 0$, let $t_1 = 1 + \max\{t \geq 0 | u(t) = 0\}$, and set $H = \min(h(t_1), h(t_1+1))$.
   If $t_1 = t-1$ and $h(t)/h(t-1) \in [1, \delta]$, set $q_3 := \gamma$, else $q_3 := \delta$.
   If $h(t+1) \geq q_3 H$, set $u(t) = 0$, else $u(t) = u(t-1)$.

Figure 2: The algorithm MoreFilling, consisting of three rules. The value t_1 in Rule 3 is the index of the first nonzero column of the current block, and H is its base height (see Definition 2).

3.1 Block partitioning

We classify the columns with zero allocation (also called zero columns) by the step in which they are set to 0: a column is of type i if it is set to 0 in Step i of our algorithm. Note that type 2 columns only occur after other zero columns. We follow the partitioning scheme from Wang et al. [8], which we describe next.

A type 1 column ends a block, and is a part of that block. If column i is a type 3 column, one block ends at column $i-1$ and the next starts at column $i+1$. (In this case, we will decide later what to do with column i.) This partitioning scheme ignores the type 2 columns (which only occur after other zero columns). Each block hence consists of zero or more type 2 columns, followed by one or more nonzero columns and possibly one final type 1 column. For each block B, the number of *nonzero* columns in B, also called the *length* of B, is denoted by $|B|$.

Let us consider the possible optimal profit on a block. We normalize the height of the columns in this block such that the height of the first column is exactly 1.

Definition 1 *We define the* BaseHeight *of a block with at least two nonzero columns as the minimum height among its first two nonzero columns.*

The BaseHeight is abbreviated by H in the algorithm.

Definition 2 *A block is called* long *if it has at least one (nonzero) column after its first fully-used column, else it is called* short*.*

Observation 1 *Consider a block of which the first nonzero column is column i. We have $h(i) = 1$. If $h(i) > h(i+1)$, then $h(i+1) \geq \beta h(i)$, or $u(i+1) = 0$. If $u(i+1) > 0$ and the last column k of this block is of type 1, we have $h(k) < \beta$.*

If $h(i) \leq h(i+1)$, then $h(i+1) < \gamma$. Moreover, almost always we have $h(j) < \delta$ for all $j > i+1$ such that j and i belong to the same block. The only exception to this is if $h(i+1) \leq \delta$, in which case we have $h(i+2) < \gamma$ if $i+2$ is part of the same block. In addition, if the last column k of the block of i is of type 1, we have $h(k) < \beta$.

We now consider the type 2 zero columns at the start of the block.

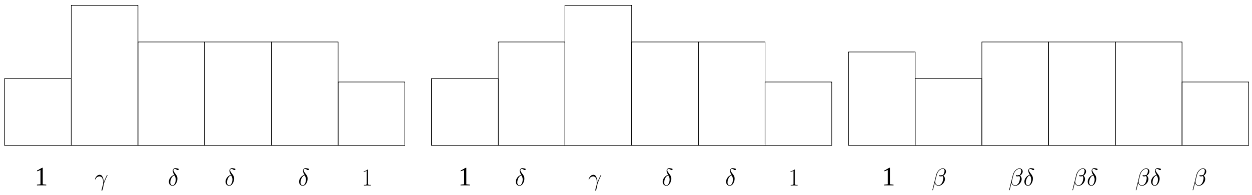

Figure 3: This figure shows the maximum possible heights of nonzero columns in a block (plus a final type 1 column). There are three cases. Let the first nonzero column of the block be t, then we have the cases $\delta h(t) < h(t+1) \leq \gamma h(t)$ (left), $h(t) \leq h(t+1) \leq \delta h(t)$ (middle), and $\beta h(t) \leq h(t+1) < h(t)$ (right). If $h(t+1) < \beta h(t)$, the block contains only a single nonzero column; if $h(t+1) > \gamma h(t)$, no block starts at time t. The third column may have height γ only if the heights are ascending and the second column has height at most δ. Shown is the case where the sixth column is of type 1; if there are additional nonzero columns instead, their maximum height is the same as that of the fifth column.

Lemma 1 (Wang et al.) *Given a sequence of columns S, if the height of each column is at least γ times the height of the previous one (where $\gamma \geq 2$), then $\text{OPT}(S) \leq \frac{\gamma^2}{\gamma^2-1}h$ where h is the height of the last column in S. This value is achieved by using every other column completely starting from the right.*

In order to efficiently deal with all the cases, we will in fact use the following estimates, which are all higher than the bound from Lemma 1. Consider a column i of height h which is preceded by type 2 columns. We make two distinctions: one based on whether $a(i) > h/\gamma$ or not (if $a(i) > h/\gamma$, then $a(i-1) = 0$), and one based on whether the block containing column i is long or short (Definition 2). The bounds used are as follows.

	$a(i) > h/\gamma$	$a(i) \leq h/\gamma$
Short block	$\varepsilon \approx 0.353$	$\varepsilon\gamma \approx 0.729$
Long block	$\eta \approx 0.505$	$\varepsilon\gamma \approx 0.729$

(1)

Naturally the profit on type 2 columns does not really depend on the type of the following block, but this assumption simplifies the analysis later.

Observation 2 *If the input contains a sequence of columns $i, \ldots, j$ such that $h(k) \geq \gamma h(k-1)$ for $k = i+1, \ldots, j$, then $\text{MF}(\{i, \ldots, j-1\}) = 0$.*

To determine the maximum optimal profit on a block, we need to consider how the type 2 blocks at the start of such a block are serviced. Depending on the exact heights of the nonzero columns, it may not be optimal to service them as described in Lemma 1. However, Observation 2 allows us to make the following assumption.

Assumption 1 *If columns $i, \ldots, j$ form a sequence of type 2 columns followed by a nonzero column j, then $h(k) = \gamma h(k-1)$ for $k = i+1, \ldots, j+1$.*

Here we conceptually round up column heights starting from the end, leaving $h(j)$ unchanged and defining $h(j-1), h(j-2), \ldots, h(i)$ by the equality in Assumption 1. This can only improve the total optimal value. Regarding MoreFilling, if the previous block ended with a type 1 column, its behavior on the immediately following type 2 columns (and hence the entire input) is unaffected by this change. If that block ended with a type 3 column, then the column following that was too high for the block to continue, and this still holds now.

Lemma 2 *For each column i, we have $a(i) = 0$ or $a(i) > h(i)/3$.*

Proof Suppose the optimal allocation for columns $i-1, i, i+1$ is $x, a(i), z$ (if one of these columns does not exist, assume that its height is zero), then we have $x = 0$ or $x = a(i)$ and we also have $z = 0$ or $z = a(i)$, so if $a(i) < h(i)/3$ we can replace $x, a(i), z$ by $0, h(i), 0$, which is always feasible. $\square$

We have the following lemma which will later simplify the analysis.

Lemma 3 *Let S be a sequence of type 2 columns $i, \dots, j$. Let the height of the final type 2 column be h. Let $V(S) := \text{OPT}(S) + \eta \cdot a(j)$. If $a(i) > 0$, then $V(S) \le \varepsilon\gamma^2 h$.*

Proof By Lemma 2 and Assumption 1, for any column $i' \ge 1$ in S, only three allocations then need to be considered: $0, h(i')/\gamma$, and $h(i')$.

Let x be the number of columns. If x is odd, the optimal allocation to S is given by Lemma 1, and this satisfies the constraint that the first column have nonzero allocation. For $x = 1$, the optimal profit on S is h. To this we add ηh to get $V(S)$, and we have $1 + \eta = \varepsilon\gamma^2$. For any odd $x \ge 3$ we have

$$V(S) = h + \frac{h}{\gamma^2} + \cdots + \frac{h}{\gamma^{x-1}}(2 - \frac{\mathcal{R}}{\gamma} + \varepsilon) \tag{2}$$

while for $x + 2$ this value becomes

$$h + \cdots + \frac{h}{\gamma^{x-1}} + \frac{h}{\gamma^{x+1}}(2 - \frac{\mathcal{R}}{\gamma} + \varepsilon)$$

which is less than (2) since $\gamma^2(2 - \frac{\mathcal{R}}{\gamma} + \varepsilon) > \gamma^2 + 2 - \frac{\mathcal{R}}{\gamma} + \varepsilon$. Therefore, $V(S) \le \varepsilon\gamma^2 h$ for all odd x.

For $x = 2$, given that the first column (of height $\frac{h}{\gamma}$) must have nonzero allocation, it is best to assign the same value to the second column, and we find $V(S) \le \frac{1}{\gamma^2}(3 - \frac{\mathcal{R}}{\gamma} + \varepsilon) < \varepsilon\gamma^2 h$ by inspection.

Given an instance with x type 2 columns (x is even), consider the instance with $x + 2$ type 2 columns. Let the height of column x be h'. If it was optimal to use column x fully, then it is now optimal to use column $x+2$ (of height $\gamma^2 h'$) fully as well. This adds $\gamma^2 h'$ to the total profit. The profit on the first x columns cannot be improved (by induction on even x), and there is no other assignment to column x that increases the profit on the last two columns to a value above $\gamma^2 h'$. Thus, we do not need to consider other allocations to column x in this case. Analogously to the case of odd x, we can now show that $V(S)$ decreases for increasing x. $\square$

Theorem 2 *MoreFilling is $\mathcal{R}$-competitive on a block which is* ***not*** *immediately followed by a type 3 column.*

Proof Let FIRST be the index of the first nonzero column in block b, and normalize $h(\text{FIRST}) = 1$. Then, the profit of our algorithm on this block is $|b|$ or $\beta|b|$, depending on whether $h(\text{FIRST}) \le h(\text{FIRST} + 1)$. By the text below Lemma 1, the optimal profit on the type 2 columns in b is at most $\varepsilon\gamma h(\text{FIRST})$ (the height of the last type 2 column is at most $h(\text{FIRST})/\gamma$, and if the last type 2 column is not used, the optimal profit on the other type 2 columns is at most $\varepsilon h(\text{FIRST})$, or at most $\eta h(\text{FIRST})$ if the block is long). For the calculations, we make the worst-case assumption that there is a type 1 column at the end of this block, and it has height 1 (it actually must have smaller height). The exception to this is a block of length 1; in that case, the type 1 column must have height at most $\beta h(\text{FIRST})$ by our algorithm.

Note that all nonzero columns in the block have height at least 1. To derive upper bounds for the optimal profit on a block, we may assume that each of these columns has the maximum height as bounded by Observation 1 (Figure 3). The only allocations that we need to consider to find the optimal profit are values that are the height of at least one column. In particular, for column FIRST, by Lemma 2 only the following allocations need to be considered: $h(\text{FIRST})$, $\frac{1}{\gamma}h(\text{FIRST})$, and possibly $\beta h(\text{FIRST})$.

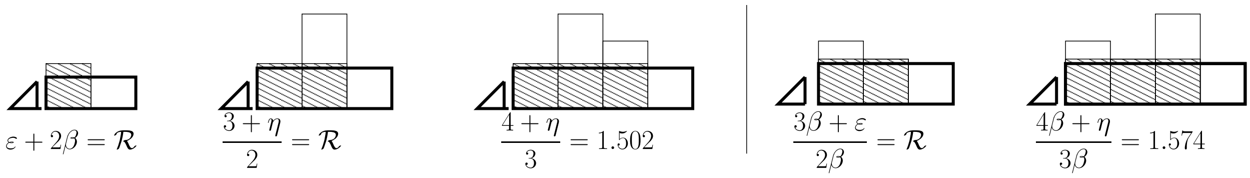

Figure 4: Possible profiles of blocks that are followed by type 1 columns. The first three diagrams show blocks which start with a fully-used column, the last two diagrams show blocks where the second column is fully-used. The triangles indicate preceding type 2 columns (that have geometrically increasing heights), of value at most $\varepsilon < 0.353$, or $\eta < 0.505$ for long blocks. The dashed blocks indicate blocks created by MoreFilling. Bold lines indicate the optimal solution for a block.

This gives us the results shown in Figure 4. The figure shows the optimal profit for each case and compares it to the profit of MoreFilling. Regarding a block of length 1 for instance, we find that it is optimal to set $a(\text{FIRST}) = a(\text{FIRST}+1) = h(\text{FIRST}+1) \leq \beta h(\text{FIRST})$, $a(\text{FIRST}-1) = 0$, and then it is possible to earn at most $\varepsilon h(\text{FIRST}) < 0.353 h(\text{FIRST})$ on the type 2 columns up to column $\text{FIRST}-2$.

For a block that contains three or more nonzero columns, note that adding a column of height δ (resp. $\delta\beta$ in case $h(\text{FIRST}+1) < h(\text{FIRST})$) adds at most δ (resp. $\delta\beta$) to the optimal profit and exactly 1 (resp. β) to the profit of MoreFilling. Since $\delta < \mathcal{R}$, this does not increase the competitive ratio above $\mathcal{R}$. $\square$

To complete our analysis, we now need to eliminate the type 3 columns. These columns are the most difficult to handle for the following reason. For type 1 columns, MoreFilling already decided in the previous time step (or earlier) to allocate zero to the type 1 column, because its height is too small compared to some previous height. Similarly, type 2 columns are allocated zero because their height is small compared to the immediately following height. For these cases, it is clear how to group the columns into blocks as described above (i.e., how to compensate for the missed profit on the zero column), and we can analyze these cases in a straightforward way as shown in Theorem 2.

In contrast, type 3 columns are allocated zero "at the last minute", and this decision does *not* depend on its own height. This is in particular troublesome in the case where column r is a type 3 zero column, and column $r-1$ was allocated $h(r) < h(r-1)$. In normal cases, column r would be allocated $h(r)$ to compensate for the fact that less than $h(r-1)$ was earned on column $r-1$. But now, column r is allocated zero, and in many cases we will have to consider the block following column r to complete the analysis.

For a type 3 column r, we denote the block immediately preceding it by $\text{BEFORE}(r)$ and the block immediately following it by $\text{AFTER}(R)$. We will abbreviate these blocks by BEFORE and AFTER if the meaning is clear from the context (which is most of the time). When analyzing column r, we always normalize the heights of the columns such that the height of the first nonzero column of the preceding block BEFORE is 1. Then $h(r) \in [\beta, \gamma]$, and if the first column of BEFORE is not fully used or the second column has height at least δ, we have $h(r) \leq \delta H$, where H is the base height of block BEFORE (else column $r-1$ would have been zeroed). We also have that

$$h(r+1) > \delta H. \tag{3}$$

Suppose a counterexample I exists, that shows that MoreFilling has a competitive ratio above $\mathcal{R}$. We next introduce a sequence of rules to handle type 3 columns. First we give an overview of the cases and the sections in which they are discussed.

Case	$a(r-1)$	$a(r)$	$a(r+1)$	other conditions	Section
1	$-$	0	$-$	$-$	3.2
2	>0	>0	>0	$\lvert\text{BEFORE}\rvert = 1$	3.3
3	>0	>0	$-$	Case 2 does not hold	3.4, Replacement rules 1–3
4	0	>0	$-$	$\lvert\text{BEFORE}\rvert > 1$	3.4, Replacement rule 4
5	0	>0	0	$\lvert\text{BEFORE}\rvert = 1$	3.4, Replacement rule 5
6	0	>0	>0	$\lvert\text{BEFORE}\rvert = 1$	3.5

Note that Cases 2 and 3 together cover all the cases where $a(r-1) = a(r) > 0$, and Case 4 covers all remaining cases where $|\text{BEFORE}| > 1$. Hence Cases 5 and 6 complete the analysis.

In Case 1, we consider column r to be a part of block BEFORE, but we assign some of the optimal profit on BEFORE to AFTER to ensure that MF is competitive on BEFORE (Assignment rules 1–2). We ensure that the profit assigned to AFTER is relatively small compared to the profit of MoreFilling on AFTER. Importantly, in this step we remove *all* type 3 zero columns which have a zero optimal allocation (remove in the sense that we do not consider them to be type 3 columns anymore in the rest of the analysis).

We analyze Case 2 directly in Section 3.3, usually by calculating the competitive ratio on the blocks BEFORE and AFTER together. We then consider a number of cases where we show that the input can be split into two independent parts such that MoreFilling is competitive on both parts (after some reassignment) These cases are distinguished by how much of the optimal profit is reassigned from the first part to the second part. Finally, one remaining case (Case 6) is shown to either reduce to one of the cases already considered (sometimes with minor changes in the values used in the calculations), or analyzed directly.

3.2 The case $a(r) = 0$

Assignment rule 1 *If $a(r) = 0$ and $h(r+1) \geq \gamma h(r)$, we assign column r as a* ***type 2*** *zero column to* AFTER*. Additionally, assign $\varepsilon\gamma h(r)$ of the optimal profit on* BEFORE *to* AFTER*.*

As long as we have $|\text{BEFORE}| \geq 2$ or $h(r) \geq h(r-1)$, this works because the allocation of MoreFilling to column $r-1$ was not adversely affected by column r. That is, $u(r-1)$ was not set to a value below $h(r-1)$ because it was followed by a column of height $h(r)$ (but possibly because it was preceded by other columns with nonzero allocation), and thus block BEFORE can be analyzed independently of column r, i.e., as if the input ended at column $r-1$. (We can just ignore the fact that some of the optimal profit on BEFORE was assigned to AFTER in this case, and apply Theorem 2 on BEFORE.)

Moreover, the profit of $\varepsilon\gamma h(r)$ is our upper bound from (1) for what the optimal solution could earn on AFTER from additional type 2 columns before column r (i.e. of heights $h(r)/\gamma, h(r)/\gamma^2, \ldots$ (from right to left)). Since these extra type 2 columns are not in fact present, we can reassign this profit from BEFORE to AFTER. We will use the same idea later in our replacement rules (Section 3.4).

Suppose $|\text{BEFORE}| = 1$ and $h(r) < h(r-1)$. Normalize such that $h(r-1) = \text{MF}(\text{BEFORE}) = 1$. Then $\text{MF}(\text{BEFORE}) = h(r) \geq \beta$. Since $a(r) = 0$, we have $\text{OPT}(\text{BEFORE}) \leq h(r-1) + \varepsilon h(r-1) - \varepsilon\gamma h(r) \leq 1 + \varepsilon - \varepsilon\gamma\beta < 1$, so $\text{OPT}(\text{BEFORE})/\text{MF}(\text{BEFORE}) < 1/\beta < \mathcal{R}$.

Assignment rule 2 *If $a(r) = 0$ and $h(r+1) < \gamma h(r)$, then if $|\text{BEFORE}| \geq 2$ or $h(r) \geq h(r-1)$, we set $h(r) = 0$ and assign column r as a* ***type 1*** *column to* BEFORE*. If $|\text{BEFORE}| = 1$ and $h(r) < h(r-1)$, assign column r to* BEFORE *and assign $\delta\varepsilon h(r)$ of the optimal profit on* BEFORE *to* AFTER*.*

We explain why this rule works. Consider a column r for which $a(r) = 0$. First of all, note that the type (or the height) of a zero column does not affect the behavior of MoreFilling on subsequent columns.

If $|\text{BEFORE}| \geq 2$ or $h(r) \geq h(r-1)$, our modification does not change the profit of MoreFilling on the modified block BEFORE (as in Assignment Rule 1), and the optimal value for I does not decrease since $a(r) = 0$. Thus MoreFilling is competitive on BEFORE (including column r, seen as a type 1 column) by Theorem 2.

Suppose $|\text{BEFORE}| = 1$ and $h(r) < h(r-1) = 1$. By rule 3 of our algorithm, we have $h(r+1) > \delta h(r)$. On hypothetical type 2 columns preceding column $r+1$, OPT could hence earn at least $\varepsilon h(r+1) > \delta\varepsilon h(r)$, and we assign $\delta\varepsilon h(r)$ of profit to AFTER. Now $\text{MF}(\text{BEFORE}) = h(r) \geq \beta$, and using $a(r) = 0$ we get

$$\text{OPT}(\text{BEFORE}) \leq (1+\varepsilon)h(r-1) - \delta\varepsilon h(r) = 1 + \varepsilon - \delta\varepsilon h(r) \leq 1 + \varepsilon - \delta\varepsilon\beta < 1 < \mathcal{R}\beta.$$

3.3 The case $|\text{BEFORE}| = 1$ and $a(r-1) = a(r) = a(r+1) > 0$

We treat this special case separately; we will encounter a very similar case later.

Case 1: $h(r) < h(r-1)$ Consider column $r+1$. If it is of type 2, then we must have $a(r+2) = 0$ by Lemma 2, since $h(r+2) > \gamma h(r+1) > \gamma\delta h(r) \geq \gamma\delta a(r) > 3a(r)$. But the total optimal profit on columns r and $r+1$ is $2a(r)$, whereas by Lemma 1 the total possible profit on type 2 columns preceding column $r+2$ is $\frac{\gamma^2}{\gamma^2-1}h(r+1) > 2.05h(r+1) \geq 2.05a(r)$. So we can assign these columns to AFTER.

If column $r+1$ is nonzero, then the profit of $2a(r)$ is less than $(1+\varepsilon)h(r+1)$ because that is at least $(1+\varepsilon)\delta a(r) = 2.12a(r)$. Hence we now find a smaller optimal profit in all cases where it was previously optimal to assign the first column of the block its full height, or where it was assigned zero. For the remaining cases, we have $a(r) \leq h(r) \leq \beta$. On BEFORE, MF earns at least $h(r)$, so we can assign $\mathcal{R}h(r)$ of profit there, in particular $(\mathcal{R}-1)h(r) - \varepsilon$ from column r, and are then left with at most $(2-\mathcal{R})h(r) - \varepsilon$ for AFTER. In all these remaining cases, in the optimal solutions which we found before, the first and second column of the block were both used, giving a profit of $2\beta + \varepsilon$ on those columns and the preceding type 2 columns, which in the current situation is at least $(2\beta+\varepsilon)\delta h(r) = 2.752h(r)$ using that $h(r+1) > \delta h(r)$. Now, we find a profit of at most $(2-\mathcal{R})h(r) - \varepsilon + 2h(r) = 2.248h(r) - \varepsilon$ there which is less.

Case 2: $h(r) \geq h(r-1)$ If $h(r) < \min(h(r+1), \gamma)$, we may increase $h(r)$ to this value without affecting the behavior of MoreFilling or decreasing the optimal profit. Then $h(r) > \delta$.

If now we no longer have $a(r-1) = a(r) = a(r+1) > 0$ (i.e., this is no longer optimal), we are done. Else, if $a(r+2) = 0$, we have a contradiction, since it is better to set $a(r-1) = 0$ and earn more than $2\delta > 3$ on the columns $r, r+1$.

If $h(r+2) \geq \delta$, we find analogously $a(r+3) = a(r+2) = a(r+1)$. If column $r+2$ is of type 1, $\text{MF}(\text{BEFORE} \cup \text{AFTER}) = \text{MF}(\{r-1, r+1\}) \geq \delta/\beta + 1$ for a competitive ratio of less than 1.25 on these two blocks. Else, if $h(r+3) \geq \delta$ as well, we have a contradiction in any event, since it is possible to earn $4\delta > 6$ by setting $a(r-1) = 0$ and $a(r+4) = 0$, which is more than this optimal solution earns on columns $r-1, \ldots, r+4$, a contradiction. Therefore $h(r+3) < \delta < h(r+1)$, and column $r+3$ is of type 1, regardless of whether column $r+1$ is nonzero or of type 2.

If column $r+1$ is of type 2, then $\text{MF}(\text{BEFORE} \cup \text{AFTER}) > 1 + \delta\gamma > 4$ (Figure 5(a)). If column $r+1$ is nonzero, we have $\text{OPT}(\text{BEFORE} \cup \text{AFTER}) = 5 + \varepsilon$ and $\text{MF}(\text{BEFORE} \cup \text{AFTER}) \geq 1 + 2\delta > 4$ (Figure 5(b)). The ratio is less than 1.5 in both cases.

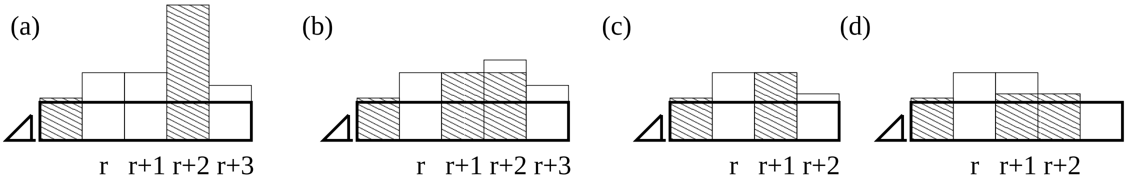

Figure 5: A special case: $|\text{BEFORE}| = 1$ and $a(r-1) = a(r) = a(r+1) > 0$. We show the first four subcases of $h(r) \geq h(r-1)$. Diagrams (a)–(b) represent $h(r+2) \geq \delta$, diagram (c) the case in which $r+2$ is of type 1, and diagram (d) the case that $r+2$ is nonzero.

We are left with the case $h(r+2) < \delta$. If $h(r+2) < \delta\beta$, column $r+2$ is of type 1, and the competitive ratio on BEFORE ∪ AFTER is at most

$$\frac{\varepsilon + 4}{1 + \delta} = 1.694 \quad \text{(Figure 5(c))}.$$

Else, if column $r+2$ is nonzero, we get a ratio of at most

$$\frac{\varepsilon + 5}{1 + 2\beta\delta} = 1.674 \quad \text{(Figure 5(d))}$$

(less if $|\text{AFTER}| > 2$). Finally, if column $r+2$ is of type 3, we have $h(r+3) > \delta h(r+2) > \delta^2\beta = 1.726$ whereas $a(r+2) = a(r-1) \leq 1$. On hypothetical type 2 columns preceding column $r+3$, OPT could earn $\varepsilon\gamma h(r+3) > 1$. Therefore we can assign $a(r+2)$ completely to the block following AFTER, and have a competitive ratio of

$$\frac{\varepsilon + 3}{1 + \beta\delta} = 1.598$$

on the blocks BEFORE and AFTER.

3.4 Replacement rules

For the remaining cases, we now introduce a set of rules that split the hypothetical counterexample I in two parts, and show that MoreFilling is $\mathcal{R}$-competitive on both parts.

Replacement rules (Table 1) Split I into two independent parts, I_1 and I_2. Part I_1 ends with column $r-1$. Replace column r by an unbounded sequence S of columns of heights $\min(\gamma a(r), h(r+1))/\gamma^i$ $(i = 1, 2, \dots)$, which appear in order of increasing height. Part I_2 consists of sequence S followed by columns $r+1$ and beyond. All columns of S are of type 2. Note that $h(r+1) > \delta \cdot \text{BASEHEIGHT}$. Finally, add some amount to the optimal profit on I_1 as described in Table 1. Let $f = 2$ if the second nonzero column of BEFORE has height less than the first one, else $f = 1$. The first column of BEFORE has height 1, and we abbreviate its BASEHEIGHT by H.

Rule 1: $a(r-1) = a(r), f = 2, |\text{BEFORE}| = 2$ See Figure 6(c).

Rule 2: $a(r-1) = a(r), f = 1, a(r+1) = 0, |\text{BEFORE}| = 1$ In this case, the optimal profit on S is at least $\varepsilon\gamma\delta a(r) > a(r)$, so nothing is left to assign to I_1. See Figure 6(a). Note that the alternative subcase $a(r+1) > 0$ is already handled by Assignment Rule 3.

Rule	Condition	$a(r)$	Optimal profit on S	Profit assigned to I_1
1	$a(r-1) = a(r), f = 2, \lvert\text{BEFORE}\rvert = 2$	β	$\varepsilon h(r+1) > \varepsilon\delta\beta$	$\beta(1-\varepsilon\delta)$
2	$a(r-1) = a(r), f = 1, \lvert\text{BEFORE}\rvert = 1,$ $a(r+1) = 0$	1	$\varepsilon\gamma h(r+1) > \varepsilon\gamma\delta$	0
3	$a(r-1) = a(r),$ other cases	δH	$\varepsilon h(r+1) > \varepsilon\delta H$	$\delta(1-\varepsilon)H$
4	$a(r-1) = 0, \lvert\text{BEFORE}\rvert > 1$	γ	$\varepsilon h(r+1) > \varepsilon\delta$	$\gamma - \varepsilon\delta$
5	$a(r+1) = a(r-1) = 0, \lvert\text{BEFORE}\rvert = 1$	γ	$\varepsilon\gamma h(r+1) > \varepsilon\gamma\delta$	$\gamma(1-\varepsilon\delta)$

Table 1: The five replacement rules. The third column in the table contains an upper bound on $a(r)$ for each case. The fourth column contains the optimal profit on S, where we have assumed that the schedule on the remaining columns (all columns but column r) is unchanged. In particular, if $a(r+1) = a(r) > h(r)/\gamma$, the final column in S cannot be used in such a schedule, but if $a(r+1) = 0$, it can be used, thus multiplying its value for the optimal solution by γ. Finally, the final column is an upper bound for the difference between the two preceding columns.

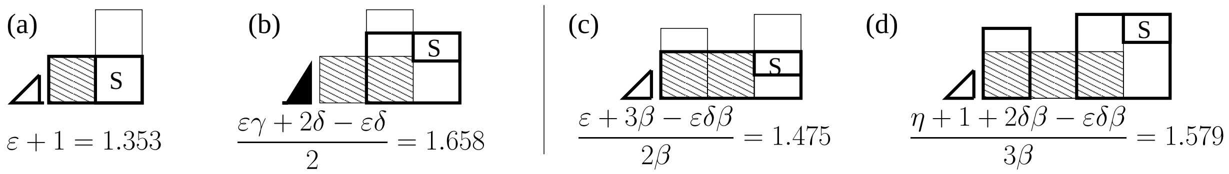

Figure 6: Possible profiles of blocks for the case $a(r-1) = a(r)$. The first two diagrams show blocks which start with a fully-used column, the last two diagrams show blocks where the second column is fully-used. For an explanation of the symbols, see Figure 4. The rectangles marked S are not assigned to the block, but instead to the following block. See Replacement rules. In diagram (a), the case where $a(r+1) = 0$ is shown (Replacement rule 2). The case of a block of one column that is not fully allocated is handled by Assignment rule 2.

Rule 3: $a(r-1) = a(r)$**, other cases** If BEFORE has no fully-used column, then $\text{OPT}(\text{BEFORE}) \le \mathcal{R}$ by Assignment Rule 1. Else, we have $h(r) \le \delta$, or column $r-1$ would be a zero column by Rule 3 of our algorithm. Since $a(r) \le h(r) \le \delta$, we only need to assign $\delta(1-\varepsilon)$ of profit to I_1, even if $a(r+1) > 0$. See Figure 6 for the optimal schedules in the various cases. As before, it can be seen that adding additional columns of height δ does not increase the competitive ratio.

Rule 4: $a(r-1) = 0, \lvert\text{BEFORE}\rvert > 1$ In this case we can only be sure of a profit of $\varepsilon\delta$ that can be assigned to I_2. However, this is sufficient in these cases. See Figure 7((b)–(d)).

Rule 5: $a(r+1) = a(r-1) = 0, \lvert\text{BEFORE}\rvert = 1$ Note that the profit assigned to I_1 is less than 1, and there is now no type 1 column at the end of BEFORE. We use that $a(r-1) = 0$. If $\lvert\text{BEFORE}\rvert = 1$ and $u(r-1) = h(r-1)$, then since $a(r-1) = 0$, the optimal profit is now at most $\varepsilon\gamma + \gamma(1-\varepsilon\delta) = 1.649$, and MoreFilling earns 1. See Figure 7(a).

This only leaves one case open, which is handled in the following section.

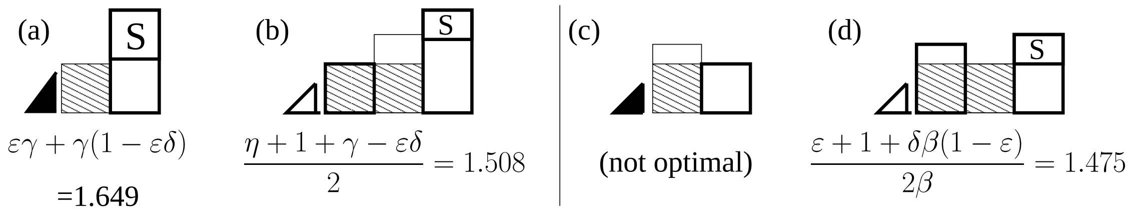

Figure 7: Possible profiles of blocks for the case $a(r-1) = 0$ (in diagram (a), $a(r+1) = 0$ as well). Black triangles are type 2 columns of total value at most $\varepsilon\gamma < 0.729$. In diagram (c), the shown bold allocation is in fact not optimal, because it is better to set $a(r-1) = a(r) = \beta$. Thus, this case is not relevant. Finally, longer blocks do not give a ratio above $\mathcal{R}$.

3.5 The final case

Due to our calculations so far, we only need to deal with the following remaining case:

$$|\text{BEFORE}| = 1, u(r-1) = h(r-1), a(r-1) = 0 \text{ and } a(r+1) = a(r) > 0. \tag{4}$$

(The case $u(r-1) = h(r) < h(r-1)$ was handled by Assignment Rule 2.) In this unresolved case, the first question we have is how much of the optimal profit on column r must be assigned to block AFTER (since if it were all counted as part of the optimal profit on block BEFORE, MoreFilling would not be competitive on BEFORE). Let x be the number of type 2 columns immediately following column r, and let r_2 be the index of the zero column immediately following $|\text{AFTER}|$.

Lemma 4 *The profit that is assigned forward from column r in the case described in (4) is at most* $(1 - \frac{\mathcal{R}}{\gamma} + \varepsilon)a(r) = \eta a(r)$. *Normalizing such that the first nonzero column of* AFTER *has height 1, this is at most* η/γ^x.

Proof We have $a(r) \le \min(h(r), h(r+1))$. We assign $(\mathcal{R}/\gamma - \varepsilon)a(r) \le (\mathcal{R}/\gamma - \varepsilon)h(r) \le (\mathcal{R} - \varepsilon\gamma)h(r-1)$ to BEFORE (to add to the $\varepsilon\gamma h(r-1)$ from the type 2 columns) and are left with $(1 - \frac{\mathcal{R}}{\gamma} + \varepsilon)a(r) \le (1 - \frac{\mathcal{R}}{\gamma} + \varepsilon)h(r+1)$. The lemma follows because the sequence of type 2 columns has (at least) geometrically increasing heights. In particular, if $x = 0$, we use that $a(r) \le h(r+1)$, and $h(r+1) = 1$ in this case. $\square$

Using Lemma 3, we have that if $x \ge 1$, we get exactly the value $\varepsilon\gamma$ for the type 2 columns plus this forward assignment from column r, which is the value that we have been calculating with in case the last type 2 column had nonzero allocation in the optimal solution. Thus, we get exactly the same analysis as before, with the difference (for $x = 1$) that OPT is required to use the type 2 column by assumption. Clearly, if we remove this requirement (and generally use the bound $\varepsilon > \eta/\gamma$ in case OPT does not use the final type 2 column), we can only get higher bounds for the competitive ratio. Hence, our previous analysis holds unless $x = 0$, i.e., there are no type 2 columns before AFTER. (Of course, if for AFTER we encounter the open case that we are discussing in this section, we just repeat; eventually we reach the end of the sequence or find a case which we can prove.) In the case $x = 0$, the optimal solution must use the first nonzero column of AFTER (as well as the preceding column). We summarize this discussion in the following lemma.

Lemma 5 *In the remaining open case, column $r + 1$ is the first nonzero column of block* AFTER*, and* $a(r+1) = a(r)$.

Lemma 6 *MoreFilling is competitive on* AFTER *even after taking the forward assignment from column r into account, unless* $|\text{AFTER}| = 1$ *and* $a(r-1) = a(r) = a(r+1)$.

Proof If AFTER is a long block, we already calculated with an optimal profit of at least η on the type 2 columns in all cases and are done. Else if AFTER is followed by a type 3 column r_2, we see in Figures 6 and 7 that MoreFilling is still $\mathcal{R}$-competitive if the profit on preceding columns is η instead of ε (or $\varepsilon\gamma$) apart from in the excluded case. If AFTER is not followed by a type 3 column, then if $|\text{AFTER}| = 1$ we get a ratio of at most $(2\beta + \eta\beta) = \mathcal{R}$, using that $a(r+1) = a(r) \leq \beta$ and Lemma 4. If $|\text{AFTER}| = 2$ and $f = 2$, then if $a(r+1) = a(r+2)$ as in Figure 4, we find a ratio of at most $(3a(r) + \eta a(r))/(2a(r)) = \mathcal{R}$ since now $a(r) \leq u(r+2)$. Else, the ratio is at most $(\eta + 1 + \beta)/(2\beta) < 1.6$. $\square$

We are left with the case $|\text{AFTER}| = 1$, so $r_2 = r + 2$ and $a(r-1) = \cdots = a(r+2)$ since type 3 columns have nonzero optimal allocation. Since r_2 is of type 3,

$$h(r_2 + 1) > \delta h(r_2 - 1) = \delta h(r + 1) \geq \delta a(r). \quad (5)$$

If $a(r_2 + 1) = 0$, we apply our Replacement rule 2, i.e., column r_2 is assigned completely to the block following AFTER. This works because on (hypothetical) type 2 columns preceding column $r_2 + 1$, OPT could earn $\varepsilon\gamma h(r_2 + 1) > h(r_2 + 1)/\delta > a(r_2)$. We then have $\text{OPT}(\text{AFTER}) \leq 1 + (1 - \frac{\mathcal{R}}{\gamma} + \varepsilon) < 1.51$.

Else, we are again in the case discussed in Section 3.3, since $a(r_2 - 1) = a(r_2) = a(r_2 + 1)$. The only difference is that the profit from previous columns is not ε but η. We can therefore follow that analysis completely and adjust the calculations where necessary. The highest ratio that we found there is $(4+\varepsilon)/(1+\delta)$, and this now becomes $(4 + \eta)/(1 + \delta) = \mathcal{R}$.

4 Lower bounds

4.1 Deterministic lower bound

We let $\varepsilon > 0$ be some very small value. Denote the online algorithm by ALG and assume that it has a competitive ratio of at most $(1 - \varepsilon)1.69595$. In our lower bound construction, the adversary uses the following strategy. See Figure 8. The values p and q are fixed constants which we will determine later; we have $p < 1$ and $q > 2$. Each rectangle represents a state. The states consist of three components: the current online allocation, the current column, and the next column. Hence, if we move from one state to another, the second column in the starting state is identical to the first column in the destination state. However, we sometimes (conceptually) rescale column heights, for instance when we move from state B to state A.

Additionally, the number $(1-\delta)x$ in the ellipse indicates that one final column of height $(1-\delta)x$ arrives, where x is the (nonzero) allocation that ALG just used. Hence, this column has no value for ALG (it can only use the column by forfeiting a profit of x on the current column). Overall, we have the following states.

State	Description
A	The online algorithm allocated 0 to the previous column (or there is no previous column). The visible input is $1, q$ (current column, next column).
B	The online algorithm allocated a nonzero value to the previous column. The visible input is q, q.
C	The online algorithm allocated 0 to the previous column. The visible input is $1, p$.
D	The online algorithm allocated a nonzero value x to the previous column. A new column of height $(1 - \delta)x$ arrives.
STOP	The input ends: the visible input is the last column of the previous state.

In total, there are in fact 8 possible states (counting D as one state), since states A–C occur in two forms: depending on the behavior of the online algorithm, the adversary may once (permanently) change the values

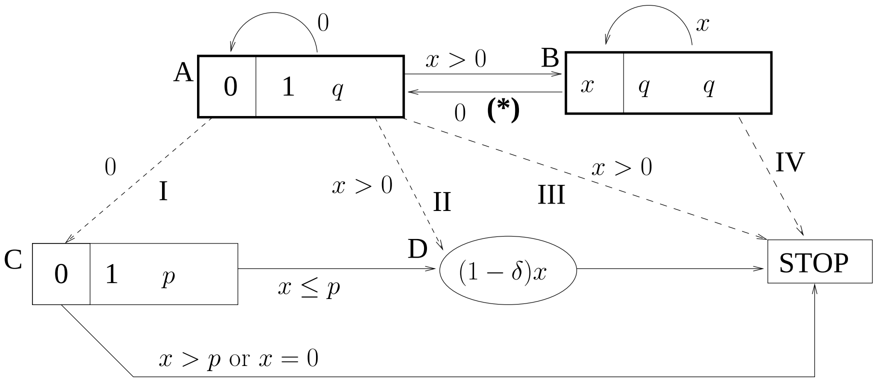

Figure 8: Lower bound strategy. The first number in states A–C represents the last allocation of ALG and the other two are the two currently visible columns. The variable x always indicates the allocation chosen by ALG. At the state change marked with (*), we *may* change q to $q' = 2.03644$ and p to $p' = 1/q'$. State change I (resp. IV) occurs if ALG remains in state A (resp. B) "too long" (see Claims 2, 3). State changes II and III occur if ALG allocates a "too low" value x in state A (Lemmas 7, 8).[1] The precise conditions for these state changes are given in the mentioned claims and lemmas; we also show that one of the state changes I-IV always occurs eventually. State change IV can happen both on an allocation of 0 and on an allocation of x.

of q and p to q' and p', respectively.

The input The input starts in state A. The default state changes are given by the normal arrows. As ALG is processing the input, moving between states A and B, the adversary keeps track of the total online profit so far, as well as of the total optimal profit. When going from state B to A, the adversary may elect to change the value of q to $q' = 2.036442$ and p to $p' = 1/q'$ (for future columns); we will specify when this happens. In both state A and B, nonstandard state changes are possible, which are indicated by dotted lines in Figure 8. The adversary forces such a state change as soon as this gives the desired lower bound. Note that in all of the cases where a dotted line is followed, the input consists of at most two more columns, and it is straightforward to calculate the resulting competitive ratio, based on the stored optimal and online profits.

For instance, if ALG allocates a certain value $x > 0$ in state A or B, and the adversary sees that this is too low to maintain the competitive ratio (because of earlier choices), the input sequence stops immediately. In state B, this may also happen when the algorithm allocates 0; hence, there is no condition next to the outgoing dotted line from B. In state A, instead of stopping immediately, one final column of height $(1-\delta)x$ may arrive, so the adversary has two options here (besides the default state change to B) if ALG allocates a nonzero value.

What we need to show is that no matter what ALG does, at some point the adversary can choose a nonstandard state change and prove the desired competitive ratio: ALG cannot stay in states A and B forever.

Output of the adversary We first describe how the adversary handles the input as long as ALG stays in A and B. This is very simple: in state A, it allocates 0 if ALG allocates $x > 0$, and alternates between the full column height and 0 otherwise (ending with 0 when ALG allocates a nonzero value). In state B, it always

[1] If the conditions given in Lemma 7 and Lemma 8 are both satisfied, the adversary could force either change; to fully specify the lower bound strategy, we say that it uses state change III in such cases.

allocates the full column height, except for one special case when the input is about to end. In that case, the allocation in state B will be the same as on the columns following it.

Claim 1 *This output is a feasible solution.*

Proof We only need to show that the adversary does not allocate two different nonzero values to adjacent columns. Given the above rules, this could only happen at a state change. In a move from A to B, the adversary always assigns 0. In a move from B to A (say at column i), the adversary allocates the full column height, and *may* also do this on column $i+1$, depending on whether there is an odd or an even number of consecutive state A columns following. But columns i and $i+1$ have the same height here. $\square$

Example We first give an example that shows a lower bound of 1.69. Here we assume that ALG allocates the highest possible amount in each time that avoids a competitive ratio of 1.69, and always allocates 0 in state B. We set $q = 2.169$ and $p = 0.710$.

Input	1	q	q	q^2	q^2	q^3	q^3	q^4	q^4	q^5	q^5	q^6	q^6	q^7	q^7	q^8
ALG	1	0	q	0	q^2	0	q^3	0	$0.98q^4$	0	$0.94q^5$	0	$0.88q^6$	0	$0.76q^7$	$0.76q^7$
OPT	0	q	0	q^2	0	q^3	0	q^4	0	q^5	0	q^6	0	q^7	0	q^8

It can be checked that ALG cannot allocate more in any step in state A (state change II would then occur); if it allocates less in any step, the sequence remains unchanged, and ALG has simply earned even less at the end. We give the calculations in the following table. Here $\text{ALG}(i)$ is the profit of ALG on phases $1, \dots, i$. We have $\text{OPT}(i-1)$ in the table because the allocation in the last column of the previous phase might still change depending on the allocation of ALG to the new column (i.e., it could be changed to $(1-\delta)x$). After ALG allocates $0.76q^7$ in phase 8, the input simply stops (state change III), since the goal ratio of 1.69 is already achieved.

Phase i	$\text{ALG}(i)$	$\text{OPT}(i-1)$	Column height	Allocation (fraction)
1	1	0	1	1
2	3.169	2.169	2.169	1
3	7.873	6.873	4.704	1
4	18.07	17.07	10.20	1
5	39.81	39.21	22.13	0.982
6	85.12	87.21	48.00	0.943
7	176.5	191.3	104.1	0.877
8	349.1	417.1	225.8	0.764
–	521.7	907.0	489.8	0.764

For example, if ALG allocates $q^4 = 22.13$ in phase 5, one final column of height $(1-\delta)q^4$ arrives (state change II). Then ALG earns $2 \cdot 22.13 + \text{ALG}(4) = 62.33$ and OPT earns $4 \cdot 22.13 + \text{OPT}(3) = 88.53 + 17.07 = 105.60$ as $\delta \to 0$, for a ratio of 1.694.

In our general construction, we will set $q = 2.14447$ and

$$p = \frac{3q^2 - 5 + \sqrt{9q^4 - 14q^2 + 9}}{8q^2 - 8} = 0.709039. \tag{6}$$

Claim 2 *For each visit of* ALG *to state A and every $\varepsilon > 0$, the number of consecutive steps in which* ALG *can allocate 0 before the adversary can force a competitive ratio of $(1-\varepsilon)\mathcal{R}$ by sending it to C is bounded.*

Proof Consider the ith visit of ALG to state A ($i \geq 1$). Let the online and optimal profit *before* this visit be ALG_i and OPT_i, respectively ($\text{ALG}_1 = \text{OPT}_1 = 0$). We want to calculate the competitive ratio if the adversary sends ALG to C when ALG assigns 0 after staying in A for N time steps, where N is even and large (state change I in Figure 8). We rescale all the column heights such that the columns in state C have heights exactly 1 and $p < 1$. During the steps that ALG spends in state A on its ith visit, the column heights increase exponentially. Hence, for each $i \geq 1$ and each real $\delta > 0$, there exists an $N(\delta)$ such that $\text{ALG}_i \leq \delta$ for $N \geq N(\delta)$.

If ALG spends N steps in state A, then on the first $N-1$ columns in state A, the adversary earns at least $\text{OPT}_i + \sum_{j=1}^{N/2} q^{-2j}$. This tends to $\text{OPT}_i + 1/(q^2-1)$ for large N. Hence, for every $i \geq 1$ and each real $\delta > 0$, there exists an $N'(\delta)$ such that the adversary earns at least $(1-\delta)/(q^2-1)$ for $N \geq N'(\delta)$.

Let $N \geq \max(N(\delta), N'(\delta))$. In state C, ALG must choose an allocation x. If $x > p$ or $x = 0$, the input stops. The total online profit is at most $1 + \text{ALG}_i \leq 1 + \delta$, and the optimal profit is at least $2p + (1-\delta)/(q^2-1)$. If $0 < x \leq p$, the next and final column has height $(1-\delta)x$. The online profit is at most $2x + \text{ALG}_i \leq 2x + \delta$, whereas the optimal profit is at least $3(1-\delta)x + (1-\delta)/(q^2-1)$. As a function of x, the competitive ratio in this case is minimized by taking $x = p$. For $\delta \to 0$, the competitive ratios in these two cases tend to

$$\mathcal{R}_1 = \frac{3p + 1/(q^2-1)}{2p} \qquad x > p \vee x = 0 \tag{7}$$

$$\mathcal{R}_2 = 2p + 1/(q^2-1) \qquad 0 < x \leq p. \tag{8}$$

These ratios are equal if we set p as in (6), and are then both $\mathcal{R} = 1.69595$. As stated, both ratios tend to this value as $\delta \to 0$. Hence, for any given $\varepsilon > 0$ and $i \geq 1$, we can find $\delta > 0$ and $N \geq \max(N(\delta), N'(\delta))$ such that the implied competitive ratio is at least $(1-\varepsilon)\mathcal{R}$ if ALG spends at least N steps in state A on its ith visit and the adversary then sends it to C. □

Claim 3 *For each visit of* ALG *to state B and every $\varepsilon > 0$, the number of consecutive steps in which* ALG *can allocate x before the adversary can force a competitive ratio of $(1-\varepsilon)\mathcal{R}$ by sending it to C is bounded.*

Proof In each step in state B, given the above strategy of the adversary, the ratio of the adversary's profit to the online profit is at least $q > 2$. Thus, after sufficiently many consecutive columns in state B (depending on the profit of ALG and of the adversary before the current visit to B), the overall competitive ratio gets arbitrarily close to $q > 2 > 1.69595$. □

Phases Given the previous two claims, we know that ALG must move back and forth between states A and B, and cannot stay in one state indefinitely. We partition the input into *phases*. A new phase starts at the start of the input and whenever ALG arrives in state A from state B. We say that a column i is of state A (B) if ALG is in state A (B) when it allocates the value for column i. Hence a phase always ends with a state B column in which ALG allocates 0, unless the input stops in state B. In each phase i, ALG allocates exactly one nonzero value (namely when it moves from state A to state B), possibly to some consecutive columns.

Definition 3 *Let $0 < x_i \leq 1$ be the* fraction *of the height of the first column to which* ALG *allocates a nonzero value in phase i.*

Claim 4 *In each phase but the last, the adversary earns at least q times as much as* ALG*. This profit ratio is monotonically nondecreasing as a function of the number of state A columns in the phase.*

Proof For a given phase j, scale the column heights in this phase so that the last column (say column i) in state A has height 1. Let the number of columns in state B be $b \geq 1$. Then ALG earns $bx_j \leq b$ whereas the adversary earns at least bq, namely on columns $i+1, \ldots, i+b$. The end of this phase looks as follows. Allocations that cause state changes are underlined.

Input	$\ldots$	q^{-3}	q^{-2}	q^{-1}	1	q	$\ldots$	q	q
State	A	A	A	A	A	B	$\ldots$	B	B
ALG	$\ldots$	0	0	0	$\underline{x_j}$	x_j	$\ldots$	x_j	$\underline{0}$
OPT	$\ldots$	q^{-3}	0	q^{-1}	0	q	$\ldots$	q	q
						b columns			

(9)

This proves that the profit ratio is at least q and grows as a function of the number of state A columns. $\square$

Corollary 1 *W.l.o.g.,* ALG *never stays in state B for more than one column.*

Proof On all state B columns, the adversary earns at least $q > 2$ times as much as ALG, as can be seen in Table (9). If ALG has a competitive ratio of at most $\mathcal{R} < 2$, then it certainly also has this ratio if it always leaves state B immediately, instead of staying there for some steps. Thus avoiding extra state B columns can only help ALG; note such columns do not affect the column heights of future columns. $\square$

Let ALG(j) (OPT(j)) be the total profit of ALG (OPT) after j phases, scaled such that the last state A column in phase j has height 1.

Claim 5 *As $j \to \infty$,* OPT$(j) \to q^3/(q^2-1) = 2.74$ *or more.*

Proof Given that the last state A column in phase j has height 1, the height of all the previous columns would be minimized if they were all of state A: moving backwards from the last column, every time there is a column of state A the height is divided by q, whereas it remains constant in columns of state B. In this case, shown in Table (9), OPT(j) is at least $q + q^{-1} + q^{-3} + \ldots$ which tends to $q^3/(q^2-1)$. Any state B columns that occur can only increase the optimal profit, since the column heights increase. $\square$

Lemma 7 *For all $j \geq 1$, if* ALG *is $\mathcal{R}$-competitive, we have $x_j > q/(2\mathcal{R}) > 0.6$.*

Proof We consider the competitive ratio if the input stops on a nonzero allocation x_j in state A (state change III in Figure 8). The adversary earns q on the final column of the input, whereas ALG earns at most another x_j there. By Claim 4, the competitive ratio is at least

$$\frac{q + q \cdot \text{ALG}(j-1)}{2x_j + \text{ALG}(j-1)} \geq \mathcal{R} \text{ for } x_j \leq \frac{q}{2\mathcal{R}} .$$

(For $j = 1$, we have ALG$(j-1) = 0$.) $\square$

We now first deal with the case where after an arbitrary prefix, ALG never spends two consecutive steps in the same state, thus from some point onwards each phase is of the form AB. We then get an input and output of the following form (using the adversary strategy defined above). Here we normalize such that the first column height in phase n is 1.

Input	q^{1-n}	q^{2-n}	q^{2-n}	q^{3-n}	$\ldots$	q^{-1}	1	1	q
State	A	B	A	B	$\ldots$	A	B	A	B
Phase	1		2		$\ldots$	$n-1$		n	
ALG	$x_1 q^{1-n}$	0	$x_2 q^{2-n}$	0	$\ldots$	$x_{n-1} q^{-1}$	0	x_n	0
OPT	0	q^{2-n}	0	q^{3-n}	$\ldots$	0	1	0	q

The following observation is immediate from the table and $x_i \in (0, 1]$.

Observation 3 *If* ALG *allocates a nonzero amount (not necessarily the full column height!) whenever it is in state A, and zero whenever it is in state B, it earns at most $\sum_{i=0}^{n-1} q^{-i}$ during n phases (if the first column height in phase n is 1), while it is possible to earn $q\sum_{i=0}^{n-1} q^{-i}$ in these phases.*

Suppose ALG acts as in Observation 3 for N consecutive phases, starting from some column i_0 in state A. We will ignore all previous phases in our calculations and only give bounds for the competitive ratio of ALG on these N phases. If we prove lower bound of $\mathcal{R}' < q$ on these phases, we have that ALG is not better than $\mathcal{R}'$-competitive by using Claim 4 on the preceding phases. We now redefine $\text{ALG}(j)$ to be the total profit of ALG after j phases, ignoring the profit ALG_0 which was obtained before these phases started, and scaled such that the first column in phase j has height 1. Define α_j by the equality

$$\text{ALG}(j) = \sum_{i=1}^{j} x_i q^{i-j} = \alpha_j \sum_{i=0}^{j-1} q^{-i}. \tag{10}$$

Since $0 < x_i \le 1$ for all $i \ge 1$, we have

$$0 < \alpha_j \le 1 \text{ for all } j. \tag{11}$$

Since we scale in each step, and the column heights grow by a factor of q in each phase, we have

$$\text{ALG}(j+1) = \frac{\text{ALG}(j)}{q} + x_{j+1}. \tag{12}$$

We will use repeatedly that for $j \ge 1$,

$$\sum_{i=0}^{j-1} q^{-i} = \frac{1-q^{-j}}{1-q^{-1}} = \frac{q-q^{1-j}}{q-1}\ . \tag{13}$$

Lemma 8 *Assume that* ALG *acts as in Observation 3 for N consecutive phases and has a competitive ratio less than $(1-\varepsilon)\mathcal{R}$. Then for all $0 < \varepsilon < 0.2$ there exists an N' (independent of N) such that for all $N' \le j \le N$, we have:*

(i) $x_j < (1-\varepsilon)\alpha_{j-1}$, and

(ii) $\alpha_j < (1-\frac{\varepsilon}{4})\alpha_{j-1}$.

Since these two statements lead to a contradiction for N large enough (by (ii) we get $\alpha_j \to 0$ for $j \to \infty$ and hence by (i) also $x_j \to 0$, contradicting that $x_j > 1/2$ by Lemma 7), this proves the lower bound for this type of algorithms.

Proof (i) We consider the input where a final column of height $x'_j = (1-\delta)x_j$ arrives (for some $\delta > 0$) after the last column of phase j, which has height q. This is state change II in Figure 8. Now, since $x_j \le 1$ it is possible to earn $4x'_j = 4(1-\delta)x_j$ on the last four columns of this modified input:

Input	q^{1-j}	q^{2-j}	q^{2-j}	q^{3-j}	$\dots$	q^{-1}	1	1	q	x'_j
State	A	B	A	B	$\dots$	A	B	A	D	$-$
Phase	1		2		$\dots$	$j-1$		j		
ALG	x_1	0	x_2	0	$\dots$	x_{j-1}	0	x_j	x_j	0
OPT	0	q^{2-j}	0	q^{3-j}	$\dots$	0	x'_j	x'_j	x'_j	x'_j

This is the only case in which the adversary deviates from its standard strategy described at the start (namely in phase $j-1$ and j). Again we use the best possible strategy for ALG (given that its assignment in phases $1,\dots,j$ is fixed). Letting $\delta \to 0$, the competitive ratio for this input is

$$\frac{4x_j + q\sum_{i=2}^{j-1} q^{-i}}{2x_j + \alpha_{j-1}\sum_{i=1}^{j-1} q^{-i}} = \frac{4(q-1)x_j + 1 - q^{2-j}}{2(q-1)x_j + \alpha_{j-1}(1-q^{1-j})} .$$

(Note the indices of the summations.) For any $\varepsilon > 0$, there exists an N_1 such that $q^{2-j} < \varepsilon$ for all $j \geq N_1$. Then using (11) it can be verified that

$$\frac{4(q-1)(1-\varepsilon)\alpha_{j-1} + 1 - \varepsilon}{2(q-1)(1-\varepsilon)\alpha_{j-1} + \alpha_{j-1}(1-\frac{\varepsilon}{q})} > (1-\varepsilon)\mathcal{R}_1.$$

(We get equality for $\alpha_{j-1} = 1/(1-4.67\varepsilon)$, but by (11) this cannot happen. Note that $1-4.67\varepsilon > 0$ since $\varepsilon < 0.2$.) Since we can select δ arbitrarily small, we must have $x_j < (1-\varepsilon)\alpha_{j-1}$ for all $j \geq N_1$.

(ii) We first verify that

$$\frac{\alpha_j}{q}\cdot\frac{q}{q-1} + (1-\varepsilon)\alpha_j < \left(1-\frac{\varepsilon}{2}\right)\alpha_j\cdot\frac{q}{q-1}$$

for $q = 2.144472$, for any $\alpha_j > 0$ and $\varepsilon > 0$ (the inequality reduces to $q/(2q-2) < 1$). Using (10), (12), (13) and (i), this shows that

$$\text{ALG}(j+1) = \frac{\alpha_j}{q}\sum_{i=0}^{j-1} q^{-j} + x_{j+1} < \left(1-\frac{\varepsilon}{2}\right)\alpha_j\cdot\frac{q}{q-1}$$

for $j \geq N_1+1$. By (10) and (13), for every $\varepsilon > 0$ there exists an N_2 such that $\text{ALG}(j+1) > \alpha_{j+1}\frac{q}{q-1}/(1+\varepsilon/4)$ for all $j \geq N_2$. Hence for all $j \geq \max(N_1+1, N_2)$, we get

$$\alpha_{j+1} < \left(1-\frac{\varepsilon}{2}\right)\alpha_j\left(1+\frac{\varepsilon}{4}\right) < \left(1-\frac{\varepsilon}{4}\right)\alpha_j.$$

We can now define $N' = \max(N_1+1, N_2)$. □

Generalizing the lower bound It remains to be shown that ALG cannot do better by acting differently than in Observation 3. Given Corollary 1, we only need to handle phases of the form A^kB for some $k \geq 1$. This is done by modifying p and q as soon as such a phase occurs after there have been *sufficiently many previous phases* (of any length). As stated, we change q to $q' = 2.03644$ and p to $p' = 1/q'$ (at the point where this phase of length $k > 1$ ends) and continue the input as before.

We will show that afterwards ALG has no choice but to allocate a nonzero value in each state A column, and moreover that this value is decreasing over time. On a 0 allocation, state change I follows, and given that $p' = 1/q'$, it is now best to allocate 1 in state C. Eventually we find a contradiction to Lemma 7.

The calculations can be checked using the following table for $k = 3$. We number the phases so that the first phase with the modified values of p and q is phase 1. The first column in phase 1 has height 1. Phase 0 was the phase of length $k > 1$, which happened after many previous phases where OPT earned almost $1/(q^3-q) = 0.12957$ (at least) due to Claim 5. (To be precise, we wait until OPT has earned at least 0.129560 on previous phases; it can be verified that this happens already after at most 12 columns, independently of the number of phases.) Using Claim 4, we use $0.12956/q$ as an upper bound for $\text{ALG}(-1)$;

to this we add $1/q$ for phase 0. Generally, we use $\text{ALG}(i-1) \leq \text{OPT}(i-1)/q$. In all calculations below, if $\text{OPT}(i-1)$ is larger than the lower bound that we use, and $\text{ALG}(i-1) = \text{OPT}(i-1)/q$, the overall competitive ratio only increases.

In the table, h_i is the height of the newly arriving column in phase i (so $h_0 = 1$). We verify that ALG can never allocate 0 in the column labeled Ratio after 0 allocation: in this case we have $\text{ALG}(i) = h_i + \text{ALG}(i-1)$, $\text{OPT}(i) = 4h_{i-1} + \text{OPT}(i-1) - h_{i-1}$. The allocation x_i is given by the formula $x_i = (\mathcal{R} \cdot \text{ALG}(i-1) - (\text{OPT}(i-1) - h_{i-1})/(4 - 2\mathcal{R})$ so that an allocation above x_i immediately leads to a competitive ratio of $\mathcal{R}$. Assuming an allocation below x_i, we find $\text{ALG}(i) \leq \text{ALG}(i-1) + x_i h_{i-1}$.

Phase	$\text{ALG}(i-1)$	$\text{OPT}(i-1) - h_{i-1}$	h_i	Ratio after 0 allocation	x_i	$x_i h_{i-1}$
1	0.5267	0.3470	2.0364	4.3470 / 2.5632 = 1.696	0.8984	0.8984
2	1.4251	1.3470	4.1471	9.4928 / 5.5722 = 1.7036	0.8640	1.7594
3	3.1845	3.3835	8.4453	19.972 / 11.630 = 1.7173	0.7999	3.3173
4	6.5018	7.5306	17.198	41.312 / 23.70 = 1.7431	0.6808	5.7493
–	18.000	33.174				

As an example, we show the two ways in which the input may end on the first line of the above table. Let $x_1' = (1-\delta)x_1$.

Input	$\ldots q^{-3}$	q^{-3}	q^{-2}	q^{-1}	1	1	q'	1
State	$\ldots B$	A	A	A	B	A	C	–
Phase	$\ldots, -2, -1$		0				1	
ALG	0.0604	0	0	q^{-1}	0	**0**	q'	0
OPT	0.12956	0	q^{-2}	0	1	1	1	1

Input	$\ldots q^{-3}$	q^{-3}	q^{-2}	q^{-1}	1	1	q'	x_1'
State	$\ldots B$	A	A	A	B	A	D	–
Phase	$\ldots, -2, -1$		0				1	
ALG	0.0604	0	0	q^{-1}	0	$\mathbf{x_1}$	$\mathbf{x_1}$	0
OPT	0.12956	0	q^{-2}	0	x_1'	x_1'	x_1'	x_1'

For $k = 2$, we find $\text{OPT}(0) - h_0 = \text{OPT}(-1) + 1/q^2 = 0.4953$ (for a long enough input) and $\text{ALG}(0) \leq 1/q + \text{OPT}(-1)/q = 0.5959$. Hence, before the first column of height 1, OPT earns more relative to ALG than in the case $k = 3$. Since the profit ratio on the remaining part of the input remains unchanged and is always more than $\mathcal{R}$ (either $4/q'$ or $2(1-\delta)$ in the first line), we get the same results as before.

Similarly, for $k > 3$ it can be verified that OPT always earns more relative to ALG on early columns than in the case $k = 3$, leading to the same conclusion as above. (Increasing k by 2 always increases the relative profit of OPT.) Hence, after at most 12 columns of the input, ALG can only use phases of the form AB, which by Lemma 8 eventually leads to a competitive ratio arbitrarily close to $\mathcal{R} = 1.69595$.

4.2 Randomized lower bound

This lower bound is due to Leah Epstein and Asaf Levin [4] and is published here with their permission. Let k be the size of the lookahead window.

Theorem 3 *The competitive ratio of any randomized algorithm with* $k > 0$ *is at least* $(k+2)\ln\frac{k+2}{k+1}$. *This gives the lower bounds* 1.21639, 1.15072 *and* 1.115718 *for* $k = 1, 2, 3$.

Note that $(k+2)\ln\frac{k+2}{k+1} > 1 + \frac{1}{2k+3} = \frac{2(k+2)}{2k+3}$, to see this, consider the function

$$g(x) = \ln(1+x) - \frac{2x}{x+2} = \ln(1+x) - 2 + \frac{4}{x+2}.$$

This function is monotonically increasing since its derivative is $\frac{1}{x+1} - \frac{4}{(x+2)^2} = \frac{x^2}{(x+1)(x+2)^2}$. Its value for $x \to 0$ is 0, so $g(x) > 0$ for any $x > 0$. Now let $x = \frac{1}{k+1}$. $g(x) = \ln\frac{k+2}{k+1} - \frac{2}{2k+3} > 0$ as required.

Proof As is standard in these kind of lower bound constructions, we use Yao's method [9]. The input consists of $k+1$ columns with height $k+2$, and the $(k+2)$-th column has a height in the interval $[k+1, k+2]$, which is distributed using the density function $f(x) = \frac{(k+1)(k+2)}{x^2}$. Indeed

$$\int_{k+1}^{k+2} f(x)dx = (k+1)(k+2)\int_{k+1}^{k+2} \frac{1}{x^2}dx = (k+1)(k+2)(-\frac{1}{x})\Big|_{x=k+1}^{k+2} = 1.$$

Consider a deterministic algorithm. If this algorithm uses at least one zero column among the first $k+1$ columns, its profit would be at most $k(k+2)+\gamma$, where γ is the realization of the height of the last column, which is at most $(k+1)(k+2)$. Thus we find at most the same value as when the value had been $k+2$ in all of the first $k+1$ columns.

We can thus restrict our attention to the case where the algorithm uses some value z in all of the first $k+1$ columns. The decision on the value of z is taken before the algorithm can see the height of the last column. If $z \leq \gamma$, then the algorithm can use the same value z for the last column, and otherwise it cannot. Let $P = \int_z^{k+2} \frac{(k+1)(k+2)}{x^2}dx$. We get that with probability P, the algorithm has a profit of $(k+2)z$, and with probability $1-P$, it only has a profit of $(k+1)z$. Its expected profit is therefore

$$\begin{aligned}
P(k+2)z + (1-P)(k+1)z = Pz + (k+1)z =& z\left(k+1+\int_z^{k+2}\frac{(k+1)(k+2)}{x^2}dx\right) \\
= z\left(k+1+(k+1)(k+2)(-\frac{1}{x})\Big|_{x=z}^{k+2}\right) =& z\left(k+1-\frac{(k+1)(k+2)}{k+2}\right)+(k+1)(k+2) \\
=& (k+1)(k+2)
\end{aligned}$$

Note that this value is independent of the choice of z done by the algorithm. We next compute the expected optimal profit. Since by using a zero column, the total profit is at most $(k+1)(k+2)$, it is at least as profitable to use the value γ in all columns, since $\gamma \geq k+1$, and there are $k+2$ columns. Thus the expected optimal profit is

$$(k+2)\int_{k+1}^{k+2} xf(x)dx = (k+2)^2(k+1)\int_{k+1}^{k+2}\frac{1}{x}dx = (k+2)^2(k+1)\ln x\Big|_{x=k+1}^{k+2} = (k+2)^2(k+1)\ln\frac{k+2}{k+1}.$$

This completes the proof. □

5 Conclusions

We have narrowed the gap for this problem to 0.056. We believe that both our lower bound and upper bound could potentially be improved, but we conjecture that the lower bound is closer to the true competitive ratio of the problem. However, it is not easy to see how to narrow the gap further. There are four cases where the analysis for our algorithm is tight; additionally, there are various cases where the analysis is nearly tight. An improved algorithm would have to achieve a better ratio in all of the very different tight cases without losing too much in other cases.

It should be possible to improve the lower bound as follows. After the online algorithm spends many steps in state A, the value of q can be (slightly) reduced, because OPT has built up some previous profit. Thus we can have another state change of the form (*) in Figure 8 (possibly more than once), and continue in the same way.

Acknowledgment The author would like to thank Leah Epstein and Asaf Levin for interesting discussions, and Marek Chrobak for pointing out some problems in the proof of the lower bound in the conference version.

References

[1] Amrinder Arora and Hyeong-Ah Choi. Channel aware scheduling in wireless networks. Technical Report 002, George Washington University, 2006.

[2] Amrinder Arora, Fanchun Jin, and Hyeong-Ah Choi. Scheduling resource allocation with timeslot penalty for changeover. *Theor. Comput. Sci.*, 369(1-3):323–337, 2006.

[3] Sem C. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. In *INFOCOM*, 2003.

[4] Leah Epstein and Asaf Levin. Private communication, 2009.

[5] William Stallings. *Wireless Communications and Networks*. Prentice Hall, 2001.

[6] Vagelis Tsibonis, Leonidas Georgiadis, and Leandros Tassiulas. Exploiting wireless channel state information for throughput maximization. In *INFOCOM*, 2003.

[7] Haitao Wang, Amitabh Chaudhary, and Danny Z. Chen. New algorithms for online rectangle filling with *k*-lookahead. *J. Comb. Optim.*, 21(1):67–82, 2011.

[8] Haitao Wang, Amitabh Chaudhary, and Danny Z. Chen. Online rectangle filling. *Theor. Comput. Sci.*, 412(39):5247–5275, 2011.

[9] Andrew Chi-Chi Yao. Probabilistic computations: towards a unified measure of complexity. In *Proc. 18th Symp. Foundations of Computer Science (FOCS)*, pages 222–227. IEEE, 1977.