ELSEVIER

Operations Research Letters 30 (2002) 415–420

Operations Research Letters

www.elsevier.com/locate/dsw

On-line scheduling of unit time jobs with rejection: minimizing the total completion time

Leah Epstein[a], John Noga[b], Gerhard J. Woeginger[c,d,*,1]

[a] *School of Computer and Media Sciences, The Interdisciplinary Center, P.O. Box 167, 46150 Herzliya, Israel*
[b] *Department of Computer Science, California State University, 18111 Nordhoff Street, Northridge, CA 91330-8295, USA*
[c] *Department of Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands*
[d] *Institut für Mathematik, Technische Universität Graz, Steyrergasse 30, A-8010 Graz, Austria*

Received 3 December 2001; received in revised form 6 May 2002; accepted 10 May 2002

Abstract

We consider on-line scheduling of unit time jobs on a single machine with job-dependent penalties. The jobs arrive on-line (one by one) and can be either accepted and scheduled, or be rejected at the cost of a penalty. The objective is to minimize the total completion time of the accepted jobs plus the sum of the penalties of the rejected jobs.

We give an on-line algorithm for this problem with competitive ratio $\frac{1}{2}(2+\sqrt{3}) \approx 1.86602$. Moreover, we prove that there does not exist an on-line algorithm with competitive ratio better than 1.63784. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Scheduling; On-line algorithm; Competitive analysis; Worst-case bounds

1. Introduction

Consider a system with a single machine and n jobs $J_1,\ldots,J_n$. Job J_j $(j=1,\ldots,n)$ has a *rejection penalty* e_j and a processing time $p_j=1$ on the machine. The machine can process at most one job at a time. For each job J_j, we must decide whether to accept that job or whether to reject it. The accepted jobs are to be scheduled on the machine, and we pay the sum of job completion times in the constructed schedule. For the rejected jobs, we pay the corresponding rejection penalties. In other words, the objective value is the total completion time of the accepted jobs plus the total penalty of the rejected jobs. We denote this objective function by an entry "Rej + $\sum C_j$" in the third field of the three-field scheduling notation (see e.g. [4]). Hence, the considered problem is $1|p_j = 1|\text{Rej} + \sum C_j$.

In the *off-line* version of this problem, all the job data (i.e., all the job penalties) are known a priori. This version of the problem is well understood. Engels et al. [2] give a polynomial time algorithm for it. In this note, however, we are mainly interested in the *on-line* version of this problem for which the job data are not known a priori: The jobs arrive one by one, and we do not have any knowledge on the future of the system. We decide whether the current job should be rejected (in which case we pay the penalty) or be accepted (in

* Corresponding author. Department of Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.

E-mail addresses: lea@idc.ac.il (L. Epstein), jnoga@ecs.csun.edu (J. Noga), g.j.woeginger@math.utwente.nl (G.J. Woeginger).

[1] Supported by the START program Y43-MAT of the Austrian Ministry of Science.

which case it is assigned to the machine, and we pay its completion time), and only then we learn about the next job and about its penalty.

The quality of an on-line algorithm A is usually measured by its *competitive ratio* or *worst-case ratio* R_A. This competitive ratio is defined by

$$R_A = \sup \{A(I)/\textsc{Opt}(I) \mid I \text{ is a sequence of jobs}\}. \tag{1}$$

Here, $A(I)$ denotes the objective value in the schedule constructed by the on-line algorithm A for the job sequence I, whereas $\textsc{Opt}(I)$ denotes the objective value in an optimal off-line schedule for I. In this note, we will prove the following theorem.

Theorem 1. *For the problem of minimizing total completion time plus total rejection penalty of unit time jobs on a single machine, there exists an on-line algorithm with competitive ratio* $\frac{1}{2}(2+\sqrt{3}) \approx 1.86602$.

Moreover, for this problem there does not exist an on-line algorithm with competitive ratio less or equal to 1.63784.

The proof of the positive result in this theorem is contained in Section 2, and the proof of the negative result is in Section 3. Section 4 gives some conclusions.

Let us briefly discuss the more general scheduling problem where the job processing times are arbitrary (and not necessarily unit time, as in our variant). Suppose that there exists an R-competitive on-line algorithm A for this problem, and confront it with a job with processing time $p_1 = 1/(R+1)$ and penalty 1. In case A rejects the job, then no further job may arrive. In this case, $A(I) = 1$, $\textsc{Opt}(I) = 1/(R+1)$, and A is not R-competitive. If, on the other hand, algorithm A decides to accept the job, a huge number H of jobs may arrive, all with zero processing times and all with rejection penalties 1. In this case, $A(I) \geqslant (H+1)/(R+1)$, $\textsc{Opt}(I) = 1$, and algorithm A again is not R-competitive. Hence, for this more general scheduling problem, any on-line algorithm A yields $R_A = \infty$, and thus must behave very poorly in the worst case; a hopeless situation! Note that in the above instances, all jobs have the same rejection penalty.

Let us briefly mention some related results from the literature. Scheduling with rejection was first considered by Bartal et al. [1] who concentrated on off-line and on-line results for non-preemptive makespan on parallel machines. For preemptive makespan on parallel machines with rejection, Seiden [5] considered the on-line variant, whereas Hoogeveen et al. [3] considered the off-line variant. Engels et al. [2] discuss off-line sum of completion times with rejection. All these papers discuss the computational complexity and approximability of the considered problems. Sgall [6] gives a comprehensive survey on on-line scheduling.

2. The algorithm

In this section we will prove our positive main result. An important parameter for the algorithm is $\alpha = \frac{1}{2}(1+\sqrt{3})$, the larger root of $\alpha^2 = \alpha + \frac{1}{2}$. Note that $\alpha^2 = \frac{1}{2}(2+\sqrt{3})$ equals the competitive ratio that is claimed in Theorem 1.

Our algorithm $\textsc{Greedy}$ is depicted in Fig. 1. Exactly as in the depicted algorithm, we will use acc_j to denote the number of accepted jobs from $J_1, \ldots, J_j$. Algorithm $\textsc{Greedy}$ is a quite simple greedy-type algorithm: when deciding about job J_j, it has the choice between processing the job at a cost of $\mathrm{acc}_{j-1} + 1$, and rejecting the job at a cost of e_j. $\textsc{Greedy}$ multiplies the former cost by α and the latter cost by 1, and then simply selects the cheaper alternative. Multiplying the costs accounts for the fact that an accepted job is more dangerous since it may increase the cost of later jobs, whereas a rejected job cannot.

For an instance I of the scheduling problem, we denote by $\textsc{Opt}(I)$ the objective value of the optimal off-line schedule and by $\textsc{Greedy}(I)$ the objective value of the schedule that is constructed by algorithm $\textsc{Greedy}$. In the rest of this section, we will prove that the on-line algorithm $\textsc{Greedy}$ satisfies the statement of Theorem 1. Suppose for the sake of contradiction that there exist instances I for which

$$\textsc{Greedy}(I) \geqslant \alpha^2\, \textsc{Opt}(I). \tag{2}$$

Consider such an instance I with the smallest number of jobs, and fix an optimal off-line schedule σ for it. Let $\varepsilon > 0$ be an infinitesimally small real number. We

```
1 Let $\alpha = \frac{1}{2}(1 + \sqrt{3})$
2 For every new job $J_j$ do
3     Let $\mathrm{acc}_{j-1}$ be the number of accepted jobs from $J_1, \ldots, J_{j-1}$
4     If $e_j > \alpha \cdot (\mathrm{acc}_{j-1} + 1)$ then accept $J_j$ else reject $J_j$
```

Fig. 1. Description of the on-line algorithm GREEDY.

define another instance I' by modifying the penalties of the jobs in I.

(Acc) For every job J_j that is accepted by the on-line algorithm, we create a job J_j' with penalty $e_j' = \alpha(\mathrm{acc}_{j-1} + 1) + \varepsilon$. In this case, $e_j' \leqslant e_j$ holds.
(Rej) For every job J_j that is rejected by the on-line algorithm, we create a job J_j' with penalty $e_j' = \alpha(\mathrm{acc}_{j-1} + 1)$. In this case, $e_j' \geqslant e_j$ holds.

Lemma 2. *Algorithm* GREEDY *accepts job J_j in instance I if and only if it accepts job J_j' in instance I'*:

Proof. For $j = 1$ this is straightforward to check. For $j \geqslant 2$, we use induction. Note that the induction hypothesis implies that the same number of jobs is accepted from $J_1, \ldots, J_{j-1}$ and from $J_1', \ldots, J_{j-1}'$. Consequently, job J_j' is accepted in I' if and only if $e_j' > \alpha(\mathrm{acc}_{j-1} + 1)$, and by construction this inequality holds if and only if job J_j is accepted in I. □

Lemma 3. *There does not exist a job J_j in instance I that is simultaneously rejected in the optimal off-line schedule σ and in the schedule constructed by algorithm* GREEDY.

Proof. Suppose otherwise, and consider the instance K that results by removing J_j from instance I. Then $\mathrm{GREEDY}(K) = \mathrm{GREEDY}(I) - e_j$, since the behavior of the on-line algorithm does not depend at all on the rejected jobs. Moreover, $\mathrm{OPT}(K) \leqslant \mathit{Opt}(I) - e_j$. This implies $\mathrm{GREEDY}(K)/\mathrm{OPT}(K) > \mathrm{GREEDY}(I)/\mathrm{OPT}(I) \geqslant \alpha^2$, and contradicts the minimality of I. □

Lemma 4. *For the* GREEDY *schedule and the optimal schedule of instances I and I', we have* $\mathrm{OPT}(I) \geqslant \mathrm{OPT}(I')$ *and* $\mathrm{GREEDY}(I) \leqslant \mathrm{GREEDY}(I')$.

Proof. First let us compare $\mathrm{OPT}(I')$ to $\mathrm{OPT}(I)$. Consider the schedule σ' for I' that accepts job J_j' if and only if job J_j is accepted in the optimal off-line schedule σ. Then the contribution of the accepted jobs to the objective value in σ and σ' is the same. By Lemma 3, every rejected job J_j in σ is accepted by GREEDY; therefore, the new penalty is computed in (Acc) and satisfies $e_j' \leqslant e_j$. Hence, the contribution of the rejected jobs to the objective value of σ is at least as large as their contribution to the objective value of σ'. Summarizing, this yields the claimed inequality $\mathrm{OPT}(I) \geqslant \mathrm{OPT}(I')$.

Now let us compare $\mathrm{GREEDY}(I')$ to $\mathrm{GREEDY}(I)$. By Lemma 2, the contribution of the accepted jobs to the objective values of the GREEDY schedules for I and I' is the same. The penalties of the rejected jobs are computed according to (Rej), and satisfy $e_j' \geqslant e_j$. This yields the claimed inequality $\mathrm{GREEDY}(I) \leqslant \mathrm{GREEDY}(I')$. □

Lemma 5. *For instance I', we have* $\mathrm{GREEDY}(I') < \alpha^2\, \mathrm{OPT}(I')$.

Proof. Denote by x the number of jobs that are rejected in the optimal schedule σ' for I', but are accepted in the GREEDY schedule. Denote by y the number of jobs that are accepted in σ' but rejected in the GREEDY schedule, and by z the number of jobs that are accepted in both schedules.

Algorithm GREEDY accepts $x + z$ jobs and rejects y jobs from instance I'. By the construction in (Rej), the penalty of every rejected job is bounded from above by $\alpha(x + z + 1)$. Therefore,

$$\mathrm{GREEDY}(I') \leqslant \tfrac{1}{2}(x + z)(x + z + 1) + \alpha\, y(x + z + 1). \tag{3}$$

The optimal schedule for instance I' accepts $y+z$ jobs and rejects x jobs. Every rejected job is accepted by Greedy. Hence, at the moment in time where the optimal schedule decides to reject its ℓth job, Greedy has already accepted at least $\ell-1$ jobs. By (Acc), the penalty of this ℓth job is at least $\alpha\ell+\varepsilon$. Therefore,

$$\text{Opt}(I') \geqslant \frac{1}{2}(y+z)(y+z+1) + \sum_{\ell=1}^{x}(\alpha\ell+\varepsilon)$$

$$> \frac{1}{2}(y+z)(y+z+1) + \frac{\alpha}{2}x(x+1). \quad (4)$$

We define

$$F(x,y,z) = \alpha^2(y+z)(y+z+1) + \alpha^3 x(x+1) - (x+z+1)(x+z+2\alpha y).$$

By using the inequalities in (3) and (4), we derive that

$$\alpha^2\,\text{Opt}(I') - \text{Greedy}(I') > \tfrac{1}{2}F(x,y,z).$$

Our goal now is to show that $F(x,y,z) \geqslant 0$ holds for any non-negative integers x, y, z. Clearly, this will prove the lemma. First, we consider the case where $x=z=0$. In this case we have $y \geqslant 1$, and this yields

$$F(0,y,0) = \alpha^2 y(y+1) - 2\alpha y = \alpha y(\alpha y + \alpha - 2) \geqslant 0.$$

From now on we assume that $x+z \geqslant 1$. The function $F(x,y,z)$ is a quadratic function in y where the coefficient of the quadratic term y^2 is positive. By applying calculus, we see that this quadratic function is minimized at $y^* = \frac{1}{\alpha}(x+z+1) - z - \frac{1}{2}$. This yields

$$F(x,y,z) \geqslant F(x,y^*,z)$$

$$= \left(x+z+1-\frac{\alpha}{2}\right)\left(x+z+1+\frac{\alpha}{2}\right) + \alpha^3 x(x+1) - (x+z+1) \times (3x+3z-2\alpha z+2-\alpha)$$

$$= (x+z+1)(2\alpha z - 2z - 2x - 1 + \alpha) - \frac{\alpha^2}{4} + \alpha^3 x(x+1) \quad (5)$$

$$= (x+z)(2\alpha z - 2z - 2x) + \alpha^3 x^2 + z(3\alpha-3) + x(\alpha^3+\alpha-3) + \left(\alpha - 1 - \frac{\alpha^2}{4}\right). \quad (6)$$

Now we use $3\alpha - 3 > \alpha^3 + \alpha - 3 > 0$ and $x+z \geqslant 1$ to bound the expression in line (6) by

$$z(3\alpha-3) + x(\alpha^3+\alpha-3) + \left(\alpha - 1 - \frac{\alpha^2}{4}\right) \geqslant (\alpha^3+\alpha-3) + \left(\alpha - 1 - \frac{\alpha^2}{4}\right) > 0.$$

Consequently, we get from lines (5) and (6) that

$$F(x,y,z) \geqslant (x+z)(2\alpha z - 2z - 2x) + \alpha^3 x^2$$

$$= x^2(\alpha^3-2) + 2xz(\alpha-2) + 2(\alpha-1)z^2$$

$$= \tfrac{3}{2}x^2(\alpha-1) - 2\sqrt{3}(\alpha-1)xz + 2(\alpha-1)z^2$$

$$= \tfrac{1}{2}(\alpha-1)(\sqrt{3}x - 2z)^2 \geqslant 0.$$

Here we have used that $\alpha^3 - 2 = \frac{3}{2}(\alpha-1)$ and that $\alpha - 2 = -\sqrt{3}(\alpha-1)$. This completes the proof of Lemma 5. □

Now the statements in Lemmas 4 and 5 together yield that

$$\text{Greedy}(I) \leqslant \text{Greedy}(I') < \alpha^2\,\text{Opt}(I') \leqslant \alpha^2\,\text{Opt}(I),$$

which blatantly contradicts our initial assumption (2). This completes the proof of the positive result stated in Theorem 1.

The above analysis of algorithm Greedy actually is tight: For a huge integer n, consider an instance I' as described above: First, there arrive $x \approx 2n$ jobs that are accepted by Greedy but rejected in the optimal schedule. Then there arrive $y=n$ joby that are rejected by Greedy but accepted in the optimal schedule. Finally, there arrive $z \approx \sqrt{3}\,n$ jobs that are accepted in both schedules. The penalties of these jobs are always fixed according to the rules (Acc) and (Rej). It can be verified that then $\text{Opt}(I') \approx (3+2\sqrt{3})\,n^2$ and $\text{Greedy}(I') \approx \alpha^2(3+2\sqrt{3})\,n^2$.

3. The lower bound

In this section, we will prove the negative result stated in Theorem 1. For an instance, I of the

k	e_k	OPT_k	ON_k
1	1.63	1.00	1
2	3.29	2.63	3
3	4.55	4.63	6
4	6.44	7.63	10
5	7.80	10.92	15
6	9.32	14.92	21
7	10.74	19.47	28
8	11.89	24.47	36
9	13.67	30.47	45
10	15.17	36.91	55
11	16.58	43.91	66
12	18.29	51.71	78
13	19.33	59.71	91
14	21.00	68.71	105
15	22.19	78.03	120
16	23.49	88.03	136
17	25.00	98.77	153
18	25.93	109.77	171
19	27.31	121.66	190
20	27.87	133.66	210
21	29.06	146.66	231
22	30.34	160.33	253
23	31.16	174.33	276
24	32.61	189.33	300
25	33.34	204.50	325
26	34.42	220.50	351
27	35.45	237.08	378
28	36.16	254.08	406
29	37.50	272.08	435
30	38.31	290.37	465
31	39.28	309.37	496
32	39.79	328.70	528
33	40.39	348.70	561
34	41.62	369.70	595
35	41.85	390.70	630
36	42.71	412.70	666
37	42.88	434.89	703
38	43.37	457.89	741
39	43.65	481.38	780
40	43.77	505.38	820
41	44.52	530.38	861
42	44.27	555.38	903
43	44.54	581.31	946
44	43.92	607.31	990

Fig. 2. A job sequence as in Lemma 6 with $R = 1.63$.

scheduling problem with n jobs and with penalties e_k $(k = 1, \dots, n)$, we denote by I_k $(1 \leqslant k \leqslant n)$ the restriction of I to the first k jobs, and we denote by OPT_k the optimal objective value of I_k.

Lemma 6. *Let $R \geqslant 1$. Assume that there exists an instance I with n unit time jobs and with penalties e_k $(k = 1, \dots, n)$ such that $e_1 = R$, $e_n \leqslant n$ and such that*

$$\tfrac{1}{2}k(k-1) + e_k \geqslant R\,\text{OPT}_k \quad \textit{for } k = 1, \dots, n. \tag{7}$$

Then for our scheduling problem, there does not exist an on-line algorithm with competitive ratio strictly less than R.

Proof. Suppose for the sake of contradiction that there exists an $(R - \varepsilon)$-competitive on-line algorithm for some $\varepsilon > 0$.

First we prove by induction that this on-line algorithm must accept all jobs when fed with the instance I. This holds for $k = 1$, since $\text{OPT}_1 = 1$ whereas $e_1 \geqslant R$. Now consider the moment in time where the kth job arrives. So far, the on-line algorithm has accepted all jobs and thus has accumulated a total job completion time of $\frac{1}{2}k(k-1)$. Then by (7), it cannot reject the kth job without increasing the on-line cost to at least R times the off-line cost. Hence, also the kth job is accepted.

Now we get a contradiction: as the on-line algorithm accepts the nth job, its cost increases to $\frac{1}{2}n(n-1) + n \geqslant \frac{1}{2}n(n-1) + e_n \geqslant R\,\text{OPT}_n$. $\square$

It is quite easy to construct job sequences as described in Lemma 6. In Fig. 2, we list such a sequence for $R = 1.63$. The columns OPT_k and ON_k list the optimal objective value and the (forced) objective value of the on-line algorithm after k jobs. With the help of a computer program, we have constructed similar sequences for values of R up to approximately 1.6378411152. The idea is to choose every value e_k as small as possible so that it just satisfies the inequality in (7). This proves the negative result stated in Theorem 1.

4. Conclusions

We have presented a positive and a negative result for the on-line single machine problem. The main open problem is of course to close the gap between the lower bound 1.63784 and the upper bound 1.86602 in Theorem 1.

Another open problem concerns the corresponding on-line problem $Pm|p_j = 1|\text{Rej} + \sum C_j$ on $m \geqslant 2$ identical parallel machines. The single machine algorithm from Section 2 easily generalizes to $m \geqslant 2$ machines: when the on-line algorithm decides about a job J_j, it either may accept the job and schedule it with the smallest possible completion time C_j, or it may reject it at a certain penalty e_j. The generalization of our algorithm Greedy to m machines accepts the job if and only if $e_j > \alpha C_j$ holds; note that for $m = 1$ machines, this exactly yields to the algorithm described in Fig. 1. Along the lines of the proof in Section 2, we can show that the competitive ratio of this on-line algorithm for $Pm|p_j = 1|\text{Rej} + \sum C_j$ equals $\frac{1}{2}(2 + \sqrt{3}) \approx 1.86602$ for every $m \geqslant 2$. The analysis can be done more or less separately for every machine, and (up to somewhat messy calculations) the argument boils down to adding up m inequalities of the type proved in Section 2 for a single machine. We leave all details to the reader.

Also the lower bound construction in Section 3 may be adapted to $m \geqslant 2$ machines. However, as m becomes larger, the resulting lower bound values rapidly decline and tend towards 1. Numerical experiments seem to indicate that these lower bound values behave like $1 + \Theta(1/m)$ as m becomes large. Denote by T_m the on-line approximability threshold of problem $Pm|p_j = 1|\text{Rej} + \sum C_j$ (that is, for every $c > T_m$ there exists a c-competitive on-line algorithm, whereas for every $d < T_m$ there does not exist a d-competitive on-line algorithm). We know that $T_m \leqslant 1.86602$ for all m. It would be interesting to understand the behavior of these thresholds. Is it true that the numbers T_m form a decreasing sequence, and that hence the scheduling problem becomes easier as the number of machines increases?

References

[1] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. Sgall, L. Stougie, Multiprocessor scheduling with rejection, SIAM J. Discrete Math. 13 (2000) 64–78.

[2] D.W. Engels, D.R. Karger, S.G. Kolliopoulos, S. Sengupta, R.N. Uma, J. Wein, Techniques for scheduling with rejection, Proceedings of the Sixth European Symposium on Algorithms (ESA'1998), Lecture Notes in Computer Science, Vol. 1461, Springer, Berlin, 1998, pp. 490–501.

[3] H. Hoogeveen, M. Skutella, G.J. Woeginger, Preemptive scheduling with rejection, Proceedings of the Eighth European Symposium on Algorithms (ESA'2000), Lecture Notes in Computer Science, Vol. 1879, Springer, Berlin, 2000, pp. 268–277.

[4] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling: algorithms and complexity, in: S.C. Graves, A.H.G. Rinnooy Kan, P.H. Zipkin (Eds.), Logistics of Production and Inventory, Handbooks in Operations Research and Management Science, Vol. 4, North-Holland, Amsterdam, 1993, pp. 445–522.

[5] S.S. Seiden, Preemptive multiprocessor scheduling with rejection, Theoret. Comput. Sci. 262 (2001) 437–458.

[6] J. Sgall, On-line scheduling, in: A. Fiat, G.J. Woeginger (Eds.), On-line Algorithms: The State of the Art, Lecture Notes in Computer Science, Vol. 1442, Springer, Berlin, 1998, pp. 196–231.