ISSN 1064-2307, Journal of Computer and Systems Sciences International, 2018, Vol. 57, No. 2, pp. 222–229. © Pleiades Publishing, Ltd., 2018.
Original Russian Text © M.G. Furugyan, 2018, published in Izvestiya Akademii Nauk, Teoriya i Sistemy Upravleniya, 2018, No. 2, pp. 52–59.

SYSTEMS ANALYSIS
AND OPERATIONS RESEARCH

Scheduling in Multiprocessor Systems with Additional Restrictions

M. G. Furugyan

Dorodnicyn Computing Center, Russian Academy of Sciences, Moscow, 119991 Russia
e-mail: rtsccas@ya.ru

Received October 19, 2017; in final form, November 14, 2017

Abstract—An admissible multiprocessor preemptive scheduling problem is solved for the given execution intervals. In addition, a number of generalizations are considered—interprocessor communications are arbitrary and may vary in time; costs for processing interruptions and switches from one processor to another are taken into account; and besides the processors, additional resources are used. Algorithms based on reducing the original problem to finding paths of a specific length in a graph, a flow problem, and an integer system of linear restrictions are developed.

DOI: 10.1134/S1064230718020077

INTRODUCTION

The design and operation of complex technical objects (planes, space observation systems, pipeline systems, nuclear reactors, etc.) frequently involve real-time multiprocessor computing systems. One of the principal problems to be solved when developing software for such systems is to construct an admissible schedule for the execution of software modules that shows which resources of the computing system are to be allocated to each module and when. Many works deal with preemptive scheduling without taking into account the restrictions on communications between processors and the costs for processing interruptions and switches from one processor to another. We highlight some works such as [1] (identical processors), [2] (arbitrary processors, same execution intervals), and [3, 4] (arbitrary processors, arbitrary execution intervals). Work [5] deals with the multiprocessor scheduling problem in a system with arbitrary interprocessor communications and costs for processing interruptions and switches taken into account. In [6], the conditions leading to the existence of a polynomial algorithm for finding an admissible schedule in a multiprocessor system with an incomplete interprocessor communication graph are obtained. In [1–6], no additional resource was assumed to exist. An admissible scheduling problem in a multiprocessor system with additional resources was considered in [7] (identical processors, one type of additional resource), [8] (arbitrary processors, one type of additional resource), [9] (identical processors, several types of additional resources), and [10] (arbitrary processors, several types of additional resources). In a number of cases, similar problems can be reduced to minimax and grid problems and methods that solve them given in [11–19].

In this work, we study the admissible multiprocessor preemptive scheduling problem for the cases when (1) there are restrictions on interprocessor communications that may vary in time; (2) costs for processing interruptions and switches are taken into account; and (3) besides processors, there are additional resources in the system with job execution times linearly depending on the number of those resources allocated to them. These problems can be reduced to finding paths of a specific length in a graph, flow problems, and an integer system of linear restrictions. Unlike [5, 6], we obtained new necessary and sufficient existence conditions for the admissible schedule.

1. STATEMENT OF THE PROBLEM

We consider a computing system consisting of m processors. The performance of the jth processor is s_j, $j = \overline{1,m}$. There is a set of jobs (tasks) $N=\{\overline{1,n}\}$ to be executed. Each task $i \in N$ has its own execution interval $[b_i, f_i]$ (the job i can be started not earlier than the instant b_i and should be finished not later than the instant f_i) and the workload Q_i of the processors required for it to be executed. The workload of the processors is represented as a set of successive cycles. Cycles of all processors are synchronized in time; i.e., the beginnings and ends of the cycles of all processors coincide. One cycle is a time unit and the variables b_i and f_i, $i \in N$, are given in cycles numbered from 1 to T, where

$$\min_{i \in N} b_i = 1, \quad \max_{i \in N} f_i = T.$$

Thus, the job i can be started not earlier than the b_ith cycle starts and should be finished not later than the f_ith cycle. The performance s_j is the workload of the processor j during one cycle. To execute the job i fully, the processor j requires $\lceil Q_i/s_j \rceil$ cycles. On the fixed cycle, each processor cannot execute more than one job while each job is executed by not more than one processor. Task execution may involve interruptions and switches from one processor to another. Interprocessor communications may vary in time and are given by the array I of dimension $m \times T \times m \times T$. Moreover, $I(j_1, k_1, j_2, k_2) = 1$ if, while it is being executed on the processor j_1, the job i can be interrupted at the end of the cycle k_1 and resumed on the processor j_2 (it may be the same one, i.e., $j_1 = j_2$) at the beginning of the cycle k_2; and $I(j_1, k_1, j_2, k_2) = 0$ if such a switch is not possible, $1 \le j_1,\ j_2 \le m, 1 \le k_1 < k_2 \le T$. Interrupting the task on the processor j_1 at the end of the cycle k_1 and switching it at the beginning of the cycle k_2 to the processor j_2 or resuming it on the same processor ($j_1 = j_2$) requires additional work from the processors j_1 and j_2, which totals $\tau(j_1, k_1, j_2, k_2)$.

Find out whether there exists an admissible execution schedule for the jobs N (i.e., the schedule such that each job is fully executed in its execution interval $(b_i,\ f_i\,]$) and find it if it exists.

2. *NP* COMPLEXITY OF THE PROBLEM INVOLVED

We show that the stated problem is *NP* hard. To do this, it is sufficient to prove that the original problem in the form of property recognition (when the only thing we need to do is to find whether an admissible schedule exists) is *NP* complete. We can easily show that this problem belongs to the *NP* class. Moreover, the known *NP* complete decomposition problem [20, 21] (the set of integers $a_1, a_2, \ldots, a_n$ is given; can it be decomposed into two nonoverlapping subsets with an identical sum?) can be polynomially reduced to it. To prove this, it is sufficient to put, in the original problem, $N = \{\overline{1, n}\}$, $m = 2$, $s_1 = s_2 = 1$, $b_i = 1, f_i = T, Q_i = a_i, i \in N$,

$$T = \left[\left(\sum_{i \in N} a_i\right)/2\right]$$

and $\tau(j_1, k_1, j_2, k_2) = 1$ for all j_1, k_1, j_2, k_2 such that $I(j_1, k_1, j_2, k_2) = 1$. In this case, interruptions and switches lead to the fact that not all jobs will be finished by the end of the Tth cycle. Therefore, the answer to the decomposition problem is positive if and only if the answer to the problem involved is positive.

3. REPRESENTATION OF THE ADMISSIBLE SCHEDULE

We give the admissible execution schedule for the jobs N as the lists $A(i)$, $i \in N$, where

$$A(i) = \{(j_1,\ k_1), (j_2,\ k_2),\ \ldots,\ (j_h,\ k_h)\}, \quad 1 \le j_1,\ j_2,\ \ldots,\ j_h \le m, \quad 1 \le k_1 < k_2 < \ldots < k_h \le T. \tag{3.1}$$

The pair (j, k) in the list $A(i)$ means that the job i is executed by the processor j in the cycle k. Obviously, if for two jobs $i_1,\ i_2 \in N,\ i_1 \ne i_2,\ (j, k) \in A(i_1),\ (j', k) \in A(i_2)$, then $j \ne j'$ since one processor cannot execute more than one job in one cycle. Moreover, if $(j_t,\ k_t) \in A(i)$ and $(j_{t+1},\ k_{t+1}) \in A(i)$, $1 \le t < h$, are two successive pairs in $A(i)$, then $I(j_t,\ k_t,\ j_{t+1},\ k_{t+1}) = 1$; i.e., there exists the respective communication between the processors j_t and j_{t+1}.

4. CONSTRUCTING A GRID MODEL

To solve the stated problem similarly to [5], we construct the grid $G_1 = (V_1,\ E_1)$, where V_1 is the set of nodes, E_1 is the set of arcs (see Fig. 1). The set V_1 consists of the following vertices: $u_1, u_2, \ldots, u_n$ are the sources, $w_1, w_2, \ldots, w_n$ are the sinks, and $x_{jk}, y_{jk}, j = \overline{1, m}$, $k = \overline{1, T}$ are the internal vertices. The set E_1 consists of the arcs $(u_i x_{jk})$, $i \in N, j = \overline{1, m}$, $k \in [b_i,\ f_i]$; $(x_{jk},\ y_{jk}), j = \overline{1, m}, k = \overline{1, T}$; $(y_{j_1 k_1},\ x_{j_2 k_2})$ for all $1 \le j_1, j_2 \le m$

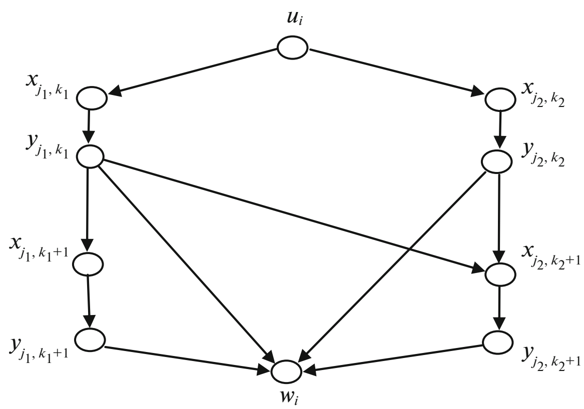

Fig. 1. Fragment of grid G_1.

and k_1, k_2 such that $1 \le k_1 < k_2 \le T, I(j_1,k_1,j_2,k_2) = 1$; (y_{jk}, w_i), $i \in N$, $j = \overline{1,m}$, $k \in [b_i, f_i]$. The node u_i corresponds to the beginning of the execution of the job i and the node w_i corresponds to the end of the execution of the job i. The node x_{jk} corresponds to the processor j and the beginning of the kth cycle, while the node y_{jk} corresponds to the processor j and the end of the kth cycle. The arc (u_i, x_{jk}) in the grid G_1 means execution of the job i can be started at the beginning of the kth cycle by the processor j, while the arc (y_{jk}, w_i) means execution of the job i by the processor j can be finished at the end of the kth cycle. The arc (x_{jk}, y_{jk}) corresponds to some task being executed by the processor j in the kth cycle, while the arc $(y_{j_1k_1}, x_{j_2k_2})$ corresponds to switching the execution of some job from the processor j_1 at the end of the k_1th cycle to the processor j_2 (it may be the same if $j_1 = j_2$) at the beginning of the k_2th cycle. Note that $|V_1| = 2n + 2mT$, and $|E_1| \le mT(0.5\ m(T-1) + 2n + 1)$.

5. NECESSARY AND SUFFICIENT EXISTENCE CONDITIONS FOR AN ADMISSIBLE SCHEDULE

We assume that each arc $(a, b) \in E_1$ of the grid G_1 has the length $l(a, b)$ specified as $l(u_i, x_{jk}) = l(y_{jk}, w_i) = 0$, $l(x_{jk}, y_{jk}) = s_j$, $l(y_{j_1k_1}, x_{j_2k_2}) = -\tau(j_1,k_1,j_2,k_2)$, $i \in N$, $1 \le j, j_1, j_2 \le m$, $1 \le k, k_1, k_2 \le T, k_1 < k_2$. Then, an arbitrary path Π_i from u_{i} to w_{i} corresponds to some execution schedule of the job $i \in N$, according to which it is executed within its execution interval $[b_i, f_i]$. If $(x_{jk}, y_{jk}) \in \Pi_i$, the task i is executed in the cycle k by the processor j, and the structure of the grid G_1 is such that $k \in [b_i, f_i]$; i.e., the job i is executed within its execution interval. The workload of the processor j associated with executing the task i in the cycle k is s_j. If $(x_{j_1k_1}, y_{j_2k_2}) \in \Pi_i$, it means that after it was executed in the cycle k_1 by the processor j_1, the job i was switched to the processor j_2 at the beginning of the cycle k_2. The total workload of the processors associated with executing the task i is decreased by the value $\tau(j_1, k_1, j_2, k_2)$. The length $l(\Pi_i)$ of the path Π_i is the total workload of the processors associated with executing the task i. If $l(\Pi_i) \ge Q_i$, the execution schedule for the task i that corresponds to this path is admissible. Two nonoverlapping paths Π_{i_1} and Π_{i_2}, $i_1, i_2 \in N, i_1 \ne i_2$, correspond to the execution schedules of the jobs i_1, i_2 that can be implemented in parallel. Thus, we arrive at the following proposition.

Lemma 1. For the admissible execution schedule for the jobs N to exist, it is necessary and sufficient that there exist n pairwise disjoint paths $\Pi_1, \Pi_2, \ldots, \Pi_n$ in the grid G_1 such that

$$l(\Pi_i) \ge Q_i, \quad i \in N. \tag{5.1}$$

Proof. *Necessity.* Suppose there exists an admissible execution schedule for the jobs N. Then, for each task $i \in N$ there exists the path Π_i from u_i to w_i such that inequality (5.1) holds for it. Since one processor cannot execute more than one job in one cycle, these paths are pairwise disjoint.

Sufficiency was proved at the beginning of this section. The lemma is proved.

Table 1. Parameters of grid G_2

Arcs	L	U	$\bar{C}$
(u_i^0, u_i)	1	1	0
(w_i, w_i^0)	1	1	0
(u_i, x_{jk})	0	1	0
(x_{jk}, y_{jk})	0	1	s_j
$(y_{j_1k_1}, x_{j_2k_2})$	0	1	$-\tau(j_1, k_1, j_2, k_2)$
(y_{jk}, w_i)	0	1	0

The path Π_i specifies the admissible execution schedule for the job i and the set of paths $\Pi_1, \Pi_2, \ldots, \Pi_n$ specifies the admissible execution schedule for the jobs N.

We state the original problem as a flow one. We define the grid $G_2 = (V_2, E_2)$, where V_2 is the set of nodes and E_2 is the set of arcs, by adding the grid G_1 with the nodes u_i^0, w_i^0 and the arcs (u_i^0, u_i), (w_i, w_i^0), $i \in N$. In the grid G_2, we consider an integer n-product flow. The nodes u_i^0 and w_i^0 are, respectively, the source and sink of the ith product. Each arc $(a, b) \in E_2$ has three parameters—the lower $L(a, b)$ and upper $U(a, b)$ boundaries of the flow and the cost $\bar{C}(a,b)$ of the flow unit. The values of these parameters for the arcs of the grid G_2 are given in Table 1, $i \in N$, $1 \le j, j_1, j_2 \le m$, $k_1 < k_2, 1 \le k \le T$, $k_1, k_2 \in [b_i, f_i]$.

Suppose g is the integer n-product flow in the grid G_2 and suppose g_i is the flow of the ith product. We introduce the designation

$$S(g, E) = \sum_{(a,b)\in E} \bar{C}(a,b)g(a,b).$$

Then, $C(g_i) = S(g_i, E_2)$ is the cost of the flow g_i. Note that since the flow g is integer (i.e., $g_i(a,b)$ is integer for all arcs $(a, b) \in E_2$), by the definitions of the lower and upper boundaries of the flows over the arcs (see Table 1), the variables $g_i(a,b)$ and $g(a,b)$ take the values 0 and 1 for each arc $(a, b) \in E_2$.

Lemma 2. For an admissible schedule to exist, it is necessary and sufficient that the integer n-product flow g exists in the grid G_2 such that u_i^0 and w_i^0 are, respectively, the source and sink of the ith product and

$$C(g_i) \ge Q_i \tag{5.2}$$

for all $i \in N$.

Proof. *Necessity.* Suppose there exists the admissible execution schedule A for the jobs N. By (3.1), $A(i) = \{(j_1, k_1), (j_2, k_2), \ldots, (j_h, k_h)$. We specify the flow g by the following rules (1) $g_i(u_i^0, u_i) = g_i(w_i^0, w_i) = 1$; (2) $g_i(u_i, x_{j_1k_1}) = g_i(y_{j_hk_h}, w_i) = 1$; (3) if $(j_t, k_t) \in A(i)$, then $g_i(x_{j_tk_t}, y_{j_tk_t}) = 1$; and (4) if $(j_t, k_t) \in A(i)$, $(j_{t+1}, k_{t+1}) \in A(i)$, then $g_i(y_{j_tk_t}, x_{j_{t+1}k_{t+1}}) = 1$. We put the flow of the ith product over all other arcs $(a, b) \in E_2$ of the grid G_2 to be zero: $g_i(a, b) = 0$. According to rules (1)–(4), the flow unit of the ith product originates from the node u_i^0, follows the arcs (u_i^0, u_i), $(u_i, x_{j_1k_1})$, then the arcs (x_{jk}, y_{jk}) that correspond to the cycles and processors, on which the task i is being executed, the arcs $(y_{jk}, x_{j'k'})$ that correspond to its switches (there are no switches if $j = j'$), and the arcs $(y_{j_hk_h}, w_i)$ and (w_i^0, w_i). A flow specified in such a way is integer, the preservation conditions are met at internal grid nodes for each product, and the nodes u_i^0 and w_i^0, respectively, are the source and sink of the ith product. The upper and lower restrictions on the values of the flows g_i over the arcs are not violated since g_i take the values 0 or 1; moreover, one processor in one cycle cannot execute more than one job. In addition, inequality (5.2) holds for each $i \in N$. Indeed, the flow unit over the arc (x_{jk}, y_{jk}) increases the variable $C(g_i)$ by s_j, and the flow unit over the arc $(y_{j_1k_1}, x_{j_2k_2})$ decreases the variable $C(g_i)$ by $\tau(j_1, k_1, j_2, k_2)$. Since, according to the admissible schedule, the total workload of the processors to execute the task i is not less than Q_i, inequality (5.2) holds for all $i \in N$.

Sufficiency. Suppose in the grid G_2 there exists an integer n-product flow such that its ith product originates from the source u_i^0, it enters the sink w_i^0, and inequality (5.2) holds for all $i \in N$. Since $L(u_i^0, u_i) = U(u_i^0, u_i) = 1$, then $g_i(u_i^0, u_i) = 1$. Similarly, $g_i(w_i, w_i^0) = 1$. Since the flow g_i is integer, its value over each arc of the grid G_2 is 0 or 1. Since the flow g_i is preserved at each internal node, there exists the unique path Π_i from u_i^0 to w_i^0, the value of the flow over which is 1. It follows from (5.2) and the structure of the grid G_2 that the path Π_i specifies the admissible schedule for the job i. It follows from the restrictions on the capacity of each arc that the paths $\Pi_1, \Pi_2, \ldots, \Pi_n$ are pairwise disjoint. Hence, this totality of the paths specifies the admissible execution schedule for the jobs N. Thus, if $(x_{jk}, y_{jk}) \in \Pi_i$, the job i is executed on the kth cycle by the processor j; if $(y_{j_1k_1}, x_{j_2k_2}) \in \Pi_i$, the job i is switched from the processor j_1 to the processor j_2 at the end of the cycle k_1, where it is resumed at the beginning of the cycle k_2 (there is no switch if $j_1 = j_2$). The lemma is proved.

Thus, to construct the admissible execution schedule for the jobs N, we need to find in the grid G_2 the multiproduct flow that satisfies the hypotheses of Lemma 2 and, if it exists, construct the schedule in the way described during the proof of sufficiency in Lemma 2. If there is no such flow, there is also no admissible schedule.

Now, we describe the necessary and sufficient existence conditions for the admissible schedule, which were stated in Lemma 2, as an integer system of linear restrictions. Find nonnegative integer values $g_i(u_i^0, u_i)$, $g_i(u_i, x_{jk})$, $g_i(x_{jk}, y_{jk})$, $g_i(y_{j_1k_1}, x_{j_2k_2})$, $g_i(y_{jk}, w_i)$, $g_i(w_i, w_i^0)$, $i \in N$, $1 \le j, j_1, j_2 \le m$, $1 \le k_1 < k_2 \le T$, $1 \le k \le T$, $I(j_1, k_1, j_2, k_2) = 1$, $(u_i, x_{jk}) \in E_2$, $(y_{jk}, w_i) \in E_2$ that satisfy the following restrictions:

$$g_i(u_i^0, u_i) = 1, \quad g_i(w_i, w_i^0) = 1, \quad i \in N, \tag{5.3}$$

$$g_i(u_i, x_{jk}) \le 1, \quad g_i(y_{jk}, w_i) \le 1, \quad i \in N, \quad j = \overline{1,m}, \quad k \in [b_i, f_i], \tag{5.4}$$

$$\sum_{i \in N} g_i(x_{jk}, y_{jk}) \le 1, \quad \sum_{i \in N} g_i(y_{j_1k_1}, x_{j_2k_2}) \le 1, \quad 1 \le j_1, \quad j_2 \le m, \quad 1 \le k_1 < k_2 \le T, \quad 1 \le k \le T, \tag{5.5}$$

$$g_i(u_i^0, u_i) = \sum_{\substack{j=\overline{1,m},\\ k \in [b_i, f_i]}} g_i(u_i, x_{jk}), \quad i \in N, \tag{5.6}$$

$$\sum_{\substack{j=\overline{1,m},\\ k \in [b_i, f_i]}} g_i(y_{jk}, w_i) = g_i(w_i, w_i^0), \quad i \in N, \tag{5.7}$$

$$g_i(u_i, x_{j_2k_2}) + \sum_{\substack{j_1=\overline{1,m},\\ k_1<k_2, k_1 \in [b_i, f_i],\\ I(j_1,k_1,j_2,k_2)=1}} g_i(y_{j_1k_1}, x_{j_2k_2}) = g_i(x_{j_2k_2}, y_{j_2k_2}), \quad i \in N, \quad j_2 = \overline{1,m}, \quad k_2 \in [b_i, f_i], \tag{5.8}$$

$$g_i(x_{j_1k_1}, y_{j_1k_1}) = \sum_{\substack{j_2=\overline{1,m},\\ k_1<k_2, k_2 \in [b_i, f_i],\\ I(j_1,k_1,j_2,k_2)=1}} g_i(y_{j_1k_1}, x_{j_2k_2}) + g_i(y_{j_1k_1}, w_i), \quad i \in N, \quad j_1 = \overline{1,m}, \quad k_1 \in [b_i, f_i], \tag{5.9}$$

$$\sum_{j=1}^{m} \sum_{k=1}^{T} g_i(x_{jk}, y_{jk})\, s_j - \sum_{j_1=1}^{m} \sum_{j_2=1}^{m} \sum_{1 \le k_1 < k_2 \le T} \tau(j_1, k_1, j_2, k_2)\, g_i(y_{j_1k_1}, x_{j_2k_2}) \ge Q_i, \quad i \in N. \tag{5.10}$$

Equalities (5.3) ensure the flow unit of the ith product is delivered from the source u_i^0 to the sink w_i^0. Inequalities (5.4) and (5.5) ensure the upper restrictions on the flows over the arcs are met, equalities (5.6)–(5.9) ensure the flows of each product are preserved in all internal nodes of the grid G_2, and inequalities (5.10) are equivalent to inequalities (5.2) from Lemma 2. Note that system (5.3)–(5.10) includes $O(nmT + m^2T^2)$ variables and $O(nmT + m^2T^2)$ linear restrictions.

6. NECESSARY EXISTENCE CONDITIONS FOR AN ADMISSIBLE SCHEDULE

In this section, we obtain the necessary existence conditions for an admissible schedule in the problem involved. They are significantly easier to check than the necessary and sufficient conditions from Section 5.

We consider the grid $G_3 = (V_3, E_3)$ constructed from the grid G_1 with the addition of the nodes u_0 (the source) and w_0 (the sink) and the arcs (u_0, u_i) and (w_i, w_0), $i \in N$. The values of the parameters of the arcs (u_0, u_i) and (w_i, w_0) are $L(u_0, u_i) = L(w_i, w_0) = 0$, $U(u_0,u_i) = U(w_i,w_0) = 1$, $\bar{C}(u_0, u_i) = \bar{C}(w_i, w_0) = 0$. The values of the parameters of the rest of the arcs of the grid G_3 are the same as in the grid G_1. In the grid G_3, we consider the one-product flow g from the source u_0 to the sink w_0 and suppose $C(g) = S(g, E_3)$ is its cost,

$$Q = \sum_{i \in N} Q_i .$$

Lemma 3. For an admissible schedule to exist in the problem involved, the grid G_3 should have a flow g such that

$$C(g) \geq Q. \tag{6.1}$$

Proof. We assume that there is an admissible schedule in the problem involved. Then, by Lemma 2, there exists the integer n-product flow g_i in the grid G_2 such that inequality (5.2) holds for all $i \in N$ (g_i is the flow of the ith product in the grid G_2). We specify the flow g in the grid G_3 as $g(u_0, u_i) = g(w_i, w_0) = 1$ for all $i \in N$, $g(a, b) = g_i(a, b)$ for all other arcs $(a, b) \in E_3$ of the grid G_3. Obviously, g is the flow. Inequality (6.1) holds due to (5.2). The lemma is proved.

Thus, we can propose the following algorithm to check whether the hypothesis of Lemma 3 is met. In the grid G_3, find the flow g of the maximal cost. To do this, we can use Orlin's algorithm [21] or the out-of-kilter algorithm [22]. If relation (6.1) holds for the found flow, the necessary existence condition for the admissible schedule is met. As applied to the grid G_3, the complexity of the out-of-kilter algorithm is $O(nmT + m^2T^2)^2$.

We consider the grid $G_4 = (V_4, E_4)$ obtained from the grid G_1 by adding the latter with the nodes u_0 (the source) and w_0 (the sink) and the arcs (u_0, u_1), (w_i, u_{i+1}), $i = 1, 2, \ldots, n-1$, and (w_n, w_0). These arcs have the following parameters: $L(u_0, u_1) = U(u_0, u_1) = L(w_i,u_{i+1}) = U(w_i,u_{i+1}) = L(w_n,w_0) = U(w_n,w_0) = 1$, $\bar{C}(u_0,u_1) = \bar{C}(w_i,w_{i+1}) = \bar{C}(w_n,w_0) = 0$. The values of the parameters of the rest of the arcs of the grid G_4 are the same as in the grid G_1. We consider the one-product flow g from the source u_0 to the sink w_0 in the grid G_4 and suppose $C(g) = S(g, E_4)$ is its cost.

For the grid G_4, the proposition holds that coincides with what Lemma 3 states. To check whether the hypothesis of Lemma 3 holds for the grid G_4, we can also use the out-of-kilter algorithm.

Note that the structure of the grid G_4 and the parameters of its arcs are specified so that one unit of the flow from the source u_0 will be delivered to the sink w_0. The out-of-kilter algorithm operates so that if we choose the initial flow to be integer while searching for the flow g, the resulting flow will be also integer. The lower and upper flows over the arcs of the grid G_4 are specified so that the flow over each arc is 0 or 1. Thus, the arcs the flow over which is 1 form the path from u_0 to w_0, with its length being greater than or equivalent to Q (the length of the arc in this case coincides with its cost). Thus, we proved the following proposition.

Lemma 4. For the admissible schedule to exist, the grid G_4 should have a simple path from u_0 to w_0 with its length greater than or equivalent to Q.

7. PROBLEM WITH ADDITIONAL RESOURCES

In this section, we assume that, apart from the processors, the system has P types of additional nonrenewable resources. The total amount of the pth type of this resource is R_p, $p = \overline{1, P}$. If the task i has r_{ip} units of the additional resource of the pth type allocated to it, $i \in N$, $p = \overline{1, P}$, the workload Q_i of the processors associated with execution of the task i is

$$q_i(r) = d_i - \sum_{p=1}^{P} a_{ip} r_{ip} ,$$

where

$$r_{ip} \in [0, \ \bar{r}_{ip}], \quad i \in N, \quad p = \overline{1, P}; \tag{7.1}$$

$$\sum_{i\in N} r_{ip} \le R_p, \quad p = \overline{1,P}; \tag{7.2}$$

a_{ip}, d_i, and $\overline{r}_{ip}$ are the given variables; $a_{ip} \ge 0$, $d_i > 0$; $\overline{r}_{ip} \ge 0$; d_i is the workload of the processors to execute the task i if there are no additional resources allocated to it; and $q_i(\overline{r}) > 0$. Thus, $Q_i \in [q_i(\overline{r}); d_i]$. Find the allocation of resources $r_{ip}, i \in N$, $p = \overline{1,P}$ such that there exists an admissible schedule or establish that there exists no such resource allocation. This resource allocation should satisfy restrictions (7.1) and (7.2). We can find examples of similar problems in [8]. In this case, the lemmas we proved above can be stated as follows.

Lemma 5. For an admissible execution schedule for the jobs N to exist, it is necessary and sufficient that there exist the allocation of resources r_{ip} satisfying restrictions (7.1) and (7.2) and n pairwise disjoint paths $\Pi_1, \Pi_2, \ldots, \Pi_n$ in the grid G_1 such that $l(\Pi_i) \ge q_i(r)$, $i \in N$.

Lemma 6. For the admissible schedule to exist, it is necessary and sufficient that there exist the allocation of resources r_{ip} satisfying restrictions (7.1) and (7.2) and the integer n-product flow g in the grid G_2 such that u_i^0 and w_i^0 are, respectively, the source and sink of the ith product and $C(g_i) \ge q_i(r)$, $i \in N$.

Now, we describe the necessary and sufficient existence conditions for the admissible schedule, which were stated in Lemma 6, as an integer system of linear restrictions. Find such allocation of resources r_{ip}, $i \in N$, $p = \overline{1,P}$ and the nonnegative integer variables $g_i(u_i^0, u_i)$, $g_i(u_i, x_{jk})$, $g_i(x_{jk}, y_{jk})$, $g_i(y_{j_1k_1}, x_{j_2k_2})$, $g_i(y_{jk}, w_i)$, $g_i(w_i, w_i^0)$, $i \in N, 1 \le j, j_1, j_2 \le m, 1 \le k_1 < k_2 \le T, 1 \le k \le T$, $I(j_1, k_1, j_2, k_2) = 1$, $(u_i, x_{jk}) \in E_2$, and $(y_{jk}, w_i) \in E_2$ that satisfy restrictions (5.3)–(5.9), (7.1), (7.2), and the inequality

$$\sum_{j=1}^{m}\sum_{k=1}^{T} g_i(x_{jk}, y_{jk}) s_j - \sum_{j_1=1}^{m}\sum_{j_2=1}^{m}\sum_{1\le k_1<k_2\le T} \tau(j_1,k_1,j_2,k_2) g_i(y_{j_1k_1}, x_{j_2k_2}) \ge q_i(r), \quad i \in N.$$

This system includes $O(nmT + m^2T^2 + nP)$ variables and $O(nmT + m^2T^2 + nP)$ linear restrictions. We introduce the designation

$$Q(r) = \sum_{i\in N} q_i(r).$$

Lemma 7. For an admissible schedule to exist in the problem involved, resources r_{ip} should be allocated in such a way that satisfies restrictions (7.1) and (7.2) and the flow g in the grid G_3 for which $C(g) \ge Q(r)$.

Lemma 8. For the admissible schedule to exist, resources r_{ip} should be allocated in such a way that satisfies restrictions (7.1) and (7.2) and the simple path from u_0 to w_0 in the grid G_4, with a length greater than or equivalent to $Q(r)$.

CONCLUSIONS

We studied an admissible multiprocessor preemptive scheduling problem for the given execution intervals and under a number of additional restrictions—interprocessor communications are arbitrary and may vary in time; costs for processing interruptions and switches from one processor to another are taken into account; and apart from the processors, additional resources are used. We proved this problem to be *NP* hard. We developed the algorithms based on reducing the original problem to the problem of searching for paths of a specific length in the graph, a flow problem, and an integer system of linear restrictions.

REFERENCES

1. V. S. Tanaev, V. S. Gordon, and Ya. M. Shafranskii, *Scheduling Theory. One-Stage Systems* (Nauka, Moscow, 1984) [in Russian].
2. T. Gonzales and S. Sahni, "Preemptive scheduling of uniform processor systems," J. Assoc. Comput. Machin. **25**, 92–101 (1978).
3. A. Federgruen and H. Groenevel, "Preemptive scheduling of uniform machines by ordinary network flow technique," Manage. Sci. **32**, 341–349 (1986).
4. P. Brucker, *Scheduling Algorithms* (Springer, Heidelberg, 2007).

5. D. S. Guz and M. G. Furugyan, "Computation scheduling in multiprocessor real-time automatic control systems with constrained processor memory," Autom. Remote Control **66**, 295 (2005).
6. B. V. Grechuk and M. G. Furugyan, "Optimal preemptive scheduling in multiprocessor systems with incomplete communication graph," Cybern. Syst. Anal. **41**, 397–402 (2005).
7. E. O. Kosorukov and M. G. Furugyan, "Some algorithms for resource allocation in multiprocessor systems," Moscow Univ. Comput. Math. Cybernet. **33**, 202 (2009).
8. M. G. Furugyan, "Computation planning in multiprocessor real time automated control systems with an additional resource," Autom. Remote Control **76**, 487 (2015).
9. M. G. Furugyan, "Making schedules in multiprocessing systems with several additional resources," J. Comput. Syst. Sci. Int. **56**, 227 (2017).
10. M. G. Furugyan, "Computation scheduling in multiprocessor systems with several types of additional resources and arbitrary processors," Moscow Univ. Comput. Math. Cybernet. **41**, 145 (2017).
11. A. A. Mironov and V. I. Tsurkov, "Hereditarily minimax matrices in models of transportation type," J. Comput. Syst. Sci. Int. **37**, 927 (1998).
12. A. A. Mironov and V. I. Tsurkov, "Minimax under nonlinear transportation constraints," Dokl. Math. **64**, 351 (2001).
13. A. A. Mironov and V. I. Tsurkov, "Transportation problems with minimax criteria," Dokl. Math. **53**, 119 (1996).
14. A. A. Mironov and V. I. Tsurkov, "Minimax in transportation models with integral constraints: I," J. Comput. Syst. Sci. Int. **42**, 562 (2003).
15. A. A. Mironov, V. V. Fedorchuk, and V. I. Tsurkov, "Minimax in transportation models with integral constraints: II," J. Comput. Syst. Sci. Int. **44**, 732 (2005).
16. A. A. Mironov and V. I. Tsurkov, "Network models with fixed parameters at the communication nodes. 1," J. Comput. Syst. Sci. Int. **32** (6), 1–11 (1994).
17. A. A. Mironov and V. I. Tsurkov, "Network models with fixed parameters at the communication nodes. 2," J. Comput. Syst. Sci. Int. **33** (3), 107–116 (1995).
18. Yu. E. Malashenko and N. M. Novikova, "Analysis of multiuser network systems taking into account uncertainty," J. Comput. Syst. Sci. Int. **37**, 292 (1998).
19. Yu. E. Malashenko and N. M. Novikova, "The analysis of multiuser network systems under uncertainty: 7. The problem of standard vulnerability analysis of a multicommodity flow network," J. Comput. Syst. Sci. Int. **38**, 589 (1999).
20. Th. H. Cormen, Ch. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms* (The MIT Press, Cambridge, MA, 2009).
21. B. Korte and J. Vygen, *Combinatorial Optimization. Theory and Algorithms* (Springer, Berlin, Heidelberg, 2006).
22. D. T. Fillips and A. Garcia-Diaz, *Fundamentals of Network Analysis* (Prentice-Hall, Englewood Cliffs, NJ, 1981).

Translated by M. Talacheva