2018 IEEE International Conference on Data Mining Workshops (ICDMW)

Online and Semi-online Vector Scheduling on A Single Machine with Rejection

Qianna Cui
College of Software
Harbin Engineering University
Harbin, PR China
cuiqianna@126.com

Haiwei Pan
College of Software
Harbin Engineering University
Harbin, PR China
Email: panhaiwei@hrbeu.edu.cn

***Abstract*—In this paper, we design an online algorithm for vector scheduling on a single machine with rejection and its competitive ratio is d, where d is the dimensions of vector. In addition, we consider two versions of semi-online vector scheduling on a single machine with rejection. In the first version, semi-online with rearrangement allows at most one job to be reassigned after scheduling all jobs, then we show a semi-online algorithm with competitive ratio $\frac{1}{2}d+2$ for $d>3$. The second version is semi-online with rejection buffer whose length$=1$, which can hold one job. When $d>3$, we also give an algorithm with competitive ratio $\frac{1}{2}d+2$.**

***Keywords*-vector scheduling; rejection; online; semi-online;**

I. Introduction

Due to computer contains storage, CPU, hard disk and other resources, multiple resource occupancy problem has been widely concerned. When a computer processes tasks, each task takes up a variety of computer resources, how to maximize the machine's resource utilization is worth studying. Li and Cui [1] raised vector scheduling with rejection on a single machine, which is describes as follows. Given a single machine and a sequence of n jobs $J=\{J_1,J_2,\ldots,J_n\}$, job J_j is characterized by $d-$dimensional resource vector $\mathbf{p}_j=\{p_j^1,p_j^2,\ldots,p_j^d\}$ and rejection cost w_j, where $p_j^k(k=1,\ldots,d)$ is the k_{th} resource load of job J_j. Job J_j can be either accepted and then assigned on the machine, or rejected and then produce corresponding rejection cost. No preemption is allowed. The goal is to find a suitable solution to minimize the sum of the maximum resource load and the total of all rejection cost. Let A and R be the set of accepted jobs and rejected jobs respectively, the objective function is $\min Z=\min(\max_k\sum_{J_j\in A}p_j^k+\sum_{J_j\in R}w_j)$. We proved the problem is NP-hard in paper [1] and proposed a $d-$approximation algorithm and a randomized algorithm with 1.585-approximation. When d is a constant, we designed an FPTAS based on the dynamic programming. If all tasks come to be processed one by one, i.e. each task with information comes until scheduling its previous task online scheduling is proposed, we study online and semi-online vector scheduling problems on a single machine with rejection.

If resource load and rejection of job J_j is unknown until the job J_{j-1} is scheduled, the online vector scheduling with rejection is valued to study. When job J_j arrives, we can either accept it and then schedule it on the machine or reject it by paying penalty. If reassignment after all jobs have been assigned or rejection buffer is allowed, we address two versions of semi-online vector scheduling with rejection. Online algorithm and semi-online algorithm are proposed for the two problems respectively. The comparative ratio [2], [3] is to measure the online and semi-online algorithm.

Scheduling with rejection was studied by Bartal $et\ al.$ [4], they showed a 2.618-competitive algorithm and a 1.618-competitive algorithm with $m=2$ for online scheduling with rejection. After that, vector scheduling got into the research area. Zhu $et\ al.$ [5] considered online vector scheduling by generalized load balancing based on the vector scheduling [6], the objective function is not the infinite norm of vector but the $L_{\ln(md)}-$ norm, they gave an $e\log(md)$ approximation algorithm. When the objective function is L_r- norm of vector, Im $et\ al.$ [7] addressed an optimal online algorithm with $\Theta(\log d/\log\log d)-$competitive for identical machines and $\Theta\log m+\log d-$competitive online algorithm for unrelated machines. A randomized algorithm for online vector scheduling problem was proposed by [8], whose competitive ratio is $\Omega(\frac{\log d}{\log\log d})$.

Many researchers paid much attention on two semi-online versions with reassignment or buffer in recently years. Semi-online with buffer and known total size of all jobs for parallel machine scheduling [9] could be solved by an optimal algorithm with competitive ratio $\frac{4}{5}$. Tan and Yu [10] studied online scheduling with reassignment on two identical machines, they showed that three lower bound are respectively $\frac{3}{2}$, $\sqrt{2}$, $\frac{4}{3}$ for reassigning the last k jobs, reassigning the last job of each machine, reassigning arbitrary k jobs. Semi-online for extension of scheduling problems were focused. Min $et\ al.$ [3] considered two semi-online scheduling problems on two identical machines with rejection, and they put forward two $\frac{3}{2}-$competitive algorithms for reassignment version and buffer version. Chen $et\ al.$ [11] considered two versions of semi-online hierarchical scheduling on two machines, in which each job has hierarchy $g=1$ or $g=2$, job with $g=1$ can be scheduled on arbitrary machine but the other must be assigned on machine two. They proposed two optimal algorithms for both versions of

2375-9259/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDMW.2018.00141

semi-online with competitive ratio $\frac{3}{2}$. Qi and Yuan studied in paper [2] similar to paper [11], what different was that the objective function replaced minimizing makespan by minimizing l_p-norm of two machines' load. They designed a unified optimal algorithm for both versions of semi-online.

In second section, we give an online algorithm for vector scheduling on a single machine with rejection(for short: VSR), whose competitive ratio is d. Two versions of semi-online vector scheduling with rejection are studied in section three, we give two algorithms for semi-online with rearrangement or buffer with competitive ratio $\frac{1}{2}d+2$. The last section conclude our results and discusses the useful extensions to the proposed problem and lists ideas for future work.

II. Online algorithm

We will gain the characterized information of current job until the job arrives after the prior job is scheduled. In paper [1], we used greedy method to obtain approximative solution for off-line VSR. Then, we use similar method to solve online VSR in this section, for job J_j, let $p_j=\sum_{k=1}^{d}p_j^k$ be synthetical resource demand. When job J_j arrives, if $w_j\le p_j$, then reject this job, otherwise, accept it. The above algorithm called H is described as follows.

Algorithm H

Step 1. For job J_j, when job $J_j=(\mathbf{p_j},w_j)$ arrives, where $\mathbf{p_j}=\{p_j^1,\ldots,p_j^d\}$, if $w_j\le\sum_{k=1}^{d}p_j^k$, then reject it.

Step 2. Else, accept and schedule it on the single machine.

Step 3. Until $j=n$, stop.

The following result can be gained with algorithm H.

Theorem 1 *Online algorithm H can obtain a feasible solution, whose objective value is no more than dZ^{OPT}, where Z^{OPT} is the optimal value of VSR.*

Proof: Let A^* and $R^*=J\setminus A$ be the set of accepted jobs and rejected jobs in optimal solution, respectively. Obviously, the optimal value is

$$Z^{OPT}=\max_k\sum\nolimits_{J_j\in A^*}p_j^k+\sum\nolimits_{J_j\in R^*}w_j.$$

Let A and R be the set of accepted and rejected jobs in online algorithm, let Z^{OUT} be objective value of online algorithm.

Case1: If all jobs are rejected in solution by online algorithm, the objective value is

$$\begin{aligned}Z^{OUT}&=W(J)=W(A^*\cup R^*)\\&\le\sum\nolimits_{J_j\in A^*}p_j+\sum\nolimits_{J_j\in R^*}w_j\\&=\sum\nolimits_{J_j\in A^*}\sum\nolimits_{k=1}^{d}p_j^k+\sum\nolimits_{J_j\in R^*}w_j\\&\le d\max_k\sum\nolimits_{J_j\in A^*}p_j^k+\sum\nolimits_{J_j\in R^*}w_j\le dZ^{OPT}\end{aligned}$$

Which is no more than dZ^{OPT}.

Case2: Otherwise, let J_l be the last one in accepted job set. Since $R=(R\cap A^*)\cup(R\cap R^*)$, where$(R\cap A^*)\cap(R\cap R^*)=\emptyset$, the total rejection cost of online algorithm is

$$\begin{aligned}\sum\nolimits_{J_j\in R}w_j&=\sum\nolimits_{J_j\in R\cap A^*}w_j+\sum\nolimits_{J_j\in R\cap R^*}w_j\\&\le\sum\nolimits_{J_j\in R\cap A^*}p_j+\sum\nolimits_{J_j\in R\cap R^*}w_j.\end{aligned}$$

Due to $A=((A\setminus\{J_l\})\cap A^*)\cup((R\setminus\{J_l\})\cap R^*)\cup\{J_l\}$, for arbitrary $k=1,\ldots,d$, the k_{th} resource load of the single machine is

$$\begin{aligned}L^k&=\sum\nolimits_{J_j\in(A\setminus\{J_l\})}p_j^k+p_l^k\\&=\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap A^*}p_j^k+\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap R^*}p_j^k+p_l^k.\end{aligned}$$

Then, the sum of resource load is

$$\begin{aligned}\sum\nolimits_{k=1}^{d}L^k&\le\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap A^*}\sum\nolimits_{k=1}^{d}p_j^k+\\&\quad\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap R^*}\sum\nolimits_{k=1}^{d}p_j^k+\sum\nolimits_{k=1}^{d}p_l^k\\&\le\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap A^*}p_j+\\&\quad\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap R^*}p_j+p_l\\&\le\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap A^*}p_j+\\&\quad\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap R^*}w_j+p_l.\end{aligned}$$

The last inequality is just for the definition of A.

(i) If job $J_l\in A^*$, the objective value of online algorithm is

$$\begin{aligned}Z^{OUT}&=\max_k L^k+\sum\nolimits_{J_j\in R}w_j\\&\le\sum\nolimits_{k=1}^{d}L^k+\sum\nolimits_{J_j\in R}w_j\\&\le\sum\nolimits_{J_j\in R\cap A^*}p_j+\sum\nolimits_{J_j\in R\cap R^*}w_j+\\&\quad\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap A^*}p_j+\sum\nolimits_{J_j\in(A\setminus\{J_l\})\cap R^*}w_j+p_l\\&\le d\max_k\sum\nolimits_{J_j\in A^*}p_j^k+\sum\nolimits_{J_j\in R^*}w_j\\&\le dZ^{OPT}.\end{aligned}$$

(ii) Similarly, if job $J_l\in R^*$, the objective value of online algorithm is

$$\begin{aligned}Z^{OUT}&\le\sum\nolimits_{J_j\in A^*}p_j+\sum\nolimits_{J_j\in R^*\setminus\{J_l\}}w_j+w_l\\&\le d\max_k\sum\nolimits_{J_j\in A^*}p_j^k+\sum\nolimits_{J_j\in R^*}w_j\\&\le dZ^{OPT}.\end{aligned}$$

Then, the objective value of online algorithm is no more than dZ^{OPT} by (i)(ii).

In conclusion, online algorithm is $d-$compatitive. It is assumed that an online instance ends with only one job,

	T_1	T_2	T_3	T_4	T_5
p_1	0.5	1	0.75	2.5	1.5
p_2	1	3	2	3	5
p_3	5	9	5	9	7
w	7	6	8	12	10

which job has resource demand with $(1, \ldots, 1)$, the synthetical resource demand is $p_j = d$, its rejection cost is d. Therefore, this job will be rejected by online algorithm, the objective value is d. However, it must be accepted in optimal solution, each dimensional resource load is 1 on the single machine and the optimal value is 1, ultimately. Thus, the competitive ratio is exactly d. ■

We give a simple example to express algorithm H and its result. Given a computer, there are five tasks to be process one by one with three resource occupancy value storage, CPU core and hard disk size, in addition rejected cost, which come one by one as following. In the following table, let p_1, p_2, p_3 be the value of storage, CPU core and hard disk size respectively, and let w be the penalty cost.

Task1 and Task3 will be rejected whose value of penalty is not exceeding the sum of resource value, and the remaining tasks will be accepted by algorithm H. Then the sum of resource load is 25 and the sum of penalty cost is 15, which lead to the objective value of example is 40. However, only Task2 is rejected in the optimal solution, and the optimal value is 32. Obviously, the approximation ratio of algorithm H is no more the dimension 3.

III. Semi-online Algorithms

In this section, we consider two versions of semi-online VSR. The first one is reassignment after all jobs have been assigned, which means that some accepted jobs can be rejected and some rejected jobs can be accepted. The second version is scheduling with buffer in rejection tache. Buffer can store some jobs, when a certain jobs arriving, it can be either assigned immediately or be placed in buffer. But buffer does not apply to jobs that have been accepted.

A. Reassignment

After all jobs have been assigned, at most k jobs can be reassigned. In this section, we design a semi-online to reassign the last one job. Let J_x be the last accepted and completed job in accepted set A, let L_{x-1} be the load of machine without job J_x, and L_x be the load of machine after processing J_x. We design a semi-online algorithm A_R for reassignment.

Algorithm A_R

Step 1. When job $J_j = (\mathbf{p_j}, w_j)$ arrives, where $\mathbf{p_j} = \{p_j^1, \ldots, p_j^d\}$, if $w_j \leq \frac{1}{2}\max\{p_j^1, \ldots, p_j^d\}$, reject it. Otherwise, accept and schedule it on the single machine.

Step 2. If all jobs have been rejected, stop.

Step 3. For job J_x, if $L_x - L_{x-1} \leq w_x$, no reassignment for job J_x, stop.

Step 4. If $L_x - L_{x-1} > w_x$, then reject J_x, stop.

Theorem 2 *Algorithm A_R for semi-online VSR is at most $(\frac{1}{2}d + 2)-$competitive, where d is more than 3.*

Proof: Let A^{OPT}, R^{OPT} be the set of optimal accepted jobs and optimal rejected jobs, Z^{OPT} be the optimal value. Analogously, let A, R, Z^{OUT} be the same mean of algorithm VR. For arbitrary job set $X \subseteq J$, let $L(X)$ be the load of set X. Due to the definition of A^{OPT}, R^{OPT} and A, R, we know that $A^{OPT} \cap R^{OPT} = \emptyset$, $A \cap R = \emptyset$ and $J = A^{OPT} \cup R^{OPT}$, $J = A \cup R$.

Case1: If algorithm A_R stop at step2, which is mean that all jobs are rejected, then

$$\begin{aligned} Z^{OUT} &= W(J) = W(A^{OPT}) + W(R^{OPT}) \\ &\leq \sum\nolimits_{J_j \in A^{OPT}} \max\{p_j^1, \ldots, p_j^d\} + W(R^{OPT}) \\ &\leq \sum\nolimits_{J_j \in A^{OPT}} \sum\nolimits_{k=1}^{d} p_j^k + W(R^{OPT}) \\ &\leq \frac{1}{2} d \max_k \sum\nolimits_{J_j \in A^{OPT}} p_j^k + W(R^{OPT}) \\ &\leq \frac{1}{2} d L(A^{OPT}) + W(R^{OPT}) \\ &\leq \max\{\frac{1}{2}d, 1\} Z^{OPT} \end{aligned}$$

Since $\max\{\frac{1}{2}d, 1\} < \frac{1}{2}d + 2$, theorem2 is satiable.

Case2: If algorithm A_R stop at step3, then objective value is $Z^{OUT} = L_x + W(R)$. Because of the definition of A^{OPT}, R^{OPT} and A, R, then $R = (R \cap R^{OPT}) \cup (R \cap A^{OPT})$ and

$$\begin{aligned} W(R) &= W(R \cap R^{OPT}) + W(R \cap A^{OPT}) \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2} \sum\nolimits_{J_j \in (R \cap A^{OPT})} \max_k p_j^k \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2} \sum\nolimits_{J_j \in (R \cap A^{OPT})} \sum\nolimits_{k=1}^{d} p_j^k \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2} d \sum\nolimits_{J_j \in (R \cap A^{OPT})} p_j^k \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2} d L(R \cap A^{OPT}) \end{aligned}$$

Obviously, $L_x - L_{x-1} \geq 0$, then $L_x = \frac{1}{2}[L_{x-1} + (L_x - L_{x-1}) + L_x] \leq L_x + \frac{1}{2}(L_x - L_{x-1})$. Since $A = (A \cap A^{OPT}) \cup (A \cap R^{OPT})$ and $L(A) \leq L(A \cap A^{OPT}) + L(A \cap R^{OPT})$, then

$$\begin{aligned} L_x &\leq L_x + \frac{1}{2}(L_x - L_{x-1}) \\ &\leq L(A \cap A^{OPT}) + L(A \cap R^{OPT}) + \frac{1}{2}(L_x - L_{x-1}) \\ &\leq L(A \cap A^{OPT}) + \sum\nolimits_{J_j \in (A \cap R^{OPT})} \max_k p_j^k \\ &\quad + \frac{1}{2}(L_x - L_{x-1}) \\ &\leq L(A \cap A^{OPT}) + 2W(A \cap R^{OPT}) + \frac{1}{2}(L_x - L_{x-1}) \end{aligned}$$

(i) If job $J_x \in A^{OPT}$, in addition $L_x - L_{x-1} \leq L(\{J_x\})$,

then

$$\begin{aligned} Z^{OUT} &\leq W(R\cap R^{OPT})+\frac{1}{2}dL(R\cap A^{OPT}) \\ &\quad +L(A\cap A^{OPT})+2W(A\cap R^{OPT})+\frac{1}{2}(L_x-L_{x-1}) \\ &\leq 2W(R^{OPT})+\frac{1}{2}dL(A^{OPT})+L(A^{OPT})+\frac{1}{2}L(\{J_x\}) \\ &\leq 2W(R^{OPT})+(\frac{1}{2}d+1+\frac{1}{2})L(A^{OPT}) \\ &\leq \max\{\frac{1}{2}d+\frac{3}{2},2\}Z^{OPT} \end{aligned}$$

(ii) If job $J_x \in R^{OPT}$ and $L_x - L_{x-1} \leq w_x$ in step3, then

$$\begin{aligned} Z^{OUT} &\leq 2W(R^{OPT})+(\frac{1}{2}d+1)L(A^{OPT})+\frac{1}{2}w_j \\ &\leq (\frac{1}{2}d+1)L(A^{OPT})+\frac{5}{2}W(R^{OPT}) \\ &\leq \max\{\frac{1}{2}d+1,\frac{5}{2}\}Z^{OPT} \end{aligned}$$

Therefore, case2 satisfy theorem2.

Case3: If algorithm A_R stop at step4, then $Z^{OUT} = L_{x-1} + W(R)$. Due to $R = R \setminus \{J_x\}$, $W(R) = W(R \setminus \{J_x\}) + w_x$. We can get the following two inequality.

$$\begin{aligned} W(R) &= W((R\setminus\{J_x\})\cap A^{OPT})+ \\ &\quad W((R\setminus\{J_x\})\cap R^{OPT})+w_x \\ &\leq \frac{1}{2}\sum\nolimits_{J_j\in((R\setminus\{J_x\})\cap A^{OPT})}\max_k p_j^k+ \\ &\quad W((R\setminus\{J_x\})\cap R^{OPT})+w_x \\ &\leq \frac{1}{2}\sum\nolimits_{J_j\in((R\setminus\{J_x\})\cap A^{OPT})}\sum\nolimits_{k=1}^{d}p_j^k \\ &\quad +W((R\setminus\{J_x\})\cap R^{OPT})+w_x \\ &\leq \frac{1}{2}d\max_k\sum\nolimits_{J_j\in((R\setminus\{J_x\})\cap A^{OPT})}p_j^k \\ &\quad +W((R\setminus\{J_x\})\cap R^{OPT})+w_x \\ &\leq \frac{1}{2}dL((R\setminus\{J_x\})\cap A^{OPT}) \\ &\quad +W((R\setminus\{J_x\})\cap R^{OPT})+w_x \end{aligned}$$

$$\begin{aligned} L_{x-1} &\leq L(A\cap A^{OPT})+L(A\cap R^{OPT}) \\ &\leq L(A\cap A^{OPT})+\sum\nolimits_{J_j\in(A\cap R^{OPT})}\max_k p_j^k \\ &\leq L(A\cap A^{OPT})+2W(A\cap R^{OPT}) \end{aligned}$$

(i) If job $J_x \in A^{OPT}$, since $L_x - L_{x-1} \leq L(\{J_x\})$ and the term $L_x - L_{x-1} > w_x$ in step4, then

$$\begin{aligned} Z^{OUT} &\leq L(A\cap A^{OPT})+2W(A\cap R^{OPT})+ \\ &\quad \frac{1}{2}dL((R\setminus\{J_x\})\cap A^{OPT})+ \\ &\quad W((R\setminus\{J_x\})\cap R^{OPT})+w_x \\ &\leq L(A^{OPT})+\frac{1}{2}dL(A^{OPT})+2W(R^{OPT}\setminus\{J_x\}) \\ &\quad +(L_x-L_{x-1}) \\ &\leq (\frac{1}{2}d+1)L(A^{OPT})+2W(R^{OPT}\setminus\{J_x\}) \\ &\quad +L(\{J_x\}) \\ &\leq (\frac{1}{2}d+2)L(A^{OPT})+2W(R^{OPT}) \\ &\leq (\frac{1}{2}d+2)Z^{OPT} \end{aligned}$$

(ii) If If job $J_x \in R^{OPT}$, then

$$\begin{aligned} Z^{OUT} &\leq (\frac{1}{2}d+1)L(A^{OPT})+2W(R^{OPT}\setminus\{J_x\})+w_x \\ &\leq (\frac{1}{2}d+1)L(A^{OPT})+2W(R^{OPT}) \\ &\leq \max\{\frac{1}{2}d+1,2\}Z^{OPT} \end{aligned}$$

So, case3 also satisfy theorem2.

To sum up, the objective value Z^{OUT} of algorithm A_R is no more than $(\frac{1}{2}d+2)Z^{OPT}$ for $d > 3$. It suggests that theorem2 is proved. ■

B. Rejection Buffer

Buffer can contain at most k jobs, in this section, we put forward the semi-online vector scheduling with buffer is available to one job. If a job arriving, it can be either assigned at once or stored in the buffer with empty buffer. When the buffer is not vacant, one job must be selected to schedule from currently arriving job and the stored job in buffer. In this section, the definition of all symbols are same as the previous section.

ALGORITHM A_B

Step 1. When job $J_1 = (\mathbf{p_1}, w_1)$ arrives, where $\mathbf{p_1} = \{p_1^1, \ldots, p_1^d\}$, if $w_1 \leq \frac{1}{2}\max\{p_1^1, \ldots, p_1^d\}$, reject it. Otherwise, store it in the rejection buffer and let J_1 be J_y.

Step 2. If no job arrives, for job J_y:

Step 2.1. If $L_y - L_{y-1} \leq w_y$, accept job J_y and schedule it on the single machine, then stop.

Step 2.2. If $L_y - L_{y-1} > w_y$, then reject it, stop.

Step 3. When job $J_j = (\mathbf{p_j}, w_j)$ comes, if $w_j \leq \frac{1}{2}\max\{p_j^1, \ldots, p_j^d\}$, reject it and then go to step 2.

Step 4. If $w_j > \frac{1}{2}\max\{p_j^1, \ldots, p_j^d\}$, then choose and accept the smaller load between job J_j and job J_y. That is mean: if $\max\{p_j^1, \ldots, p_j^d\} \leq \max\{p_y^1, \ldots, p_y^d\}$ then accept job J_j, otherwise accept job J_y. Afterwards, store the other job in the rejection buffer and let the job be J_y, go to step 2.

Theorem 3 *Algorithm A_B for semi-online VSR is at most $\frac{1}{2}d+2-$competitive, where d is more than 3.*

Proof: Case1: If algorithm A_B reject all jobs, then it will obtain the same result as Theorem2.

Case2: If algorithm A_B stop at step 2.1, Then job J_y is the last completed job in A, and the objective value of A_B is $Z^{OUT} = L_y + W(R)$. Because of the definition of A^{OPT}, R^{OPT} and A, R, then

$$\begin{aligned} W(R) &= W(R \cap R^{OPT}) + W(R \cap A^{OPT}) \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2}\sum\nolimits_{J_j \in (R \cap A^{OPT})} \max_k p_j^k \\ &\leq W(R \cap R^{OPT}) + \frac{1}{2} dL(R \cap A^{OPT}) \end{aligned}$$

Obviously, $L_y - L_{y-1} \geq 0$, then $L_y = \frac{1}{2}[L_{y-1} + (L_y - L_{y-1}) + L_y] \leq L_y + \frac{1}{2}(L_y - L_{y-1})$ and

$$\begin{aligned} L_y &\leq L_y + \frac{1}{2}(L_y - L_{y-1}) \\ &\leq L(A \cap A^{OPT}) + L(A \cap R^{OPT}) + \frac{1}{2}(L_y - L_{y-1}) \\ &\leq L(A \cap A^{OPT}) + \sum\nolimits_{J_j \in (A \cap R^{OPT})} \max_k p_j^k \\ &\quad + \frac{1}{2}(L_y - L_{y-1}) \\ &\leq L(A \cap A^{OPT}) + 2W(A \cap R^{OPT}) + \frac{1}{2}(L_y - L_{y-1}) \end{aligned}$$

(i) If job $J_y \in A^{OPT}$, in addition $L_y - L_{y-1} \leq L(\{J_y\})$, then

$$\begin{aligned} Z^{OUT} &\leq W(R \cap R^{OPT}) + \frac{1}{2} dL(R \cap A^{OPT}) \\ &\quad + L(A \cap A^{OPT}) + 2W(A \cap R^{OPT}) + \frac{1}{2}(L_y - L_{y-1}) \\ &\leq 2W(R^{OPT}) + \frac{1}{2} dL(A^{OPT}) + L(A^{OPT}) + \frac{1}{2} L(\{J_y\}) \\ &\leq 2W(R^{OPT}) + (\frac{1}{2}d + 1 + \frac{1}{2}) L(A^{OPT}) \\ &\leq \max\{\frac{1}{2}d + \frac{3}{2}, 2\} Z^{OPT} \end{aligned}$$

(ii) If job $J_y \in R^{OPT}$ and $L_y - L_{y-1} \leq w_y$ in step2.1, then

$$\begin{aligned} Z^{OUT} &\leq 2W(R^{OPT}) + (\frac{1}{2}d + 1) L(A^{OPT}) + \frac{1}{2} w_j \\ &\leq (\frac{1}{2}d + 1) L(A^{OPT}) + \frac{5}{2} W(R^{OPT}) \\ &\leq \max\{\frac{1}{2}d + 1, \frac{5}{2}\} Z^{OPT} \end{aligned}$$

Case3: If algorithm A_B stop at step2.2, then $Z^{OUT} = L_{y-1} + W(R)$. Due to $R = R \setminus \{J_y\}$, $W(R) = W(R \setminus \{J_y\}) + w_y$. We can get the following two inequality.

$$\begin{aligned} W(R) &= W((R \setminus \{J_y\}) \cap A^{OPT}) + \\ &\quad W((R \setminus \{J_y\}) \cap R^{OPT}) + w_y \\ &\leq \frac{1}{2}\sum\nolimits_{J_j \in ((R \setminus \{J_y\}) \cap A^{OPT})} \max_k p_j^k + \\ &\quad W((R \setminus \{J_y\}) \cap R^{OPT}) + w_y \\ &\leq \frac{1}{2} dL((R \setminus \{J_y\}) \cap A^{OPT}) + \\ &\quad W((R \setminus \{J_y\}) \cap R^{OPT}) + w_y \end{aligned}$$

$$\begin{aligned} L_{y-1} &\leq L(A \cap A^{OPT}) + L(A \cap R^{OPT}) \\ &\leq L(A \cap A^{OPT}) + 2W(A \cap R^{OPT}) \end{aligned}$$

(i) If job $J_y \in A^{OPT}$, since $L_y - L_{y-1} \leq L(\{J_y\})$ and the term $L_y - L_{y-1} > w_y$ in step4, then

$$\begin{aligned} Z^{OUT} &\leq L(A \cap A^{OPT}) + 2W(A \cap R^{OPT}) + \\ &\quad \frac{1}{2} dL((R \setminus \{J_y\}) \cap A^{OPT}) + \\ &\quad W((R \setminus \{J_y\}) \cap R^{OPT}) + w_y \\ &\leq L(A^{OPT}) + \frac{1}{2} dL(A^{OPT}) + 2W(R^{OPT} \setminus \{J_y\}) \\ &\quad + (L_y - L_{y-1}) \\ &\leq (\frac{1}{2}d + 1) L(A^{OPT}) + 2W(R^{OPT} \setminus \{J_y\}) \\ &\quad + L(\{J_y\}) \\ &\leq (\frac{1}{2}d + 2) L(A^{OPT}) + 2W(R^{OPT}) \\ &\leq (\frac{1}{2}d + 2) Z^{OPT} \end{aligned}$$

(ii) If If job $J_y \in R^{OPT}$, then

$$\begin{aligned} Z^{OUT} &\leq (\frac{1}{2}d + 1) L(A^{OPT}) + 2W(R^{OPT} \setminus \{J_y\}) + w_y \\ &\leq (\frac{1}{2}d + 1) L(A^{OPT}) + 2W(R^{OPT}) \\ &\leq \max\{\frac{1}{2}d + 1, 2\} Z^{OPT} \end{aligned}$$

In conclusion, the competitive ratio of algorithm A_B is no more than $(\frac{1}{2}d+2)Z^{OPT}$ for $d > 3$. It implies that theorem3 is hold.

■

IV. Conclusion

We have studied online and semi-online vector scheduling on a single machine with rejection, we showed an online algorithm with $d-$competitive. In the future, we can extend what we considered to online vector scheduling on parallel machines with rejection. Semi-online with rearrangement and buffer was $\frac{1}{2}d + 2-$competitive, in which the value of k is 1 and d is more than 3. When $d \leq 3$, our results is more than d, in other words, the competitive ratio of semi-online algorithm is more than that of online

algorithm, which is unreasonable. Hence, the future is worth working on designing a better algorithm for semi-online vector scheduling with low dimensional vector.

ACKNOWLEDGMENT

The authors would like to thank some researchers for improving our paper with worthy suggestions.

REFERENCES

[1] W.D. Li and Q.N. Cui, *Vector scheduling with rejection on a single machine*, 4OR-A Quarterly Journal of Operations Research, 16: $95-104$, 2018.

[2] X.L. Qi and J.J. Yuan, *Semi-online hierarchical scheduling for l_p-norm load balancing with buffer or rearrangements*, 4OR-A Quarterly Journal of Operations Research, doi: 10.1007/s10288-016-0334-y.

[3] X. Min, Y.Q. Wang, J. Liu and M. Jiang, *Semi-online scheduling on two identical machine with rejection*, Journal of Combinatorial Optimization, doi: 10.1007/s10878-011-9435-x.

[4] Y. Bartal, S. Leonardi, A.M. Spaccamela, J. Sgall and L. Stougie, *Multiprocessor scheduling with rejection*, SIAM J. DISCRETE MATH, 13(1): $64-78$, 2000.

[5] X.J. Zhu, Q. Li, W.Z. Mao and G.H. Chen, *Online vector scheduling and generalized load balancing*, Journal of Parallel and Distributed Computing, 74: $2304-2309$, 2014.

[6] C. Chekuri and S. Khanna, *On multi-dimensional packing problems*, SIAM Journal on Computing, 33(4): $837-851$, 2004.

[7] S. Im, N. Kell, J.Kulkarni and D. Panigrahi, *Tight bounds for online vector scheduling*, 2015 IEEE $56th$ Annual Symposium on Foundations of Computer Science, doi: 10.1109/FOCS.2015.39.

[8] Y. Azar, I.R. Cohen and D. Panigrahi, *Randomized algorithms for online vector load balancing*, SIAM License or Copyright, http://www.siam.org/journals/ojsa.php.

[9] G. Dosa and Y. He, *Semi-online algorithms for parallel machien scheduling problems*, Computing, 72, $355-363$, 2004.

[10] Z.Y. Tan and S.H. Yu, *Online scheduling with reassignment*, Operations Research Letters, 36, $250-254$, 2008.

[11] X. Chen, Z.Z. Xu, G. Dosa, X. Han and H. Jiang, *Semi-online hierarchical scheduling problems with buffer or rearrangements*, Information Processing Letters, 113, $127-131$, 2013.