Hindawi Publishing Corporation
Advances in Operations Research
Volume 2012, Article ID 748597, 9 pages
doi:10.1155/2012/748597

*Research Article*

# The $C^{\max}$ Problem of Scheduling Multiple Groups of Jobs on Multiple Processors at Different Speeds

**Wei Ding**

*Department of Mathematics, Sun Yat-Sen University, Guangzhou 510275, China*

Correspondence should be addressed to Wei Ding, dingwei@mail.sysu.edu.cn

Received 1 April 2012; Revised 4 July 2012; Accepted 19 July 2012

Academic Editor: Ching-Jong Liao



We mainly study the $C^{\max}$ problem of scheduling $n$ groups of jobs on $n$ special-purpose processors and $m$ general-purpose processors at different speeds provided that the setup time of each job is less than $\alpha$ times of its processing time. We first propose an improved LS algorithm. Then, by applying this new algorithm, we obtain two bounds for the ratio of the approximate solution $T^{\mathrm{LS}}$ to the optimal solution $T^*$ under two different conditions.

## 1. Introduction

It is a well-studied problem to minimize the makespan in scheduling $n$ jobs $\{J_1, J_2, \ldots, J_n\}$ on $m$ identical machines $\{1, 2, \ldots, m\}$, where processing job $J_j$ immediately after $J_i$ needs a setup time $w(i, j)$. As it is NP-hard (cf. [1]), quite a few authors have made their efforts to its approximate and heuristic algorithms, as well as the corresponding worst-case analysis.

In 1969, Graham [2] showed in his fundamental paper that the bound of this scheduling problem is $2 - 1/m$ as $w(i, j) = 0$ under the LS (List Scheduling) algorithm and the tight bound is $4/3 - 1/3m$ under the LPT (Longest Processing Time) algorithm. In 1993, Ovacik and Uzsoy [3] proved that the bound is $4 - 2/m$ as $w(i, j) \leq t_j$, where $t_j$ is the processing time of the job $J_j$, under the LS algorithm. In 2003, Imreh [4] studied the online and offline problems on two groups of identical processors at different speeds, presented the LG (Load Greedy) algorithm, and showed that the bound about minimizing the makespan is $2 + (m - 1)/k$ and the bound about minimizing the sum of finish time is $2 + (m - 2)/k$, where $m$ and $k$ are the numbers of two groups of identical processors. Gairing et al. [5] proposed a simple combinatorial algorithm for the problem of scheduling $n$ jobs on $m$ processors at different speeds to minimize a cost stream and showed that it is effective and of low complexity.

Besides the above well-studied scheduling problem, one may face the problem of scheduling multiple groups of jobs on multiple processors in real production systems, such as, the problem of processing different types of yarns on spinning machines in spinning mills. Recently, the problem of scheduling multiple groups of jobs on multiple processors at same or different speeds were studied provided that each job has no setup time. In 2006, Ding [6] studied the problem of scheduling $n$ groups of jobs on one special-purpose processor and $n$ general-purpose processors at same speeds under an improved LPT algorithm. In 2008, Ding [7] investigated the problem of scheduling $n$ groups of jobs on $n$ special-purpose processors and $m$ general-purpose processors at same speeds under an improved LPT algorithm. In 2009, Ding [8] presented an improved LS algorithm for the $Q_{m+2}/r_j/C_{\max}$ scheduling problem on $m$ general-purpose processors and two special-purpose processors. In 2010, Ding [9] studied a heuristic algorithm of the $Q//C_{\max}$ problem on multitasks with uniform processors. In the same year, Ding and Zhao [10] discussed an improved LS algorithm for the problem of scheduling multiple groups of jobs on multiple processors at the same speed provided each job has a setup time.

Recently, Ding and Zhao [11] investigated an improved LS algorithm for the problem of scheduling multiple jobs on multiple uniform processors at different speeds provided that each job has a setup time. However, if each job has a setup time, then the problem of scheduling multiple groups of jobs on multiple processors at different speeds has not been studied yet. Note that the LPT algorithm and the improved LPT algorithm are not effective ways to deal with such a problem if each job has a setup time. Meanwhile, the classical LS algorithm is only useful to solve the problem of scheduling one group of jobs on multiple processors at same or different speeds. Therefore, our purpose of this study is to propose an improved LS algorithm based on the classical LS algorithm and the fact that the optimal solution $T^*$ is bigger than the average finish time of all processors, see the inequality (3.6) below, and to use this new algorithm to analyze this problem of scheduling multiple groups of jobs on multiple processors at different speeds provided that each job has a setup time.

The remainder of the paper is organized as follows. In Section 2, we proposed an improved LS algorithm for this scheduling problem. In Section 3, we obtain two bounds for the ratio of the approximate solution $T^{\mathrm{LS}}$ to the optimal solution $T^*$ under the improved LS algorithm.

*Notation 1.* As above and henceforth, we let $L_i$ $(i = 1,\ldots,n)$ denote the $i$th group of jobs, and let $M_i$ $(i = 1,\ldots,n)$ and $M_{n+j}$ $(j = 1,\ldots,m)$ denote the set of jobs on the $i$th special-purpose processor and the set of jobs on the $j$th general-purpose processor, respectively. Let $n_r$ $(r = 1,\ldots,n)$ denote the number of jobs in the $r$th group. We then use $J(r,i)$ $(r = 1,\ldots,n;$ $i = 1,\ldots,n_r)$ to denote the $i$th job of the $r$th group and use $t(r,i)$ $(r = 1,\cdots,n;\ i = 1,\ldots,n_r)$ to denote the processing time of $J(r,i)$. Let $P_r$ $(r = 1,\ldots,n)$ denote the set of the processing time $t(r,i)$ $(r = 1,\ldots,n_r)$. Moreover, we denote by $s_i$ $(i = 1,\ldots,n)$ the speed of the special-purpose processor $i$ and by $s_{n+j}$ $(j = 1,\ldots,m)$ the speed of the general-purpose processor $n+j$, respectively.

Note that the speeds of general-purpose processors are less than those of special-purpose processors in real production systems. For simplicity, we take $s_{n+j} = 1$ $(1 \le j \le m)$ and assume $s_i \ge 1$ $(i = 1,\ldots,n)$. If the job $J(h,j)$ $(h = 1,\ldots,n;\ j = 1,\ldots,n_h)$ is processed after the job $J(l,i)$ $(l = 1,\ldots,n;\ i = 1,\ldots,n_l)$, then we use $w(l,i;h,j)$ to denote the setup time the processor needs.

If the job $J(r,i)$ is assigned to the processor $k$ $(k = 1,2,\ldots,n+m)$, then we write $J(r,i) \in M_k$. Let $ML_k(k = 1,2,\ldots,n+m)$ stand for the set of jobs being processed to the processor $k$ and let

$$MT_k := \sum_{J(h,j)\in M_k} \left( w(*,*;h,j) + \frac{t(h,j)}{s_k} \right), \quad k = 1,2,\ldots,n+m. \tag{1.1}$$

Then, we use $MT_k(k = 1,2,\ldots,n+m)$ to denote the actual finish time of the processor $k$. Next, we write $T^{\mathrm{LS}} = \max_{1\le k\le n+m}\{MT_k\}$ as the actual latest finish time of $n+m$ processors under the improved LS algorithm and $T^*$ as the actual latest finish time of $n+m$ processors under the optimal algorithm, respectively. We finally denote $T^{\mathrm{LS}}$ by the approximate solution under the improved LS algorithm, $T^{\mathrm{LS}}/T^*$ by the bound of a scheduling problem under the improved LS algorithm.

## 2. An Improved LS Algorithm

In the section, we will propose an improved LS algorithm for the problem of scheduling multi groups of jobs on multi processors at different speeds provided that each job has a setup time.

The algorithm is defined by the fact that whenever a processor becomes idle for assignment, the first job unexecuted is taken from the list and assigned to this processor. If there are no less than one processor being idle, then the algorithm chooses the processor with the smallest index. If the processor is a special-purpose processor for some group, then the first job unexecuted in this group is assigned to the processor. If the processor is a general-purpose processor, then the job with the smallest second index is assigned to the processor. If there are several groups of jobs with the same second index, then the job with the smallest first index is assigned. In addition, there is an arbitrary order for any job in any group at the beginning of being processed.

The steps of the improved LS algorithm are the following.

*Step 1* (Initialization). Set $Q_1 = \{1,2,\ldots,n\}$, $Q_2 = \{n+1,n+2,\ldots,n+m\}$, $i_r = 1$, $ML_r = \emptyset$, $MT_r = 0$, $r \in Q_1$.

*Step 2* (Choose the first idle processor). If for some $r \in Q_1$, $i_r > n_r$, then set $Q_1 = Q_1 - \{r\}$ (i.e., all jobs in the group $L_r$ have been assigned). If $Q_1 = \emptyset$, then go to Step 5 (i.e., all jobs in all groups have been assigned). Set $p = \min\{k' \mid MT_{k'} = \min_{k\in Q_1\bigcup Q_2} MT_k\}$, (i.e., seek the first idle processor).

*Step 3* (Choose the job). If $p \le n$, then set $r = p$, $q = i_p$, $i_p = i_p + 1$ (i.e., the special-purpose processor is the first idle processor, then the first job waiting for assignment in the $p$th group is assigned). If $p > n$ (i.e., the general-purpose processor is the first idle processor), then set $h = \min\{r' \mid i_{r'} = \min_{r\in Q_1} i_r\}$ (i.e., the job with the smallest second index in nonempty groups is assigned), $r = h$, $q = i_h$, $i_h = i_h + 1$.

*Step 4.* Update the assignment and the latest finish time of the processor $p$. Set $ML_p = ML_p + \{J(r,q)\}$ and $MT_p = MT_p + w(*,*;r,q) + (t(r,q)/s_p)$. Then go to Step 2.

*Step 5.* Output the assignment $ML_k$, $k = 1,2,\ldots,n+m$, for every processor and the latest finish time

$$T^{\mathrm{LS}} = \max_{1\le k\le n+m}\{MT_k\}. \tag{2.1}$$

## 3. Analysis of the Improved LS Algorithm

In the section, we obtain two bounds for the ratio of the approximate solution $T^{\mathrm{LS}}$ to the optimal solution $T^*$ under two different conditions.

**Theorem 3.1.** *Consider the problem of scheduling $n$ groups of jobs $\{L_1, L_2, \ldots, L_n\}$ on $\{1,2,\ldots,n\}$ special-purpose processors and $\{n+1,\ n+2,\ldots,\ n+m\}$ general-purpose processors at different speeds provided each job has a setup time. Assume that $w(l,i;h,j) \le \alpha t(h,j)$ for all $l,h,i,j$. If the optimal solution $T^*$ is bigger than the processing time $t(r,j)$ of the latest finish job $J(r,j)$, then the bound of this scheduling problem under the improved LS algorithm is*

$$\frac{T^{LS}}{T^*} \le \frac{(n+m-1)(\alpha+(1/s_k))+(\alpha+2)\sum_{i=1}^{n} s_i}{n+m} \tag{3.1}$$

*for any $\alpha \ge 0$, where $s_k$ is the speed of the latest finish processor.*

*Proof.* Based on the improved LS algorithm, we may assume that some processor $k$ $(1 \le k \le n+m)$ is the latest finish processor and the latest finish job is $J(r,j)$ $(1 \le r \le n,\ 1 \le j \le n_r)$. Then on the processor $k$, we have

$$T^{\mathrm{LS}} = MT_k. \tag{3.2}$$

On other processors, we have

$$MT_i \ge MT_k - \left(w(*,*;r,j) + \frac{t(r,j)}{s_k}\right), \quad i = 1,2,\ldots,\ n+m,\ i \ne k. \tag{3.3}$$

By the assumption $w(*,*;r,j) \le \alpha t(r,j)$, $T^* \ge t(r,j)$, (3.2) and (3.3), we get

$$\begin{aligned} MT_i &\ge T^{\mathrm{LS}} - \left(w(*,*;r,j) + \frac{t(r,j)}{s_k}\right) \\ &\ge T^{\mathrm{LS}} - \left(\alpha + \frac{1}{s_k}\right)t(r,j) \\ &\ge T^{\mathrm{LS}} - \left(\alpha + \frac{1}{s_k}\right)T^*, \quad i = 1,2,\ldots,\ n+m,\ i \ne k. \end{aligned} \tag{3.4}$$

Thus

$$\begin{aligned}\sum_{i=1}^{n+m} MT_i &= MT_k + \sum_{\substack{i=1\\ i\neq k}}^{n+m} MT_i \\ &\geq (m+n)T^{\mathrm{LS}} - (m+n-1)\left(\alpha + \frac{1}{s_k}\right)T^*.\end{aligned} \tag{3.5}$$

On the other hand, since $T^*$ is the optimal solution, it follows that

$$T^* \geq \frac{\sum_{l=1}^{n}\sum_{i=1}^{n_l} t(l,i)}{\sum_{i=1}^{n+m} s_i}. \tag{3.6}$$

In view of the assumption and (3.6), we deduce

$$\begin{aligned}\sum_{i=1}^{n+m} MT_i &= \sum_{i=1}^{n+m} \sum_{\{t(h,p)\}\in ML_i} \left(w(*,*;h,p) + \frac{t(h,p)}{s_i}\right) \\ &\leq \sum_{i=1}^{n+m} \sum_{\{t(h,p)\}\in ML_i} \left(\alpha + \frac{1}{s_i}\right) t(h,p) \\ &\leq (\alpha+2) \sum_{i=1}^{n+m} \sum_{\{t(h,p)\}\in ML_i} t(h,p) \\ &\leq (\alpha+2) \sum_{h=1}^{n} \sum_{p=1}^{n_h} t(h,p) \\ &\leq (\alpha+2) T^* \sum_{i=1}^{n+m} s_i.\end{aligned} \tag{3.7}$$

Using (3.5) and (3.7), we have

$$(\alpha+2)T^* \sum_{i=1}^{n+m} s_i \geq \sum_{i=1}^{n+m} MT_i \geq (m+n)T^{\mathrm{LS}} - (m+n-1)\left(\alpha + \frac{1}{s_k}\right)T^*. \tag{3.8}$$

This yields

$$\left((m+n-1)\left(\alpha + \frac{1}{s_k}\right) + (\alpha+2)\sum_{i=1}^{n+m} s_i\right)T^* \geq (m+n)T^{\mathrm{LS}}. \tag{3.9}$$

Therefore

$$\frac{T^{\mathrm{LS}}}{T^*} \leq \frac{(n+m-1)(\alpha + (1/s_k)) + (\alpha+2)\sum_{i=1}^{n} s_i}{n+m}. \tag{3.10}$$

This completes the proof of the theorem. □

**Table 1:** The processing time of jobs.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $P_1$ | $t(1,1)=9$ | $t(1,2)=6$ | $t(1,3)=3$ | $t(1,4)=3$ | $t(1,5)=3$ | $t(1,6)=3$ | $t(1,7)=1$ | $t(1,8)=1$ | $t(1,9)=8$ |
| $P_2$ | $t(2,1)=6$ | $t(2,2)=4$ | $t(2,3)=2$ | $t(2,4)=2$ | $t(2,5)=2$ | $t(2,6)=2$ | | | |
| $P_3$ | $t(3,1)=3$ | $t(3,2)=2$ | $t(3,3)=1$ | $t(3,4)=1$ | $t(3,5)=1$ | $t(3,6)=1$ | | | |

**Table 2:** The setup time of jobs.

| | | |
|---|---|---|
| $w(0,0;1,1)=9$ | $w(0,0;2,1)=6$ | $w(0,0;3,1)=3$ |
| $w(1,1;1,3)=3$ | $w(2,1;2,3)=2$ | $w(3,1;3,2)=2$ |
| $w(1,3;1,4)=3$ | $w(2,3;2,4)=2$ | $w(3,2;3,3)=1$ |
| $w(1,4;1,5)=3$ | $w(2,4;2,5)=2$ | $w(3,3;3,6)=1$ |
| $w(1,5;1,6)=3$ | $w(2,5;2,6)=2$ | $w(2,2;3,3)=1$ |
| $w(1,6;1,9)=8$ | $w(0,0;2,2)=4$ | $w(3,3;3,5)=1$ |
| $w(0,0;1,2)=6$ | $w(1,2;1,7)=1$ | $w(3,5;1,8)=1$ |
| | $w(*,*;*,*)=0$ for other jobs. | |

*Example 3.2.* Consider the following scheduling problem. Assume that there are three groups of jobs and each group separately owns two special-purpose processors and jointly owns three general-purpose processors. Assume further that $\alpha = 1$, $s_1 = 3$, $s_2 = 2$, $s_3 = 1$, $s_4 = 1$, $s_5 = 1$, (see Tables 1 and 2).

The schedule for this example under the improved LS algorithm is found in Table 3.

The schedule for this example under the optimal algorithm is arranged as follows. found in Table 4.

In this example, we have $n = 3$, $m = 2$, $\alpha = 1$ and $s_k = 3$. Thus, we get

$$T^{\mathrm{LS}} = \min\left\{\frac{92}{3}, 18, 18, 20, 20\right\} = \frac{92}{3}, \quad T^* = 8,$$
$$\frac{T^{\mathrm{LS}}}{T^*} = \frac{23}{6} \leq \frac{(3+2-1)(1+1/3)+(1+2)(3+2+1)}{3+2} = \frac{14}{3}, \tag{3.11}$$

which is consistent with the conclusion of Theorem 3.1.

If we do not know whether or not $T^*$ is bigger than the processing time $t(r,j)$ of the latest finish job $J(r,j)$, then we have the following result.

**Theorem 3.3.** *Consider the scheduling problem in Theorem 3.1. Assume that $w(l,i;h,j) \leq \alpha t(h,j)$ for all $l,h,i,j$. Then the bound of this scheduling problem under the improved LS algorithm is*

$$\frac{T^{LS}}{T^*} \leq \frac{(n+m-1)(\alpha s_k+1)+(\alpha+2)\sum_{i=1}^{n} s_i}{n+m} \tag{3.12}$$

*for any $\alpha \geq 0$, where $s_k$ is the speed of the latest finish processor.*

**Table 3:** The improved LS schedule.

| Processors | Processing times of jobs | | | | | | $MT_i$ |
|---|---|---|---|---|---|---|---|
| Processor 1 ($s_1 = 3$) | $t(1,1) = 9$ | $t(1,4) = 3$ | $t(1,6) = 3$ | $t(1,9) = 8$ | | | 92/3 |
| Processor 2 ($s_2 = 2$) | $t(2,1) = 6$ | $t(2,3) = 2$ | $t(2,4) = 2$ | $t(2,6) = 2$ | | | 18 |
| Processor 3 ($s_3 = 1$) | $t(3,1) = 3$ | $t(3,2) = 2$ | $t(3,3) = 1$ | $t(3,4) = 1$ | $t(3,5) = 1$ | $t(3,6) = 1$ | 18 |
| Processor 4 ($s_4 = 1$) | $t(1,2) = 6$ | $t(1,5) = 3$ | $t(1,7) = 1$ | | | | 20 |
| Processor 5 ($s_5 = 1$) | $t(2,2) = 4$ | $t(1,3) = 3$ | $t(2,5) = 2$ | $t(1,8) = 1$ | | | 20 |

**Table 4:** The optimal schedule.

| Processors | Processing times of jobs | | | | | $MT_i$ |
|---|---|---|---|---|---|---|
| Processor 1 ($s_1 = 3$) | $t(1,5) = 3$ | $t(1,4) = 3$ | $t(1,3) = 3$ | $t(1,2) = 6$ | $t(1,1) = 9$ | 8 |
| Processor 2 ($s_2 = 2$) | $t(2,5) = 2$ | $t(2,4) = 2$ | $t(2,3) = 2$ | $t(2,2) = 4$ | $t(2,1) = 6$ | 8 |
| Processor 3 ($s_3 = 1$) | $t(3,5) = 1$ | $t(3,4) = 1$ | $t(3,3) = 1$ | $t(3,2) = 2$ | $t(3,1) = 3$ | 8 |
| Processor 4 ($s_4 = 1$) | $t(1,6) = 3$ | $t(1,7) = 1$ | $t(1,8) = 1$ | $t(2,6) = 2$ | $t(3,6) = 1$ | 8 |
| Processor 5 ($s_5 = 1$) | $t(1,9) = 8$ | | | | | 8 |

*Proof.* Based on the improved LS algorithm, we may assume that some processor $k$ $(1 \le k \le n+m)$ is the latest finish processor and the latest finish job is $J(r,j)$ $(1 \le r \le n,\ 1 \le j \le n_r)$. Then on the processor $k$, we have

$$T^{\mathrm{LS}} = MT_k. \tag{3.13}$$

On other processors, we have

$$MT_i \ge MT_k - \left( w(*,*;r,j) + \frac{t(r,j)}{s_k} \right), \quad i = 1,2,\ldots,n+m, i \ne k. \tag{3.14}$$

By the assumption $w(*,*;r,j) \le \alpha t(r,j)$, $T^* \ge t(r,j)/s_k$, (3.13) and (3.14), we get

$$\begin{aligned} MT_i &\ge T^{\mathrm{LS}} - \left( w(*,*;r,j) + \frac{t(r,j)}{s_k} \right) \\ &\ge T^{\mathrm{LS}} - \left( \alpha + \frac{1}{s_k} \right) t(r,j) \\ &\ge T^{\mathrm{LS}} - (\alpha s_k + 1) T^*, \quad i = 1,2,\ldots,n+m, i \ne k. \end{aligned} \tag{3.15}$$

Thus

$$\begin{aligned} \sum_{i=1}^{n+m} MT_i &= MT_k + \sum_{\substack{i=1 \\ i \ne k}}^{n+m} MT_i \\ &\ge (m+n) T^{\mathrm{LS}} - (m+n-1)(\alpha s_k + 1) T^*. \end{aligned} \tag{3.16}$$

On the other hand, since $T^*$ is the optimal solution, it follows that

$$T^* \geq \frac{\sum_{l=1}^{n} \sum_{i=1}^{n_l} t(l,i)}{\sum_{i=1}^{n+m} s_i}. \tag{3.17}$$

In view of the assumption and (3.17), we deduce

$$\begin{aligned}
\sum_{i=1}^{n+m} MT_i &= \sum_{i=1}^{n+m} \sum_{\{t(h,p)\} \in ML_i} \left( w(*,*;h,p) + \frac{t(h,p)}{s_i} \right) \\
&\leq \sum_{i=1}^{n+m} \sum_{\{t(h,p)\} \in ML_i} \left( \alpha + \frac{1}{s_i} \right) t(h,p) \\
&\leq (\alpha+2) \sum_{i=1}^{n+m} \sum_{\{t(h,p)\} \in ML_i} t(h,p) \\
&\leq (\alpha+2) \sum_{h=1}^{n} \sum_{p=1}^{n_h} t(h,p) \\
&\leq (\alpha+2) T^* \sum_{i=1}^{n+m} s_i.
\end{aligned} \tag{3.18}$$

Using (3.16) and (3.18), we have

$$(\alpha+2) T^* \sum_{i=1}^{n+m} s_i \geq \sum_{i=1}^{n+m} MT_i \geq (m+n) T^{\mathrm{LS}} - (m+n-1)(\alpha s_k + 1) T^*. \tag{3.19}$$

This yields

$$\left( (m+n-1)(\alpha s_k + 1) + (\alpha+2) \sum_{i=1}^{n+m} s_i \right) T^* \geq (m+n) T^{\mathrm{LS}}. \tag{3.20}$$

Therefore

$$\frac{T^{\mathrm{LS}}}{T^*} \leq \frac{(n+m-1)(\alpha s_k + 1) + (\alpha+2) \sum_{i=1}^{n} s_i}{n+m}. \tag{3.21}$$

This completes the proof of the theorem. □

## Acknowledgments

This work was partially supported by NSFC (No. 10971234). The author thanks the referee for the valuable comments and suggestions.

## References


[1] P. Schuurman and G. J. Woeginger, "Polynomial time approximation algorithms for machine scheduling: ten open problems," *Journal of Scheduling*, vol. 2, no. 5, pp. 203–213, 1999.

[2] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, pp. 416–429, 1969.

[3] I. M. Ovacik and R. Uzsoy, "Worst-case error bounds for parallel machine scheduling problems with bounded sequence-dependent setup times," *Operations Research Letters*, vol. 14, no. 5, pp. 251–256, 1993.

[4] C. Imreh, "Scheduling problems on two sets of identical machines," *Computing*, vol. 70, no. 4, pp. 277–294, 2003.

[5] M. Gairing, B. Monien, and A. Woclaw, "A faster combinatorial approximation algorithm for scheduling unrelated parallel machines," *Theoretical Computer Science*, vol. 380, no. 1-2, pp. 87–99, 2007.

[6] W. Ding, "A type of scheduling problem on general-purpose machinery and $n$ group tasks," *OR Transactions*, vol. 10, no. 4, pp. 122–126, 2006.

[7] W. Ding, "A type of scheduling problem on $m$ general-purpose machines and $n$-groups of tasks with uniform processors," *Acta Scientiarum Naturalium Universitatis Sunyatseni*, vol. 47, no. 3, pp. 19–22, 2008.

[8] W. Ding, "An improved LS algorithm for the $Q_{m+2}/r_j/C_{max}$ scheduling problem on $m$ general-purpose machines and two special-purpose machines," *Communication on Applied Mathematics and Computation*, vol. 23, no. 2, pp. 26–34, 2009.

[9] W. Ding, "Heuristic algorithm for the $Q//C_{max}$ problem on multi-tasks with uniform processors," *Acta Scientiarum Naturalium Universitatis Sunyatseni. Zhongshan Daxue Xuebao. Ziran Kexue Ban*, vol. 49, no. 1, pp. 5–8, 2010.

[10] W. Ding and Y. Zhao, "An improved LS algorithm for the problem of scheduling multi groups of jobs on multi processors at the same speed," *Algorithmic Operations Research*, vol. 5, no. 1, pp. 34–38, 2010.

[11] W. Ding and Y. Zhao, "An analysis of LS algorithm for the problem of scheduling multiple jobs on multiple uniform processors with ready time," *Pacific Journal of Optimization*, vol. 7, no. 3, pp. 551–564, 2011.