

Improved On-line Broadcast Scheduling with Deadlines*

Stanley P. Y. Fung[†] Feifeng Zheng[‡] Wun-Tat Chan[§] Francis Y. L. Chin[§]
Chung Keung Poon[¶] Prudence W.H. Wong^{||}

Abstract

We study an on-line broadcast scheduling problem in which requests have deadlines, and the objective is to maximize the weighted throughput, i.e., the weighted total length of the satisfied requests. For the case where all requested pages have the same length, we present an online deterministic algorithm named BAR and prove that it is 4.56-competitive. This improves the previous algorithm of Kim and Chwa [11] which is shown to be 5-competitive by Chan et al. [4]. In the case that pages may have different lengths, we give a $(\Delta + 2\sqrt{\Delta} + 2)$ -competitive algorithm where Δ is the ratio of maximum to minimum page lengths. This improves the $(4\Delta + 3)$ -competitive algorithm of [4]. We also prove an almost matching lower bound of $\Omega(\Delta/\log \Delta)$. Furthermore, for small values of Δ we give better lower bounds.

1 Introduction

Data broadcast scheduling is a core problem in many applications that involve distribution of information from a server to a large group of receivers. In contrast to the traditional point-to-point mode of communication, broadcasting technologies are employed so that different clients requesting the same data can be satisfied simultaneously by only one broadcast. For example, in Hughes' DirecPC system [12], clients make requests over phone lines and the server satisfies the requests through broadcasts via a satellite. Typical information that will be broadcasted include movies (video on-demand), stock market quotation, traffic and landmark information, etc. Very often, the information to be disseminated is time critical and thus it is important to meet the deadlines of the requests.

Motivated by these applications, we study the following *On-line Scheduling of Broadcasts* (Broadcasting): We are given a set of pages to be broadcasted to clients upon request. Each request r has four attributes, namely, $p(r)$: the requested page, $a(r)$: the arrival time, $d(r)$: the deadline by which the requested page has to be received in its entirety, and $w(r)$: the weight of the request. A request is not known until it arrives, i.e., at time $a(r)$. When it arrives, all $p(r)$, $d(r)$ and $w(r)$ become known. When the server broadcasts a page, all requests to the same page

*The work described in this paper was fully supported by grants from the Research Grants Council of the Hong Kong SAR, China [CityU 1198/03E, HKU 7142/03E, HKU 5172/03E], an NSF Grant of China [No. 10371094], and a Nuffield Foundation Grant of UK [NAL/01004/G].

[†]Department of Computer Science, University of Leicester, United Kingdom. Email: pyfung@mcs.le.ac.uk

[‡]School of Management, Xi'an JiaoTong University, China. Email: zhengff@mailst.xjtu.edu.cn

[§]Department of Computer Science, The University of Hong Kong, Hong Kong. Email: {wtchan, chin}@cs.hku.hk

[¶]Department of Computer Science, City University of Hong Kong, China. Email: ckpoon@cs.cityu.edu.hk

^{||}Department of Computer Science, University of Liverpool, United Kingdom. Email: pwong@csc.liv.ac.uk

that have arrived will receive the content of the page simultaneously. Upon completion, each of these requests will be satisfied, provided that the completion time is before its respective deadline. The server is allowed to abort the current page it is broadcasting before its completion and start a new one. To satisfy an aborted request, the requested page has to be broadcasted again from the beginning. Thus it is an on-line scheduling problem with preemptions and restarts. Our goal is to maximize the total weighted throughput, i.e., the total weighted lengths of all satisfied requests.

Related work. Most of the previous works on the problem of on-line broadcast scheduling concentrate on minimizing the flow time where the flow time of a request is the time elapsed between its arrival and its completion. For example, [9, 5, 6, 7] studied the problem of minimizing the total flow time while Bartal and Muthukrishnan [2] studied the minimization of the maximum flow time. Aksoy and Franklin [1] presented a practical parameterized algorithm and evaluated it with extensive experiments. While the flow time is important and related to how the clients perceive the responsiveness of the system, the objective of maximizing the throughput is crucial for applications in which requests are associated with deadlines. Jiang and Vaidya [8] considered the problem of maximizing the percentage of satisfied requests assuming knowledge of the probability distribution of the requests. A different model of scheduling broadcasts with deadlines is studied in [10].

Kim and Chwa [11] were the first to design algorithms with provable worst case performance bounds for this problem. In particular, one of their results is a 5.828-competitive algorithm for the case where all pages have the same length. Using a tighter analysis, Chan et al [4] showed that Kim and Chwa's algorithm actually has competitive ratio at most 5, through a reduction to a job scheduling problem with cancellations. It was however shown in [14] that we need new techniques to further improve the bound. For the case of different page lengths, let Δ be the ratio between the length of the longest and shortest page. There is a $(4\Delta + 3)$ -competitive algorithm [4] and a $\sqrt{\Delta}$ lower bound [11, 4].

A related problem is the on-line interval scheduling problem studied by Woeginger [13]. Translated into our terminologies, his problem is to schedule requests with tight deadlines (i.e., the length of time interval between its arrival time and deadline is exactly the length of the requested page) and pages have different lengths. He proved that when the page length and the weight of requests can be arbitrarily related, no deterministic algorithm can have constant competitive ratio. He went on to give a heuristic that is 4-competitive for several special cases in which the page length and the weight of requests satisfy certain relationship. In particular, the heuristic works for the case of unit page length and arbitrary weights. He then complemented his upper bound with several lower bounds, including a tight lower bound for this special case of unit page length. This lower bound of 4 carries to the unit page length case of our problem. In some sense, our problem is a generalization of Woeginger's problem allowing non-tight intervals.

Our results. In this paper we give a number of improved results for **Broadcasting**. Our first contribution is to give an improved algorithm for the case of unit page length. We consider the deadlines of the requests, a parameter ignored by previous algorithms, in our scheduling decision. By considering the fact that some of the currently-serving requests might have distant deadlines and can be served later after the completion of new requests, we improve the competitive ratio from 5 [4] to 4.56. In Section 3 we describe this algorithm and its analysis.

Our second contribution is to give an improved algorithm for the case of different page lengths. The algorithm utilizes the simple observation that we should favor a request which gives more profit

and finishes earlier comparing with the one under broadcast. From this we improve the competitive ratio from $4\Delta + 3$ from $\Delta + 2\sqrt{\Delta} + 2$, where Δ is the ratio between the length of the longest and shortest page. This is discussed in Section 4.

Thirdly, we prove a lower bound which almost matches the above upper bound; in Section 5 we show that any deterministic algorithm has a competitive ratio at least $\Omega(\Delta/\log \Delta)$. This improves the previous $\sqrt{\Delta}$ result.

All existing lower bound proofs for the case of different page lengths do not work very well when $\Delta > 1$ is small, and thus the lower bound for those cases is still 4, that being the lower bound for the unit page length case. In Section 6 we describe our fourth contribution of proving better lower bounds for these cases. The lower bound for the competitive ratio is improved to e.g., 4.245, 4.481, 4.707 and 5.873 for $\Delta = 2, 3, 4$ and 10 respectively. The result is obtained by extending the lower bound proof for the unit page length case [13] to the case of different page lengths.

2 Notations

We first state the problem formally. Assume there are n pages, P_1, \dots, P_n , in the system. A request for some page may arrive in arbitrary time with arbitrary deadline. If a page is fully broadcasted, then a request for that page is *satisfied* if the requests arrive before the broadcast of the page, and the broadcast finishes before the deadline of the request. A broadcast can be aborted at any time, and can later be restarted from the beginning.

A schedule S is a sequence of broadcasts J_1, J_2, \dots where each broadcast J_i is a set of requests to the same page started being served at time $s(J_i)$. The broadcasts are indexed such that $s(J_i) < s(J_{i'})$ for $i < i'$. For convenience, we will write J_i to represent both the set of requests and the requested page. Let $l(J_i)$ be the length of the page broadcasted by J_i . If $s(J_i) + l(J_i) > s(J_{i+1})$, then the broadcast J_i is *aborted* by J_{i+1} ; otherwise J_i is said to be *completed*. The *profit* of a completed request is its weight times the length of the page it requests. The profit of a broadcast J_i , denoted by $|J_i|$, is the sum of $w(r) \times l(J_i)$ over all r in J_i . We denote by $|S|$ the total profit of completed broadcasts in the schedule S , i.e., we only count those satisfied requests. The objective is to maximize the total profit of satisfied requests $|S|$ during a time period.

Given an input I (a set of requests) and an algorithm \mathcal{A} , we denote by $\mathcal{A}(I)$ and $\mathcal{O}(I)$ the schedules produced by \mathcal{A} and by an optimal offline algorithm \mathcal{O} on I respectively. When I is understood from the context, we will simply denote the schedules by \mathcal{A} and \mathcal{O} respectively. To gauge the quality of the schedules produced by an on-line algorithm, the competitive ratio analysis [3] is often used. The competitive ratio of algorithm \mathcal{A} is defined as $r_{\mathcal{A}} = \sup_I \frac{|\mathcal{O}(I)|}{|\mathcal{A}(I)|}$.

3 Unit Page Length: the BAR Algorithm

In this section, we consider the case where each page is of the same length. Thus we assume without loss of generality that broadcasting each page requires one unit of time. We present our BAR algorithm (for Bi-level Abortion Ratio) and prove that it is 4.56-competitive.

At any moment, there is a (possibly empty) pool of requests that have arrived and are waiting to be served. Requests in the pool will be discarded when their deadlines cannot be met even if the server starts to serve them immediately. The algorithm is triggered either when a broadcast is completed or when a new request arrives. In the former case, the server will pick the page with the maximum total profit of requests in the pool to be broadcasted next. When a new request

arrives while the server is broadcasting a page, the server will either continue the current broadcast J (and add the new requests to the pool), or abort J in favor of a new broadcast R with the most requests in the pool (including newly arrived requests and possibly part of J). The decision is made according to the relative profits of J and R . It will also consider the previous broadcast and the deadlines of requests currently in the system.

More precisely, let J_0 be the broadcast aborted by J . If J does not abort any broadcast, define $J_0 = \phi$ (the empty set). Let J' be the set of requests with maximum total profit that can be satisfied in a single broadcast after completing J , assuming no more requests arrive. Similarly R' denotes the set of requests with maximum total profit that can be satisfied in one broadcast after completing R , if we abort J and start R now. See Figure 1. Let α, β be some constants such that $1.5 \leq \alpha < 2 \leq \beta < 2.5$. The exact values will be determined later. If either one of the following conditions is satisfied, we abort J and start R :

C1: $\beta|J| \leq |R|$ and $\beta^2|J_0| \leq |R|$, or

C2: $\alpha|J| \leq |R| < \beta|J|$, $\beta|J| + |J'| \leq |R| + |R'|$, and $\beta|J_0| \leq |J|$.

Otherwise, we continue with the broadcasting of J and add the new requests into the pool.

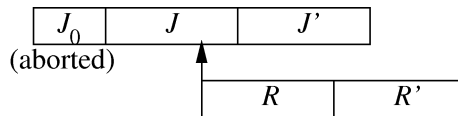


Figure 1: BAR is broadcasting J and determining whether to abort J and start R .

We give some intuitive rationale behind these conditions. In previous algorithms [11, 4], the abortion is simply determined by considering whether $\beta|J| \leq |R|$. This completely ignores the deadlines of the requests. The improvement of BAR comes from the introduction of [C2], which gives an alternative abortion condition with a lower threshold α and, as we show below, considers the deadlines of requests.

The first part of condition [C1] is the usual abortion condition by considering the profits of requests. The second part of [C1] enforces some technical properties that are required in bounding the profits of the requests, despite some abortions being caused by condition [C2].

The second part of condition [C2] utilizes deadlines of requests. Rather than directly comparing the deadlines of requests, we consider the profit of requests that can later be satisfied (before their deadlines) in deciding whether to abort the page currently being broadcast. Suppose an abortion happens due to condition [C2]. Then some part of R' must come from J (which is aborted). Otherwise, the requests in R' come from the pool only, and since the broadcast J' will finish earlier than R' , R' is a possible choice of J' . Hence $|R'| \leq |J'|$ and the condition $\beta|J| + |J'| \leq |R| + |R'|$ cannot be true. So $R' \cap J$ is not empty, and they must be requesting the same page. That means once condition [C2] happens, there must be some requests in J that have long deadlines, long enough to be completed after R is completed. Therefore, even though $|R|$ is only α times larger than $|J|$, we may still satisfy enough requests to achieve a good competitive ratio.

The third part of condition [C2] ensures there will not be two consecutive abortions caused by [C2], so that this weaker-threshold condition will not be used too often.

The remaining of this section is devoted to the proof of the following theorem.

Theorem 3.1. *BAR is 4.56-competitive for Broadcasting with unit page lengths.*

3.1 Basic Subschedules

We now elicit certain useful structures in a schedule produced by BAR. A sequence of broadcasts (J_1, \dots, J_k) is called a *chain* if J_i is aborted by J_{i+1} for all $i = 1, \dots, k-1$ and J_1 is preceded by either an idle interval or a completed broadcast. A chain $C = (J_1, \dots, J_k)$ in which J_k is completed is called a *basic subschedule*. Thus a basic subschedule consists of a sequence of zero or more aborted broadcasts followed by a completed broadcast; and the sequence cannot be extended at the front. Furthermore, the broadcast before an idle interval must be completed. Therefore, the whole schedule can be decomposed into a sequence of basic subschedules.

Consider an arbitrary basic subschedule, $B = (J_1, J_2, \dots, J_k)$ where J_k is a completed broadcast and all the others are aborted ones. The total profit of requests satisfied by BAR in this basic subschedule is $|B| = |J_k|$. To analyze the competitive ratio, we will associate with B a carefully chosen set of requests satisfied by the optimal offline algorithm \mathcal{O} .

Consider a broadcast J by BAR. We can make use of condition [C1] and/or [C2] to argue that requests started by \mathcal{O} while BAR is broadcasting J cannot be too large, if these requests are available in the pool maintained by BAR. Note, however, that \mathcal{O} can also serve requests with arbitrary large profits that have been satisfied by BAR before without violating [C1, C2]. Thus, we classify the requests satisfied by \mathcal{O} into two types according to whether the request has been satisfied by BAR at the time \mathcal{O} starts them. (Since \mathcal{O} is an offline algorithm, we assume that it will never abort a broadcast. Thus, saying that a request is started by \mathcal{O} is equivalent to saying that it will be satisfied by \mathcal{O} .) More precisely, we define J_i^* , for $i = 1, \dots, k$, as the set of requests started by \mathcal{O} within the interval $[s(J_i), s(J_{i+1}))$ but have not been satisfied by BAR before time $s(J_k)$, where $s(J_i)$ is the start time of J_i and we take $s(J_{k+1}) = s(J_k) + 1$. Also, we define B^* as the set of requests in J_k that are started by \mathcal{O} after the basic subschedule B is completed. We will try to obtain an upper bound on $\sum_{i=1}^k |J_i^*| + |B^*|$.

Note that if a broadcast J_i is aborted by a broadcast J_{i+1} due to condition [C1], the ratio $|J_{i+1}|/|J_i|$ is at least β . However if the abortion is due to condition [C2], $|J_{i+1}|/|J_i|$ may be smaller than β . Nevertheless, we can still bound the profits of J_i 's and J_i^* 's by geometric series in the following lemmas. Consider a chain $C = (J_1, \dots, J_k)$. It is said to have *big endian* if $|J_k| \geq \beta|J_{k-1}|$, meaning that the last abortion is due to condition [C1]; and *small endian* otherwise, indicating that it is a [C2] abortion. If C has only one broadcast, we take $J_{k-1} = \phi$. Thus C will be considered to have big endian.

Lemma 1. *Consider a chain $C = (J_1, \dots, J_k)$ with big endian. Then $|J_i| \leq |J_k|/\beta^{k-i}$ for all $i = 1, \dots, k$.*

Proof. Note that $|J_k| \leq |J_k|/\beta^0$ and since C has big endian, $|J_{k-1}| \leq |J_k|/\beta$. Now assume $|J_i| \leq |J_k|/\beta^{k-i}$ and $|J_{i-1}| \leq |J_k|/\beta^{k-i+1}$ for some $i \leq k$ and consider $|J_{i-2}|$. If $|J_{i-2}| \leq |J_{i-1}|/\beta$, then $|J_{i-2}| \leq |J_k|/\beta^{k-i+2}$. Otherwise, $|J_{i-2}| > |J_{i-1}|/\beta$ (it is a [C2] abortion) and hence J_i aborts J_{i-1} by condition [C1]. Hence $|J_{i-2}| \leq |J_i|/\beta^2 \leq |J_k|/\beta^{k-i+2}$. \square

Next, we bound the profit of J_i^* using the bound on J_i as follows.

Lemma 2. *Consider a chain $C = (J_1, \dots, J_k)$ with big endian. Then $|J_i^*| < |J_k|/\beta^{k-i-1}$ for all $i = 1, \dots, k$. Hence $\sum_{i=1}^k |J_i^*| < \frac{\beta^2}{\beta-1} \left(1 - \frac{1}{\beta^k}\right) |J_k|$.*

Proof. The second part of the lemma follows easily from the first part by summing up the geometric progression: $\sum_{i=1}^k |J_i^*| < \sum_{i=1}^k |J_k|/\beta^{k-i-1} = \frac{\beta^2}{\beta-1} \left(1 - \frac{1}{\beta^k}\right) |J_k|$. Now we prove the first part. Observe that $|J_k^*| < \beta|J_k|$. This is because J_k^* is present in the system at time $s(J_k^*)$ but has not been satisfied by BAR. So if $|J_k^*| \geq \beta|J_k|$, BAR would have aborted J_k by condition [C1] at time $s(J_k^*)$. Similarly, we have $|J_1^*| < \beta|J_1| \leq |J_k|/\beta^{k-2}$ by Lemma 1. Hence the lemma is true for $i = 1$ and $i = k$. Now consider $|J_i^*|$ for $i = 2, \dots, k-1$.

Case (1): $\beta|J_{i-1}| \leq |J_i|$

Then $|J_i^*| < \beta|J_i|$ (otherwise, $|J_i^*| \geq \beta|J_i|$ and $|J_i^*| \geq \beta^2|J_{i-1}|$ and hence BAR would have aborted J_i at the start time of J_i^*). Therefore $|J_i^*| < |J_k|/\beta^{k-i-1}$ by Lemma 1.

Case (2): $\beta|J_{i-1}| > |J_i|$

Then J_i aborts J_{i-1} by condition [C2] and hence J_{i+1} aborts J_i by condition [C1]. Hence $\beta|J_i| \leq |J_{i+1}|$ and $\beta^2|J_{i-1}| \leq |J_{i+1}|$ by construction of BAR. Therefore, $|J_i^*| < |J_{i+1}|$ (otherwise, $\beta|J_i| \leq |J_{i+1}|$ and $\beta^2|J_{i-1}| \leq |J_{i+1}|$). Therefore $|J_i^*| < |J_k|/\beta^{k-i-1}$ by Lemma 1 again. \square

The following lemma bounds the total profits of requests served by \mathcal{O} in a basic subschedule.

Lemma 3. *Consider a basic subschedule $B = (J_1, \dots, J_k)$. If B has small endian,*

$$\sum_{j=1}^k |J_j^*| + |B^*| \leq \left(\beta + \frac{1}{\beta-1}\right) \beta|J_{k-1}| + |J_k| + |J'_k| - \frac{\beta}{\beta-1}|J_1|.$$

If B has big endian,

$$\sum_{j=1}^k |J_j^*| + |B^*| \leq \left(\alpha + \frac{\beta}{\beta-1} + 1\right) |J_k| + |J'_k| - \frac{\beta}{\beta-1}|J_1|.$$

Proof. Suppose B has small endian. We observe that the chain (J_1, \dots, J_{k-1}) must have big endian by construction of BAR. (Note that $k \geq 2$ if B has small endian since $k = 1$ implies B has big endian.) By Lemma 2, we have $\sum_{j=1}^{k-1} |J_j^*| < \frac{\beta^2}{\beta-1} \left(1 - \frac{1}{\beta^{k-1}}\right) |J_{k-1}|$. Also, we have $|J_k^*| < \beta^2|J_{k-1}|$ for otherwise, J_k^* would have aborted J_k due to condition [C1]. Thus $\sum_{j=1}^k |J_j^*| < \frac{\beta^3}{\beta-1} \left(1 - \frac{1}{\beta^k}\right) |J_{k-1}|$.

As for B^* , we note that $\beta|J_{k-1}| + |J'_{k-1}| \leq |J_k| + |J'_k|$ since J_k aborts J_{k-1} by condition [C2]. Moreover, $|B^*| \leq |J'_{k-1}|$ because requests in B^* have deadlines no earlier than that of J'_{k-1} and they have not been satisfied by BAR at time $s(J'_{k-1})$. Hence $|B^*| \leq |J_k| - \beta|J_{k-1}| + |J'_k|$.

Combining these two bounds, we have

$$\begin{aligned} & \sum_{j=1}^k |J_j^*| + |B^*| \\ & \leq \frac{\beta^3}{\beta-1} \left(1 - \frac{1}{\beta^k}\right) |J_{k-1}| + (|J_k| - \beta|J_{k-1}| + |J'_k|) \\ & = \left(\beta + \frac{1}{\beta-1}\right) \beta|J_{k-1}| + |J_k| + |J'_k| - \frac{1}{(\beta-1)\beta^{k-3}} |J_{k-1}| \\ & \leq \left(\beta + \frac{1}{\beta-1}\right) \beta|J_{k-1}| + |J_k| + |J'_k| - \frac{\beta}{\beta-1} |J_1| \end{aligned}$$

where the last inequality follows from Lemma 1.

Suppose B has big endian. Then we have $\sum_{j=1}^{k-1} |J_j^*| < \frac{\beta}{\beta-1} \left(1 - \frac{1}{\beta^{k-1}}\right) |J_k|$ by the first part of Lemma 2. For $|J_k^*| + |B^*|$, it is analyzed as follows. Since $|J_k| \geq \beta |J_{k-1}|$ and J_k^* does not abort J_k , $|J_k^*|$ cannot be as big as $\beta |J_k|$ and so either (1) $|J_k^*| < \alpha |J_k|$, or (2) $\alpha |J_k| \leq |J_k^*| < \beta |J_k|$ and $\beta |J_k| + |J_k'| > |J_k^*| + |(J_k^*)'|$. In case (1), we have $|J_k^*| + |B^*| < (\alpha + 1) |J_k|$ since B^* is a subset of J_k . In case (2), we have $|B^*| \leq |(J_k^*)'|$ because \mathcal{O} can only start the requests in B^* after J_k^* is completed and those requests in J_k that can be completed in B^* can also be completed in $(J_k^*)'$. Hence $|J_k^*| + |B^*| \leq |J_k^*| + |(J_k^*)'| < \beta |J_k| + |J_k'|$. In either case, we have $|J_k^*| + |B^*| < (\alpha + 1) |J_k| + |J_k'|$ since $\beta < \alpha + 1$. Hence,

$$\begin{aligned} & \sum_{j=1}^k |J_j^*| + |B^*| \\ & \leq \frac{\beta}{\beta-1} |J_k| + (\alpha + 1) |J_k| + |J_k'| - \frac{1}{(\beta-1)\beta^{k-2}} |J_k| \\ & \leq \left(\alpha + \frac{\beta}{\beta-1} + 1 \right) |J_k| + |J_k'| - \frac{\beta}{\beta-1} |J_1| \end{aligned}$$

where again the last inequality follows from Lemma 1. \square

In Lemma 3, no matter B has big or small endian, we can bound $|J_k'|$ from above by the profit of the first broadcast in the basic subschedule after B . If B is followed by an idle interval, then we can actually argue that $|J_k'| = 0$. That is, we associate $|J_k'|$ with the basic subschedule following B . By the same token, B will also be associated with such value from the preceding basic subschedule. In the next subsection we will see how this association is used in the analysis.

3.2 Aggregated Subschedules

Let $B_i = (J_{i,1}, \dots, J_{i,k_i})$ denote the i -th basic subschedule in a sequence of basic subschedules. For notational convenience define $J_{i,0} = \phi$, $L_i = J_{i,k_i}$ (the last broadcast in the basic subschedule B_i), and $SL_i = J_{i,k_i-1}$ (the second last broadcast in the basic subschedule).

Lemma 4. *The last basic subschedule before an idle interval must have big endian.*

Proof. Suppose the lemma is false and the last basic subschedule B_i before an idle interval has small endian. Then there must be some requests in SL_i that have long enough deadlines so that they can be completed after B_i . (The broadcast SL_i must exist or else the basic subschedule has big endian.) Hence BAR will not be idle after this basic subschedule, contradicting the assumption in the lemma. \square

Based on the above lemma, we can partition the original schedule into a number of *aggregated subschedules*, each of which containing zero or more basic subschedules with small endians followed by one basic subschedule with big endian.

Consider an arbitrary aggregated subschedule, $A = (B_1, \dots, B_m)$ where for $i = 1, \dots, m$, $B_i = (J_{i,1}, \dots, J_{i,k_i})$ is a basic subschedule with k_i broadcasts.

Obviously, the total profit of requests satisfied by BAR in A is

$$|A| = \sum_{i=1}^m |L_i|. \quad (1)$$

Also, since $|L_i| \geq \alpha|SL_i|$ for $i = 1, \dots, m-1$, we have

$$|A| \geq \alpha \left(\sum_{i=1}^{m-1} |SL_i| \right) + |L_m|. \quad (2)$$

Recall that B_i 's have small endians for $i = 1, \dots, m-1$. For these basic subschedules, we have $k_i \geq 2$ since basic subschedules with only one broadcast must have big endians by definition. By condition [C2], $\beta|SL_i| + |(SL_i)'| \leq |L_i| + |(L_i)'|$ where $|(L_i)'| \leq |J_{i+1,1}|$ because $(L_i)'$ is a candidate set of requests to be served after L_i is completed. Hence we have $\beta|SL_i| \leq |L_i| + |J_{i+1,1}|$, and together with (1),

$$|A| \geq \sum_{i=1}^{m-1} \beta|SL_i| + |L_m| - \sum_{i=2}^m |J_{i,1}|. \quad (3)$$

On the other hand, the total profit of requests satisfied by \mathcal{O} and associated with aggregated subschedule A is:

$$\begin{aligned} |A^*| &= \left(\sum_{j=1}^{k_1} |J_{1,j}^*| + \dots + \sum_{j=1}^{k_m} |J_{m,j}^*| \right) + (|B_1^*| + \dots + |B_m^*|) \\ &\leq \sum_{i=1}^{m-1} \left(\beta + \frac{1}{\beta-1} \right) \beta|SL_i| + \left(\alpha + \frac{\beta}{\beta-1} \right) |L_m| \\ &\quad + \sum_{i=1}^m |L_i| + \sum_{i=2}^m |J_{i,1}| + |(L_m)'| - \frac{\beta}{\beta-1} \sum_{i=1}^m |J_{i,1}| \end{aligned} \quad (4)$$

where the inequality follows from Lemma 3, and the fact that $|(L_i)'| \leq |J_{i+1,1}|$ for $1 \leq i \leq m-1$. Consider (1) + (2) $\times (\frac{\beta^2}{\alpha})$ + (3) $\times \frac{1}{\beta-1}$. We have

$$\begin{aligned} &\left(1 + \frac{\beta^2}{\alpha} + \frac{1}{\beta-1} \right) |A| \\ &\geq \sum_{i=1}^m |L_i| + \left(\sum_{i=1}^{m-1} \beta^2 |SL_i| + \frac{\beta^2}{\alpha} |L_m| \right) \\ &\quad + \frac{1}{\beta-1} \left(\sum_{i=1}^{m-1} \beta |SL_i| + |L_m| - \sum_{i=2}^m |J_{i,1}| \right) \\ &= \sum_{i=1}^m |L_i| + \sum_{i=1}^{m-1} \left(\beta + \frac{1}{\beta-1} \right) \beta |SL_i| + \left(\frac{\beta^2}{\alpha} + \frac{1}{\beta-1} \right) |L_m| - \frac{1}{\beta-1} \sum_{i=2}^m |J_{i,1}| \\ &\geq |A^*| + \frac{\beta}{\beta-1} |J_{1,1}| - |(L_m)'| \\ &\geq |A^*| + |J_{1,1}| - |(L_m)'| \end{aligned} \quad (5)$$

where the second-last inequality follows from (4) and holds as long as $\frac{\beta^2}{\alpha} + \frac{1}{\beta-1} \geq \frac{\beta}{\beta-1} + \alpha$. We can bound $|(L_m)'|$ from above by the profit of the first broadcast (i.e., $|J_{1,1}|$) in the next aggregated

subschedule. Thus, if we have a sequence of aggregated subschedules A_1, \dots, A_l , then from (5) we have

$$\left(\frac{\beta^2}{\alpha} + \frac{\beta}{\beta-1}\right) (|A_1| + \dots + |A_l|) \geq |A_1^*| + \dots + |A_l^*| - |J'| \quad (6)$$

where J is the last broadcast in A_l . Since there is no more broadcast after A_l , $|J'| = 0$.

If the whole aggregated subschedule consists of only one basic subschedule with big endian, i.e., $A = (B_1)$, then $|A| = |L_1|$ and

$$\begin{aligned} |A^*| &= \sum_{j=1}^{k_1} |J_{1,j}^*| + |B_1^*| \\ &\leq \left(\alpha + \frac{\beta}{\beta-1} + 1\right) |L_1| + |(L_1)'| - \frac{\beta}{\beta-1} |J_{1,1}| \\ &\leq \left(\frac{\beta^2}{\alpha} + \frac{1}{\beta-1} + 1\right) |A| + |(L_1)'| \\ &= \left(\frac{\beta^2}{\alpha} + \frac{\beta}{\beta-1}\right) |A| + |(L_1)'|. \end{aligned}$$

Again $|(L_1)'| = 0$ and thus inequality (6) still holds.

The condition $\frac{\beta^2}{\alpha} + \frac{1}{\beta-1} \geq \frac{\beta}{\beta-1} + \alpha$ can be satisfied by having $\alpha^2 + \alpha \leq \beta^2$, i.e., $\alpha \leq \sqrt{\beta^2 + \frac{1}{4}} - \frac{1}{2}$. Setting $\alpha = \sqrt{\beta^2 + \frac{1}{4}} - \frac{1}{2}$, the competitive ratio of BAR is

$$\begin{aligned} &\frac{\sum_{i=1}^l |A_i^*|}{\sum_{i=1}^l |A_i|} \\ &\leq \frac{\beta}{\beta-1} + \frac{\beta^2}{\alpha} \\ &= \frac{\beta}{\beta-1} + \frac{\beta^2}{\sqrt{\beta^2 + \frac{1}{4}} - \frac{1}{2}} \\ &= \frac{3}{2} + \frac{1}{\beta-1} + \sqrt{\beta^2 + \frac{1}{4}}. \end{aligned}$$

Taking the derivative of $\frac{3}{2} + \frac{1}{\beta-1} + \sqrt{\beta^2 + \frac{1}{4}}$ and equating it with zero, we get the equation $\beta(\beta-1)^2 - \sqrt{\beta^2 + \frac{1}{4}} = 0$. Solving for the equation, we get that the minimum value of $\frac{3}{2} + \frac{1}{\beta-1} + \sqrt{\beta^2 + \frac{1}{4}}$ is approximately 4.561 attained when $\beta \approx 2.015$, and $\alpha \approx 1.576$.

4 Variable Length Pages: Upper Bound

We now turn our attention to the case where the pages can have different lengths. Let Δ be the ratio of lengths of the longest and shortest page. For simplicity assume Δ is an integer (if Δ is not an integer, the result in this section still holds after replacing Δ with $\lceil \Delta \rceil$). Without loss of

generality assume the shortest page is of length 1 and the longest is of length Δ . Recall that the objective is to maximize the total profit (i.e., weighted length) of satisfied requests.

In this section we give an improved algorithm ACE (for ‘Another Completes Earlier’), which makes use of a simple observation: if the broadcast of a page for a newer request has the same or larger profit and an earlier completion time than the page currently being broadcast, we should abort the current page in favor of the newer page.

Algorithm ACE. Let $\beta = 1 + \sqrt{\Delta}$. Let J be the page currently being broadcast. A new request for page R (together with all pending requests for R in the pool) will abort J if either one of the following holds: (1) $|R| \geq \beta|J|$, or (2) $|R| \geq |J|$ and the completion time of R (if we start R now) is earlier than the completion time of J (if we continue to broadcast J). When a broadcast completes, broadcast a page with the maximum profit among pending requests.

Theorem 4.1. *ACE is $(\Delta + 2\sqrt{\Delta} + 2)$ -competitive for Broadcasting with variable page lengths.*

Proof. Let \mathcal{A} be the schedule produced by ACE. As before we divide the schedule into a set of basic subschedules, and we focus on a single basic subschedule. We prove the competitiveness by bounding, for each basic subschedule, the maximum profit of an optimal offline schedule \mathcal{O} that can be charged to this basic subschedule, using an appropriate charging scheme. We will first assume that all abortions in \mathcal{A} are due to condition (1) of the algorithm, and that each broadcast in \mathcal{A} runs for Δ time units before they are aborted or completed. We will remove these assumptions later.

Consider a basic subschedule (J_1, \dots, J_k) . We have $|J_i| \leq |J_k|/\beta^{k-i}$ since all abortions are due to condition (1). Consider the broadcasts started by \mathcal{O} within the time of broadcast of J_i . There can be at most Δ such broadcasts. If the requests in these broadcasts are completed in \mathcal{A} before they are started in \mathcal{O} , we charge their profits to those earlier broadcasts in \mathcal{A} . Now consider those requests that are not completed in \mathcal{A} before. For any broadcast O in \mathcal{O} completed before J_i is completed or aborted, we have $|O| < |J_i|$ since otherwise they would abort J_i in \mathcal{A} . For any broadcast O in \mathcal{O} that is completed after J_i is completed or aborted, we have $|O| < \beta|J_i|$ or else it would also abort J_i in \mathcal{A} . In this case no more broadcasts from \mathcal{O} after this O are charged to this J_i .

Therefore, in the worst case, \mathcal{O} schedules at most $(\Delta - 1)$ length-1 broadcasts each of profit at most $|J_i|$, followed by a length-1 broadcast of profit at most $\beta|J_i|$. Therefore, the total profit in \mathcal{O} corresponding to J_i is at most $(\Delta - 1 + \beta)|J_i|$. Summing over all J_i 's, and considering that J_k may be served in \mathcal{O} later and will be charged to this basic subschedule, we have that the total profits in \mathcal{O} charged to this basic subschedule is at most

$$\sum_{i=1}^k (\Delta - 1 + \beta)|J_i| + |J_k| \leq (\Delta - 1 + \beta) \sum_{i=1}^k \frac{|J_k|}{\beta^{k-i}} + |J_k| < \left(\frac{\beta(\Delta - 1 + \beta)}{\beta - 1} + 1 \right) |J_k|.$$

Hence the competitive ratio is $\frac{\beta(\Delta-1+\beta)}{\beta-1} + 1$. By setting $\beta = 1 + \sqrt{\Delta}$, this ratio is minimized and is equal to $\Delta + 2\sqrt{\Delta} + 2$.

Now we return to our assumptions. We can view the above proof as giving an upper bound on the profit of the broadcast made by \mathcal{O} during each broadcast made in \mathcal{A} (excluding those mapped to earlier completed broadcasts in the online schedule). So, we can regard each basic subschedule as having a corresponding ‘worst’ optimal schedule which, under our charging scheme, is ‘compatible’ to the basic subschedule. We show that if a basic subschedule does not satisfy our

assumptions, then there is another basic subschedule satisfying the assumptions and which has a larger competitive ratio, i.e. the ratio of its corresponding worst optimal schedule's profit to the basic subschedule's profit is larger. Hence, we only need to bound the competitive ratio for the case where the assumptions are satisfied.

First we show that we can assume all abortions in \mathcal{A} are due to condition (1) of the algorithm. Suppose there is a page J aborted by another page R using condition (2) in a basic subschedule. We construct another basic subschedule by replacing the two broadcasts J and R with a single broadcast with starting time $s(J)$ and completion (or abortion) time same as that of R , and with a profit same as that of R .

The profit of this new basic subschedule is the same as the original one, so the profit of the online algorithm remains unchanged. As for the optimal offline algorithm, the following lemma shows that the schedule \mathcal{O} remains compatible w.r.t. to the new basic subschedule. Therefore, the maximum possible profit of the optimal offline algorithm w.r.t. this new basic subschedule is at least as large as the original offline schedule. In this way, we have shown that the competitive ratio of the original instance is no larger than the new one without the condition (2) abortion.

Lemma 5. *The optimal schedule \mathcal{O} remains compatible w.r.t. the new basic subschedule, in the sense that the requests in each broadcast in \mathcal{O} that are pending in the online algorithm will not cause an abortion in the basic subschedule.*

Proof. Suppose J starts at time t_1 and would finish at time t_2 if not aborted, and R aborts J at time t_3 and finish at time t_4 . We have that $t_1 < t_3 < t_4 < t_2$ and $|J| < |R|$. Consider a broadcast O in \mathcal{O} which starts at time u_1 and finishes at time u_2 , where $t_1 \leq u_1 < t_4$. Here we only consider the part of O that is not charged to earlier requests completed by the online algorithm.

Case 1. $t_1 \leq u_1 < t_3$ and $u_2 \geq t_2$. In this case $|O| \leq \beta|J| < \beta|R|$.

Case 2. $t_1 \leq u_1 < t_3$ and $u_2 < t_2$. In this case $|O| \leq |J| < |R|$.

Case 3. $t_3 \leq u_1 < t_4$ and $u_2 \geq t_4$. In this case $|O| \leq \beta|R|$.

Case 4. $t_3 \leq u_1 < t_4$ and $u_2 < t_4$. In this case $|O| \leq |R|$.

In all cases the broadcast O would not cause abortion in the new basic subschedule. \square

The same transformation is applied if R in turn is aborted by another broadcast with condition (2), and so on. At the end, all abortions are of condition (1).

For the second assumption, if there is any broadcast with length $< \Delta$ in the schedule, we replace it with another broadcast with length Δ and the same profit. The profit of \mathcal{A} remains the same. The offline schedule \mathcal{O} (with suitably adjusted starting times of the broadcasts) remains compatible with the new basic subschedule and hence the maximum possible offline profit corresponding to the basic subschedule can only increase. \square

5 Variable Page Lengths: Lower Bound

In this section, we give a lower bound on the competitive ratio of any online algorithm for Broadcasting. Recall that Δ is the ratio between the length of the longest and shortest page; we assume Δ is an integer, otherwise we can simply replace it with $\lfloor \Delta \rfloor$. Our result shows that the competitive ratio of any online algorithm cannot be smaller than $\Omega(\Delta / \log \Delta)$.

Theorem 5.1. *The competitive ratio of any deterministic online algorithm for Broadcasting cannot be smaller than $\Omega(\Delta / \log \Delta)$.*

Proof. Assume that there are two pages, P and Q whose lengths are Δ and 1, respectively. Given any online algorithm \mathcal{A} , we construct a sequence of requests as follows. At time 0, a request for P arrives with deadline at time Δ , i.e., it has a tight deadline. The weight of the request is 1. There are at most Δ request for Q , denoted by r_i for $0 \leq i \leq \Delta - 1$. r_i arrives consecutively, i.e., $a(r_i) = i$, and they all have tight deadlines, i.e., $d(r_i) = i + 1$. The weight of r_i , i.e., $w(r_i)$, is $\Delta^r(i+1)^k$ where r and k are some constants which will be defined later. If \mathcal{A} broadcasts Q at any time t , no more request of r_i arrives for $i > t$.

Now we analyze the performance of \mathcal{A} against the optimal offline algorithm \mathcal{O} . There are two cases. (1) If \mathcal{A} satisfies the request for P by broadcasting P at time 0, \mathcal{O} will satisfy all requests r_i by broadcasting Q at time i for $0 \leq i \leq \Delta - 1$. Hence, we have $|\mathcal{A}| = \Delta$ and $|\mathcal{O}| = \sum_{i=0}^{\Delta-1} \Delta^r(i+1)^k$. Since $\sum_{i=1}^x i^y = \Theta(x^{y+1}/(y+1))$, $|\mathcal{O}|/|\mathcal{A}| = \Theta(\Delta^{r-1}\Delta^{k+1}/(k+1)) \geq \Theta(\Delta^{r+k}/(k+1))$.

(2) If \mathcal{A} broadcasts Q at time t , only r_t can be satisfied. However, \mathcal{O} can either satisfy the request for P by broadcasting P at time 0 or satisfy all r_i by broadcasting Q at time i for $0 \leq i \leq t$. We have $|\mathcal{A}| = \Delta^r(t+1)^k$ and $|\mathcal{O}| = \max\{\Delta, \sum_{i=0}^t \Delta^r(i+1)^k\}$. Hence, $|\mathcal{O}|/|\mathcal{A}| = \max\{\Delta, \Theta(\Delta^r(t+1)^{k+1}/(k+1))\}/\Delta^r(t+1)^k = \max\{\Delta^{1-r}/(t+1)^k, \Theta((t+1)/(k+1))\}$. In order to minimize the ratio, \mathcal{A} should choose $t = (\Delta^{1-r}(k+1))^{1/(k+1)} - 1$. In that case, the ratio is $\Theta(\Delta^{(1-r)/(k+1)}/(k+1)^{k/(k+1)}) \geq \Theta(\Delta^{(1-r)/(k+1)}/(k+1))$.

In order to maximize the minimum ratio among the two cases, we let $r = (1 - k - k^2)/(k + 2)$. Hence, the competitive ratio is $\Theta(\Delta^{1-1/(k+2)}/(k+1)) \geq \Theta(\Delta^{1-1/(k+2)}/(k+2))$. We further let $k + 2 = \ln \Delta$ where the function $\Delta^{1-1/(k+2)}/(k+2)$ achieves the maximum, i.e., $\Delta^{1-1/\ln \Delta}/\ln \Delta$. Since $\Delta^{1/\ln \Delta}$ is the constant e , i.e., the base of natural logarithm, we have proved that the competitive ratio is $\Omega(\Delta/\log \Delta)$. \square

6 Lower Bound for Small Δ

The current lower bound for the broadcasting problem is 4 when $\Delta = 1$ [13]. The $\Omega(\Delta/\log \Delta)$ lower bound we just proved as well as the previous $\sqrt{\Delta}$ one [4] give very small lower bounds (much smaller than 4) for small values of Δ . In this section we give better lower bounds for this case. This is achieved by further refining the idea used in [13].

We first show the following lemma, which is similar to the one in [13] but with an extra $k > 1$ term.

Lemma 6. *Consider a strictly increasing sequence $S(\alpha, k)$ of positive numbers satisfying the recurrence*

$$v_{i+1} \leq \alpha v_i - k(v_1 + \dots + v_i).$$

Then when $(1 + \alpha - k)^2 - 4\alpha < 0$, $S(\alpha, k)$ is finite.

Proof. We consider the sequence $\{x_i\}_{i \geq 1}$ where $x_1 \geq v_1$ and $x_{i+1} = \alpha x_i - k(x_1 + \dots + x_i)$, i.e., defined by equality instead of inequality sign. It can be shown that $v_i \leq x_i$ as long as $x_i > 0$, so it is sufficient to consider x_i . Define $S_i = x_1 + \dots + x_i$. We have

$$\begin{aligned} S_{i+1} &= x_{i+1} + S_i \\ &= \alpha x_i - kS_i + S_i \\ &= \alpha(S_i - S_{i-1}) + (1 - k)S_i \\ &= (\alpha + 1 - k)S_i - \alpha S_{i-1} \end{aligned}$$

which is a second-order recurrence equation. The roots of the characteristic equation are $\frac{1}{2}(\alpha + 1 - k \pm \sqrt{D})$, where $D = (\alpha + 1 - k)^2 - 4\alpha$. When $D < 0$, the sequence $\{S_i\}$ exhibits a sinusoidal form and therefore is negative for finite i . Hence x_i cannot be always positive and $S(\alpha, k)$ is finite. \square

Let α be the unique positive real root of the equation

$$2\alpha^2 - 4\alpha^{3/2} + 2\alpha + 1 = \sqrt{(\Delta - 1)^2 + 4\alpha^2} + \Delta. \quad (7)$$

The following table shows some values of α .

Δ	1	2	3	4	10	very large
α	4	4.245	4.481	4.707	5.873	$\sqrt{\Delta}$

Theorem 6.1. *For $\Delta \geq 2$, no deterministic algorithm for Broadcasting can be better than α -competitive, where α is the unique positive root of (7).*

Proof. Let $0 < \epsilon < 1$ be an arbitrary real. We give an adversary \mathcal{O} that forces any online algorithm \mathcal{A} to have a competitive ratio of $\alpha - \epsilon$. We simplify the notation by calling a request with profit p simply as ‘the request p ’ or ‘the page p ’ (in the construction below, each request is asking for a different page).

We use the SET(*,*,*,*) notation in Woeginger’s paper [13]. For $0 < v \leq w$ and $d, \delta > 0$, define SET(v, w, d, δ) to be a set of requests $\{j_1, j_2, \dots, j_q\}$ with the following properties:

- The profits $v(j_i)$ of the requests fulfill $v(j_1) = v$, $v(j_q) = w$ and $v(j_i) < v(j_{i+1}) \leq v(j_i) + \delta$ for $1 \leq i \leq q - 1$.
- The arrival time $a(j_i)$ and deadline $d(j_i)$ of the requests fulfill $0 < a(j_1) < a(j_2) < \dots < a(j_q) < d < d(j_1) < d(j_2) < \dots < d(j_q)$. Note that every pair of requests overlaps.
- All requests have tight deadlines and length Δ .

We will use some very small positive numbers d , δ and $\delta_i = \delta/2^i$. The adversary will proceed in several steps. In step i , $i \geq 1$, some SET($v_{i-1}, w_{i-1}, d_{i-1}, \delta_{i-1}$) will be released. There are also requests $x_{i-1}^{(1)}, \dots, x_{i-1}^{(\Delta-1)}$. Depending on \mathcal{A} ’s action, another request y_{i-1} may be released as well. All the $x_{i-1}^{(j)}$ ’s and y_{i-1} have length 1 and tight deadlines.

At step 1, SET(v_0, w_0, d_0, δ_0) = SET(1, α, d, δ) arrives at time 0. Note that \mathcal{A} can only satisfy one of the requests in a set. Also, $x_0^{(j)}$ with profit 1 arrives at time $j - 1$, for $j = 1, \dots, \Delta - 1$.

If \mathcal{A} broadcasts either the first request in the set or one of the $x_0^{(j)}$ ’s, no more requests arrive (including those unreleased $x_0^{(j)}$ ’s). \mathcal{A} obtains a profit of 1 and \mathcal{O} broadcasts page α , achieving a competitive ratio of α .

So suppose \mathcal{A} broadcasts a page b_0 , $1 < b_0 \leq \alpha$. Let the request immediately preceding it in the set be a_0 , where $a_0 \geq b_0 - \delta$. The adversary releases a request y_0 where $d(y_0) = (d(a_0) + d(b_0))/2$ and $a(y_0) = d(y_0) - 1$. Its profit will be specified later. Note that the deadline of y_0 is between that of a_0 and b_0 . The spans (interval between arrival time and deadline) of the $x_0^{(j)}$ ’s and y_0 do not overlap but together almost cover the span of a_0 . See Figure 2.

(1) If \mathcal{A} aborts b_0 and broadcasts y_0 , then a shifted copy of SET(v_1, w_1, d_1, δ_1) is released, where $v_1 = y_0$, $w_1 = \alpha v_1 - b_0$, $d_1 = d(y_0) - d(a_0)$, and the shifting is done in such a way that the arrival

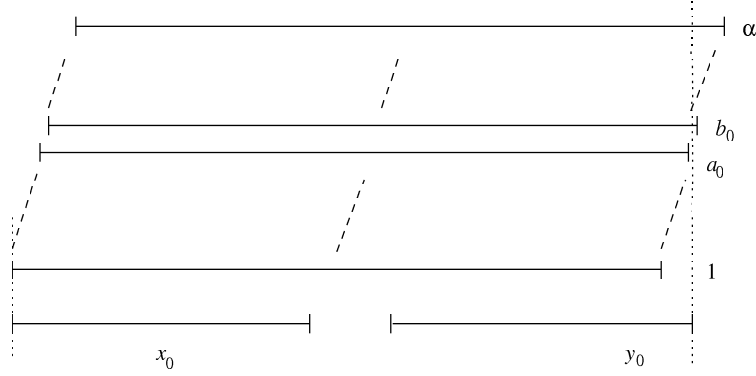


Figure 2: SET(1, α, d, δ) together with $x_0^{(1)}$ and y_0 . Here $\Delta = 2$.

times of all the new requests lie in the interval $[d(a_0), d(y_0)]$. \mathcal{A} must abort and switch to the new set or else \mathcal{O} broadcasts pages w_1 and a_0 , giving a competitive ratio $(\alpha v_1 - b_0 + a_0)/y_0 \geq \alpha - \delta$.

(2) If \mathcal{A} continues broadcasting b_0 , then again a shifted copy of SET(v_1, w_1, d_1, δ_1) is released, where $v_1 = b_0$, $w_1 = \alpha v_1 - (\sum x_0^{(j)} + y_0) = \alpha v_1 - (\Delta - 1 + y_0)$, $d_1 = d(b_0) - d(y_0)$, and the shifting is done in such a way that the arrival times of all the new requests lie in the interval $[d(y_0), d(b_0)]$. \mathcal{A} must abort and switch to the new set or else \mathcal{O} broadcasts pages w_1 , all the $x_0^{(j)}$'s, and y_0 , giving a competitive ratio $(\Delta - 1 + y_0 + w_1)/b_0 = \alpha$.

Cases (1) and (2) can be unified as follows. Set the value of y_0 such that the ratios b_0/y_0 and $(\Delta - 1 + y_0)/b_0$ are equal, i.e. $y_0 = \frac{\sqrt{(\Delta-1)^2 + 4b_0^2} - (\Delta-1)}{2}$. Denote this common ratio by k_1 . Let $k = \frac{\sqrt{(\Delta-1)^2 + 4\alpha^2} + (\Delta-1)}{2\alpha}$. Note that $k \leq k_1$. Then in either case we define $w_1 = \alpha v_1 - k v_1$, where $v_1 = b_0$ or y_0 depending on which page \mathcal{A} broadcasts in the previous set. If \mathcal{A} does not switch to the new set, \mathcal{O} gets at least $w_1 + k_1 v_1 - \delta = \alpha v_1 + (k_1 - k)v_1 - \delta \geq \alpha v_1 - \delta$, while \mathcal{A} only gets v_1 , so the competitive ratio is at least $\alpha - \epsilon$ for δ sufficiently small.

In general, in Step i , \mathcal{A} is serving either a request b_{i-1} in SET($v_{i-1}, w_{i-1}, d_{i-1}, \delta_{i-1}$) or y_{i-1} . Let a_{i-1} be the request immediately preceding b_{i-1} in the set. The adversary releases SET(v_i, w_i, d_i, δ_i) and $x_i^{(j)}$'s for the next Step ($i + 1$), with $v_i = b_{i-1}$ or y_{i-1} (depending on which one \mathcal{A} broadcasts), $w_i = \max\{\alpha v_i - k(v_1 + \dots + v_i), v_i\}$, $d_i = (d(b_{i-1}) - d(a_{i-1}))/2$, and $x_i = v_i$. If \mathcal{A} serves y_{i-1} in the previous step, all requests in this set arrive between $d(a_{i-1})$ and $d(y_{i-1})$, otherwise (if \mathcal{A} serves b_{i-1}) all requests in this set arrive between $d(y_{i-1})$ and $d(b_{i-1})$.

Let $k_{i+1} = \frac{\sqrt{(\Delta-1)^2 + 4(b_i/v_i)^2} + (\Delta-1)}{2b_i/v_i}$. If \mathcal{A} does not switch to requests in the $(i + 1)$ -th SET, or switch to the lowest-profit request in it, or switch to one of the $x_i^{(j)}$'s, no more requests arrive and \mathcal{O} gets a profit at least

$$\begin{aligned} & w_i + (k_1 v_1 - \delta) + \dots + (k_i v_i - \delta_{i-1}) \\ &= \alpha v_i - k(v_1 + \dots + v_i) + (k_1 v_1 + \dots + k_i v_i) - \sum_{j=1}^{i-1} (\delta/2^{j-1}) \\ &\geq \alpha v_i - 2\delta. \end{aligned}$$

Since \mathcal{A} only gets a profit of v_i , the ratio is at least $\alpha - \epsilon$ for δ sufficiently small.

If \mathcal{A} switches to broadcast a page b_i in the new set, a request y_i with profit $v_i \left(\frac{\sqrt{(\Delta-1)^2 + 4(b_i/v_i)^2} - (\Delta-1)}{2} \right)$ is released, and a new SET(*,*,*,*) is released with parameters and arrival times depending on whether \mathcal{A} broadcasts b_i or y_i . Then the same process repeats.

In the beginning, we have $v_i < v_{i+1} < w_i = \alpha v_i - k(v_1 + \dots + v_i)$. However, by Lemma 6, this cannot go on forever for appropriately chosen α and k . Suppose at Step $(i+2)$ the recurrence terminates, i.e. $w_{i+2} = v_{i+2} > \alpha v_{i+2} - k(v_1 + \dots + v_{i+2})$. In the final step $(i+3)$, a SET($v_{i+2}, v_{i+2}, d_{i+2}, \delta_{i+2}$) consisting of a single request with profit v_{i+2} is released. No matter how \mathcal{A} chooses it obtains a profit of v_{i+2} , while \mathcal{O} obtains a profit at least

$$\begin{aligned} v_{i+2} + \sum_{j=1}^{i+2} (k_j v_j - \delta_{j-1}) \\ &\geq \alpha v_{i+2} - k \sum_{j=1}^{i+2} v_j + \sum_{j=1}^{i+2} k_j v_j - 2\delta \\ &\geq \alpha v_{i+2} - 2\delta, \end{aligned}$$

so the ratio is still at least $\alpha - \epsilon$ for δ sufficiently small.

We have $k = \frac{\sqrt{(\Delta-1)^2 + 4\alpha^2} + (\Delta-1)}{2\alpha}$, and Lemma 6 requires $(1 + \alpha - k)^2 - 4\alpha < 0$. They together gives equation (7). This α is the lower bound on the competitive ratio. \square

7 Conclusion

In this paper we give improved competitive algorithms for the broadcasting problem, both in the case of unit page length and different page lengths. We also give lower bounds for the variable page length case. In the unit page length case, whether the true bound is different from scheduling tight intervals (which has a tight bound of 4) remains unknown. In the variable page length case, whether the competitive ratio is linear in Δ or not also remains open.

References

- [1] D. Aksoy and M. Franklin. Scheduling for large scale on-demand data broadcast. In *Proc. of IEEE INFOCOM*, pages 651–659, 1998.
- [2] Y. Bartal and S. Muthukrishnan. Minimizing maximum response time in scheduling broadcasts. In *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, pages 558–559, 2000.
- [3] A. Borodin and R. El-yaniiv. *Online computation and competitive analysis*. Cambridge University Press, 1998.
- [4] Wun-Tat Chan, Tak-Wah Lam, Hing-Fung Ting, and Prudence W.H. Wong. New results on on-demand broadcasting with deadline via job scheduling with cancellation. In *10th International Computing and Combinatorics Conference*, LNCS 3106, pages 210–218, 2004.
- [5] J. Edmonds and K. Pruhs. Broadcast scheduling: when fairness is fine. In *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 421–430, 2002.

- [6] T. Erlebach and A. Hall. NP-hardness of broadcast scheduling and inapproximability of single-source unsplittable min-cost flow. In *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 194–202, 2002.
- [7] R. Gandhi, S. Khuller, Y.A. Kim, and Y.C. Wan. Algorithms for minimizing response time in broadcast scheduling. *Algorithmica*, 38(4):597–608, 2004.
- [8] S. Jiang and N. Vaidya. Scheduling data broadcasts to “impatient” users. In *Proc. ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pages 52–59, 1999.
- [9] B. Kalyanasundaram, K. Pruhs, and M. Velauthapillai. Scheduling broadcasts in wireless networks. In *Proc. 8th European Symposium on Algorithms*, LNCS 1879, pages 290–301, 2000.
- [10] B. Kalyanasundaram and M. Velauthapillai. On-demand broadcasting under deadline. In *Proc. 11th European Symposium on Algorithms*, volume 2832 of *LNCS*, pages 313–324, 2003.
- [11] J.-H. Kim and K.-Y. Chwa. Scheduling broadcasts with deadlines. *Theoretical Computer Science*, 325(3):479–488, 2004.
- [12] DirecPC Home Page. <http://www.direcpc.com/>.
- [13] Gerhard J. Woeginger. On-line scheduling of jobs with fixed start and end times. *Theoretical Computer Science*, 130:5–16, 1994.
- [14] Feifeng Zheng, Francis Y. L. Chin, Stanley P. Y. Fung, Chung Keung Poon, and Yinfeng Xu. A tight lower bound for job scheduling with cancellation. *Information Processing Letters*, 97(1):1–3, 2006.