Revenue Management in a Job-Shop:
A Dynamic Programming Approach

Jacek Błażewicz

Institute of Computing Science

Poznań University of Technology

Piotrowo 3A, 60-965 Poznań, Poland

e-mail: blazewic@sol.put.poznan.pl

phone: +48 61 790790

Erwin Pesch

Institute of Information Systems, FB 5 - Faculty of Economics

University of Siegen

Hoelderlinstrasse 3, 57068 Siegen, Germany

e-mail: pesch@fb5.uni-siegen.de

phone: +49 271 7402420

Małgorzata Sterna

Institute of Computing Science

Poznań University of Technology

Piotrowo 3A, 60-965 Poznań, Poland

e-mail: Malgorzata.Sterna@cs.put.poznan.pl

phone: +48 61 8528503 ext.278

Frank Werner

Faculty of Mathematics

Otto-von-Guericke-University

PSF 4120, 39016 Magdeburg, Germany

email: frank.werner@mathematik.uni-magdeburg.de

phone: +48 391 6712025

Abstract

The paper concerns the revenue management problem where a set of customer demands of expected size contribute to the company's revenue in an expected amount that is a function of the demand size and importance of the particular customer. To maximize the company's revenue we are looking for an allocation of the limited resource capacity over a certain time horizon in order to produce the customer's demand within its given due date. The problem mentioned can be modeled as the job shop scheduling problem with the weighted late work criteria.

Within the paper, we propose a dynamic programming method, which can be used to solve the problem under consideration optimally. Moreover, we determine the complexity status of the case analyzed as binary NP-hard.

Keywords: revenue management, job-shop scheduling, late work criteria, dynamic programming

Introduction

Revenue management is essentially the process of allocating resources to the right customer at the right time and the right price. The focus is on maximizing profit or revenue and it has led to an increased profit in a variety of industries. Still, some of the most important applications are in the airline business, i.e. where the basis question arises whether or not to accept or reject a booking request (for a flight) within a specific booking class at a given fare (cf. McGill and van Ryzin (1999), Wirtz et al. (2002)). In this paper, we are faced the situation where a set of customer demands of expected size contribute to the company's revenue in an expected amount that is a function of the demand size and importance of the particular customer. The company is supposed to answer the question how the limited resource capacity over a certain time horizon should be allocated in order to produce the customer's demand within its given due date in a way that maximizes the company's revenue. This leads to the question which customer orders are to be accepted and which orders should be rejected, i.e. the expected revenue of the latter will be lost because their production cannot be finished before the due date. Thus, rejected orders are late work which could only be produced after the due date.

Due date involving criteria are performance measures often used in practical applications (cf. Błażewicz et al. (2001), Brucker (1998), Pinedo and Chao (1999)). Generally, they represent the customer point of view allowing to minimize the delay of orders realized in a system. Classical objective functions of this type, such as maximum lateness or total tardiness are calculated with regard to the quantity of the delay, while the late work criterion allows for minimizing the amount of work executed after given due dates.

The late work objective function has not been widely investigated, although it finds many practical applications, e.g. in data collecting in control systems (Błażewicz (1984), Błażewicz and Finke (1987)), supporting agriculture technologies (Błażewicz et al. (2000, 2003a, 2003b), Sterna (2000)) or designing production plans within predefined time periods in manufacturing systems (Sterna (2000)) or recently in revenue management (McGill and van Ryzin (1999), Wirtz et al. (2002)).

The late work criteria was proposed in the context of parallel machines (Błażewicz (1984), Błażewicz and Finke (1987)) and then applied to the one-machine scheduling problem (Potts and Van Wassenhove (1991a, 1991b). More recently, some general complexity results were obtained (Błażewicz et al. (2000, 2003a)), which allow one to consider problems with the late work criterion as more complicated than analogous problems with the maximum lateness objective function. Then, the late work performance measure has been investigated in the dedicated machine environment (Błażewicz et al. (2003a, 2003b), Sterna (2000)).

In the paper, we consider a non-preemptive scheduling problem with the total weighted late work criterion and a common due date in the two-machine job-shop environment. The problem basically arises for a medium sized manufacturer (approximately 1500 employees) producing parts (e.g. power brake units, booster, etc.) on demand for most of the existing automobile companies in Europe, America, and Asia.

A set of jobs (customer orders, e.g. 30000 booster of a particular type) consists of two tasks that have to be processed on two machines M_1 and M_2 in a predefined order (for each job). The tasks' processing times p_{i1} and p_{i2} reflect the expected order size for the two different machines. A machine can process only one job at a time and a job cannot be handled on both machines simultaneously. Within our earlier research, we have shown that analogous problems in open-shop (Błażewicz et al. 2003a) and flow-shop systems (Błażewicz et al. 2003b) are binary *NP*-hard. With regard to the hardness of the flow-shop problem, the job-shop one (being its generalization) is also computationally hard (Garey and Johnson (1979)).

Here, we propose a pseudopolynomial time dynamic programming method solving the problem considered. This approach is substantially different to the much simpler approach in case of a flow-shop. That allows us to classify this case as binary *NP*-hard and to finish the research on two-machine weighted shop scheduling with a common due date.

As we have mentioned, 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. In the two-machine job shop environment without preemptions, the late work Y_i for job $J_i \in J$ is determined as the sum of late parts of tasks T_{i1} and T_{i2}, executed after a 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} their completion times, the late work for job J_i is given by (cf. Figure 1): $Y_i = \sum_{j=1,2} \min\{\max\{0, C_{ij} - d\}, p_{ij}\}$.

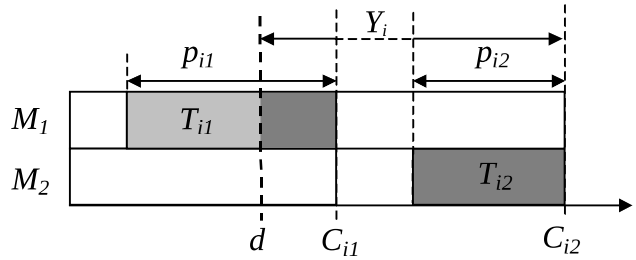

Figure 1. Non-preemptive late work for the two-machine job-shop

To determine the total weighted late work we consider the expected revenue losses in the system, i.e. we sum up late work for all jobs (where $n = |J|$) taking into account their given weights (customer importance) w_i, i.e.: $Y_w = \sum_{i=1}^{n} w_i Y_i$.

1. Dynamic Programming Approach

The dynamic programming approach presented in this paper was inspired by methods designed for cases with the weighted number of late jobs as an objective function (Józefowska, Jurisch and Kubiak (1994)).

Let the set of jobs J be partitioned into two subsets J^1 and J^2 containing all jobs with the first (or only) task processed on machine M_1 and M_2, respectively. We can assume that early jobs are processed in Jackson's order (Jackson (1956)). Jackson's rule states that jobs from J^1 proceed J^2 on M_1, while on M_2 jobs from J^2 are executed before J^1 (for both sets jobs containing only one task are performed as the last ones). Sets J^1, J^2 are scheduled according to Johnsons's rule (Johnson (1954)), so within sets J^1 and J^2 all jobs J_i with $p_{i1} \leq p_{i2}$ are sequenced in non-decreasing order of p_{i1}, while the rest, with $p_{i1} > p_{i2}$, is scheduled in non-increasing order of p_{i2}. Jackson's order is optimal from the schedule length point of view. Thus, scheduling early jobs in this sequence, we obtain the shortest subschedule of early jobs, the maximum machine utilization and, in consequence, the maximum amount of the weighted early work for those jobs. Similarly as for the flow shop problem (Błażewicz et al. (2003b)), we use the fact that maximizing the total weighted early work is equivalent to minimizing the total weighted late work, which is the criterion under consideration.

Based on the above observation, that all early jobs have to be scheduled in Jackson's order, for any subset of early jobs $J' \subseteq J^1 \cup J^2$ in an optimal solution, we can assume that jobs from $J^1 \cap J'$ precede jobs from $J^2 \cap J'$ on M_1, and oppositely jobs from $J^2 \cap J'$ precede jobs from $J^1 \cap J'$ on M_2. Moreover, we can assume that the first job of both sets $J^1 \cap J'$ and $J^2 \cap J'$ starts at time zero on machines M_1 and M_2, respectively.

Because of *NP*-hardness of the considered problem, we have to check all possible schedules to determine an optimal one. The search in the solution space is performed in a systematic way according to a dynamic programming approach. Taking into account all properties of the problem, we can restrict significantly the set of solutions explicitly analyzed.

Actually, there are only three possible schemes of an optimal solution, which have to be compared. They reflect the different situations when there are two jobs, one or no job with tasks executed partially late on machines M_1 or M_2 (cf. Figure 2). Denoting with J^P a set of jobs with partially late tasks, we have to consider:

- $J^P = \{J_a, J_b\}$, i.e. there are on both machines partially late tasks belonging to two different jobs, where J_a denotes a job partially late on M_1 and J_b is a job partially late on M_2,
- $J^P = \{J_x\}$, i.e. there is one partially late task, either on M_1 or on M_2, belonging to job J_x,
- $J^P = \emptyset$, i.e. there is no partially late task in a system.

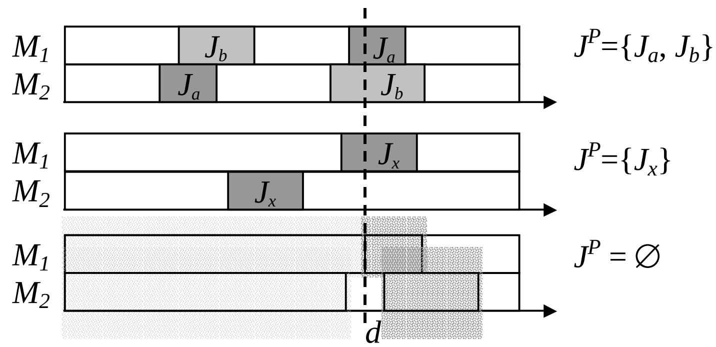

Figure 2. Possible sets of jobs with partially late tasks - J^P

For a particular set J^P, we renumber the remaining jobs $J\backslash J^P$ in Jackson's order obtaining the sequence $\hat{J} = (\hat{J}_1, \ldots, \hat{J}_u, \hat{J}_{u+1}, \ldots, \hat{J}_{\tilde{n}})$, where $\hat{J}_1, \ldots, \hat{J}_u \in J^1\backslash J^P$, $\hat{J}_{u+1}, \ldots, \hat{J}_{\tilde{n}} \in J^2\backslash J^P$, u denotes the number of jobs with the first (only) task on M_1 and $\tilde{n}$ denotes the number of jobs to be scheduled (besides J^P). To find an optimal order of jobs subject to set J^P, we have to choose an optimal variant of scheduling particular jobs $\hat{J}_k \in J\backslash J^P$ (i.e. $\hat{J}_k \in \hat{J}$). Job $\hat{J}_k$ may be executed early, totally late or early on its first machine and totally late on the second one (cf. Figure 3). No task of job $\hat{J}_k \in J\backslash J^P$ can be performed partially late, because, in this case, $\hat{J}_k$ would have to be an element of J^P.

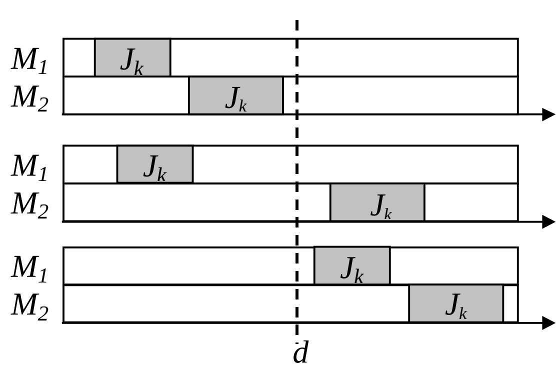

Figure 3. Possible ways of scheduling jobs $\hat{J}_k \in J\backslash J^P$

To find an optimal solution of the problem, we have to analyze all possible sets of jobs with partially late tasks J^P. For a particular set J^P, we calculate the initial conditions ($f_{\tilde{n}+1}$) determining the amount of weighted early work corresponding to this set. Then, we consider the remaining jobs $\hat{J}_k \in \hat{J}$ calculating for them the recurrence relations (f_k) denoting the amount of the weighted early work obtained for set $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\} \cup J^P$. First, we analyze all jobs with the first (only) task executed on machine M_2, i.e. $k = \tilde{n}, \ldots, u+1$, and then, using slightly different recurrence relations, all jobs with the first (only) task executed on machine M_1, i.e. $k = u, \ldots, 1$. The last jobs in Jackson's order in both sets, i.e. $\hat{J}_{\tilde{n}}$ and $\hat{J}_u$, are treated in a special way. The value obtained for the first job $\hat{J}_1$ (f_1) denotes the weighted early work for all jobs $\{\hat{J}_1, \ldots, \hat{J}_{\tilde{n}}\} \cup J^P$ subject to set J^P. After analyzing all possible sets J^P, we determine the optimal weighted early work for the problem under consideration. Then, restoring decisions taken during dynamic programming calculations for an optimal set J^P, we schedule optimally particular tasks from $J \backslash J^P$. All early jobs have to be executed before a common due date in Jackson's order, while the remaining jobs are performed between those early ones and J^P in an arbitrary order. Figure 4 depicts a general scheme of an optimal solution (for particular instances of the problem some subschedules might be empty).

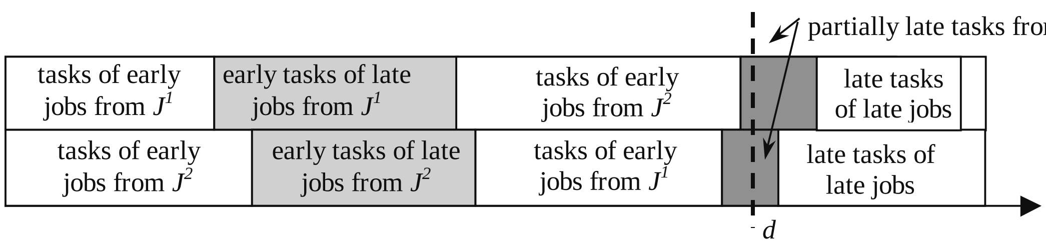

Figure 4. General scheme of an optimal solution of problem $J2 \mid d_i = d \mid Y_w$

The dynamic programming approach sketched above is presented in detail in the following sections. The initial conditions are presented in Section 2, while Section 3 contains the formulation of the recurrence relations. In Section 4, a meta-code of the DP approach is provided together with a short complexity analysis.

2. Initial Conditions

To find an optimal solution of problem $J2 \mid d_i = d \mid Y_w$, we have to analyze all possible sets of jobs with partially late tasks – J^P. The weighted early work corresponding to this set is determined by the initial conditions defined as $f_{\tilde{n}+1}$ (A, t_1, L_1, B, t_2, L_2), where $\tilde{n} = |J \backslash J^P|$. Function $f_{\tilde{n}+1}$ denotes the maximum amount of the weighted early work provided that the early tasks of jobs from J^P (if any) start exactly at time A on M_1, and exactly at time B on M_2. If there is no early task on machine M_1, M_2, then the execution of the partially late task on this machine may start at any time after A, B, respectively. There are exactly t_1, t_2 units of early tasks and exactly L_1, L_2 units of partially late tasks on machines M_1 and M_2, respectively.

As we have mentioned there are 3 possible cases, when set J^P contains two jobs, one or no job. Now, we present formulations of the initial condition for those cases.

Case I.1 $J^P = \{J_a, J_b\}$

Let set J^P contain two jobs, i.e. $J^P = \{J_a, J_b\}$. Depending on the job type, we have to analyze four subcases:

Case I.1.1 $J_a \in J^1,\ J_b \in J^1$ (cf. Figure 5.1)

if $0 \le A \le \min\{d - L_1, d - L_2\} - p_{b1}$ and $t_1 = p_{b1}$ and $0 < L_1 < p_{a1}$ and
$0 \le B \le d - L_2$ and $t_2 = 0$ and $0 < L_2 < p_{b2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_a L_1 + w_b (p_{b1} + L_2) \quad (1)$$

else $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty$ (2)

Case I.1.2 $J_a \in J^1,\ J_b \in J^2$ (cf. Figure 5.2)

if $0 \le A \le d - L_1$ and $t_1 = 0$ and $0 < L_1 < p_{a1}$ and
$0 \le B \le d - L_2$ and $t_2 = 0$ and $0 < L_2 < p_{b2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_a L_1 + w_b L_2 \quad (3)$$

else $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty$ (4)

Case I.1.3 $J_a \in J^2,\ J_b \in J^2$ (cf. Figure 5.3)

if $0 \le A \le d - L_1$ and $t_1 = 0$ and $0 < L_1 < p_{a1}$ and
$0 \le B \le \min\{d - L_1, d - L_2\} - p_{a2}$ and $t_2 = p_{a2}$ and $0 < L_2 < p_{b2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_a (p_{a2} + L_1) + w_b L_2 \quad (5)$$

else $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty$ (6)

Case I.1.4 $J_a \in J^2,\ J_b \in J^1$ (cf. Figure 5.4)

if $0 \le A \le \min\{d - L_1, d - L_2\} - p_{b1}$ and $t_1 = p_{b1}$ and $0 < L_1 < p_{a1}$ and
$0 \le B \le \min\{d - L_1, d - L_2\} - p_{a2}$ and $t_2 = p_{a2}$ and $0 < L_2 < p_{b2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_a (p_{a2} + L_1) + w_b (p_{b1} + L_2) \quad (7)$$

else $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty$ (8)

Determining the initial value of the weighted early work, we count early parts of jobs J_a, J_b for feasible values of parameters A, B, t_1, t_2, L_1 and L_2 (Terms 1, 3, 5, 7). Early tasks (if any) on M_1,

M_2 must exactly fill intervals of lengths t_1, t_2, respectively. Similarly, partially late tasks M_1, M_2 have to fill exactly intervals of lengths L_1, L_2. Finally, A and B have to be properly chosen to ensure that jobs J_a, J_b are the jobs with partially late tasks. Infeasible parameter values (Terms 2, 4, 6, 8) lead to the initial criterion value equal to minus infinity. That means that such solutions are rejected.

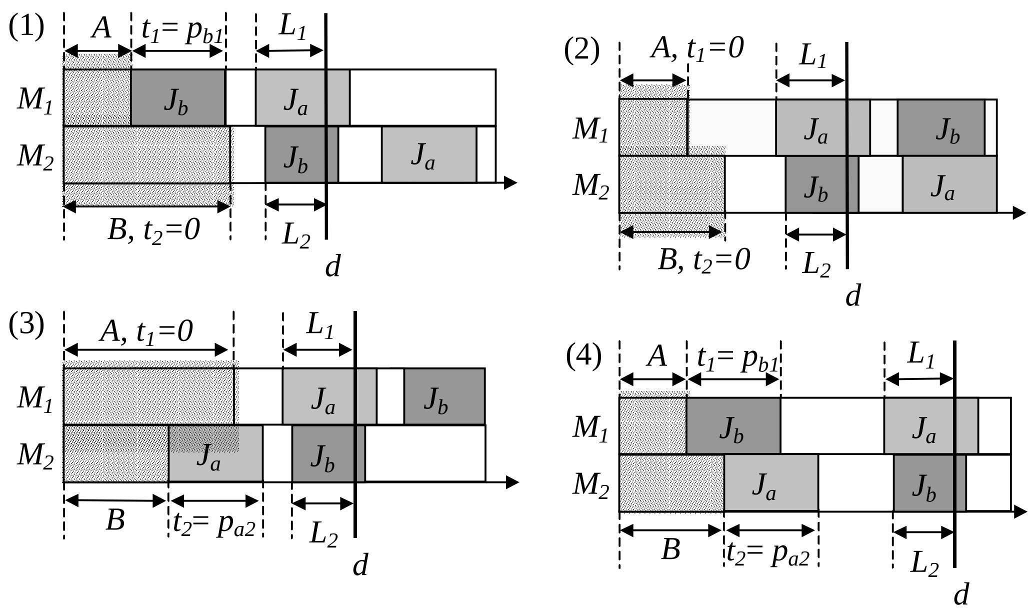

Figure 5. Initial conditions for different sets $J^P = \{J_a, J_b\}$

Case I.2 $J^P = \{J_x\}$

Let J^P contain only one job, i.e. $J^P = \{J_x\}$. In this case, we have only two subcases depending on the type of job J_x.

Case I.2.1 $J_x \in J^1$ (cf. Figure 6)

if $0 \le A \le d - L_2 - p_{x1}$ and $t_1 = p_{x1}$ and $L_1 = 0$ and

$0 \le B \le d - L_2$ and $t_2 = 0$ and $0 < L_2 < p_{x2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_x (p_{x1} + L_2) \quad (9)$$

if $0 \le A \le d - L_1$ and $t_1 = 0$ and $0 < L_1 < p_{x1}$ and

$0 \le B \le d$ and $t_2 = 0$ and $L_2 = 0$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_x L_1 \quad (10)$$

if otherwise, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty \quad (11)$$

Similarly as for a two-job set J^P, we detect infeasible parameter values (Term 11). For feasible parameter values, we check two possible ways of scheduling job J_x: with a partially late task on M_2 (Term 9, Figure 6.1) and with a partially late task on M_1 (Term 10, Figure 6.2).

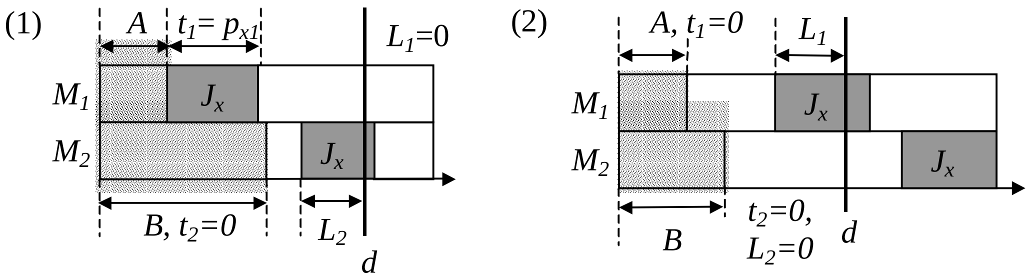

Figure 6. Initial conditions for $J_x \in J^1$, when J_x is partially late on M_2 (1) and on M_1 (2)

Case I.2.2 $J_x \in J^2$ (cf. Figure 7)

if $0 \le A \le d - L_1$ and $t_1 = 0$ and $0 < L_1 < p_{x1}$ and

$0 \le B \le d - L_1 - p_{x2}$ and $t_2 = p_{x2}$ and $L_2 = 0$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_x (p_{x2} + L_1) \tag{12}$$

if $0 \le A \le d$ and $t_1 = 0$ and $L_1 = 0$ and

$0 \le B \le d - L_2$ and $t_2 = 0$ and $0 < L_2 < p_{x2}$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = w_x L_2 \tag{13}$$

if otherwise, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty \tag{14}$$

As in the previous case, we detect infeasible parameter values (Term 14) and for feasible parameter values, we check two possible ways of scheduling job J_x: with a partially late task on M_1 (Term 12, Figure 7.1) and on M_2 (Term 13, Figure 7.2).

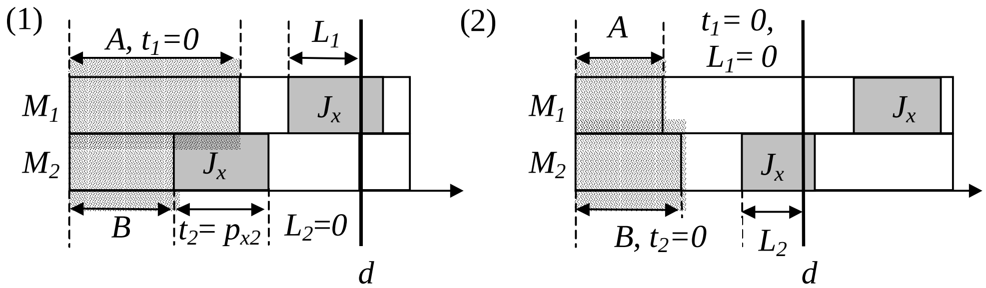

Figure 7. Initial conditions for $J_x \in J^2$, when J_x is partially late on M_1 (1) and on M_2 (2)

Case I.3 $J^P = \emptyset$

Finally, we have to analyze the case when no partially late task exists in the system, for which the initial conditions are formulated below. Such a situation occurs, when on a particular machine a task finishes/starts exactly at time d or there is idle time around a common due date.

if $0 \le A \le d$ and $t_1 = 0$ and $L_1 = 0$ and

$0 \le B \le d$ and $t_2 = 0$ and $L_2 = 0$, then

$$f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = 0 \tag{15}$$

else $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2) = -\infty$ (16)

Taking into account the fact that all parameters of function $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2)$ are bounded by $O(d)$ the calculation of the initial conditions for any set J^P takes $O(d^6)$ time.

3. Recurrence Relations

After determining the initial conditions for a particular set J^P, we calculate the recurrence relations for the remaining jobs $\hat{J}_k \in J\backslash J^P$, numbered according to Jackson's rule as $\hat{J}_1, \ldots, \hat{J}_u, \hat{J}_{u+1}, \ldots, \hat{J}_{\tilde{n}}$. As we have mentioned, first, we analyze jobs with the first (only) task on M_2 ($k = \tilde{n}, \ldots, u+1$). Then, jobs with the first (only) task on M_1 are taken into account ($k = u, \ldots, 1$). Analyzing job $\hat{J}_k$, we determine the amount of the weighted early work for jobs $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\}\cup J^P$ based on the recurrence relation $f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)$. The meaning of the parameters changes slightly depending on the job type, whether $\hat{J}_k \in J^2\backslash J^P$ or $\hat{J}_k \in J^1\backslash J^P$.

Case R.1 $k = \tilde{n}, \ldots, u+1$ (i.e. $\hat{J}_k \in J^2\backslash J^P$)

For job $\hat{J}_k \in J^2\backslash J^P$, processed first on M_2 then on M_1, $f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)$ denotes the maximum amount of the weighted early work of jobs $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\}\cup J^P$ provided that (cf. Figure 8):

- the first job from this set starts processing exactly at time B on M_2 and not earlier than at time A on M_1 (jobs from $J^1\backslash J^P$ will be scheduled within time A in the following stages of the DP method),
- there are exactly r_2 time units in the interval $[B, d]$ reserved for scheduling second tasks of jobs from $J^1\backslash J^P$ in the following stages (within this interval jobs from $J^2\backslash J^P$ are not scheduled on M_2),

- there are exactly r_1 time units in interval $[A, d]$ reserved for processing jobs from $J^2\backslash J^P$ on M_1 (all tasks of early jobs from $J^2\backslash J^P$ have to be executed within this interval),
- the first tasks of tardy jobs from $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\}\cup J^P$ are processed exactly t_2 time units on M_2 before d and exactly T_2 units are reserved on M_2 before d for the first tasks of tardy jobs $\hat{J}_i$ from $J^2\backslash J^P$ for $i < k$,
- there are exactly L_1 (L_2) units of partially late tasks on M_1, M_2 (they belong to jobs J_a, J_b or J_x).

Parameters t_1, T_1 are not important at this stage of the analysis (those intervals are embedded within A from the point of view of job $\hat{J}_k$). They play analogous roles as t_2, T_2 for jobs from $J^1\backslash J^P$ in the following stages of DP. Parameter F denotes the assumed completion time of the last early job from $J^1\backslash J^P$ on M_1. Actually, it is important only for job $\hat{J}_{\tilde{n}}$ when the initial conditions $f_{\tilde{n}+1}$ are called. The precise value of this parameter is settled during the remaining steps of the method when set $J^1\backslash J^P$ is considered.

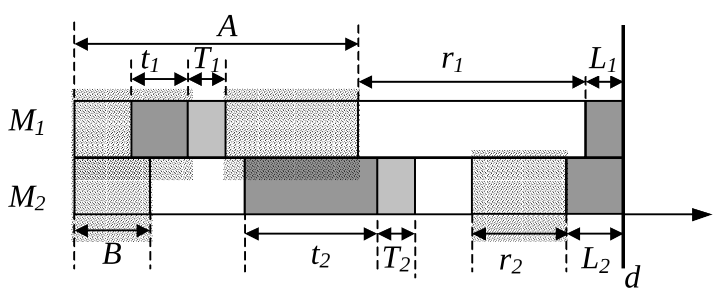

Figure 8. Parameters of the recurrence relations for $\hat{J}_k \in J^2\backslash J^P$

Jobs $\hat{J}_k \in J^2\backslash J^P$ are analyzed from $k = \tilde{n}$ to u+1. Determining the recurrence relation f_k for $\hat{J}_k$ we use the result obtained for $\hat{J}_{k+1}$ (f_{k+1}). For this reason, the formulation of the recurrence relations for the last job $\hat{J}_{\tilde{n}}$, requiring the result of calculations of the initial condition $f_{\tilde{n}+1}$, is slightly different. It is calculated as the first one, however, for the sake of clarity, we will show it later.

For jobs $\hat{J}_k \in J^2\backslash J^P$, where $k = \tilde{n}-1, \ldots, u+1$, the recurrence relations are as follows:

if $B + t_2 + T_2 + r_2 + L_2 \le d$ and $A + r_1 + L_1 \le d$ and $t_1 + T_1 \le A$ and $F \le A$, then

if $B + p_{k2} + t_2 + T_2 + r_2 + L_2 \le d$ and $\max\{A, B+p_{k2}\} + p_{k1} + L_1 \le d$ and $p_{k1} \le r_1$, then

$f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2) = \max\{$

$w_k(p_{k1}+p_{k2}) + f_{k+1}(\max\{A,B+p_{k2}\}+p_{k1}, t_1, T_1, r_1-p_{k1}, L_1, F, B+p_{k2}, t_2, T_2, r_2, L_2),$ (17)

$w_kp_{k2} + f_{k+1}(A, t_1, T_1, r_1, L_1, F, B, t_2-p_{k2}, T_2+p_{k2}, r_2, L_2)$ if $p_{k2} \le t_2,$ (18)

$f_{k+1}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)\}$ (19)

else

$f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2\ ) = \max\{$

$w_k p_{k2} + f_{k+1}(A, t_1, T_1, r_1\ , L_1, F,\ B, t_2 - p_{k2}, T_2 + p_{k2}\ r_2, L_2)$ if $p_{k2} \le t_2$, (20)

$f_{k+1}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)\}$ (21)

else

$f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2) = -\infty$ (22)

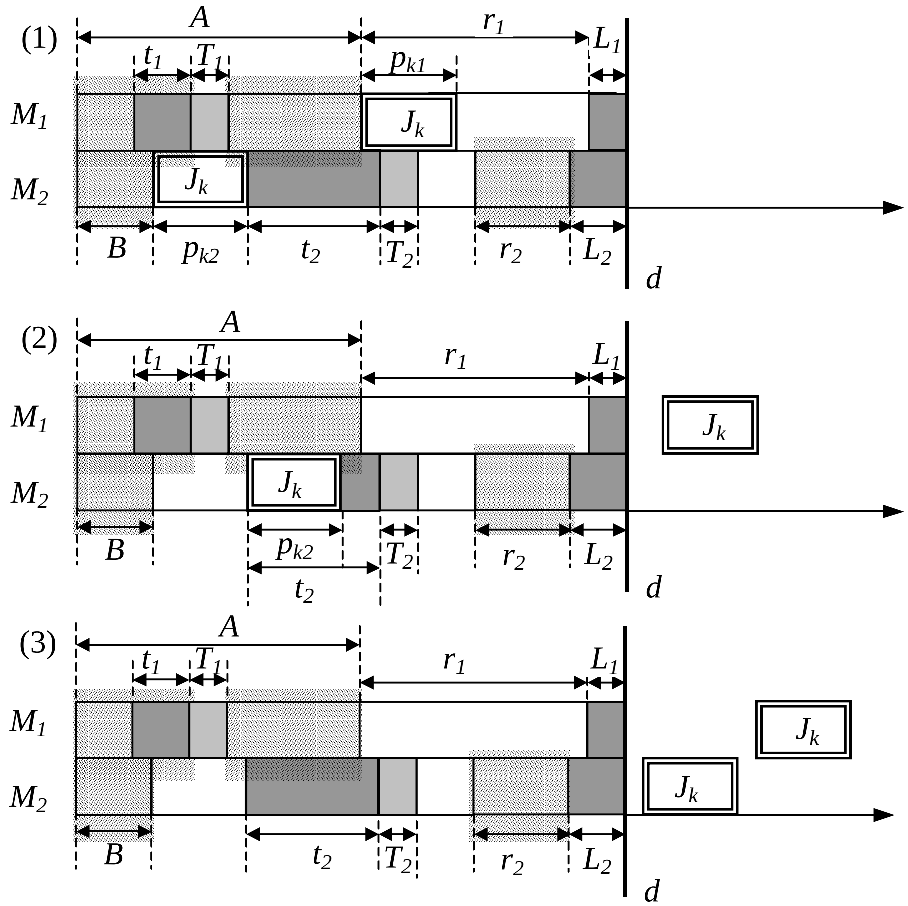

Figure 9. Recurrence relations for $\hat{J}_k \in J^2 \backslash J^P$ executed early (1), early only on M_2 (2), totally late (3)

If parameter values are infeasible (i.e. gaps between A, B and d are not long enough to contain intervals r_1, L_1 and t_2, T_2, r_2, L_2, respectively, or A is too small to contain t_1, T_1, then the function takes minus infinity value (Term 22). Otherwise, we have to check all possible ways of scheduling job $\hat{J}_k$ and select the best one (ensuring the maximum weighted early work). If job $\hat{J}_k$ can be scheduled early (Terms 17-19, Figure 9) then we compare 3 possible subschedules, when this job is early (Term 17, Figure 9.1), only its first task is early (Term 18, Figure 9.2) and the job is totally late (Term 19, Figure 9.3). The case when $\hat{J}_k$ is early only on M_2 is under consideration only, if interval t_2 is long enough to contain the whole task of $\hat{J}_k$. If job $\hat{J}_k$ cannot be scheduled early

(Terms 20,21) then only 2 cases are possible when only its first task is early (assuming that t_2 is long enough, Term 20) or it is totally late (Term 21).

As mentioned, the recurrence relations for the last job from set $J^2\backslash J^P$ ($\hat{J}_{\tilde{n}}$) are formulated differently:

if $B + t_2 + T_2 + r_2 + L_2 \le d$ and $A + r_1 + L_1 \le d$ and $t_1 + T_1 \le A$ and $F \le A$, then

if $B + p_{\tilde{n}2} + t_2 + T_2 + r_2 + L_2 \le d$ and $\max\{A, B + p_{\tilde{n}2}\} + p_{\tilde{n}1} + L_1 \le d$ and $p_{\tilde{n}1} \le r_1$, then

$f_{\tilde{n}}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2) = \max\{$

$w_{\tilde{n}}(p_{\tilde{n}1} + p_{\tilde{n}2}) + f_{\tilde{n}+1}(F, t_1, L_1, B + p_{\tilde{n}2}, t_2, L_2),$ (23)

$w_{\tilde{n}}p_{\tilde{n}2} + f_{\tilde{n}+1}(F, t_1, L_1, B, t_2 - p_{\tilde{n}2}, L_2)$ if $p_{\tilde{n}2} \le t_2,$ (24)

$f_{\tilde{n}+1}(F, t_1, L_1, B, t_2, L_2)\}$ (25)

else

$f_{\tilde{n}}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2) = \max\{$

$w_{\tilde{n}}p_{\tilde{n}2} + f_{\tilde{n}+1}(F, t_1, L_1, B, t_2 - p_{\tilde{n}2}, L_2)$ if $p_{\tilde{n}2} \le t_2,$ (26)

$f_{\tilde{n}+1}(F, t_1, L_1, B, t_2, L_2)\}$ (27)

else

$f_{\tilde{n}}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2) = -\infty$ (28)

The case study for $\hat{J}_{\tilde{n}}$ is identical as for other jobs $\hat{J}_k \in J^2\backslash J^P$. The only difference is that calculating $f_{\tilde{n}}$, we determine the weighted early work for jobs $\{\hat{J}_{\tilde{n}}\} \cup J^P$ and we have to use the criterion value $f_{\tilde{n}+1}$ calculated for J^P (not f_{k+1} calculated for another job $\hat{J}_{k+1} \in J^2\backslash J^P$). Function $f_{\tilde{n}+1}$ is defined for a different parameter set than the recurrence relation f_k. Parameters representing reserved intervals T_1, r_1, T_2, r_2 are not important for J^P. Similarly parameter F becomes the first parameter of the function $f_{\tilde{n}+1}$, it is used for determining the possible starting time for J^P on M_1, i.e. parameter F determines the value of A for J^P. Similarly as for $\hat{J}_{\tilde{n}}$, we have to change the formulation of the recurrence relations for jobs $\hat{J}_k \in J^2\backslash J^P$ which contain only one task, requiring machine M_2 (p_{k1}=0). In those cases, precedence constraints do not exist and we remove Term 17 from the definition of f_k (and Term 23 if $\hat{J}_{\tilde{n}}$ contains only one task).

In the presented the recurrence relations, all parameters of the analyzed function $f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)$ are bounded by $O(d)$. Thus, determining the recurrence relations for particular job $\hat{J}_k \in J^2\backslash J^P$ takes $O(d^{11})$ time.

The analysis of $J^2\backslash J^P$ is followed by the analysis of jobs with the first (only) task on machine M_1. As we have mentioned, the recurrence relations have to be adjusted to a different type of precedence constraints between tasks.

Case R.2 $k = u, \ldots, 1$ (i.e. $\hat{J}_k \in J^1 \backslash J^P$)

For job $\hat{J}_k \in J^1 \backslash J^P$, processed first on M_1 then on M_2, $f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)$ denotes the maximum amount of the weighted early work of jobs $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\} \cup J^P$ provided that (cf. Figure 10):

- the first job from this set starts processing exactly at time A on M_1 and not earlier than at time B on M_2 (jobs from $J^2 \backslash J^P$ have been scheduled within interval B in DP stages described above),
- there are exactly r_1 time units in the interval $[A, d]$ reserved for scheduling second tasks of jobs from $J^2 \backslash J^P$ (actually, they have been already scheduled there in the previous stages), within this interval jobs from $J^1 \backslash J^P$ are not scheduled on M_1,
- there are exactly r_2 time units in interval $[B, d]$ reserved for processing jobs $\hat{J}_i$ from $J^1 \backslash J^P$ for $i < k$ on M_2,
- the first tasks of tardy jobs from $\{\hat{J}_k, \ldots, \hat{J}_{\tilde{n}}\} \cup J^P$ are processed exactly t_1 time units on M_1 before d and exactly T_1 units are reserved on M_1 before d for the first tasks of tardy jobs $\hat{J}_i$ from $J^1 \backslash J^P$ for $i < k$,
- there are exactly L_1 (L_2) units of partially late tasks on M_1, M_2 (they belong to jobs J_a, J_b or J_x).

Similarly as in Case R.1, parameters t_2, T_2 are not important at this stage of analysis (those intervals are embedded within B from the point of view of job $\hat{J}_k$). The sixth parameter of the function f_k denoting the completion time of the last early job from $J^1 \backslash J^P$ on M_1 is determined by the value of A, so it is not a variable from the point of view of jobs from $J^1 \backslash J^P$ (in the contrary to the jobs from $J^2 \backslash J^P$, for which the mentioned parameter is a variable denoted by F).

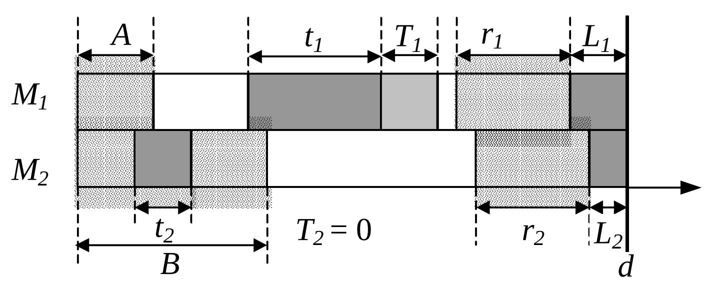

Figure 10. Parameters of the recurrence relations for $\hat{J}_k \in J^1 \backslash J^P$

Jobs $\hat{J}_k \in J^1 \backslash J^P$ are analyzed from $k = u$ to 1. Again, determining the recurrence relation f_k for $\hat{J}_k$ we use the result obtained for $\hat{J}_{k+1}$ (f_{k+1}). For this reason, the formulation of the recurrence relations for the last job $\hat{J}_u$, requiring value f_{u+1}, is slightly different, because $\hat{J}_{u+1}$ belongs to $J^2 \backslash J^P$ not to $J^1 \backslash J^P$. f_u is calculated as the first one, however, for the sake of clarity, we will show it later, as in Case R.1.

For jobs $\hat{J}_k \in J^1 \backslash J^P$, where $k = u\text{-}1, \ldots, 1$, the recurrence relations are as follows:

if $A + t_1 + T_1 + r_1 + L_1 \le d$ and $B + r_2 + L_2 \le d$ and $t_2 \le B$ and $T_2 = 0$, then

if $A + p_{k1} + t_1 + T_1 + r_1 + L_1 \le d$ and $\max\{A + p_{k1}, B\} + p_{k2} + r_2 + L_2 \le d$, then

$f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = \max\{$

$$w_k(p_{k1}+p_{k2}) + f_{k+1}(A+p_{k1}, t_1, T_1, r_1, L_1, A+p_{k1}, \max\{A+p_{k1},B\}+p_{k2}, t_2, T_2, r_2+p_{k2}, L_2), \tag{29}$$

$$w_k p_{k1} + f_{k+1}(A, t_1\text{-}p_{k1}, T_1 + p_{k1}, r_1, L_1, A, B, t_2, T_2, r_2, L_2) \text{ if } p_{k1} \le t_1, \tag{30}$$

$$f_{k+1}(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)\} \tag{31}$$

else

$f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = \max\{$

$$w_k p_{k1} + f_{k+1}(A, t_1 - p_{k1}, T_1 + p_{k1}, r_1, L_1, A, B, t_2, T_2, r_2, L_2) \text{ if } p_{k1} \le t_1, \tag{32}$$

$$f_{k+1}(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)\} \tag{33}$$

else

$$f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = -\infty \tag{34}$$

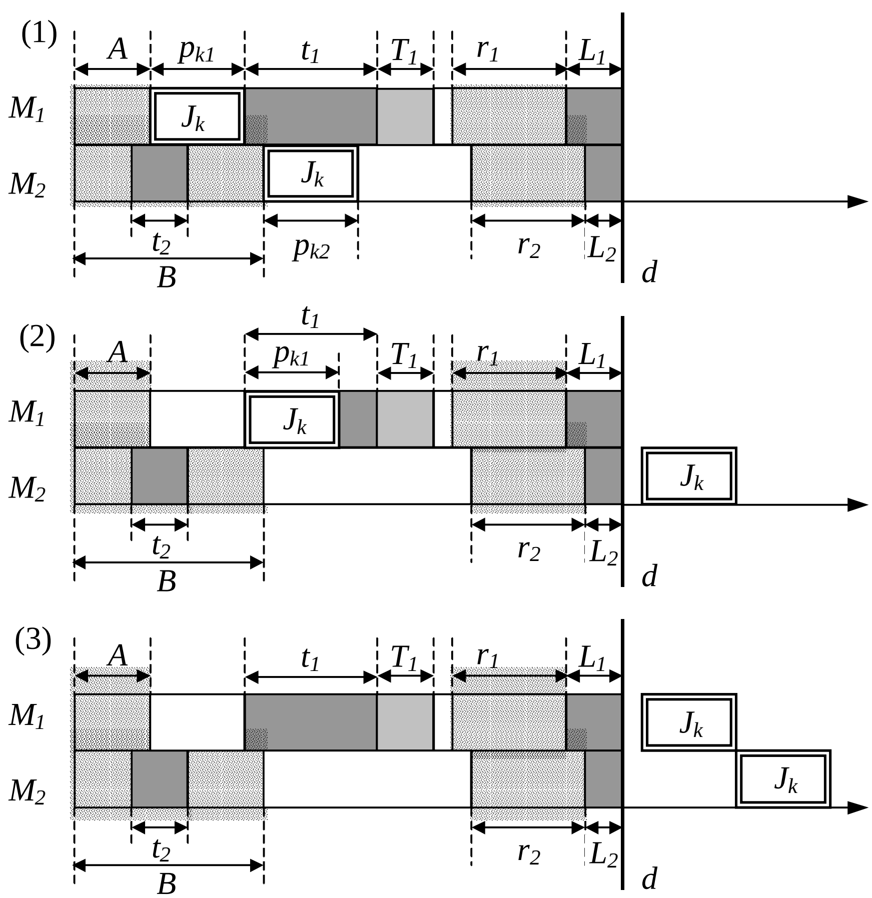

Figure 11. Recurrence relations for $\hat{J}_k \in J^1 \backslash J^P$ executed early (1), early only on M_1 (2), totally late (3)

If parameter values are infeasible (i.e. gaps between A, B and d are not long enough to contain intervals t_1, T_1, r_1, L_1, and r_2, L_2 respectively, or B is too small to contain t_2, or T_2 is different from 0, i.e. there is a reserved interval for jobs from J^2, although all jobs from this set have been already considered), then the function takes the minus infinity value (Term 34). Otherwise, we have to analyze all possible ways of executing job $\hat{J}_k$ and choose the best one subject to the weighted early work. If job $\hat{J}_k$ can be scheduled early (Terms 29-31, Figure 11) then we compare 3 possible solutions, when this job is early (Term 29, Figure 11.1), only its first task is early (Term 30, Figure 11.2) and the job is totally late (Term 31, Figure 11.3). The case when $\hat{J}_k$ is early only on M_1 is under consideration only, if interval t_1 is long enough to contain the whole task of $\hat{J}_k$. If job $\hat{J}_k$ cannot be scheduled early (Terms 32, 33) then 2 cases are possible: when only its first task is early (assuming that t_1 is long enough, Term 32) or it is totally late (Term 33).

As we have announced, the recurrence relations for the last job from set $J^1\backslash J^P$, i.e. job $\hat{J}_u$, are formulated differently:

if $A + t_1 + T_1 + r_1 + L_1 \le d$ and $B + r_2 + L_2 \le d$ and $t_2 \le B$ and $T_2 = 0$, then

if $A + p_{u1} + t_1 + T_1 + r_1 + L_1 \le d$ and $\max\{A + p_{u1}, B\} + p_{u2} + r_2 + L_2 \le d$, then

$f_u(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = \max\{$

$w_u(p_{u1}+p_{u2})+ f_{u+1}(A+p_{u1}+t_1+T_1,t_1,T_1,d-(A+p_{u1}+t_1+T_1+L_1),L_1,A+p_{u1},0,t_2,T_2,r_2+p_{u2},L_2)$ (35)

$w_u p_{u1}+f_{u+1}(A+t_1+T_1,t_1-p_{u1},T_1+p_{u1},d-(A+p_{u1}+t_1+T_1+L_1),L_1,A,0,t_2,T_2,r_2,L_2)$ if $p_{u1}\le t_1$, (36)

$f_{u+1}(A + t_1 + T_1, t_1 , T_1, d - (A + t_1 + T_1 + L_1), L_1, A, 0, t_2, T_2, r_2, L_2)\}$ (37)

else

$f_u(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = \max\{$

$w_u p_{u1}+f_{u+1}(A+t_1+T_1,t_1-p_{u1},T_1+p_{u1},d-(A+p_{u1}+t_1+T_1+L_1),L_1,A,0,t_2,T_2,r_2,L_2)$ if $p_{u1}\le t_1$, (38)

$f_{u+1}(A + t_1 + T_1, t_1 , T_1, d - (A + t_1 + T_1 + L_1), L_1, A, 0, t_2, T_2, r_2, L_2)\}$ (39)

else

$f_u(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2) = - \infty$ (40)

We consider for $\hat{J}_u$ the same cases as for other jobs $\hat{J}_k \in J^1\backslash J^P$, but determining the weighted early work for jobs $\{\hat{J}_u, \hat{J}_{u+1}, \ldots, \hat{J}_{\tilde{n}}\}\cup J^P$, we have to use the criterion value f_{u+1} calculated for a job from set $J^2\backslash J^P$. Switching from set $J^1\backslash J^P$ to $J^2\backslash J^P$, i.e. calling function f_{u+1}, we assume that B equals to 0, because early jobs from $J^2\backslash J^P$ start at time 0 on M_2. Then, A is extended with intervals t_1 and T_1 (parameter A takes the value $A+t_1+T_1$ or $A+t_1+T_1+p_{u1}$ for job $\hat{J}_{u+1}$, depending on the way $\hat{J}_u$ is scheduled). From the point of view of job $\hat{J}_{u+1}$, all jobs from $J^1\backslash J^P$ have to be executed before A, despite the fact whether they are early or partially late. Then, we determine interval r_1, not used by jobs from $J^1\backslash J^P$, as d-(A+p_{u1}+t_1+T_1+L_1) or d-(A+t_1+T_1+L_1), depending on the way $\hat{J}_u$ is executed. For $\hat{J}_{u+1}$, we have to know exactly the length of the interval not used by jobs from $J^1\backslash J^P$. Finally, we settle the value of the sixth parameter of the function (denoted as F for $\hat{J}_{u+1}$) to the completion time of the last early job from $J^1\backslash J^P$ on M_1.

As for $\hat{J}_k \in J^2\backslash J^P$, in the case of $\hat{J}_k \in J^1\backslash J^P$ having only one task, requiring machine M_1 ($p_{k2} = 0$), we remove Term 29 from f_k definition (and Term 35 if job J_u contains only one task).

Similarly as in the case of jobs from set $J^2\backslash J^P$, for $\hat{J}_k \in J^1\backslash J^P$ all parameters in the recurrence relations $f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)$ are bounded by $O(d)$. But, as we have mentioned, for this type of jobs, the sixth parameter of the function, denoting the completion time of the last early job from $J^1\backslash J^P$ on M_1, is determined based on the value of A. For this reason, the time complexity of the calculations of the recurrence relations decreases to $O(d^{10})$ for a particular job $\hat{J}_k \in J^1\backslash J^P$.

To determine the maximum weighted late work subject to a given set J^P, one has to select the maximum value of $f_1(0, t_1, 0, r_1, L_1, 0, B, t_2, 0, 0, L_2)$ for $0 \le t_1, r_1, L_1, B, t_2, L_2 \le d$, where function f_1 denotes the weighted early work for all jobs $\{\hat{J}_1, \ldots, \hat{J}_{\tilde{n}}\} \cup J^P$. Changing parameters t_1, L_1, t_2, L_2, we check solutions obtained for all possible amounts of early tasks of late jobs, while changing r_1 and B, we reserve different amounts of time on M_1 and M_2 for jobs from $J^2\backslash J^P$. Determining the maximal total weighted early work for a particular set J^P takes $O(d^6)$ time.

4. Dynamic Programming Framework

In this section, we review the suggested approach for the problem under consideration. As we have mentioned, to find an optimal solution of problem $J2 \mid d_i = d \mid Y_w$, we have to analyze all possible sets J^P of jobs with partially late tasks on machines M_1, M_2. For a particular set J^P, the dynamic programming calculations can be described by the algorithm given below.

procedure $DP(J^P)$
    set $\hat{J} = J\backslash J^P$, $\tilde{n} = |\hat{J}|$ and $u = |J^1\backslash J^P|$
    renumber jobs from $\hat{J}$ in Jackson's order as $\hat{J}_1, \ldots, \hat{J}_u, \hat{J}_{u+1}, \ldots, \hat{J}_{\tilde{n}}$
    for $0 \le A, t_1, L_1, B, t_2, L_2 \le d$
        calculate initial conditions $f_{\tilde{n}+1}(A, t_1, L_1, B, t_2, L_2)$

for $0 \le A,t_1,T_1,r_1,L_1,F,B,t_2,T_2,r_2,L_2 \le d$

calculate recurrence relations $f_{\tilde{n}}(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)$

for $k = \tilde{n}-1$ to $u+1$ do

for $0 \le A,t_1,T_1,r_1,L_1,F,B,t_2,T_2,r_2,L_2 \le d$

calculate recurrence relations $f_k(A, t_1, T_1, r_1, L_1, F, B, t_2, T_2, r_2, L_2)$

for $0 \le A,t_1,T_1,r_1,L_1,B,t_2,T_2,r_2,L_2 \le d$

calculate recurrence relations $f_u(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)$

for $k = u-1$ to 1 do

for $0 \le A,t_1,T_1,r_1,L_1,B,t_2,T_2,r_2,L_2 \le d$

calculate recurrence relations $f_k(A, t_1, T_1, r_1, L_1, A, B, t_2, T_2, r_2, L_2)$

$$Y_w(J^P) = \max_{0 \le t_1,r_1,L_1,B,t_2,L_2 \le d} \{ f_1(0, t_1, 0, r_1, L_1, 0, B, t_2, 0, 0, L_2)\}$$

First, we renumber all jobs, except jobs from J^P, in Jackson's order. Then, the initial conditions are determined for a given set J^P. In the next steps, the recurrence relations are calculated for jobs with the first (only) task performed on M_2. For job $\hat{J}_{\tilde{n}}$, we use a specialized formulation of the recurrence function. Then, the recurrence relations are determined for the remaining jobs with the first (only) task performed on machine M_1. Again, for job J_u, a specialized recurrence function is applied. Finally, we determine the maximum amount of the weighted early work, $Y_w(J^P)$, which can be obtained subject to a given set J^P.

As we have mentioned in the previous section, the initial conditions are calculated in $O(d^6)$ time. Fixing the recurrence relations for a single job requires at most $O(d^{11})$ time, but this stage of the algorithm has to be repeated for all jobs from $J \backslash J^P$, i.e. $O(n)$ times. The maximum criterion value can be found in $O(d^6)$ time. Thus, the overall complexity of the considered procedure is $O(nd^{11})$.

The calculations described above, have to be performed for all possible sets J^P, i.e. containing two jobs, one or no job with a task partially late on machine M_1 or M_2, in order to find an optimal solution of the problem under consideration. Consequently, dynamic programming calculations have to be repeated for all $O(n^2)$ two-job sets, all $O(n)$ one-job sets and for an empty set J^P. That gives the complexity $O(n^3d^{11})$.

After determining the set J^{P*}, which results in the schedule with the maximum (optimal) weighted early work Y_w^*, we have to construct an optimal solution based on the decisions taken during DP calculations for J^{P*}. They divided $J \backslash J^{P*}$ into five subsets (cf. Figure 12):

- $J^{E(1)}$ and $J^{E(2)}$ containing early jobs from J^1, J^2 respectively,
- $J^{L(1)}$ and $J^{L(2)}$ containing jobs from $J^1 \backslash J^{P*}$, $J^2 \backslash J^{P*}$ with first task early and the second task totally late,

- J^L containing totally late jobs.

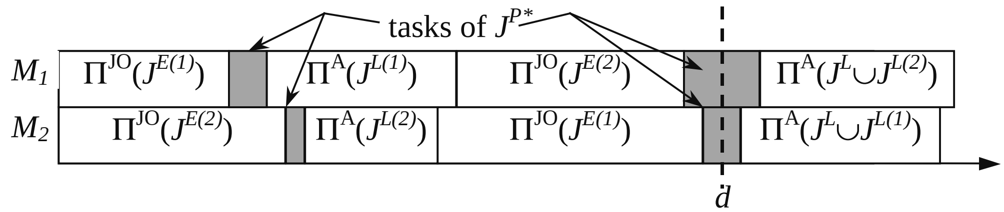

Figure 12. The structure of an optimal solution

To build a schedule on machine M_1, denoted as Π_1, first we execute early jobs from $J^{E(1)}$ in Jackson's order obtaining subschedule $\Pi^{JO}(J^{E(1)})$. It is followed by the early task of a job from J^{P*} and, then, by tasks from $J^{L(1)}$ executed in arbitrary order (subschedule $\Pi^{A}(J^{L(1)})$). After those tasks of partially late jobs, we perform second tasks of early jobs from $J^{E(2)}$ in Jackson's order obtaining subschedule $\Pi^{JO}(J^{E(2)})$. Then, the partially late task of a job from J^{P*} has to be scheduled followed by arbitrarily ordered late tasks of jobs from $J^L \cup J^{L(2)}$ (subschedule $\Pi^{A}(J^L \cup J^{L(2)})$). Schedule Π_2 on machine M_2 is constructed in a similar way. Depending on the problem instance some subschedules mentioned above may be empty.

The construction of an optimal schedule does not increase the overall complexity of the dynamic programming approach summarized below.

$J = \{J_1, .., J_n\}$
for each pair $\{J_a, J_b\} \subseteq J$ calculate $DP(\{J_a, J_b\})$
for each $J_x \in J$ calculate $DP(\{J_x\})$
for $J^P = \emptyset$ calculate $DP(\emptyset)$
set $Y_w{*} = \max \{ Y_w(J^P) \}$, i.e. the optimal total weighted early work for all sets J^P analyzed
set J^{P*} to be a set with $Y_w(J^P) = Y_w{*}$
based on dynamic programming results for the set J^{P*} determine $J^{E(1)}$, $J^{E(2)}$, $J^{L(1)}$, $J^{L(2)}$ and J^L
construct an optimal schedule as:
Π_1=($\Pi^{JO}(J^{E(1)})$,early task from J^{P*}on M_1, $\Pi^{A}(J^{L(1)})$,$\Pi^{JO}(J^{E(2)})$, partially late task from J^{P*} on M_1, $\Pi^{A}(J^L \cup J^{L(2)})$)
Π_2=($\Pi^{JO}(J^{E(2)})$,early task from J^{P*}on M_2, $\Pi^{A}(J^{L(2)})$,$\Pi^{JO}(J^{E(1)})$, partially late task from J^{P*} on M_1, $\Pi^{A}(J^L \cup J^{L(1)})$)

The presented method allows us to find an optimal solution of problem $J2 \mid d_i = d \mid Y_w$ in pseudo-polynomial time. Thus, we can classify this scheduling case as binary *NP*-hard.

5. Conclusions

The paper presents a dynamic programming approach for the revenue management problem, which can be modeled as a job-shop scheduling problem ($J2 \mid d_i = d \mid Y_w$) with the total weighted late work criterion and a common due date. The *NP*-hardness of the flow-shop problem, $F2 \mid d_i = d \mid Y_w$, being a special case of $J2 \mid d_i = d \mid Y_w$, resulted in the *NP*-hardness of the job-shop case. But, it was not settled, whether the latter problem is binary or unary *NP*-hard.

Proposing a solution method with pseudo-polynomial time complexity, we have proven the binary *NP*-hardness of the problem considered.

References

Błażewicz, Jacek. (1984). "Scheduling preemptible tasks on parallel processors with information loss," *Recherche Technique et Science Informatiques* 3/6, 415-420.

Błażewicz, Jacek, Klaus Ecker, Erwin Pesch, Günter Schmidt, and Jan Węglarz. (2001). *Scheduling Computer and Manufacturing Processes*. 2nd edn., Berlin Heidelberg New York : Springer.

Błażewicz, Jacek and Gerd Finke. (1987). "Minimizing mean weighted execution time loss on identical and uniform processors," *Information Processing Letters* 24, 259-263.

Błażewicz, Jacek, Erwin Pesch, Małgorzata Sterna, and Frank Werner. (2000). "Total late work criteria for shop scheduling problems." In: K. Inderfurth, G. Schwödiauer, W. Domschke, F. Juhnke, P. Kleinschmidt, and G. Wäscher (eds.), *Operations Research Proceedings 1999*, 354-359, Berlin.

Błażewicz, Jacek, Erwin Pesch, Małgorzata Sterna, and Frank Werner. (2003a). "Open shop scheduling problems with late work criteria," *Discrete Applied Mathematics*, to appear.

Błażewicz, Jacek, Erwin Pesch, Małgorzata Sterna, and Frank Werner. (2003b). "The two-machine flow-shop problem with weighted late work criterion and common due date," *European Journal of Operational Research*, to appear.

Brucker, Peter. (1998). *Scheduling Algorithms*. 2nd edn., Berlin Heidelberg New York: Springer.

Garey, R. Michael and David S. Johnson. (1979). *Computers and Intractability*. San Francisco: W.H. Freeman and Co.

Jackson, R. James. (1956). "An extension of Johnson's results on job shop scheduling," *Naval Research Logistics Quarterly* 3, 201-203.

Johnson, M. S. (1954). "Optimal two- and three-stage production schedules," *Naval Research Logistics Quarterly* 1, 61-68.

Józefowska, Joanna, Bernd Jurisch, and Wiesław Kubiak. (1994). "Scheduling shops to minimize the weighted number of late jobs," *Operation Research Letters* 16/5, 277-283.

McGill, I. Jeff and Garrett J. Van Ryzin. (1999). "Revenue management. Research overview and prospects," *Transportation Science* 33, 233-256.

Pinedo, Michael and Xiuli Chao. (1999). *Operation Scheduling with Applications in Manufacturing and Services*. Boston: Irwin/McGraw-Hill.

Potts, N. Chris and Luk N. Van Wassenhove. (1991a). "Single machine scheduling to minimize total late work," *Operations Research* 40/3, 586-595.

Potts, N. Chris and Luk N. Van Wassenhove. (1991b). “Approximation algorithms for scheduling a single machine to minimize total late work,” *Operations Research Letters* 11. 261-266.

Sterna, Małgorzata. (2000). *Problems and Algorithms in Non-Classical Shop Scheduling*. Poznań: Scientific Publishers of the Polish Academy of Sciences.

Wirtz, Jochen, Sheryl E. Kimes, Jeannette Ho Pheng Theng, and Paul Patterson. (2002). “*Revenue management: resolving potential customer conflicts*,” Working Paper of Cornell University.