Math. Meth. Oper. Res. (2007) 65: 461–481
DOI 10.1007/s00186-006-0127-8

ORIGINAL ARTICLE

Peter Brucker · Yu N. Sotskov
Frank Werner

Complexity of shop-scheduling problems with fixed number of jobs: a survey

Received: 3 December 2004 / Accepted: 10 October 2006 / Published online: 6 February 2007

Abstract The paper surveys the complexity results for job shop, flow shop, open shop and mixed shop scheduling problems when the number n of jobs is fixed while the number r of operations per job is not restricted. In such cases, the asymptotical complexity of scheduling algorithms depends on the number m of machines for a flow shop and an open shop problem, and on the numbers m and r for a job shop problem. It is shown that almost all shop-scheduling problems with two jobs can be solved in polynomial time for any regular criterion, while those with three jobs are NP-hard. The only exceptions are the two-job, m-machine mixed shop problem without operation preemptions (which is NP-hard for any non-trivial regular criterion) and the n-job, m-machine open shop problem with allowed operation preemptions (which is polynomially solvable for minimizing makespan).

1 Introduction

In 2004, there was the 50th anniversary of the publication of the first polynomial algorithm given by Johnson (1954) for a flow shop problem with two machines and makespan minimization. In Johnson (1954) and in the majority of the subsequent papers on scheduling theory basically systems with a *fixed number of machines* and an unrestricted number of jobs have been investigated. In this survey, we present

P. Brucker (✉)
Universität Osnabrück, Fachbereich Mathematik, Osnabrück, 49069 Germany
E-mail: Peter.Brucker@uni-osnabrueck.de

Y. N. Sotskov
United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Surganov St. 6, Minsk 220012, Belarus

F. Werner
Otto-von-Guericke-Universität, Fakultät für Mathematik, PSF 4120, Magdeburg 39016, Germany

all known complexity results for shop-scheduling problems with a *fixed number of jobs* and an unrestricted number of machines or operations per job. The most important results obtained for a job shop problem with $m \geq n$ are presented in this survey with proofs.

The **job shop** problem may be formulated as follows. Given are n jobs $J = \{J_1, J_2, \ldots, J_n\}$ and m machines $M_1, M_2, \ldots, M_m$. Job J_i consists of a sequence $O_{1i}, O_{2i}, \ldots, O_{r_i,i}$ of r_i operations which must be processed in the given order, i.e. operation $O_{j+1,i}$ cannot start before operation O_{ji} is completed, j=1, 2, ..., $r_i - 1$. Associated with each operation O_{ji} there are a processing time $p_{ji} > 0$ and a machine $\mu_{ji} \in \{M_1, M_2, \ldots, M_m\}$. Operation O_{ji} must be processed for p_{ji} time units on machine μ_{ji}. Each job can be processed by at most one machine at a time and each machine can process at most one operation at a time. We assume that all jobs and machines are available from starting time zero. If preemption of operations is forbidden, then equality $c_{ji} = s_{ji} + p_{ji}$ must hold, where s_{ji} and c_{ji} denote the starting time and completion time of operation O_{ji}, respectively. The job shop problem is to find a schedule which minimizes the given objective function $\Phi = \Phi(C_1, C_2, \ldots, C_n)$, where C_i means the completion time of job J_i, i.e. $C_i = c_{r_i,i}$. The majority of known results for shop-scheduling problems has been obtained for makespan minimization $\mathcal{C}_{max} = \max\{C_i : J_i \in J\}$, and for the minimization of the sum of the job completion times $\sum \mathcal{C}_i = \sum_{i=1}^{n} C_i$.

Depending on the type of job routes through m machines, which may be fixed or not before scheduling, we obtain different classes of shop-scheduling problems. In a flow shop, routes for all jobs are fixed and identical, namely: $\mu_{1i} = M_1, \mu_{2i} = M_2, \ldots, \mu_{mi} = M_m$ for each job $J_i \in J$.

In an open shop, job routes are not fixed before scheduling (and thus the choice of job routes is a part of the decision for an open shop problem). It is known only that each job has exactly one operation on each machine (the same as for a flow shop).

In a job shop, routes of all jobs are fixed and *can be* different for different jobs. Thus, a flow shop is a special case of a job shop. Note also that in a job shop, a route of a job may have machine repetition or (and) absence of some machines. Thus, in contrast to flow shop and open shop in which equality $r_i = m$ holds for each job J_i, in a job shop each of the three possibilities is allowed: $r_i < m$, $r_i = m$ or $r_i > m$.

In a mixed shop, there are n_J jobs with fixed routes (as in a job shop) and $n_O = n - n_J$ jobs whose routes are not fixed (as in an open shop). However, equality $r_i = m$ must hold for each job in a mixed shop.

For the classification of scheduling problems the three-field form $\alpha|\beta|\gamma$ is used, where α characterizes the type of the processing system and the number of machines. Thus, symbol F is used for the notation of a flow shop problem, O for an open shop problem, J for a job shop problem and X for a mixed shop problem. The parameter β defines a set of job constraints. The position γ specifies the objective function. E.g. Johnson's algorithm [1] has been developed for a flow shop problem with two machines and minimization of makespan, i.e. for problem $F2||\mathcal{C}_{\max}$.

As it was already mentioned, the majority of known results of scheduling theory have been obtained for the case of a fixed number of machines m when the number of jobs $n > m$ can be arbitrarily large. In particular, the asymptotical complexity of the algorithm given by Johnson (1954) is $O(n\log n)$ and, hence, it is polynomial in the number of jobs n. In this paper, we survey all known results for the case of a fixed number of jobs which define the boundary between easy and hard problems. The complexity of the considered algorithms depends on the number m for flow and open shop problems and on m and $r = \max\{r_i : J_i \in J\}$ for job shop problems.

Along with the theoretical significance of the case $n > m$, it has practical importance, in particular, for scheduling problems arising in modern microelectronics when it is necessary to design a specific processing system with a fixed set of jobs, where a rather cheap standard equipment (machines) can be used.

The paper is organized as follows. In Sect. 2, we present a table defining the boundary between NP-hard and polynomially solvable job shop problems with a fixed number of jobs. Sections 3 and 4 describe polynomial algorithms for job shop problems with two jobs. Sections 4, 5 and 6 describe polynomial algorithms for job shop problems with two machines and a fixed number of jobs. NP-hardness of the job shop problem with $n = m = 3$ without operation preemption (with $n = 3, m = 2$ with operation preemption) is proven in Sect. 7 (Sect. 8). In Sects. 9 and 10 we present tables defining the boundary between NP-hard and polynomially solvable problems of flow shop, open shop and mixed shop type. Concluding remarks are given in Sect. 11. An appendix contains the NP-hardness proof of problem $J2 \mid n = 3, pmtn|C_{\max}$. Hereafter, 'pmtn' means allowance of operation preemption.

2 Job shop problems

In Table 1, we present known results concerning the complexity of algorithms for job shop problems with an unrestricted number of operations per job. Besides the notations given in the previous section, in Table 1 the following notation is used: $L = \max\{L_i : J_i \in J, L_i = \sum_{j=1}^{r_i} p_{ji}\}$. The symbol $[p_{ji}]$ means that processing times of all operations should be natural numbers. The symbol MPM in the fourth column specifies that the processing times of all operations and the set of machines on one of which the operation can be processed. In rows 11 and 12 of Table 1 any non-decreasing function $f_i(C_i)$ can be considered.

In the last column of Table 1, publications are listed in which the corresponding algorithms are given or NP-hardness of the problem is proved. Since a number of the results has been independently obtained by different authors, we present the papers in an increasing order of their years of publication.

3 Two jobs without preemptions

Polynomial algorithms which construct optimal schedules for two jobs (see rows 1, 3–8 in Table 1) are based on a geometrical representation of schedules for problems $J \mid n = 2|C_{\max}$, which has been originally suggested in Akers (1956) and Akers and Friedman (1956) and developed in Brucker (1988); Szwarc (1960) and Hardgrave and Nemhauser (1963). In Sotskov (1985, 1991), the geometric algorithm was generalized for problems $J \mid n = 2|\Phi$ and $J \mid n = 2, pmtn|\Phi$. Next, we describe this approach for problem $J \mid n = 2|C_{\max}$ and prove its efficiency.

The job shop problem with two jobs may be formulated as a shortest path problem in the plane with regular objects as obstacles. Figure 1 shows a shortest

Table 1 Complexity of job shop problems

Row	Number of jobs	Number of machines	Constraints	Criterion	Complexity	References
1	2	m	$r \leq m$	$C_{\max}$	$O(m^2)$	Szwarc (1960), Hardgrave and Nemhauser (1963)
2	2	m		Φ	$O(r^2 \log_2 r)$	Sotskov (1985, 1991)
3	2	m	$pmtn$	Φ	$O(r^3)$	Sotskov (1985, 1991)
4	2	m		Φ	$O(m \log_2 m)$	Sotskov (1985, 1991)
5	2	m	$pmtn, r \leq m$	Φ	$O(m^2)$	Sotskov (1985, 1991)
6	2	m		$C_{\max}$	$O(r^2 \log_2 r)$	Sotskov (1985),Brucker (1988)
7	2	m	MPM	$C_{\max}$	$O(r^3)$	Brucker and Schlie (1990)
8	n	m	$[p_{ji}]$	$C_{\max}$	$O(2^n L^n)$	Servach (1983), Brucker et al. (1989)
9	3	2		$C_{\max}$	$O(r^4)$	Kravchenko and Sotskov (1996)
10	k	2		$C_{\max}$	$O(r^{2k})$	Brucker (1994)
11	k	2		$\max f_i(C_i)$	$O(r^{k^2+2k})$	Brucker et al. (1994)
12	k	2		$\sum f_i(C_i)$	$O(r^{k^2+2k})$	Brucker et al. (1994)
13	k	2		$\sum w_i C_i$	$O(r^k)$	Brucker et al. (1994)
14	3	5	$pmtn$	$\sum C_i$	Binary NP-hard	Sotskov (1989, 1990, 1991)
15	3	5		$C_{\max}$	Binary NP-hard	Sotskov (1989, 1990, 1991)
16	3	5	$pmtn$	$C_{\max}$	Binary NP-hard	Sotskov (1989, 1990, 1991)
17	3	5		$\sum C_i$	Binary NP-hard	Sotskov (1989, 1990, 1991)
18	3	3	$pmtn$	$\sum C_i$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
19	3	3		$C_{\max}$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
20	3	3	$pmtn$	$C_{\max}$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
21	3	3		$\sum C_i$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
22	3	2	$pmtn$	$C_{\max}$	Binary NP-hard	Brucker et al. (1999)
23	3	2	$pmtn$	$\sum C_i$	Binary NP-hard	Brucker et al. (1999)

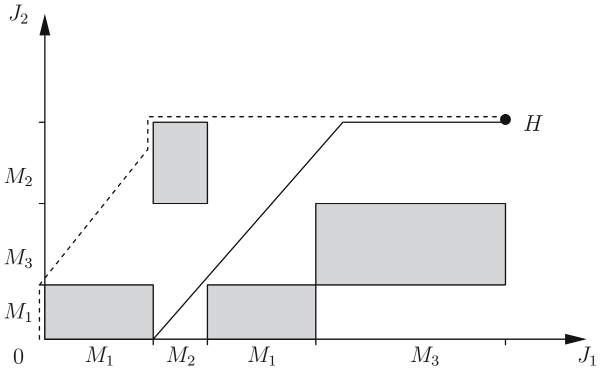

Fig. 1 Path problem with obstacles

path problem with obstacles which corresponds to a job shop problem with two jobs and $r_1 = 4$ and $r_2 = 3$. The processing times of the operations of job J_1 (job J_2) are represented by intervals on the x-axis (y-axis) which are arranged in the order in which the corresponding operations are to be processed. Furthermore, the intervals are labelled by the machines on which the corresponding operations must be processed.

A feasible schedule corresponds to a path from initial vertex $O = (0, 0)$ to terminal vertex H. This path has the following properties:

- the path consists of segments which are either parallel to one of the axes (an operation of only one job is processed) or diagonal (operations of both jobs are processed in parallel);
- the path has to avoid the interior of any rectangular obstacle of the form $I_1 \times I_2$, where I_1 and I_2 are intervals on the x-axis and y-axis which correspond to the same machine (this follows from the fact that two operations cannot be processed simultaneously on the same machine and that operation preemption is not allowed);
- the length of the path, which corresponds to the schedule length, is equal to:

length l of horizontal parts plus length of vertical parts plus (length of diagonal parts)/$\sqrt{2}$.

In general, we have consecutive intervals $I_\nu^x (I_\nu^y)$ of length $p_{\nu 1}(p_{\nu 2})$ where $\nu = 1, 2, \ldots, r_1(r_2)$. Furthermore, the rectangles $I_i^x \times I_j^y$ are forbidden regions if and only if $\mu_{i1} = \mu_{j2}$. Finally, define

$$a = \sum_{\nu=1}^{r_1} p_{\nu 1} \quad \text{and} \quad b = \sum_{\nu=1}^{r_2} p_{\nu 2}.$$

We have to find a shortest path from vertex $O = (0, 0)$ to vertex $H = (a, b)$ which has only horizontal, vertical, or diagonal segments and which does not pass through the interior of any of the forbidden regions. Next, we show that the problem can be reduced to an unrestricted shortest path problem in an appropriate network N.

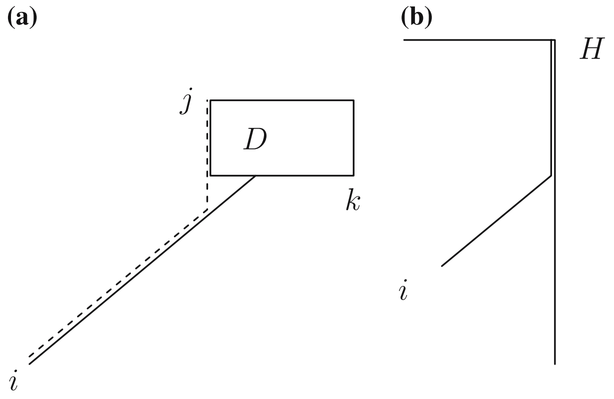

Fig. 2 Definition of arcs

Furthermore, the unrestricted path problem may be solved in time which is linear in the number of obstacles.

The network $N = (V, A, l)$ is constructed as follows. The set of vertices V consists of O, H, and the set of north-west corners (NW-corners) and south-east corners (SE-corners) of all obstacles. O is considered as a degenerate obstacle in which both the NW-corner and the SE-corner coincide. Each vertex $i \in V\backslash\{H\}$ coincides with at the most two arcs going out of i. To construct these arcs, we go from i diagonally in a NE direction until we hit either the boundary of the rectangle defined by O and H or the boundary of an obstacle. In the first case, H is the only successor of i and arc (i, H) consists of the path which goes from i diagonally to the boundary and continues along the boundary to H (see Fig. 2). If we hit an obstacle D, then there are two arcs (i, j) and (i, k) where j and k are the NW-corner and SE-corner of D, respectively. The corresponding polygons are shown in Fig. 2.

The length $l(i, j)$ of an arc (i, j) is equal to the length of the vertical or horizontal piece plus the length of the projection of the diagonal piece on the x-axis (or equivalently, y-axis).

It is easy to see that an O-H-path (i.e. a path from O to H) in $N = (V, A, l)$ corresponds to a feasible schedule, and its length is equal to the corresponding $C_{\max}$-value. Furthermore, an optimal schedule corresponds with a shortest $O-H$-path in N.

The network N is acyclic and has $O(s)$ arcs where s in the number of obstacles. Thus, a shortest O-H-path can be calculated in time $O(s)$. In general $s = r^2$ where $r = \max\{r_1, r_2\}$.

To construct the network $N = (V, A, l)$ we apply a line sweep with the SW–NE-line $y-x = c$. We move this line parallel to itself from north-west to south-east. The sweep line intersects the obstacles creating an ordered set of intervals. Let S be the corresponding set of obstacles, together with the order induced by the intervals on the sweep line. We keep track of changes in S during the sweep. There are two possible events where changes occur.

- If the sweep line hits an NW-corner i of an obstacle D_l, then we have to insert D_l into the ordered set S. If there is a next element D_h in S, which has an NW-corner j and a SE-corner k, then the arcs (i, j) and (i, k) must be inserted

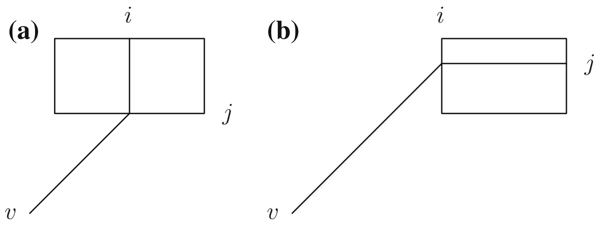

Fig. 3 Going through obstacles

into A and we have to calculate $d(i, j)$ and $l(i, k)$. Otherwise, we have to insert (i, H) into A and calculate $l(i, H)$.
- The sweep line hits a SE-corner i of an obstacle D_l. If there is a next element D_h in S with an NW-corner j and a SE-corner k, then we insert (i, j) and (i, k) into A and calculate $l(i, j)$ and $l(i, k)$. Otherwise, we have to insert (i, H) into A and calculate $l(i, H)$. Finally, D_l must be deleted from S.

The arcs (i, j), (i, k) and the distances $d(i, j)$, $d(i, k)$ can be calculated in constant time. Furthermore, if we use a balanced tree, e.g. a 2–3 tree (see Aho et al. (1974)), then the insert and delete operations can be done in $O(\log r)$ time. Thus, N can be constructed in $O(r^2 \log r)$ time.

4 Two jobs with preemptions

Next, we describe the modification of the above algorithm for problem $J \mid n = 2, pmtn \mid C_{\max}$. If operation preemption is allowed in the previous problem, then we may go horizontally or vertically through obstacles. For this reason, the network $N = (V, A, l)$ must be defined differently.

The vertex set is defined recursively as follows:

- O is a vertex.
- If v is a vertex and the half-line starting from v and going in a NE-direction hits an obstacle, then v has the two successors i and j shown in Fig. 3a or b.
- If v is a vertex and the half-line starting from v and going in a NE-direction hits the borderline given by the rectangle defined by O and H, then H is the only successor of v.

We denote the modified network by $\overline{N}$.

Arcs (i, j) and arc lengths $d(i, j)$ are defined as before in Sect. 3. If in the above algorithm for problem $J \mid n = 2 \mid C_{\max}$ the “NW-corner” and “SE-corner” are replaced by “north boundary point” and “east boundary point”, respectively, this modified algorithm applied to $\overline{N}$ solves problem $J \mid n = 2, pmtn \mid C_{\max}$. The running time of this modified algorithm is bounded by $O(r^3)$. This can be seen as follows.

Consider for each SE-corner or NW-corner v, which can be reached from O, the unique path starting from v which avoids the boundaries of the obstacles it hits. If such a path hits the SW-corner t of an obstacle (or a boundary point t' of the rectangle R, defined by O and H), this path terminates in t (or in H). There are at

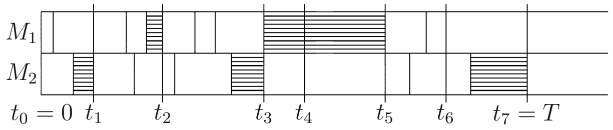

Fig. 4 Block structure of a schedule

most $O(r_1, r_2)$ of these paths and each has at the most $r_1 + r_2$ arcs. Furthermore, the set of all these paths covers all arcs going through the interior of obstacles. Thus, the total number of arcs which go through the interior of obstacles is bounded by $O(r^3)$. Because the number of arcs not going through the interior of obstacles is bounded by $O(r_1 r_2)$ and the total computation time of the shortest path algorithm is proportional to the number of all arcs, we have an $O(r^3)$-algorithm.

Finally, we remark that all of the geometric methods also work if we allow repetition of machines.

5 Two machines, no preemptions

It is necessary to note that job shop problems with a fixed number of jobs are not NP-hard in the strong sense (unary NP-hard) since there exists a pseudo-polynomial algorithm for the decision problem, for example, in Servach (1983) and Brucker et al. (1989) (see row 8 in Table 1). The algorithms specified in rows 10–13 of Table 1 are polynomial (since k represents a constant). However, these algorithms will be really effective only for sufficiently small values k and r.

We give a shortest path formulation for the two-machine job shop problem with k jobs and makespan objective. The corresponding network is acyclic and has at the most $O(r^k)$ vertices. It can be constructed in time $O(r^{2k})$. Thus, the job shop problem can be solved in $O(r^{2k})$ time. We consider only *active* schedules, i.e. schedules in which no operation can be started earlier without violating feasibility. Each schedule can be transformed into an active one without increasing the $C_{\max}$-value. Given an active schedule S, we have a unique sequence $t_0 = 0 < t_1 < t_2 < \cdots < t_q$ of all times at which either two operations begin processing jointly on both machines or one operation begins processing on one machine while an idle period is starting on the other machine (see Fig. 4). We define $t_{q+1} = T$ where T is the $C_{\max}$-value of the schedule.

Furthermore, we call the set D^ν of operations scheduled in the interval $[t_\nu, t_{\nu+1}]$ $(\nu = 0, 1, \ldots, q)$ a **block**. For a block D, let D^1 (D^2) be the set of operations of D processed on machine M_1 (machine M_2) and denote the sum of processing times of all operations in D^1 (D^2) by $l_D^1 (l_D^2)$. A block D associated with the interval $[t_i, t_{i+1}]$ has the properties that

- all jobs in D^1 (D^2) are scheduled in the interval $[t_i, t_i + l_D^1]$ (interval $[t_i, t_i + l_D^2]$),
- $t_{i+1} = t_i + \max\{l_D^1, l_D^2\}$, and
- one machine is idle in the interval $[\min\{l_D^1, l_D^2\}, t_{i+1}]$.

The value $l_D = \max\{l_D^1, l_D^2\} = t_{i+1} - t_i$ is called the **length** of block D.

It follows that a schedule is defined by a sequence of blocks and the schedule length is the sum of the lengths of all blocks in that sequence.

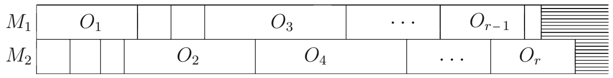

Fig. 5 Decomposition of a block

To formulate the job shop problem as a shortest path problem in some network N we characterize blocks in the following way. For a given block, let us consider the vector $\mathbf{u} = (u(j))_{j=1}^{k}$ where $u(j)$ is the index of the last operation $O_{u(j),j}$ of job j which is processed in the previous block. Thus, the block can be described by pairs $(\mathbf{u}, \mathbf{v})$ of index tuples. The network $N = (V, A, l)$ is defined by

- the set V of all index tuples $\mathbf{u} = (u(j))$ with $0 \le u(j) \le n_j$ for $j = 1, 2, \ldots, k$. The vector $\mathbf{s}$ with $s(j) = 0$ for all $j = 1, 2, \ldots, k$ is the **initial vertex**, the vector $\mathbf{t}$ with $t(j) = n_j$ for all $j = 1, 2, \ldots, k$ is the **terminal vertex** of the network,
- $(\mathbf{u}, \mathbf{v}) \in A$ if and only if $\mathbf{u} < \mathbf{v}$ and the set of operations

$$\{O_{i(j),j} \mid j = 1, 2, \ldots, k;\ u(j) < i(j) \le v(j)\}$$

  defines a block, and
- for each $(\mathbf{u}, \mathbf{v}) \in A$, its length $l(\mathbf{u}, \mathbf{v})$ is the length of the block corresponding to $(\mathbf{u}, \mathbf{v})$.

In network N, each $\mathbf{s}$–$\mathbf{t}$-path p corresponds to a feasible schedule and the length of p is equal to the makespan of this schedule. We obtain this schedule by scheduling the blocks corresponding to the arcs in p consecutively as shown in Fig. 4. Furthermore, there is an $\mathbf{s}$–$\mathbf{t}$-path corresponding to an optimal schedule. Thus, to solve the job shop problem we have to find a shortest $\mathbf{s}$–$\mathbf{t}$-path in N.

It remains to show that the network can be constructed in time $O(r^{2k})$. For this purpose we will present an $O(r^k)$-algorithm which, for a given vertex $\mathbf{u}$, finds all arcs $(\mathbf{u}, \mathbf{v})$ which define a block. W.l.o.g. let $\mathbf{u} = \mathbf{s}$ where $\mathbf{s}$ is the initial vertex defined in the previous section.

A block may be decomposed as shown in Fig. 5. This decomposition is defined by operations $O_i (i = 1, 2, \ldots, r)$ with the property that O_i with odd (even) index are processed on machine M_1 (machine M_2) and each pair of consecutive operations O_i and O_{i+1} are overlapping. The O_i are called **main operations**.

We construct a directed graph G with $O(r^k)$ vertices and $O(kr^k)$ arcs such that all blocks starting in $\mathbf{s}$ can be found by visiting all vertices in this graph. This can be done in $O(kr^k)$ time.

G consists of $\mathbf{s}$ and of noninitial vertices defined by pairs $(\mathbf{u}, h)$ of index-tuples $\mathbf{u} = (u(j))$ together with a job index h defining a main operation $O_{u(h),h}$. Vertex $(\mathbf{u}, h)$ defines a block consisting of all operations O_{ij} with $i \le u(j)$ where $j = 1, 2, \ldots, k$. Vertex $(\mathbf{v}, l)$ is a successor of vertex $(\mathbf{u}, h)$ if $\mathbf{v}$ is derived from $\mathbf{u}$ by adding a single operation, i.e. by incrementing one index $u(j)$ by one. If a main operation is added, then we have $h \neq j = l$. In this case, $O_{u(j),l}$ must finish later than $O_{u(h),h}$. Otherwise, $l = h$ and the added operation $O_{u(j),j}$ must finish earlier than the main operation $O_{u(h),h}$. The successors of $\mathbf{s}$ are the vertices representing blocks consisting only of a main operation. Clearly, each vertex has at most k successors. Thus, the graph can be searched in $O(kr^k)$ time which is $O(r^k)$ if k is fixed. Therefore, the network N can be constructed in time $O(r^{2k})$.

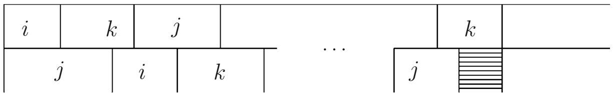

Fig. 6 Block decomposition for $n = 3$ and no machine repetition

6 Three jobs, two machines, no machine repetition

We use the previous algorithm described in Sect. 5 to solve the special case with three jobs and no machine repetition. Due to the fact that no machine repetition is allowed blocks have a special structure which may be characterized by the following two properties (see Fig. 6) which can be easily verified:

- with the exception of a first and a last operation, each operation in the block overlaps exactly two other operations;
- the job indices are repeated periodically, i.e. the operations ordered according to increasing starting times yield a sequence O_{ri}, O_{sj}, O_{tk}, $O_{r+1,i}$, $O_{s+1,j}$, $O_{t+1,k}$, $O_{r+2,i}$, ...

Notice, that a block may also consist of exactly one operation.

For a block B there are at most three one-element blocks which may be a successor of B. In a successor block of B containing more than one operation, two operations begin at the same time. Given these two operations, there is a unique maximal block satisfying conditions (a) and (b) and this maximal block can be constructed in $O(r)$ time. Each initial part of this maximal block may be a successor of B. Due to the fact that there are only these two possibilities to start a successor block of B, it follows that block B has at most $O(r)$ successors and these successors can be constructed in $O(r)$ time.

This implies that the network has at most $O(r^4)$ arcs and can be constructed in $O(r^4)$ time. Thus, we have an $O(r^4)$-algorithm.

Surprisingly, the job shop problem with two machines and three jobs is NP-hard if we allow machine repetition. This is shown in Sect. 8 and the appendix.

7 Three jobs

In this section, we show how to prove that problem $J3 \mid n = 3 \mid C_{\max}$ is binary NP-hard.

The complexity of problem $J3 \mid n = 3 \mid C_{\max}$ is determined through a reduction from the following version of the partitioning problem, known as even–odd partition.

PARTITION: Given positive integers e_i $(i = 1, 2, \ldots, 2k)$ with $\sum_{i=1}^{2k} e_i = 2E = \Delta$. Does there exist a partition of the index set $A = \{1, 2, \ldots, 2k\}$ into subsets A_1 and A_2 such that $\sum_{i \in A_1} e_i = E$ and A_1 includes exactly one element from each pair $(2i - 1, 2i)$ where $i = 1, 2, \ldots, k$?

For problem $J3 \mid n = 3 \mid C_{\max}$ we first allow repetition of machines.

Theorem 1 *Problem* $J3 \mid n = 3 \mid C_{\max}$ *is binary* NP*-hard.*

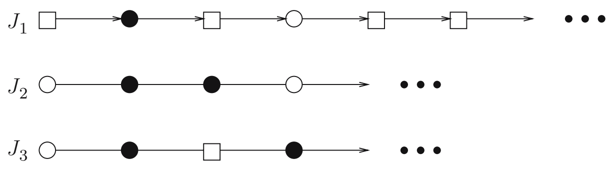

Fig. 7 $J3 \mid n = 3 \mid C_{\max}$-instance

Proof Given an instance of PARTITION the corresponding instance of $J3 \mid n = 3 \mid C_{\max}$ is constructed as follows. Let H be an integer with $H > 8E$. The structure of the $J3 \mid n = 3 \mid C_{\max}$-instance can be divided into k identical periods. One period is shown in Fig 7. Operations to be processed on M_1, M_2, and M_3 are represented by white circles, black circles, and squares, respectively.

The processing times of the corresponding operations are as follows:

$$\begin{aligned}
&p_{6l+1,1} = \sum_{\nu=-1}^{l-1} \Delta_{2\nu+1,2\nu+2},\ p_{6l+2,1} = H - \sum_{\nu=-1}^{l-1} \Delta_{2\nu+1,2\nu+2},\\
&p_{6l+3,1} = H + \sum_{\nu=-1}^{l-1} \Delta_{2\nu+1,2\nu+2} + e_{2l+2},\ p_{6l+4,1} = \Delta_{2l+1,2l+2},\\
&p_{6l+5,1} = 2H - \Delta_{2l+1,2l+2},\ p_{6l+6,1} = 2H,\\
&p_{4l+1,2} = H,\ p_{4l+2,2} = 2H,\ p_{4l+3,2} = 2H,\ p_{2l+4,2} = H,\\
&p_{4l+1,3} = e_{2l+1},\ p_{4l+2,3} = e_{2l+2},\ p_{4l+3,3} = \Delta_{2l+1,2l+2},\ p_{4l+4,3} = H,\\
&\text{where} \quad \Delta_{j,j+1} = e_j - e_{j+1},\ \Delta_{-1,0} = 0 \text{ and } l = 0, 1, \ldots, k-1.
\end{aligned}$$

It can be shown that problem PARTITION has a solution if and only if the corresponding $J3 \mid n = 3 \mid C_{\max}$-instance has a solution with $C_{\max} \leq 6kH + E$. □

By adding separating operations with very small processing times $\varepsilon > 0$ the NP-hardness proof also for $J3 \mid n = 3 \mid C_{\max}$ without machine repetition can be accomplished.

8 Two machines, three jobs with preemptions

We now consider the preemptive two-machine version of the problem discussed in Sect. 7. This problem is usually denoted by $J2 \mid n = 3, pmtn \mid C_{\max}$. First, we will show that this problem can be solved pseudopolynomially. This follows from the fact that problem $Jm \mid n = k, pmtn \mid C_{\max}$ can be solved in time $O(rT^{k-1})$ where

$$T = \max_{j=1}^{k} \sum_{i=1}^{n_j} p_{ij} \quad \text{and} \quad r = \max_{j=1}^{k} n_j.$$

In the second part we state the main result of this section, namely problem $J2 \mid n = 3, pmtn \mid C_{\max}$ is NP-hard.

We start with an observation which is essential for the pseudopolynomial algorithm for problem $Jm \mid n = k, pmtn \mid C_{\max}$ and which is also useful for the NP-hardness proof.

Given a schedule S, a **slice** of S is a maximal time period $[t, t']$ such that during the whole period $[t, t']$ each machine is either idle or continuously processes exactly one operation. Clearly, a schedule S **decomposes** into a sequence of slices, i.e. if $C_{\max}$ is the makespan of S, then there are unique slices $[t_0, t_1]$, $(t_i, t_{i+1}](i = 1, 2, \dots, h-1)$ with $t_0 = 0 < t_1 < \cdots < t_h = C_{\max}$.

Lemma 1 *For problem $J \mid pmtn \mid C_{\max}$ there exists an optimal schedule which has a slice decomposition $[t_0, t_1]$, $(t_i, t_{i+1}]$ $(i = 1, 2, \dots, h-1)$ such that at each time $t_i (i = 1, 2, \dots, h)$ at least one operation finishes.*

This result follows from the fact that an arbitrary schedule can be transformed into a schedule satisfying the required conditions without increasing the objective function.

Next, we show that problem $Jm \mid n = k, pmtn \mid C_{\max}$ can be formulated as a problem of finding a shortest path in an acyclic network with $O(rT^{k-1})$ arcs. Thus, the problem can be solved in time $O(rT^{k-1})$.

The construction of the network N is based on the fact, that there exists an optimal schedule which decomposes into slices such that each slice contains a finishing operation, i.e. for a slice $(t_i, t_{i+1}]$ there exists a job J_j such that some operation O_{uj} finishes at time t_{i+1}.

We may associate with t_{i+1} an operation O_{uj} which finishes at time t_{i+1}. Furthermore, for each job J_ν, $\nu = 1, 2, \dots, k$, we denote by s_ν the total processed time of job J_ν at time t_{i+1}. Note that $s_j = \sum_{i=1}^{u} p_{ij}$. Therefore, we associate with t_{i+1} a vertex

$$v = (O_{uj}, s_1, \dots, s_{j-1}, s_{j+1}, \dots, s_k)$$

in the network N. There are at most $rkT^{k-1} = O(rT^{k-1})$ vertices. A vertex

$$v' = (O_{l'j'}, s'_1, \dots, s'_{j'-1}, s'_{j'+1}, \dots, s'_k)$$

is a successor of vertex v in network N, if (v, v') defines a slice. $O_{l'j'}$ is a finishing operation of this slice and

$$l_{v,v'} = \begin{cases} \sum_{\nu=1}^{l'} p_{\nu j'} - s_{j'} & \text{if } j' \neq j \\ p_{l'j'} & \text{otherwise} \end{cases}$$

is its length.

Each vertex has at most $k \cdot \binom{k}{m}$ successors which is a constant number if m and k are fixed. Thus, network N has $O(rT^{k-1})$ arcs.

Notice that there are several sources and sinks. Each path from a source to a sink in network N corresponds with a feasible schedule and the length of the path corresponds with the makespan of the corresponding schedule. Due to Lemma 1 there exists an optimal schedule which is represented by a path in network N. Thus, to solve problem $Jm \mid n = k, pmtn \mid C_{\max}$ we have to find a shortest path from a source to a sink in N which can be done in $O(rT^{k-1})$ time.

Again, the problem PARTITION will be reduced to problem $J2 \mid n = 3 \mid C_{\max}$. The corresponding reduction is similar to the reduction from PARTITION to $J3 \mid n = 3, pmtn \mid C_{\max}$. The proof of the following theorem is given in the appendix.

Theorem 2 *Problem $J2 \mid n = 3, pmtn \mid C_{\max}$ is binary NP-hard.*

An immediate consequence of the results in this section is that the job shop problem with two machines, three jobs, and unit processing times is binary NP-hard if we allow machine repetition.

The investigation of job shop problems with a fixed number of jobs has led to an unexpected result concerning the influence of preemptions of an operation. For the first time, a problem has been specified for which there exists a polynomial algorithm for the case when preemptions are forbidden while it is NP-hard in the case of allowed preemptions (see rows 11, 12 and rows 22, 23 in Table 1). Before the publication of Brucker et al. (1994, 1999), for all investigated problems for which a polynomial algorithm exists in the case of forbidden operation preemptions, there exists also a polynomial algorithm in the case of allowed operation preemptions.

9 Flow shop and open shop

Since a flow shop is a special case of a job shop, the polynomial algorithms given in Table 1 for job shop problems can also be used for determining an optimal solution for a flow shop. Therefore, in Table 2 only those algorithms are presented which have a smaller complexity in comparison with the algorithms for the corresponding job shop problem.

In Gonzalez and Sahni (1976), it has been proven that problem $O3 \mid\mid C_{\max}$ is NP-hard and an algorithm of complexity $O(m)$ for problem $O2 \mid\mid C_{\max}$ as well as an algorithm of complexity $O(n^2m^2)$ for problem $O \mid pmtn \mid C_{\max}$ have been developed. Based on these results, one can conclude by taking into account the symmetry of jobs and machines for problems $O \mid\mid C_{\max}$ and $O \mid pmtn \mid C_{\max}$ that problem $O \mid n = 3 \mid C_{\max}$ is NP-hard while problems $O \mid n = 2 \mid C_{\max}$ and $O \mid n = k, pmtn \mid C_{\max}$ are polynomially solvable.

It is necessary to note that for criterion $\sum C_i$ and other classical criteria open shop problems do not possess the symmetry property of jobs and machines. Nevertheless, in Shakhlevich and Strusevich (1990) an algorithm of linear time in the number of machines has been suggested for problems $O \mid n = 2 \mid \Phi$ and $O \mid n = 2, pmtn \mid \Phi$ for an arbitrary regular optimality criterion.

10 Mixed shop

In paper Masuda et al. (1985), a mixed shop problem has been considered for the first time in which $n_J \le n$ jobs of the set J have the same fixed route (as for the flow shop problem), and routes of $n_O \le n$ jobs of set O are not fixed (as for the open shop problem). It is assumed that equality $n = n_J + n_O$ holds.

In Table 3, known results about the complexity of problems $X \mid\mid \Phi$ with a fixed number of jobs are given. Number n_J means the number of jobs with fixed routes

Table 2 Complexity of algorithms for flow and open shop problems

Problem type	Number of jobs	Number of machines	Constraints	Crite-rion	Complexity	References
F	2	m		Φ	$O(m \log m)$	Sotskov (1985, 1991)
F	2	m	$pmtn$	Φ	$O(m^2)$	Sotskov (1985, 1991
F	3	m		$C_{\max}$	Binary NP-hard	Sotskov (1989, 1990, 1991)
F	3	m	$pmtn$	$C_{\max}$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
F	3	m		$\sum C_i$	Binary NP-hard	Sotskov (1989, 1990, 1991)
F	3	m	$pmtn$	$\sum C_i$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
F	3	m	$pmtn$, $p_{ji} > 0$	$C_{\max}$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
F	3	m	$pmtn$, $p_{ji} > 0$	$\sum C_i$	Binary NP-hard	Sotskov and Shakhlevich (1990, 1995)
O	2	m		$C_{\max}$	$O(m)$	Gonzalez and Sahni (1976)
O	n	m	$pmtn$	$C_{\max}$	$O(n^2 m^2)$	Gonzalez and Sahni (1976)
O	2	m		Φ	$O(m)$	Shakhlevich and Strusevich (1990)
O	2	m	$pmtn$	Φ	$O(m)$	Shakhlevich and Strusevich (1990)
O	3	m		$C_{\max}$	Unary NP-hard	Gonzalez and Sahni (1976), Strusevich (1986)
O	3	m		$\sum C_i$	Unary NP-hard	Sotskov and Shakhlevich (1990)

Table 3 Complexity of algorithms for mixed shop problems

Number of jobs n_J	Number of jobs n_O	Number of machines	Constraints	Criterion	Complexity	References
1	1	∞		Φ	$O(r)$	Shakhlevich and Sotskov (1994)
1	∞	2		$C_{\max}$	Unary NP-hard	Shakhlevich et al. (1999)
1	1	∞		$\sum C_i$	Binary NP-hard	Shakhlevich and Sotskov (1994)
k	1	2		$C_{\max}$	$O(r^{2k}2^l)$	Shakhlevich et al. (1999)
1	1	∞		$C_{\max}$	Binary NP-hard	Shakhlevich and Sotskov (1994)
2	1	m		$C_{\max}$	$O((r^2 \log r)^{lm+1})$	Shakhlevich et al. (1999)
1	1	∞		$C_{\max}$	Binary NP-hard	Shakhlevich and Sotskov (1994)
2	∞	2	*pmtn*	$C_{\max}$	$O(r^3 + n_O)$	Shakhlevich et al. (1999)
k	1	m	*pmtn*	$C_{\max}$	$O(rT^{k+l-1})$	Shakhlevich et al. (1999)
2	1	3	*pmtn*	$C_{\max}$	Binary NP-hard	Shakhlevich et al. (1999)
1	∞	∞	*pmtn*	$C_{\max}$	$O(nm(\min\{nm, m^2\}+ +m\log n) + r)$	Shakhlevich et al. (1999)

which may be different for different jobs. The sign ∞ in the second and third column of Table 3 specifies that the corresponding parameter can be arbitrarily.

In Shakhlevich et al. (2000), a review of known results for mixed shop problems is given when the numbers $n_J \leq n$ and $n_O \leq n$ of the jobs of the set J are not fixed. It is necessary to note that, if the problem is NP-hard for the makespan criterion, it remains NP-hard for all other regular criteria traditionally considered in scheduling theory.

11 Conclusion

The literature in scheduling theory includes a large number of reviews and books devoted to one-stage and multi-stage processing systems. In this review, the border between polynomially solvable and NP-hard scheduling problems with a fixed number of jobs and an unlimited number of machines is given.

In particular, as it follows from Table 1, there is a polynomial algorithm for any regular optimization criterion in the case of two jobs while the problem with three jobs is NP-hard for any criterion traditionally considered in scheduling theory. The construction of a makespan optimal schedule without operation preemptions for a mixed shop problem is possible in polynomial time, if the number of jobs and machines is fixed provided that one of the parameters m and n_J is not larger than two.

From Tables 1, 2 and 3 it follows that the classification of scheduling problems with a fixed number of jobs is completed. Possible directions of further work in this area can be the improvement of the complexity of efficient algorithms for the criteria $C_{\max}$ and $\sum C_i$ and also the investigation of similar scheduling problems for other optimization criteria.

Acknowledgments This work has been supported by INTAS (projects 00-217 and 03-51-5501).

Appendix

Proof of Theorem 2 We reduce problem PARTITION polynomially to the following decision problem. Does there exist a schedule for $J2 \mid n = 3, pmtn \mid C_{\max}$ such that $C_{\max} \leq y$?

Given an instance of problem PARTITION we construct the following instance for $J2 \mid n = 3, pmtn \mid C_{\max}$. Let $H > 3\Delta$ and set $y = 41Hk + 2\Delta k + 3.5\Delta$. The structure of the $J2 \mid n = 3, pmtn \mid C_{\max}$ - instance can be divided into k periods as shown in Figure 8(a–c). In these figures, operations to be processed on machine M_1 (machine M_2) are represented by white (black) circles. The processing times of the operations are shown above the circles.

First, we show that if problem PARTITION has a solution, then a schedule with $C_{\max} \leq y$ exists.

Assume that there is a partition A_1, A_2 of $A = \{1, 2, \ldots, 2k\}$ with

$$\sum_{i \in A_1} e_i = \sum_{i \in A_2} e_i = E = 0.5\Delta.$$

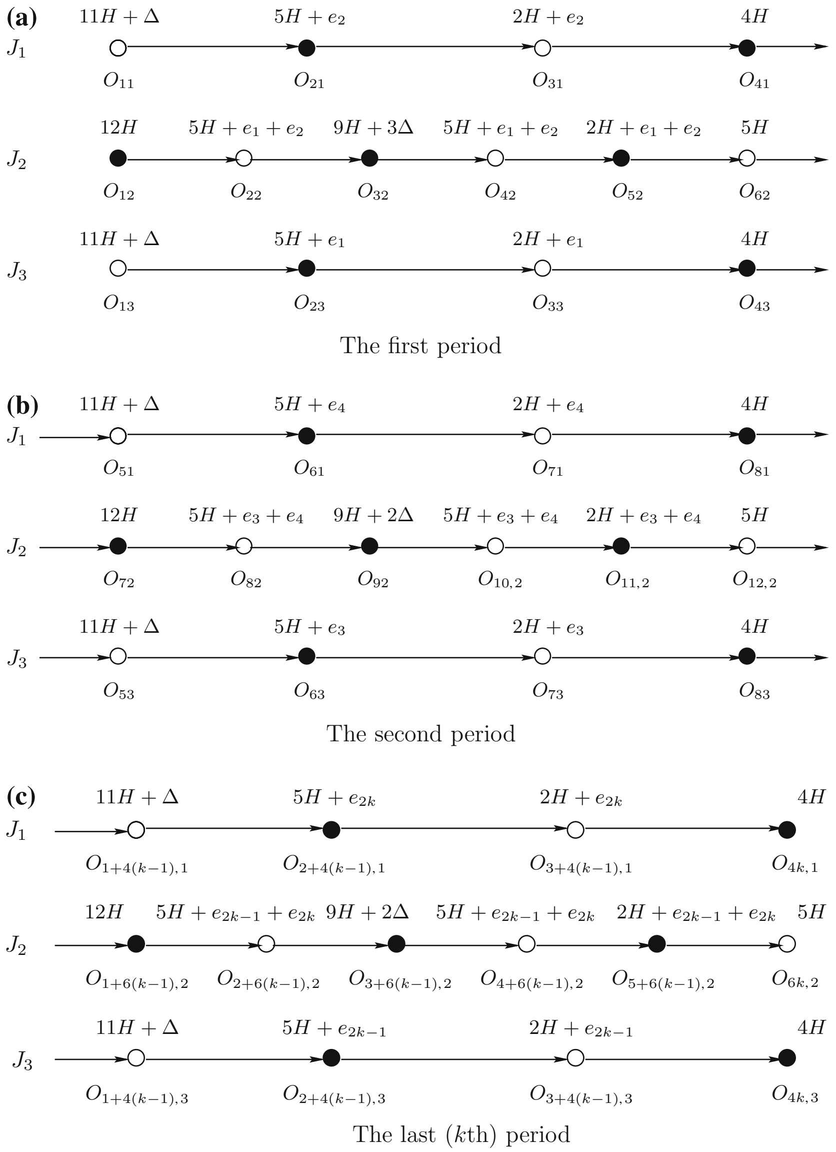

Fig. 8 **a** The first period; **b** the second period **c** the last (kth) period

Then we construct a schedule for the instance of problem $J2 \mid n = 3; pmtn \mid C_{\max}$ which consists of k blocks. The ith block corresponds to the ith period of the instance.

If $e_1 \in A_1$ and $e_2 \in A_2$, the first block is as shown in Fig. 9a. Figure 9b shows the first block for the case $e_1 \in A_2$, $e_2 \in A_1$. In both figures it is possible to switch O_{43} and O_{41} in the last three slices. Furthermore, in each of these figures as well

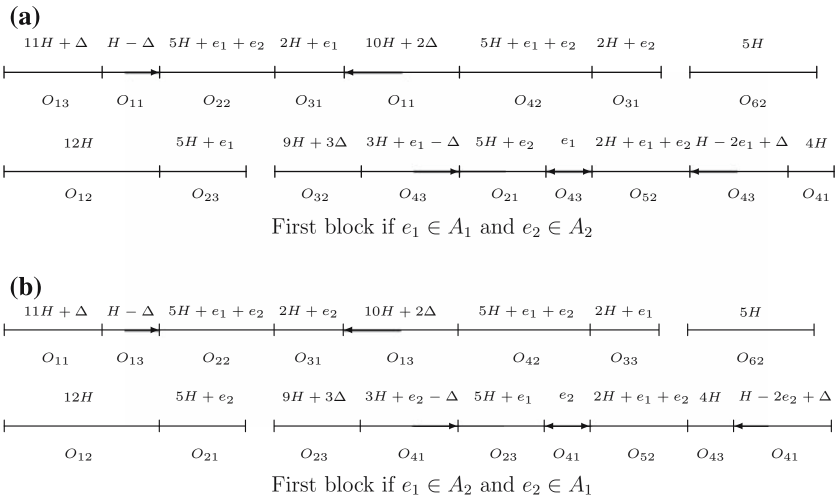
(a)
$11H+\Delta$ $H-\Delta$ $5H+e_1+e_2$ $2H+e_1$ $10H+2\Delta$ $5H+e_1+e_2$ $2H+e_2$ $5H$
O_{13} O_{11} O_{22} O_{31} O_{11} O_{42} O_{31} O_{62}
$12H$ $5H+e_1$ $9H+3\Delta$ $3H+e_1-\Delta$ $5H+e_2$ e_1 $2H+e_1+e_2$ $H-2e_1+\Delta$ $4H$
O_{12} O_{23} O_{32} O_{43} O_{21} O_{43} O_{52} O_{43} O_{41}
First block if $e_1 \in A_1$ and $e_2 \in A_2$
(b)
$11H+\Delta$ $H-\Delta$ $5H+e_1+e_2$ $2H+e_2$ $10H+2\Delta$ $5H+e_1+e_2$ $2H+e_1$ $5H$
O_{11} O_{13} O_{22} O_{31} O_{13} O_{42} O_{33} O_{62}
$12H$ $5H+e_2$ $9H+3\Delta$ $3H+e_2-\Delta$ $5H+e_1$ e_2 $2H+e_1+e_2$ $4H$ $H-2e_2+\Delta$
O_{12} O_{21} O_{23} O_{41} O_{23} O_{41} O_{52} O_{43} O_{41}
First block if $e_1 \in A_2$ and $e_2 \in A_1$

Fig. 9 **a** First block if $e_1 \in A_1$ and $e_2 \in A_2$; **b** first block if $e_1 \in A_2$ and $e_2 \in A_1$

as in the following figures the first line corresponds with M_1 and the second line corresponds with M_2. Preemptions are indicated by arrows.

For the case $e_1 \in A_1$, $e_2 \in A_2$ the extensions of the first block (see Fig. 9a) are shown for the cases $e_3 \in A_1$, $e_4 \in A_2$ and $e_3 \in A_2$, $e_4 \in A_1$ in Fig. 10a, b, respectively. As we can choose either O_{41} or O_{43} to be the last operation in all partial schedules can be combined.

Similarly, we get extensions for the case $e_1 \in A_2$, $e_2 \in A_1$ (see Fig. 9b) if in the extension we interchange corresponding operations of the first and third job. Note, that (as in the first block) in the second block we can choose O_{83} or O_{81} to be scheduled as the last operation. Thus, in each Figs. 9a, b and 10a, b we have two possible schedules.

Similarly, we construct blocks $\nu = 3, 4, \ldots, k$. The resulting schedule has length y as shown in Fig. 11. In this figure we suppose that $A_1 = \{e_1, e_3, \ldots, e_{2k-1}\}$ and $A_2 = \{e_2, e_4, \ldots, e_{2k}\}$.

Next, we show that if a schedule with $C_{\max} \leq y$ exists for the instance, then problem PARTITION has a solution. Due to Lemma 1 we may consider only schedules in which at least one operation finishes at the end of each slice.

Taking into account $\sum_{\mu_{ij}=M_1} p_{ij} = \sum_{\mu_{ij}=M_2} p_{ij} = 41 \cdot H \cdot k + 2\Delta \cdot k + 3 \cdot \Delta$, the total idle time of machine M_1 as well as the total idle time of machine M_2 is not more than 0.5Δ in the desired schedule.

Through an extensive case analysis it can be shown that in order to satisfy the bound 0.5Δ for the idle time, we must schedule the operations from the first period in one of the four ways shown in Fig. 9 (remember that O_{43} can be scheduled as last operation instead of O_{41}), the operations from the second period in one of the four ways shown in Fig. 10, and so on. Thus, for the processing lengths on both machines in each period we have two possibilities. These possibilities are indicated in Fig. 12.

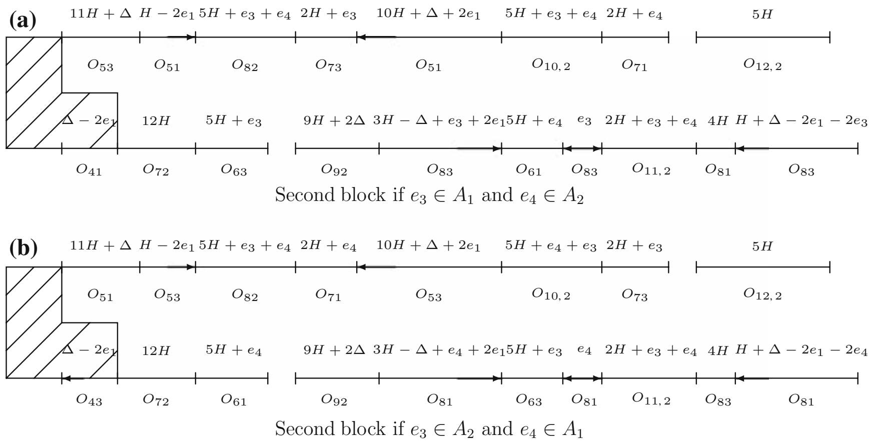

Fig. 10 **a** Second block if $e_3 \in A_1$ and $e_4 \in A_2$; **b** Second block if $e_3 \in A_2$ and $e_4 \in A_1$

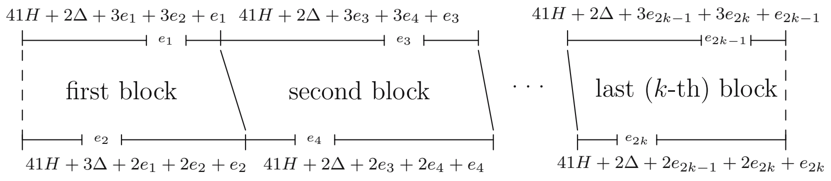
41H + 2Δ + 3e1 + 3e2 + e1
41H + 2Δ + 3e3 + 3e4 + e3
41H + 2Δ + 3e2k−1 + 3e2k + e2k−1
e1
e3
e2k−1
first block
second block
· · ·
last (k-th) block
e2
e4
e2k
41H + 3Δ + 2e1 + 2e2 + e2
41H + 2Δ + 2e3 + 2e4 + e4
41H + 2Δ + 2e2k−1 + 2e2k + e2k

Fig. 11 Schedule for the instance

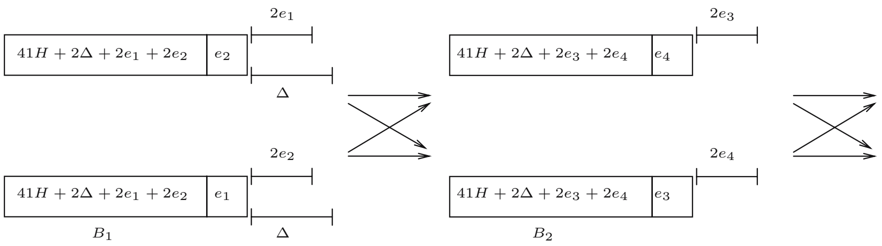

Fig. 12 Lower bounds for the completion times on the machines

Let $B_i = 41H + 2\Delta + 2e_{2i} + 2e_{2i-1}$ for $i = 1, 2, \ldots, k$. It follows from Fig.12 that for some set $A_1 \subset \{1, 2, \ldots, 2k\}$ such that A_1 includes exactly one element from each pair $(2i-1, 2i)$ $(i = 1, 2, \ldots, k)$

- $\sum_{i=1}^{k} B_i + \sum_{i \in A_1} e_i + \Delta$ is a lower bound for the completion time on machine M_2,
- $\sum_{i=1}^{k} B_i + \sum_{i \in A_1} e_i + 2(\Delta - \sum_{i \in A_1} e_i)$ is a lower bound for the completion time on
  machine M_1.

Thus, we have $\sum_{i=1}^{k} B_i + \sum_{i \in A_1} e_i + \Delta \le y = 41Hk + 2\Delta k + 3.5\Delta = \sum_{i=1}^{k} B_i + 1.5\Delta$ which implies $\sum_{i \in A_1} e_i \le 0.5\Delta$ and $\sum_{i=1}^{k} B_i + \sum_{i \in A_1} e_i + 2(\Delta -$

$\sum_{i\in A_1} e_i) \leq y$ which implies $\sum_{i\in A_1} e_i \geq 0.5\Delta$. Therefore, $\sum_{i\in A_1} e_i = 0.5\Delta$ and we conclude that A_1 solves the problem PARTITION.

It is easy to see that the above reduction of problem PARTITION to problem $J2 \mid n = 3, pmtn \mid C_{\max}$ is a polynomial one, and thus Theorem 2 is proven. $\square$

References

Akers SB (1956) A graphical approach to production scheduling problems. Oper Res 4:244–245

Akers SB, Friedman J (1956) A non-numerical approach to production scheduling problems. Oper Res 3:429–442

Aho AV, Hopcroft JE, Ullman JD (1974) The design and analysis of computer algorithms. Addison-Wesley, Reading

Brucker P (1988) An efficient algorithm for the job-shop problem with two jobs. Computing, 40:353–359

Brucker P (1994) A polynomial algorithm for the two machine job-shop scheduling problem with a fixed number of jobs. Oper Res Spekt 16:5–7

Brucker P, Jurisch B, Meyer W (1989) Geometric methods for solving the job-shop scheduling problem. Preprint Heft 127, Fachbereich Mathematik/Informatik, Universität Osnabrück

Brucker P, Kravchenko SA, Sotskov YuN (1994) On the complexity of two machine job-shop scheduling with regular objective functions. Oper Res Spect 16:5–7

Brucker P, Kravchenko SA, Sotskov YuN (1999) Preemptive job-shop scheduling problems with a fixed number of jobs. Math Methods Oper Res 49:41–76

Brucker P, Schlie R (1990) Job-shop scheduling with multipurpose machines. Computing, 45:369–375

Gonzalez T, Sahni A (1976) Open shop scheduling to minimize finish time. J Associat Comput Mach 23:665–679

Hardgrave WH, Nemhauser GL (1963) A geometric model and graphical algorithm for a sequencing problem. Oper Res 11:889–900

Johnson SM (1954) Optimal two- and three-stage production schedules with setup times included. Naval Res Logist Quart 1:61–68

Kravchenko SA, Sotskov YuN (1996) Optimal makespan schedule for three jobs on two machines. Math Methods Oper Res 43:233–238

Masuda T, Ishii H, Nishida T (1985) The mixed shop scheduling problem. Discr Appl Math 11:175–186

Servach VV (1983) On the Akers–Friedman problem. Upravljajemye Syst Novosibirsk 23:70–81 (in Russian)

Shakhlevich NV, Strusevich VA (1990) Scheduling two jobs in a multi-machine open shop to minimize an arbitrary regular penalty function. Report 9125/A, Erasmus University Rotterdam, The Netherlands

Shakhlevich NV, Sotskov YuN (1994) Scheduling two jobs with fixed and nonfixed routines. Computing, 52:17–30

Shakhlevich NV, Sotskov YuN, Werner F (1999) Shop-scheduling problems with fixed and nonfixed machine orders of jobs. Ann Oper Res 92:281–304

Shakhlevich NV, Sotskov YuN, Werner F (2000) Complexity of mixed shop scheduling problems: A survey, Euro J Oper Res 120:343–351

Sotskov YuN (1985) Optimal scheduling of two jobs with a regular criterion. Institute of Engineering Cybernetics of the Academy of Sciences of BSSR, Minsk, pp. 86–95 (in Russian)

Sotskov YuN (1989) Complexity of scheduling with fixed number of jobs. Dokl Akad Nauk BSSR. 33:488–491 (in Russian)

Sotskov YuN (1990) Complexity of optimal scheduling of three jobs, Kibernetika, N 5:74–78 (in Russian)

Sotskov YuN (1991) The complexity of shop-scheduling problems with two or three jobs. Euro J Oper Res 53:326–336

Sotskov YuN, Shakhlevich NV (1990) NP-hardness of optimal scheduling three jobs. Vestsi Akad Navuk BSSR Ser Fiz-Mat Navuk 4:96–101 (in Russian)

Sotskov YuN, Shakhlevich NV (1995) NP-hardness of shop-scheduling problems with three jobs. Discr Appl Mathe 59:237–266

Strusevich VA (1986) On the possibility of constructing optimal makespan schedules for multistage systems with nonfixed routes. Vestsi Akad Navuk BSSR Ser Fiz-Mat Navuk 6:43–48 (in Russian)

Szwarc W (1960) Solution of the Akers-Friedman scheduling problem. Oper Res 8:782–788