Journal of Southeast University (English Edition) Vol. 28, No. 3, pp. 349 - 353 Sept. 2012 ISSN 1003—7985

Single-machine scheduling with preventive periodic maintenance and resumable jobs in remanufacturing system

Liu Biyu[1,2] Chen Weida[1]

([1]School of Economics and Management, Southeast University, Nanjing 211189, China)
([2]School of Management Science and Engineering, Anhui University of Technology, Maanshan 243000, China)

Abstract: A single-machine scheduling with preventive periodic maintenance activities in a remanufacturing system including resumable and non-resumable jobs is studied. The objective is to find a schedule to minimize the makespan and an LPT-LS algorithm is proposed. Non-resumable jobs are first scheduled in a machine by the longest processing time (LPT) rule, and then resumable jobs are scheduled by the list scheduling (LS) rule. And the worst-case ratios of this algorithm in three different cases in terms of the value of the total processing time of the resumable jobs (denoted as S_2) are discussed. When S_2 is longer than the spare time of the machine after the non-resumable jobs are assigned by the LPT rule, it is equal to 1. When S_2 falls in between the spare time of the machine by the LPT rule and the optimal schedule rule, it is less than 2. When S_2 is less than the spare time of the machine by the optimal schedule rule, it is less than 2. Finally, numerical examples are presented for verification.

Key words: single-machine scheduling; preventive periodic maintenance; resumable jobs; LPT-LS algorithm
doi: 10.3969/j.issn.1003-7985.2012.03.017

Most literature on scheduling theory assumes that there are only non-resumable jobs. In fact, this assumption may not be valid in a real production situation such as in the reprocessing workshop of a remanufacturing industry. There also exist resumable jobs. In the remanufacturing industry, in order to reassemble finished products, new components are required since the recovery rate of return components can never reach 100%. In general, a disassembly order is always released first and then the disassembly result determines whether a purchasing order is needed or not[1]. And it is necessary to reprocess the disassembling cores as early as possible in order to determine whether new components are purchased or not. Our objective is to find a schedule to minimize the makespan. For simplicity, we only discuss off-line cores and assume that the reprocessing time has been determined by worker experience. It is common to observe in practice that before a machine is maintained it has some spare time but there is not a non-resumable job whose processing time is exactly equal to the machine's spare time and no job can be scheduled into the machine in the same period. In order to make the best of the machine capacity, we can schedule the resumable jobs after processing the non-resumable jobs (In the remanufacturing industry, we assume that there is zero setup time). Therefore, scheduling non-resumable and resumable jobs has gradually become a common practice in many remanufacturing enterprises. With a proper scheduling rule, the workshop can improve production efficiency, resulting in increased productivity and a high degree of customer satisfaction[2].

Some literature studies the single-machine scheduling problem with single maintenance and non-resumable jobs. Liao and Chen [3] considered a scheduling problem with the objective of minimizing the maximum tardiness. They proposed a branch-and-bound algorithm to derive the optimal schedule and a heuristic algorithm for large-sized problems. They also provided computational results to demonstrate the efficiency of their heuristics. Lee[4] showed that the longest processing time (LPT) rule has a tight worst-case ratio of 4/3 for the objective of minimizing the makespan, and he also presented heuristics for other objectives, such as minimizing the maximum tardiness, the number of tardy jobs, and the total weighted completion time, etc. Lee and Liman[5] proved that the worst-case ratio of the shortest processing time (SPT) rule is 9/7 for the objective of minimizing the total completion time. Graves and Lee[6] extended the problem to consider semi-resumable jobs. Ma et al.[7] discussed a single-machine scheduling problem with an unavailable period in a semiresumable case to minimize makespan and presented a LPT-based heuristic. Chen[8] considered a single-machine scheduling problem with periodic maintenance to find a schedule that minimizes the number of tardy jobs subject to periodic maintenance and nonresumable jobs. An effective heuristic and a branch-and-bound algorithm were proposed to find the optimal schedule. Wu

Received 2012-01-09.

Biographies: Liu Biyu (1981—), female, graduate, lecturer; Chen Weida (corresponding author), male, doctor, professor, cwd@seu.edu.cn.

Foundation items: The National Natural Science Foundation of China (No. 70971022, 71271054), the Scientific Research Innovation Project for College Graduates in Jiangsu Province (No. CXLX_0157), the Scientific Research Foundation of the Education Department of Anhui Province (No. 2011sk123).

Citation: Liu Biyu, Chen Weida. Single-machine scheduling with preventive periodic maintenance and resumable jobs in remanufacturing system[J]. Journal of Southeast University (English Edition), 2012, 28(3): 349 - 353. [doi: 10.3969/j.issn.1003-7985.2012.03.017]

and Lee [9] discussed a special problem where a jobs' processing time is a decreasing function of its position in the schedule. When jobs are resumable, it is shown that the SPT rule provides the optimal schedule, and for the non-resumable case, a mixed integer programming technique is proposed. Yang et al. [10] studied a single-machine scheduling with the job-dependent aging effects under multiple maintenance activities and variable maintenance duration considerations simultaneously to minimize the makespan. Ji et al. [11] considered a single machine scheduling problem with periodic maintenance for the objective of minimizing the makespan. They first show that the worst-case ratio of the classical longest processing time algorithm is 2.

In this paper we consider the scheduling problem with periodic maintenance to minimize the makespan for both non-resumable and resumable jobs. First, we propose an algorithm by which the jobs are scheduled. Then we discuss the worst-case ratios of this algorithm in three different cases in terms of the value of the total processing time of the resumable jobs. And, further, some instances are given to confirm the results.

1 Problem Description

Formally, the considered problem can be described as follows:

We assume that there are n independent jobs $J = \{J_{11}, J_{12}, \ldots, J_{1n_1}, \ldots, J_{21}, J_{22}, \ldots, J_{2n_2}\}$, including non-resumable and resumable ones which are reprocessed on a single machine, and assume that all of these jobs' arrival time is zero. Here non-resumable ones are those jobs which must be restarted if they cannot be finished before maintenance activity; resumable ones are those jobs which can be reprocessed based on the previous process even if they are not finished before maintenance activity. The number of the former is n_1 and that of the latter is n_2. Obviously, n_1 plus n_2 is equal to n. They are defined as group 1 and group 2, respectively.

The processing time of job J_{ij} is p_{ij} (when $i = 1$, $j = 1, 2, \ldots, n_1$; when $i = 2$, $j = 1, 2, \ldots, n_2$). We assume that $T \geqslant p_{1j}$ for each $j = 1, 2, \ldots, n_1$; otherwise, there is trivially no feasible schedule. Let C_{ij} be the completion time of job J_{ij}, then the objective is to minimize the makespan, which is defined as $C_{\max} = \max_{i=1,2;j=1,2,\ldots,n_i} C_j^{(i)}$. Using the three-field notation of Graham et al. [12], we denote this scheduling problem as $1/(nr-r) - pm/C_{\max}$. It can be easily seen that this problem is strongly NP-hard [4], but no approximation algorithm has been provided and analyzed in the literature.

We consider each interval between two consecutive maintenance activities as a batch with a capacity T which corresponds to the length of the time interval between two consecutive maintenance periods, and t is the amount of time required to perform each maintenance activity. Thus, a schedule π can be denoted as $\pi = (B_1, M_1, B_2, M_2, \ldots, M_{L-1}, B_L)$, where M_i is the i-th maintenance activity, L is the number of batches, and B_i is the i-th batch of jobs. An illustration of the considered problem in the form of a Gantt chart is given in Fig. 1, where $J_{[1j]}$ denotes the non-resumable job placed in the j-th position of the given schedule.

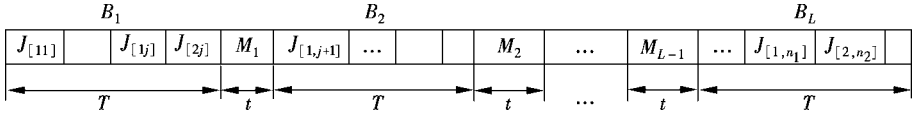

Fig. 1 Illustration of the problem under consideration

We use the worst-case ratio to measure the quality of an approximation algorithm. Specifically, for the makespan problem, let C_P denote the makespan produced by an approximation algorithm P, and C_{OPT} denote the makespan produced by an optimal off-line algorithm. Then the worst-case ratio of algorithm P is defined as the smallest number c, i.e. $C_P/C_{OPT} \leqslant c$.

2 The Algorithm and Its Worst-Case Ratio

In this section we analyze the LPT-LS algorithm and its worst-case ratio. Before analyzing the LPT-LS algorithm, we first define two kinds of classical algorithms, which are the LPT and LS algorithms, respectively.

The LPT rule is a classical heuristic for solving scheduling problems. It can be formally described as follows.

Algorithm 1 (LPT) All the jobs are re-ordered such that $p_{11} \geqslant p_{12} \geqslant \ldots \geqslant p_{1n_1}$; then these jobs are processed consecutively as early as possible.

The LS algorithm is a classical polynomial time approximation algorithm for solving scheduling problems. It can be described as follows.

Algorithm 2 (LS) When a job is available (ties are broken arbitrarily), it is assigned to the processing interval of the machine where it can be finished as early as possible.

Here we define the LPT-LS algorithm as follows:

Algorithm 3 (LPT-LS) The first step is to schedule the non-resumable jobs by the LPT rule; the second step is to schedule the resumable jobs by the LS rule for this considered scheduling problem.

Let $C_{LPT\text{-}LS}$ be the makespan of all scheduled jobs including non-resumable and resumable jobs; n_{OPT} is the number of machine working periods (corresponding to the bin numbers of bin packing problems) by an optimal off-line algorithm; L_{OPT} is the machine working time length in the last period by an optimal off-line algorithm ($0 \leqslant L_{OPT} \leqslant$

T); C_{LPT} is the makespan produced by the LPT algorithm (i.e., the completion time of job J_{1j}); n_{LPT} is the number of machine working periods by the LPT algorithm; L_{LPT} is the machine working time length in the last period by the LPT algorithm ($0 \leqslant L_{\mathrm{LPT}} \leqslant T$).

The makespan of the optimal schedule is

$$C_{\mathrm{OPT}} = (n_{\mathrm{OPT}} - 1)(T + t) + L_{\mathrm{OPT}} \tag{1}$$

While the makespan of the LPT schedule is

$$C_{\mathrm{LPT}} = (n_{\mathrm{LPT}} - 1)(T + t) + L_{\mathrm{LPT}} \tag{2}$$

Eq. (2) subtracts Eq. (1), and then we obtain

$$C_{\mathrm{LPT}} - C_{\mathrm{OPT}} = (n_{\mathrm{LPT}} - n_{\mathrm{OPT}})(T + t) + L_{\mathrm{LPT}} - L_{\mathrm{OPT}} \tag{3}$$

Lemma 1[11] The worst-case ratio of the LPT algorithm for periodic maintenance is 2, i.e., $C_{\mathrm{LPT}}/C_{\mathrm{OPT}} = 2$.

This first step of the scheduling is similar to the problem that Ji et al.[11] have researched. They prove that the worst-case ratio of the classical LPT algorithm is 2. And they show that there is no polynomial time approximation algorithm with a worst-case ratio less than 2 unless $P = NP$, which implies that the LPT algorithm is possibly the best. So, for non-resumable jobs the LPT algorithm is also possibly the best.

For the problem $1/(nr - r) - pm/C_{\max}$, there exist three cases about the worst-case ratio of the LPT-LS algorithm in terms of the value of S_2 (The total processing time of Group 2 jobs). The conclusions are proved in the following theorems.

First, we define a function $f(x) = k$ when $k < x \leqslant k + 1$, and k is a positive integer. Obviously, $f(S_i/T)$ is the frequency of machine maintenance, where S_i is the total processing time of all scheduled jobs (i.e. S_1 is the total processing time of Group 1 jobs). And it is easy to obtain

$$n_i - 1 \leqslant 2f\left(\frac{S_i}{T}\right) \tag{4}$$

In the following we discuss the worst-case ratio when S_2 takes different values.

Theorem 1 When $S_2 \geqslant C_{\mathrm{LPT}} - S_1 - (n_{\mathrm{LPT}} - 1)t$, $C_{\mathrm{LPT\text{-}LS}}/C_{\mathrm{OPT}} = 1$.

Proof It means that the total processing time of resumable jobs is longer than the machine spare time after scheduling non-resumable jobs. Obviously, in this situation, the resumable jobs can be assigned into the machine's spare time so that the machine is always processing jobs without maintenance. We can obtain that

$$C_{\mathrm{LPT\text{-}LS}} = (S_1 + S_2) + f\left(\frac{S_1 + S_2}{T}\right)t \tag{5}$$

It is the optimal algorithm. That is $C_{\mathrm{LPT\text{-}LS}}/C_{\mathrm{OPT}} = 1$.

Lemma 2 While $n_{\mathrm{LPT}} \geqslant 2$ in the LPT schedule, $S_1 > n_{\mathrm{LPT}}T/2$.

Proof Note that the total processing time of the jobs in any two pairwise used bins is strictly greater than T by the LPT rule. And in all bins, there is at most one bin in which the total processing time of the jobs is less than $T/2$. Regardless of which bin it is, the total processing time in it plus any other bin's must be greater than T. We discuss it as follows:

Case 1 If n_{LPT} is even, then

$$S_1 = \sum_{k=1}^{n_{\mathrm{LPT}}} B_k = \sum_{j=1}^{n_1} p_{1j} > \frac{bT}{2} \tag{6}$$

Case 2 If n_{LPT} is odd, we can choose $(n_{\mathrm{LPT}} - 1)$ bins including the bin in which the jobs' processing time is less than $T/2$ and separate them into $(n_{\mathrm{OPT}} - 1)/2$ groups with two pairwise bins. The total processing time of jobs in the remaining bin (denoted as B_x) is greater than $T/2$.

Then

$$S_1 = \sum_{k=1}^{n_{\mathrm{LPT}}} B_k = \sum_{j=1}^{n_1} p_{1j} = \frac{(b-1)T}{2} + t(B_x) > \frac{bT}{2} \tag{7}$$

Hence, we obtain

$$S_1 > \frac{bT}{2} \tag{8}$$

Theorem 2 When $C_{\mathrm{OPT}} - S_1 - (n_{\mathrm{OPT}} - 1)t \leqslant S_2 < C_{\mathrm{LPT}} - S_1 - (n_{\mathrm{LPT}} - 1)t$, $C_{\mathrm{LPT\text{-}LS}}/C_{\mathrm{OPT}} < 2$.

Proof It means that the total processing time of resumable jobs is shorter than the machine spare time after scheduling non-resumable jobs by the LPT rule and longer than the optimal algorithm. It is obvious that

$$C_{\mathrm{LPT}} \leqslant n_{\mathrm{LPT}}T + (n_{\mathrm{LPT}} - 1)t \tag{9}$$

Note that

$$n_{\mathrm{LPT}} - 1 \leqslant 2f\left(\frac{S_1}{T}\right) \tag{10}$$

Combining Eqs. (8), (10) and Lemma 3, we can obtain

$$\frac{C_{\mathrm{LPT}}}{S_1 + f\left(\frac{S_1}{T}\right)t} \leqslant \frac{n_{\mathrm{LPT}}T + (n_{\mathrm{LPT}} - 1)t}{S_1 + f\left(\frac{S_1}{T}\right)t} < \frac{n_{\mathrm{LPT}}T + (n_{\mathrm{LPT}} - 1)t}{\frac{n_{\mathrm{LPT}}T}{2} + f\left(\frac{S_1}{T}\right)t} \leqslant \frac{n_{\mathrm{LPT}}T + (n_{\mathrm{LPT}} - 1)t}{\frac{n_{\mathrm{LPT}}T}{2} + \frac{(n_{\mathrm{LPT}} - 1)}{2}t} = 2 \tag{11}$$

Theorem 3 When $S_2 < C_{\mathrm{OPT}} - S_1 - (n_{\mathrm{OPT}} - 1)t$, $C_{\mathrm{LPT-LS}}/C_{\mathrm{OPT}} < 2$.

Proof It means that the total processing time of resumable jobs is shorter than the machine spare time after scheduling non-resumable jobs by the optimal algorithm. Then the resumable jobs can be scheduled into those bins which still have spare volume; i.e., excluding the last bin, the other bins can be fully loaded without any spare

volume. So it is similar to the scheduling problem which Ji et al. [11] have researched.

Therefore, the worst-case ratios of the LPT-LS algorithm for the problem $1/(nr-r)-pm/C_{\max}$ in terms of the value of S_2 are as follows:

1) When $S_2 \geqslant C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$, $C_{\text{LPT-LS}}/C_{\text{OPT}} = 1$;

2) When $C_{\text{OPT}} - S_1 - (n_{\text{OPT}}-1)t \leqslant S_2 < C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$, $C_{\text{LPT-LS}}/C_{\text{OPT}} < 2$;

3) When $S_2 < C_{\text{OPT}} - S_1 - (n_{\text{OPT}}-1)t$, $C_{\text{LPT-LS}}/C_{\text{OPT}} < 2$.

3 Numerical Examples

In order to verify the results, some numerical examples are presented. We assume that there exist some different cases in a reprocessing workshop and some data are listed in Tab. 1. We denote $S_1 = \sum p_{1j}$, $S_2 = \sum p_{2j}$.

Tab. 1 Some data in a reprocessing workshop

No.	T	t	n	n_1	n_2	p_{ij}
1	15	0.5	4	3	1	$p_{11}=6, p_{12}=8, p_{13}=5, p_{21}=2$
2	20	0.5	8	7	1	$p_{11}=14, p_{12}=12, p_{13}=10, p_{14}=6$ $p_{15}=6, p_{16}=4, p_{17}=3, p_{21}=1.5$
3	40	1.5	9	7	2	$p_{11}=28, p_{12}=24, p_{13}=20, p_{14}=12$ $p_{15}=10, p_{16}=11, p_{17}=7, p_{21}=1, p_{22}=0.5$

Case 1 With regard to No. 1 in Tab. 1, we schedule these four jobs by the LPT rule and the optimal schedule rule, respectively and give an illustration of scheduling in Fig. 2.

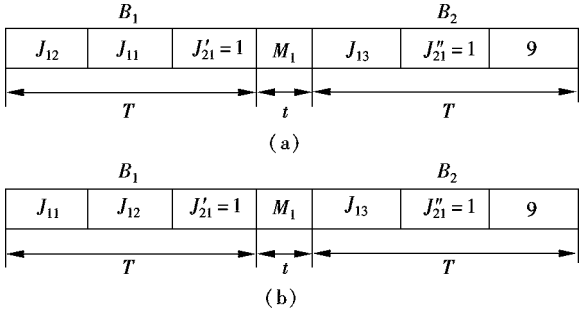

Fig. 2 Illustration of the considered problem in case 1. (a) LPT-LS algorithm; (b) Optimal schedule algorithm

Combining the data of the first line in Tab. 1 and Fig. 2, we can obtain Tab. 2.

Tab. 2 Scheduling result of case 1

S_1	n_{LPT}	$C_{\text{LPT-LS}}$	$C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$	n_{OPT}	C_{OPT}	S_2
19	2	20.5	1	2	20.5	2

From Tab. 2, it is obvious that

$$S_2 > C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$$

It means that S_2 is longer than the remaining time of the machine after assigning the non-resumable jobs by the LPT rule. It is easy to obtain that

$$\frac{C_{\text{LPT-LS}}}{C_{\text{OPT}}} = \frac{20.5}{20.5} = 1$$

It is in accordance with Theorem 1.

Case 2 With regard to No. 2 in Tab. 1, we schedule these eight jobs by the LPT rule and the optimal schedule rule, respectively and give an illustration of scheduling in Fig. 3.

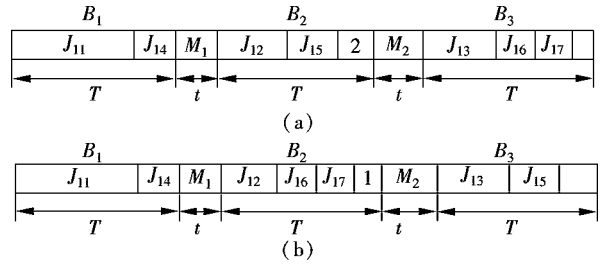

Fig. 3 Illustration of the considered problem in case 2. (a) LPT-LS alogrithm; (b) Optimal schedule algorithm

Combining the data of the second line in Tab. 1 and Fig. 3, we can obtain Tab. 3. We denote $\pi_{\text{L}} = C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$, $\pi_{\text{O}} = C_{\text{OPT}} - S_1 - (n_{\text{OPT}}-1)t$.

Tab. 3 Scheduling result of case 2

S_1	n_{LPT}	$C_{\text{LPT-LS}}$	π_{L}	n_{OPT}	C_{OPT}	π_{O}	S_2
55	3	58	2	3	57.5	1	1.5

From Tab. 3, it is obvious that

$$C_{\text{OPT}} - S_1 - (n_{\text{LPT}}-1)t < S_2 < C_{\text{LPT}} - S_1 - (n_{\text{LPT}}-1)t$$

It means that S_2 falls in between the remaining time of the machine by the LPT rule and the optimal schedule rule.

$$\frac{C_{\text{LPT-LS}}}{C_{\text{OPT}}} = \frac{C_{\text{LPT}}}{C_{\text{OPT}}} = \frac{58}{57.5} < 2$$

It is in accordance with Theorem 2.

Case 3 With regard to No. 3 in Tab. 1, we schedule these nine jobs by the LPT rule and the optimal schedule rule, respectively and give an illustration of scheduling in Fig. 4.

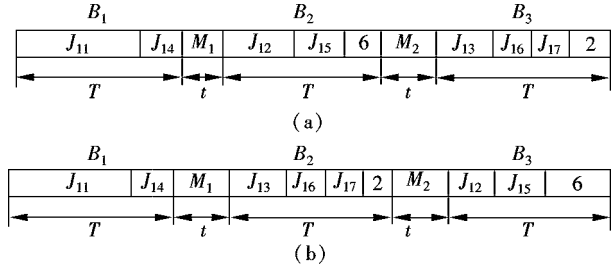

Fig. 4 Illustration of the considered problem in case 3. (a) LPT-LS algorithm; (b) Optimal schedule algorithm

Combining the data of the third line in Tab. 1 and Fig. 4, we can obtain Tab. 4.

Tab. 4 Scheduling result of case 3

S_1	n_{LPT}	$C_{\text{LPT-LS}}$	π_{L}	n_{OPT}	C_{OPT}	π_{O}	S_2
112	3	121	6	3	117	2	1.5

From Tab. 4, it is obvious that

$$S_2 < C_{\mathrm{OPT}} - S_1 - (n_{\mathrm{LPT}} - 1)t$$

It means that S_2 is less than the remaining time of the machine by the optimal schedule rule.

$$\frac{C_{\mathrm{LPT\text{-}LS}}}{C_{\mathrm{OPT}}} = \frac{121}{117} < 2$$

It is in accordance with Theorem 3.

According to the three numerical examples, the worst-case ratios of the LPT are in accordance with the results given in section 3.

4 Conclusion

For the problem $1/(nr-r) - pm/C_{\max}$, this paper proposes an LPT-LS algorithm and discusses the worst-case ratios of this algorithm in three different cases in terms of the value of the total processing time of the resumable jobs. In this research, we assume that the processing time is fixed and off-line. In future research, it is worth considering that the processing time is random and on-line. It is also worth researching the scheduling problem considering other objectives in parallel-machine systems.

References

[1] Tang O, Grubbstr O M R W, Zanoni S. Planned lead time determination in a make-to-order remanufacturing system [J]. *International Journal of Production Economics,* 2007, **108**(1/2): 426 - 435.

[2] Arts R H P M, Knapp G M, Mann Jr L. Some aspects of measuring maintenance in the process industry [J]. *Journal of Quality in Maintenance Engineering,* 1998, **4**(1): 6 - 11.

[3] Liao C J, Chen W J. Single-machine scheduling with periodic maintenance and nonresumable jobs [J]. *Computers and Operations Research,* 2003, **30**(9): 1335 - 1347.

[4] Lee C Y. Machine scheduling with an availability constraint [J]. *Journal of Global Optimization,* 1996, **9**(3/4): 395 - 416.

[5] Lee C Y, Liman S D. Single machine flow-time scheduling with scheduled maintenance [J]. *Acta Informatica,* 1992, **29**(4): 375 - 382.

[6] Graves G H, Lee C Y. Scheduling maintenance and semi-resumable jobs on a single machine [J]. *Naval Research Logistics,* 1999, **46**(7): 845 - 863.

[7] Ma Y, Yang S L, Chu C B. Minimizing makespan in semiresumable case of single-machine scheduling with an availability constraint [J]. *Systems Engineering—Theory & Practice*, 2009, **29**(4): 128 - 134.

[8] Chen W J. Minimizing number of tardy jobs on a single machine subject to periodic maintenance [J]. *Omega,* 2009, **37**(3): 591 - 599.

[9] Wu C C, Lee W C. A note on single-machine scheduling with learning effect and an availability constraint [J]. *The International Journal of Advanced Manufacturing Technology,* 2007, **33**(5/6): 540 - 544.

[10] Yang S J, Yang D L. Minimizing the makespan on single-machine scheduling with aging effect and variable maintenance activities [J]. *Omega,* 2010, **38**(6): 528 - 533.

[11] Ji M, He Y, Cheng T C E. Single-machine scheduling with periodic maintenance to minimize makespan [J]. *Computers & Operations Research,* 2007, **34**(6): 1764 - 1770.

[12] Graham R L, Lawler E L, Lenstra J K, et al. Optimization and approximation in deterministic sequencing and scheduling: a survey [J]. *Annals of Discrete Mathematics,* 1979, **5**: 287 - 326.

预防性周期维护下考虑可中断工件的再制造单机调度

刘碧玉[1,2] 陈伟达[1]

([1] 东南大学经济管理学院,南京 211189)

([2] 安徽工业大学管理科学与工程学院,马鞍山 243000)

摘要:研究预防性周期维护策略下再制造系统中可中断和不可中断 2 类工件的单机调度问题. 以最小化完工时间为目标,提出了 LPT-LS 算法,该算法首先按 LPT (longest processing time)规则安排不可中断工件,然后按 LS (list scheduling)规则安排可中断工件. 并根据可中断工件的总加工时间(记为 S_2)分 3 种情况证明了该算法的最坏情况比,结论如下:当 S_2 大于按 LPT 规则安排不可中断工件后机器的空闲时间时,最坏情况比为 1;当 S_2 介于分别按 LPT 规则和 OPT(最优排序)规则安排不可中断工件后机器的空闲时间之间时,最坏情况比小于 2;当 S_2 小于按 OPT 规则安排不可中断工件后机器的空闲时间时,最坏情况比小于 2. 最后通过算例验证了结论的正确性.

关键词:单机调度;预防性周期维护;可中断工件;LPT-LS 算法

中图分类号:F273