

# A tight lower bound for job scheduling with cancellation <sup>☆</sup>

Feifeng Zheng <sup>a</sup>, Francis Y.L. Chin <sup>b</sup>, Stanley P.Y. Fung <sup>b,\*</sup>,  
Chung Keung Poon <sup>c</sup>, Yinfeng Xu <sup>a</sup>

<sup>a</sup> School of Management, Xi'an Jiao Tong University, China

<sup>b</sup> Department of Computer Science, The University of Hong Kong, Hong Kong, China

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

Received 19 October 2004; received in revised form 31 August 2005

Available online 5 October 2005

Communicated by M. Yamashita

## Abstract

The Job Scheduling with Cancellation problem is a variation of classical scheduling problems in which jobs can be cancelled while waiting for execution. In this paper we prove a tight lower bound of 5 for the competitive ratio of any deterministic online algorithm for this problem, for the case where all jobs have the same processing time.

© 2005 Elsevier B.V. All rights reserved.

**Keywords:** On-line algorithms; Scheduling; Lower bounds

## 1. Introduction

The Job Scheduling with Cancellation problem (JS-Cancellation) is first put forward by Chan et al. [2]. A sequence of jobs arrive at arbitrary time for scheduling on a single machine. The jobs have different processing times, profits and deadlines. A job is not known until it arrives, and the processing time, deadline and profit are known when it arrives. At each time only one job can be scheduled. When a job is being

processed, it can be preempted and restarted later from the beginning. Furthermore, there is also a sequence of *cancel requests* arriving at arbitrary time. A cancel request specifies a released job to be cancelled. If a job is being processed or is completed, a cancel request for the job will be ignored, otherwise the job will be cancelled. The objective is to maximize the total profit of satisfied jobs (i.e., those completed before their deadlines). A real-life example for JS-Cancellation is that printing jobs on a printer can be cancelled when they are waiting.

To gauge the quality of the schedules produced by an on-line algorithm, the competitive ratio analysis [1] is often used. Given an input  $I$  (a set of jobs) and an algorithm  $A$ , we denote by  $S_A(I)$  and  $S^*(I)$  the schedules produced by  $A$  and by an optimal offline algorithm on  $I$ , respectively. The competitive ratio of algorithm  $A$  is defined as  $r_A = \sup_I \frac{|S^*(I)|}{|S_A(I)|}$ , where  $|S|$  denotes the total profit of completed jobs in the schedule  $S$ .

<sup>☆</sup> The work described in this paper was fully supported by a grant from NSF of China [No. 10371094] and two grants from the Research Grants Council of the Hong Kong SAR, China [CityU 1071/02E and HKU 7142/03E].

\* Corresponding author.

*E-mail addresses:* zhengff@mailst.xjtu.edu.cn (F. Zheng), chin@cs.hku.hk (F.Y.L. Chin), pyfung@cs.hku.hk (S.P.Y. Fung), ckpoon@cs.cityu.edu.hk (C.K. Poon), yfxu@mail.xjtu.edu.cn (Y. Xu).

In [2] a 5-competitive algorithm for JS-Cancellation is given for the case where all jobs have the same processing time. In this short note we show that when all jobs have the same processing time, no deterministic online algorithm for JS-Cancellation is better than 5-competitive, thus giving a matching lower bound.

## 2. Lower bound of JS-Cancellation

For a job  $J$ , denote by  $a(J)$ ,  $p(J)$  and  $d(J)$  its arrival time, processing time and deadline, respectively. Without loss of generality, let  $p(J) = 1$ , which means each job is one unit in length.

**Theorem 2.1.** *In the case that all jobs have the same processing time, no deterministic online algorithm for JS-Cancellation is better than 5-competitive.*

**Proof.** The main idea of our proof is similar to that in Theorem 4.4 of Woeginger's paper [4], which gives a lower bound for an interval scheduling problem. First, we state an important lemma that is proved in [4].

**Lemma 1** [4]. *For  $2 < \alpha < 4$ , let  $S(\alpha)$  be a strictly increasing sequence of positive numbers  $\langle v_1, v_2, \dots \rangle$  satisfying the inequality  $v_{i+2} \leq (\alpha - 1)v_{i+1} - \sum_{j=1}^i v_j$  for every  $i \geq 0$ . Then  $S(\alpha)$  is finite.*

Define job sets  $\text{SET}(v_{\min}, v_{\max}, d, \delta) = \{j_1, \dots, j_q\}$  with the following two properties:

- (p1) The profits  $v(j_i)$  satisfy  $v(j_1) = v_{\min}$ ,  $v(j_q) = v_{\max}$ , and  $v(j_i) < v(j_{i+1}) \leq v(j_i) + \delta$  for  $1 \leq i \leq q - 1$ ;
- (p2) The jobs  $j_i$  in SET satisfy that  $0 < a(j_1) < a(j_2) < \dots < a(j_q) < d < a(j_1) + 1$ .

In the following, we set  $\alpha = 4 - \varepsilon/2$  where  $0 < \varepsilon < 1$  is an arbitrary constant. We will show how the optimal algorithm, denoted by OPT, acts according to the behavior of a deterministic online algorithm, denoted by A. We will introduce a list  $L(\varepsilon)$  of jobs, such that OPT can force A to behave poorly on  $L(\varepsilon)$  and to have a worst case ratio of at least  $5 - \varepsilon$ . OPT proceeds in several steps. In every step, some  $\text{SET}(*, *, *, *)$  is fed to A. In  $\text{SET}(*, *, *, *)$ , all the jobs have indefinite deadlines. The exact structure of  $\text{SET}(*, *, *, *)$  depends on the action of A during the preceding steps. After a finite number of steps, the schedule constructed by A will have profit a factor of approximately  $\alpha + 1 = 5 - \varepsilon/2$  away from that of the optimum schedule.

Let  $\delta < \varepsilon/4$  and  $\delta_i = \delta/2^i$  such that  $\sum_{i=0}^{\infty} \delta_i = 2\delta$ . The exact value of  $\delta$  will be determined later.

In the first step, a  $\text{SET}(\mu, \alpha\mu, 1, \delta)$  is released where  $\mu$  is any positive number. If A processes the first job with profit  $\mu$ , then OPT processes the last job with profit  $\alpha\mu$ , and no more jobs arrive. Immediately after OPT begins to serve the last job, there comes cancel requests to cancel all the jobs except the first one. So A can only gain a profit of  $\mu$ . OPT first finishes the last job with profit  $\alpha\mu$  and then processes the first job due to its indefinite deadline, so that OPT can gain a profit of  $(\alpha + 1)\mu$  in total, which makes A lose. Otherwise, if A processes some job  $J_1$  with profit  $v_1 > \mu$ , then OPT processes the job arriving just before  $J_1$  which has profit at least  $v_1 - \delta$ . Denote this job as  $J'_1$ . OPT also issues cancel requests for all jobs except  $J_1$ . Let  $d_1$  denote the difference between the completion times of  $J_1$  and  $J'_1$ . Define  $w_1 = (\alpha - 1)v_1$ . Go to the next step.

In the second step, a shifted copy of  $\text{SET}(v_1, w_1, d_1, \delta_1)$  is released, in which all jobs arrive after the completion time of  $J'_1$  but before the completion time of  $J_1$ . Hence, when the new set of jobs arrives, OPT has finished  $J'_1$  but A has not finished  $J_1$  yet. In the following we discuss three selections A may choose.

- (S1) A does not interrupt  $J_1$ . Then no more set of jobs arrive and after OPT begins to process the last job in the new set with profit  $w_1$ , there arrive cancel requests to cancel all the jobs in  $\text{SET}(v_1, w_1, d_1, \delta_1)$ . Then A can only finish  $J_1$  and gain a profit of  $v_1$ . OPT can first process  $J'_1$  in the first set, then the last job with profit  $w_1$  in the second set, and finally  $J_1$  (note that it is never canceled), so that the total profit OPT gains is at least  $(v_1 - \delta) + w_1 + v_1 = (\alpha + 1)v_1 - \delta$ . By setting  $\delta < \varepsilon v_1/2$ , A loses again. So when  $\text{SET}(v_1, w_1, d_1, \delta_1)$  releases, A is forced to interrupt  $J_1$ .
- (S2) A chooses the first job in the new set with profit  $v_1$ . Then OPT acts the same as that in (S1) and after OPT begins to serve the job with profit  $w_1$ , there arrive cancel requests to cancel all the jobs except the first one in  $\text{SET}(v_1, w_1, d_1, \delta_1)$ . Then A only gains  $v_1$ , and OPT can gain  $(\alpha + 1)v_1 - \delta$ . A loses again. Since both A and OPT gain the same profits as that in (S1), respectively, we will ignore this case in subsequent steps.
- (S3) A chooses some job  $J_2$  in the new set with profit  $v_2 > v_1$ . Let  $J'_2$  be the job arriving just before  $J_2$  which has profit at least  $v_2 - \delta_1$ . Then a cancel request arrives to cancel job  $J_1$ . Furthermore, immediately after OPT starts processing  $J'_2$ , there arrive cancel requests for all jobs in the second

set except  $J_2$ . Denote the difference of the completion times of  $J_2$  and  $J'_2$  by  $d_2$ , and define  $w_2 = (\alpha - 1)v_2 - v_1$ . Go on to the next step in which a SET( $v_2, w_2, d_2, \delta_2$ ) is released.

This is repeated over and over again. In step  $(i + 1)$ ,  $i \geq 1$ , there is a shifted copy of SET( $v_i, w_i, d_i, \delta_i$ ) in which all jobs arrive after OPT finishes  $J'_i$  but before A finishes  $J_i$ . The last job has profit  $w_i = (\alpha - 1)v_i - \sum_{j=1}^{i-1} v_j$ .

A may select action (S1) or (S3). If A selects (S1), i.e., it does not interrupt  $J_i$ , then it can gain only  $v_i$  and OPT can gain total profit  $\sum_{j=1}^i (v_j - \delta_{j-1}) + w_i + v_i = (\alpha + 1)v_i - \sum_{j=0}^{i-1} \delta_j > (\alpha + 1)v_i - 2\delta$ , then A loses when  $\delta$  is sufficiently small. So, A is forced to select (S3), i.e., it aborts  $J_i$  and selects a new job  $J_{i+1}$  with profit  $v_{i+1}$ . Define the job arriving just before  $J_{i+1}$  as  $J'_{i+1}$  with profit at least  $v_{i+1} - \delta_i$ . Then the number  $w_{i+1}$  is determined according to the equation

$$w_{i+1} = \max \left\{ (\alpha - 1)v_{i+1} - \sum_{j=1}^i v_j, v_{i+1} \right\}. \quad (1)$$

Move to the next step  $(i + 2)$  with SET( $v_{i+1}, w_{i+1}, d_{i+1}, \delta_{i+1}$ ), in which all the new set of jobs arrive after the completion time of  $J'_{i+1}$  but before the completion time of  $J_{i+1}$ .

In the beginning, we have  $v_{i+1} < v_{i+2} < w_{i+1} = (\alpha - 1)v_{i+1} - \sum_{j=1}^i v_j$ . But according to Lemma 1, after a finite number of steps the number  $v_{i+2}$  must be greater than  $(\alpha - 1)v_{i+1} - \sum_{j=1}^i v_j$ , and then  $w_{i+2} = v_{i+2}$  must hold due to Eq. (1). In the following step  $(i + 3)$ , there arrives a SET( $v_{i+2}, w_{i+2}, d_{i+2}, \delta_{i+2}$ ) with only two jobs  $J_{i+3}$  and  $J'_{i+3}$ , where  $a(J_{i+3}) = a(J'_{i+3})$ ,  $v_{i+2} = w_{i+2} \geq (\alpha - 1)v_{i+2} - \sum_{j=1}^{i+1} v_j$ , and  $a(J_{i+3})$  is later than the completion time of  $J'_{i+2}$  but earlier than the completion time of  $J_{i+2}$ . Similarly, A has to abort  $J_{i+2}$  and selects  $J_{i+3}$  (or if A selects  $J'_{i+3}$  the analysis is the same). OPT processes  $J'_{i+3}$  first, and after OPT begins  $J'_{i+3}$  there arrives a cancel request to cancel  $J'_{i+3}$ . So A only finishes  $J_{i+3}$  and gains profit  $v_{i+2}$ . OPT can process  $J_{i+3}$  after it finishes  $J'_{i+3}$ , then OPT gains a total profit at least

$$\begin{aligned} & \sum_{j=1}^{i+2} (v_j - \delta_{j-1}) + w_{i+2} + v_{i+2} \\ & \geq \sum_{j=1}^{i+2} (v_j - \delta_{j-1}) + (\alpha - 1)v_{i+2} - \sum_{j=1}^{i+1} v_j + v_{i+2} \\ & > (\alpha + 1)v_{i+2} - 2\delta \\ & \geq (5 - \varepsilon)v_{i+2} \end{aligned}$$

as long as  $\delta \leq \varepsilon v_{i+2}/4$ . Thus, OPT can gain total a profit  $5 - \varepsilon$  times what A gains.  $\square$

### 3. Concluding remarks

In this paper we give an optimal lower bound for the JS-Cancellation problem. In fact, JS-Cancellation is introduced in Chan et al.'s paper [2] as a reduction to give better upper bounds for an online broadcast scheduling problem [3,2]. In this problem, clients request for pages stored in a server. Requests arrive online and each is associated with a deadline. When the server broadcasts a page, all requests to the same page that have arrived will receive the content of the page simultaneously. The server is allowed to abort the current page it is broadcasting before its completion and start a new one. An aborted page can later be broadcasted again from the beginning. The goal is to maximize the total weighted throughput, i.e., the total weighted lengths of all satisfied requests. A 5-competitive algorithm for broadcast scheduling follows from the 5-competitive algorithm for JS-Cancellation. Our optimal lower bound for JS-Cancellation shows that it is impossible to further improve the upper bound of broadcast scheduling using the JS-Cancellation approach, and new ideas are required. In [5] a new 4.56-competitive algorithm for broadcast scheduling is given, which considers the deadlines of requests in addition to their weights.

In fact JS-Cancellation is, in a sense, a job scheduling problem with variable (or unknown) deadlines. Our result shows that JS-Cancellation is provably harder than broadcast scheduling: without knowledge of deadlines, any algorithm has to be at least 5-competitive, while with this information a 4.56-competitive algorithm exists.

### References

- [1] A. Borodin, R. El-yaniv, Online Computation and Competitive Analysis, Cambridge University Press, Cambridge, 1998.
- [2] W.-T. Chan, T.-W. Lam, H.-F. Ting, P.W.H. Wong, New results on on-demand broadcasting with deadline via job scheduling with cancellation, in: Proc. 10th Internat. Computing and Combinatorics Conference, in: Lecture Notes in Comput. Sci., vol. 3106, Springer, Berlin, 2004, pp. 210–218.
- [3] J.-H. Kim, K.-Y. Chwa, Scheduling broadcasts with deadlines, in: Proc. 9th Internat. Computing and Combinatorics Conference, in: Lecture Notes in Comput. Sci., vol. 2697, Springer, Berlin, 2003, pp. 415–424.
- [4] G.J. Woeginger, On-line scheduling of jobs with fixed start and end times, Theoret. Comput. Sci. 130 (1994) 5–16.
- [5] F. Zheng, S.P.Y. Fung, F.Y.L. Chin, C.K. Poon, Y. Xu, Improved on-line broadcast scheduling with deadlines, Manuscript, 2004.