UNIVERSITÄT KAISERSLAUTERN

On the Solution Region for Certain Scheduling Problems with Preemption

Heidemarie Bräsel and Natalia Shakhlevich

Preprint Nr. 271

ISSN 0943-8874

Februar 1996

FACHBEREICH MATHEMATIK

On the Solution Region for Certain Scheduling Problems with Preemption *

Heidemarie Bräsel
Universität Kaiserslautern
Erwin-Schrödinger-Straße
67663 Kaiserslautern
Germany

Natalia Shakhlevich
Academy of Sciences of Belarus
Surganov Street 6
220012 Minsk
Belarus

Abstract: The paper deals with parallel-machine and open-shop scheduling problems with preemptions and arbitrary nondecreasing objective function. An approach to describe the solution region for these problems and to reduce them to minimization problems on polytopes is proposed. Properties of the solution regions for certain problems are investigated. It is proved that open-shop problems with unit processing times are equivalent to certain parallel-machine problems, where preemption is allowed at arbitrary time. A polynomial algorithm is presented transforming a schedule of one type into a schedule of the other type.

Keywords: parallel-machine problem, open-shop problem, scheduling polytope.

1 Introduction

We consider processing systems of two types: single-stage processing systems with parallel machines and multi-stage open-shop systems. For both processing systems a set of jobs $\mathcal{N} = \{J_1, \dots, J_n\}$ is given which has to be processed by a set of machines $\mathcal{M} = \{M_1, \dots, M_m\}$. Following the traditional classification scheme introduced by Graham et al. [6], we denote the scheduling problem as $\alpha \mid \beta \mid \gamma$, where α indicates the machine environment, β gives the job characteristics or some additional assumptions, and γ indicates the optimality criterion.

In the case of a parallel-machine system all machines are identical and each job can be processed by any machine ($\alpha = P$). The processing time p_i of job J_i is independent of the machine. Therefore, in the problem $P \mid\mid F$ we have to construct a job order on each machine for which a given objective function F is minimal. If in addition all processing times are equal to 1 we choose the parameter β by $p_i = 1$, i.e. the problem $P \mid p_i = 1 \mid F$ is considered.

In the case of an open-shop problem ($\alpha = O$) all machines are different and each job $J_i \in \mathcal{N}$ has to be processed by all machines of the set $\mathcal{M}$, the sequence of operations being arbitrary. The processing time of job J_i on machine M_k is p_{ik}. Here we have to determine a feasible combination of job and machine orders for which a given objective function F takes its minimal value. This problem is denoted by $O \mid\mid F$. In the problem $O \mid p_{ik} = 1 \mid F$ the processing times of all operations are equal: $p_{ik} = 1$.

*Supported by DFG, project ScheMA, and by INTAS, project 93-257

If preemption is forbidden, then a schedule s may be given by start times or completion times of all operations. If preemption is allowed, then we denote it by parameter *pmtn* in the β-field. In this case any operation may be interrupted at any time moment and continued later on so that the sum of processing times of all parts of the operation is equal to the given value p_i or p_{ik}. The schedule may be given by start times (or completion times) of all parts of the operations and by durations of these parts.

The objective function $F = F(C_1(s), \ldots, C_n(s))$ depends on the completion times $C_1(s), \ldots, C_n(s)$ of jobs $J_1, \ldots, J_n$ in the schedule s and it is nondecreasing in respect to each of its arguments, i.e. if for two schedules s and s' the condition $C_i(s) \leq C_i(s')$ holds for all $J_i \in \mathcal{N}$, then the inequality

$$F(C_1(s), \ldots, C_n(s)) \leq F(C_1(s'), \ldots, C_n(s'))$$

is satisfied.

To simplify further discussions we denote the problem $P \mid pmtn \mid F$ as *Problem A* and the problem $O \mid pmtn \mid F$ as *Problem B*. Furthermore, *Problem A'* and *Problem B'* are the corresponding problems with the additional condition $p_i = 1$ and $p_{ik} = 1$, respectively. We assume that $n > m$.

To solve general parallel-machine and open-shop problems one can consider only so-called dense schedules. In a *dense schedule* any machine is idle if and only if there is no job waiting for this machine (see Bárány / Fiala T. [2]). It is evident that there exist optimal schedules for the problems A and B which belong to this class.

The main difficulty for solving problems with preemption in comparison with analogous problems without preemption is the following. The set of the dense schedules for problems without preemption is finite while the set of the dense schedules for the problems with preemption is uncountable. In the following we name the set of all dense schedules *the solution region*. We show that the solution region can be described by an union of polytopes of the same structure. Thus to solve initial scheduling problem we reduce it to a problem of minimizing function F on the solution region.

The dimension of the constructed scheduling polytopes and the number of polytopes is not polynomial in n. Nevertheless our approach may be useful to solve scheduling problems with a fixed number of jobs. Moreover, it is unlikely to construct a polynomial algorithm to solve the problems $P|pmtn|\sum \omega_i C_i$ and $O|pmtn|\sum C_i$ if the number of jobs is unlimited and the number of machines is greater than 1 (the NP-hardness of thesee problems was proved by Livshits / Rublinecky [7], Bruno et al. [4], Du / Leung [5]).

Thus if initial scheduling problem is rather general (arbitrary processing times of operations, non-symmetric objective function), then the application of our approach is justified only for limited number of jobs. For instance, it may be usefull, to solve efficiently the problem $O|n = 3, pmtn|F$ for $F = \sum \omega_i C_i$ or $F = \sum \omega_i C_i^2$. On the other hand, if the problem is more specific, it can be solved efficiently. In particular, we investigate some properties of the unit-time problems and obtain the solutions of some problems with unlimited n.

The paper is organized as follows. In Section 2 we present a technique for the description of the solution region of *Problem A* and *Problem B*. We show that these problems can be reduced to the determination of a global optimum on an union of polytopes. In Section 3 we prove the equivalence between *Problem A'* and *Problem B'*. In Section 4 we demonstrate how the results of Sections 2 and 3 can be applied for certain objective functions: symmetric functions and e-quasi-concave functions. The special cases of *Problem A'* and *Problem B'* when $n = m+1$ are considered in Section 5. Concluding remarks are presented in Section 6.

2 The Description of the Solution Region

In this paper we will use the notion "state" to describe any schedule. In frameworks of one state $< h_l >$ all processes in the system do not change. So a schedule can be given by the sequence of states $(< h_1 >, < h_2 >, \dots, < h_q >)$. Each state $< h_l >, 1 \le l \le q$, is characterized

i) by the duration $\tau_l \ge 0$;

ii) by the set of jobs $\mathcal{N}_l = \{J_{i_1}, \dots, J_{i_\nu}\} \subseteq \mathcal{N}$, $|\mathcal{N}_l| = \nu \le \min\{n, m\}$, which are processed continuously in this time slot;

iii) by the set of machines $\mathcal{M}_l = \{M_{k_1}, \dots, M_{k_\nu}\} \subseteq \mathcal{M}$, $|\mathcal{M}_l| = |\mathcal{N}_l|$, which process jobs $\mathcal{N}_l$ within the state.

When one state finishes and the next one starts following situations are possible:

- the processing of some jobs from $\mathcal{N}_l$ may stop;
- the processing of some jobs from $\mathcal{N} \setminus \mathcal{N}_l$ may start;
- the machine processing certain job from $\mathcal{N}_l$ may be replaced by another machine.

Now we split the set of the dense schedules S into $n!$ subsets $S_{(i_1,i_2,\dots,i_n)}$ where $(i_1, i_2, \dots, i_n)$ is a permutation of the job numbers $1, 2, \dots, n$. Class $S_{(i_1,i_2,\dots,i_n)}$ contains those dense schedules s for which condition

$$C_{i_1}(s) \le C_{i_2}(s) \le \dots \le C_{i_n}(s) \tag{1}$$

is satisfied. To construct an optimal schedule s^* for initial scheduling problem one can construct optimal schedules relating to each subset and then determine the global optimal schedule among these local optimal schedules.

At first we consider the parallel-machine problem (Subsection 2.1) and then the open-shop problem (Subsection 2.2).

2.1 The Solution Region of Problem A

Now we describe an algorithm for constructing all dense schedules $s \in S_{(1,2,\dots,n)}$. The idea is to construct at first such a special sequence of the states that condition (1) can be satisfied (it is realized by Algorithm 1). Finally we describe the complete model where the durations of the states are determined.

Since in *Problem A* all machines are identical, it is not necessary to consider characteristic iii) of a state. So in Algorithm 1 only characteristic ii) is essential. It means that two states $< h_i >$ and $< h_j >$ are considered to be different if $\mathcal{N}_i \neq \mathcal{N}_j$ for $i \neq j$.

Algorithm 1 enumerates all pairwise different states in a special order. It is obvious that number q of pairwise different states can be determined for $n \geq m$ by

$$q = \sum_{v=1}^{m} \binom{n}{v}.$$

Clearly, the number of all states which contain job J_1 is given by

$$q_1 = \sum_{v=0}^{m-1} \binom{n-1}{v}.$$

More generally, the number of all states which contain job J_l and do not contain any job J_i with $i < l$ can be calculated as

$$q_l = \begin{cases} \sum\limits_{v=0}^{m-1} \binom{n-l}{v} & \text{for} \quad 1 \leq l \leq n-m, \\ \sum\limits_{v=0}^{n-l} \binom{n-l}{v} & \text{for} \quad n-m+1 \leq l \leq n. \end{cases}$$

Observe that the overall number of pairwise different states q is equal to $\sum\limits_{l=1}^{n} q_l$.

Algorithm 1: Enumeration of all different states for the parallel-machine problem

S1: k:=1;
S2: **for** $l := 1$ **to** n **do**
begin
S3: construct a set of states H_l, $|H_l| = q_l$, which contain job J_l and do not contain any job J_i with $i < l$. Order all states in H_l by nonincreasing cardinalities of the job sets:
$H_l := \{< h_k >, < h_{k+1} >, \ldots, < h_{k+q_l-1} >\}$ with
$|N_{h_k}| \geq |N_{h_k+1}| \geq \ldots \geq |N_{h_{k+q_k-1}}|$;
S4: $k := k + q_l$;
end.

It is easy to see that the complexity bound of Algorithm 1 is $O(q)$. Fig. 1 represents the sequence of the states in the case of four jobs and three machines. We will discuss about the states marked by double rectangles later on.

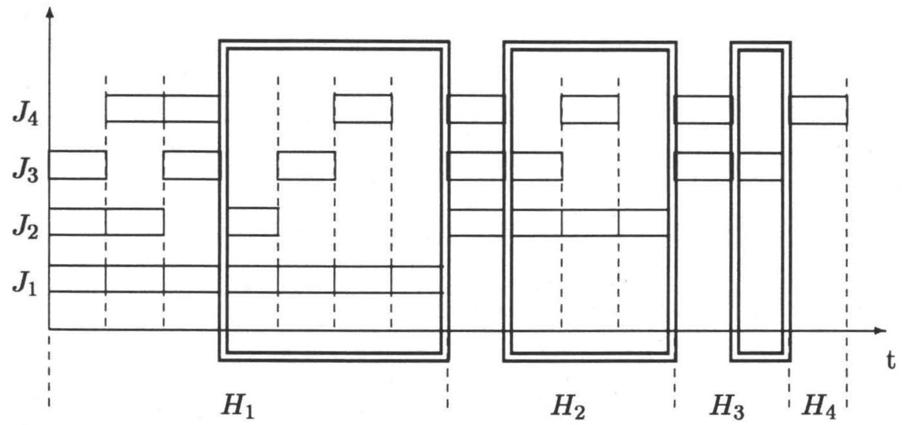


Fig. 1 *Enumeration of the states in the case of $n = 4$ and $m = 3$ in the class $S_{(1,2,3,4)}$*

Theorem 1 *Let $\overline{s}$ be a feasible schedule from the class $S_{(1,2,\ldots,n)}$ and its sequence of the states differs from a sequence of the states of schedule $s \in S_{(1,2,\ldots,n)}$ constructed by Algorithm 1. Then there exists a transformation of $\overline{s}$ into s, and schedule s is not worse than schedule $\overline{s}$, i.e.*

$$C_i(s) \leq C_i(\overline{s}) \quad \text{for } 1 \leq i \leq n. \qquad (2)$$

Proof: The proof consists of two parts. Firstly we describe a transformation of $\overline{s}$ to a schedule without repetitions of the states. Afterwards we modify this schedule into s.

1. Let schedule $\overline{s} \in S_{(1,2,\ldots,n)}$ be given by the sequence of states $(G_1, < h_u >, G_2, < h_u >, G_3)$, where G_1, G_2, G_3 are some subsets of the states. We construct a new schedule $(G_1, G_2, < h_u >, < h_u >, G_3)$ in which two states $< h_u >$ can be considered as a one state of longer duration. Observe that the completion times of the jobs remain the same or some of them may become smaller.

Fulfilling such a transformation for all states that occur in $\overline{s}$ more than once we obtain a schedule without repetitions of the states.

2. Consider now a schedule $\overline{s}$ without repetitions of the states. Time interval $[0, C_1(\overline{s})]$ consists of states H_1 and perhaps of some other states G_1 in which job J_1 is not processed.

If G_1 is empty, then we proceed with the next time interval. Otherwise we push all such states after the states H_1. It decreases the completion time of job J_1 and does not change the completion times of all other jobs.

The next time interval $[\tau(H_1), C_2(\overline{s})]$ with $\tau(H_1) = \sum_{l \in H_1} \tau_l$, consists of states H_2 and perhaps of some other states G_2 in which job J_2 is not processed. Applying the above shifting we decrease now the completion time of job J_2 to the value $\tau(H_1) + \tau(H_2)$ and do not change the completion times of all other jobs, and so on.

After such a transformation is carried out to the time interval $[\sum_{l=1}^{n-2} \tau(H_l), C_{n-1}(\overline{s})]$, we obtain a schedule with the same job completion times like schedule s. It may differ from the schedule s only by the order of jobs within each set $H_l, 1 \leq l \leq n$. Reordering states in the obtained schedule within each set H_l does not change job completion times and leads to schedule s. Since all transformations above do not increase completion time of any job, then condition (2) is satisfied. ■

Theorem 1 means that we can restrict ourselves by schedules with the sequence of states constructed by Algorithm 1. Now we show that for *Problem A* the number q of states which are constructed in Algorithm 1 can be reduced. Consider state $< h_l > \in H_k$ which is connected with processing jobs $\mathcal{N}_l$, $\mathcal{N}_l \subseteq \{J_k, J_{k+1}, \ldots, J_n\}$. Let call the state $< h_l >$ a *full state* if $|\mathcal{N}_l| = \min\{m, n-k\}$, e.g. maximal number of jobs is processed within such a state. It is easy to prove that for *Problem A* the following statement is valid.

Theorem 2 *Schedule s constructed by Algorithm 1 is a dense one if all its states $< h_l >$, $1 \leq l \leq q$, with nonzero durations ($\tau_l > 0$) are full states.*

Proof: Suppose s is a dense schedule and nevertheless there exists a nonfull state $< h_l >$. Then some job $J_i \in \mathcal{N}_l$ is not processed within this state, some machine is idle within the same state and job J_i is processed later on within other states. Since all machines are identical, the job can be processed by this machine within state $< h_l >$. This contradicts to the definition of a dense schedule. ∎

Therefore, it is possible to consider schedules consisting of full states only. In Fig. 1 all nonfull states constructed by Algorithm 1 for the problem with three jobs and four machines are marked by double rectangles. In Fig. 2 these states are excluded due to Theorem 2.

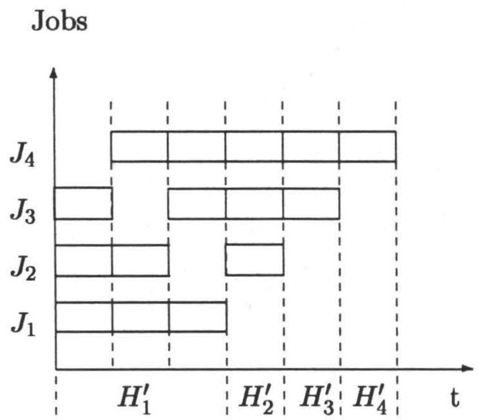


Fig. 2 *Enumeration of the full states in the case of $n = 4$ and $m = 3$ in the class $S_{(1,2,3,4)}$*

The number q' of different full states H' is limited by

$$q' = \binom{n}{m} + m - 1.$$

Algorithm 1 can be easily modified to an algorithm which enumerates full states H', $H' \subset H$. To this end we split the set H' into subsets H'_l, $l = 1, \ldots, n$. As in Algorithm 1, states H'_l contain job J_l and do not contain any job J_i with $i < l$. The number q'_l of the full states H'_l can be calculated as

$$q'_l = \begin{cases} \binom{n-l}{m-1} & \text{for} \quad 1 \leq l \leq n-m, \\ 1 & \text{for} \quad n-m+1 \leq l \leq n. \end{cases}$$

Thus we have to modify Step 3 in Algorithm 1.

Algorithm 2: Enumeration of all full states for the parallel-machine problem

S1: k:=1;
S2: **for** $l := 1$ **to** n **do**
  begin
S3:   construct a set of full states H'_l, $|H'_l| = q'_l$, which contain job J_l and do not contain any job J_i with $i < l$;
S4: $k := k + q'_l$;
  end.

Now we are able to calculate the durations of the states of the schedule $s \in S_{(1,2,\ldots,n)}$. We construct a system of n linear equations which describes the processing of n jobs of the schedule $s \in S_{(1,2,\ldots,n)}$ under the condition that processing time of job J_i, $1 \leq i \leq n$, is p_i. In this purpose we construct a $(n \times q')$-matrix Z where each column Z_j, $1 \leq j \leq q'$, describes the state $< h_j >$:

$$z_{ij} = \begin{cases} 1, & \text{if } J_i \text{ is processed in the state } < h_j >, \\ 0, & \text{otherwise.} \end{cases}$$

So the duration of processing job J_i, $1 \leq i \leq n$, within schedule s can be calculated as $\sum_{j=1}^{q'} z_{ij}\tau_j$, and this sum must be equal to p_i. Adding restrictions $\tau_j \geq 0$, $j = 1, 2, \ldots, q'$, we obtain a mixed system of n linear equations and q' inequalities where the number of variables is equal to q':

$$Z\tau = p, \tag{3}$$
$$\tau \geq 0. \tag{4}$$

Here τ and p are the vectors of all τ_j and p_i, respectively.

To transform the objective function $F(C_1, C_2, \ldots, C_n)$ into a function $\tilde{F}(\tau_1, \tau_2, \ldots, \tau_{q'})$, we construct the expressions for $C_1, C_2, \ldots, C_n$:

$$C_1(s) = \sum_{j=1}^{q'_1} \tau_j, \qquad C_2(s) = \sum_{j=1}^{q'_1+q'_2} \tau_j, \qquad \ldots, \qquad C_n(s) = \sum_{j=1}^{q'} \tau_j.$$

For the class of the schedules $S_{(1,\ldots,n)}$ the *Problem A* is transformed into the following optimization problem:

$$\boxed{\min\{\tilde{F}(\tau_1, \tau_2, \ldots, \tau_{q'}) \mid Z\tau = p, \ \tau \geq 0\}}$$

To solve the *Problem A* one needs to consider all classes of the schedules $S_{(i_1,i_2,\ldots,i_n)}$ for all different permutations $(i_1, i_2, \ldots, i_n)$ of job numbers $1, 2, \ldots, n$. A new order of jobs for $S_{(i_1,i_2,\ldots,i_n)}$ changes the right hand side of equations (3) and the minimizing function $\tilde{F}(\tau_1, \tau_2, \ldots, \tau_{q'})$. The optimal schedule for the initial problem has to be selected among the schedules which are optimal in the classes $S_{(i_1,i_2,\ldots,i_n)}$.

Thus the solution region of the *Problem A* can be decribed by an union of polytopes. Each of them contains dense schedules of a certain class $S_{(i_1,i_2,\ldots,i_n)}$.

Observe that the problem is quite easier in the case of identical processing times and symmetric objective function. We remind that a function is *symmetric* if any permutation of its arguments does not change its value, i.e.

$$F(C_1, C_2 \ldots, C_n) = F(C_{i_1}, Ci_2, \ldots, C_{i_n})$$

holds for any permutation $(i_1, i_2, \ldots, i_n)$. In this case we have to consider only one class of the schedules which defines one polytope.

2.2 The Solution Region of Problem B

To solve *Problem B* we again consider a set of the schedules in the classes $S_{(i_1,i_2,\ldots,i_n)}$ for all different permutations $(i_1, i_2, \ldots, i_n)$ of job numbers $1, 2, \ldots, n$, and construct an optimal schedule for each class.

Algorithm 3 constructs a sequence of the states of schedules $S_{(1,2,\ldots,n)}$. It is similar to Algorithm 1 which enumerates all q states (full and nonfull) because Theorem 2 is not valid for the *Problem B*. In addition Algorithm 3 assigns concrete machine number for each job from each state.

Algorithm 3: Enumeration of all different states for the open-shop problem

S1: k:=1;
S2: **for** $l := 1$ **to** n **do**
    begin
S3:   construct a set of states H_l as in Step 3 of Algorithm 1;
S4: $k := k + q_l$;
    end.
S5: $q := k$;
S6: **for** $l := 1$ **to** q **do**
    begin
S7: $m' := |\mathcal{N}_l|$;
S8:   determine number r_l:
        if $m' = m$ **then** $r_l = m!$;
        if $m' < m$ **then** $r_l = \frac{m!}{(m-m')!}$;
S9:   split state $< h_l >$ into r_l new states;
S10:   assign the machine numbers to jobs $\mathcal{N}_l$:
        if $m' = m$ **then** enumerate all permutations of machine numbers $1, \ldots, m$;
        if $m' < m$ **then** enumerate all arrangements of machine numbers $1, \ldots, m$ taken m'.
    end

Fig. 3 represents a sequence of states of the schedule $s_{(1,2,\ldots,n)}$ obtained by Algorithm 3. The complexity bound of Algorithm 3 linearly depends on number r of different states:

$$r = \binom{n}{m} m! + \sum_{l=1}^{m-1} \binom{n}{m-l} \frac{m!}{(m-l)!}.$$

Here two states $< h_i >$ and $< h_j >$, $i \neq j$, are different if $\mathcal{N}_i \neq \mathcal{N}_j$, or if $\mathcal{N}_i = \mathcal{N}_j$ and at least one job from $\mathcal{N}_i$ is processed by different machines within these two states.

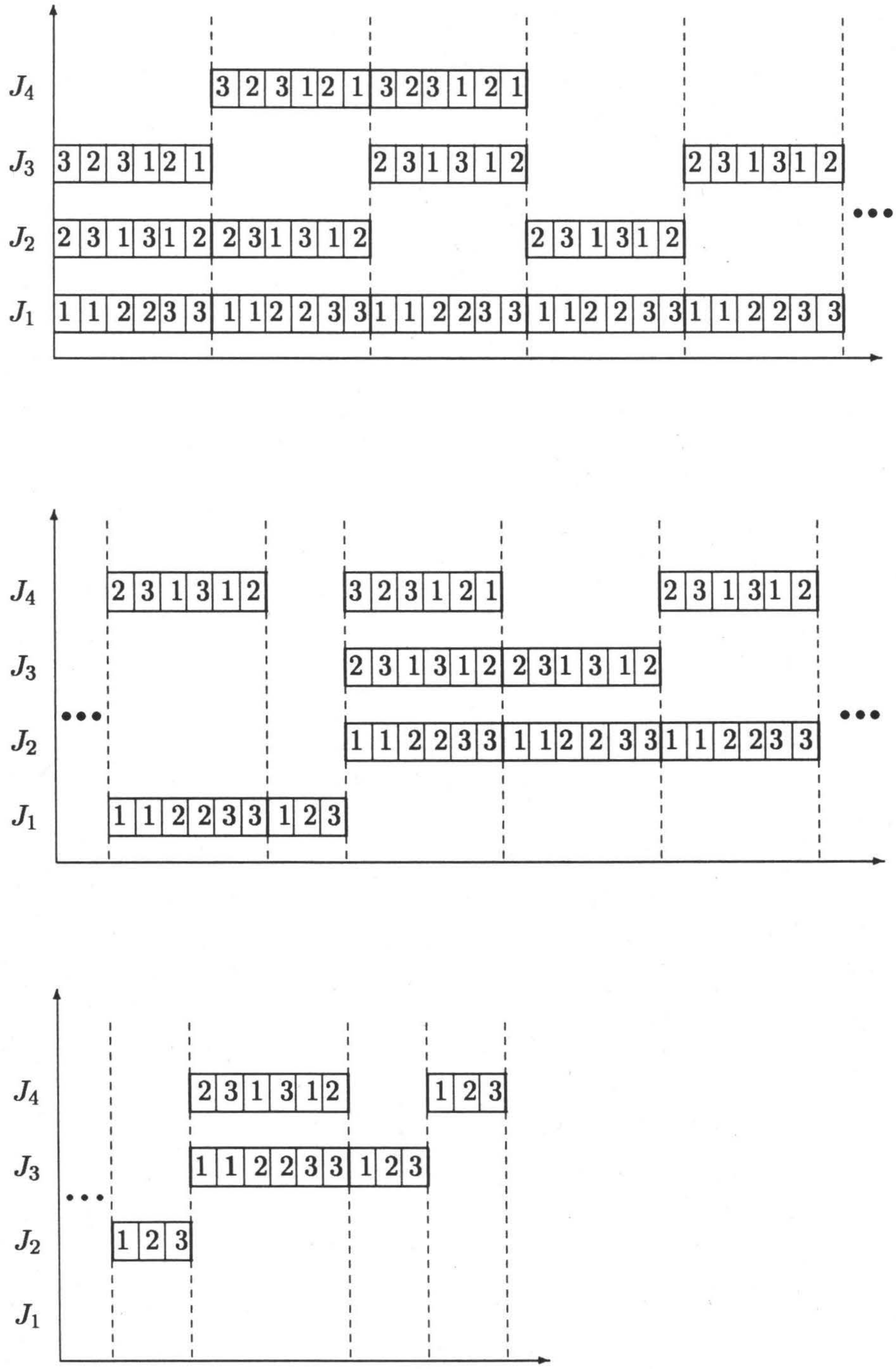


Fig. 3 *Enumeration of the states of the problem* $O3|n = 4, pmtn|F$ *in the class* $S_{(1,\ldots,n)}$

To determine the durations of the states of dense schedules from the class $S_{(1,,2,\ldots,n)}$ we proceed in a similar way as in the case of *Problem A*.

We construct a system of nm linear equations which describes fulfilling m operations for each job $\{J_1, J_2, \ldots, J_n\}$ within schedule $s \in S_{(1,2,\ldots,n)}$ under the condition that processing job $J_i \in \mathcal{N}$ on machine $M_k \in \mathcal{M}$ needs p_{ik} time. In this purpose we construct m matrices $Z^k, k = 1, \ldots, m$, of dimension $n \times r$. Each column $Z^k_j, 1 \leq j \leq r$, of matrix $Z^k, 1 \leq k \leq m$, describes state $< h_j >$:

$$z^k_{ij} = \begin{cases} 1, \text{ if } J_i \text{ is processed on machine } M_k \text{ in the state } < h_j >, \\ 0, \text{ otherwise.} \end{cases}$$

So the duration of processing job $J_i, 1 \leq i \leq n$, on machine M_k within the whole schedule $s \in S_{(1,2,\ldots,n)}$ can be calculated as $\sum_{j=1}^{r} z^k_{ij}\tau_j$ and it should be equal to p_{ik}.

Adding restrictions $\tau_j \geq 0, j = 1, 2, \ldots, r$, we obtain a mixed system of nm linear equations and r inequalities, the number of variables being equal to r:

$$\begin{aligned} Z^k\tau &= P^k, \qquad k = 1, \ldots, m, & (5) \\ \tau &\geq 0. & (6) \end{aligned}$$

Here P^k is n-vector of processing times of jobs $\mathcal{N}$ by machine M_k:

$$P^k = \begin{pmatrix} p_{1k} \\ p_{2k} \\ \vdots \\ p_{nk} \end{pmatrix}.$$

To transform the objective function $F(C_1, C_2, \ldots, C_n)$ into a function $\tilde{F}(\tau_1, \tau_2, \ldots, \tau_r)$, we construct the expressions for $C_1, C_2, \ldots, C_n$:

$$C_1(s) = \sum_{j=1}^{r_1+\ldots+r_{q_1}} \tau_j, \qquad C_2(s) = \sum_{j=1}^{r_1+\ldots+r_{q_2}} \tau_j, \qquad \ldots, \qquad C_n(s) = \sum_{j=1} r_1 + \ldots + r_{q_n}\tau_j.$$

For the class of the schedules $S_{(1,\ldots,n)}$ the *Problem B* is transformed into the following optimization problem:

$$\boxed{\min\{\tilde{F}(\tau_1, \tau_2, \ldots, \tau_r) \mid Z^k\tau = P^k, \text{ with } 1 \leq k \leq m, \ \tau \geq 0\}}$$

As in the case of the *Problem A*, the scheduling polytope costructed is a bounded set and its dimension is equal to $r - n$.

To solve the *Problem B* one needs to consider all classes of the schedules $S_{(i_1,i_2,\ldots,i_n)}$ for all different permutations $(i_1, i_2, \ldots, i_n)$ of job numbers $1, 2, \ldots, n$. The optimal schedule for the initial problem has to be selected among the schedules which are optimal in the classes $S_{(i_1,i_2,\ldots,i_n)}$.

2.3 An example

Now we illustrate Algorithm 3 by the following example. Consider the problem $O2|n = 3, Pmtn|\sum \omega_i C_i$. Processing times of operations and weights of the jobs are given by

$$P = [p_{ik}] = \begin{pmatrix} 3 & 5 \\ 7 & 2 \\ 1 & 4 \end{pmatrix}, \quad \omega_1 = 2, \quad \omega_2 = 1, \quad \omega_3 = 2.$$

An optimal solution obtained by technique of Section 2.2 is represented by the following Gantt chart.

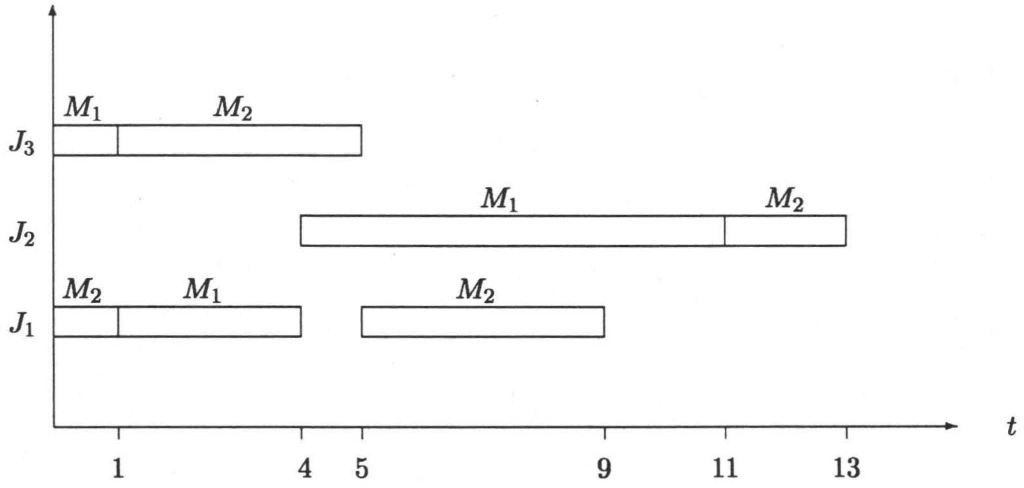


Fig. 4 *Optimal schedule for the problem* $O2|n = 3, pmtn|\sum \omega_i C_i$

Applying the same technique for minimizing objective function $F = \sum \omega_i C_i^2$ we obtain the same optimal schedule.

3 The equivalence of Problem A' and Problem B'

In what follows we consider two scheduling problems to be equivalent if there exists an algorithm which transforms any schedule s_a of the first problem into a schedule s_b of the second one with $C_i(s_a) = C_i(s_b)$ for all jobs $J_i \in \mathcal{N}$. To transform arbitrary feasible schedule s_a of the problem *Problem* A' to equivalent schedule s_b of the *Problem* B' we define a *splitting rule* which is applied to all states $< h_l >$ of schedule s_a:

- split the state $< h_l >$ into m substates $< h_l^1 >, < h_l^2 >, \ldots, < h_l^m >$ of equal durations $\tau_l^1, \tau_l^2, \ldots, \tau_l^m$.
- assign the following machine order to each job $J_{i_r} \in \mathcal{N}_l$, $1 \le r \le \nu$:

$$\begin{array}{llllll} J_{i_1}: & M_1 & \to M_2 & \to M_3 & \to \ldots \to M_{m-1} & \to M_m, \\ J_{i_2}: & M_m & \to M_1 & \to M_2 & \to \ldots \to M_{m-2} & \to M_{m-1}, \\ \ldots & & & & & \\ J_{i_\nu}: & M_{m-\nu+2} & \to M_{m-\nu+3} & \to M_{m-\nu+4} & \to \ldots \to M_{m-\nu} & \to M_{m-\nu+1}. \end{array}$$

The sequences of the machine numbers are obtained by cyclic permutations of numbers $1, 2, \ldots, m$. Since $|\mathcal{N}_l| \leq m$, there does not exist a machine which processes two or more jobs simultaneously within one state $< h_l^k >, 1 \leq l \leq q, 1 \leq k \leq m$. Now we are able to prove the following theorem.

Theorem 3 *Problem A' and Problem B' are equivalent.*

Proof: It is evident that any schedule s_b for *Problem B'* is a feasible schedule for *Problem A'* as well. For the transformation of an arbitrary schedule s_a to an equivalent schedule s_b the above described splitting rule is used. Let s_a be given by the the sequence of states $< h_l >$ with durations τ_l, $1 \leq l \leq q$. According to the splitting rule each state $< h_l >$ is broken into m substates $< h_l^1 >, < h_l^2 >, \ldots, < h_l^m >$ of equal durations $\tau_l^1 = \tau_l^2 = \ldots = \tau_l^m = \frac{1}{m}\tau_l$ and the corresponding machine orders are fixed. Now we show that each job $J_i \in \mathcal{N}$ is processed by machine $M_k \in \mathcal{M}$ with $p_{ik} = 1$ within schedule s_b.

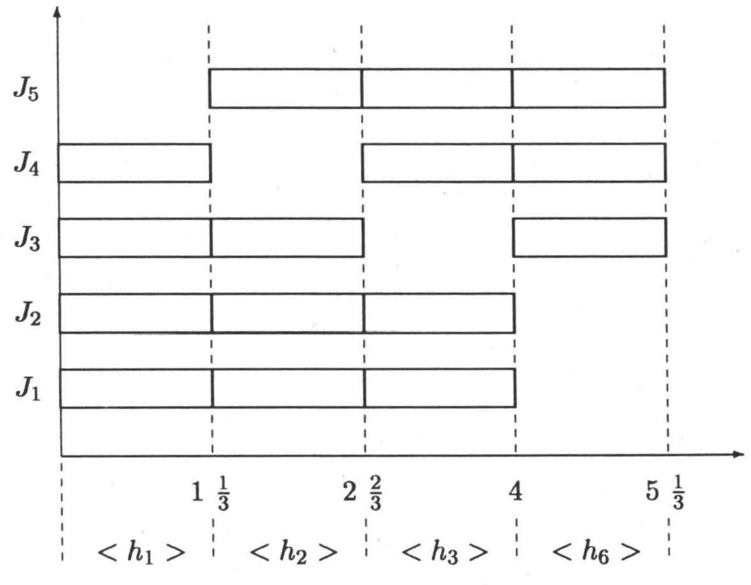
Let G_i be the set of states $< h_l >$ in which job J_i is processed in initial schedule s_a . Then $p_i = \sum_{<h_l> \in G_i} \tau_l = m$.

If $< h_l > \in G_i$, then in schedule s_b job J_i is processed on machine M_k within one of the states $< h_l^1 >, < h_l^2 >, \ldots$, or $< h_l^m >$, and the duration of such state is equal to $\frac{1}{m}\tau_l$. So within schedule s_b the following holds:

$$p_{ik} = \sum_{<h_l> \in G_i} \frac{1}{m}\tau_l.$$

Since for schedule s_a the relation $\sum_{<h_l> \in G_i} \tau_l = p_i = m$ is valid, then $p_{ik} = \frac{1}{m}m = 1$ is satisfies. ∎

The complexity bound of this transformation is equal to $O(qm^2)$, where q is the number of states with nonzero duration within schedule s_a. Figure 5 illustrates the transformation from the schedule s_a with $n = 5$ and $m = 4$ given in a) to the schedule s_b, see b).



a)

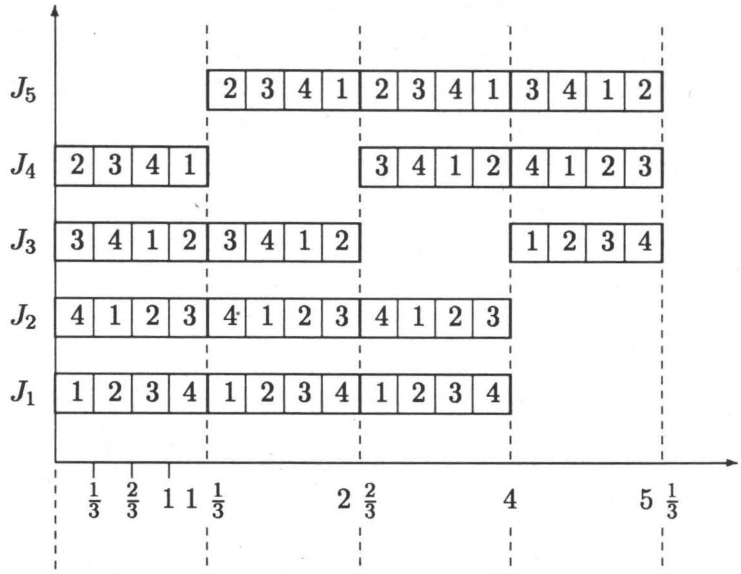


Fig. 5 *Transformation of schedule s_a into a schedule s_b*

In this example it is impossible to construct an open-shop schedule s_b without preemption. But there exists a wide class of schedules s_a which can be transformed into equivalent schedules s_b without preemption. This is the class of so-called "integer-schedules". A schedule is an *integer-schedule* if the start and finish times of all operations and of all their parts are integers. In this case all states of s_a have integer durations. A transformation of an integer schedule s_a for *Problem A'* into an integer schedule s_b for *Problem B'* is proposed in Brucker et al. [3], (with the time complexity $O(nm(\log nm)^2)$), and in Tautenhahn [12] (with the time complexity $O(nm\log(n+m))$). So if there exists an integer optimal schedule for *Problem A'*, then there exists also an optimal schedule without preemption for corresponding *Problem B'*. A list of problems with this property is given in [3]. But in general it is not the case for the same problems if release dates and due dates are not integral, and for the problems with other objective functions. For such problems the splitting rule can be applied.

4 On sufficient conditions for existing optimal schedule without preemption for the problem with preemption

In this section we consider nondecreasing objective functions (the definition is given in the Introduction). First we remind definitions of concave function, quasiconcave function and e-quasiconcave function (see Tanaev [11]).

Let E^n denotes the set of all n-vectors, and E_0^n denotes the set of all n-vectors with components from the set $\{0,1,-1\}$.

Function $F(x), x=(x_1,x_2,\ldots,x_n)$, is *concave*, if for any two vectors $x^{(1)}, x^{(2)} \in E^n$ and any number λ with $0 \le \lambda \le 1$ the following inequality is valid:

$$F(\lambda x^{(1)} + (1-\lambda)x^{(2)}) \ge \lambda F(x^{(1)}) + (1-\lambda)F(x^{(2)}).$$

Function $F(x)$ is *quasiconcave* if for any two vectors $x^{(1)}, x^{(2)} \in E^n$ and any number λ with $0 \leq \lambda \leq 1$ the following inequality is valid:

$$F(\lambda x^{(1)} + (1-\lambda)x^{(2)}) \geq \min\{F(x^{(1)}), F(x^{(2)})\}.$$

Function $F(x)$ is *e-quasiconcave* if for any vector $x^{(1)} \in E^n$, any vector $e \in E_0^n$, numbers α $(\alpha > 0)$ and λ $(0 \leq \lambda \leq 1)$ the following inequality is valid:

$$F(\lambda x^{(1)} + (1-\lambda)x^{(2)}) \geq \min\{F(x^{(1)}), F(x^{(2)})\},$$

where $x^{(2)} = x^{(1)} + \alpha e$ holds.

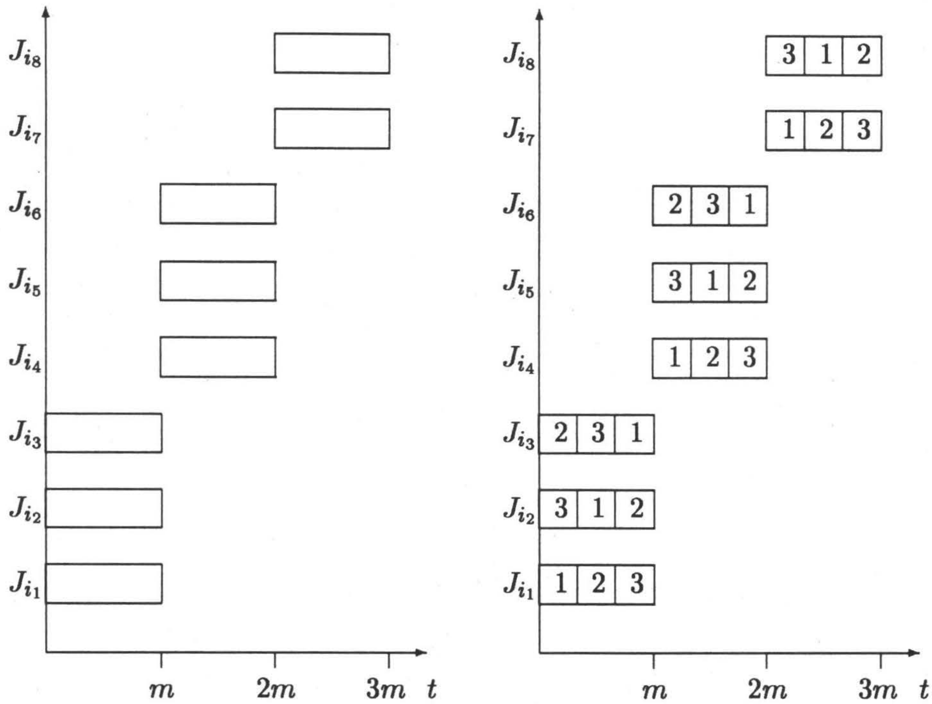
Due to the definition any concave function is quasiconcave, and any quasiconcave function is e-quasiconcave. It is easy to get sure that there exist e-quasiconcave functions which are not quasiconcave, and there also exist quasiconcave functions which are not concave.

It is known (see McNaughton [8]) that there exists an optimal schedule without preemption for the parallel machine problem with preemption if $F = \sum \omega_i C_i$. The most general result was established in [11]. It was proved that there exists nonpreemtive opimal schedule for *Problem A* if F is an e-quasiconcave function. Using Theorem 3 we generalize this result for *Problem B'* and thus we formulate a sufficient condition for existing optimal schedule without preemption for the problem with preemption.

Theorem 4 *If the objective function F of Problem B' is nondecreasing e-quasiconcave, then there exists an optimal schedule without preemption.*

Proof: Consider the equivalent *Problem A'* with the same objective function. Due to [11] there exists optimal schedule s_a without preemption. In this case the processing each job of such a schedule consists of only one state of length m. Due to Theorem 3 *Problem A'* and *Problem B'* are equivalent, and corresponding optimal schedule s_b for *Problem B'* can be obtained by the splitting rule. Splitting each state of length m into m substates of equal duration gives m unit-time operations of the schedule s_b. ■

In Figure 6 the proof is illustrated by an example with $n = 8$ and $m = 3$.



a) *b)*

Fig. 6 *Optimal schedules for the problems $P3|n = 8, pmtn|F$ and $O3|n = 8, pmtn|F$ if F is nondecreasing e-quasiconcave function*

Thus to construct a schedule s_b^* which is optimal for *Problem B'* with *e*-quasiconcave function F it is necessary to choose suitable class $S_{(i_1,i_2,\dots,i_n)}$ of the schedules, which gives optimal schedule for initial problem, i.e to choose suitable sequence of jobs $(J_{i_1}, J_{i_2}, \dots, J_{i_n})$ for concrete objective function. For some objective functions such sequence of jobs can be easily defined.

Consider *Problem B'* with objective function $F = \sum \omega_i C_i$. Function F is concave, and so Theorem 4 can be applied for it. It is evident that $n!$ schedules $s^*_{(1,2,\dots,n)}$, $s^*_{(2,1,\dots,n)}$, ..., $s^*_{(n,n-1,\dots,1)}$ which are optimal in corresponding classes, give $n!$ permutations of completion times (C_{i_1}, C_{i_2}, ..., C_{i_n}). Linear form $\sum\limits_{\mu=1}^{n} \omega_{i_\mu} C_{i_\mu}$ takes its minimal value for the sequence of jobs which corresponds to nonincreasing order of coefficients ω_i: $\omega_{i_1} \geq \omega_{i_2} \geq \dots \omega_{i_n}$. So to determine the appropriate class of schedules $S_{(i_1,i_2,\dots,i_n)}$ it is necessary $O(n \log n)$ elementary operations. The complexity of constructing optimal schedule s_a of the chosen class $S_{(i_1,i_2,\dots,i_n)}$ is $O(n)$. The complexity of constructing equivalent schedule s_b for *Problem B'* using Algorithm 3 is $O(nm)$ since the number q of the states is equal to $\lceil \frac{n}{m} \rceil$. Thus the overall complexity is equal to $O(nm + n \log n)$. The schedule constructed is represented in Fig. 6,b. It is optimal for both problems $O \mid p_{ik} = 1, pmtn \mid F$ and $O \mid p_{ik} = 1 \mid F$ (with or without preemption). The problem without preemption was considered in Strusevich [10]. Our schedule s_b is the same as the schedule constructed by algorithm in [10], the complexity bound being also the same.

Thus Theorem 4 gives another foundation for algorithm in [10] and makes it possible to solve similar *Problem B'* with more complicated objective functions. For instance, function $F = \sum \omega_i C_i^{\frac{1}{k}}$ is concave if k is a positive integer, $k > 1$, and so the same algorithm works as in the case of $F = \sum \omega_i C_i$. So the schedule, represented in Fig. 6,b is optimal also for

$F = \sum \omega_i C_i^{\frac{1}{k}}$ if jobs $(J_{i_1}, J_{i_2}, \ldots, J_{i_n})$ are sequenced in nonincreasing order of coefficients ω_i.

5 The solution region of Problem A and B when n=m+1

We consider problem *Problem A'* with $n = m + 1$ and show that the set of the dense schedules can be described by a mixed system of linear restrictions which defines $(m-1)$-simplex. Due to Theorem 3 analogous result is also valid for the corresponding *Problem B'*.

Theorem 5 *With respect to the class $S_{(1,2,\ldots,n)}$ the solution region of problem A' is an $(m-1)$-simplex if $n = m + 1$.*

Proof: Construct a sequence of states for schedule $s_{(1,2,\ldots,n)}$ using Algorithm 1 and exclude nonfull states (see Fig. 7).

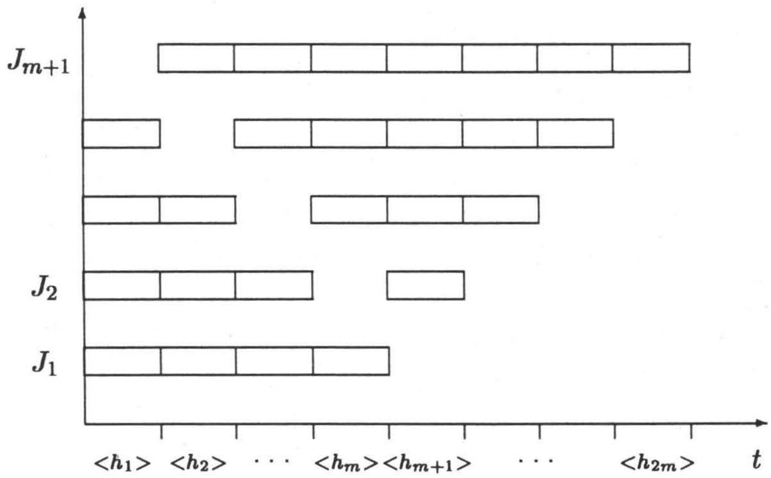


Fig. 7 *A sequence of states constructed by Algorithm 1 for Problem A'*

Corresponding system of restrictions is

$$\left\{ \begin{array}{llll} \sum\limits_{l=1}^{m} \tau_l & = & m, & \\ \sum\limits_{\substack{l=1 \\ l \neq m-k}}^{m+k} \tau_l & = & m, & k = 1, 2, \ldots, m, \\ \tau_l & \geq & 0, & l = 1, 2, \ldots, 2m. \end{array} \right. \tag{7}$$

Using first $m + 1$ equations one can construct relations for variables $\tau_m, \tau_{m+1}, \ldots, \tau_{2m}$:

$$\begin{aligned}
\tau_m &= m - \sum_{l=1}^{m-1} \tau_l, \\
\tau_{m+1} &= m - \sum_{l=1}^{m-1} \tau_l, \\
\tau_{m+2} &= \sum_{l=1}^{m-2} \tau_l + 2\tau_{m-1} - m, \\
\tau_{m+k} &= \tau_{m-k+1} - \tau_{m-k+2}, \qquad k = 3, 4, \ldots, m.
\end{aligned} \tag{8}$$

Because of the first two equalities $\tau_m = \tau_{m-1}$ holds. Substitute the above relations in inequalities $\tau_l \geq 0, \quad l = m+1, \ldots, 2m$. This gives equivalent system of restrictions:

$$\left\{ \begin{aligned}
\sum_{l=1}^{m-1} \tau_l &\leq m, \\
\sum_{l=1}^{m-2} \tau_l + 2\tau_{m-1} &\geq m, \\
\tau_{m-k+1} &\geq \tau_{m-k+2}, \quad k = 3, 4, \ldots, m.
\end{aligned} \right. \tag{9}$$

It is easy to see that inequality system (9) is equivalent to initial mixed system (7). From the first two inequalities we obtain the inequality $\tau_{m-1} \geq 0$. Taking into account the last $m-2$ inequalities of system (9) we conclude that inequalities $\tau_l \geq 0, l = 1, 2, \ldots, m-1$, of the initial system (7) which are absent in system (9) are also valid.

Since the rank of the matrix of system (9) is equal to $m-1$, the solution set of inequality system (9) is simplex. ■

The corner points of this simplex are:

τ_1	τ_2	τ_3	$\ldots$	τ_{m-3}	τ_{m-2}	τ_{m-1}
1	1	1	$\ldots$	1	1	1
$\frac{m}{m-1}$	$\frac{m}{m-1}$	$\frac{m}{m-1}$	$\ldots$	$\frac{m}{m-1}$	$\frac{m}{m-1}$	$\frac{m}{m-1}$
$\frac{m}{m-2}$	$\frac{m}{m-2}$	$\frac{m}{m-2}$	$\ldots$	$\frac{m}{m-2}$	$\frac{m}{m-2}$	0
$\frac{m}{m-3}$	$\frac{m}{m-3}$	$\frac{m}{m-3}$	$\ldots$	$\frac{m}{m-3}$	0	0
			$\ldots$			
$\frac{m}{3}$	$\frac{m}{3}$	$\frac{m}{3}$	$\ldots$	0	0	0
$\frac{m}{2}$	$\frac{m}{2}$	0	$\ldots$	0	0	0
m	0	0	$\ldots$	0	0	0.

The remaining variables τ_k, $m \leq k \leq 2m$ can be calculated by means of (8).

Let us now consider traditional scheduling objective functions which are symmetric: C_{max} and $\sum C_i$. *Problem A'* and *Problem B'* can be easily solved for C_{max} if one interchanges the set of jobs $\mathcal{N}$ and the set of machines $\mathcal{M}$. It gives the equivalent problems for which $n < m$. In this case the dense schedule of *Problem A'* consists of only one state with all jobs being processed. Corresponding dense schedule for *Problem B'* can be obtained by applying Algorithm 3, the overall complexity of constructing optimal schedule being $O(nm)$.

For *Problem A'* and *Problem B'* with $n = m+1$ and criterion $F = \sum C_i$ the following property is valid.

Theorem 6 *If $F = \sum C_i$, then all points of simplex (9) correspond to optimal schedules of* Problem A' *(or* Problem B'*).*

Proof: We calculate completion times C_i of all jobs of $\mathcal{N}$ using expressions (8). Summarizing these values we obtain that $F = \sum_{i=1}^{n} C_i = m^2 + 2m$. It means that the value of F does not depend on $\tau_1, \ldots, \tau_{m-1}$, and so any point of the simplex defines an optimal schedule. ∎

6 Concluding Remarks

Investigations of polyhedral approaches to machine scheduling problems started ten years ago with the basic paper from Balas [1]. Afterwards a great number of publications appeared in this topic. Queyranne and Schulz give an extensive overview on the results obtained in these years in [9] with 104 references. These investigations are devoted mainly to single-machine scheduling problems without preemption. The scheduling polyhedron is defined as the convex hull of the vectors corresponding to feasible schedules. All these approaches differs from the results presented in this paper in the following:

- - The scheduling polyhedron describes all feasible schedules, and it is unbounded.
  - The solution region describes a bounded subset of schedules, which contains "best" schedules.
- - In the frameworks of polyhedron approach a feasible schedule is represented by a vector of the job completion times.
  - We represent a schedule as a sequence of the so-called "states" and this gives a possibility to describe not only single-machine schedules, but the open-shop schedules as well.
- - Scheduling polyhedron describes nonpreemptive schedules.
  - In our approach points of the solution region correspond to feasible schedules with or without preemption.

Finally, the sphere of application of our approach is not restricted to unit-time problems only. In future we are going to describe a polynomial-time algorithm for minimizing linear and quadratic objective functions for the problem $O|n = k, Pmtn|F$ when number of jobs n is limited and number of machines m is nonrestricted.

References

[1] BALAS E.: *On the facial structure of scheduling polyhedra*, Mathematical Prograqmming Study, 1985, 24, P. 179-218.

[2] BÁRÁNY I., FIALA T.: *Nearly optimum solution of multimachine scheduling problems*, Szigma, 1982, 15, P. 177–191 (in Hungarian).

[3] BRUCKER P., JURISCH B., JURISCH M.: *Open shop problems with unit time operations*, ZOR, 1993, 37, P. 59-73.

[4] BRUNO J., COFFMAN E.G.J., SETHI R.: *Scheduling independent tasks to reduce mean finishing time*, Comm. ACM, 1974, 17 (7), P. 382-387.

[5] DU J., LEUNG J. Y.-T.: *Minimizing mean flow time in two-machine open shops and flow shops*, Journal of Algorithms, 1993, 14, P. 24-44.

[6] GRAHAM R.E., LAWLER E.L., LENSTRA J.K., RINNOOY KAN A.H.G.: *Optimization and approximation in deterministic sequencing and scheduling: A survey*, Ann. Discrete Math., 1979, 5, P.287-326.

[7] LIVSHITS A.M., RUBLINECKY V.I.: *On comparitive complexity of some discrete optimization problems*, Computational Mathematics and Computers, Kharkov, 1972, 3, P. 78-85 (in Russian).

[8] MCNAUGHTON R.: *Scheduling with deadlines and loss functions*, Manag. Sci., 1959, 6, N 1, P.1-12.

[9] QUEYRANNE M., SCHULZ, A.S.: *Polyhedral Approaches to Machine Scheduling*, Technische Universität, Berlin Report Nr.408/1994.

[10] STRUSEVICH V.A. *Minimizing sum of completion times for the open shop*, Vestnik BGU, Seria 1, fiz., mat., mekch, 1988, V.1, P. 44-46 (in Russian).

[11] TANAEV V.S.: *Preemptions in deterministic processing systems with parallel identical machines*, Vestsi AN BSSR. Seria fiz.-mat. navuk, 1973, V.6, P. 44-48 (in Russian).

[12] TAUTENHAHN T.: em Open-shop Probleme mit Einheitsbearbeitungszeiten, PhD-Dissertation, Fakultät für Mathematik, Otto-von-Guericke Universität, Magdeburg, 1993 (in German).