Semi-online Machine Covering on Two Hierarchical Machines with Discrete Processing Times

Gangxiong Wu[1](✉) and Weidong Li[1,2](✉)

[1] School of Mathematics and Statistics, Yunnan University, Kunming 650504, China
1317630396@qq.com, weidongmath@126.com
[2] Dianchi College of Yunnan University, Kunming 650504, China

Abstract. In this paper, we study the semi-online machine covering problem on two hierarchical machines, whose objective is to maximize the minimum machine load. When the processing times are discrete by $\{1, 2, 2^2, \ldots, 2^k\}$ with $k \geq 2$, we prove that no algorithm can have a competitive ratio less than 2^k and present an optimal semi-online algorithm with competitive ratio 2^k.

Keywords: Machine covering · Semi-online · Competitive ratio
Hierarchy · Discrete processing times

1 Introduction

Given m hierarchical machines and n jobs, each job can only be processed on a subset of the machines and each job can only be processed on a machines. The hierarchical scheduling problem, denoted by $P|GoS|C_{max}$, is to minimize the maximum load of all machines (makespan). Hwang et al. [2] studied the offline problem $P|GoS|C_{max}$ and designed an approximation algorithm with the mankspan no more than $\frac{5}{4}$-times the optimum for $m = 2$, and no more than $2 - \frac{1}{m-1}$-times the optimum for $m \geq 3$. Ou et al. [7] designed a 4/3-approximation algorithm and a polynomial time approximation scheme (PTAS, for short) for $P|GoS|C_{max}$. Li et al. [6] designed an efficient PTAS with running time $O(nlogn)$ for a special case of the problem $P|GoS|C_{max}$ and present a simple fully polynomial time approximation scheme (FPTAS, for short) with running time $O(n)$ for the problem $P_m|GoS|C_{max}$, where m is a constant. For the online version, Park et al. [8] and Jiang et al. [3] designed an optimal online algorithm with a competitive ratio of $\frac{5}{3}$ for the case of two machines, respectively. Wu et al. [10] designed several optimal semi-online scheduling algorithm on two hierarchical machines. Zhang et al. [11] designed some optimal online algorithms on two hierarchical machines with tightly-grouped processing times.

Machine covering on hierarchical machines with the objective of maximizing the minimum machine load, denoted by $P|GoS|C_{min}$, is not a well-studied scheduling problem. Li et al. [4] presented a PTAS for $P|GoS|C_{min}$. Wu et al.

L. Li et al. (Eds.): NCTCS 2018, CCIS 882, pp. 1–7, 2018.
https://doi.org/10.1007/978-981-13-2712-4_1

[9] designed two semi-online optimal algorithms for $P|GoS|C_{min}$ on two hierarchical machines, when both the processing time and the class of the largest job are known. Luo et al. [5] presented an optimal online algorithm with a competitive ratio of $(1+\alpha)$ for $P|GoS|C_{min}$ on two hierarchical machines, when the processing time of each job is bounded by an interval $[1,\alpha]$. Chassid and Epstein [1] considered the machine covering problem on two hierarchical machines of possibly different speeds.

In this paper, we consider the online machine covering problem on two hierarchical machines with discrete processing times. The processing time of all jobs are discrete by $\{1, 2, 2^2, \ldots, 2^k\}$, where $k \geq 2$. We prove that no algorithm can have a competitive ratio less than 2^k and give an optimal algorithm with the competitive ratio of 2^k. The paper is organized as follows. Section 2 gives some basic definitions. Section 3 presents an optimal semi-online algorithm. Section 4 presents concluding remarks.

2 Preliminaries

We are given two machines and a series of jobs arriving online which are to be scheduled irrevocably at the time of their arrivals. The first machine can process all the jobs while the second one can process only part of the jobs. The arrival of a new job occurs only after the current job is scheduled. Let $J = \{J_1, J_2, \ldots, J_n\}$ be the set of all jobs arranged in the order of arrival. We denote each job as J_i with p_i and g_i, where $p_i > 0$ is the processing time (also called job size) of the job J_i and $g_i \in \{1, 2\}$ is the hierarchy of the job J_i. If $g_i = 1$, the job J_i must be processed by the first machine, and if $g_i = 2$, the job J_i can be processed by either of the two machines. p_i and g_i are not known until the arrival of the job J_i.

The schedule can be seen as the partition of J into two subsets, we denote as $<S_1, S_2>$, where S_1 and S_2 contain job indices assigned to the first and the second machine, respectively. Let $p(S_1) = \sum_{J_i \in S_1} p_i$ and $p(S_2) = \sum_{J_i \in S_2} p_i$ denote the load of the first machine and the second machine, respectively.

For the first i jobs, we define that T^i denote total processing time, TG_1^i is total processing time the jobs with hierarchy 1; p_{max}^i is the largest job time; $p(S_1^i)$ denote total processing time of the jobs scheduled on M_1 after the job J_i is scheduled; $p(S_2^i)$ is total processing time of the jobs scheduled on M_2 after the job J_i is scheduled; $V_i(opt)$ denote the optimal minimum machine load after scheduling the job J_i; V_{opt} is the optimal function value of the problem in an offline version; V_{out} denote the output objective function value by a algorithm.

So, according to the define of above, we have $S_1 = S_1^n$ and $S_2 = S_2^n$. The minimum value of $p(S_1)$ and $p(S_2)$, i.e., $min\{p(S_1), p(S_2)\}$, is defined as the minimum machine load of the schedule $<S_1, S_2>$. The objective is to find a schedule $<S_1, S_2>$ that maximizes the minimum machine load.

For the first i jobs, let $L^i = min\{T^i - TG_1^i, \frac{T^i}{2}, T^i - p_{max}^i\}$ and L^i is a standard upper bound of the optimal minimum machine load. Then we can get following lemma.

Lemma 1. *The optimal minimum machine load is at most L^i after scheduling the job J_i.*

Definition 1. *For a job sequence J and an algorithm, then the competitive ratio of algorithm is defined as the smallest η such that for any J, $V_{opt} \leq \eta V_{out}$.*

At first, we give a lower bounded for the problem.

Theorem 1. *There exists no algorithm with a competitive ratio less than 2^k.*

Proof: Consider an algorithm B and the following job sequence. The first job J_1 with $p_1 = 1$ and $g_1 = 2$. If algorithm B schedules J_1 on M_1, we further generate the last job J_2 with $p_2 = 2^k$ and $g_2 = 1$ must be scheduled on M_1. Therefore, we have $V_{opt} = 1$ and $V_{out} = 0$, which lead to the competitive ratio is unbounded.

Otherwise, if algorithm B schedules J_1 with $p_1 = 1$ and $g_1 = 2$ on M_2. The job J_2 with $p_2 = 2^k$ and $g_2 = 2$, the algorithm B must schedule J_2 with $p_2 = 2^k$ and $g_2 = 2$ on M_1. If the algorithm B schedule J_2 with $p_2 = 2^k$ and $g_2 = 2$ on M_2, we have $V_{opt} = 1$ and $V_{out} = 0$, which lead to the competitive ratio is unbounded. The job J_3 with $p_3 = 2^k$ and $g_3 = 1$ must be scheduled on M_1. We have $V_{opt} = 2^k$ and $V_{out} = 1$. Hence, there exists no algorithm with a competitive ratio less than 2^k.

3 An Optimal Semi-online Algorithm

In the section, we consider that the hierarchical load balancing problem on two machines with discrete processing times. All processing times belong to $\{1, 2, 2^2, \ldots, 2^k\}$, where $k \geq 2$ in this problem. We present an optimal algorithm.

Algorithm A

Input : $J_i = (p_i, g_i)$
Output : $< S_1, S_2 >$;
Step 0: $S_1^0 = \emptyset$, $S_2^0 = \emptyset$, $i = 1$;
Step 1: On receiving $J_i = (p_i, g_i)$, update T^i, TG_1^i, p_{max}^i and L^i;
Step 2: If $g_i = 1$, schedule J_i on M_1. Go to **Step 4**;
Step 3: If $g_i = 2$ and when $p(S_1^{i-1}) < \frac{1}{2^k} L^i$, schedule J_i on M_1.
Else schedule it on M_2. Go to **Step 4**;
Step 4: If there is a new job, let $i = i + 1$ and go to **Step 1**.
Else, output S_1 and S_2.

For the problem and the algorithm, we define that $V_{out} = min\{p(S_1^n), p(S_2^n)\}$ is the output of the Algorithm A and V_{opt} is the output of the optimal offline algorithm.

Lemma 2. *If Algorithm A schedule the job J_i with $g_i = 2$ on M_1, then $L^i \neq T^i - TG_1^i$.*

Proof: According to Algorithm A, if the job J_i with $g_i = 2$ is scheduled on M_1, we have $p(S_1^{i-1}) < \frac{1}{2^k} L^i$.

If $L^i = min\{T^i - TG_1^i, \frac{T^i}{2}, T^i - p_{max}^i\} = T^i - TG_1^i$, then we get $T^i - TG_1^i \leq \frac{T^i}{2}$, which implies $TG_1^i \geq \frac{T^i}{2}$. Since $p(S_1^{i-1}) \geq TG_1^{i-1} = TG_1^i$, then we have

$$p(S_1^{i-1}) \geq \frac{T^i}{2} \geq L^i > \frac{L^i}{2^k}$$

and it is contradictory with $p(S_1^{i-1}) < \frac{L^i}{2^k}$. Thus, the proof is complete.

Lemma 3. *If Algorithm A schedule the job J_i with $g_i = 2$ on M_1 and $L^i = T^i - p_{max}^i$, then $p(S_2^i) \geq \frac{1}{2^k}(T^i - TG_1^i)$.*

Proof: Since $L^i = T^i - p_{max}^i$, according to the definition of L^i, we get $T^i - p_{max}^i \leq \frac{T^i}{2}$, which means

$$p_{max}^i \geq \frac{T^i}{2}.$$

In the first i jobs, we denote job J_j where $j \in \{1, 2, 3 \cdots i\}$ has largest processing time, i.e., $p_j = p_{max}^i$. Now, we will discuss two cases:

Case 1. $p_{max}^i \neq p_i$.

If J_j belongs to S_1^{i-1}, we have

$$p(S_1^{i-1}) \geq p_{max}^i \geq \frac{T^i}{2} > \frac{1}{2^k} L^i$$

and this is contradictory with that Algorithm A schedule the job J_i on M_1.

If J_j belongs to S_2^i, we have

$$p(S_2^i) \geq p_{max}^i \geq \frac{T^i}{2} \geq \frac{1}{2^k} T^i \geq \frac{1}{2^k}(T^i - TG_1^i). \tag{1}$$

Case 2. $p_{max}^i = p_i$.

We have

$$T^i - p_{max}^i = p(S_2^i) + p(S_1^{i-1}). \tag{2}$$

Since Algorithm A schedule the job J_i on M_1, we have

$$p(S_1^{i-1}) < \frac{1}{2^k} L^i = \frac{1}{2^k}(T^i - p_{max}^i). \tag{3}$$

Hence, according to the inequalities of (2), (3), we have

$$p(S_2^i) = T^i - p_{max}^i - p(S_1^{i-1}) > T^i - p_{max}^i - \frac{1}{2^k}(T^i - p_{max}^i) = \frac{2^k - 1}{2^k}(T^i - p_{max}^i).$$

Since $k \geq 2$ and according to the inequalities of (3), we have

$$p(S_2^i) > \frac{2^k - 1}{2^k}(T^i - p_{max}^i) > (2^k - 1)p(S_1^{i-1}) \geq 3p(S_1^{i-1}). \tag{4}$$

Since $p(S_1^{i-1}) \geq 1$, we have $p(S_2^i) > 3$. Since $p_i = p_{max}^i \geq \frac{T^i}{2} > p(S_1^{i-1})$ and $p_i \leq 2^k$, we have

$$\frac{T^i - TG_1^i}{p(S_2^i)} \leq \frac{T^i}{p(S_2^i)} = 1 + \frac{p(S_1^{i-1}) + p_i}{p(S_2^i)} < 1 + \frac{2^{k+1}}{3} < 2^k. \tag{5}$$

The proof is complete.

Lemma 4. *If Algorithm A schedule the job J_i with $g_i = 2$ on M_1 and $L^i = \frac{T^i}{2}$, then $p(S_2^i) \geq \frac{T^i - TG_1^i}{2^k}$.*

Proof: Since the job J_i with $g_i = 2$ is scheduled on M_1, we have $p(S_1^{i-1}) < \frac{L^i}{2^k} = \frac{T^i}{2 \times 2^k}$. Since

$$L^i = min\{T^i - TG_1^i, \frac{T^i}{2}, T^i - p_{max}^i\} = \frac{T^i}{2}.$$

So, we have $T^i - p_{max}^i \geq \frac{T^i}{2}$ holds, which implies $p_{max}^i \leq \frac{T^i}{2}$. Then, we have $p_i \leq p_{max}^i \leq \frac{T^i}{2}$ and

$$p(S_2^i) = T^i - p(S_1^{i-1}) - p_i > T^i - \frac{T^i}{2 \times 2^k} - \frac{T^i}{2} = \frac{2^k - 1}{2 \times 2^k} T^i.$$

Since $k \geq 2$ and $T^i \geq T^i - TG_1^i$, we have

$$p(S_2^i) > \frac{2^k - 1}{2 \times 2^k} T^i > \frac{T^i}{2^k} \geq \frac{T^i - TG_1^i}{2^k}.$$

We complete the proof.

Theorem 2. *If $V_{out} = min\{p(S_1^n), p(S_2^n)\} = p(S_1^n)$, then $\frac{V_{opt}}{V_{out}} \leq 2^k$.*

Proof: According to the question, we know that $S_2^n \neq \emptyset$.

We assume that the job J_i is the last job that scheduled on M_2. According to Algorithm A, we have $p(S_1^{i-1}) \geq \frac{1}{2^k} L^i$. Since

$$L^n - L^i \leq T^n - T^i \tag{6}$$

and all the jobs arrived after the job J_i will be scheduled on M_1.

According to the definition of L^n and $L^n \geq V_{opt}$, we have

$$\begin{aligned} \frac{1}{2^k} L^i + (L^n - L^i) &\geq \frac{1}{2^k}(L^i - L^n) + \frac{L^n}{2^k} + (L^n - L^i) \\ &\geq (1 - \frac{1}{2^k})(L^n - L^i) + \frac{L^n}{2^k} \\ &\geq 0. \end{aligned} \tag{7}$$

Then, according to inequality (7), we have

$$\frac{1}{2^k} L^i + (L^n - L^i) \geq \frac{1}{2^k} L^n. \tag{8}$$

So, according to the inequalities of (6), (8), we have

$$p(S_1^n) = p(S_1^{i-1}) + (T^n - T^i) \geq \frac{1}{2^k}L^i + (L^n - L^i) \geq \frac{1}{2^k}L^n \geq \frac{1}{2^k}V_{opt}. \quad (9)$$

Thus, according to inequality (9), when $V_{out} = min\{p(S_1^n), p(S_2^n)\} = p(S_1^n)$, we have

$$\frac{V_{opt}}{V_{out}} \leq 2^k.$$

We complete the proof.

Theorem 3. *The competitive ratio of Algorithm A is* 2^k.

Proof: According to Theorem 2, if $V_{out} = min\{p(S_1^n), p(S_2^n)\} = p(S_1^n)$, then

$$\frac{V_{opt}}{V_{out}} \leq 2^k. \quad (10)$$

Therefore, we only need to prove when $V_{out} = min\{p(S_1^n), p(S_2^n)\} = p(S_2^n)$, the inequality (10) holds.

We discuss the following two cases:

Case 1. Algorithm A doesn't schedule jobs with hierarchy 2 on M_1.

In this case, according to the definition of L^n. We have

$$p(S_2^n) = T^n - TG_1^n \geq L^n \geq V_{opt} > \frac{1}{2^k}V_{opt}. \quad (11)$$

Case 2. At least one job with hierarchy 2 is scheduled on M_1.

Let J_a denote the last job with $g_a = 2$ that scheduled on M_1. According to Lemmas 2, 3 and 4, we have

$$p(S_2^a) \geq \frac{1}{2^k}(T^a - TG_1^a).$$

Since remaining the jobs with hierarchy 2 are scheduled on M_2 after job J_a and we have $k \geq 2$ and

$$T^n - TG_1^n \geq T^a - TG_1^a,$$

then we get

$$\begin{aligned}
p(S_2^n) &= p(S_2^a) + ((T^n - TG_1^n) - (T^a - TG_1^a)) \\
&\geq \frac{1}{2^k}(T^a - TG_1^a) + ((T^n - TG_1^n) - (T^a - TG_1^a)) \\
&= \frac{1 - 2^k}{2^k}(T^a - TG_1^a) + (T^n - TG_1^n) \\
&\geq \frac{1 - 2^k}{2^k}(T^n - TG_1^n) + (T^n - TG_1^n) \quad (12) \\
&= \frac{1}{2^k}(T^n - TG_1^n) \\
&\geq \frac{1}{2^k}L^n \\
&\geq \frac{1}{2^k}V_{opt}.
\end{aligned}$$

According to the definition of V_{out} and the inequalities of (11), (12). We have the inequality (10) holds.

According to Theorem 1, the optimal competitive ratio of Algorithm A is 2^k. We complete the proof of competitive ratio.

4 Conclusion

In the paper, we study the semi-online version of hierarchical scheduling problem on two parallel machines with the objective of maximizing the minimum machine load. If the processing times are discrete by $\{1, 2, 2^2, \ldots, 2^k\}$, where $k \geq 2$. We prove the lower bound of the competitive ratio of any online algorithm is 2^k and present an algorithm which is shown to be optimal.

Acknowledgement. The work is supported in part by the National Natural Science Foundation of China [Nos. 11761078, 61662088], the Natural Science Foundation of Education Department of Yunnan Province [No. 2017ZZX235], IRTSTYN and Program for Excellent Young Talents, Yunnan University.

References

1. Chassid, O., Epstein, L.: The hierarchical model for load balancing on two machines. J. Comb. Optim. **15**(4), 305–314 (2008)
2. Hwang, H.C., Chang, S.Y., Lee, K.: Parallel machine scheduling under a grade of service provision. Comput. Oper. Res. **31**(12), 2055–2061 (2004)
3. Jiang, Y., He, Y., Tang, C.: Optimal online algorithms for scheduling two identical machines under a grade of service. J. Zhejiang Univ. Sci. A. **7**(3), 309–314 (2006)
4. Li, J., Li, W., Li, J.: Polynomial approximation schemes for the max-min allocation problem under a grade of service provision. Discret. Math. Algorithms Appl. **1**(3), 355–368 (2009)
5. Luo, T., Xu, Y.: Semi-online hierarchical load balancing problem with bounded processing times. Theor. Comput. Sci. **607**, 75–82 (2015)
6. Li, W., Li, J., Zhang, T.: Two approximation schemes for scheduling on parallel machines under a grade of service provision. Asia-Pac. J. Oper. Res. **29**(5), Article No. 1250029 (2012)
7. Ou, J., Leung, J.Y.T., Li, C.L.: Scheduling parallel machines with inclusive processing set restrictions. Nav. Res. Logist. **55**(4), 328–338 (2008)
8. Park, J., Chang, S.Y., Lee, K.: Online and semi-online scheduling of two machines under a grade of service provision. Oper. Res. Lett. **34**(6), 692–696 (2006)
9. Wu, Y., Cheng, T.C.E., Ji, M.: Optimal algorithm for semi-online machine covering on two hierarchical machines. Theor. Comput. Sci. **531**, 37–46 (2014)
10. Wu, Y., Ji, M., Yang, Q.: Optimal semi-online scheduling algorithm on two parallel identical machines under a grade of service provision. Int. J. Prod. Econ. **135**(1), 367–371 (2012)
11. Zhang, A., Jiang, Y., Fan, L., Hu, J.: Optimal online algorithms on two hierarchical machines with tightly-grouped processing times. J. Comb. Optim. **29**(4), 781–795 (2015)