Available online at www.sciencedirect.com

ScienceDirect

Discrete Optimization 3 (2006) 366–381

DISCRETE OPTIMIZATION

www.elsevier.com/locate/disopt

On relocation problems with multiple identical working crews

A.V. Kononov[a], B.M.T. Lin[b,*]

[a] *Sobolev Institute of Mathematics, Novosibirsk, Russia*
[b] *Department of Information and Finance Management, Institute of Information Management, National Chiao Tung University, Hsinchu 300, Taiwan*

Received 19 August 2004; received in revised form 8 May 2006; accepted 29 June 2006
Available online 22 August 2006

Abstract

The relocation problem was formulated from a public housing project. In its basic form, a set of buildings needed to be torn down and erected by a single working crew. Given a fixed budget, the relocation problem seeks to determine a feasible reconstruction sequence of the old buildings. This problem has been shown to be mathematically equivalent to the classical two-machine flowshop of makespan minimization. In this paper, we consider a variant where multiple working crews are available for the redevelopment project. Most of our results center on the situations where all buildings require the same redevelopment time. We first present a strong NP-hardness proof for the case with two working crews. Then, we give a negative result about the approximability of the studied problem. Approximation algorithms and associated performance-ratio analysis are designed for the cases with unbounded as well as bounded numbers of machines.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Public housing project; Job scheduling; Resource constraints; Parallel machines; NP-hardness; Approximation

1. Introduction

The relocation problem was first proposed and formulated from a house redevelopment project in East Boston Area [8,12,10]. The public housing project was initiated to rehabilitate a set of old buildings. Before the redevelopment of a building, all its residents must be evacuated and accommodated in some temporary housing units. The capacity of a new building is not necessarily the same as that of the old one. If a building is demolished, say for setting up a municipal park or a parking structure, its new capacity becomes zero. On the other hand, a new building of several housing units can be developed from a parking structure where the original housing capacity is zero. Furthermore, residents might be relocated to different sites. Under a specified budget for preparing the temporary housing units, the authority needed to determine a reconstruction sequence such that all residents could be temporarily housed during the redevelopment process. In addition to the public housing issue, the relocation problem also has applications relevant to database management [1]. The financial constraint problem [11] can be also treated as a special case of the relocation problem. Or, we may simply treat the development of a building as a project to be conducted in which investments are required for all projects and each project's profit might be positive or negative.

* Corresponding author. Tel.: +886 3 5131472; fax: +886 3 5729915.
E-mail addresses: alvenko@math.nsc.ru (A.V. Kononov), bmtlin@mail.nctu.edu.tw (B.M.T. Lin).

1572-5286/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.disopt.2006.06.003

A special case of the relocation problem (RP) involves a single working crew, i.e. no two or more buildings can be simultaneously developed. The relocation problem with a single crew can also be formulated as a single-machine scheduling problem subject to resource constraints. Consider a set of n jobs $\mathcal{N} = \{J_1, J_2, \ldots, J_n\}$ to process on a single machine and a common pool of Ω_0 units of single-type resources available for the jobs. Job J_i, $1 \le i \le n$, is allowed to start its processing only if the machine is free and there are at least α_i units of resources available in the pool. The processing will consume the acquired resources and takes p_i time units. When job J_i is completed it will immediately return β_i units to the pool. No preemption is allowed. The problem is to determine a schedule such that all jobs can be successfully processed. In a feasible schedule, resources must be sufficient to support the processing of any job at any time. The relocation problem differs from conventional resource-constrained scheduling problems [2, 6] in the sense that the net contributions, $\delta_i = \beta_i - \alpha_i$, made by job J_i can be zero, negative or positive.

Kaplan and Amir [9] showed that for the single crew case the minimum amount of resources guaranteeing the existence of feasible schedules is equivalent to the sum of idle times on machine two in the classical two-machine flowshop [7] by letting α_i and β_i be the processing times of a job on the two machines of a flowshop. Therefore, the feasibility testing of the single-crew relocation problem can be solved in $O(n \log n)$ time. Note that the theme of the single-crew relocation problem centers on the feasibility only. In the original model presented by Kaplan [8], job processing times and multiple working crews (or, machines) are addressed. That is, the buildings may have different redevelopment times, and the redevelopment of several buildings can be overlapped if there are free crews and sufficient resources. In scheduling theory terms, for a particular schedule σ, let $\Omega_t(\sigma)$ be the amount of resources available at time $t \ge 0$, $s_i(\sigma)$ the starting time of job J_i, and $C_i(\sigma)$ the completion time of job J_i. We say that schedule σ satisfies the *resource* condition if

$$\Omega_t(\sigma) = \Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) \le t\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) \le t < C_i(\sigma)\}} \alpha_i \ge 0. \tag{1}$$

The objective is to find a feasible schedule whose makespan is minimum. Kaplan [8] designed a myopic algorithm to compose approximate schedules to minimize the makespan. Amir and Kaplan [1] later gave an NP-hardness proof, based upon a reduction from the Partition problem. The complexity status of the case with two identical parallel machines and unit-execution-time (UET) jobs has however remained open since it was mentioned in 1988. In this paper, we show that this case is strongly NP-hard even if all jobs make non-negative contributions, i.e. $\delta_i \ge 0, 1 \le i \le n$. Moreover, non-approximability properties and approximation algorithms for the cases with bounded and unbounded numbers of identical parallel machines will also be addressed.

The rest of this paper is organized as follows. In Section 2 we introduce a notion of mirror instance and give an auxiliary result that will be deployed throughout this paper. Relations between the relocation problem to bin packing will be included, too. Section 3 presents a basic property and a proof for the strong NP-hardness of the problem with two machines. Non-approximability is addressed in Section 4. Section 5 is dedicated to the development and analysis of several approximation algorithms. Finally, we shall give some concluding remarks in Section 6.

2. Preliminaries

Mirror instance and mirror schedule. Now we introduce a notion of mirror instance and obtain a simple but useful result about the equivalence between the relocation problems with non-negative and non-positive net contributions. The result will be deployed throughout this paper.

Given an instance I of RP and a particular schedule σ, throughout this paper $Z(\sigma)$ denotes the makespan of feasible schedule σ and OPT(I) stands for the optimal solution value of instance I. We define a *mirror* instance I_d as follows. Set $\alpha'_i = \beta_i$, $\beta'_i = \alpha_i$, $1 \le i \le n$, and $\Omega'_0 = \Omega_0 + \sum_{i=1}^{n}(\beta_i - \alpha_i)$. For schedule σ of I, we set

$$s_i(\sigma_d) = Z(\sigma) - C_i(\sigma) \quad \text{and} \quad C_i(\sigma_d) = Z(\sigma) - s_i(\sigma) \tag{2}$$

as the starting and completion times of job J_i to define schedule $\sigma_d(I_d)$ of instance I_d. In schedule $\sigma_d(I_d)$ all jobs are executed on the same machines as in schedule $\sigma(I)$. Schedule $\sigma_d(I_d)$ is the mirror schedule of $\sigma(I)$ and vice versa.

Lemma 1. *Schedule $\sigma_d(I_d)$ is feasible with respect to Ω'_0 and its makespan is equal to that of $\sigma(I)$.*

Proof. It is obvious that schedule $\sigma_d(I_d)$ satisfies the scheduling conditions. We just need to check that the resource constraints (1) also hold. In other words, we want to show that $\Omega'_\tau(\sigma_d) \geq 0$ for every $\tau \in [0, Z(\sigma)]$.

If $\tau = Z(\sigma)$, then all jobs have completed at time $Z(\sigma)$ in $\sigma(I)$. We get

$$\Omega'_{Z(\sigma)}(\sigma_d) = \Omega'_0 + \sum_{\{J_i \in \mathcal{N}\}} (\beta'_i - \alpha'_i) = \Omega'_0 - \sum_{\{J_i \in \mathcal{N}\}} (\beta_i - \alpha_i) = \Omega_0 \geq 0.$$

Let $\tau < Z(\sigma)$ and $t = Z(\sigma) - \tau$. We have

$$\begin{aligned}\Omega'_\tau(\sigma_d) &= \Omega'_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma_d) \leq \tau\}} (\beta'_i - \alpha'_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma_d) \leq \tau < C_i(\sigma_d)\}} \alpha'_i \\ &= \Omega_0 + \sum_{\{J_i \in \mathcal{N}\}} (\beta_i - \alpha_i) + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma_d) \leq \tau\}} (\alpha_i - \beta_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma_d) \leq \tau < C_i(\sigma_d)\}} \beta_i.\end{aligned}$$

Using Eq. (2) and collecting entries of the summation terms we get

$$\begin{aligned}\Omega'_\tau(\sigma_d) &= \Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) < t\}} \beta_i - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) < t\}} \alpha_i \\ &= \Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) < t\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) < t \leq C_i(\sigma)\}} \alpha_i.\end{aligned}$$

Let $t_0 < t$ be the last moment when some job starts or finishes. If no job starts before time t we set $t_0 = 0$. We can rewrite the above equality as follows.

$$\begin{aligned}&\Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) < t\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) < t \leq C_i(\sigma)\}} \alpha_i \\ &\quad = \Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) \leq t_0\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) \leq t_0 < C_i(\sigma)\}} \alpha_i \\ &\quad = \Omega_{t_0}(\sigma) \geq 0.\end{aligned}$$

Therefore, the proof is complete. □

Note that Lemma 1 implies that all results formulated below for the problems with positive (non-negative) net contributions also hold for the problems with negative (non-positive) net contributions.

Bin packing and relocation problems. From this point on we assume that all jobs have unit execution time. It is obvious that there exits an optimal schedule in which all jobs start and complete at integral time points and are executed during unit integral intervals. We can rewrite Eq. (1) as follows.

$$\Omega_t(\sigma) = \Omega_0 + \sum_{\{J_i \in \mathcal{N} | C_i(\sigma) \leq t\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} | s_i(\sigma) = t\}} \alpha_i \geq 0. \tag{3}$$

Let us consider the relocation problem with an unbounded number of parallel identical machines and all jobs have zero net contributions. In this case, the relocation problem is equivalent to the classical Bin Packing problem. Indeed, all jobs in the optimal schedule are executed in unit integral intervals. Unit intervals in the relocation problem correspond to bins in the Bin Packing problem, and the length of a schedule corresponds to the number of used bins. In the case when the number of machines is bounded by m, we get the Cardinality Constrained Bin Packing problem in which the maximum number of items packed into a bin is bounded by a positive integer m. It is well known that the last problem is strongly NP-hard for any $m \geq 3$ and can be reduced to the Matching Problem for $m = 2$. In the next section we shall prove the strong NP-hardness for the two-machine relocation problem with unit execution times and $\alpha_i < \beta_i$ for all jobs J_i, $i = 1, \ldots, n$.

Once the relocation problem is shown to be strongly NP-hard we are interested in developing approximation algorithms for finding good near-optimal solutions. We say that a polynomial-time algorithm A is a *ρ-approximation* algorithm (or provides a *ρ-approximation*) for a minimization problem, if for any instance I of the problem it outputs a feasible solution with the makespan at most $\rho \times \mathrm{OPT}(I)$. We say that a polynomial-time algorithm A is an *asymptotic ρ-approximation* algorithm for a makespan minimization problem, if there exists a constant c such that for any instance I of the problem it outputs a feasible solution with the makespan at most $\rho \times \mathrm{OPT}(I) + c$.

As a corollary of the results for Bin Packing, we have that unless $P = \mathrm{NP}$, there is no ρ-factor approximation algorithm for the unbounded relocation problem for any $\rho < \frac{3}{2}$ even if $\alpha_i = \beta_i$ and $p_i = 1$ for all i. However, there exist some algorithms for Bin Packing which have good asymptotic performances. The well-known result by Fernandez de la Vega and Lueker [5] implies that for any fixed $\varepsilon > 0$ and each instance I of the unbounded RP with zero net contribution and unit execution time, there exists a linear time algorithm A_ε such that $A_\epsilon(I) \leq (1+\epsilon)OPT(I) + \frac{1}{\epsilon^2}$, where $A_\epsilon(I)$ is the approximate makespan reported by algorithm A_ϵ. In the case with arbitrary α_i and β_i, the relocation problem is harder than Bin Packing from an approximation point of view also. In Section 4, we shall show that unless $P = \mathrm{NP}$, there is no ρ-asymptotic approximation algorithm for the unbounded relocation problem for any $\rho < \frac{4}{3}$.

3. Two machines

Now we consider the case where exactly two machines are available and $\delta_i > 0$ for any job J_i. To facilitate our discussion, we introduce another version of the relocation problem with dedicated parallel machines (RPD). For the case of dedicated parallel machines we assume that each job must be processed on a specified machine. We start from the case with two dedicated machines and all jobs have positive contributions. We denote this problem by RPD_2^+. Though the corresponding problem without resource requirements is trivial, we shall show that RPD_2^+ is NP-hard in the strong sense. We start with formulating an NP-complete problem that is to be used in our proof.

NUMERICAL MATCHING WITH TARGET SUMS (NMTS)[SP17, [4]]

Instance: Disjoint sets $X = \{x_1, \dots, x_n\}$ and $Y = \{y_1, \dots, y_n\}$, each containing n positive integers, and a target vector $(B_1, B_2, \dots, B_n)$.

Question: Can $X \cup Y$ be partitioned into n disjoint sets $A_1, A_2, \dots, A_n$ each containing exactly one element from each of X and Y such that for $1 \leq i \leq n$, $\sum_{e_j \in A_i} e_j = B_i$?

Without lost of generality we assume that

$$\sum_{i=1}^{n}(x_i + y_i) = \sum_{i=1}^{n} B_i. \tag{4}$$

As mentioned in [4] the strong NP-completeness of the NMTS problem can be obtained by a transformation from the well-known Numerical 3-Dimensional Matching problem.

NUMERICAL 3-DIMENSIONAL MATCHING (N3DM)[SP16, [4]]

Instance: Constant E and three disjoint sets of positive integers $X = \{x_1, \dots, x_n\}$, $Y = \{y_1, \dots, y_n\}$ and $Z = \{z_1, \dots, z_n\}$ such that $E/4 < x_i < E/2$, $E/4 < y_i < E/2$, $E/4 < z_i < E/2$, for $1 \leq i \leq n$.

Question: Can $X \cup Y \cup Z$ be partitioned into n disjoint sets $A_1, A_2, \dots, A_n$ such that each A_i, $1 \leq i \leq n$, contains exactly one element from each of X, Y and Z, and $\sum_{e_j \in A_i} e_j = E$?

Without loss of generality, $E/4$ is assumed to be integer, for otherwise we can multiply all integers by 4 and get an equivalent problem with the required property. Indeed, since each set A_i, $1 \leq i \leq n$, contains exactly one element from Z we can assign elements from Z to sets $A_1, A_2, \dots, A_n$ in an arbitrary way. Let set A_1 contain element z_1, set A_2 contain element z_2, and so on. Now we have to assign exactly one element from each of X and Y, such that, for $1 \leq i \leq n$, $\sum_{e_j \in A_i \setminus \{z_i\}} e_j = E - z_i$. It follows that N3DM can be polynomially transformed into NMTS. Moreover, we note that the bounds for elements of X, Y and Z in the N3DM problem imply the corresponding bounds for elements of X, Y and $B_1, B_2, \dots, B_n$ in the NMTS problem. For elements of X, Y, and $B_1, B_2, \dots, B_n$ in NMTS, there exists E such that $E/4 < x_i < E/2$, $E/4 < y_i < E/2$ and $E/2 < B_i < 3E/4$ for $1 \leq i \leq n$.

Theorem 1. *The* RPD_2^+ *problem is NP-hard in the strong sense.*

Proof. To establish the NP-hardness of the two-machine relocation problem, we present a polynomial-time reduction from the NMTS problem. Suppose that we are given two sets of integers $\{x_1, x_2, \dots, x_n\}$, and $\{y_1, y_2, \dots, y_n\}$, and a collection of targets $B_1, B_2, \dots, B_n$. Without loss of generality, we assume $B_1 \leq B_2 \leq \dots \leq B_n$. We group all targets with the same value together in a subset. Assume there are k different target values and let m_i be the cardinality of ith subset. We get $B_1 = \cdots = B_{m_1} < B_{m_1+1} = \cdots = B_{m_1+m_2} < \cdots < B_{m_1+m_2+\cdots+m_{k-1}+1} = \cdots = B_{m_1+m_2+\cdots+m_k} = B_n$. Let $M = 1 + \max_{i=1,\dots,k} m_i$. Set $\bar{B}_i = B_{m_1+m_2+\cdots+m_i}$ for all $1 \le i \le k$.

We define an instance I of RPD_2^+ as follows. In instance I, there are two machines, $2n$ basic jobs J_{1i}, $i = 1, 2, \dots, n$, and J_{2i}, $i = 1, 2, \dots, n$, and $k - 1$ connecting jobs $\bar{J}_i$, $i = 1, \dots, k - 1$. All connecting jobs and jobs J_{1i}, $i = 1, 2, \dots, n$, have to be executed on machine one, and jobs J_{2i}, $i = 1, 2, \dots, n$, have to be executed on machine two. For each integer x_i, we create basic job J_{1i} with $\alpha_{1i} = 2Mx_i$ and $\beta_{1i} = 2Mx_i + 1$. For each integer y_i, we create basic job J_{2i} with $\alpha_{2i} = 2My_i$ and $\beta_{2i} = 2My_i + 1$. Note that

$$\beta_{1i} > \alpha_{1i} > ME/2 \quad \text{for all } i = 1, \dots, n. \tag{5}$$

We use $\bar{\alpha}_i$, $\bar{\beta}_i$, $\bar{\delta}_i$ to denote the amount of required resources, amount of returned resources and net contributions of connecting job $\bar{J}_i$. For each connecting job $\bar{J}_i$, define $\bar{\alpha}_i = 2M\bar{B}_i + 2m_i$ and $\bar{\beta}_i = 2M\bar{B}_{i+1}$. Note that $\bar{\delta}_i = 2M\bar{B}_{i+1} - (2M\bar{B}_i + 2m_i) = 2M(\bar{B}_{i+1} - \bar{B}_i) - 2m_i \ge 2M - 2m_i > 0$. The inequality implies $\bar{\alpha}_i < \bar{\alpha}_{i+1}$ for all $i = 1, \dots, k - 2$. Moreover, we have

$$\bar{\alpha}_i = 2M\bar{B}_i + 2m_i > 2M\bar{B}_1 = 2MB_1 > ME. \tag{6}$$

Let the amount of initial resources be $\Omega_0 = 2MB_1$. We want to show that the NMTS problem has an affirmative answer if and only if instance I has a schedule σ with a makespan less than or equal to $n + k - 1$.

We first note that the maximum amount of available resources at any time cannot be greater than

$$\begin{aligned}
\Omega_0 + \sum_{i=1}^{n} \delta_{1i} + \sum_{i=1}^{n} \delta_{2i} + \sum_{i=1}^{k-1} \bar{\delta}_i &= 2MB_1 + 2n + \sum_{i=1}^{k-1} (2M\bar{B}_{i+1} - 2M\bar{B}_i - 2m_i) \\
&= 2MB_1 + \sum_{i=1}^{k} 2m_i + \sum_{i=1}^{k-1} (2M\bar{B}_{i+1} - 2M\bar{B}_i - 2m_i) \\
&\le 2M\bar{B}_k + 2m_k \\
&\le 2MB_n + 2M - 2 \\
&= 2M\left(B_n + 1 - \frac{1}{M}\right) \\
&< \frac{3ME}{2}.
\end{aligned} \tag{7}$$

The last strict inequality follows from the fact that B_n and $E/4$ are integer. Eq. (7) together with Eqs. (5) and (6) imply that no connecting job can be processed in parallel with any other job. This means that the makespan of any feasible schedule is at least $n + k - 1$ and that if the schedule has a makespan of $n + k - 1$, then all basic jobs have to be processed in pairs.

Proposition 1. *In any feasible schedule σ, connecting job $\bar{J}_i$, $1 \le i \le k-1$, cannot start before time $\sum_{j=1}^{i} m_j + i - 1$.*

Proof. The proof is established by induction on index i. Let us consider $\bar{J}_1$. Since $\bar{\alpha}_1 = 2M\bar{B}_1 + 2m_1 > 2M\bar{B}_1$, no connecting job can start its execution at time 0. Recall that the net contribution of each basic job is equal to 1 and at most two jobs can be simultaneously processed. As a consequence, job $\bar{J}_1$ cannot start before time m_1.

Assume the proposition is valid for some $i \ge 1$. Then, the amount of resources available at time $\sum_{j=1}^{i} m_j + i$ does not exceed $2M\bar{B}_1 + \sum_{j=1}^{i} 2m_j + \sum_{j=1}^{i} (2M\bar{B}_{j+1} - 2M\bar{B}_j - 2m_j) = 2M\bar{B}_{i+1}$. All connecting jobs $\bar{J}_j$ with $j \ge i + 1$ have $\bar{\alpha}_j \ge 2M\bar{B}_{i+1} + 2m_{i+1}$. It follows that job J_{i+1} cannot start before time $\sum_{j=1}^{i} m_j + i + m_{i+1} = \sum_{j=1}^{i+1} m_j + (i + 1) - 1$. The induction proof is therefore complete.

We now show the if-and-only-if relation between the NMTS problem and the RPD_2^+ problem.

IF: Suppose that in the NMTS problem, there is a partition of the $2n$ integers into n disjoint sets, $A_1, A_2, \dots, A_n$, each of which consists of exactly two elements such that $\sum_{e_j \in A_i} e_j = B_i$, $1 \le i \le n$. We group the basic jobs in pairs in accordance with their indices in the solution to the NMTS problem. The equality $\sum_{\{j: e_j \in A_i\}} \alpha_j = 2MB_i$, $1 \le i \le n$, readily follows. Let $m_0 = 0$. We schedule connecting job $\bar{J}_i$ at time $\sum_{j=1}^{i} m_j + i - 1$ for all $1 \le i \le k - 1$, and

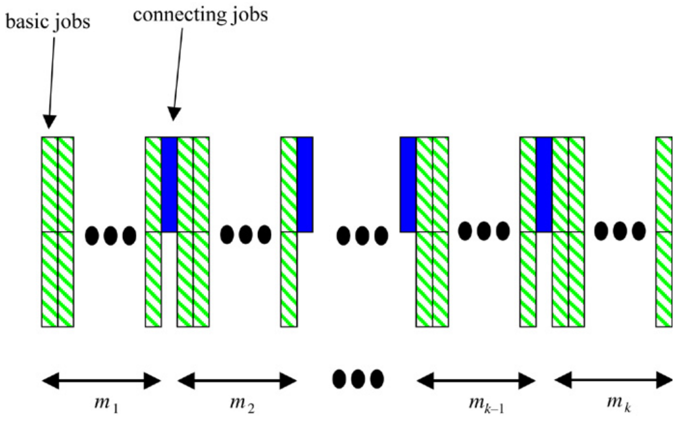

Fig. 1. Configuration of an optimal schedule of the proof for Theorem 1.

arrange the pairs of basic jobs with total resource requirement $2M\bar{B}_i$, $1 \le i \le k$ in the integral time slots within the interval $[\sum_{j=0}^{i-1} m_j + i - 1, \sum_{j=1}^{i} m_j + i - 1)$ in arbitrary order. Refer to Fig. 1 for the configuration. It is easy to check that this schedule is feasible and has a makespan equal to $n + k - 1$.

ONLY IF: Suppose there is a schedule σ with all jobs completed at time $n + k - 1$. As aforementioned, in σ all basic jobs must be scheduled in pairs. We number the pairs of basic jobs from 1 to n as they appear in schedule σ. Let B_i' be a sum of two integers, whose indices correspond to the indices of basic jobs of pair i. If $B_i' = B_i$ for all i then we have the desired partition of the NMTS problem. Otherwise, Eq. (4) implies that there exists index i that satisfyies $B_i' > B_i$. Let $B_i = \bar{B}_l$. Then, i must be smaller than or equal to $\sum_{j=1}^{l} m_j$. Proposition 1 implies that at most $l - 1$ connecting jobs can be finished at time $\sum_{j=1}^{l} m_j + l - 1$. It further implies that basic jobs from pair i start at or before time $t \le \sum_{j=1}^{l} m_j + l - 2$. The amount of resources available at time t is

$$\begin{aligned} \Omega_t(\sigma) &\le \Omega_0 + 2\sum_{j=1}^{i} \delta_j + \sum_{j=1}^{l-1} \bar{\delta}_i \\ &\le 2M\bar{B}_1 + 2\sum_{j=1}^{l} m_j + \sum_{j=1}^{l-1} (2M\bar{B}_{j+1} - 2M\bar{B}_j - 2m_j) \\ &= 2M\bar{B}_l + 2m_l < 2M\bar{B}_l + 2M \le 2M\bar{B}_i'. \end{aligned}$$

The last inequality follows from the inequality $B_i' > B_i$ and the integrality of B_i' and B_i. The proof of the theorem follows from contradiction to the assumption that there exists index i such that $B_i' > B_i$. $\square$

The above reduction from NMTS establishes the strong NP-hardness of the RPD_2^+ problem. In the following, we perform a transformation from RPD_2 to the studied problem RP_2^+. Let us consider the decision problem of RPD_2 as defined below.

RELOCATION PROBLEM WITH TWO DEDICATEDMACHINES (RPD_2)

Instance: Two parallel dedicated machines, Ω_0 units of initial resources, set $\mathcal{N}$ of jobs, $|\mathcal{N}| = n + l$, jobs $J_1, \ldots, J_n$ have to be executed by machine 1 and jobs $J_{n+1}, \ldots, J_{n+l}$ have to be executed by machine 2, $0 < l < n$. Each job $J_i \in \mathcal{N}$ has processing length 1, resource requirements α_i and β_i.

Question: Is there a two-machine schedule for $\mathcal{N}$ that meets makespan n and obeys the resource constraints Eq. (3)?

The unary NP-completeness of RPD_2 follows from Theorem 1 concerning RPD_2^+. Now we show that RPD_2 polynomially transforms to RP_2^+. Let I be an instance of RPD_2 with $n + l$ jobs. We keep the notation used in the proof of Theorem 1 in the following proof. Define $\Psi = \Omega_0 + \sum_{J_i \in \mathcal{N}} \delta_i$. We construct an instance $\bar{I}$ of RP_2^+ as follows. Instance $\bar{I}$ also consists of $n + l$ jobs. We set $\bar{\Omega}_0 = \Psi + \Omega_0$, $\bar{\alpha}_i = \Psi + \alpha_i$ and $\bar{\beta}_i = \Psi + \beta_i$ for $i = 1, \ldots, n$. These jobs are called connecting jobs. Also we set $\bar{\alpha}_i = \alpha_i$ and $\bar{\beta}_i = \beta_i$ for basic jobs J_i, $i = n + 1, \ldots, n + l$. We show that the instance I of RPD_2 has a schedule with $\mathrm{OPT}(I) = n$ if and only if the instance $\bar{I}$ of RP_2^+ has a schedule with $\mathrm{OPT}(\bar{I}) = n$.

First, we note that for RPD_2 any feasible schedule $\sigma(I)$ of instance I is also a feasible schedule of instance $\bar{I}$. The amount of resources available at any time depends on the amount of initial resources and the total contribution of the completed jobs. The amount of initial resources is increased from Ω_0 to $\Omega_0 + \Psi$, and the total contribution of all jobs remain the same. It follows that the amount of available resources at any time increases by Ψ. Since all connecting jobs have to be executed by machine 1, each time interval must contain at most one connecting job. Therefore, the resource consumption will increase by at most Ψ in each unit time interval and $\sigma(\bar{I})$ is also feasible.

Since Ψ is the upper bound on the maximum amount of resources available at any time, no two connecting jobs can be processed in parallel. This means that the makespan of any feasible schedule is at least n. Let $\bar{\sigma}(\bar{I})$ be a feasible schedule with makespan n. It follows that in each unit time interval from 0 to n, some connecting job is executed. Because there are only two machines, at most one basic job can be assigned to the same unit time interval in $\bar{\sigma}(\bar{I})$. Thus we can suppose that all connecting jobs are dispatched to machine 1 and all basic jobs to machine 2. The feasibility of $\bar{\sigma}(\bar{I})$ implies that for all $t = 0, 1, \ldots, n$,

$$\begin{aligned} 0 \le \bar{\Omega}_t(\bar{\sigma}(\bar{I})) &= \Psi + \Omega_0 + \sum_{\{J_i \in \mathcal{N} \mid C_i(\bar{\sigma}) \le t\}} (\bar{\beta}_i - \bar{\alpha}_i) - \sum_{\{J_i \in \mathcal{N} \mid s_i(\bar{\sigma}) = t\}} \bar{\alpha}_i \\ &= \Psi + \Omega_0 + \sum_{\{J_i \in \mathcal{N} \mid C_i(\bar{\sigma}) \le t\}} (\beta_i - \alpha_i) - \sum_{\{J_i \in \mathcal{N} \mid s_i(\bar{\sigma}) = t\}} \alpha_i - \Psi = \Omega_t(\bar{\sigma}(I)). \end{aligned}$$

Thus we have proved that $\bar{\sigma}(I)$ is a feasible schedule of instance I and has the same makespan n. The above discussion is concluded with the following theorem.

Theorem 2. *The* RP_2^+ *problem is* NP*-hard in the strong sense.*

4. Non-approximability

In this section, we consider the relocation problem with an unbounded number of parallel identical machines (RP_∞) and present a negative result concerning its approximability.

Theorem 3. *Unless* $P = \mathrm{NP}$*, there is no* ρ*-asymptotic approximation algorithm for the* RP_∞ *problem for any* $\rho < \frac{4}{3}$*, even if* $\alpha_i \le \beta_i$ *and* $p_i = 1$ *for all* i*.*

Proof. We consider an instance I of the Partition problem: Given a finite set of integers $X = \{e_1, e_2, \ldots, e_n\}$ with $\sum_{e_i \in X} e_i = 2E$, is there a subset $X_1 \subseteq X$ such that $\sum_{e_i \in X_1} e_i = E$?

We define an instance I' of the RP_∞ problem as follows. Let $K > 0$ be a fixed integer. We introduce $n(K+1)$ basic jobs J_{ki} with $\alpha_{ki} = (E+1)^k e_i$ and $\beta_{ki} = (E+1)^k e_i + \frac{1}{n}$ for all $k = 0, 1, \ldots, K$ and $i = 1, 2, \ldots, n$. Note that for convenience parameter β_{ki} does not abide by the integer constraint. The violation can be resolved by scaling all parameters with a factor n. The jobs are grouped into $K+1$ sets of n jobs with the same first index, i.e. set $N_k = \{J_{k1}, J_{k2}, \ldots, J_{kn}\}$. We define K connecting jobs $\bar{J}_k$, $1 \le k \le K$, with $\bar{\alpha}_k = E(E+1)^{k-1} + 1$ and $\bar{\beta}_k = E(E+1)^k$. The amount of initial resources is $\Omega_0 = E$.

We use the notation of schedule concatenation as follows: $\sigma = N_0 N_1 N_2 \ldots N_l$ means that the jobs of set N_i are followed by the jobs of set N_{i+1} in schedule σ for all $i = 0, 1, 2, \ldots, l-1$. It is easy to see that in any feasible schedule σ, the jobs are processed in the following order $\sigma = N_0\{\bar{J}_1\}N_1\{\bar{J}_2\}N_2\{\bar{J}_3\}N_3 \ldots \{\bar{J}_K\}N_K$. Since the amount of initial resources is equal to E, only the jobs from set N_0 are available for processing at time zero. Moreover, since the net contribution of each basic job is equal to $\frac{1}{n}$, only the connecting job $\bar{J}_1$ becomes available when all jobs from set N_0 finish their processing. After finishing job $\bar{J}_1$, the amount of resources is equal to $E(E+1)$ and we can schedule only jobs from set N_1. This line of reasoning continues for the remaining jobs (Fig. 2). In schedule σ, at the time job J_i, $i = 1, \ldots, K$ is to be processed, it will consumes all available resources. It follows that no job can be processed in parallel with a connecting job J_i. Now consider the jobs from any set N_k, $0 \le k \le K$. They require $\sum_{i=1}^n \alpha_{ki} = \sum_{i=1}^n (E+1)^k e_i = 2E(E+1)^k$ units of resources. The amount of resources available after finishing jobs in $N_0\{\bar{J}_1\}N_1\{\bar{J}_2\}N_2\{\bar{J}_3\}N_3 \ldots \{\bar{J}_k\}$ is equal to $E(E+1)^k$. Since $\delta_{ki} = \frac{1}{n}$ and all e_i are integers we can schedule the jobs of set N_k in time interval of length 2 if and only if there exists a partition of the job set

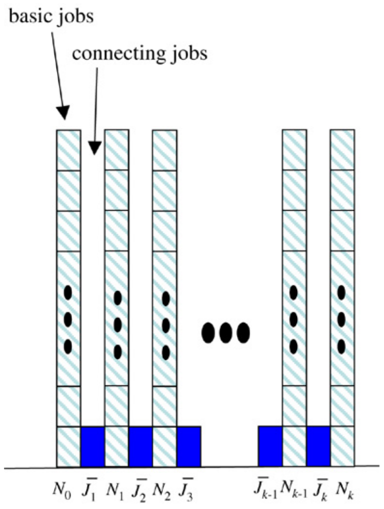
basic jobs
connecting jobs
N_0 $\bar{J}_1$ N_1 $\bar{J}_2$ N_2 $\bar{J}_3$ $\bar{J}_{k-1}$ N_{k-1} $\bar{J}_k$ N_k

Fig. 2. Configuration of the optimal schedule in the proof of Theorem 2.

into sets X_1 and X_2 such that $\sum_{e_i \in X_1} e_i = \sum_{e_i \in X_2} e_i = E$. Let $\text{OPT}(I')$ be the value of an optimal schedule for instance I'. If there exists a subset $X_1 \subseteq X$ such that $\sum_{e_i \in X_1} e_i = E$ in the instance I of the Partition problem, then the makespan $\text{OPT}(I') = 2(K+1) + K = 3K + 2$. If I is a negative instance of the Partition problem, then $\text{OPT}(I') = 3(K+1) + K = 4K + 3$.

Now suppose there is a ρ-asymptotic approximation algorithm A with $\rho < \frac{4}{3}$ for the RP_∞ problem. Let $\rho = \frac{4}{3} - \varepsilon$ for some fixed $\varepsilon \in (0, \frac{1}{3}]$ and let $c \geq 0$ be the fixed constant from the definition of asymptotic approximation algorithm. Let us consider instance I' of the RP_∞ problem as described above with fixed $K = \frac{c}{3\varepsilon}$. Without lost of generality, we assume that K is integer. If instance I is an affirmative instance of the Partition problem, then algorithm A produces a schedule with a makespan of

$$A(I') \leq \rho \times \text{OPT}(I') + c = \left(\frac{4}{3} - \varepsilon\right)(3K+2) + c = 4K + \frac{8}{3} - 2\varepsilon - 3K\varepsilon + c < 4K + \frac{8}{3}.$$

Thus, algorithm A finds the schedule with a makespan less than $4K + 3$. It follows that the Partition problem can be solved in polynomial time, a contradiction to the conjecture $\text{P} \neq \text{NP}$. □

5. Approximation algorithms

In this section we shall design and analyze some greedy heuristics for the relocation problem. The discussion is based upon a relaxation form similar to the case with preemption.

In schedule σ, job J_i is said to be *available* at time t, $t = 0, \ldots, Z(\sigma) - 1$ if $\alpha_i \leq \Omega_t(\sigma)$. Assume the processing of each job could be divided into non-consecutive fragments or fractions. Fragments of a job can be processed in one or more non-consecutive unit-time intervals. Let $0 \leq x_{it} \leq 1$ be the fragment of job J_i which is processed in time interval $[t, t+1)$. Then, job J_i requires $\alpha_i x_{it}$ units of resources at time t and returns $\beta_i x_{it}$ units of resources at time $t+1$. If $x_{it} > 0$, then the processing time of job J_i in time interval $[t, t+1)$ does not depend on the value x_{it} and the processing occupies the whole interval. We say that job J_i is *uncompleted* at time $\tau \geq 1$ if $\sum_{t=0}^{\tau-1} x_{it} < 1$. All jobs are uncompleted at time 0. An assignment σ of jobs to unit time intervals is a feasible schedule of the Relaxed Relocation Problem (RRP) if the following conditions are satisfied:

1. $x_{it} = 0$ for each job J_i with $\alpha_i > \Omega_t(\sigma)$, for all $t = 0, \ldots, Z(\sigma) - 1$,
2. $\sum_{i=1}^{n} \alpha_i x_{it} \leq \Omega_t(\sigma)$ for all $t = 0, \ldots, Z(\sigma) - 1$, and
3. $\sum_{t=0}^{Z(\sigma)-1} x_{it} = 1$, for all $i = 1, \ldots, n$.

Note that an assignment of jobs to unit time intervals is a feasible schedule of the original problem when it satisfies the above three conditions and $x_{it} \in \{0, 1\}$. It follows that the value of the optimal solution to the relaxed relocation problem (RRP) is a lower bound on the value of optimal solution to the original relocation problem (RP∞).

Let I be an instance of the relocation problem. We consider the relaxed version RRP. The *skyline* of schedule $\sigma(I)$ is a vector $\overrightarrow{v_\sigma} = [\Omega_0(\sigma), \Omega_1(\sigma), \ldots, \Omega_{Z(\sigma)}(\sigma)]$. Let σ and σ' be two feasible schedules and $T \leq \min\{Z(\sigma), Z(\sigma')\}$. We say that skyline $\overrightarrow{v_\sigma}$ is greater than or equal to skyline $\overrightarrow{v_{\sigma'}}$ over interval [0,T], and denote by $\overrightarrow{v_\sigma} \prec_T \overrightarrow{v_{\sigma'}}$, if $\Omega_t(\sigma) \geq \Omega_t(\sigma')$ for all $0 \leq t \leq T$. Skyline $\overrightarrow{v_\sigma}$ is maximum over interval $[0, T]$ if $\overrightarrow{v_\sigma} \prec_T \overrightarrow{v_{\sigma'}}$ for any feasible schedule σ'. Next, we present a greedy algorithm for RRP and discuss some related properties. Starting from time interval [0, 1], the greedy algorithm schedules available jobs so as to maximize the total return of the scheduled jobs. Step by step the algorithm solves the well-known Fractional Knapsack problem on the set of available jobs.

FRACTIONAL KNAPSACK

Instance: Nonnegative integers $n, \alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n, \bar{y_1}, \bar{y_2}, \ldots, \bar{y_n}$ and V.

Task: Find numbers $y_1, y_2, \ldots, y_n$ such that $0 \leq y_i \leq \bar{y_i}$ for all $i = 1, \ldots, n$, and $\sum_{j=1}^n \alpha_j y_j \leq V$, and $\sum_{j=1}^n \beta_j y_j$ is maximum.

The following observation suggests a simple algorithm which requires sorting the elements appropriately:

Proposition 2 ([3]). *Let $\alpha_1, \ldots, \alpha_n$, $\beta_1, \ldots, \beta_n$ and V be nonnegative integers with*

$$\frac{\beta_1}{\alpha_1} \geq \frac{\beta_2}{\alpha_2} \geq \cdots \geq \frac{\beta_n}{\alpha_n}$$

and let $k = \min\{j \in \{1, \ldots, n\} : \sum_{i=1}^j \alpha_i \bar{y_i} > V\}$. Then an optimal solution to the given instance of Fractional Knapsack is defined by

$$y_j := \bar{y}_j, \quad \text{for } j = 1, \ldots, k-1;$$
$$y_k := \frac{V - \sum\limits_{j=1}^{k-1} \alpha_j \bar{y}_j}{\alpha_k};$$
$$y_j := 0, \quad \text{for } j = k+1, \ldots, n.$$

Greedy Algorithm 1.

```
1. Set $t := 0$ and $V := \Omega_0$.
   Set $\bar{x}_j := 1$ for $j = 1, \ldots, n$.
2. For $j := 1$ to $n$ do:
    If $\alpha_j \leq V$ then set $\bar{y}_j := \bar{x}_j$
    else set $\bar{y}_j := 0$.
3. If $\bar{y}_j = 0$ for all $j = 1, \ldots, n$
   then stop (No more fragment can be assigned.)
   else solve the Fractional Knapsack Problem.
4. Let $y_j$, $j = 1, \ldots, n$ be a solution to the Fractional Knapsack Problem.
   Set $x_{jt} := y_j$, $\bar{x}_j := \bar{x}_j - x_{jt}$ for $j = 1, \ldots, n$.
   Set $V := V + \sum_{j=1}^n x_{jt}\delta_j$ and $t := t + 1$.
   Go to 2.
```

Lemma 2. *Let I be an instance of the* RRP$^+$ *problem.*

1. *If there exists some uncompleted job when Greedy Algorithm 1 terminates, then there is no feasible schedule of instance I for the* RRP *problem.*
2. *If there exists a feasible schedule for instance I, then Greedy Algorithm 1 provides an optimal solution σ^*.*
3. *Skyline $\overrightarrow{v_{\sigma^*}}$ is maximum over the interval* $[0, Z(\sigma^*)]$.

Proof. Suppose that there exists nonempty set N' of unscheduled jobs when Greedy Algorithm 1 terminates. Inequality $\alpha_i > \Omega_0 + \sum_{j \in N \setminus N'} \delta_j$ must hold for each job $J_i \in N'$. Let there be some feasible schedule σ and $J_i \in N'$ be the job that starts first over all jobs from N'. Let N'' be the set of jobs that have completed their processing in schedule σ before the commencement of job J_i. Note that $N'' \subseteq N \setminus N'$. Then, $\alpha_i \leq \Omega_0 + \sum_{j \in N''} \delta_j \leq$

$\Omega_0 + \sum_{j \in N \setminus N'} \delta_j < \alpha_i$. The second inequality follows because all jobs have nonnegative net contributions. The contradiction implies the validity of the first proposition.

Let y_{it} be any feasible solution of instance I. We show that solution y_{it} can be transformed into the solution x_{it} produced by Greedy Algorithm 1 without increasing the makespan. Let $[\tau, \tau + 1)$ be the first time interval in which there exists some job J_i with $x_{i\tau} > y_{i\tau}$. Since both solutions are the same before time τ, they have the same resource level Ω_τ at time τ. It follows that the sets of available jobs are also the same. We enumerate the available jobs in the same order as they were enumerated in Greedy Algorithm 1 and let J_j be the job with the smallest index such that $x_{j\tau} > y_{j\tau}$. Since $\sum_{t=0}^{\tau-1} x_{jt} = \sum_{t=0}^{\tau-1} y_{jt}$, there exists some time point $\tau' > \tau$ with $y_{j\tau'} > 0$. If $\sum_{J_i \in N} y_{i\tau}\alpha_i < \Omega_\tau$, then we increase $y_{j\tau}$ by

$$\frac{\min\left\{\alpha_j y_{j\tau'}, \Omega_\tau - \sum_{i \in N} \alpha_i y_{i\tau}\right\}}{\alpha_j}.$$

In the other case, there exists some job J_k with $k \geq j$ and $y_{k\tau} > 0$. Let $\alpha = \min\{\alpha_j y_{j\tau'}, \alpha_k y_{k\tau}\}$. Since all jobs have nonnegative net contributions, $\Omega_{\tau'}$ must be greater than or equal to Ω_τ and job J_k is available at time τ'. We swap a $\frac{\alpha}{\alpha_j}$-portion of job J_j and a $\frac{\alpha}{\alpha_k}$-portion of job J_k. Denote by V'_t the amount of resources at time t in the derived schedule. Then, $V'_t = \Omega_t$ for all $t \leq \tau$ and $t > \tau'$. For $\tau < t \leq \tau'$, we have $V'_t = \Omega_t + \beta_j \frac{\alpha}{\alpha_j} - \beta_k \frac{\alpha}{\alpha_k} = \Omega_t + \alpha(\frac{\beta_j}{\alpha_j} - \frac{\beta_k}{\alpha_k}) \geq \Omega_t$. It follows that all resource requirements are satisfied and the new schedule is feasible. Repeating this process, we can finally transform the solution y_{it} into solution x_{it} without increasing the makespan. Furthermore, the amount of resources available at each time point will not decrease. As a sequel, propositions 2 and 3 in the lemma are established. □

Note that Greedy Algorithm 1 deals with the case where all jobs have non-negative contributions. Given instance I of the RRP with $\delta_i \leq 0$ for all job $J_i \in N$, we let $t_d = Z(\sigma^*) - t - 1$. Then, assignment $y_{it_d} = x_{it}$ gives us a mirror schedule σ_d of σ. Therefore, we can apply Greedy Algorithm 1 to the mirror instance I_d and use the obtained solution for constructing an optimal schedule.

Using the above results, we now consider the relaxed relocation problem with arbitrary δ_i. Let N^+ be the set of jobs with $\alpha_i \leq \beta_i$ and N^- be the set of jobs with $\alpha_i > \beta_i$. The following result is a simple consequence of the feasibility testing procedure of Kaplan and Amir [1].

Lemma 3. *If there exists a feasible schedule for an instance I of problem* RRP*, then the exists a feasible schedule in which the jobs of set N^- start after finishing all jobs from set N^+.*

Let I be an instance of the RRP problem with job set N and initial amount of resources Ω_0. We define two new instances I^+ and I^- as follows. Instance I^+contains the jobs of $N^+$and the amount of initial resources is equal to Ω_0. Instance I^- contains the jobs of N^- and the amount of initial resources is equal to $\Omega_0 + \sum_{J_i \in N^+} \delta_i$. The following procedure is designed to provide solutions to the RRP problem.

Greedy Algorithm 2.

1. Apply Greedy Algorithm 1 to I^+. Let $\sigma(I^+)$ be the schedule output by the procedure.
2. Apply Greedy Algorithm 1 to the mirror instance I_d^- of instance I^-. Denote the obtained schedule by $\sigma(I_d^-)$. Let $\sigma(I^-)$ be a mirror schedule of $\sigma(I_d^-)$.
3. Report schedule $\sigma(I) = \sigma(I^+)\sigma(I^-)$ for the RRP problem, where $\sigma(I^+)\sigma(I^-)$ denotes schedule concatenation by appending $\sigma(I^-)$ to $\sigma(I^+)$.

Lemma 4. *Let I be an instance of the* RRP *problem. If there exists uncompleted jobs when Greedy Algorithm* 2 *terminates at Steps* 1 *or* 2*, then there exists no feasible schedule for instance I.*

Proof. The lemma directly follows from Lemma 3 and the first proposition of Lemma 2. □

Let $\sigma^*(I) = \sigma^*(I^+)\sigma^*(I^-)$ be a schedule obtained by Greedy Algorithm 2 on instance I. We say that

- time slot $[t, t + 1)$ is *full* if $\sum_{J_i \in N} \alpha_i x_{it} = \Omega_t(\sigma^*)$,

- time slot $[t, t+1)$ is *vacant* if $\sum_{J_i \in N} \alpha_i x_{it} < \Omega_t(\sigma^*)$, and there exists job $J_i \in N$ such that $x_{it} = 1$.
- time slot $[t, t+1)$ is *dummy* if $\sum_{J_i \in N} \alpha_i x_{it} < \Omega_t(\sigma^*)$, and $x_{it} < 1$ for all jobs $J_i \in N$.

Now we list some simple properties of schedule $\sigma^*(I)$ which will be used to establish the proofs of performance ratios.

Lemma 5. *Let $\sigma_d^*(I)$ be a mirror schedule of $\sigma^*(I)$ and $t_d = Z(\sigma^*) - t - 1$. If time slot $[t, t+1)$ is full in $\sigma^*(I)$, then time slot $[t_d, t_d + 1)$ is full in $\sigma_d^*(I)$.*

Proof. Since time slot $[t, t+1)$ is full, we readily have that $\Omega_{t_d}(\sigma_d^*) = \Omega_{t+1}(\sigma^*) = \Omega_t(\sigma^*) - \sum_{J_i \in N} \alpha_i x_{it} + \sum_{J_i \in N} \beta_i x_{it} = \sum_{J_i \in N} \beta_i x_{it} = \sum_{J_i \in N} \beta_i y_{it_d}$. □

For notational convenience, we denote by I_t the time interval $[t, t+1)$. Also, denote by Φ^+ the set of full time slots in $\sigma^*(I^+)$, Φ^- the set of full time slots in $\sigma^*(I^-)$, Ψ^+ the set of vacant time slots in $\sigma^*(I^+)$ and Ψ^- the set of vacant time slots in $\sigma^*(I^-)$. Let $A = \sum_{J_i \in N} \alpha_i$.

Lemma 6. *If $|\Psi^+| = 1$, then*

$$A > \sum_{I_t \in \Phi^+} \Omega_t(\sigma^*) + \sum_{I_t \in \Phi^-} \Omega_t(\sigma^*). \tag{8}$$

If $|\Psi^+| > 1$ and $\tau_1, \ldots, \tau_{|\Psi^+|}$ be the vacant time slots in $\sigma^*(I^+)$, then

$$A > \sum_{I_t \in \Phi^+} \Omega_t(\sigma^*) + \sum_{I_t \in \Phi^-} \Omega_t(\sigma^*) + \sum_{i=1}^{|\Psi^+|-1} \Omega_{\tau_i}(\sigma^*). \tag{9}$$

Proof. By the definition of full time slots, the total size of the jobs arranged in these full time slots exceeds the first two terms of the right-hand-side of the inequalities (8) and (9).

If $|\Psi^+| = 1$, then the strict inequality in (8) follows from the fact that the vacant time slot has at least one whole job.

If $|\Psi^+| > 1$, we consider two vacant time slots I_{τ_i} and $I_{\tau_{i+1}}$ for $i = 1, \ldots, |\Psi^+| - 1$. By the definition of vacant slots, there exists job $J_j \in N$ with $x_{j\tau_{i+1}} = 1$. Since this job does not start at time τ_i, we have $\alpha_j > \Omega_{\tau_i}(\sigma^*)$. Also, $x_{j\tau_{i+1}} = 1$ implies that this job is not counted when we count the jobs in full time slots. It follows that $\sum_{i=1}^{|\Psi^+|} \sum_{j=1}^{n} \alpha_j x_{j\tau_i} > \sum_{i=1}^{|\Psi^+|-1} \Omega_{\tau_i}(\sigma^*)$. □

Lemma 7. *For any feasible schedule $\sigma(I)$,*

$$Z(\sigma) \geq \text{OPT} \geq |\Phi^+| + |\Psi^+| + |\Phi^-|.$$

Proof. Let $\sigma^*(I) = \sigma^*(I^+)\sigma^*(I^-)$ be the schedule obtained by Greedy Algorithm 2 for input instance I. Let $T = Z(\sigma^*(I^+))$. By Lemma 2, skyline $\overrightarrow{v_{\sigma^*}} = [\Omega_0(\sigma^*), \Omega_1(\sigma^*), \ldots, \Omega_T(\sigma^*)]$ is maximum over interval $[0, T]$. Moreover, since instance I^+ contains only jobs with non-negative net contributions we have

$$\Omega_0(\sigma^*) \leq \Omega_1(\sigma^*) \leq \cdots \leq \Omega_T(\sigma^*). \tag{10}$$

Let $\nu_1 < \nu_2 < \cdots < \nu_{|\Phi^+|+|\Psi^+|-1}$ be the full time slots and vacant time slots except the last one in $\sigma^*(I^+)$. We consider a special vector $\overrightarrow{v^*} = [\Omega_{\nu_1}(\sigma^*), \Omega_{\nu_2}(\sigma^*), \ldots, \Omega_{\nu_{|\Phi^+|+|\Psi^+|-1}}(\sigma^*)]$. Inequality (10) implies that $\Omega_{\nu_t}(\sigma^*) \geq \Omega_t(\sigma^*)$ for any t, $0 \leq t \leq \nu_{|\Phi^+|+|\Psi^+|-1} + 1$. Thus in any schedule the total amount of resources available in the first $|\Phi^+| + |\Psi^+| - 1$ time slots does not exceed $\sum_{t=1}^{|\Phi^+|+|\Psi^+|-1} \Omega_{\nu_t}(\sigma^*) = \sum_{I_t \in \Phi^+} \Omega_t + \sum_{i=1}^{|\Psi^+|-1} \Omega_{\tau_i}$.

Now consider the mirror instance I_d. Let $\mu_1 < \mu_2 < \cdots < \mu_{|\Phi^-|}$ be the full time slots in $\sigma_d^*(I^-)$. Let $T_d = Z(\sigma_d^*(I^-))$. From Lemma 2, we have that skyline $\overrightarrow{v_{\sigma_d^*}} = [\Omega_0(\sigma_d^*), \Omega_1(\sigma_d^*), \ldots, \Omega_{T_d}(\sigma_d^*)]$ is maximum. We determine a special mirror vector $\overrightarrow{v_d^*} = [\Omega_{\mu_1}(\sigma_d^*), \Omega_{\mu_2}(\sigma_d^*), \ldots, \Omega_{\mu_{|\Phi^-|}}(\sigma_d^*)]$. As a consequence, the total amount of resources available in the last $|\Phi^-|$ time slots does not exceed $\sum_{t=0}^{|\Phi^-|} \Omega_{\mu_t}(\sigma_d^*) = \sum_{I_t \in \Phi^-} \Omega_t$.

Assume σ is a feasible schedule with a makespan less than $|\Phi^+| + |\Phi^-| + |\Psi^+|$. It follows that the total capacity of all time slots in σ cannot be greater than $\sum_{I_t \in \Phi^+} \Omega_t + \sum_{I_t \in \Phi^-} \Omega_t + \sum_{i=1}^{|\Psi^+|-1} \Omega_{\tau_i}$. However, Lemma 6 implies that the total requirement of all jobs is

$$A > \sum_{I_t \in \Phi^+} \Omega_t + \sum_{I_t \in \Phi^-} \Omega_t + \sum_{i=1}^{|\Psi^+|-1} \Omega_{\tau_i}.$$

We get a contradiction to the assumption that schedule σ is feasible. □

Following the same line of reasoning, we have the following result.

Lemma 8. *For any feasible schedule* $\sigma(I)$,

$$Z(\sigma) \geq \text{OPT} \geq |\Phi^-| + |\Psi^-| + |\Phi^+|.$$

Proof. Similar to the proof of Lemma 7. □

With the above discussion about the relaxed problem RRP, we now return to the original relocation problem. Let σ be a schedule obtained by Greedy Algorithm 2 for the relaxed version. Job J_i is called *fragmented* if it is executed in at least two time slots in σ. We round all fractional values of assignment x_{it} to either 0 or 1.

Rounding Algorithm.

```
1. Apply Greedy Algorithm 2 on instance $I$.
2. For each fragmented job $J_i$ do
   (a) If $\delta_i \geq 0$, then set $\tau = \min\{t | x_{it} > 0\}$.
   (b) If $\delta_i < 0$, then set $\tau = \max\{t | x_{it} > 0\}$.
   (c) Insert an empty slot at time $\tau$.
   (d) Collect all fragments of job $J_i$ and schedule it as a whole in this time slot.
3. Eliminate all empty time slots and report the schedule.
```

Theorem 4. *For problem* RP_∞, *Rounding Algorithm produces a feasible schedule, if exists, and has a performance ratio of* 2.

Proof. First, we show that Rounding Algorithm can produce a feasible schedule, if exists. By virtue of Lemma 4, Greedy Algorithm 2 constructs a feasible schedule of RRP. Adding a new empty slot does not violate the feasibility of this schedule. Let job J_i be a job with fractional assignment and $\delta_i \geq 0$ and $\tau = \min\{t | x_{it} > 0\}$. Moving any part of job J_i to the left of the time horizon does not decrease the resource capacity of any time slot. Thus, we need only to check if there are sufficient resources for job J_i in the new time slot. Since $x_{i\tau} > 0$, job J_i is available at time τ and $\alpha_i \leq \Omega_\tau \leq \Omega_{\tau+1}$. The last inequality follows from the fact that all jobs have positive net contributions in $[\tau, \tau+1)$. Therefore, we can schedule job J_i in the new time slot which starts at time $\tau + 1$. Similar reasoning for the mirror schedule is applied to Step 2 for the jobs with negative contributions, $\delta_i < 0$.

Note that the number of infeasible jobs does not exceed the number of full time slots. Furthermore, all dummy time slots are eliminated at the end of Rounding Algorithm. Therefore, the makespan of the obtained schedule $Z(\sigma)$ is no greater than $2|\Phi^+| + 2|\Phi^-| + |\Psi^+| + |\Psi^-|$. This fact together with the results of Lemmas 7 and 8 lead to inequality $Z(\sigma) \leq 2 \times \text{OPT}$. The proof is complete. □

Next we proceed to the problem with a finite number of machines. Let σ be a feasible schedule of RP obtained by Rounding Algorithm. We derive schedule σ' by reorganizing the sets of jobs in the intervals in which the number of scheduled jobs exceeds the number of machines.

Splitting Algorithm.

```
1. Let $I$ be an instance of RP. Apply Rounding Algorithm to $I$ and let $\sigma$ be the derived schedule.
2. Let $m$ be the number of machines and $n_t$ be the number of jobs scheduled in interval $[t, t+1)$ for $t = 0, \dots, Z(\sigma) - 1$.
   **For** all $t$ such that $n_t > m$ **do:**
     Set $\tau := \lceil \frac{n_t}{m} \rceil$
```

Schedule n_t jobs in intervals $[t, t+1), [t+1, t+2), \dots, [t+\tau-1, t+\tau)$ such that each of the first $\tau - 1$ intervals contain m jobs.
For each job J_i with $C_i(\sigma) \geq t+1$ set $C_i(\sigma) = C_i(\sigma) + \tau$.

3. Stop.

Theorem 5. *Splitting Algorithm is a* $(3 - \frac{2}{m})$*-approximation algorithm for the* RP *problem with m identical parallel machines.*

Proof. Let σ be the schedule obtained by Splitting Algorithm. We say that time interval $[t, t+1)$ is *complete* if exactly m jobs are allocated within this interval in schedule σ, otherwise time interval $[t, t+1]$ is incomplete. Let h and l be the numbers of complete and incomplete intervals correspondingly, i.e. $Z(\sigma) = l + h$.

Note that after the splitting of "overloaded" unit time intervals, we get at most one new unit time interval that has less than m jobs. It follows that l does not exceed the length of schedule obtained by Rounding Algorithm. From Theorem 4 we have that $l \leq 2 \times \text{OPT}$. On the other hand, for any instance of the relocation problem with n UET jobs and m machines we get $\frac{hm+l}{m} \leq \text{OPT}$. The inequality follows from the fact that each complete interval contains exactly m jobs and each uncomplete interval contains at least one job. Combining the above facts, we thus get

$$\frac{Z(\sigma)}{\text{OPT}} \leq \frac{l+h}{\max\{\frac{l}{2}, \frac{hm+l}{m}\}}.$$

In the following, we show that $\frac{Z(\sigma)}{\text{OPT}}$ is no greater than $3 - \frac{2}{m}$.
Case 1: $l/2 \geq (hm+l)/m$

The assumption leads to $l \geq 2(hm+l)/m$. We have the following chain of derivations:

$$\begin{aligned}\frac{Z(\sigma)}{\text{OPT}} &\leq \frac{2(l+h)}{l} \\ &= 2 + \frac{2h}{l} \\ &\leq 2 + \frac{hm}{hm+l} \quad (\text{ because } l \geq 2(hm+l)/m) \\ &= 3 - \frac{l}{hm+l} \\ &\leq 3 - \frac{l}{\frac{lm}{2}} \quad (\text{ because } lm \geq 2(hm+l)) \\ &= 3 - \frac{2}{m}.\end{aligned}$$

Case 2: $l/2 < (hm+l)/m$

We have the following chain of derivations:

$$\begin{aligned}\frac{Z(\sigma)}{\text{OPT}} &\leq \frac{l+h}{\frac{hm+l}{m}} \\ &= \frac{m(l+h)}{hm+l} \\ &= 1 + \frac{lm-l}{hm+l} \\ &\leq 1 + \frac{lm-l}{\frac{lm}{2}} \\ &= 3 - \frac{2}{m}.\end{aligned}$$

From the above analysis, we complete the proof. □

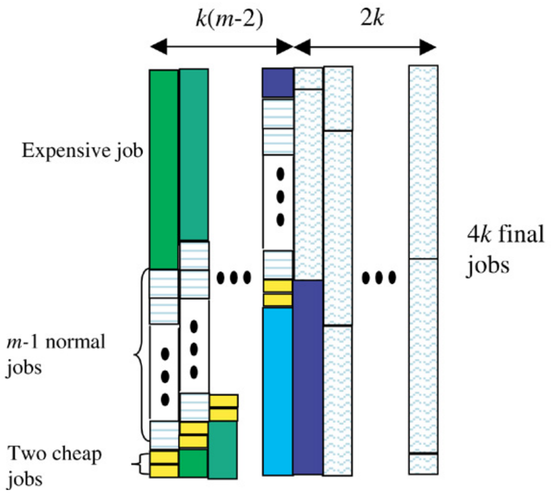

Fig. 3. Schedule produced by Greedy Algorithm 2.

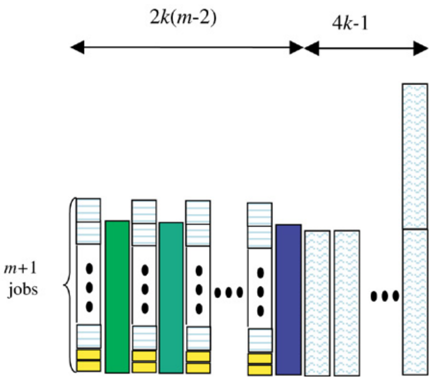

Fig. 4. Schedule obtained by Rounding Algorithm.

We finish our study on approximation algorithms with an instance of the relocation problem which shows that the performance ratios of Rounding Algorithm and Splitting Algorithm are tight. Consider sets of jobs, $N_i, i = 1, \ldots, k(m-2)+1$. Each of the first $k(m-2)$ sets contains $(m+2)$ jobs and the last set contains $4k$ jobs.

1. Each set $N_i, i = 1, \ldots, k(m-2)$, contains
   (a) Two "cheap" jobs with $\alpha_{ij} = 1/m^2$ for $i = 1, \ldots, k(m-2)$, and $j = 1, 2$.
   (b) $m-1$ "normal" jobs with $\alpha_{ij} = 1/(2(m-1))$ for $i = 1, \ldots, k(m-2)$ and $j = 3, \ldots, m+1$.
   (c) One "expensive" job with $\alpha_{im+2} = 1/2$ for $i = 1, \ldots, k(m-2)$.
2. All jobs of the last set $N_{k(m-2)+1}$ are the same and have $\alpha_{ij} = 1/2 - (m-2)/(2m^2)$ for $i = k(m-2)+1$, and $j = 1, \ldots, 4k$. We call these jobs as "final" jobs.

Let $\Omega_0 = 1$ and $\alpha_{ij} = \beta_{ij}$ for all jobs. The jobs are input in lexicographic order, i.e at first all jobs from set N_1 in the increasing order of indices, then all jobs from set N_2 and so on. We do not break ties in this case. Therefore, our approximation algorithm will process the jobs in accordance with the input sequence. In a certain way Greedy Algorithm 2 reports the worst schedule (Fig. 3) in this case. It is easy to check that Greedy Algorithm 2 obtains the schedule for the relaxed relocation problem and the makespan of this schedule is equal to km. Recall that this value is a lower bound for both of the two cases RP_∞ and RP. Rounding Algorithm then gets the schedule shown in Fig. 4 of the RP_∞ problem with makespan $2km-1$. Further, Splitting Algorithm outputs the schedule shown in Fig. 5 of the RP problem with makespan $k(3m-2)-1$.

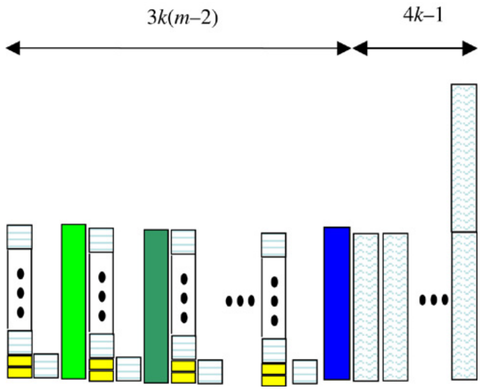

Fig. 5. Schedule obtained by Splitting Algorithm.

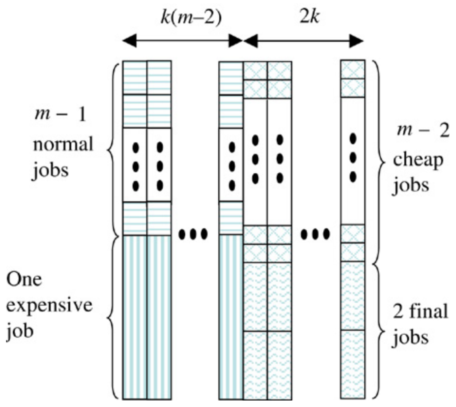

Fig. 6. Optimal schedule of the example instance for approximation algorithms.

The optimal schedule can be obtained in the following way (Fig. 6). We group $m - 1$ "normal" jobs and one "expensive" job in one unit-time interval. The execution of all such jobs requires $k(m - 2)$ time slots. Then, we schedule $m - 2$ "cheap" jobs with two "final" jobs in some time slots. Because we have $2k(m - 2)$ "cheap" jobs and $4k$ "final" jobs, we need $2k$ time slots to accommodate all these jobs. Finally, we have a schedule with makespan km, which matches the lower bound on the makespan of the instance. Choosing k as big as we want, we can show that the performance ratios of Rounding Algorithm and Splitting Algorithm are tight.

6. Conclusion

In this paper, we have considered a variant of the relocation problem where identical parallel machines are available. The problem was shown to be unary NP-hard even if all jobs have the same processing time and all jobs have positive contributions. For the case with an unbounded number of machines, we have given a negative result that there is no approximation algorithm having a performance ratio less than 4/3. On the other hand, we developed a 2-approximate algorithm for the case with an unbounded number of machines. Based upon this algorithm, a $(3-2/m)$-approximation algorithm for the case with a bounded number of machines has been proposed. An instance was given to show the tightness of the performance ratio of our proposed algorithm.

While the generic relocation problem without temporal considerations is equivalent to the two-machine flowshop scheduling problem, it is interesting to study other scheduling issues in the relocation problem incorporating temporal parameters and/or constraints. As an example, the relocation problem in two-machine flowshop (instead of two parallel machines) is of practical significance. The two machines can be interpreted as a working team for demolishing old buildings and the second team for constructing new buildings. In a sense, the second machine of the flowshop is designated for recycling resources and the resources of a job can be further utilized only if the recycling process of it is completed.

Acknowledgements

The authors are grateful to the anonymous referees for their patience in reviewing the earlier versions of this paper. Their comments have greatly improved the presentation. Especially, we highly appreciate one of the referees who identified several unclarified points and provided suggestions for reorganizing the structure of this paper.

The authors are partially supported by a Taiwan–Russia joint project under grant NSC-94-2416-H-009-013 (Taiwan) and RFBR RP05H01 (05-06-90606-HHCa, Russia).

References

[1] A. Amir, E.H. Kaplan, Relocation problems are hard, International Journal of Computer Mathematics 25 (1988) 101–110.
[2] J. Blazewicz, J.K. Lenstra, A.H.G. Rinnooy Kan, Scheduling subject to resource constraints: Classification and complexity, Discrete Applied Mathematics 5 (1983) 11–24.
[3] G.B. Dantzig, Discrete variable extremum problems, Operations Research 5 (1957) 266–277.
[4] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freedman, San Francisco, California, 1979.
[5] W. Fernandez de la Vega, G.S. Lueker, Bin packing can be solved within $1 + \varepsilon$ in linear time, Combinatorica 1 (1981) 349–355.
[6] P.L. Hammer, Scheduling under Resource Constraints — Deterministic Models, Annals of Operations Research, 7 Scheduling under resource constraints, J.C. Baltzer AG, Switzerland, 1986.
[7] S.M. Johnson, Optimal two- and three-stage production schedules with setup times included, Naval Research Logistics Quarterly 1 (1954) 61–67.
[8] E.H. Kaplan, Relocation models for public housing redevelopment programs, Planning and Design 13 (1986) 5–19.
[9] E.H. Kaplan, A. Amir, A fast feasibility test for relocation problems, European Journal of Operational Research 35 (1988) 201–205.
[10] E.H. Kaplan, O. Berman, Orient heights housing projects, Interfaces 18 (1988) 14–22.
[11] J.-X. Xie, Polynomial algorithms for single machine scheduling problems with financial constraints, Operations Research Letters 21 (1997) 39–42.
[12] PHRG, New Lives for Old Buildings: Revitalizing Public Housing Project, Public Housing Group, Department of Urban Studies and Planning, MIT, Cambridge, Massachusetts, 1986.