ISSN 0081-5438, Proceedings of the Steklov Institute of Mathematics, 2015, Vol. 290, pp. 166–177.

On the Complexity of Constructing Multiprocessor Little-Preemptive Schedules

E. V. Shchepin[a]

Received March 15, 2015

Abstract—We present a full and correct proof of the fact that the problem of constructing an optimal schedule for the OPEN SHOP problem with at most $m-3$ preemptions for an m-processor system is NP-hard. We also show that the proof of this result given by E. Shchepin and N. Vakhania in Ann. Oper. Res. **159**, 183–213 (2008) is incorrect.

DOI: 10.1134/S0081543815060152

1. INTRODUCTION

The OPEN SHOP scheduling problem consists in the following: There are m machines (or processors) $M_1,\dots,M_m$ and n jobs $J^1,\dots,J^n$ (or tasks). Each job is split into parts (operations), $J^k=\{J^k_1,\dots,J^k_m\}$, where the operation J^k_i belongs to the kth job and should be processed by the ith machine. For every operation, its processing time[1] on the machine to which it is assigned is known. It is required to compile a minimum-length schedule of jobs for machines; i.e., for each machine and every moment of time, it is required to point out a job that a machine should process so that all the jobs are processed in the minimum possible time. In the OPEN SHOP problem, there are only two constraints on scheduling: at every moment of time, each job can be processed by at most one machine and each machine can process at most one job. The processing order of operations of a job can be arbitrary.

A *preemption* in a schedule is a situation when the already started processing of an operation on some machine is interrupted, although the operation is not completed, and the machine is switched to another job (or is idle for some time) and returns to the processing of this operation after some time. As shown in [1], the non-preemptive version of the OPEN SHOP problem is NP-hard. On the other hand, in [2], a polynomial algorithm for the OPEN SHOP problem is presented that generates an optimal schedule with at most $4m^2-6m+3$ preemptions.

The main result of the present paper consists in proving the fact that the version of the OPEN SHOP problem with at most $m-3$ preemptions remains NP-hard. This result was announced in [3]; however, the proof proposed there, as pointed out in [4] and shown in the present paper, contains a significant gap.

A job consisting of one nonfictitious operation will be called *simple*, and a job that has more than one nonfictitious operation will be called *composite*. If all jobs are simple, then an optimal schedule of the open shop can be constructed in linear time. However, as shown in the present study, the appearance of at least one composite job makes the problem NP-hard.

It remains an open question whether one can construct an optimal schedule in polynomial time for the OPEN SHOP problem with at most $m-2$ preemptions. A positive result in this direction is contained in [5], where a polynomial-time (and even linear-time) algorithm is constructed that generates an optimal schedule with at most $m-2$ preemptions for the OPEN SHOP problem in the

[a] Steklov Mathematical Institute of Russian Academy of Sciences, ul. Gubkina 8, Moscow, 119991 Russia.
E-mail address: scepin@mi.ras.ru

[1] Operations may be *fictitious*, i.e., they may require no time for processing.

case of so-called *acyclic* distributions of jobs on machines; this class of distributions includes those with a single composite job as a particular case.

By a *scheme* of a distribution of jobs on machines we will mean a simplicial complex whose vertices are in one-to-one correspondence with the machines and whose maximal simplices (i.e., those that are not proper subsets of other simplices) are in one-to-one correspondence with the jobs; moreover, for every job, the vertex set of the simplex associated with this job must correspond to the set of machines to which nonfictitious operations of this job are assigned. A distribution of jobs on machines is said to be *acyclic* if its scheme has no one-dimensional cycles that are not homologous to zero. Acyclic distributions were introduced in [6]; however, the definition given there operates not with the entire scheme but only with its one-dimensional skeleton, which was called there a (full) preemption graph.

It follows from what is proved in the present study that the problem of constructing an optimal schedule with at most $m-3$ preemptions for the OPEN SHOP problem with acyclic distribution of jobs is NP-hard.

In scheduling theory, the importance of the OPEN SHOP problem with acyclic distribution is associated with the fact that one can reduce to it the general problem of compiling an optimal schedule of independent jobs with preemptions for unrelated processors (see [7]). The difference between these two problems is that in the latter the jobs can be arbitrarily divided into parts and distributed among processors. It turns out that the division minimizing the load (the total processing time of all jobs assigned to a given machine) of the most loaded machine can be made (in polynomial time) acyclic (see [8]). Therefore, the construction of an optimal schedule for a system of unrelated processors reduces to the OPEN SHOP problem with acyclically distributed jobs.

Notice also that the fact that $m-2$ is the NP-critical number of preemptions for the OPEN SHOP problem with acyclic distribution of jobs plays an essential role in the result of [3] stating that the problem of constructing an optimal schedule becomes NP-hard if the number of preemptions is not greater than $2m-3$ for a *system without slow processors* (which is a particular case of a system of unrelated processors). The present study fills a gap in the proof of the indicated fact. Thus, the result from [3] concerning systems without slow processors can be considered completely proved only after the publication of the present paper.

2. PARTITION AND OPEN SHOP

In this section, we repeat the arguments of [3] underlying the proof of Theorem 13 in [3], which states that the problem of constructing an optimal schedule with at most $m-3$ preemptions for the m-machine OPEN SHOP problem with acyclic distribution is NP-hard.

Let $C=\{c_1,\dots,c_n\}$, where $c_i\in\mathbb{N}$, $i=1,\dots,n$, and $S/2$, where $S=\sum_{i=1}^n c_i>2$ is an even number, form an input of the PARTITION problem.[2] Recall that the problem consists in determining whether there exists a subset of C with the sum of elements equal to $S/2$.

Given an input of the PARTITION problem with a set C and a number $m\ge 3$, we define a shop $O(C,m)$. Namely, $O(C,m)$ includes $1+n(m-2)+2m-2=(n+2)m-2n-1$ jobs and m machines $\{M_i\}_{i=1}^m$. The set of jobs is split into the following three categories:

1. There exists exactly one composite job I, which we call the *common* job. It includes nonfictitious operations on all machines. The processing time of the part of I assigned to the ith machine M_i is 1 for all $i=1,\dots,m$. Hence, the total processing time of the job I is m.

2. The jobs of the second type are called *partition jobs*. All these jobs are simple. Partition jobs are assigned to all machines except for the first, M_1, and the last, M_m (which are called *extremal*

[2] In this section, n denotes exclusively the number of elements of the set C in the PARTITION problem; in the rest of the paper, n stands for the number of jobs.

machines, while the remaining machines are said to be *intermediate*). There are $(m-2)n$ partition jobs in total; the jth, $j \leq n$, partition job assigned to the machine M_i, $1 < i < m$, is denoted by $P_{i,j}$. The processing time of the job $P_{i,j}$ is c_j/S^{2i}. Notice that the total processing time of all partition jobs on M_i is $1/S^{2i-1} < 1$.

3. The jobs of the third type are called *fixers* and are denoted by

$$F_1, F_2^+, F_2^-, F_3^+, F_3^-, \dots, F_{m-1}^+, F_{m-1}^-, F_m.$$

All these jobs are simple. As the notation suggests, there is only one fixer on each of the extremal machines M_1 and M_m, and there are two fixers on each intermediate machine. The processing time of the fixers F_1 and F_m is $m-1$. The processing time of the first fixer F_i^+ of M_i, $1 < i < m$, is $i - 1 - 1/(2S^{2i-1})$, while the processing time of the second fixer F_i^- is $m - i - 1/(2S^{2i-1})$.

A machine is said to be *idle* at some moment (or on some time interval) if no job is assigned to the machine at this moment (on this interval, respectively). We will say that a schedule σ is *tight* if all machines are engaged from the beginning to the end (i.e., to the completion time of the last job). A tight schedule without preemptions is uniquely defined by the processing order of jobs on each machine.

The following statement is obvious.

Lemma 1. *Any tight schedule for the shop $O(C,m)$ is optimal and has makespan m.*

It is useful to notice that if σ is an optimal schedule for the shop $O(C,m)$, then the common job I must be being processed at every moment up to the end of σ (because the processing time of I is equal to the makespan of the schedule σ).

In [3], a proof of the following statement is presented.

Theorem 1. *The problem of constructing an optimal schedule with at most $m-3$ preemptions for the shop $O(C,m)$ is* NP*-hard.*

The proof of this theorem is based on reducing the NP-complete PARTITION problem (with input C) to $O(C,m)$. Namely, given a solution $\sum_{i \leq k} c_i = S/2$ to the PARTITION problem, we can for any m construct a tight schedule of makespan m without preemptions for $O(C,m)$ in linear time (see Lemma 1). We construct this schedule as follows. On M_1, we first process the fixer F_1 and then the common job I. On the last machine, we first process I and then F_m. On the other machines M_i, we first process the fixer F_i^+, then all $P_{i,j}$, $j \leq k$ (in an arbitrary order), and then I, which is followed by the remaining partition jobs $P_{i,j}$, $j > k$, and the second fixer F_i^- is processed last.

To prove the reduction in the opposite direction, we introduce the following definition. A schedule σ is said to be a *partitioning* schedule if there exists an intermediate machine M_i in σ and a moment of time t such that the processing of any partition job on M_i is either completed by the time t or starts not earlier than t and the total processing time of the completed partition jobs on M_i at time t is equal to the total processing time of the remaining partition jobs. The moment t is called a *partitioning* time in the schedule σ.

Lemma 2. *Given any feasible schedule σ for $O(C,m)$ of makespan m with at most $m-3$ preemptions, one can construct another schedule σ' of makespan m with at most $m-3$ preemptions in linear time such that σ' is a partitioning schedule.*

Suppose that Lemma 2 is valid. Let σ be a feasible schedule for $O(C,m)$ of makespan m with at most $m-3$ preemptions. According to Lemma 2, in linear time one can construct a schedule σ' that contains an intermediate machine with a partitioning time. Such a switching point can be found in linear time by verifying all switching points of the schedule on intermediate machines. It is clear that any partitioning time gives a solution to the PARTITION problem. This proves Theorem 1.

3. COUNTEREXAMPLE

So, we can see that everything hinges on Lemma 2, whose original proof occupied more than ten pages and, in the referees' opinion, was too lengthy. Therefore, it was significantly reduced in [3]. In this section, we show that the reduced proof is invalid, while the rest of the paper reproduces a somewhat improved original proof of Lemma 2.

The simplified proof ("in the opposite direction") given in [3, p. 208] makes nothing of the fact that the sizes of the partition jobs of the shop $O(C,m)$ on different machines are different. Therefore, this argument is disproved below by an example in which the lengths of all partition jobs on different machines are identical.

Consider a shop $O'(C,m)$ that has the same three types of jobs as $O(C,m)$. Just as in $O(C,m)$, the *common* job I is evenly distributed among the machines, and the total processing time of the job I is m. The partition jobs are again assigned to all *intermediate* machines; however, in contrast to $O(C,m)$, the processing time of the jth partition job, $j \le n$, assigned to the machine M_i, $1 < i < m$, is c_j/S irrespective of the machine. Thus, the total processing time of all partition jobs on the same machine is equal to one. The lengths of the fixers on the intermediate machines are defined in such a way that the processing time of the first fixer F_i^+ of M_i, $1 < i < m$, is $i - 1 - 1/2$, while the processing time of the second fixer F_i^- is $m - i - 1/2$.

Let us show that for the shop $O'(C,m)$ there exists an optimal schedule with two preemptions such that on every machine all partition jobs are processed successively one after another in the same order. Such a schedule is not a partitioning one and gives nothing for the solution of the PARTITION problem. The schedule σ' has the following structure:

(1) on the first machine, the common job is processed in the interval $[0,1)$, and then the fixer;

(2) on the ith intermediate machine, $i = 2,\dots,m-1$, first a fixer is processed in the interval $[0, i-3/2)$, then all partition jobs are processed successively in the interval $[i-3/2, i-1/2)$, and then the common job is processed in the interval $[i-1/2, i+1/2)$. The second fixer is processed last;

(3) on the last machine, the fixer is processed in the interval $[0,1)$. In the interval $[1,3/2)$, the common job is processed. In the interval $[3/2, m-1/2)$, the fixer is again processed. Finally, in the interval $[m-1/2, m)$, the remaining part of the common job is processed.

The schedule σ' is tight and has only two preemptions, both on the last machine.

4. PROOFS

A *schedule* for an *m-processor* system $\mathcal{M} = \{M_1,\dots,M_m\}$ with n *jobs* (*tasks*) $\mathcal{J} = \{J^1,\dots,J^n\}$ shows which job should be made by a given processor (*machine*) at every time point. Formally, a schedule σ can be considered as a subset in $\mathcal{J} \times \mathcal{M} \times \mathbb{R}^+$, where $\mathbb{R}^+$ is the set of nonnegative real numbers (time). The condition $(J,M,t) \in \sigma$ means that the job J is processed by the machine M at time t according to σ. The least T for which $\sigma \subset \mathcal{J} \times \mathcal{M} \times [0,T]$ is called the *makespan* of the schedule and is denoted by $\|\sigma\|$.

For a pair $(J,M) \in \mathcal{J} \times \mathcal{M}$, the set $\sigma(J,M) = \{t \in \mathbb{R}^+ \mid (J,M,t) \in \sigma\}$ may consist of a single left-closed (and right-open) interval or may be a left-closed multi-interval, i.e., a union of two or more disjoint left-closed intervals. The total length of these intervals from $\sigma(J,M)$ is denoted by $|\sigma(J,M)|$ and called the *processing time* of the job J on the machine M according to the schedule σ.

A *feasible schedule* σ must satisfy the following conditions:

(1) for each $M \in \mathcal{M}$, the set of jobs $\sigma(M,t) = \{J \in \mathcal{J} \mid (J,M,t) \in \sigma\}$ contains at most one element;

(2) for each $J \in \mathcal{J}$, the set of machines $\sigma(J,t) = \{M \in \mathcal{M} \mid (J,M,t) \in \sigma\}$ contains at most one element;

(3) for each $J \in \mathcal{J}$, the equality $\sum_{M\in\mathcal{M}}(|\sigma(J,M)|/M(J)) = 1$ holds, where $M(J)$ denotes the processing time of the job J by the processor M.

A feasible schedule with minimum makespan is said to be *optimal*.

Switchings and components. The maximum (connected) time interval during which a machine processes one job is called a *component* of a schedule. A schedule can be defined by specifying all its components. Formally, a component of a schedule σ is a triple $(J, M, [p, q))$, where J is a job, M is a machine, and $[p, q)$ is a time interval that is a connected component of $\sigma(J, M)$. We will call the component $(J, M, [p, q))$ a *J-component of the schedule σ on the machine M* or a *(J, M)-component of the schedule σ*.

The boundary points of the set $\sigma(J, M)$ are called *switching points* of the job J on M or *J-switching points on M*. At such a time point, M interrupts the processing of the job J or restarts it.

Preemptions. A job J is said to be *split* on a machine M if $\sigma(J, M)$ is a multi-interval, i.e., if it consists of two or more components. The *number of preemptions* $\mathrm{pr}(\sigma(J, M))$ *of a job J on a machine M in a schedule* σ is defined as the number of components in $\sigma(J, M)$ reduced by one. The number $\mathrm{pr}^\sigma(M) = \sum_{J\in\mathcal{J}} \mathrm{pr}(\sigma(J, M))$ is called the *number of preemptions of the schedule σ on the machine M*; the sum $\mathrm{pr}(\sigma) = \sum_{M\in\mathcal{M}} \mathrm{pr}^\sigma(M)$ is the total *number of preemptions of the schedule σ*.

Next, notice that the number of switching points is closely related to the number of preemptions. The following lemma presents an exact formula; its proof is left to the reader.

Lemma 3. *The number of preemptions of a tight schedule on a machine M is $s - n - 1$, where s is the number of all switching points on M and n is the number of jobs assigned to M.*

Schedule editing. There are three schedule editing operations: *cutting*, *inserting*, and *moving*. Let p and q, $p < q$, be time points.

Cutting. We say that a schedule σ' is obtained from a schedule σ by the (p, q)-*cutting* on a machine M_i if the following conditions hold:

(1) $\sigma'(J, M) = \sigma(J, M)$ for all $M \neq M_i$ and all J;

(2) $\sigma'(M_i, t) = \sigma(M_i, t)$ for $t < p$;

(3) $\sigma'(M_i, t) = \sigma(M_i, t + q - p)$ for $t \geq p$.

Note that if σ is tight, then the schedule σ' obtained from σ by the (p, q)-cutting on M_i is also tight and the completion time of the last job on M_i in σ' is less by $q - p$ than that in σ. Next, if p and q are switching points in σ, then the number of preemptions in σ' is not greater than that in σ. Moreover, this number may even be less by one if M_i processes the same job immediately before the time p and immediately after the time q in σ.

Insertion. We say that a schedule σ' is obtained from a schedule σ by the (p, q)-*insertion* of a job J on a machine M_i if the following conditions hold:

(1) $\sigma'(\widetilde{J}, M) = \sigma(\widetilde{J}, M)$ for all $M \neq M_i$ and all $\widetilde{J}$;

(2) $\sigma'(M_i, t) = \sigma(M_i, t)$ for $t \leq p$;

(3) $\sigma'(M_i, t) = J$ for $t \in [p, q)$;

(4) $\sigma'(M_i, t) = \sigma(M_i, t - (q - p))$ for $t \geq q$.

The tightness of σ' follows from the tightness of σ, and the completion time of the last job on M_i in σ' is greater by $q - p$ than that in σ. Next, if p is a switching point in σ on M_i, then the insertion cannot create preemptions of any job different from the inserted job J; the insertion generates a new preemption of the job J if J has already been assigned to M_i in σ and p is not a boundary point of $\sigma(J, M_i)$.

Moving. We use the notation (p, i, q) for a component of the schedule that assigns a job J to the machine M_i on the time interval $[p, q)$. A *moving*, denoted by $(p, q) \to (p', q')$, removes a whole J-component (p, i, q) of a schedule σ from the interval $[p, q)$ and inserts it into the interval $[p', q')$ (of the same length) on the same machine M_i. Thus, the moving is performed in two steps: the (p, q)-cutting followed by the (p', q')-insertion.

The moving does not affect the tightness of a schedule and the completion time $|\sigma|_{M_i}$ of the last job on the machine M_i if $q' \le |\sigma|_{M_i}$. Notice that if $p > p'$ and p' is a switching point in σ, then the (p', q')-insertion does not create any new preemption of jobs processed on M_i. Hence, any moving $(p, q) \to (p', q')$ of a J-component does not increase the number of preemptions. Moreover, if M_i processes the same job immediately before the time p and after the time q according to σ, then the number of preemptions of the resulting schedule σ' is less by one than that of σ. If p is not a switching point, then a moving of a J-component can increase the number of preemptions at most by 1.

If M_i is an extremal machine, then a moving $(p, q) \to (p', q')$ of its I-component does not increase the number of preemptions for $0 < p < q < |\sigma|_{M_i}$. Indeed, since only two jobs (the common job and the fixer) are processed on M_i, the cutting of the I-component reduces the number of preemptions of the fixer by one.

The following lemma summarizes the above remarks.

Lemma 4. *Suppose a schedule σ' is obtained from a schedule σ by a moving $(p, q) \to (p', q')$ of its J-component on a machine M_i and $q' \le |\sigma|_{M_i}$. Then*

(1) *σ' is tight if σ is tight*;

(2) $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma) + 1$;

(3) $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$ *if $p' < p$ and p' is a switching point of σ on M_i*;

(4) $\mathrm{pr}(\sigma') < \mathrm{pr}(\sigma)$ *and* $\mathrm{pr}(\sigma'(J, M_i)) < \mathrm{pr}(\sigma(J, M_i))$ *if $p' < p$ and p' is a switching point for J on M_i according to σ*;

(5) $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$ *if $i = 1$ or $i = m$.*

Schedule preprocessing. Now we introduce procedures based on cutting, inserting, and moving that allow us to simplify the structure of a schedule while not increasing its makespan. A sequence of I-components $(p_1, i_1, q_1), (p_2, i_2, q_2), \dots, (p_k, i_k, q_k)$ will be called an *I-loop* if $q_j = p_{j+1}$ for all $j < k$ and $i_1 = i_k$; we will say that this I-loop *passes through the machines* $M_{i_2}, M_{i_3}, \dots, M_{i_{k-1}}$. An I-component (p, i, q) will be called *unit* if $q - p = 1$, and an I-loop will be called *integral* if all its components except for the first and last ones are unit and every intermediate machine passed by this loop has at least one preemption.

We say that an I-component (p, i, q) *splits* a job J if both p and q are switching points for J, i.e., if M_i processes J immediately before the time p and immediately after the time q.

Lemma 5. *If an I-component splits another job, then any moving of this I-component does not increase the total number of preemptions.*

Proof. The cutting of an I-component in this case reduces the number of preemptions, and the insertion can again increase their number, but at most by 1. Hence, the total number of preemptions does not increase. □

Lemma 6. *For any tight schedule σ, one can construct in linear time another tight schedule σ' such that $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$ and any machine M_i with preemptions but without I-preemptions according to σ has at most one preemption in σ' (and this preemption is a splitting generated by the single I-component).*

Proof. Let $(p, i, p+1)$ be the only I-component on M_i, and let $\mathrm{pr}^\sigma(M_i) > 0$. A tight schedule σ' with at most one preemption on M_i can be obtained as follows. First, we construct a tight schedule on M_i for all jobs assigned, according to σ, to M_i except for I. This schedule has no preemptions. Then we perform the $(p, p+1)$-insertion of the job I on M_i. The schedule obtained satisfies our requirements on M_i. We perform an analogous procedure on all other machines without I-preemptions and obtain a schedule σ' such that $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$. □

Lemma 7. *For any tight schedule σ, one can construct in linear time another tight schedule σ' without integral I-loops such that $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$.*

Proof. By Lemma 6, we can assume without loss of generality that all machines with unit I-components in the schedule σ have at most one preemption. If σ contains an integral loop $(p, i, q), (q, i_1, q+1), (q+1, i_2, q+2), \dots, (q+k-1, i_k, q+k), (q+k, i, r)$, then each of the machines $M_{i_1}, \dots, M_{i_k}$ has exactly one preemption generated by the corresponding I-component. Then, according to Lemma 5, the moving of these components does not increase the number of preemptions. Hence, a schedule σ' obtained from σ by the following sequence of movings will have a smaller number of preemptions than σ: on M_i the moving $(q+k, r) \to (q, r-k)$ is performed, on M_{i_1} the moving $(q, q+1) \to (r-k, r-k+1)$ is performed, on M_{i_2} the moving $(q+1, q+2) \to (r-k+1, r-k+2)$ is performed, and so on; the last moving $(q+k-1, q+k) \to (r-1, r)$ is performed on M_{i_k}. The first of the movings described reduces the number of preemptions on M_i, and all the subsequent movings do not increase the number of preemptions. Now, if σ' still contains an integral loop, we again apply the same procedure. The process cannot recur more than $\mathrm{pr}(\sigma)$ times. □

We will say that an intermediate machine M_i in a schedule σ is *incorrect* if σ has no preemptions on M_i and the common job I is assigned to M_i first or last.

Lemma 8. *For any tight schedule σ of the shop $O(C, m)$, one can construct in linear time another tight schedule σ' such that $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$ and σ' has no incorrect machines.*

Proof. Note that σ cannot have more than two incorrect machines to which I is assigned first or last. Let M_i be an incorrect machine on which I is processed first (the second case is similar). The schedule σ' to be constructed on the basis of σ coincides with σ on all machines except M_1 and M_i. We define σ' on M_1 as follows: $\sigma'(I, M_1) = [0, 1)$ and $\sigma'(F_1, M_1) = [1, m)$.

Now, let $(p_1, 1, q_1), (p_2, 1, q_2), \dots, (p_k, 1, q_k)$ be an increasing sequence of all I-components on the machine M_1 in σ. Since $p_1 > 0$, the number of preemptions of the schedule σ on M_1 is equal to $2k-2$ if $q_k = m$ and is equal to $2k-1$ otherwise. The schedule σ' on M_i is defined as follows. First, we perform the $(0, 1)$-cutting on M_i, after which there remain no preemptions on M_i. Second, we perform step by step the (p_i, q_i)-insertions of the job I on M_i. All these insertions except the last one increase the number of preemptions at most by 2. The last insertion increases the number of preemptions by 1 if $q_k = m$. Hence, the number of preemptions of the schedule σ' on M_i does not exceed the number of preemptions of the schedule σ on M_1, and $\mathrm{pr}^\sigma(M_i) = \mathrm{pr}^{\sigma'}(M_1) = 0$. Thus, $\mathrm{pr}(\sigma') \le \mathrm{pr}(\sigma)$. □

Proper times. Starting from this moment, we will deal only with a tight nonpartitioning schedule σ for $O(C, m)$ without integral loops and incorrect machines. By Lemmas 7 and 8, it suffices to prove Lemma 2 for such σ.

For a real x, denote by $\langle x \rangle$ its *fractionality*, i.e., the distance from x to the nearest integer.

A noninteger moment of time t is called a *proper time* for an intermediate machine M_i according to a schedule σ if it is a sum of an integer and the processing times of a proper subset (i.e., a nonempty subset smaller than the entire set) of jobs assigned in σ to M_i and different from I. For the machines M_1 and M_m, we define a proper time as an arbitrary integer. The following inequalities are valid for intermediate machines M_i.

Lemma 9. *The fractionality of a proper time t of an intermediate machine M_i of a tight schedule for $O(c, m)$ satisfies the inequalities*

$$\frac{1}{S^{2i}} \le \langle t \rangle \le \frac{1}{S^{2i-1}}.$$

Proof. Any proper time t of M_i can be represented as $f + p$, where f is either zero or the sum of processing times of one or two fixers and p is the sum of processing times of some set of partition jobs. (Notice that, by definition, $f + p > 0$ is not integer, because this sum cannot include all fixers and partition jobs.) Since at least one of the jobs is included in $f + p$, the definition of processing times of the partition jobs and fixers immediately implies the inequality $\langle f + p \rangle \ge S^{-2i}$. To verify the validity of the second inequality, notice that $0 \le p \le S^{1-2i}$ and that the list of possible values of f is 0, $i - 1 - S^{1-2i}/2$, $m - i - S^{1-2i}/2$, and $m - 1 - S^{1-2i}$. A simple check shows that in any case the inequality $\langle f + p \rangle \le S^{1-2i}$ is valid. □

We will say that a moment of time is *strongly improper* for a machine M_i (according to σ) if it is proper for some other machine.

Lemma 10. *In a tight schedule of the shop $O(c, m)$, a strongly improper time for any intermediate machine is not proper for it.*

Proof. By Lemma 9, the fractionality of a proper time of an intermediate machine M_i belongs to the interval $[S^{-2i}, S^{1-2i}]$. If t is a proper time for an intermediate machine M_j, $j \ne i$, then $\langle t \rangle$ belongs to the interval $[S^{-2j}, S^{1-2j}]$. However, the intersection of these intervals is empty. If M_j is an extremal machine, then all its proper times are integer. □

Lemma 11. *If t_1 and t_2 are strongly improper times of an intermediate machine M_i (with respect to a nonpartitioning tight schedule of the shop $O(c, m)$), then their difference is not a proper time for this machine.*

Proof. Suppose that t_1 and t_2 are proper for different machines M_j and M_k, respectively. If their difference t_3 were a proper time for M_i, then, for $l = \max\{\ell \in \{i, j, k\} \mid \ell < m\}$, two numbers among $S^{2l-2}t_i$, $i = 1, 2, 3$, would be integer, while the fractionality of the third would belong to the interval $[S^{-2}, S^{-1}]$ by Lemma 9. This is impossible for $S^{2l-2}t_3 = S^{2l-2}t_2 - S^{2l-2}t_1$ and, hence, for $t_3 = t_2 - t_1$ as well.

Now, assume that t_1 and t_2 are proper for the same machine M_j. It is clear that $j \ne m$. If $i < j < m$, then the fractionality of their difference is not greater than $2S^{1-2j}$ by Lemma 9, whereas the fractionality of any proper time of M_i is not less than $S^{-2i} \ge S^{2-2j} > 2S^{1-2j}$. If $i > j$, then $S^{2j}(t_2 - t_1)$ is integer, whereas any proper time of M_i, being multiplied by S^{2j}, has nonzero fractionality, as follows from the estimates of Lemma 9. □

The following lemma is an immediate corollary to the definition of proper times.

Lemma 12. *If σ is a tight nonpartitioning schedule for $O(c, m)$, then, for any correct machine M_i without preemptions, all switching points except* 0 *and m are proper.*

Improper components. We want to prove that a tight nonpartitioning schedule σ for $O(C, m)$ without integral loops and incorrect machines has at least $m - 2$ preemptions. If σ has preemptions on all intermediate machines, then the number of preemptions is at least $m - 2$ and our claim is valid. It remains to consider the case when some intermediate machine has no preemptions.

Denote by s_i and f_i the starting and finishing times of the common job I on the machine M_i. Note that I has no preemptions on M_i if and only if $f_i - s_i = 1$. Let M_i, M_j be a pair of machines without preemptions such that $f_i \le s_j$. We call such a pair *σ-successive* if there is no machine M_k without preemptions in σ such that $f_i \le s_k \le f_k \le s_j$.

It is obvious that a successive pair cannot be formed by extremal machines; therefore, at least one of the machines of the pair is intermediate. Since a strongly improper time for an intermediate machine is improper, s_j is in fact strictly greater than f_i. Hence, there exists at least one machine with preemptions to which an I-component is assigned within the interval $[f_i, s_j]$ (since the job I must be being processed at any moment of time before the end of σ).

We say that an I-component (p, k, q) *precedes* another I-component (p', k', q') if $q \le p'$, and that an I-component (p, k, q) is (i, j)-*intermediate* if $f_i \le p \le q \le s_j$.

An I-component (p, k, q) is said to be *improper* for an extremal machine M_k if $q - p < 1$ and $0 < p < q < m$. Such a component is said to be *improper* for an intermediate machine M_k if, in addition, both p and q are improper times for M_k.

An improper I-component both ends of which are strongly improper is itself called *strongly improper.*

Lemma 13. *Let M_i, M_j be a σ-successive pair. Then*

(1) *there exists an (i, j)-intermediate improper I-component*;

(2) *if the earliest scheduled (i, j)-intermediate improper I-component belongs to an intermediate machine, then its starting time is strongly improper*;

(3) *if the last scheduled (i, j)-intermediate improper I-component belongs to an intermediate machine, then its finishing time is strongly improper.*

Proof. The difference $s_j - f_i$ is not an integer; otherwise s_j an f_i would be proper for both machines M_i and M_j, which is impossible in view of Lemma 10, because at least one of these machines is intermediate. Since the length of I coincides with the makespan of the schedule, this job is being processed during the whole interval $[f_i, s_j)$; in particular, there is an I-component in this interval. Let (p_1, k_1, q_1) be the earliest nonunit I-component scheduled after f_i. If M_{k_1} is an extremal machine, then this I-component is improper just because it is nonunit, and assertion (1) is valid. If M_{k_1} is an intermediate machine, then p_1 may differ from f_i only by an integer (by the sum of processing times of unit I-components); therefore, it is proper for M_i and strongly improper for M_{k_1}. If q_1 is not proper for M_{k_1}, then this component is improper, and the validity of assertions (1) and (2) is proved. Suppose that q_1 is a proper time for M_{k_1}. Then we continue to search for an improper I-component. The difference $s_j - q_1$ is again noninteger, because it is the difference of proper times of different machines. Hence, there is at least one nonunit I-component scheduled between q_1 and s_j. Let (p_2, k_2, q_2) be the earliest such component. If $k_2 = 1$ or $k_2 = m$, then this component is improper, and the lemma is proved. Suppose that $1 < k_2 < m$. Since σ has no integral loops, $k_2 \ne k_1$. Since the moment p_2 is the sum of q_1 and an integer, it is proper for M_{k_1} and is therefore strongly improper for M_{k_2}. If q_2 is improper, then (p_2, k_2, q_2) is improper, and assertions (1) and (2) are proved. Otherwise, we continue the search and find the next I-component (p_3, k_3, q_3), and so on. This process cannot recur more times than there are preemptions in σ, and we obtain an improper I-component with a strongly improper starting time. Thus, assertions (1) and (2) are proved.

The proof of assertion (3) is similar to the above proof; one should just consider the latest components instead of the earliest ones. □

Marked components. We apply Lemma 13 in order to mark intermediate improper I-components for any σ-successive pair (M_i, M_j). Namely, among (i, j)-intermediate improper I-components on an extremal machine, we mark the earliest one. For an intermediate machine, if there is only one (i, j)-intermediate improper I-component, we mark this component (it will have strongly improper starting and finishing times in view of Lemma 13). If there are more than one (i, j)-intermediate improper I-component, we mark the earliest and the latest of them; a pair of such components is called a *twin couple.* The I-components of a twin couple may be assigned either to the same machine or to different machines.

By $\mathrm{mr}(M)$ we will denote the number of marked I-components assigned to the machine M. If $\mathrm{mr}(M) > 0$, then the machine M is said to be *marked.* Let M be a machine with preemptions.

Then we define the *reduced number of preemptions* on M as $\mathrm{pr}^-(M) = \mathrm{pr}^\sigma(M) - 1$. For a machine without preemptions, the reduced number of preemptions is assumed to be zero.

Lemma 14. *If M is an extremal marked machine, then* $\mathrm{pr}^-(M) \geq \mathrm{mr}(M)$.

Proof. Notice that the number of switchings on M is at least $2\,\mathrm{mr}(M) + 2$. Applying Lemma 3 for $n = 2$ and $s = 2\,\mathrm{mr}(M) + 2$, we obtain $\mathrm{pr}^\sigma(M) \geq 2\,\mathrm{mr}(M) + 2 - 2 - 1 = 2\,\mathrm{mr}(M) - 1$. If $\mathrm{mr}(M) > 1$, then $2\,\mathrm{mr}(M) - 1 \geq \mathrm{mr}(M) + 1$; if $\mathrm{mr}(M) = 1$, then the job I has at least one preemption and the fixer also has at least one preemption, because the marked component is processed in the intermediate position on M. □

Lemma 15. *If $\mathrm{mr}(M_k) > 2$ for an intermediate machine M_k, then* $\mathrm{pr}^-(M_k) \geq \mathrm{mr}(M_k)$.

Proof. First, consider the case when there is a nonmarked I-component on the machine M_k, which is equivalent to the inequality $\mathrm{pr}(\sigma(I, M_k)) \geq \mathrm{mr}(M_k)$. (If there are more than one nonmarked component, then the assertion of the lemma is obviously valid.) In this case, to prove the lemma, it suffices to find a split job on M_k that is different from I. Let p_1 be the starting time of the earliest marked I-component on M_k. If only components of jobs different from I are processed in the interval $(0, p_1)$, then at least one of these components must be a fractional (incomplete) component of an appropriate job since the length of this interval is improper. If there is a nonmarked I-component in this interval, then we consider the interval (q, m) whose left end is the finishing time of the last marked I-component. Now, regarding this interval, we can state that it does not contain I-components and that its length is again improper. Therefore, we find that there exist split jobs different from I.

So, we may assume that $\mathrm{pr}(\sigma(I, M_k)) < \mathrm{mr}(M_k)$, i.e., that all I-components on M_k are marked. Next, we consider the case when there are at least three marked I-components (p_1, k, q_1), (p_2, k, q_2), and (p_3, k, q_3), $p_1 < p_2 < p_3$, that do not contain twin couples. Hence, all these components are intermediate for three different σ-successive pairs of machines. In this case, each of the intervals $(0, p_1)$, (q_1, p_2), (q_2, p_3), and (q_3, m) contains a subinterval of unit length in which only components of jobs different from I are processed. Hence, in each of these intervals, a fixer is scheduled. If the same fixer is scheduled in two different intervals, then it has a preemption; if a fixer is scheduled in three different intervals, then it has at least two preemptions. It is easy to see that in all cases there are at least two preemptions of the fixers. On the other hand, the number of preemptions of the job I is at least $\mathrm{mr}(M_k) - 1$. As a result, the total number of preemptions on M_k is not less than $\mathrm{mr}(M_k) - 1 + 2 = \mathrm{mr}(M_k) + 1$.

Now, consider the case when there are exactly three marked components on M_k of which the first two form a twin couple. Since p_1 and q_3 are improper, the intervals $(0, p_1)$ and (q_3, m) contain a fractional component of some job, say, J_1 for $(0, p_1)$ and J_2 for (q_3, m). If $J_1 \neq J_2$, then $\mathrm{pr}^\sigma(M_k) \geq \mathrm{pr}(\sigma(I, M_k)) + 2 = \mathrm{mr}(M_k) + 1$, and the validity of the lemma is proved. Suppose that $J_1 = J_2$. Notice that Lemma 13 implies that p_1 and q_2 are strongly improper for M_k. Moreover, either p_3 or q_3 is strongly improper. First, assume that p_3 is strongly improper. Then the interval (q_2, p_3) has improper length according to Lemma 11; therefore, it contains a fractional component of some job, say, J_3. If $J_3 \neq J_1$, then arguments similar to those used for J_2 prove our lemma. If $J_3 = J_1$, then J_1 has at least two preemptions, so $\mathrm{pr}^\sigma(M_k) \geq \mathrm{pr}(\sigma(I, M_k)) + \mathrm{pr}(\sigma(J_1, M_k)) \geq \mathrm{mr}(M_k) - 1 + 2$, and the lemma is proved.

Now, suppose that q_3 is strongly improper. Since all I-components on M_k are marked, the sum of lengths of all I-intervals (p_i, q_i) is 1, while the sum of lengths of the intervals (q_1, p_2) and (q_2, p_3) is $q_3 - p_1 - 1$ and is improper by Lemma 11. Hence, one of these intervals must contain a fractional component of the job. Now, the proof can be completed in the same way as this was done in the cases analyzed above.

The case when a twin couple follows a marked I-component is analogous. Finally, it remains to analyze the case when there are exactly four marked I-components on M_k that form two different twin couples. Then each of the three intervals $(0, p_1)$, (q_2, p_3), and (q_4, m) has improper length and, hence, contains a fractional component of a job different from I, so we again find an additional preemption. □

Lemma 16. *If* $\mathrm{mr}(M_k) = 2$ *and at least one marked component on* M_k *is strongly improper, then* $\mathrm{pr}^-(M_k) \geq 2$.

Proof. Suppose that (p, k, q) is a strongly improper marked I-component. In this case, the second marked I-component (p', k, q') has one strongly improper end. Suppose that $q < p'$ and q' is strongly improper (hence, p' is improper). The length of the interval $(0, p)$ is improper for M_k; hence, this interval contains a fractional component of some job, say J_1. The same is valid for the interval (q', m). If this interval contains a fractional component of some other job J_2, then we have already found three split jobs: I, J_1, and J_2, and the lemma is proved. Suppose $J_2 = J_1$. If I has more than two components on M_k, then I has two preemptions and J_1 gives another one, so we again have three preemptions on M_k. Suppose I has exactly one preemption on M_k. In this case, the length of the interval (q, p') is less by one than the length of the interval (p, q'); the latter is improper as the difference of two strongly improper times (Lemma 11). Thus, the length of (q, p') is improper, and this interval contains a fractional component of a job. If this component belongs to the job J_1, then J_1 has two preemptions, and together with the preemption of I we obtain three preemptions on M_k. If this component belongs to some other job, then we again obtain three split jobs.

The case when p' is strongly improper is simpler, because we immediately find that the length of the interval (q, p') is improper as the difference of strongly improper moments of time. The case $q' < p$ is analogous. □

Lemma 17. *Every machine* M_k *with marked jobs has at least two preemptions.*

Proof. If I has more than one preemption on M_k, then we already have the promised number of preemptions. Suppose that I has exactly one preemption and that (p, k, q) is a marked I-component. Either the interval $(0, p)$ or the interval (q, m) does not contain the components of the job I. Since this interval, in addition, has improper length, it must contain a fractional component of some job different from I. Thus, we have found two split jobs on M_k. □

Proof of Lemma 2. Finally, we are ready to prove the main lemma. As already mentioned above, it suffices to consider the case when some intermediate machine has no preemptions (otherwise there are at least $m - 2$ preemptions). Let k be the number of machines without preemptions and $m - k$ be the number of machines with preemptions. The total number of preemptions $\mathrm{pr}(\sigma)$ is equal to $m - k$ plus the sum of the reduced numbers of preemptions. Hence, it suffices to prove that $\mathrm{pr}^-(\sigma) \geq m - 2 - (m - k) = k - 2$.

Denote by p the number of σ-successive pairs for which a twin couple is marked. Then the total number of marked components is $k - 1 + p$. Indeed, the number of σ-successive pairs is $k - 1$. Each such pair generates one or two marked components, while the number of pairs generating two components is p.

We call a machine *distinguished* if it contains exactly two marked components each of which has a twin (these two components may belong to different twin couples). Denote by q the number of all distinguished machines. Since the total number of twins is $2p$ and the number of twins assigned to distinguished machines is $2q$, we obtain $2q \leq 2p$, i.e., $q \leq p$.

By Lemma 17, each distinguished machine has at least two preemptions. So, the distinguished machines make a total contribution of at least q to $\mathrm{pr}^-(\sigma)$. The number of marked components assigned to nondistinguished machines is equal to $(k - 1 + p) - 2q \geq k - 1 - q$. By Lemmas 14–16, each marked component on a nondistinguished machine makes a contribution of at least 1 to $\mathrm{pr}^-(\sigma)$. Hence, the sum of reduced preemptions is estimated from below by the number $q + (k - 1 - q) = k - 1$ (this estimate is even better than required; it is valid only if there exists an intermediate machine without preemptions).

ACKNOWLEDGMENTS

This work is supported by the Russian Science Foundation under grant 14-50-00005.

REFERENCES

1. T. Gonzalez and S. Sahni, "Open shop scheduling to minimize finish time," J. Assoc. Comput. Mach. **23**, 665–679 (1976).
2. E. L. Lawler and J. Labetoulle, "On preemptive scheduling of unrelated parallel processors by linear programming," J. Assoc. Comput. Mach. **25**, 612–619 (1978).
3. E. V. Shchepin and N. Vakhania, "On the geometry, preemptions and complexity of multiprocessor and shop scheduling," Ann. Oper. Res. **159**, 183–213 (2008).
4. E. V. Shchepin and N. Vakhania, "A note on the proof of the complexity of the little-preemptive open-shop problem," Ann. Oper. Res. **191**, 251–253 (2011).
5. E. Shchepin and N. Vakhania, "Little-preemptive scheduling on unrelated processors," J. Math. Model. Algorithms **1**, 43–56 (2002).
6. E. V. Shchepin and N. Vakhania, "Task distributions on multiprocessor systems," in *Theoretical Computer Science: Exploring New Frontiers of Theoretical Informatics: Proc. Int. Conf. IFIP TCS 2000, Sendai (Japan), 2000* (Springer, Berlin, 2000), Lect. Notes Comput. Sci. **1872**, pp. 112–125.
7. E. V. Shchepin, "On the geometry of multiprocessor distributions," Tr. Mat. Inst. im. V.A. Steklova, Ross. Akad. Nauk **239**, 323–331 (2002) [Proc. Steklov Inst. Math. **239**, 306–314 (2002)].
8. E. V. Shchepin and N. Vakhania, "An optimal rounding gives a better approximation for scheduling unrelated machines," Oper. Res. Lett. **33**, 127–133 (2005).

Translated by I. Nikitin