ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

DÉPARTEMENT DE MATHÉMATIQUES
Chaire de Recherche Opérationnelle
CH-1015 LAUSANNE

e-mail: wkubiak@mun.ca

Preemptive open shop scheduling with multiprocessors: polynomial cases and applications

Dominique de Werra, Tamás Kis, Wieslaw Kubiak

ORWP 04/06
May 2004

Preemptive open shop scheduling with multiprocessors: polynomial cases and applications

Dominique de Werra, Tamás Kis, Wieslaw Kubiak
Ecole Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland,
e-mail: dewerra.ima@epfl.ch
Computer and Automation Institute, 1111 Budapest, Kende str. 13-17, Hungary
e-mail: tamas.kis@sztaki.hu
Faculty of Business Administration, Memorial University of Newfoundland,
St. John's, NL, Canada, e-mail: wkubiak@mun.ca

Abstract

This paper addresses a multiprocessor generalization of the preemptive open-shop scheduling problem. The set of processors is partitioned into two groups and the operations of the jobs may require either single processors in either group or simultaneously all processors from the same group. We consider two variants depending on whether preemptions are allowed at any fractional time point or only at integral time points. We shall show that the former problem can be solved in polynomial time, and provide sufficient conditions under which the latter problem is tractable. Applications to course scheduling and hypergraph edge coloring are also discussed.

Keywords: preemptive open shop scheduling, multiprocessor operations, polynomial time algorithms

1 Introduction

The open shop scheduling model has been used for various types of scheduling problems including in particular course scheduling (or school timetabling). Whereas the basic open shop model is able to handle only the simplest instances of timetabling problems, some extensions are able to capture more intricate scenarios with additional requirements.

In this paper we present an extension of the open shop model which comprises multiprocessors or groups of processors. Such a model finds applications in processor scheduling and also when dealing with some university timetabling problems, where classes (i.e., groups of students) are grouped for some special lectures given to several classes simultaneously.

We shall briefly discuss complexity results related to this extension and describe situations where the optimal schedule with respect to the makespan

objective can be determined in polynomial time. We refer the reader to Berge [3] for all graph theoretical terms not defined here and to Błażewicz et al. [6] and to Garey and Johnson [12] for basic definitions of scheduling theory as well as for fundamental concepts and results of computational complexity.

2 Open shop scheduling with multiprocessors

Let us first describe the basic open shop scheduling problem which will be generalized. A collection $\mathcal{P} = \{P_1, \ldots, P_m\}$ of processors is given along with a set $\mathcal{J} = \{J_1, \ldots, J_n\}$ of jobs. A job J_j consists of at most m operations $O_{j1}, \ldots, O_{jm}$, where O_{jh} requires processor P_h for processing. The *processing time* of O_{jh} is a non-negative integer number b_{jh}; if $b_{jh} = 0$ then J_j is not processed on processor P_h. It is assumed that each processor can work on at most one operation at a time and similarly no two operations of the same job can be processed simultaneously.

Now we generalize this model by partitioning the set $\mathcal{P}$ into groups $\mathcal{G}_1, \ldots, \mathcal{G}_p$ of processors, called *multiprocessors*. In addition to the operations O_{jh} involving only one processor P_h, (these will be called *individual operations*), there are *group operations* $\hat{O}_{j1}, \ldots, \hat{O}_{jp}$; $\hat{O}_{j\ell}$ must be processed on all processors in group $\mathcal{G}_\ell$ simultaneously. The processing time of $\hat{O}_{j\ell}$ will be denoted by $a_{j\ell}$; if $a_{j\ell} = 0$ then J_j has no group operation on $\mathcal{G}_\ell$.

We confine our discussion to the preemptive model, i.e., the processing of operations can be interrupted and resumed at a later moment in time. When an operation of a job is interrupted then another operation of the same job may be processed before the same operation is resumed. We will distinguish between two models: (1) the *fractional model*, when preemption may occur at fractional time points and (2) *the integral model*, when preemption may occur only at integral time points. In either case a *schedule* specifies for each operation a set of time intervals where the operation must be processed contiguously. A schedule is *fractional* (*integral*) if preemption of processing occurs at fractional (only at integral) time points.

Remark 1 This definition allows schedules of length non-polynomial in the length of the instance. Therefore, a polynomial algorithm with respect to a particular objective function may exist only if there always exists an optimal schedule of polynomially bounded length.

The completion time of a job in a schedule is the least time point by which all operations of the jobs are fully processed. In this paper we will consider only the minimization of the maximum job completion time or *makespan* under the assumption that there are only $p = 2$ multiprocessors.

If there were no group operations, then both the fractional and the integral problems could be modeled by a bipartite multigraph $G = (\mathcal{J} \cup \mathcal{P}, E, b)$,

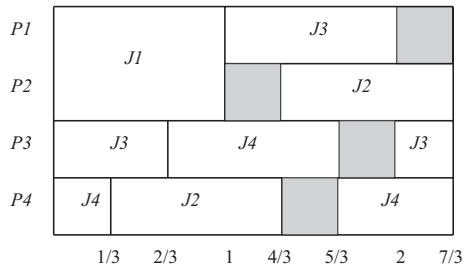

Figure 1: A fractional optimal schedule.

such that (J_j, P_h) is an edge in E with multiplicity b_{jh} if and only if $b_{jh} > 0$. According to *Kőnig's edge coloring theorem*, the edges of a bipartite multigraph G can be colored with K colors if and only if $K \geq \Delta(G)$ where $\Delta(G)$ is the maximum degree (with edge multiplicities) of the nodes of G (see Berge [3]). Edge colorings of bipartite multigraphs can be constructed in polynomial time by using multicolors (matchings with multiplicities), for details see Gabow and Kariv [11], Gonzalez and Sahni [13]. In this case there is no advantage of allowing preemptions at fractional time points.

In contrast, when there are group operations and preemptions are allowed at fractional time points, the length of an optimal schedule need not be integral even if all problem data are integral. This issue is illustrated in Figure 1. There are four processors partitioned into two groups $\mathcal{G}_1 = \{P_1, P_2\}$, $\mathcal{G}_2 = \{P_3, P_4\}$. Job 1 has only one group operation on $\mathcal{G}_1$, whereas jobs J_2, J_3 and J_4 have two individual operations each. All operations have unit length. As can be seen, the minimum makespan is 7/3. On the other hand, in the integral model the minimum makespan is 3.

After a brief literature review of known results (Section 3), we define additional notation and establish common properties of the two models (Section 4). Then we discuss the fractional model in Section 5, and establish sufficient conditions under which the integral version of the problem can be solved in polynomial time (Section 6, 7). Applications to timetabling and coloring the edges of hypergraphs that generalize bipartite multigraphs conclude the paper (Section 8).

3 Literature review

Open shop scheduling with multiprocessor operations (or tasks) is a novel issue in scheduling theory; polynomial time algorithms for problems with two or three stages or with a fixed number of stages and unit time operations can be found in (Brucker and Krämer [7], and Brucker and Krämer [8]). There are several results on scheduling multiprocessor tasks in a parallel processor environment. When the tasks are not assigned to processors in

advance (not dedicated), the complexity of the problem with and without preemption has been analized by e.g. Błażewicz et al. [4], Du and Leung [10]. In contrast, in the *dedicated model* each task requires the simultaneous use of a pre-specified set of processors. This model has been studied by e.g., Błażewicz et al. [5], Hoogeveen et al. [14], Kubale [16] and by Jansen and Porkolab [15], see also the survey of Drozdowski [9]. It is instructive to note that under the assumption that all tasks require two processors simultaneously and preemption is allowed, then minimizing the makespan is NP-hard when preemption of execution may occur only at integral time points [16], and it is polymonially solvable if preemption may occur at any fractional time point [15].

The preemptive integral open shop model has been used in (de Werra [18]) for handling school timetabling problems: A collection $\mathcal{J} = \{J_1, \ldots, J_n\}$ of teachers is given along with a collection $\mathcal{P} = \{P_1, \ldots, P_m\}$ of classes; each class is a group of students that follow exactly the same program. A collection E of lectures O_{jh} is given; lecture O_{jh} is given by teacher J_j to the class P_h. It may be the case that the same lecture is given several times. All lectures are assumed to have a unit duration.

In (Asratian and de Werra [2]) a generalization of the above timetabling model, the *university timetabling* problem (UTP), has been formulated. In addition to the above data, a partition of the set $\mathcal{P}$ of classes into groups $\mathcal{G}_1, \ldots, \mathcal{G}_p$ is given; furthermore there is a collection $\mathcal{E}$ of lectures, called *group lectures*, each of which is given by a teacher J_j to an entire group $\mathcal{G}_l$ of classes. Such a situation occurs in various educational systems and often there is a rather small number p of groups, i.e., typically $p \leq 4$ (see [2] for a discussion).

Concerning the computational complexity of the multiprocessor problems studied in this paper, Asratian and de Werra [2] have shown that even if there are only $p = 3$ multiprocessors, the makespan minimization problem in the integral model is unary NP-hard.

An even stronger result was established in (de Werra, Asratian and Durand [19]) based on a special case occurring in timetabling. We give here its formulation in terms of open shop: A job J_j will be called *homogenous* if its operations are all group operations or all individual operations. It turns out that minimizing the makespan in the integral model with $p = 3$ multiprocessors and all jobs homogenous is already unary NP-hard.

4 Preliminaries

The input of both the fractional and the integral problems consists of two matrices A and B, where $A = (a_{j\ell})$ is a $(n \times p)$-matrix and $B = (b_{jh})$ is a $(n \times m)$-matrix. The entries of both matrices are non-negative integral numbers.

An algorithm for solving the makespan minimization problem is *polynomial* if its running time is polynomial in the input length, i.e., in the size of the matrices A and B. It is *strongly polynomial* if its running time is polynomial in the dimension of these matrices, i.e., in n, m and p.

Let $\Delta(\mathcal{G}_\ell) = \sum_j a_{j\ell}$, $\ell = 1, 2$; $d(J_j) = \sum_h b_{jh}$, $j = 1, \ldots, n$; and $d(P_h) = \sum_j b_{jh}$, $h = 1, \ldots, m$.

A feasible schedule (one respecting all constraints) can be divided into four parts:

(a) only group operations on $\mathcal{G}_1$ and only individual operations on the processors in $\mathcal{G}_2$,

(b) only group operations on both processor groups,

(c) only individual operations on processors in $\mathcal{G}_1$ and only group operations on $\mathcal{G}_2$, and

(d) only individual operations on all processors.

If r denotes the length of part (b), and w the length of part (d), then the schedule can be rearranged so that the first $\Delta(\mathcal{G}_1) - r$ time units belong to part (a), the next r time units to part (b), the interval $[\Delta(\mathcal{G}_1), \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r]$ constitutes part (c), and finally part (d) is processed in $[\Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r + w]$. It immediately follows that the makespan of the schedule is a linear function of w and r:

$$T = \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r + w. \tag{1}$$

5 The fractional model

In order to solve the fractional problem, we propose a linear program with the following decision variables:

- r: the length of part (b),
- w: the length of part (d),
- $x_{j\ell}$: the amount of time group operation $\hat{O}_{j\ell}$ is processed in part (b),
- y_{jh}: the amount of time individual operation O_{jh} is processed in part (d).

We will show that the optimal solutions of the following linear program LP (2) represent optimal fractional schedules with minimum makespan.

$$\min\ (w-r) \qquad \text{s.t.} \tag{2a}$$

$$\sum_j b_{jh} - (\Delta(\mathcal{G}_2) - r) \leqslant \sum_j y_{jh} \leqslant w, \quad \forall h \in \mathcal{G}_1 \tag{2b}$$

$$\sum_j b_{jh} - (\Delta(\mathcal{G}_1) - r) \leqslant \sum_j y_{jh} \leqslant w, \quad \forall h \in \mathcal{G}_2 \tag{2c}$$

$$\sum_h y_{jh} \leqslant w, \quad \forall j \tag{2d}$$

$$0 \leqslant y_{jh} \leqslant b_{jh} \tag{2e}$$

$$\sum_j x_{j1} = r \tag{2f}$$

$$\sum_j x_{j2} = r \tag{2g}$$

$$x_{j1} + x_{j2} \leqslant r, \quad \forall j \tag{2h}$$

$$0 \leqslant x_{j\ell} \leqslant a_{j\ell} \tag{2i}$$

$$\sum_{h \in \mathcal{G}_1} (b_{jh} - y_{jh}) + a_{j2} - x_{j2} \leqslant \Delta(\mathcal{G}_2) - r, \quad \forall j \tag{2j}$$

$$\sum_{h \in \mathcal{G}_2} (b_{jh} - y_{jh}) + a_{j1} - x_{j1} \leqslant \Delta(\mathcal{G}_1) - r, \quad \forall j \tag{2k}$$

The objective function (2a) prescribes minimization of $w - r$, which is the only variable term in (1). Inequalities (2b)-(2e) ensure, on the one hand, that part (d) of the schedule can be done in w time units and, on the other hand, that the total load of any processor working on individual operations in part (a) and (c) does not exceed $\Delta(\mathcal{G}_1) - r$ and $\Delta(\mathcal{G}_2) - r$ time units, respectively. In particular, the right hand side inequality in (2b) guarantees that the total load of processor P_h, $h \in \mathcal{G}_1$ does not exceed w in part (d), whereas the left hand side inequality in (2b) ensures that the remaining load of P_h left for part (c) will not exceed $\Delta(\mathcal{G}_2) - r$. Similarly, the right hand side inequality in (2c) ensures that the total load of processor P_h, $h \in \mathcal{G}_2$ does not exceed w in part (d), whereas the left hand side inequality in (2c) ensures that the remaining load of P_h left for (a) will not exceed $\Delta(\mathcal{G}_1) - r$. The constraint (2d) ensures that each job j is processed no longer than w in part (d). Finally, (2e) ensures that individual operation O_{jh} is not processed longer than its processing time b_{jh} in part (d).

Inequalities (2f)-(2i) guarantee that the selected portions of the group operations can be done in part (b). In particular, constraints (2f) and (2g) ensure the total load of multiprocessors $\mathcal{G}_1$ and $\mathcal{G}_2$ in part (b) equal r. Constraints (2h) make sure that group operations of any job are processed no

longer than r time unit in part (b). Finally, (2i) ensures that no group operation $\hat{O}_{j\ell}$ is processed longer than its processing time $a_{j\ell}$.

The last two constraints require that the total work done on a job in part (a) or (c) of the schedule be not more than the length of the corresponding period. Notice that their left hand sides are non-negative numbers by (2e) and (2i).

It is not hard to see that any feasible fractional schedule must satisfy all of the above constraints. Moreover, we have the following:

Lemma 1 *Any feasible solution of LP (2) can be transformed into a feasible (fractional) preemptive schedule with makespan (1) in strongly polynomial time. Moreover, if the solution is integral, then the resulting preemptive schedule is also integral.*

Proof First notice that the four parts of the schedule can be treated independently, that is, it suffices to construct a feasible schedule for each part separately, then joining the parts in any order gives the desired result.

To construct a feasible schedule for any part, observe that when considering the parts separately, we face the problem of finding a preemptive open shop schedule which can be solved by the procedure of Gonzalez and Sahni in strongly polynomial time, cf. Lemma 3.8 in [13]. Namely, for part (d) this observation is obvious, concerning e.g., part (a) consider the preemptive open shop problem with jobs $J_1, \ldots, J_n$ and processors P_h, $h \in \mathcal{G}_2$, and with the additional processor $\hat{\mathcal{G}}_1$ representing the group of processors $\mathcal{G}_1$. The processing time of J_j on P_h is $b_{jh} - y_{jh}$, whereas its processing time on $\hat{\mathcal{G}}_1$ is $a_{j1} - x_{j1}$. A similar construction applies for part (c), while for part (b) the preemptive open shop scheduling problem consists of the jobs J_j to be performed on two processors, $\hat{\mathcal{G}}_1$ and $\hat{\mathcal{G}}_2$, with processing times $x_{j\ell}$.

The second part of the theorem is a simple consequence of the procedure of Gonzalez and Sahni [13]. □

Corollary 1 *The fractional model can be solved in strongly polynomial time.*

Proof As we have already noted, any feasible schedule must satisfy the constraints of LP (2). Consequently, by Lemma 1, an optimal solution of LP (2) yields an optimal solution of the scheduling problem.

Since all entries in the constraint matrix of LP (2) are 0, +1 or −1, the linear program and thus the makespan minimization problem in the fractional model can be solved in strongly polynomial time by the method of Tardos [17]. □

Remark 2 An open question remains whether LP (2) can be solved by a direct combinatorial method.

6 The integral model

In this section we provide sufficient conditions that enable us to solve the integral problem in polynomial time. To this end, we restrict the set of jobs to binary jobs that we define below.

Firstly, we distinguish between four types of operations: $(g,1)$, $(g,2)$, $(i,1)$ and $(i,2)$. A group operation is of type (g,ℓ) if it requires the multiprocessor $\mathcal{G}_\ell$. An individual operation is of type (i,ℓ) if it requires a single processor from the group $\mathcal{G}_\ell$. We say that a job is *binary* if its operations are of at most two different types. Thus a binary job cannot have e.g., a group operation and individual operations on processors in each of the processor groups.

We can further partition the set of binary jobs as $\mathcal{J}^{gi} \cup \mathcal{J}^{ig} \cup \mathcal{J}^{gg} \cup \mathcal{J}^{ii}$, where $\mathcal{J}^{gi}$ consists of all jobs with operations of type $(g,1)$, and $(i,1)$ or $(i,2)$, $\mathcal{J}^{ig}$ contains all jobs with operations of type $(g,2)$, and $(i,1)$ or $(i,2)$, all jobs in $\mathcal{J}^{gg}$ have only group operations, and all jobs in $\mathcal{J}^{ii}$ have only individual operations.

From now on we always assume that all jobs are binary. Under this assumption, by using the results of Section 5, we shall show that $\Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) + \lceil w^* - r^* \rceil$ is the minimum makespan for the integral model, where w^* and r^* are taken from any optimal solution to LP (2).

First, notice that for fixed values w and r, the x and y that satisfy all constraints of LP (2) constitute a compatible flow in a capacitated network. To see this, observe that for binary jobs the LP (2) can be equivalently re-written in the following form, which we refer to as LP (3).

$$\min\ (w - r) \qquad \text{s.t.} \tag{3a}$$

$$\sum_{j \in \mathcal{J} \setminus \mathcal{J}^{gg}} b_{jh} - (\Delta(\mathcal{G}_2) - r) \leqslant \sum_{j \in \mathcal{J} \setminus \mathcal{J}^{gg}} y_{jh} \leqslant w, \quad \forall h \in \mathcal{G}_1 \tag{3b}$$

$$\sum_{j \in \mathcal{J} \setminus \mathcal{J}^{gg}} b_{jh} - (\Delta(\mathcal{G}_1) - r) \leqslant \sum_{j \in \mathcal{J} \setminus \mathcal{J}^{gg}} y_{jh} \leqslant w, \quad \forall h \in \mathcal{G}_2 \tag{3c}$$

$$\sum_{h \in \mathcal{G}_1 \cup \mathcal{G}_2} y_{jh} \leqslant w, \quad \forall j \in \mathcal{J}^{ii} \tag{3d}$$

$$\sum_{h \in \mathcal{G}_1} y_{jh} \leqslant w, \quad \forall j \in \mathcal{J}^{gi} \cup \mathcal{J}^{ig} \tag{3e}$$

$$\sum_{h \in \mathcal{G}_2} y_{jh} \leqslant w, \quad \forall j \in \mathcal{J}^{gi} \cup \mathcal{J}^{ig} \tag{3f}$$

$$0 \leqslant y_{jh} \leqslant b_{jh} \tag{3g}$$

$$\sum_{j \in \mathcal{J}^{gg} \cup \mathcal{J}^{gi}} x_{j1} = r \tag{3h}$$

$$\sum_{j \in \mathcal{J}^{gg} \cup \mathcal{J}^{ig}} x_{j2} = r \tag{3i}$$

$$x_{j1} + x_{j2} \leqslant r, \quad \forall j \in \mathcal{J}^{gg} \tag{3j}$$

$$0 \leqslant x_{j\ell} \leqslant a_{j\ell} \tag{3k}$$

$$\sum_{h \in \mathcal{G}_1} b_{jh} + a_{j2} - (\Delta(\mathcal{G}_2) - r) \leqslant \sum_{h \in \mathcal{G}_1} y_{jh} + x_{j2}, \quad \forall j \in \mathcal{J}^{ig} \tag{3l}$$

$$\sum_{h \in \mathcal{G}_2} b_{jh} + a_{j1} - (\Delta(\mathcal{G}_1) - r) \leqslant \sum_{h \in \mathcal{G}_2} y_{jh} + x_{j1}, \quad \forall j \in \mathcal{J}^{gi} \tag{3m}$$

When w and r are fixed, LP (3) is equivalent to a capacitated network flow problem, denoted by $N(w, r)$.

We are now ready to formulate a polynomial time algorithm for the makespan minimization problem in the integral model. The algorithm finds an optimal solution in the following three steps:

1. Solve the LP (3). Let (w^*, r^*, x^*, y^*) be an optimal solution.

2. Modify LP (3) as follows: replace the objective function (3a) by $\min r$, and add the constraint

$$w - r = \lceil w^* - r^* \rceil \tag{4}$$

   to the existing constraints. Call the resulting system LP'.

3. Solve LP', let (w, r, x, y) be an optimal solution. Find a compatible flow in $N(w, r)$ with w and r fixed to the values of this optimal solution. Convert the solution into an open shop schedule.

Clearly, the above algorithm runs in strongly polynomial time.

To prove that it finds an optimal solution to the makespan minimization problem with integral preemptions, it suffices to show that the r found in step 3 of the algorithm is integral. Since this implies integrality of w, it follows that all bounds in $N(w, r)$ are integral. Therefore, $N(w, r)$ admits an integral compatible flow, which, by Lemma 1, can readily be converted into a solution for the preemptive open shop scheduling problem.

Before proving the next theorem recall that a square matrix M is *doubly stochastic* if all entries are non-negative and each row and column sums up to 1. Moreover, a square matrix Π is a *permutation matrix* if it is an integral doubly stochastic matrix. By the Birkhoff-von Neumann theorem every doubly stochastic matrix is a convex combination of permutation matrices, that is, there exist permutation matrices $\Pi^1, \ldots, \Pi^q$ and positive real numbers $\lambda_1, \ldots, \lambda_q$ such that $M = \sum_{i=1}^{q} \lambda_i \Pi^i$ and $\sum_i \lambda_i = 1$.

Theorem 1 *If all jobs are binary, r is integral in any optimal solution to LP'.*

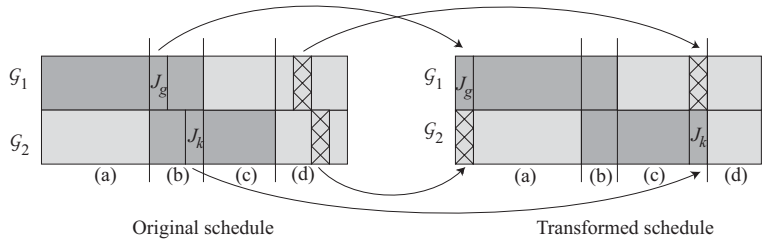

Figure 2: The transformation of the solution. In the transformed schedule parts (b) and (d) become shorter.

Proof Let $S = (w, r, x, y)$ be an optimal solution to LP' and suppose r is fractional. Due to constraints (3h)-(3j), there can be at most two jobs with $x_{j1} + x_{j2} = r$. For these jobs, if they exist, either x_{j1} or x_{j2} is fractional, as r is fractional by assumption. Therefore, we can choose jobs J_g and J_k with multiprocessor operations on $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively, such that x_{g1} and x_{k2} are fractional and $x_{j1} + x_{j2} < r$ for all other $j \in \mathcal{J} \setminus \{g, k\}$. Notice that we may have $g = k$. Let ε be the minimum of $r - \lfloor r \rfloor$, x_{g1}, x_{k2}, and $\min_{j \in \mathcal{J} \setminus \{g,k\}}(r - (x_{j1} + x_{j2}))$. By the choice of g and k, the latter minimum, and thus, ε are positive.

For job J_g, we claim that at least one of the following inequalities holds: $a_{g1} - x_{g1} + \sum_{h \in \mathcal{G}_2}(b_{gh} - y_{gh}) < \Delta(\mathcal{G}_1) - r$ or $\sum_{h \in \mathcal{G}_2} y_{gh} < w$. If not, then $a_{g1} - x_{g1} + \sum_{h \in \mathcal{G}_2} b_{gh} = \Delta(\mathcal{G}_1) - r + w$ would follow. But the right hand side of this equality is integral, hence, x_{g1} is integral as well, a contradiction. Let $\delta_g = \Delta(\mathcal{G}_1) - r - (a_{g1} - x_{g1} + \sum_{h \in \mathcal{G}_2}(b_{gh} - y_{gh}))$. By the same token, for job J_k at least one of the following holds: $a_{k2} - x_{k2} + \sum_{h \in \mathcal{G}_1}(b_{kh} - y_{kh}) < \Delta(\mathcal{G}_2) - r$ or $\sum_{h \in \mathcal{G}_1} y_{kh} < w$. Let $\delta_k = \Delta(\mathcal{G}_2) - r - (a_{k2} - x_{k2} + \sum_{h \in \mathcal{G}_1}(b_{kh} - y_{kh}))$.

We shall determine a set of job-processor pairs $A \subseteq \mathcal{J} \times \mathcal{P}$ such that the solution $S' = (w', r', x', y')$ for LP' obtained from S by decreasing each of w, r, x_{g1}, x_{k2} and y_{jh} ($(j, h) \in A$) by a small amount $\mu > 0$ (to be chosen later) is feasible to LP'. Since $r' < r$, this will lead to a contradiction. The effect of this transformation on the schedule is illustrated in Figure 2.

We now provide a set of conditions for A to ensure feasibility of S' for LP'. First, call a job J_{j_0} (processor P_{h_0}) *tight* if $\sum_h y_{j_0 h} = w$ ($\sum_j y_{j h_0} = w$) in S. For each tight processor P_{h_0}, A will contain a pair (j, h_0), ensuring the right hand side inequalities (3b) and (3c) for S'. To satisfy the left hand side inequalities in (3b) and (3c), A will not contain distinct pairs $(j_1, h), (j_2, h)$ for any h. To meet (3d), for each tight job J_{j_0}, A will contain a pair (j_0, h) for some h, and to ensure that (3l) and (3m) hold for S', A will not contain distinct pairs $(j, h_1), (j, h_2)$ with $h_1, h_2 \in \mathcal{G}_\ell$. By the same token, if $\delta_g = 0$ ($\delta_k = 0$), then A will not assign job J_g to a processor in $\mathcal{G}_2$ (job J_k to a processor in $\mathcal{G}_1$). We will show that a set of pairs A satisfying all of the above conditions always exists. We distinguish between two cases:

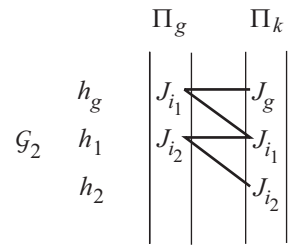

Figure 3: The combination of Π^g and Π^k.

Case 1. If $\delta_g > 0$ or $\delta_k > 0$, define the matrix $M = \begin{pmatrix} C & Y \\ Y^T & D \end{pmatrix}$ for S, where Y is a $n \times m$ matrix with $Y_{jh} = y_{jh}$, Y^T is its transpose, C and D are diagonal matrices complementing each row and column sum of M to w. Since $w > 0$, $(1/w)M$ is a doubly stochastic matrix, therefore, by the Birkhoff-von Neumann theorem, $(1/w)M$ is a convex combination of permutation matrices.

Consider a particular convex decomposition of $(1/w)M$ and choose a permutation matrix Π in it according to the following criteria. If $\delta_g > 0$ and $\delta_k > 0$, then choose Π arbitrarily and let $\mu := \min\{\varepsilon, w \cdot \lambda_\Pi, \delta_g, \delta_k\}$, where λ_Π is the coefficient of Π in the convex decomposition. If $\delta_g = 0$, then choose Π with $\Pi_{gg} = 1$ (such a Π exists in any convex decomposition of $(1/w)M$, because $\sum_{h \in \mathcal{G}_2} y_{gh} < w$ by assumption, hence, $C_{gg} > 0$), and let $\mu := \min\{\varepsilon, w \cdot \lambda_\Pi, \delta_k\}$. Finally, if $\delta_k = 0$, choose Π with $\Pi_{kk} = 1$ (again, such a Π exists as $C_{kk} > 0$) and let $\mu := \min\{\varepsilon, w \cdot \lambda_\Pi, \delta_g\}$. One may verify that the set of pairs $A = \{(j,h) \in \mathcal{J} \times \mathcal{P} \mid \Pi_{j,n+h} = 1\}$ satisfies all requirements.

Case 2. If $\delta_g = \delta_k = 0$, then we cannot always apply directly the above proof technique to determine A. Since $\delta_g = 0$ we know that any convex decomposition of $(1/w)M$ contains a permutation matrix Π^g with $\Pi^g_{gg} = 1$, and since $\delta_k = 0$, it also contains a Π^k with $\Pi^k_{kk} = 1$. If $\Pi^g_{kk} = 1$ or job J_k has no individual operations on any processor in $\mathcal{G}_1$, then define A as in Case 1 with respect to $\Pi = \Pi^g$. Similarly, if $\Pi^k_{gg} = 1$ or J_g has no individual operations on any processor in $\mathcal{G}_2$, then define A is in Case 1 with respect to $\Pi = \Pi^k$. If none of the above applies, we shall combine Π^g and Π^k to a new 0/1 matrix Π, where Π will not be a permutation matrix, and define A with respect to Π as in Case 1. Initially, let $\Pi = \Pi^k$. Since $\Pi^k_{gg} = 0$ by assumption, there exists a processor $h_g \in \mathcal{G}_2$ such that $\Pi_{g,n+h_g} = 1$. Set $\Pi_{g,n+h_g}$ to 0. Now, if processor h_g is not tight, then we can stop, the set A defined with respect to Π would meet all requirements. However, if processor h_g is tight, there must be some job, different to J_g, assigned to it when defining A. To choose this job, observe that there exists some

job J_{i_1} such that $\Pi^g_{i_1,n+h_g} = 1$, since processor h_g is tight. Therefore, set $\Pi_{i_1,n+h_g}$ to 1. Now if $\Pi_{i_1,n+h} = 0$ for all $h \in \mathcal{G}_2 \setminus \{h_g\}$, then we can stop. Otherwise, let h_1 be the processor in $\mathcal{G}_2 \setminus \{h_g\}$ such that $\Pi_{i_1,n+h_1} = 1$. Set $\Pi_{i_1,n+h_1}$ to 0 and continue the procedure with processor h_1 as above (see Fig. 3 for illustration). Notice that J_k cannot be part of the above chain of jobs and processors, as J_k is binary and it has individual operations on processors in $\mathcal{G}_1$, by assumption. Since the number of jobs and processors is finite and no cycling can occur, the algorithm will terminate. In the end, the set A defined with respect to Π in the same way as in Case 1, and $\mu = \min\{\varepsilon, w \cdot \lambda_{\Pi^g}, w \cdot \lambda_{\Pi^k}\}$ satisfy all requirements. □

7 A special case

A binary job J_j will be called *simple* if either it has no individual operations or it has no group operations or all its individual and group operations are on the same group of processors.

Simple jobs allow us to considerably simplify our three step algorithm of Section 6. Namely, suppose we want to find a schedule with makespan T in the integral model, if one exists. Then in any feasible schedule of length T, r is at least $r_{\min}(T) := \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - T\}$, as one may verify.

Lemma 2 *If an instance with all simple jobs admits an integral preemptive schedule of length T, then it also admits one of length T and with $r = r_{\min}(T)$.*

Proof Let $\mathcal{S}$ be an integral preemptive schedule with makespan T and overlap $r > r_{\min}(T) = \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - T\}$ in part (b) of $\mathcal{S}$. Consider the block X, see Figure 4, of length $r - r_{\min}(T)$ starting at $\Delta(\mathcal{G}_1) - r$ on multiprocessor $\mathcal{G}_2$. Let Y be the block of individual operations on processors in $\mathcal{G}_2$ in part (d) of length $r - r_{\min}(T) \leqslant w$ starting at time $\Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r$, see Figure 4.

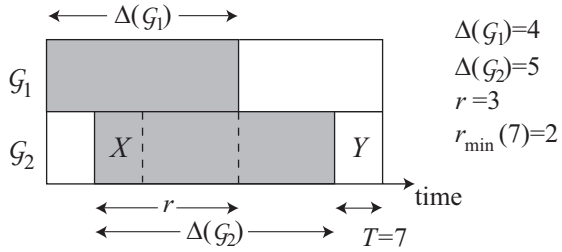

Figure 4: A schedule with non-minimal overlap.

We may then swap the blocks X and Y. The resulting schedule $\mathcal{S}'$ is feasible: since the jobs are simple by assumption, the jobs with their

group operation on $\mathcal{G}_2$ have no individual operations on processors in $\mathcal{G}_1$, moreover, the jobs with individual operations on processors in $\mathcal{G}_2$ have no group operations on $\mathcal{G}_1$. Clearly, the size of $\mathcal{S}'$ remains T. Finally, the size of part (b) in $\mathcal{S}'$ is precisely $r_{\min}(T)$. □

Corollary 2 *An open shop scheduling problem with two multiprocessors and all simple jobs has an optimal schedule with a minimum overlap of group operations.*

By the above corollary, there is a purely combinatorial algorithm for solving the integral model with all simple jobs. Namely, find the smallest T such that the network flow $N(w, r)$ with $r = \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - T\}$ and $w = T - (\Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r)$ admits a compatible flow. Such a T can be determined by dichotomic search between e.g., the trivial lower and upper bounds 0 and $\sum_{jh} b_{jh} + \sum_{j\ell} a_{j\ell}$, respectively. This procedure is polynomial, but not strongly polynomial in the input.

Another option is to compute the optimal makespan in strongly polynomial time by solving LP (2). If (w^*, r^*, x^*, y^*) is the optimal solution, then the optimal makespan is $T^* = \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) + \lceil w^* - r^* \rceil$. Letting $r_{\min}(T^*) = \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - T^*\} = \max\{0, -\lceil w^* - r^* \rceil\}$, solve the network flow $N(w, r)$ with $r = r_{\min}(T^*)$ and $w = r_{\min}(T^*) + \lceil w^* - r^* \rceil$. Notice that if $\lceil w^* - r^* \rceil < 0$, then the above calculation shows that there exists an optimal solution with $w = 0$, whereas if $\lceil w^* - r^* \rceil \geqslant 0$, then there is an optimal solution with $r = 0$.

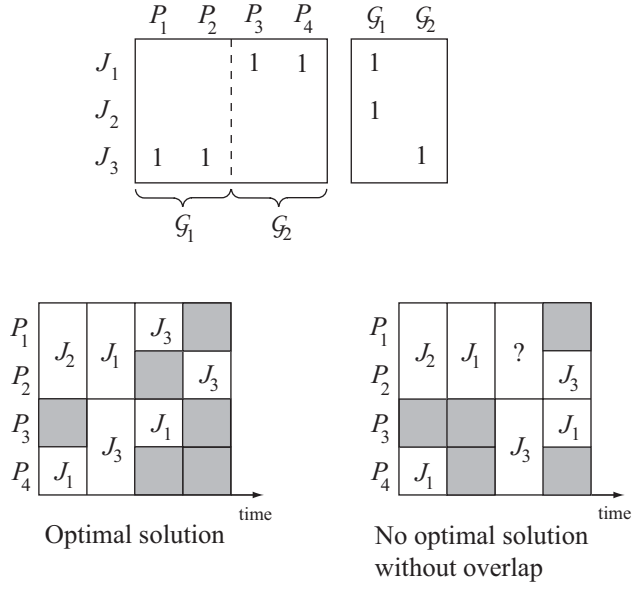

Figure 5: No optimal schedule with $r = r_{\min}(4)$.

Notice that Lemma 2 does not hold for binary jobs in general as the example in Figure 5 indicates. There, jobs J_1, J_3 are binary but not simple, and J_2 is simple. The optimum makespan is 4, thus $r_{\min}(4) = \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - 4\} = 0$, but there exists no feasible schedule with makespan 4 and $r = 0$.

As for the number of preemptions, observe that we may rearrange arbitrarily the schedule of the group operations on $\mathcal{G}_1$ in part (a) without changing the size of any part and without affecting feasibility. Moreover, the same is true for part (b) between $\Delta(\mathcal{G}_1) - r_{\min}(T)$ and $\Delta(\mathcal{G}_1)$. We may in particular reorder the group operations on $\mathcal{G}_1$ between 0 and $\Delta(\mathcal{G}_1)$ so that no group operation on $\mathcal{G}_1$ is preempted. Similarly, we may reorder the group operations on $\mathcal{G}_2$ between $\Delta(\mathcal{G}_1)$ and $\Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - r_{\min}(T)$ so that no group operation is preempted. Hence, we have shown the following result:

Lemma 3 *Given any instance of the makespan minimization problem in the integral model with 2 multiprocessors and all simple jobs, there exists an optimal schedule $\mathcal{S}$ with makespan T^* with the following properties:*

a) the overlap r of group operations attains the minimum:

$$r = \max\{0, \Delta(\mathcal{G}_1) + \Delta(\mathcal{G}_2) - T^*\} = r_{\min}(T^*),$$

b) there are no preemptions for the group operations on multiprocessor $\mathcal{G}_1$,

c) there are at most $r_{\min}(T^*) - 1$ preemptions for the group operations on $\mathcal{G}_2$. □

8 Applications

University timetabling. In [2] the *professor-lecturer model* has been introduced in which some type of teachers have only group-lectures (professors), and the others have only individual lectures (lecturers). This timetabling model corresponds to the preemptive open shop scheduling problem in the integral model, where professors and lecturers constitute the jobs, and classes correspond to processors. By the results of Section 7, there exists a minimum length university timetable with minimum overlap of group lectures. Moreover, when teachers are able to give two types of courses (lectures for individual classes or lectures to groups of classes), then the problem can still be solved in polynomial time (cf. Section 6).

Hypergraph edge coloring. Given an instance of the preemptive multiprocessor scheduling problem, finding an integral preemptive schedule is equivalent to determining a feasible edge coloring of the hypergraph $H = (\mathcal{P} \cup \mathcal{J}, E \cup \mathcal{E})$ with the least number of colors. The node set of H is $\mathcal{P} \cup \mathcal{J}$;

the edge set $E \cup \mathcal{E}$ consists of edges E representing the individual operations and hyperedges $\mathcal{E}$ representing the group operations. More precisely, each individual operation O_{jh} is represented by an edge $\{P_h, J_j\} \in E$ with multiplicity b_{jh} and each group operation $\hat{O}_{j\ell}$ is represented by a hyperedge $\{\mathcal{G}_\ell, J_j\} \in \mathcal{E}$ with multiplicity $a_{j\ell}$, where a *hyperedge* $\{\mathcal{G}_\ell, J_j\}$ consists of job J_j and all processors in $\mathcal{G}_\ell$. A coloring of the edges consists of a set of matchings with integral multiplicities.

These types of hypergraphs generalize bipartite multigraphs, but they do not belong to known classes, like balanced, normal or with the Kőnig-Egerváry property (see Berge [3]). Yet, our procedure enables us to decide whether an edge coloring with K colors exists in polynomial time.

Acknowledgments

The research of Tamás Kis has been supported by the Hungarian Scientific Research Fund, grant no. T046509. The research of Wieslaw Kubiak has been supported by the Natural Sciences and Engineering Research Council of Canada grant OPG0105675 and the Institute of Mathematics of the Ecole Polytechnique Fédérale de Lausanne. The supports are gratefully acknowledged.

References

[1] P.K. Ahuja, T.L. Magnanti and J.B. Orlin, Network Flows, Prentice-Hall (London 1993).

[2] A.S. Asratian and D. de Werra, A generalized Class-Teacher Model for some Timetabling Problems, Eur. J. Oper. Res. 143 (2002) 531-542.

[3] C. Berge, Graphs and Hypergraphs, North Holland (Amsterdam 1973).

[4] J. Błażewicz, M. Drabowski, J. Weglarz, Scheduling multiprocessor tasks to minimize schedule length, IEEE Trans. Comput. 35:5 (1986) 389–393.

[5] J. Błażewicz, P. Dell'Olmo, M. Drozdowski and M. G. Speranza, Scheduling multiprocessor tasks on three dedicated processors, Information Processing Letters 41 (1992) 275-280.

[6] J. Błażewicz, K.H. Ecker, E. Pesch, G. Schmidt and J. Weglarz, Scheduling computer and Manufacturing Processes, Springer-Verlag (Berlin 1996).

[7] P. Brucker and A. Krämer, Shop scheduling problems with multiprocessor tasks on dedicated processors, Annals of Op. Res. 57 (1995) 13-27.

[8] P. Brucker and A. Krämer, Polynomial algorithms for resource-constrained and multiprocessor task scheduling problems, Eur. J. Oper. Res. 90 (1996) 214-226.

[9] M. Drozdowski, Scheduling multiprocessor tasks – An overview, Eur. J. Oper. Res. 94:2 (1996) 215-230.

[10] J. Du and J.Y.-T. Leung, Complexity of scheduling parallel task systems, SIAM J. Discrete Mathematics 2:4 (1989) 473-487.

[11] H.N. Gabow and O. Kariv, Algorithms for edge coloring bipartite graphs and multigraphs, SIAM J. Computing 11:1 (1982) 117-129.

[12] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, WH Freeman & Co. (San Francisco, 1979).

[13] T. Gonzalez and S. Sahni, Open shop scheduling to minimize finish time, J. ACM 23:4 (1976) 665-679.

[14] J. A. Hoogeveen, S. L. van de Velde, B. Veltman, Complexity of scheduling multiprocessor tasks with prespecified processor allocations, Discrete Applied Mathematics, 55 (1994) 259-272.

[15] K. Jansen, L. Porkolab, Preemptive Scheduling with Dedicated Processors: Applications of Fractional Graph Coloring, Journal of Scheduling 7:1 (2004) 35-48.

[16] M. Kubale, Preemptive versus nonpreemptive scheduling of biprocessor tasks on dedicated processors, Eur. J. Oper. Res. 94:2 (1996) 242-251.

[17] É. Tardos, A strongly polynomial algorithm to solve combinatorial linear programs, Operations Research 34 (1986) 362-370.

[18] D. de Werra, An introduction to timetabling, Eur. J. Oper. Res. 19 (1985) 151-162.

[19] D. de Werra, A.S. Asratian, S. Durand, Complexity of some special types of timetabling problems, Journal of Scheduling 5 (2002) 171-183.