Available online at www.sciencedirect.com

SCIENCE DIRECT®

European Journal of Operational Research 165 (2005) 408–415

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

The two-machine flow-shop problem with weighted late work criterion and common due date

Jacek Błażewicz [a], Erwin Pesch [b], Małgorzata Sterna [a,*,1], Frank Werner [c]

[a] *Institute of Computing Science, Poznań University of Technology, Piotrowo 3A, 60-965 Poznań, Poland*
[b] *FB 5—Faculty of Economics, Institute of Information Systems, University of Siegen, Hölderlinstr. 3, 57068 Siegen, Germany*
[c] *Faculty of Mathematics, Otto-von-Guericke-University, PSF 4120, 39016 Magdeburg, Germany*

Received 1 October 2002; accepted 1 March 2003
Available online 28 May 2004

Abstract

The paper is on the two-machine non-preemptive flow-shop scheduling problem with a total weighted late work criterion and a common due date ($F2|d_i = d|Y_w$). The late work performance measure estimates the quality of the obtained solution with regard to the duration of late parts of tasks not taking into account the quantity of this delay. We prove the binary NP-hardness of the problem mentioned by showing a transformation from the partition problem to its decision counterpart. Then, a dynamic programming approach of pseudo-polynomial time complexity is formulated.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Scheduling; Flow-shop; Late work criteria; NP-hardness; Dynamic programming

1. Introduction

Rapid development of complex manufacturing systems and growing demand for an efficient production and project planning open a wide field of possible applications of the scheduling theory, cf. [3,7,13]. Trying to cover various realistic problems, besides proposing novel approaches and models, new parameters and criteria are also considered.

The paper describes a performance measure based on the amount of weighted late work in a system. The late work criterion takes into account the amount of work executed after predefined due dates ignoring the quantity of its delay. This objective function finds many practical applications e.g. in control systems during the process of data collecting [2,4]. Moreover, late work scheduling can support agriculture technologies [14,15,17] and the design of production or project execution plans within predefined time periods in manufacturing systems [17].

[*] Corresponding author. Tel.: +48-61-8528503x278; fax: +48-61-8771525.
E-mail addresses: blazewic@sol.put.poznan.pl (J. Błażewicz), pesch@fb5.uni-siegen.de (E. Pesch), malgorzata.sterna@cs.put.poznan.pl (M. Sterna), frank.werner@mathematik.uni-magdeburg.de (F. Werner).
[1] The author has been awarded the Annual Stipend for Young Scientists of the Foundation for Polish Science.

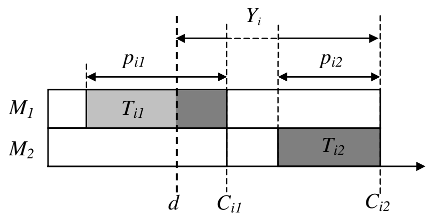

Fig. 1. An example of the non-preemptive late work definition for the two-machine shop case.

The late work objective function was first proposed in the context of parallel machines [2,4] and then applied to the one-machine scheduling problem [14,15]. Moreover, some general complexity results were obtained [5,6,17] which allow one to consider problems with the late work criterion as more complicated than the analogous problems with the maximum lateness objective function. The late work performance measure has been recently applied to the shop environment, especially to open-shop systems [6,17]. The idea of this performance measure initiated also a different research stream dedicated to an imprecise computation model, where particular tasks are divided into optional and mandatory parts, cf. [1,8,10,16]. This approach finds many practical applications in the hard real time scheduling problems arising e.g. in the aviation or flight control.

In this paper, we consider a non-preemptive scheduling problem with the total weighted late work criterion and a common due date in a two-machine flow-shop environment, which yet has not been investigated for this performance measure. A thorough analysis shows that the problem mentioned is binary NP-hard.

Let us set up the subject more formally. The late work performance measure estimates the quality of obtained solutions with regard to the amount of late parts of jobs not taking into account the quantity of the delay of fully late ones [2].

In the two-machine shop environment considered, where pre-emption is not allowed, the late work Y_i for job J_i has to be calculated by summing up late parts of tasks T_{i1} and T_{i2}, executed after the common due date d, on machines M_1 and M_2, respectively. Denoting as p_{i1}, p_{i2} the processing times of tasks T_{i1}, T_{i2} and as C_{i1}, C_{i2} the completion times of those tasks, the late work for job J_i is given by

$$Y_i = \sum_{\substack{T_{ij} \\ j=1,2}} \min\{\max\{0, C_{ij} - d\}, p_{ij}\}$$

(cf. Fig. 1). Summing up late work for all jobs J_i, $i = 1, \ldots, n$, and taking into account their given weights w_i, we calculate the total weighted late work criterion value as $Y_w = \sum_{i=1}^{n} w_i Y_i$.

The organisation of the work is as follows. In Section 2, we present an NP-hardness proof for problem $F2|d_i = d|Y_w$. Then, in Section 3, we propose a dynamic programming approach solving it. Both results allow us to classify the problem as NP-hard in the ordinary sense. Section 4 concludes the paper.

2. NP-hardness proof for problem $F2|d_i = d|Y_w$

We prove that the weighted case of the two-machine non-preemptive flow-shop scheduling problem with a common due date is binary NP-hard. We construct a proof for its decision counterpart by transforming the *partition problem* formulated as follows [9].

Definition 1. Let a finite set A be given and a positive integer size $s(a_i)$ for each element $a_i \in A$. The decision version of the partition problem is: Does there exist a subset $A' \subseteq A$ such that $\sum_{a_i \in A'} s(a_i) = \sum_{a_i \in A \setminus A'} s(a_i)$?

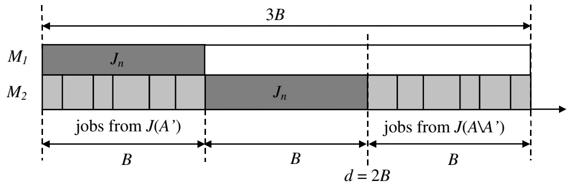

Fig. 2. The schedule corresponding to a solution of the partition problem.

Theorem 1. *The decision counterpart of problem $F2|d_i = d|Y_w$ is NP-complete.*

Proof. For a given instance of the partition problem, we construct an instance of problem $F2|d_i = d|Y_w$, as follows:

$$n = |A| + 1, \quad d = 2B = \sum_{a_i \in A} s(a_i),$$

$$p_{i1} = 0, \quad p_{i2} = s(a_i), \quad w_i = 1 \quad \text{for } J_i \in J(A),$$

$$p_{n1} = B, \quad p_{n2} = B, \quad w_n = B^2.$$

Thus, we have a set of n jobs $J = J(A) \cup \{J_n\}$, where the set $J(A)$ contains $n - 1$ jobs corresponding to the elements of the set A from the partition problem and J_n is an additional job with a very big weight. We will show that the partition problem has a solution if and only if there exists a schedule in problem $F2|d_i = d|Y_w$ of a criterion value not exceeding B.

If the partition problem has a solution, then the set A can be divided into two subsets A' and $A \setminus A'$ such that $\sum_{a_i \in A'} s(a_i) = \sum_{a_i \in A \setminus A'} s(a_i) = B$. Denoting with $\Pi_k(\mathcal{J})$ an arbitrary sequence of jobs from the set $\mathcal{J}$ on a machine M_k, the corresponding solution of the scheduling problem is constructed as follows: $\Pi_1(\mathcal{J}) = (J_n)$, $\Pi_2(\mathcal{J}) = (\Pi_2(J(A')), J_n, \Pi_2(J(A \setminus A')))$ (cf. Fig. 2). On machine M_1 only job J_n is executed, while on M_2 jobs corresponding to the elements of A' precede J_n, and J_n is succeeded by jobs corresponding to the elements of $A \setminus A'$.

Due to the construction of the presented schedule, it is optimal with regard to the total weighted late work because the only job with a big weight (J_n) is executed completely before the due date and the remaining ones are performed on machine M_2 without idle times between the tasks. The amount of late work is equal to B and all the late jobs from $J(A \setminus A')$, corresponding to the elements of the subset $A \setminus A'$, have a unary weight. Hence, the criterion value equals $\sum_{i=1}^{n} w_i Y_i = B$ and there exists a solution of the scheduling problem with criterion value not higher than B.

If problem $F2|d_i = d|Y_w$ has a solution, then there exists a schedule with criterion value not higher than $\sum_{i=1}^{n} w_i Y_w = B$. Taking into account the fact that all problem parameters are integers, the smallest possible portion of a task which can be late is equal to one unit. Each late unit of job J_n would increase the criterion value of $w_n = B^2 > B$. Hence, job J_n must be processed early in any optimal solution. Because J_n occupies each machine for B time units and it is first executed on M_1 and then on M_2, we have a free gap in the time period $[0, B]$ on machine M_2. This gap of length B must be completely filled with tasks in order not to exceed B units of the total weighted late work. Otherwise, idle time occurs before J_n on M_2 and more than B units of work have to be executed after J_n on this machine, i.e. after the due date d, which would make the criterion value bigger than B. The mentioned partition of jobs completed before and after J_n on M_2 defines the solution of the partition problem. □

3. Dynamic programming approach for problem $F2|d_i = d|Y_w$

We can state that the analysed problem $F2|d_i = d|Y_w$ is NP-hard in the ordinary sense [9] because it can be solved by a dynamic programming approach of pseudo-polynomial time complexity presented below. The method considered follows the approach proposed by Józefowska et al. [12] for problem $F2|d_i = d|U_w$, where U_w denotes the weighted number of late jobs. However, the dynamic programming method presented is extended with additional ideas necessary to cover the peculiarities of the problem analysed.

First, we observe that, for the search of an optimal solution of the problem, all early jobs can be scheduled in Johnson's order (cf. [5,12,17]). Johnson's rule [11] states that all jobs J_i with $p_{i1} \leqslant p_{i2}$ are sequenced in non-decreasing order of p_{i1}, while the rest, with $p_{i1} > p_{i2}$, in non-increasing order of p_{i2}. This rule is optimal from the schedule length point of view. Thus, applying it to a selected subset of early jobs ensures the shortest partial schedule length before a due date, the maximal machine utilisation and, consequently, the optimal value of the total weighted late work criterion. Consequently, a proper selection of the first late job and, then, early jobs are crucial elements in the solution process. Thus, after determining the best first late job, to obtain an optimal solution, we select some early jobs and schedule them in Johnson's order before the first late one. Moreover, we use the fact that minimising the total weighted late work is equivalent to maximising the total weighted early work.

In our approach, we consider each job as the first late job and settle remaining decisions, optimal for this selection, by a dynamic programming method based on a recurrence function. For the sake of clarity, we denote with $\widehat{J}_n$ the job selected as the first late job and number the remaining jobs from $\widehat{J}_1$ to $\widehat{J}_{n-1}$ in Johnson's order.

For the selected first late job $\widehat{J}_n$, we calculate initial conditions $f_n(A, B, t, a)$, while for the remaining jobs $\widehat{J}_{n-1}, \dots, \widehat{J}_1$ a recurrence function $f_k(A, B, t, a)$ is determined. The function $f_k(A, B, t, a)$ denotes the maximum amount of early work of jobs $\widehat{J}_k, \widehat{J}_{k+1}, \dots, \widehat{J}_n$ assuming that $\widehat{J}_n$ is the first late job, the first job among $\widehat{J}_k, \widehat{J}_{k+1}, \dots, \widehat{J}_n$ starts on machine M_1 exactly at time A and not earlier than at time B on M_2. Moreover, exactly t time units are reserved for executing jobs $\widehat{J}_1$ to $\widehat{J}_{k-1}$ after $\widehat{J}_n$ on M_1 before the common due date d. Finally, we assume that no job ($a = 0$) or exactly one job ($a = 1$) among $\widehat{J}_1$ to $\widehat{J}_{k-1}$ is partially executed on machine M_1 after $\widehat{J}_n$ before d. In consequence, $f_1(0, 0, 0, 0)$ denotes the total weighted early work in the system under the assumption that $\widehat{J}_n$ is the first late job in the schedule.

In Section 3.1 we define the initial conditions, while in Section 3.2 the recurrence relations are presented. Section 3.3 provides a general framework of the dynamic programming approach proposed together with the complexity analysis.

3.1. Initial conditions

To find an optimal solution of the problem analysed, we have to consider each job as the first late job $\widehat{J}_n$. The initial conditions $f_n(A, B, t, a)$, determining the weighted early work in the system obtained by executing $\widehat{J}_n$ as the first late job, can be formulated as follows:

if $[(d - p_{n2} < B \leqslant d$ and $A \leqslant d - p_{n1})$ or $(d - p_{n1} - p_{n2} < A < d - p_{n1}$ and $B \leqslant d)]$ and $0 \leqslant t \leqslant d - p_{n1} - A$ and $a \in \{0, 1\}$, then

$$f_n(A, B, t, a) = w_n p_{n1} + w_n(d - \max\{A + p_{n1}, B\}); \tag{1}$$

if $d - p_{n1} < A \leqslant d$ and $B \leqslant d$ and $t = 0$ and $a = 0$, then

$$f_n(A, B, t, a) = w_n(d - A); \tag{2}$$

if otherwise, then

$$f_n(A, B, t, a) = -\infty. \tag{3}$$

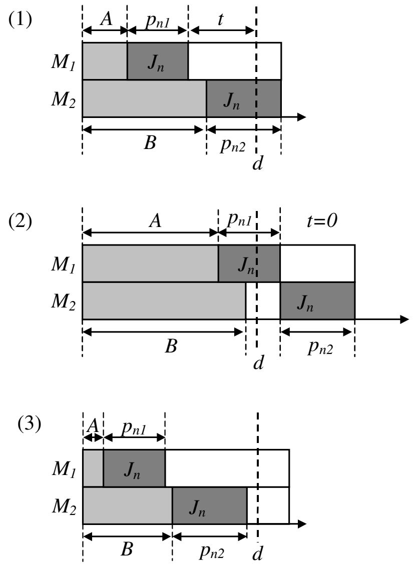

Fig. 3. The case study of the initial conditions for arbitrary parameter values.

In Term 1, we assume that the first late job $\widehat{J}_n$ is early on M_1 and partially early on M_2 (cf. Fig. 3(1)). In this case, there still might be a job among $\widehat{J}_{n-1},\dots,\widehat{J}_1$ which is partially early on M_1 (processed after $\widehat{J}_n$ but before d), so the condition of Term 1 is valid for $a \in \{0,1\}$. Case 2 concerns the situation when job $\widehat{J}_n$ is only partially early on M_1 and its second operation is late (cf. Fig. 3(2)). Consequently, no other job can be partially early on M_1 and the variable a has to be equal to 0 in the condition of Term 2. All other cases are infeasible (Term 3), because job $\widehat{J}_n$ would not be the first late one (e.g. Fig. 3(3)). Thus, we set the function value representing the amount of the total weighted early work to minus infinity.

We determine the initial conditions presented above for all possible values of A, B, $t (0 \leqslant A, B, t \leqslant d)$ and $a \in \{0,1\}$.

3.2. Recurrence relations

After determining the initial conditions values for a certain selection of the first late job $\widehat{J}_n$, we consider all remaining jobs $\widehat{J}_k$ in reverse Johnson's order, for $k = n-1,\dots,1$ calculating the following recurrence relations:

if $A + p_{k1} \leqslant d$ and $\max\{A + p_{k1}, B\} + p_{k2} \leqslant d$, then

$$\begin{aligned} f_k(A,B,t,a) = \max\{ & f_{k+1}(A+p_{k1}, \max\{A+p_{k1},B\}+p_{k2},t,a) + w_k(p_{k1}+p_{k2}), && (4)\\ & f_{k+1}(A,B,t,a), && (5)\\ & f_{k+1}(A,B,t+p_{k1},a) + w_k p_{k1}, && (6)\\ & \max\{f_{k+1}(A,B,t+T,1) + w_k T : 1 \leqslant T < p_{k1} \text{and } t+T \leqslant d\} \quad \text{for } a = 0\}; && (7) \end{aligned}$$

if $A + p_{k1} > d$ or $\max\{A + p_{k1}, B\} + p_{k2} > d$, then

$$\begin{aligned} f_k(A,B,t,a) = \max\{ & f_{k+1}(A,B,t,a), && (8)\\ & f_{k+1}(A,B,t+p_{k1},a) + w_k p_{k1}, && (9)\\ & \max\{f_{k+1}(A,B,t+T,1) + w_k T : 1 \leqslant T < p_{k1} \text{and } t+T \leqslant d\} \quad \text{for } a = 0\}; && (10) \end{aligned}$$

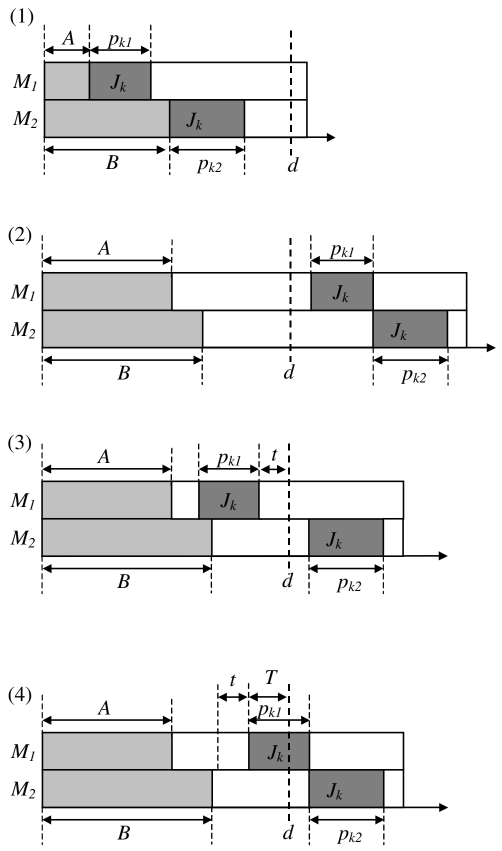

Fig. 4. The case study of the recurrence function for arbitrary parameter values.

if otherwise, then

$$f_k(A, B, t, a) = -\infty. \tag{11}$$

The first group of Terms 4–7 concerns the cases when job $\widehat{J}_k$ can be executed totally early. The particular situation, when $\widehat{J}_k$ is scheduled totally early, is analysed in Term 4, depicted in Fig. 4(1). In Terms 8–10 we consider the cases, when $\widehat{J}_k$ must be late because the gap in a partial schedule where this job must be placed according to Johnson's order is too small. We take into account the situations when $\widehat{J}_k$ is totally late (Terms 5 and 8, Fig. 4(2)), when it is early only on M_1 (Terms 6 and 9, Fig. 4(3)) and, finally, when it is only partially early on M_1 (Terms 7 and 10, Fig. 4(4)).

In both cases, when $\widehat{J}_k$ can and cannot be totally early, we choose the best variant of scheduling $\widehat{J}_k$ among all possible ones ensuring the optimality of the partial schedule obtained.

All other schedules (i.e. all other parameters values, e.g. such that job $\widehat{J}_k$ is partially early on M_2 being, in this way, the first late job), are infeasible, which is taken into account in Term 11.

It is worth to be mentioned, that Terms 7 and 10 are calculated only if the variable a equals to 0. That means, that because $\widehat{J}_k$ is partially early on M_1, no other job among the remaining ones $\widehat{J}_{k-1}, \ldots, \widehat{J}_1$ can be partially early on this machine. Moreover, the variable a equals 1 in the recurrence relation $f_{k+1}(A, B, t + T, 1)$ called in those terms. That means, that from the point of view of jobs $\widehat{J}_{k+1}, \ldots, \widehat{J}_n$ (already considered in the dynamic programming calculations), there is a job with a smaller index value (i.e $\widehat{J}_k$), whose first operation is partially early on M_1.

Similarly as in the case of the initial conditions, we determine the recurrence relations presented above for all jobs $\widehat{J}_k$, $k = n-1, \ldots, 1$, and all possible values of A, B, t $(0 \leqslant A, B, t \leqslant d)$ and $a \in \{0, 1\}$. The maximal value of the total weighted early work, which can be obtained in the flow-shop system assuming that $\widehat{J}_n$ is the first late job is given by $f_1(0,0,0,0)$.

3.3. Dynamic programming method

To find an optimal solution of the problem analysed, we have to consider each job J_i as the first late job $\widehat{J}_n$ and determine the total weighted early work $F_i = f_1(0,0,0,0)$ corresponding to this selection. Job J_i for which F_i takes the maximal value is the optimal first late job J_*. An optimal schedule is constructed based on the dynamic programming calculations performed for this first late job J_*. We schedule all jobs selected as early ones before J_* in Johnson's sequence. Then, if it is the case, we place jobs assigned to the interval between the end of J_* on M_1 and d, and, finally, we perform fully late jobs in an arbitrary order after d. The general framework of the dynamic programming approach proposed is given below.

```
$J = \{J_1, \ldots, J_n\}$;
for $i = 1$ to $n$ do
begin
    set $\widehat{J} = J \setminus \{J_i\}$;
    set $\widehat{J}_n = J_i$ as the first late job;
    renumber jobs from $\widehat{J}$ in Johnson's order as $\widehat{J}_1, \ldots, \widehat{J}_{n-1}$;
    calculate initial conditions $f_n(A,B,t,a)$ for $0 \leqslant A,B,t \leqslant d$ and $a \in \{0,1\}$;
    for $k = n-1$ to 1 do
        calculate recurrence relations $f_k(A,B,t,a)$ for $0 \leqslant A,B,t \leqslant d$ and $a \in \{0,1\}$;
    set $F_i = f_1(0,0,0,0)$ as the total weighted early work subject to the first late job $\widehat{J}_n$ (i.e. $J_i$)
end;
set $F^* = \max_{i=1,\ldots,n}\{F_i\}$ as the optimal total weighted early work;
set $J_*$ to be a job with $F_i = F^*$;
based on dynamic programming results for the first late job $J_*$ determine:
    $J^E$—the set of early jobs,
    $J^P$—the set of jobs performed between $J_*$ and $d$ on $M_1$,
    $J^L$—the set of late jobs;
construct an optimal schedule by:
    executing jobs from $J^E$ in Johnson's order followed by $J_*$,
    performing the first tasks of jobs from $J^P$ after $J_*$ before $d$ on $M_1$ (if $J^P$ contains a job partially early on
    $M_1$, then it is executed as the last one from set $J^P$; the sequence of remaining jobs from $J^P$ is arbitrary),
    executing jobs from $J^L$ after $d$ in an arbitrary order.
```

In the presented approach, the calculation of $f_k(A,B,t,a)$ for any job $\widehat{J}_k$ takes time bounded by p_{k1}, where it is sensible to consider only jobs with $p_{k1} = O(d)$. Because the value $f_k(A,B,t,a)$ is determined for all possible values $A,B,t (0 \leqslant A,B,t \leqslant d)$ and $a \in \{0,1\}$, calculations for a particular selection of the first late job takes $O(nd^4)$ time. Thus, the analysis of all possible selections of the first late jobs requires $O(n^2d^4)$ time. Choosing the best value among $f_1(0,0,0,0)$ corresponding to different first late jobs $(O(n))$ and the construction of an optimal schedule $(O(n \log n)$, cf. [11]) do not change the overall complexity of the approach. Thus, the dynamic programming method proposed has a pseudo-polynomial time complexity and, consequently, problem $F2|d_i = d|Y_w$ can be classified as binary NP-hard.

4. Conclusions

The paper presents some results of the research on the flow-shop scheduling problem with the total weighted late work criterion, which yet has not been investigated in this machine environment. The possibility of a practical application of this performance measure in manufacturing systems modelled as the shop ones makes this research field especially interesting (cf. [17]).

We have proved the binary NP-hardness of problem $F2|d_i = d|Y_w$ showing the transformation from the partition problem to this scheduling case. Moreover, we have proposed a dynamic programming approach of pseudo-polynomial time complexity proving in this way the ordinary NP-hardness of this problem. Thus the flow-shop case analysed has the same complexity status as the analogous scheduling problem in open-shop environment $O2|d_i = d|Y_w$ [6]. Finally, owing to the fact that the flow-shop problem is a special case of the job-shop one, where all jobs have exactly the same sequence of the execution on the machines, we can state that problem $J2|d_i = d|Y_w$ is also NP-hard.

References

[1] R. Bettati, D. Gillies, C.C. Han, K.J. Lin, C.L. Liu, J.W.S. Liu, W.K. Shih, Recent results in real-time scheduling, in: A.M. van Tilborg, G.M. Koob (Eds.), Foundations of Real-Time Computing: Scheduling and Resource Management, Kluwer Academic Publishers, Boston, 1991, pp. 129–156.

[2] J. Błażewicz, Scheduling preemptive tasks on parallel processors with information loss, Recherche Technique et Science Informatiques 3 (6) (1984) 415–420.

[3] J. Błażewicz, K. Ecker, E. Pesch, G. Schmidt, J. Węglarz, Scheduling Computer and Manufacturing Processes, second ed., Springer, Berlin, Heidelberg, New York, 2001.

[4] J. Błażewicz, G. Finke, Minimizing mean weighted execution time loss on identical and uniform processors, Information Processing Letters 24 (1987) 259–263.

[5] J. Błażewicz, E. Pesch, M. Sterna, F. Werner, Total late work criteria for shop scheduling problems, in: K. Inderfurth, G. Schwödiauer, W. Domschke, F. Juhnke, P. Kleinschmidt, G. Wäscher (Eds.), Operations Research Proceedings 1999, Springer, Berlin, 2000, pp. 354–359.

[6] J. Błażewicz, E. Pesch, M. Sterna, F. Werner, Open shop scheduling problems with late work criteria, Discrete Applied Mathematics 134 (2004) 1–24.

[7] P. Brucker, Scheduling Algorithms, second ed., Springer, Berlin, Heidelberg, New York, 1998.

[8] J.Y. Chung, W.K. Shih, J.W.S. Liu, D.W. Gillies, Scheduling imprecise computations to minimize total error, Microprocessing and Microprogramming 27 (1989) 767–774.

[9] M.R. Garey, D.S. Johnson, Computers and Intractability, W.H. Freeman and Co., San Francisco, 1979.

[10] K. Ho, J.Y.T. Leung, W.D. Wei, Minimizing maximum weighted error for imprecise computation tasks, Journal of Algorithms 16 (1994) 431–452.

[11] S.M. Johnson, Optimal two- and three-stage production schedules, Naval Research Logistics Quarterly 1 (1954) 61–68.

[12] J. Józefowska, B. Jurisch, W. Kubiak, Scheduling shops to minimize the weighted number of late jobs, Operation Research Letters 16 (5) (1994) 277–283.

[13] M. Pinedo, X. Chao, Operation Scheduling with Applications in Manufacturing and Services, Irwin/McGraw-Hill, Boston, 1999.

[14] C.N. Potts, L.N. Van Wassenhove, Single machine scheduling to minimize total late work, Operations Research 40/3 (1991) 586–595.

[15] C.N. Potts, L.N. Van Wassenhove, Approximation algorithms for scheduling a single machine to minimize total late work, Operations Research Letters 11 (1991) 261–266.

[16] W.K. Shih, J.W.S. Liu, J.Y. Chung, Algorithms for scheduling imprecise computations with timing constraints, SIAM Journal of Computing 20 (1991) 537–552.

[17] M. Sterna, Problems and Algorithms in Non-Classical Shop Scheduling, Scientific Publishers of the Polish Academy of Sciences, Poznań, 2000.