

# On-demand Bounded Broadcast Scheduling with Tight Deadlines\*

Chung Keung Poon<sup>1</sup>

Feifeng Zheng<sup>2</sup>

Yinfeng Xu<sup>2</sup>

<sup>1</sup> Department of Computer Science, City University of Hong Kong, Hong Kong

<sup>2</sup> School of Management, Xi'an Jiaotong University, Xi'an, Shaanxi, China

Email: ckpoon@cs.cityu.edu.hk,

zhengff@mailst.xjtu.edu.cn, yfxu@mail.xjtu.edu.cn

## Abstract

We investigate a scheduling problem motivated by pull-based data delivering systems where there is a server keeping a number of pages; and clients requesting the same page can be satisfied simultaneously by one broadcast.

The HEU algorithm of Woeginger (1994) is proven to be optimal in maximizing the number of satisfied requests when the pages have equal length and the requests have tight deadlines. However, we show that when there are maximum bounds on the number and weight of requests at any time in the system, the HEU algorithm is not optimal. We then propose a modified algorithm, VAR, which is optimal for this case.

*Keywords:* Competitive analysis; Broadcast scheduling; Tight deadline.

## 1 Introduction

In this paper, we study an on-line scheduling problem motivated by pull-based broadcasting systems. In such a system, there is a server with a number of pages to be broadcasted to clients upon requests. Each request  $r$  arrives at an arbitrary time  $a(r)$  and requests for some page  $p(r)$ . The request is satisfied if page  $p(r)$  is broadcasted completely before the deadline  $d(r)$ . Since multiple requests to the same page can be satisfied simultaneously in a single broadcast, the challenge here is to determine the schedule, i.e., which page to broadcast at what time, so as to satisfy as many simultaneous requests as possible. More precisely, our objective is to maximize the number of satisfied requests. Furthermore, we assume that the requests are given in an on-line fashion: the attributes of a request  $r$  are only known at the time of arrival  $a(r)$ . Before that, nothing about  $r$  (not even its existence) is known. Thus we need to design an on-line algorithm which makes decisions on the fly as the requests come. An on-line algorithm  $A$  is said to be  $c$ -competitive if on every input, the number of satisfied requests in the schedule produced by an optimal offline algorithm, denoted by OPT, is at most  $c$  times that of  $A$ .

It is easy to see that without pre-emption, an on-line algorithm  $A$  cannot have a bounded competitive ratio. Imagine  $A$  is broadcasting a page for a set  $J$  of

requests when another set  $R$  of  $\alpha|J|$  new requests for another page with tight deadlines arrive. If  $A$  does not abort  $J$ , OPT will broadcast the page requested by  $R$  and no other requests come later. Thus, the number of requests satisfied by OPT is  $\alpha$  times that satisfied by  $A$ . Since  $\alpha$  can be arbitrarily large,  $A$  does not have a bounded competitive ratio. In this paper, we consider the *preemption-restart model* in which the server may pre-empt a broadcast before its completion and the pre-empted request can only be satisfied by broadcasting the requested page from the beginning again.

Previously, Jiang and Vaidya (1998) studied the problem assuming knowledge of the probability distribution of the requests. Kim and Chwa (2003) were the first to design algorithms with provable worst case performance bounds without assuming such knowledge. When pages are of equal length, they presented an online algorithm (called AC) and proved it to be 5.828-competitive. Using a tighter analysis, Chan et al (2004) showed that their GREEDY algorithm is 5-competitive for the general case and 4-competitive for the special case where all requests have tight deadlines. (The GREEDY algorithm is actually the AC algorithm with a modified parameter which we called the *abortion ratio*.)

This special case of tight deadlines is especially interesting. Here, a request  $r$  must be served immediately upon its arrival or else it can never be satisfied. This case is actually related to the *weighted interval scheduling problem* in which the input is a set of weighted intervals with fixed starting and ending points and the goal is to schedule the intervals so that no two scheduled intervals overlap and the weighted sum of scheduled intervals is maximized. By viewing the number of requests to the same page in our broadcast scheduling problem as the weight of an interval in the weighted interval scheduling problem, it is easy to see that algorithms for on-line interval scheduling can be applied to our problem. For this problem, Woeginger (1994) proposed the HEU algorithm (which is the same as GREEDY) and proved it to be 4-competitive. Furthermore, he showed that it is optimal by giving a matching lower bound for any deterministic algorithm.

To overcome the lower bound of 4, Miyazawa and Erlebach (2004) considered randomization. In particular, they designed a randomized algorithm called VirtualWeight that has an expected competitive ratio of 3 when the weights are monotonic non-decreasing and the intervals have agreeable endpoints, i.e., intervals that have earlier start point will have earlier end point. This leads to the following question: Are there other special cases of the problem in which we can obtain an on-line algorithm with competitive ratio less than 4?

In many situations, various kinds of information to be broadcasted may be of different importance, or the

\*This research was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administration Region, China [Project No. City U 1164/04E] and by NSF of China under grants 10371094 and 70471035.

Copyright ©2006, Australian Computer Society, Inc. This paper appeared at Computing: The Australasian Theory Symposium (CATS2006), Hobart, Australia. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 51. Barry Jay and Joachim Gudmundsson, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

usefulness of the same information varies for different requesters or at distinct time periods. To model this situation, we assume that the requests are weighted to represent their different importance. Moreover, there may be an upper bound on the number of requests that can be satisfied by one broadcast in some situations. For example, it can be due to the limitation of the physical system, or simply because the number of clients in a system is bounded. Based on the above two factors, if the weight of an arbitrary request has a bounded range, the maximum total weight gained in a broadcast (i.e., the sum of the weight of every request satisfied in that broadcast) is also bounded. We denote by  $M$  such an upper bound.

### 1.1 Our Results

In this paper, we show that when each page has equal length and there is an upper bound  $M$  on the satisfied weight in one broadcast, the deterministic algorithm, HEU, has competitive ratio less than 4. Note that the lower bound of Woeginger applies to the unit length job case as well. Thus, the upper bound on the total weight of satisfied requests in one broadcast is essential for achieving a competitive ratio of less than 4. Furthermore, we show that HEU is actually not an optimal algorithm in this case. We show that there exists an on-line algorithm with competitive ratio arbitrarily close to  $c^*$  where  $c^*$  is a value depending on  $M$ . The main novelty in our algorithm is that it adaptively changes its abortion ratio according to the recent history. We also show a matching lower bound for any deterministic algorithm. Thus our algorithm is optimal for this case. To implement the algorithm, we need to compute the value of  $c^*$ . Unfortunately, we are not able to derive a closed-form formula for  $c^*$  in terms of  $M$ . For different values of  $M$ , we compute numeric values on  $c^*$  and compare them with those of HEU.

### 1.2 Related Work

Broadcast scheduling is such an important problem that various different models have been studied in the literature. The one closest to ours seems to be that of Kalyanasundaran and Velauthapillai (2003) which assumes that the server has a number of files partitioned into pages of equal length and a client can receive a file as long as the pages are sent in a cyclic order. In another model, the server is allowed to break down a page into different parts and send them over the network simultaneously. Yet, some may assume that the receivers have certain amount of buffer so that part of a page previously broadcasted can be cached. All these are not allowed in our model.

For our model, most of the previous works 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, Acharya and Muthukrishnan (1998) and Edmonds and Pruhs (2002) studied the problem of minimizing the total flow time while Bartal and Muthukrishnan (2000) studied the minimization of the maximum flow time. Aksoy and Franklin (1998) presented a practical parameterized algorithm and evaluated it with extensive experiments.

The rest of the paper is organized as follows. Section 2 gives some basic notations and preliminaries on our problem. In section 3 we analyze the competitiveness of HEU, and in section 4 we present and analyse our heuristic algorithm VAR. We prove a lower bound of competitive ratio in section 5 and present some numeric computations on the competitive ratios of HEU and VAR in section 6. Finally, section 7 concludes this paper.

## 2 Notations and Preliminaries

Since all the pages have equal length, we assume without loss of generality that each page has unit length. Given an input  $I$  (a set of requests) and an on-line algorithm  $A$ , we denote by  $S_A(I)$  and  $S^*(I)$  the schedules produced by  $A$  and by an optimal off-line algorithm on  $I$  respectively. When  $A$  and  $I$  are understood from the context, we will simply denote the schedules by  $S$  and  $S^*$  respectively.

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  $i < i'$ . Denote by  $|J_i|$  the total weight of requests in  $J_i$ . For convenience, assume that each request has weight at least 1 and hence  $|J_i| \geq 1$ . If  $s(J_i) + 1 > s(J_{i+1})$ , then the broadcast  $J_i$  is *aborted* by  $J_{i+1}$ ; otherwise  $J_i$  is said to be *completed*. We denote by  $|S|$  the total weight of satisfied requests in the schedule  $S$ , i.e., we only count those completed requests.

To analyze a schedule produced by an arbitrary algorithm  $A$ , we often find it convenient to partition the schedule into *basic subschedules*. A basic subschedule is a maximal sequence of broadcasts  $\delta = (J_0, J_1, \dots, J_m)$  such that for all  $0 \leq i \leq m-1$ ,  $J_i$  is aborted by  $J_{i+1}$  and only  $J_m$  is completed. Since the sequence is maximal, there is either an idle interval or a completed broadcast immediately before  $J_0$ . Note that  $m$  may equal 0 if there is no aborted broadcasts in  $\delta$ .

## 3 The HEU Algorithm

In this section we will analyse the performance of HEU. For simplicity, we assume  $M = 2^k$  for some integer  $k \geq 1$ .

Suppose HEU is broadcasting a page for a set  $J$  of requests when new requests arrive and let  $R$  be the request set with the largest weight. (If there are more than one pages having requests with the same weight, ties are broken arbitrarily.) Then HEU aborts  $J$  if and only if  $R$  is at least twice the size of  $J$ . (We say that HEU has abortion ratio 2.) Note that there is no pending requests since the requests have tight deadlines. Also, aborted requests will never be satisfied.

**Theorem 3.1** *Suppose each broadcast can satisfy a total weight of at most  $M = 2^k$  for some positive integer  $k$ . Then HEU is  $4(1 - 1/M)$ -competitive.*

**Proof.** The proof technique is similar to that of Chan et al. (2004). Consider a basic subschedule  $\delta = (J_0, J_1, \dots, J_m)$  produced by HEU. Recall that  $J_i$  ( $0 \leq i \leq m-1$ ) is aborted while  $J_m$  is completed by HEU. Thus, HEU gains  $|J_m|$  in this basic subschedule.

For OPT, denote by  $O_i$  the broadcast OPT starts when HEU is serving  $J_i$  ( $0 \leq i \leq m$ ), i.e.,  $O_i$  is the broadcast started by OPT in the interval  $[s(J_i), s(J_{i+1}))$  for  $i < m$  and in  $[s(J_i), s(J_i) + 1)$  for  $i = m$ . Let  $|O_i|$  be the total weight of requests in  $O_i$ . Since OPT has complete knowledge of all the requests, we assume that all its broadcasts will run to completion. Thus, OPT can gain  $\sum_{i=0}^m |O_i|$  within this basic subschedule. On the other hand, requests in  $J_i$  can never be satisfied later by OPT due to their tight deadlines. We will show that  $\sum_{i=0}^m |O_i|$  is at most  $4(1 - 1/M)$  of  $|J_m|$ . Clearly, this implies the same overall competitive ratio.

By the construction of HEU,  $|J_i| \leq \frac{1}{2}|J_{i+1}|$  ( $0 \leq i \leq m-1$ ) and hence  $|J_m| \geq 2^{m-i}|J_i| \geq 2^m|J_0|$ .

On the other hand, we have  $|O_i| < 2|J_i| \leq |J_{i+1}|$ , or else HEU will abort  $J_i$  to start  $O_i$ , contradicting that  $J_i$  is aborted by  $J_{i+1}$ . Thus,  $|O_i| < (\frac{1}{2})^{m-i-1}|J_m|$  and hence

$$\begin{aligned} \sum_{i=0}^m |O_i| &< \sum_{i=0}^{m-1} (\frac{1}{2})^{m-i-1} |J_m| + |O_m| \\ &= 2[1 - (\frac{1}{2})^m] |J_m| + |O_m|. \end{aligned}$$

Since  $|J_0| \geq 1$ , we have  $|J_m| \geq 2^m |J_0| \geq 2^m$ . On the other hand,  $|J_m|$  is no more than the upper bound  $M = 2^k$ . Hence,  $m \leq k$ . We will discuss the relationship of  $m$  and  $k$  in two cases.

(Case 1:  $m = k$ ). In this case  $J_m$  is  $J_k$ . Since  $2^k \leq |J_k| \leq 2^k$ , we have  $|J_k| = 2^k = M$  and then  $|O_k| \leq M = |J_k|$ . Thus,  $\sum_{i=0}^k |O_i| < \{2[1 - (\frac{1}{2})^k] + 1\} |J_k| \leq 4[1 - (\frac{1}{2})^k] |J_k|$ .

(Case 2:  $m \leq k - 1$ ). Since  $O_m$  will not abort  $J_m$ ,  $|O_m| \leq 2|J_m|$  holds. Then,  $\sum_{i=0}^m |O_i| < \{2[1 - (\frac{1}{2})^m] + 2\} |J_m| \leq \{2[1 - (\frac{1}{2})^{k-1}] + 2\} |J_m| = 4[1 - (\frac{1}{2})^k] |J_m|$ .

In both cases,  $\sum_{i=0}^m |O_i|/|J_m| < 4[1 - (\frac{1}{2})^k]$ , that is, HEU is  $4(1 - 1/M)$ -competitive. ■

#### 4 The VAR Algorithm

In this section we will present and analyze our algorithm, called VAR, which is a heuristic depending on the upper bound  $M$ .

Suppose VAR is broadcasting a page for the set  $J$  of requests when new requests arrive. Let  $R$  be the largest set of requests for the same page, i.e., largest total weight, ties broken arbitrarily. Further, let  $J'$  be the empty set if the machine is idle before  $J$  was started. Otherwise, let  $J'$  be the set of broadcasts aborted by  $J$ . Then VAR will abort  $J$  to serve  $R$  if and only if  $|R| \geq c(|J| - |J'|)$  where  $c$  is a constant in the range  $(2, 4)$  to be determined according to  $M$ . Thus, we can view VAR as aborting  $J$  if  $|R|/|J|$  is larger than a certain abortion ratio which is changing according the previous aborted job.

Consider a basic subschedule  $\delta = (J_0, J_1, \dots, J_m)$  produced by VAR. Define a sequence of values  $\{r_i\}_{0 \leq i < m}$  such that  $r_i = |J_{i+1}|/|J_i|$ . Also, define another sequence  $\{\tilde{r}_i\}_{0 \leq i < m}$  such that  $\tilde{r}_0 = c$ , and for  $i \geq 1$ ,  $\tilde{r}_i = c(1 - \frac{1}{\tilde{r}_{i-1}})$ . Then the following two lemmas hold.

**Lemma 1** *For all  $i \geq 0$  such that  $\tilde{r}_i > 1$ , we have  $\tilde{r}_{i+1} < \tilde{r}_i$ .*

**Proof.**  $\tilde{r}_1 = c - c/\tilde{r}_0 < \tilde{r}_0 = c$ . Suppose that  $\tilde{r}_i < \tilde{r}_{i-1}$  hold for some  $i > 0$ . Then we have that  $\tilde{r}_{i+1} = (c - \frac{c}{\tilde{r}_i}) < (c - \frac{c}{\tilde{r}_{i-1}}) = \tilde{r}_i$  and the lemma follows. ■

**Lemma 2** *For all  $i \geq 0$  such that  $\tilde{r}_i > 1$ , we have  $r_i \geq \tilde{r}_i$ .*

**Proof.** By construction of VAR, we have that  $r_0 \geq c = \tilde{r}_0$ . Suppose that  $r_i \geq \tilde{r}_i$  holds for some  $i \geq 0$ . Then by construction of VAR,  $r_{i+1} = |J_{i+2}|/|J_{i+1}| \geq c(1 - \frac{|J_i|}{|J_{i+1}|}) = c(1 - \frac{1}{r_i}) \geq c(1 - \frac{1}{\tilde{r}_i}) = \tilde{r}_{i+1}$ . Hence, the lemma follows. ■

**Lemma 3** *Assume  $2 < c < 4$ . Then for any  $M \geq 2$ , there exists an integer  $N$  such that  $\tilde{r}_N > 1$  but  $\tilde{r}_{N+1} \leq 1$ .*

**Proof.** We will assume  $\tilde{r}_i > 1$  for all  $i \geq 0$  and derive a contradiction. Expressing  $\tilde{r}_i$  in terms of  $\tilde{r}_{i+1}$ , we have

$$\tilde{r}_i = \frac{c}{c - \tilde{r}_{i+1}}.$$

Consider the difference between consecutive elements in the sequence  $\{\tilde{r}_i\}$ , we have

$$\begin{aligned} \tilde{r}_i - \tilde{r}_{i+1} &= \frac{c - c\tilde{r}_{i+1} + \tilde{r}_{i+1}^2}{c - \tilde{r}_{i+1}} \\ &\geq \frac{c(1 - c/4)}{c - \tilde{r}_{i+1}} \\ &> \frac{c(1 - c/4)}{c - 1} \end{aligned}$$

where for the first inequality, we use the fact that  $(\tilde{r}_{i+1} - \frac{c}{2})^2 \geq 0$ , and for the second inequality, we use the assumption that  $\tilde{r}_{i+1} > 1$ . Define  $\Delta = \frac{c(1-c/4)}{c-1}$ . So  $\Delta > 0$  (for  $2 < c < 4$ ) and is independent of the subscript  $i$ . That means  $\tilde{r}_0 > \tilde{r}_N + N\Delta > 1 + N\Delta$  which is greater than  $c$  for  $N \geq 3/\Delta$ , a contradiction. Hence the lemma follows. ■

Note that  $N$  is monotonic increasing with  $c$ . We claim that the product  $\tilde{r}_0 \tilde{r}_1 \dots \tilde{r}_N$  is also monotonic increasing with  $c$ . First,  $\tilde{r}_0 = c$  and  $\tilde{r}_1 = c - 1$  are obviously monotonic increasing with  $c$ . Suppose that  $\tilde{r}_i = c(1 - \frac{1}{\tilde{r}_{i-1}})$  is monotonic increasing with  $c$  for some  $1 < i < N$ , then  $\tilde{r}_{i+1} = c(1 - \frac{1}{\tilde{r}_i})$  is monotonic increasing with  $c$  for  $(1 - \frac{1}{\tilde{r}_i})$  is monotonic increasing with  $\tilde{r}_i$ . Given an  $M \geq 2$ , we define  $c^*$  as the (unique) value of  $c \in (2, 4)$  such that  $\tilde{r}_0 \tilde{r}_1 \dots \tilde{r}_N = M$ . Then we have the following theorem.

**Theorem 4.1** *For any  $M \geq 2$ , VAR is  $c$ -competitive if  $c > c^*$  and  $2 < c < 4$ .*

**Proof.** Similar to the analysis in Theorem 3.1, we consider a basic subschedule  $\delta = (J_0, J_1, \dots, J_m)$  produced by VAR. Denote by  $|\delta|$  what VAR gains in  $\delta$ , and by  $|\delta^*|$  what OPT gains in  $\delta$  respectively. Let  $O_i$  and  $|O_i|$  be as defined before.

By construction of VAR and definition of  $r_i$ ,  $|J_m| \geq r_{m-1}|J_{m-1}| \geq \dots \geq r_{m-1} \dots r_0 |J_0| \geq r_{m-1} \dots r_0$ .

We claim that  $m - 1 < N$ . Otherwise, by Lemma 2, definition of  $c^*$ , and the assumption that  $c > c^*$ , we have that  $|J_m| \geq r_N \dots r_1 r_0 \geq \tilde{r}_N \dots \tilde{r}_1 \tilde{r}_0 > M$ , contradicting the assumption that each broadcast can satisfy a total weight of at most  $M$ . In other words, an arbitrary basic subschedule has finite length.

By the construction of  $\delta$ , VAR gains  $|J_m|$  in  $\delta$ , i.e.,  $|\delta| = |J_m|$ . Now we bound  $|\delta^*|$ . For OPT,  $|O_i| < c(|J_i| - |J_{i-1}|)$  and  $|O_0| < c|J_0|$ . Otherwise  $O_i$  would have aborted  $J_i$ , contradicting that  $J_i$  is aborted by  $J_{i+1}$ . Thus,  $|\delta^*| = \sum_{i=0}^m |O_i| < c|J_0| + \sum_{i=1}^m c(|J_i| - |J_{i-1}|) = c|J_m|$ .

Since all requests have tight deadlines, OPT cannot start any broadcast in  $\delta$  after VAR does. Hence,  $|\delta^*|/|\delta| \leq c$  and VAR is  $c$ -competitive. ■

#### 5 A Lower Bound

In this section we will prove that VAR is an optimal algorithm for this case, that is,  $c^*$  is a lower bound of the competitive ratio for any deterministic algorithm for this problem.

**Theorem 5.1** *No deterministic algorithm can be better than  $c^*$ -competitive when all requests have tight deadlines.*

**Proof.** The main idea of the proof is similar to that of Woeginger (1994). We start our proof by a lemma and a definition of request sets. With the definition of  $\tilde{r}_i$ , we have the following lemma.

**Lemma 4** *Let  $\{r_i^*\}_{i \geq 0}$  be the sequence  $\{\tilde{r}_i\}_{i \geq 0}$  when  $c = c^*$ . Then  $r_0^* = c^*$ ,  $r_1^* = c^* - 1$ , and for  $i > 1$ ,  $r_i^* = c^* - 1 - \sum_{p=1}^{i-1} \prod_{q=1}^p \frac{1}{r_{i-q}^*}$ .*

**Proof.** By the definition of  $r_i^*$ , we have that  $r_0^* = c^*$ ,  $r_1^* = c^* - 1$ , and for  $i > 1$ ,  $r_i^* = (c^* - \frac{c^*}{r_{i-1}^*})$ . Then  $r_2^* = (c^* - \frac{c^*}{r_1^*}) = c^* - 1 - \frac{c^* - r_1^*}{r_1^*} = c^* - 1 - \frac{1}{r_1^*}$ . Suppose  $r_i^* = c^* - 1 - \sum_{p=1}^{i-1} \prod_{q=1}^p \frac{1}{r_{i-q}^*}$  holds. Then  $c^* - r_i^* = 1 + \sum_{p=1}^{i-1} \prod_{q=1}^p \frac{1}{r_{i-q}^*}$ . We have that

$$\begin{aligned} r_{i+1}^* &= c^* - \frac{c^*}{r_i^*} \\ &= c^* - 1 - \frac{c^* - r_i^*}{r_i^*} \\ &= c^* - 1 - \frac{1 + \sum_{p=1}^{i-1} \prod_{q=1}^p \frac{1}{r_{i-q}^*}}{r_i^*} \\ &= c^* - 1 - \sum_{p=1}^i \prod_{q=1}^p \frac{1}{r_{i+1-q}^*}. \end{aligned}$$

Hence, the lemma follows. ■

We define request sets  $SET(J_i, R_i, \sigma_i) = \{K_{i,1}, \dots, K_{i,q}\}$  where  $K_{i,j}$  is a set of requests for the same page,  $J_i = K_{i,1}$ ,  $R_i = K_{i,q}$ , and all requests in  $SET$  are of tight deadlines. Thus, each request needs to be served on its arrival or else it cannot be satisfied. Here,  $\sigma_i = \sigma/2^i$  such that  $\sum_{i=0}^{\infty} \sigma_i < 2\sigma$  where  $\sigma > 0$  is an arbitrarily small real number and its value will be determined later. Denote by  $|K_{i,j}|$  the total weight of requests in  $K_{i,j}$ .  $SET$  has the following three properties:

1.  $|K_{i,j}| < |K_{i,j+1}| \leq |K_{i,j}| + \sigma_i$  for  $1 \leq j \leq q - 1$ .
2.  $|R_i| = r_i^* |J_i|$  where by Lemma 4,  $r_0^* = c^*$  and for  $i \geq 1$ ,  $r_i^* = c^* - 1 - \sum_{p=1}^{i-1} \prod_{q=1}^p \frac{1}{r_{i-q}^*}$ .
3. Denote by  $a(K_{i,j})$  the arrival time of request set  $K_{i,j}$ . Then  $a(K_{i,j})$  satisfies  $0 < a(K_{i,1}) < \dots < a(K_{i,q}) < a(K_{i,1}) + 1 < \dots < a(K_{i,q}) + 1$ .

Since  $a(K_{i,q}) < a(K_{i,1}) + 1$ , all request sets  $K_{i,j}$  pairwise collide with each other.

Let  $0 < \theta < 1$  be an arbitrarily small real. We will introduce an input list of request sets  $SET(*, *, *)$  and an adversary that forces an arbitrary on-line algorithm  $A$  to act poorly on the list and to have the competitive ratio at least  $c^* - \theta$ . The adversary proceeds in steps and in each step,  $A$  is fed by a  $SET(*, *, *)$ , whose exact structure depends on the behavior of  $A$  during the preceding steps, and  $A$  is forced to abort the current broadcast. After finite steps, what  $A$  gains will be a factor of approximately  $c^* - \theta$  away from that of OPT.

Step 1. The adversary releases  $SET(J_0, R_0, \sigma_0)$  where  $|J_0| = 1$ .  $A$  can process at most one of the request sets due to the construction of  $SET$ . If  $A$  selects the smallest request set  $J_0$ , the adversary will

release no more requests. OPT will serve  $R_0$  and what OPT gains is  $c^*$  times of what  $A$  does, making  $A$  lose. Thus,  $A$  has to select some other request set  $J_1$ . We define the request set arriving immediately before  $J_1$  as  $J'_1$ . OPT will serve  $J'_1$ . By construction of  $SET$ , we have  $|J'_1| \geq |J_1| - \sigma_0$ .

Step 2. The adversary presents  $SET(J_1, R_1, \sigma_1)$  to  $A$  so that it comes after OPT completes  $J'_1$  but before  $A$  completes its current service. If  $A$  does not abort the current broadcast, it cannot serve any request in  $SET(J_1, R_1, \sigma_1)$  and the adversary releases no more requests later. OPT will then select to serve  $R_1$  after finishing  $J'_1$ . In this case  $A$  completes  $|J_1|$  requests and OPT gains  $|R_1| + |J'_1| \geq r_1^* |J_1| + (|J_1| - \sigma_0) = c^* |J_1| - \sigma$ , and then  $A$  loses for  $\sigma$  can be arbitrarily small. Therefore,  $A$  is forced to abort the current broadcast when  $SET(J_1, R_1, \sigma_1)$  arrives. Moreover,  $A$  will not select the smallest set of requests in  $SET(J_1, R_1, \sigma_1)$  or else it loses for the weight of the smallest request set is the same as that of the aborted one. Denote by  $J_2$  the new set of requests selected by  $A$ , and we define the set of requests arriving immediately before  $J_2$  as  $J'_2$ . OPT will then select to serve  $J'_2$  after finishing  $J'_1$ . Similarly,  $|J'_2| \geq |J_2| - \sigma_1$ .

This is repeated over and over again. In step  $i$ ,  $SET(J_{i-1}, R_{i-1}, \sigma_{i-1})$  releases after OPT completes  $J'_{i-1}$  but before  $A$  completes its current service.  $A$  is forced again to abort the current broadcast. Otherwise if  $A$  continues the current broadcast with  $|J_{i-1}|$  requests, the adversary will release no more requests. OPT will then select to serve  $R_{i-1}$  after finishing  $J'_{i-1}$ . In this case,  $A$  gains  $|J_{i-1}|$  and OPT gains

$$|R_{i-1}| + \sum_{j=1}^{i-1} |J'_j| \geq r_{i-1}^* |J_{i-1}| + \sum_{j=1}^{i-1} |J_j| - \sum_{j=0}^{i-2} \sigma_j. \quad (1)$$

Since  $J_{i-1}$  is a request set in  $SET(J_{i-2}, R_{i-2}, \sigma_{i-2})$ ,  $|J_{i-1}| \leq |R_{i-2}| = r_{i-2}^* |J_{i-2}| \leq \dots \leq (\prod_{p=j}^{i-2} r_p^*) |J_j|$  holds for  $1 \leq j \leq i - 2$ , and hence  $\sum_{j=1}^{i-2} |J_j| \geq (\sum_{j=1}^{i-2} \prod_{p=1}^{i-j-1} \frac{1}{r_{i-1-p}^*}) |J_{i-1}|$ . Moreover,  $\sum_{j=0}^{i-2} \sigma_j < 2\sigma$ . Together with Lemma 4, inequality (1) turns out to be

$$\begin{aligned} &|R_{i-1}| + \sum_{j=1}^{i-1} |J'_j| \\ &> \left( r_{i-1}^* + 1 + \sum_{j=1}^{i-2} \prod_{p=1}^{i-j-1} \frac{1}{r_{i-1-p}^*} \right) |J_{i-1}| - 2\sigma \\ &= c^* |J_{i-1}| - 2\sigma. \end{aligned} \quad (2)$$

Then  $A$  loses for  $\sigma$  can be arbitrarily small. So,  $A$  has to make an abortion in step  $i$ .

However, by Lemmas 1 and 3, we know that  $r_i^*$  is decreasing as  $i$  increases and eventually  $r_{N+1}^* \leq 1$ . So, the list of  $SET(*, *, *)$  has finite length.

Specifically, assume  $A$  is broadcasting a set of request  $J_{N+1}$ . Then at step  $N + 2$ , we give  $SET(J_{N+1}, R_{N+1}, \sigma_{N+1})$  (which contains only two request sets both of weight  $|J_{N+1}|$ ) after OPT completes  $J'_{N+1}$  but before  $A$  completes  $J_{N+1}$  in  $SET(J_N, R_N, \sigma_N)$ . Whether  $A$  continues its current broadcast or starts a new set of requests in  $SET(J_{N+1}, R_{N+1}, \sigma_{N+1})$ , it gains  $|J_{N+1}|$ . On the other hand, OPT starts  $R_{N+1}$  after finishing  $J'_{N+1}$

in  $SET(J_N, R_N, \sigma_N)$  and then gains totally

$$\begin{aligned} & |R_{N+1}| + \sum_{j=1}^{N+1} |J'_j| \\ & \geq r_{N+1}^* |J_{N+1}| + \sum_{j=1}^{N+1} |J'_j| \\ & > c^* |J_{N+1}| - 2\sigma \end{aligned}$$

using the same analysis as inequality (2). Now set  $\sigma \leq \theta |J_{N+1}|/2$ . Then,  $A$  has competitive ratio at least  $c^* - \theta$ . Since  $\theta$  can be arbitrarily small, the theorem is proved. ■

## 6 Numeric Values

According to the definition of  $c^*$ , it depends on  $M = 2^k$ . In this section, we will discuss the variation of  $c^*$  and compare it with the competitive ratio of HEU for different value of  $M$ . In the computer program, we set  $(\tilde{r}_0 \tilde{r}_1 \cdots \tilde{r}_N) - M \leq 0.001$ , the criteria of ending computation. Figure 1 shows the relationship between  $k$  and the competitive ratio of VAR and HEU respectively where  $c^{HEU}$  denotes the competitive ratio of HEU.

$n$	$c^*$	$k$	$\tilde{r}_N$	$\tilde{r}_{N+1}$
1	2.9312	3	1.414	0.859
2	3.7486	10	1.177	0.564
3	3.9206	20	1.103	0.366
4	3.9962	100	1.024	0.094

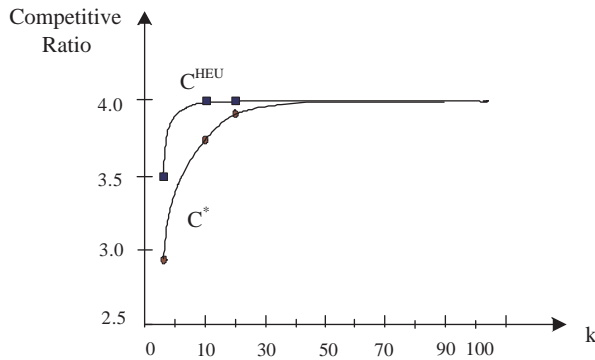


Figure 1: Relationship between  $c^*$  and  $k$ .

For VAR, when  $k = 3$ , its competitive ratio  $c^* = 2.9312$ , and as  $k$  increases,  $c^*$  increases with an obviously decreasing speed and asymptotically approaches 4 from below. Hence, we conjecture that if  $M \rightarrow \infty$ , VAR has the competitive ratio 4 in the case of tight deadline.

By figure 1,  $c^* < c^{HEU}$  holds for each  $k$ . That is, VAR is better than HEU. For instance, when  $k = (3, 10, 20, 100)$ ,  $c^* = (2.9312, 3.7486, 3.9206, 3.9962)$  and by Theorem 3.1,  $c^{HEU} = (3.50, 3.9961, 4.00, 4.00)$ .

## 7 Conclusion

In this paper, we discuss the broadcasting problem with the upper bound on the number and weight of requests in the system. We proposed the VAR algorithm and prove its competitiveness. By experiment, we show that VAR is better than HEU introduced by Woeginger (1994) in the case of tight deadline.

We also prove that VAR matches the lower bound of competitive ratio for deterministic algorithms.

Since both VAR and HEU do not consider the deadlines of requests, their competitive ratios when not all requests have tight deadlines are simply those in the tight deadline case plus one. It is conceivable that the competitive ratio can be further improved in the case of arbitrary deadline if the deadlines are taken into consideration. In fact, Zheng et al. (2004) have constructed a 4.56-competitive algorithm in Kim's model, in which they considered the deadlines of requests. An obvious open problem is: when there is an upper bound on request weight and number, does there exist a better algorithm that considers request deadline?

## References

- Aacharya, S. & Muthukrishnan, S. (1998), Scheduling on-demand broadcasts: new metrics and algorithms, *in* 'ACM/IEEE International Conference on Mobile Computing and Networking', pp. 43–54.
- Aksoy, D. & Franklin, M. (1998), Scheduling for large-scale on-demand data broadcasting, *in* 'IEEE Conference on Computer Communications (INFOCOM'98)', pp. 652–659.
- Bartal, Y. & Muthukrishnan, S. (2000), Minimizing maximum response time in scheduling broadcasts, *in* 'Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms', pp. 558–559.
- Chan, W., Lam, T., Ting, H. & Wong, P. (2004), New results on on-demand broadcasting with deadline via job scheduling with cancellation, *in* '10th International Computing and Combinatorics Conference (COCOON'04)', Vol. 3106 of *Lecture Notes in Computer Science*, pp. 210–218.
- Edmonds, J. & Pruhs, K. (2002), Broadcast scheduling when fairness is fine, *in* 'Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms', pp. 421–430.
- Jiang, S. & Vaidya, N. (1998), Scheduling algorithms for a data broadcast system: minimizing variance of the response time, Technical Report TR98-005, Texas A&M University.
- Kalyanasundaram, B. & Velauthapillai, M. (2003), On-demand broadcasting under deadline, *in* '11th European Symposium on Algorithms', Vol. 2832 of *Lecture Notes in Computer Science*, pp. 313–324.
- Kim, J. & Chwa, K. (2003), Scheduling broadcasts with deadlines, *in* '9th International Computing and Combinatorics Conference (COCOON'03)', Vol. 2697 of *Lecture Notes in Computer Science*, pp. 415–424.
- Miyazawa, H. & Erlebach, T. (2004), 'An improved randomized on-line algorithm for a weighted interval selection problem', *Journal of Scheduling* **7**, 293–311.
- Woeginger, G. (1994), 'On-line scheduling of jobs with fixed start and end times', *Theoretical Computer Science* **130**, 5–16.
- Zheng, F., Fung, S., Chin, F., Poon, C. & Xu, Y. (2004), Improved on-line broadcast scheduling with deadlines. manuscript.