Far East Journal of Applied Mathematics
Volume 62, Number 1, 2012, Pages 49-57

Published by Pushpa Publishing House, Allahabad, INDIA

THE ON-LINE PREEMPTIVE SCHEDULING ON PARALLEL MACHINES WHICH HAVE NONSIMULTANEOUS MACHINE AVAILABLE TIMES

Lin Wang and Zhenfu Yan

School of Management
Qufu Normal University
Rizhao, Shandong, 276826
P. R. China
e-mail: rzwanglin@163.com

School of Information Science and Engineering
Rizhao Polytechnic
Rizhao, Shandong, 276826
P. R. China
e-mail: zfyan@sina.com

Abstract

The on-line preemptive scheduling on parallel machines which have nonsimultaneous machine available times is firstly delivered in this paper. For the problem of minimizing the makespan, we show an algorithm which of the worst-case performance ratio is $x^m \Big/ \left[(x-1)\sum_{i=1}^{m} s_i x^{i-1}\right],$ where $x = \dfrac{m}{m-1}.$

2010 Mathematics Subject Classification: 90B35, 68M20.

Keywords and phrases: machine available time, on-line, preemption, worst-case performance ratio.

Received December 12, 2011

1. Introduction

We consider the problem of scheduling a list $(J_1, J_2, \cdots)$ of on-line jobs preemptively on m identical parallel machines. In such a setting, the jobs arrive one by one. Job J_j becomes known (with its existence and its processing time p_j) only when job J_{j-1} has already been scheduled, which gives rise to the name on-line scheduling. Job processing can be preemptive, i.e., the processing of any job can be interrupted and resumed later. The machines are parallel, which allows any machine to process any job, and differ only in their processing speeds. A job J_j of processing time p_j requires p_j/s_i time units for a machine of speed s_i to complete. As usual, it is required that each machine can process at most one job at a time and each job can be processed by at most one machine at a time.

In a classical parallel machine scheduling problem, we have m identical machines $M_1, M_2, ..., M_m,$ which are all available at time zero, i.e., $a_j = 0,\ j = 1, 2, ..., m.$ For this case, Chen et al. [3] derive an approximation algorithm with worst-case guarantee $m^m/(m^m - (m-1)^m)$ for every $m \geq 2$, which increasingly tends to $e/(e-1) \approx 1.58$ as $m \to \infty.$

If the m machines are not simultaneously available, i.e., $a_j \neq 0,\ j = 1, 2, ..., m,$ how to find a schedule that minimizes the maximum completion time, or *makespan*. In this paper, we will develop an approximation algorithm with worst-case ratio $x^m \Big/ \left[(x-1)\sum_{i=1}^{m} s_i x^{i-1}\right]$, where $x = \dfrac{m}{m-1}.$

2. The Approximation Algorithm

Let m denote the number of machines, let $s_i > 0$ be the speed of machine $M_i,\ i = 1, 2, ..., m.$ We assume that $s_1 \leq s_2 \leq \cdots \leq s_m.$ Without loss of generality, we assume that $s_m = 1.$ In this paper, we solve the case of non-decreasing speed ratios, i.e., $s_{i-1}/s_i \leq s_i/s_{i+1}$ for $2 \leq i \leq m-1.$

We use the following notations:

time t : the time immediately after the tth job has been scheduled,

L_i^t : the load of machine i after the arrival of t jobs,

$Q_i^t = L_i^t + a_i$,

OPT^t : the optimal off-line makespan at time t,

$S^t = \sum_{j=1}^{t} p_j + \sum_{i=1}^{m} a_i$,

$$r = \frac{x^m}{(x-1)\sum_{i=1}^{m} s_i x^{i-1}}.$$

Each job i $(i = 1, 2, \ldots, n)$ is associated with a processing time p_i, if they become available for processing at time zero and can be interrupted, each machine has its own speed s_i and preparation time $a_i \geq 0$, then the completion time of the machine has three obviously lower bound: $\max_{1\leq i\leq m} a_i$, $\min_{1\leq i\leq m} a_i + \max_{1\leq k\leq m-1} \frac{P_k}{S_k}$, $\left(\sum_{j=1}^{n} p_j + \sum_{i=1}^{m} a_i\right) / \sum_{i=1}^{m} s_i$, where P_k means the sum of the k longest processing times, and S_k means the sum of the k fastest machines' speeds, then we have

$$LB = \max\left\{ \max_{1\leq i\leq m} a_i,\ \min_{1\leq i\leq m} a_i + \max_{1\leq k\leq m-1} \frac{P_k}{S_k},\ \frac{\sum_{j=1}^{n} p_j + \sum_{i=1}^{m} a_i}{\sum_{i=1}^{m} s_i} \right\}. \quad (2.1)$$

The algorithm maintains the following three invariants. These invariants are a generalization of the invariants defined for identical machines in [7],

(a) At any time t, $Q_1^t \leq Q_2^t \leq \cdots \leq Q_m^t$.

(b) At any time t, $Q_m^t \leq rLB^t$.

(c) At any time *t*, for every $1 \le k \le m$, $\sum_{i=1}^{k} s_i Q_i^t \le \dfrac{\sum_{i=1}^{k} s_i x^{i-1}}{\sum_{i=1}^{m} s_i x^{i-1}} S^t$.

A new job J_{t+1} (which arrives at time $t+1$) is assigned as follows.

First calculate LB^{t+1} by equality (2.1), then the following intervals are reserved. On machine M_m, the interval: $I_m = [Q_m^t, rLB^{t+1}]$; and on machine M_i $(1 \le i \le m-1)$, the interval: $I_i = [Q_i^t, Q_{i+1}^t]$.

To assign J_{t+1}, go from I_m to I_1, putting a part of the job, as large as possible in each interval, until all the job is assigned. After the assignment there will be some fully occupied intervals $I_{l+1}, \ldots, I_m$ some empty intervals $I_1, \ldots, I_{l-1}$ and a partially or fully occupied interval I_l.

Next, we show that it is always possible to partition J_{t+1} among those intervals.

Lemma 2.1. *If the invariants are fulfilled at step t, then the reserved intervals are sufficient to assign* J_{t+1}.

Proof. The total weight that can be assigned to all intervals is

$$A = (rLB^{t+1} - Q_m^t)s_m + \sum_{i=1}^{m-1}(Q_{i+1} - Q_i^t)s_i$$

$$= rLB^{t+1} + \sum_{i=1}^{m}(s_{i-1} - s_i)Q_i^t$$

$$= rLB^{t+1} + \sum_{i=1}^{m}\left(\frac{s_{i-1}}{s_i} - \frac{s_i}{s_{i+1}}\right)\sum_{j=1}^{i} s_j Q_j^t.$$

Since $s_{i-1}/s_i \le s_i/s_{i+1}$, we can use the third invariant for each value of *j* and get that the above is at least

$$A \geq rLB^{t+1} + \frac{S^t}{\sum_{i=1}^{m}}\left(\frac{s_{i-1}}{s_i} - \frac{s_i}{s_{i+1}}\right)\sum_{j=1}^{i} s_j x^{j-1}$$

$$= \frac{1}{\sum_{i=1}^{m} s_i x^{i-1}}\left[\frac{x^m}{x-1} LB^{t+1} + S^t\left(\frac{s_{i-1}}{s_i} - \frac{s_i}{s_{i+1}}\right)\sum_{j=1}^{i} s_j x^{j-1}\right]$$

$$= \frac{1}{\sum_{i=1}^{m} s_i x^{i-1}}\left[\frac{x^m}{x-1} LB^{t+1} + \left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m\right)S^t\right].$$

We consider two cases:

1. $LB^{t+1} \geq p_{t+1} \geq \dfrac{S^{t+1}}{\sum_{i=1}^{m} s_i}$

2. $LB^{t+1} \geq \dfrac{S^{t+1}}{\sum_{i=1}^{m} s_i} \geq p_{t+1}.$

We show that the assignment is successful in both cases.

Case 1. Since $S^t = S^{t+1} - p_{t+1} \leq p_{t+1}\left(\sum_{i=1}^{m} s_i - 1\right).$

So,

$$A \geq \frac{1}{\sum_{i=1}^{m} s_i x^{i-1}}\left[\frac{x^m}{x-1} p_{t+1} + p_{t+1}\left(\sum_{i=1}^{m} s_i - 1\right)\left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m\right)\right]$$

$$= \frac{p_{t+1}}{\sum_{i=1}^{m} s_i x^{i-1}}\left[\frac{x^m}{x-1} + \left(\sum_{i=1}^{m} s_i - 1\right)\left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m\right)\right]$$

$$= \frac{p_{t+1}}{\sum_{i=1}^{m} s_i x^{i-1}}\left[\frac{x^m}{x-1} + \sum_{i=1}^{m} s_i x^{i-1} - \frac{x^m}{x-1}\right]$$

$$= p_{t+1}.$$

Case 2. In this case,

$$LB^{t+1} \geq p_{t+1} + \frac{S^t}{\sum_{i=1}^{m} s_i},$$

and,

$$S^t = S^{t+1} - p_{t+1} \geq p_{t+1}\left(\sum_{i=1}^{m} s_i - 1\right),$$

so,

$$\begin{aligned}
A &\geq \frac{1}{\sum_{i=1}^{m} s_i x^{i-1}} \left\{ \frac{x^m}{x-1} p_{t+1} + \frac{x^m}{(x-1)\sum_{i=1}^{m} s_i} S^t \right. \\
&\qquad \left. + \left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m \right) S^t \right\} \\
&\geq \frac{p_{t+1}}{\sum_{i=1}^{m} s_i x^{i-1}} \left\{ \frac{1}{\sum_{i=1}^{m} s_i} \left(\frac{x^m}{x-1} + \frac{x^m}{x-1}\left(\sum_{i=1}^{m} s_i - 1 \right) \right) \right. \\
&\qquad \left. + \left(\sum_{i=1}^{m} s_i - 1 \right)\left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m \right) \right\}. \\
&\geq \frac{p_{t+1}}{\sum_{i=1}^{m} s_i x^{i-1}} \left\{ \frac{x^m}{x-1} + \frac{1}{x-1}\left((x-1)\sum_{i=1}^{m} s_i x^{i-1} - x^m \right) \right\} \\
&\geq \frac{p_{t+1}}{\sum_{i=1}^{m} s_i x^{i-1}} \left\{ \frac{x^m}{x-1} + \sum_{i=1}^{m} s_i x^{i-1} - \frac{x^m}{x-1} \right\} \\
&= p_{t+1}.
\end{aligned}$$

To complete the proof of the algorithm, we need to show that all invariants are kept after an assignment of a job. This is clear for the first two invariants, from the definition of the algorithm.

Lemma 2.2 *If the invariants are fulfilled after step t, then they are also satisfied after step $t+1$.*

Proof. According to the definition of the algorithm, there exists a machine l such that for $i < l$, $Q_i^{t+1} = Q_i^t$, for $l < i \le m$, $Q_i^{t+1} = Q_{i+1}^t$, and $Q_l^t < Q_l^{t+1} \le Q_{l+1}^t$ (for convenience let $Q_{m+1}^t = rLB^{t+1}$).

If $k < l$, then

$$\sum_{i=1}^{k} s_i Q_i^{t+1} = \sum_{i=1}^{k} s_i Q_i^t \le \frac{\sum_{i=1}^{k} s_i x^{i-1}}{\sum_{i=1}^{m} s_i x^{i-1}} S^t$$

$$\le \frac{\sum_{i=1}^{k} s_i x^{i-1}}{\sum_{i=1}^{m} s_i x^{i-1}} S^{t+1}.$$

If $l \le k \le m$, then we need to show

$$\sum_{i=k+1}^{m} s_i Q_i^{t+1} \ge \frac{\sum_{i=k+1}^{m} s_i x^{i-1}}{\sum_{i=1}^{m} s_i x^{i-1}} S^{t+1}. \tag{2.2}$$

Since $l < i \le m$, $Q_i^{t+1} = Q_{i+1}^t$, so

$$\sum_{i=k+1}^{m} s_i Q_i^{t+1}$$

$$= rLB^{t+1} + \sum_{i=k+2}^{m} s_{i-1} Q_i^t$$

$$= rLB^{t+1} + \sum_{i=k+2}^{m} s_i Q_i^t \left(\frac{s_{k+1}}{s_{k+2}}\right) + \sum_{i=k+3}^{m} \left(\frac{s_{i-1}}{s_i} - \frac{s_{k+1}}{s_{k+2}}\right) s_i Q_i^t$$

$$= rLB^{t+1} + \sum_{i=k+2}^{m} s_i Q_i^t \left(\frac{s_{k+1}}{s_{k+2}}\right) + \sum_{j=k+3}^{m} \left(\sum_{i=j}^{m} s_i Q_i^t\right)\left(\frac{s_{j-1}}{s_j} - \frac{s_{j-2}}{s_{j-1}}\right)$$

$$\geq \frac{1}{\sum_{i=1}^{m} s_i x^{i-1}} \left[\frac{x^m}{x-1} LB^{t+1} + \left(\sum_{i=k+1}^{m-1} s_i x^i\right) S^t\right].$$

Let $p_{t+1} = \mu S^{t+1}$. Then $LB^{t+1} \geq \max\left\{\mu, 1/\sum_{i=1}^{m} s_i\right\} S^{t+1}$ and $S^t = (1-\mu) S^{t+1}$. Simple calculations show that inequality (2.2) holds.

3. Conclusion

We have given an approximation algorithm for the on-line preemptive scheduling on parallel machines which have nonsimultaneous machine available times, its worst-case performance ratio is

$$\frac{A(I)}{OPT(I)} \leq \frac{A(I)}{LB} \leq \frac{x^m}{(x-1)\sum_{i=1}^{m} s_i x^{i-1}}.$$

If $s_i = 1$, then its worst-case performance ratio is $\dfrac{m^m}{m^m - (m-1)^m}$.

References

[1] C. Y. Lee, Parallel machine scheduling with nonsimultaneous machine available time, Discrete Appl. Math. 30 (1991), 53-61.

[2] Jianjun Wen and Donglei Du, Preemptive on-line scheduling for two uniform processors, Oper. Res. Lett. 23 (1998), 113-116.

[3] B. Chen, A. van Vliet and G. J. Woeginger, An optimal algorithm for preemptive on-line scheduling, Oper. Res. Lett. 18 (1995), 127-131.

[4] Yuzhong Zhang, Shouyang Wang, Bo Chen and Shuxia Zhang, On-line preemptive scheduling on uniform machines, J. Syst. Sci. Complex. 14 (2001), 373-377.

[5] Guo-Hui Lin, En-Yu Yao and Yong He, Parallel machine scheduling to maximize the minimum load with nonsimultaneous machine available times, Oper. Res. Lett. 22 (1998), 75-81.

[6] Leah Epstein and Jiří Sgall, A lower bound for on-line scheduling on uniformly related machines, Oper. Res. Lett. 26 (2000), 17-22.

[7] Leah Epstein, Optimal preemptive on-line scheduling on uniform processors with non-decreasing speed ratios, Oper. Res. Lett. 29 (2001), 93-98.