Journal of Global Optimization **27:** 293–311, 2003.
© 2003 *Kluwer Academic Publishers. Printed in the Netherlands.*

Single Machine Scheduling of Unit-time Jobs with Controllable Release Dates

T.C. EDWIN CHENG[1]⋆ and NATALIA V. SHAKHLEVICH[2]⋆⋆
[1]*Department of Management, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong (E-mail: mscheng@polyu.edu.hk).*
[2]*National Academy of Sciences of Belarus, Surganov Street 6, 220012 Minsk, Belarus (E-mail: shah@newman.bas-net.by)*

Abstract. The paper presents a bicriterion approach to solve the single-machine scheduling problem in which the job release dates can be compressed while incurring additional costs. The two criteria are the makespan and the compression cost. For the case of equal job processing times, an $O(n^4)$ algorithm is developed to construct integer Pareto optimal points. We discuss how the algorithm developed can be modified to construct an ε-approximation of noninteger Pareto optimal points. The complexity status of the problem with total weighted completion time criterion is also established.

Key words: Scheduling, deterministic, single machine, controllable release dates, multiple criteria

1. Introduction

We study a single-machine scheduling problem in which the jobs are ready for processing at their release dates, which can be varied within certain limits, with a view to constructing an optimal makespan schedule. In processing systems with controllable release dates, a lower and an upper bound of the release date are given for each job. The controllable release date is said to be "compressible" if the actual release date is modeled as a function of the amount of compression from its upper bound. Compressing the release dates may decrease the completion times of the jobs but incurs additional costs. The objective is to minimize the makespan of the schedule as well as the compression cost.

Scheduling problems with controllable release dates commonly arise in manufacturing systems where the preprocessing of the jobs depends on a common resource such as fuel, catalyzer, raw materials, etc. Real-life examples of such problems are given in Janiak (1986, 1991, 1998) in the context of steel production which involves preheating of iron ingots (see Williams (1986)).

⋆ This research was partially supported by The Hong Kong Polytechnic University under grant number PolyU 5245/99H.
⋆⋆ Supported in part by INTAS (Project INTAS-96-0820).

The scheduling problem can be formulated as follows. A set of jobs $N = \{1, \dots, n\}$ is to be processed on a single machine. The processing times p_i of the jobs are equal and, without loss of generality, we assume $p_i = 1$, $i = 1, \dots, n$. For every job i, $i \in N$, its normal (latest) release date $\overline{r}_i$ is given, which is the instant at which job i is ready for processing. Each normal release date $\overline{r}_i$ can be compressed by some amount x_i to the value $r_i = \overline{r}_i - x_i$ at a cost $b_i x_i$. A lower bound $\underline{r}_i$ is also given for the compressed release date r_i, i.e., $\underline{r}_i \leqslant r_i \leqslant \overline{r}_i$. For the sake of simplicity we normalize the data in such a way that we have unit processing times $p_i = 1$ but rational lower and upper bounds of the release dates $\underline{r}, \overline{r}_i, i = 1, \dots, n$. A schedule can be specified by the start times of the jobs T_i, or equivalently by the completion times of the jobs C_i, $i = 1, \dots, n$. The objective is to minimize the makespan of the schedule

$$\mathcal{C} = \mathcal{C}_{\max} = \max_{i \in N}\{C_i\},$$

together with the compression cost

$$\mathcal{K} = \sum_{i=1}^{n} b_i x_i.$$

We will consider the bicriterion problem P1 and two single-criterion problems P2 and P3, where one of the criteria is treated as a constraint. Using the three-field notation scheme introduced in Graham et al. (1979), we denote

– P1 by $1|p_i = 1, \overline{r}_i - x_i|\mathcal{C}_{\max}, \sum b_i x_i$,
– P2 by $1|p_i = 1, \overline{r}_i - x_i, \mathcal{C}_{\max} \leqslant C| \sum b_i x_i$ (C is a given constant),
– P3 by $1|p_i = 1, \overline{r}_i - x_i, \sum b_i x_i \leqslant K|\mathcal{C}_{\max}$ (K is a given constant).

We will indicate when necessary whether a restriction on integer lower and upper bounds of the release dates $\underline{r}_i, \overline{r}_i$ and compression amounts x_i is imposed.

Since the general case of the problem with arbitrary job processing times is strongly NP-hard (see Janiak (1991), Nowicki and Zdrzalka (1990)), the past research has been focused on constructing approximation algorithms for the general problem and exact algorithms for some special cases. Approximation algorithms are presented in Janiak (1991, 1998). The special cases investigated deal with equal upper bounds on the release dates ($\overline{r}_i = \overline{r}$, $i = 1, \dots, n$), arbitrary processing times, and common or proportionate compression cost functions (Cheng and Janiak (1994), Janiak (1986, 1991, 1998), Li (1994, 1995)). For these special cases, the efficient frontier for the bicriterion problem P1 can be constructed in $O(n^2)$ time (Cheng and Janiak (1994)), while problems P2 and P3 are solvable in $O(n \log n)$ time (Cheng and Janiak (1994); Li (1994)).

To the best of our knowledge, all results known for the problem with arbitrary (unequal) $\overline{r}_i$ are related to NP-hardness proofs (see Janiak (1991, 1998); Nowicki and Zdrzalka (1990)). In this paper, we extend the research on the problem with arbitrary $\overline{r}_i$ by presenting a number of polynomial time algorithms for the special case with equal job processing times. The importance of unit-time problems in multicriteria scheduling has been stressed in Chen and Bulfin (1990). This special case "approximates" the situation when job processing times differ by relatively small amounts. While job processing times are equal, their release dates and compression costs may be different: the jobs can be released by different suppliers which may have individual capacities and individual opportunities to speed up their processes.

The bicriterion problem P1 is the main subject of the present study. For this problem, we develop an $O(n^4)$ algorithm that constructs Pareto optimal schedules with integer compression amounts x_i and integer lower and upper bounds of the release dates $\underline{r}_i, \overline{r}_i, i = 1, \dots, n$ (see Section 2). Problems P1, P2, and P3 with noninteger $\underline{r}_i, \overline{r}_i$ and x_i are investigated in Sections 3-5. In Section 3, we solve problem P1 with noninteger compression amounts by presenting an $O(n^4\lceil B/\varepsilon\rceil)$ algorithm to construct an ε-approximation of Pareto schedules, where ε is a given accuracy and $B = \sum_{i=1}^n b_i$. In Section 4 we develop an $O(n^3)$ algorithm for problem P2 and in Section 5 we develop two algorithms for problem P3 for integer and noninteger parameters with complexities $O(n^3 \log n)$ and $O(n^3(\log n + \log B/\varepsilon))$, respectively. The special cases of P1, P2 and P3 with equal compression costs of the jobs $b_i = b, i = 1, \dots, n$, are considered in Section 6. The complexity of the problem with the total weighted completion time criterion $\sum w_i C_i$ is discussed in Section 7. Finally, some conclusions are presented in Section 8.

2. Problem P1 with integer $\overline{r}_i$, $\underline{r}_i$ and x_i

In this section, we show that the bicriterion problem $1|p_i = 1, \overline{r}_i - x_i|\mathcal{C}_{\max}, \sum b_i x_i$ with integer upper and lower bounds of the release dates $\underline{r}_i, \overline{r}_i$ and integer compression amounts x_i is solvable in $O(n^4)$ time by presenting an algorithm which determines all Pareto optimal points.

DEFINITION 1. A schedule s is Pareto optimal with respect to the functions $(\mathcal{C}, \mathcal{K}) = (\mathcal{C}_{\max}, \sum b_i x_i)$ if there does not exist another schedule s' with $\mathcal{C}(s') \leqslant \mathcal{C}(s)$, $\mathcal{K}(s') \leqslant \mathcal{K}(s)$, and at least one of these two inequalities is strict.

The Pareto optimal set $\mathcal{P}$ for the problem under consideration is a set of points in the $(\mathcal{C}, \mathcal{K})$-space with makespan values from the set $\{\underline{C}, \underline{C} + 1, \dots, \overline{C}\}$, where $\underline{C}$ ($\overline{C}$) is the makespan value which corresponds to the optimal solution of the problem $1|p_i = 1, \underline{r}_i|\mathcal{C}_{\max}$ ($1|p_i = 1, \overline{r}_i|\mathcal{C}_{\max}$, respectively). Since $\underline{C} \leqslant \underline{r}_{\max} + n$, $\overline{C} \geqslant \overline{r}_{\max} + 1$, where $\overline{r}_{\max} = \max_{i \in N}\{\overline{r}_i\}$, $\underline{r}_{\max} = \max_{i \in N}\{\underline{r}_i\}$, the number of Pareto optimal points is pseudopolynomial: it is at least equal to $(\overline{r}_{\max} + 1) - (\underline{r}_{\max} + n)$ and this bound is tight. In fact, there is no need to enumerate explicitely all integer Pareto optimal points. We will show that some of these points belong to linear segments in the $(\mathcal{C}, \mathcal{K})$-space which can be specified by their endpoints. The number of the endpoints is $O(n)$ and each adjacent breakpoint can be obtained from the previous one in $O(n^3)$ time, ensuring the time complexity $O(n^4)$ of our approach.

We will denote by $(\mathcal{C}^0, \mathcal{K}^0)$ the Pareto optimal point with makespan $\mathcal{C}^0 = \overline{C}$ and compression cost $\mathcal{K}^0 = 0$, and by $(\mathcal{C}^g = \mathcal{K}^g)$ the Pareto optimal point with makespan $\mathcal{C}^g = \mathcal{C}^0 - g$ and compression cost $\mathcal{K}^g > 0$, $g = 1, 2, \dots, \overline{C} - \underline{C}$.

2.1. THE FIRST n PARETO OPTIMAL POINTS

At first we recall that the optimal schedule for the single-machine problem with given release dates of the jobs can be obtained by sequencing jobs in nondecreasing order of the release dates. Hence, it is easy to construct an initial Pareto optimal point $(\mathcal{C}^0, \mathcal{K}^0) = (\overline{C}, 0)$ with uncompressed release dates $\overline{r}_i$ of the jobs as the solution of the problem $1|p_i = 1, \overline{r}_i|\mathcal{C}_{\max}$. The initial schedule s^0 may consist of several blocks with the jobs in each block being processed contiguously. The next Pareto optimal point $(\mathcal{C}^1, \mathcal{K}^1)$ with $\mathcal{C}^1 = \mathcal{C}^0 - 1$ can be constructed by means of compressing the release dates of the jobs from the last block by the optimal compression amounts $x_1^*, \dots, x_n^*$, which minimize the compression cost function $\mathcal{K} = \sum_{i=1}^n b_i x_i$. Proceeding with the next Pareto optimal point $(\mathcal{C}^g, \mathcal{K}^g)$, $g \geqslant 2$, a similar problem can be solved each time until there is no possibility of further shifting the last block due to the restrictions on the minimum release dates of the jobs $\underline{r}_i$.

To simplify our discussion, we first consider the case when the initial schedule s^0 consists of a single block of n jobs which is processed in the time interval $[T^0, \mathcal{C}^0]$, $T^0 = \mathcal{C}^0 - n$. Then each next Pareto optimal schedule s^g also consists of a single block. Otherwise the jobs from the first block can be started later without increasing the value of $\mathcal{C}^g$, but decreasing the compression cost $\mathcal{K}^g$.

We describe how a Pareto optimal schedule s^g, $0 \leqslant g \leqslant n-1$, which is optimal for the problem $1|p_i = 1, \overline{r}_i - x_i, \mathcal{C}_{\max} \leqslant \mathcal{C}^0 - g| \sum b_i x_i$ and consists of a single block can be modified to obtain the next schedule s^{g+1}, which is optimal for $1|p_i = 1, \overline{r}_i - x_i, \mathcal{C}_{\max} \leqslant \mathcal{C}^0 - (g+1)| \sum b_i x_i$. In what follows, we denote the start time of schedule s^g by T^g, its completion time by $\mathcal{C}^g$, $\mathcal{C}^g = T^g + n$. The jobs are processed within unit length time-slots of the form $[\tau_i, \tau_i + 1]$, $i = 1, \dots, n$. We will refer to such time-slots as to τ_i.

To transform schedule s^g into s^{g+1}, we construct a network with vertices $V = \{v_0, v_1, \dots, v_n\}$ and arcs $A = \{(v_i, v_j)|1 \leqslant i \leqslant n,\ 0 \leqslant j \leqslant n-1\}$. Vertices V correspond to time-slots $T^g - 1, T^g, \dots, \mathcal{C}^g - 1$, and we denote a time-slot corresponding to vertex v_i by τ_i. Each arc (v_i, v_h) represents the optimal transfer of the job which is processed in schedule s^g in time-slot τ_i (say, job j) to a time-slot τ_h corresponding to vertex v_h. The length of arc (v_i, v_h) is determined by the compression/decompression cost for this transfer and is given by the formula:

$$\varphi(v_i, v_h) = \begin{cases} b_j(\min\{\overline{r}_j, \tau_i\} - \min\{\overline{r}_j, \tau_h\}), & \text{if } \underline{r}_j \leqslant \tau_h, \\ \infty, & \text{otherwise.} \end{cases} \tag{1}$$

It is easy to see that any path from v_n to v_0 determines a chain of job transfers which leads to a schedule processed in $[T^{g+1}, \mathcal{C}^{g+1}] = [T^g - 1, \mathcal{C}^g - 1]$, and the length of this path is equal to the change in compression cost function.

THEOREM 1. *Given a Pareto optimal schedule s^g consisting of a single block and the corresponding network (V, A), the shortest path from v_n to v_0 determines the transformation of s^g into the next Pareto optimum schedule s^{g+1}.*

Proof. Let job $j \in N$ in schedule s^g be processed in time-slot τ_i corresponding to vertex v_i of network (V, A). Since there exist arcs from v_i to all vertices except for v_n, then network (V, A) specifies all feasible transfers of job j to unit time-slots from $[T^{g+1}, \mathcal{C}^{g+1}]$. An arbitrary path from v_n to v_0 specifies such job transfers that in the resulting schedule all jobs are processed in different unit time-slots from $[T^{g+1}, \mathcal{C}^{g+1}]$. Alternatively, for any schedule σ^{g+1} processed in $[T^{g+1}, \mathcal{C}^{g+1}]$, there exists a path in (V, A) which determines the transformation of the current schedule s^g into σ^{g+1} and the length of this path determines the cost of this transformation $\mathcal{K}(\sigma^{g+1}) - \mathcal{K}(s^g)$. Since schedule s^g is Pareto optimal, there does not exist a cycle of negative length in network (V, A). It is clear that if $\mathcal{K}(s^{g+1}) - \mathcal{K}(s^g) \leqslant \mathcal{K}(\sigma^{g+1}) - \mathcal{K}(s^g)$ for any schedule σ^{g+1} processed in $[T^{g+1}, \mathcal{C}^{g+1}]$, then s^{g+1} is a Pareto optimal schedule and it determines a Pareto optimal point $(\mathcal{C}^{g+1}, \mathcal{K}^{g+1})$ in the $(\mathcal{C}, \mathcal{K})$-space. □

Observe that if the shortest path from vertex v_n to v_0 is equal to infinity, then due to (1), there does not exist any feasible compression of the release dates which decreases $\mathcal{C}_{\max}$.

Based on Theorem 1, we can formulate an algorithm to construct the first n Pareto optimal points. The subsequent points will be constructed by a specialized faster algorithms described later.

```
ALGORITHM "First Points"
Input:  Pareto optimal schedule $s^0$ with jobs from $N$ processed contiguously in
        $[T^0, \mathcal{C}^0]$ in nondecreasing order of their uncompressed release dates $\overline{r}_i$
Output: a set of Pareto optimal schedules $s^g, g = 1, \dots, n$, and corresponding
        points $(\mathcal{C}^g, \mathcal{K}^g)$ in the $(\mathcal{C}, \mathcal{K})$-space
1.      $K^0 := 0$
2.      DO  $g = 0$ TO  $n - 1$
3.        For schedule $s^g$, construct network $(V, A)$ with $n + 1$ vertices and $n^2$ arcs.
          Calculate the arcs' weights by (1)
4.        Find the shortest path from $v_n$ to $v_0$
5.        IF the length of this path $\lambda$ is equal to $\infty$, THEN Stop
6.        ELSE construct schedule $s^{g+1}$ by modifying $s^g$ in accordance with the
          solution of the shortest path problem. Set $\mathcal{C}^{g+1} := \mathcal{C}^g - 1$, $K^{g+1} := K^g + \lambda$
      END
```

The correctness of algorithm "First Points" follows from Theorem 1. The time complexity of Step 3 is $O(n^2)$ since there are n^2 arcs and the complexity of calculating the length of an arc is $O(1)$. The shortest path problem can be solved in $O(n^3)$ time for the network with negative arc lengths and in presence cycles of nonnegative length (see Ahuja et al. (1993)), and hence the next Pareto optimal schedule s^{g+1} and corresponding point $(\mathcal{C}^{g+1}, \mathcal{K}^{g+1})$ can be obtained in $O(n^3)$ time.

Algorithm “First Points” terminates under one of the following conditions:

- either in Step 5, $\lambda = K(s^g) - K(s^{g+1}) = \infty$, i.e., the release dates of some jobs are compressed to their minimum values and the makespan of schedule s^g cannot be reduced,
- or the schedule s^n is obtained with all release dates compressed and this schedule corresponds to the Pareto optimal point $(\mathcal{C}^n, \mathcal{K}^n)$, $\mathcal{C}^n = \mathcal{C}^0 - n$.

Since the number of Pareto optimal points constructed by algorithm “First Points” is not larger than n, its overall time complexity is $O(n^4)$.

2.2. INTERMEDIATE AND LAST PARETO OPTIMAL POINTS

In this section, we consider the remaining Pareto optimal points after the first n points have been constructed. The remaining points are split into two groups: intermediate and last points. The *last points*, $(\mathcal{C}^g, \mathcal{K}^g)$, $g \in \{\overline{C} - \underline{C} - m + 1, \dots, \overline{C} - \underline{C}\}$, correspond to the Pareto optimal schedules in which at least one job starts at its minimum release date. The number of the last Pareto optimal points $m \leqslant n$. Indeed, if in schedule s^l job i starts at its minimum release date $\underline{r}_i$, then $\mathcal{C}^l \leqslant \underline{r}_i + n$, and there does not exist a schedule s^{l+n} with $\mathcal{C}^{l+n} \leqslant \underline{r}_i$. *Intermediate points* $(\mathcal{C}^g, \mathcal{K}^g)$, $g \in \{n+1, \dots, \overline{C} - \underline{C} - m\}$, correspond to the Pareto optimal schedules in which all jobs have compressed release dates and there is no job starting at its minimum release date.

Due to Theorem 1, the algorithm based on solving the shortest path problem can be applied to obtain not only the first n Pareto optimal points, but for all intermediate and last points. This can be made in $O(n^3(\overline{C} - \underline{C}))$ time. As we will show, the last points can be constructed by a faster $O(n^2)$ algorithm and the intermediate Pareto optimal points belong to a linear segment in the $(\mathcal{C}, \mathcal{K})$-space, which can be specified by its two endpoints.

First we characterize the structure of intermediate Pareto optimal schedules.

THEOREM 2. *If in schedule s^g, $g \geqslant n$, there is no job j starting at its minimum release date $\underline{r}_i$, then all jobs are sequenced in nondecreasing order of b_i.*

Proof. First we observe that in schedule s^g, $g \geqslant n$, all jobs have compressed release dates and their starting times are less than $\min_{i=1,\dots,n}\{\overline{r}_i\}$. Suppose Theorem 2 does not hold and in schedule s^g there is a pair of jobs i, j processed in two consecutive time slots $\tau, \tau + 1$, respectively, such that $b_i > b_j$. Then s^g is not

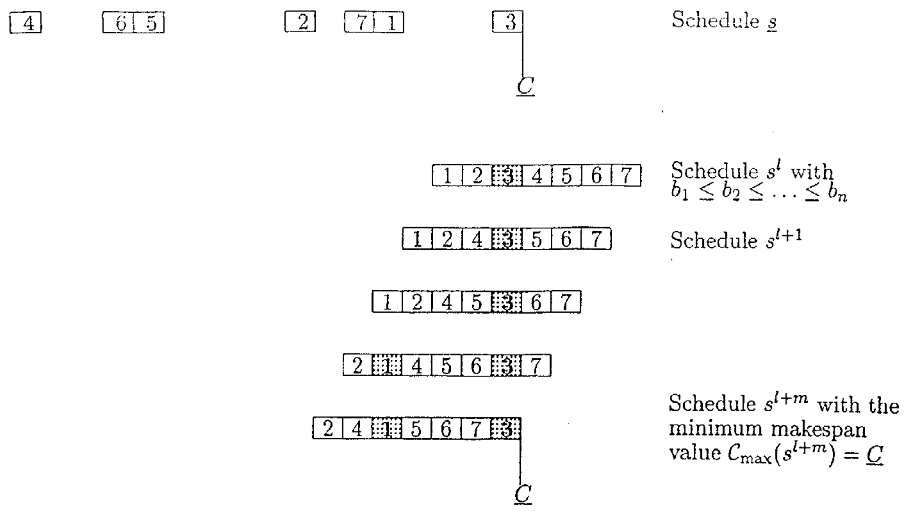
Pareto optimal schedule: interchanging jobs i and j does not violate the minimum release dates $\underline{r}_i, \underline{r}_j$ and changes the compression cost $\mathcal{K}$ by $b_j - b_i < 0$. □

It follows from Theorem 2 that in schedule s^n all jobs are sequenced in nondecreasing order of b_i. If in schedule s^g, $g \geqslant n$, there is no job starting at its minimum release date $\underline{r}_i$, then the job order in the next schedule s^{g+1} is the same as in s^g, i.e., s^{g+1} is obtained from s^g by shifting the whole schedule one unit earlier without changing job order. It means that the Pareto optimal points $(\mathcal{C}^g, \mathcal{K}^g)$, $g > n$, are obtained by shifting schedule s^n until at least one release date is compressed to its minimum value. Let s^l, $l > n$, be such a shifted schedule. Then all Pareto optimal points with $\mathcal{C} \in \{\mathcal{C}^l, \mathcal{C}^l + 1, \dots, \mathcal{C}^n\}$ belong to a linear segment in the $(\mathcal{C}, \mathcal{K})$-space, which connects point $(\mathcal{C}^l, \mathcal{K}^l)$ and point $(\mathcal{C}^n, \mathcal{K}^n)$. Hence to specify intermediate Pareto optimal points $(\mathcal{C}^g, \mathcal{K}^g)$, it suffices to specify the endpoints $(\mathcal{C}^n, \mathcal{K}^n)$ and $(\mathcal{C}^l, \mathcal{K}^l)$. It is clear that $(\mathcal{C}^l, \mathcal{K}^l)$ belongs to the subset of the last points, $l = \overline{C} - \underline{C} - m + 1$, and that the time complexity of constructing $(\mathcal{C}^l, \mathcal{K}^l)$ is $O(n)$.

Now we describe how the last Pareto optimal points $(\mathcal{C}^{l+1}, \mathcal{K}^{l+1}), \dots, (\mathcal{C}^{l+m}, \mathcal{K}^{l+m})$, $m \leqslant n$, can be obtained in $O(n^2)$ time. This algorithm first determines the minimum makespan value $\underline{C}$ and then constructs points $(\mathcal{C}^{l+1}, \mathcal{K}^{l+1}), \dots, (\mathcal{C}^{l+m}, \mathcal{K}^{l+m})$, $\mathcal{C}^{l+m} = \underline{C}$, in a straightforward way.

```
ALGORITHM "Last Points"
Input:    Pareto optimal schedule $s^l$ with jobs from $N$ processed contiguously
          in $[T^l, \mathcal{C}^l]$ and at least one job $i$ starting at its minimum release date
          $\underline{r}_i$; corresponding Pareto optimal point $(\mathcal{C}^l, \mathcal{K}^l)$ in the $(\mathcal{C}, \mathcal{K})$-space
Output:   a set of Pareto optimal schedules $s^g$, $g = l+1, \dots, l+m$, $m \leqslant n$,
          and points $(\mathcal{C}^g, \mathcal{K}^g)$ in the $(\mathcal{C}, \mathcal{K})$-space
1. Number the jobs in nondecreasing order of $b_i$ (this order corresponds to the job
   sequence in schedule $s^l$)
2. Determine the minimum makespan value $\underline{C}$ by constructing auxiliary schedule
   $\underline{s}$ with minimum release dates $\underline{r}_i$. The jobs in $\underline{s}$ are sequenced in nondecreasing
   order of $\underline{r}_i$
3. $g := l$, $\mathcal{K}^{g+1} := \mathcal{K}^g$
4. WHILE $\mathcal{C}^g > \underline{C}$ DO
5.    $\tau := T^g - 1$ (the first free time-slot)
6.    FOR $i = 1$ TO $n$
7.    Consider a job processed in time-slot $\tau_i$ (say, job $j$)
      IF $\underline{r}_j \leqslant \tau$, THEN move job $j$ from time-slot $\tau_i$ to time-slot $\tau$ and set
      $\mathcal{K}^{g+1} := \mathcal{K}^{g+1} + b_j(\min\{\tau_i, \overline{r}_j\} - \tau)$, $\tau := \tau_i$
      END FOR
8.    $C^{g+1} := \mathcal{C}^g - 1$,   $T^{g+1} := T^g - 1$,   $g := g + 1$,   $\mathcal{K}^{g+1} := \mathcal{K}^g$
   END WHILE
```

The working of the above algorithm is illustrated in Fig. 1.



Schedule $\underline{s}$
$\underline{C}$
Schedule s^l with $b_1 \leq b_2 \leq \ldots \leq b_n$
Schedule s^{l+1}
Schedule s^{l+m} with the minimum makespan value $C_{\max}(s^{l+m}) = \underline{C}$
$\underline{C}$

Figure 1. The working of Algorithm “Last Points”.

The correctness of algorithm “Last Points” follows from the fact that given Pareto optimal schedule s^g, it determines the next schedule s^{g+1} in full accordance with the algorithm based on solving the shortest path problem (see algorithm “First Points”). The complexity of constructing schedule $\underline{s}$ is $O(n \log n)$, while each schedule s^{g+1} is constructed from s^g, $g = l, \ldots, l+m-1$, in $O(n)$ time. Steps 4–8 are repeated no more than n times. So the overall complexity of constructing the last Pareto optimal points is $O(n^2)$. It is less than the complexity of constructing the first n Pareto optimal points as the algorithm deals with compressed jobs, which are sequenced in nondecreasing order of compression costs.

We have shown that if the first schedule s^0 consists of a single block of jobs, then constructing the first n Pareto optimal points takes $O(n^4)$ time, and constructing the last $m \leqslant n$ Pareto optimal points takes $O(n^2)$ time, while the intermediate Pareto optimal points belong to a linear segment. The overall complexity can be estimated as $O(n^4)$. We proceed now with the general case.

2.3. GENERAL CASE WITH INITIAL SCHEDULE s^0 CONSISTING OF SEVERAL BLOCKS

Suppose now that the initial schedule s^0 consists of several blocks of jobs. To construct the Pareto optimal point $(\mathcal{C}^{g+1}, \mathcal{K}^{g+1})$ from point $(\mathcal{C}^g, \mathcal{K}^g)$ which consists of several blocks of jobs, we consider the last block only and apply one of the following approaches.

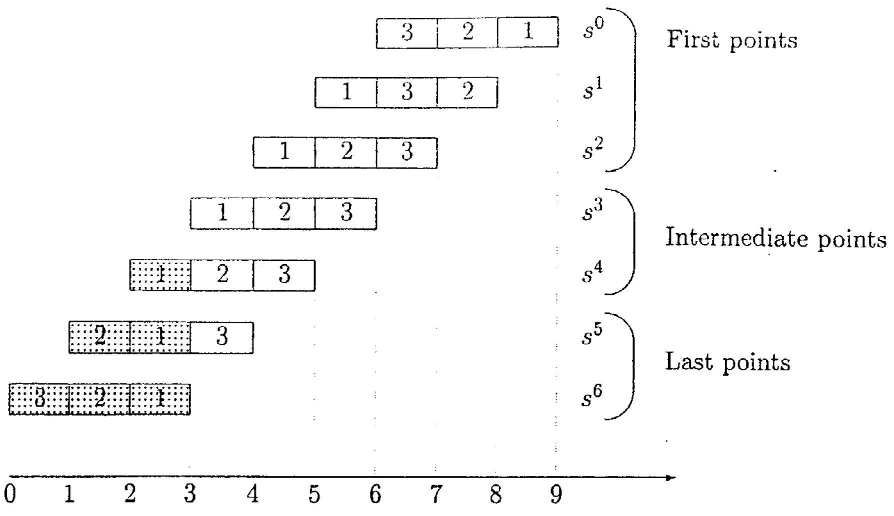


Figure 2. Pareto optimal schedules for Example 1.

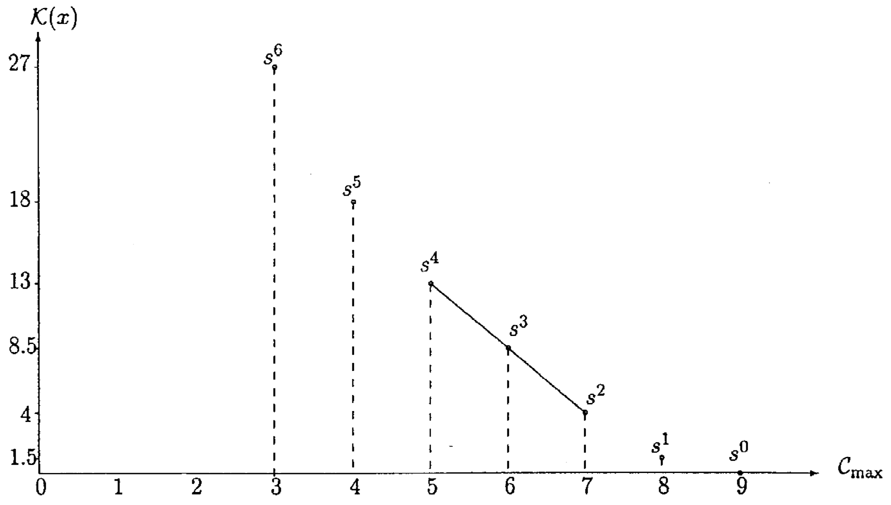


Figure 3. Pareto optimal points for Example 1.

i) If there are jobs in the last block with uncompressed release dates, then we use algorithm “First Points” . Note that network (V, A) should be constructed for the jobs from the last block.

ii) If all jobs in the last block have compressed release dates and no one job starts at its minimum release date, then we shift the last block to the previous one or until the release date of at least one job is compressed to its minimum value $\underline{r}_i$.

iii) If there is a job in the last block which starts at its minimum release date $\underline{r}_i$, then we apply algorithm “Last Points” to the last block.

Since the shortest path problem is solved no more than n times, shifting the last block is done also no more than n times, and algorithm "Last Points" is used only once, the total complexity of constructing all Pareto optimal points is $O(n^4)$.

2.4. EXAMPLES

We illustrate the approach described by the following example of the problem $1|p_i = 1, \overline{r}_i - x_i|\mathcal{C}_{\max}, \sum b_i x_i$.

EXAMPLE 1. Construct integer Pareto optimal schedules and corresponding Pareto optimal points in the $(\mathcal{C}, \mathcal{K})$-space for the problem with three jobs. The lower and upper bounds of the release dates and compression costs are given by the table:

i	1	2	3
$\overline{r}_i$	8	7	6
$\underline{r}_i$	2	1	0
b_i	0.5	1	3

The initial schedule s^0 consists of a single block processed in time interval $[6, 9]$. This schedule, together with subsequent Pareto optimal schedules $s^1, \ldots, s^6$, are given in Fig. 2. The corresponding Pareto optimal points are presented in Fig. 3. The first three schedules s^0, s^1, s^2 with $\mathcal{C}_{\max} \in \{7, 8, 9\}$ are obtained by solving three shortest path problems, the next two schedules s^3, s^4 with $\mathcal{C}_{\max} \in \{5, 6\}$ are obtained by shifting the whole block of jobs without changing their order $(1, 2, 3)$ until the starting time of job 1 becomes equal to $\underline{r}_1 = 2$. The last two schedules s^5, s^6 with $\mathcal{C}_{\max} \in \{3, 4\}$ are obtained by algorithm "Last Points" .

If the problem is not restricted to finding Pareto optimal schedules with integer compression amounts, then efficient schedules with noninteger starting times should be considered and the problem becomes much more complicated. The following example demonstrates that the job sequence of noninteger Pareto optimal schedule can differ from the job sequences of the two nearest integer Pareto optimal schedules.

EXAMPLE 2. Construct integer and noninteger Pareto optimal points for the problem with three jobs. The lower and upper bounds of the release dates and compression costs are given by the table:

i	1	2	3
$\overline{r}_i$	1	2	3
$\underline{r}_i$	0	0	0
b_i	1.8	10	1

This problem can be solved by enumerating different job sequences. The set of Pareto optimal points is given in Fig. 4 by a bold piecewise linear curve. The Pareto optimal schedule with $\mathcal{C}_{\max} = 3.8$ has the job sequence $(1, 3, 2)$ while the nearest integer points with $\mathcal{C}_{\max} = 3$ and $\mathcal{C}_{\max} = 4$ have job sequences $(3, 1, 2)$ and $(1, 2, 3)$, respectively.

In the next two sections, we investigate problems P1, P2, and P3 with arbitrary compression amounts starting first with problems P2 and P3 and using the results obtained to solve problem P1.

3. Problem P1 with arbitrary $\overline{r}_i$, $\underline{r}_i$ and x_i

In this section, we consider problem P1 with arbitrary compression amounts and describe how an ε-approximation (ε-kernel) of the set of Pareto optimal points $\mathcal{P}$ can be generated.

DEFINITION 2. For an arbitrary $\varepsilon > 0$, a set $\mathcal{P}_\varepsilon$ is called an ε-approximation of $\mathcal{P}$ if for each $(C, K) \in \mathcal{P}$ there exists $(C_\varepsilon, K_\varepsilon)$ such that $C_\varepsilon \leqslant C + \varepsilon$, $K_\varepsilon \leqslant K + \varepsilon$.

Consider first the problem under an assumption that the schedule s^0 consists of a single block. If (C^g, K^g) and (C^{g+1}, K^{g+1}) are two consecutive integer Pareto optimal points constructed by the approach described in Section 2, then $C^g - C^{g+1} = 1$ and $K^{g+1} - K^g \leqslant B$, where $B = \sum_{i=1}^n b_i$.

To construct ε-approximation of $\mathcal{P}$, we split each unit-line interval $[C^g, C^{g+1}]$ into subintervals of length ε/B such that for two consecutive points C' and C'', we have $C' - C'' = \varepsilon/B$ and $K'' - K' \leqslant \varepsilon$. We choose $\epsilon \leqslant \varepsilon/B$ such that $1/\epsilon$ is integer and construct Pareto optimal points with makespan values $C \in \bigcup_{k=0}^{1/\epsilon-1} \bigcup_{g=1}^{G} \{\overline{C} - k\epsilon - g\}$, where $G = \overline{C} - \underline{C} - 1$.

If $k = 0$, i.e., $C \in \{\overline{\overline{C}}, \overline{C} - 1, \dots, \overline{C} - g, \dots, \underline{C} + 1\}$, then we start with schedule s^0 with incompressed release dates and with makespan $\overline{C}$ and apply the $O(n^4)$ approach from Section 2.

If $k = 0$, i.e., $C \in \{\overline{C} - k\epsilon, \overline{C} - k\epsilon - 1, \dots, \overline{C} - k\epsilon - g, \dots, \underline{C} - k\epsilon + 1\}$, then we start with auxiliary schedule with makespan $C = \overline{C} + 1 - k\epsilon$ which can be obtained by shifting schedule s^0 by $1 - k\epsilon$ time units later. Then each next Pareto optimal schedule for $C = \overline{C} - k\epsilon - (g + 1)$ can be constructed from the previous one with $C = \overline{C} - k\epsilon - g$ in $O(n^3)$ time by solving the shortest path problem as in algorithm “First Points”. To construct the “last points”, we again can solve no more than n shortest path problems. Thus, the time complexity of constructing optimal schedules $C \in \{\overline{C} - k\epsilon, \dots, \underline{C} - k\epsilon + 1\}$ is $O(n^4)$. Since k takes $1/\epsilon$ different values, overall time complexity of constructing $\mathcal{P}_\varepsilon$ for problem P1 can be estimated as $O(n^4 \lceil B/\varepsilon \rceil)$.

If initial schedule s^0 consists of several blocks each block starting at its earliest time, then for fixed value k, $k = 0, \dots, 1/\epsilon 1$, we modify s^0 by starting each its block later at the nearest time $h - k\epsilon$, where h is integer. Then the distance between

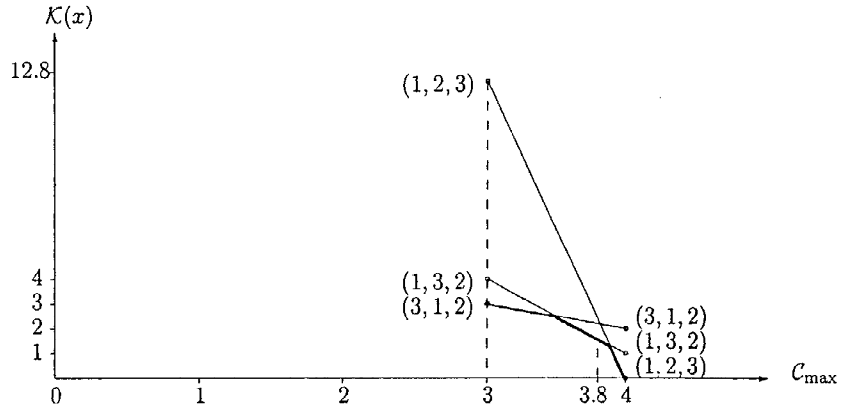


Figure 4. Pareto optimal points for Example 2.

two consecutive blocks is integer and the algorithm from Section 2.5 can be applied without increasing the total complexity $O(n^4\lceil B/\varepsilon\rceil)$.

4. Problems P2 with arbitrary $\overline{r}_i$, $\underline{r}_i$ and x_i

First we consider problem P2 with restricted makespan and with arbitrary (integer or noninteger) compression amounts x_i, i.e., $1|p_i = 1, \overline{r}_i - x_i, \mathcal{C}_{\max} \leqslant C| \sum b_i x_i$. Since function $\mathcal{K}$ is monotone, an optimal schedule for problem P2 completes exactly at C and starts at $T = C - n$, and problem P2 can be formulated as an assignment problem:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} z_{ij}, \\
\text{s.t.} \quad & \sum_{j=1}^{n} z_{ij} = 1, \quad i = 1, \dots, n, \\
& \sum_{i=1}^{n} z_{ij} \leqslant 1, \quad j = 1, \dots, n, \\
& z_{ij} \in \{0, 1\}, \quad i = 1, \dots, n, \quad j = 1, \dots, n,
\end{aligned}
\tag{2}
$$

where w_{ij} is the cost of assigning job i to time-slot $[T + j - 1, T + j]$:

$$
w_{ij} = \begin{cases} b_i \max\{\overline{r}_i - (T + j - 1), 0\}, & \text{if } T + j - 1 \geqslant \underline{r}_i, \\ \infty, & \text{otherwise.} \end{cases}
$$

It follows that problem P2 can be solved in $O(n^3)$ time for any C, integer or noninteger.

5. Problem P3 with $\overline{r}_i$, $\underline{r}_i$ and x_i

First we develop an algorithm to problem P3 with integer $\overline{r}_i, \underline{r}_i$ and x_i under an assumption that the initial schedule s^0 consists of a single block. As we have shown in Section 2, the algorithm can be easily modified if the initial schedule s^0 consists of several blocks without increasing its time complexity. Then we will present an approach to solve problem P3 with arbitrary $\overline{r}_i, \underline{r}_i$ and x_i.

```
ALGORITHM "Schedule with $\sum b_i x_i \leqslant K$"
Input:   Constant $K$
Output:  An optimal schedule $s^*$ for problem P3
1. Apply algorithm from the previous section to solve problem P2 with $C = \overline{C} - n$ and $C = \underline{C} + n$. Construct schedules $s^n$ and $s^l$ with integer compression amounts and $\mathcal{C}_{\max}(s^n) = \overline{C} - n$, $\mathcal{C}_{\max}(s^l) = \underline{C} + n$, determine $\mathcal{K}^n$ and $\mathcal{K}^l$.
2. IF $\mathcal{K}^n \leqslant K \leqslant \mathcal{K}^l$, then determine makespan $C^*$ of the optimal schedule $s^*$ by finding the point $(C^*, \mathcal{K}^*)$ in the segment with endpoints $(\mathcal{C}^n, \mathcal{K}^n)$ and $(\mathcal{C}^l, \mathcal{K}^l)$.
   Construct $s^*$ by the algorithm from the previous section with $C = C^*$.
3. ELSE apply binary search over $n$ first integer points or over $n$ last integer points to construct schedules $s^*$, with maximum value $K^*$ satisfying $\mathcal{K} \leqslant K$.
```

The correctness of algorithm "Schedule with $\sum b_i x_i \leqslant K$" follows from the properties of the Pareto optimal set for problem P1 with integer compression amounts. Steps 1–2 use the algorithm from the previous section to solve problem P2 and their time complexity is $O(n^3)$. In Step 3, binary search checks $O(\log\ n)$ K-values each of which is obtained by solving problem P2 on $O(n^3)$ time, i.e., total complexity of this step and of the whole algorithm is $O(n^3 \log\ n)$.

To solve problem P3 with arbitrary $\overline{r}_i, \underline{r}_i$ and x_i we first find integer solution $\mathcal{C}^*, \mathcal{K}^*$ as described above. IF $\mathcal{K}^n \leqslant K \leqslant \mathcal{K}^l$ then $\mathcal{C}^*, \mathcal{K}^*$ is an intermediate point in the segment with endpoints $\mathcal{C}^n, \mathcal{K}^n$ and $\mathcal{C}^l, \mathcal{K}^l$, and the corresponding C-value can be determined as

$$C = \mathcal{C}^l + (\mathcal{C}^n - \mathcal{C}^l)\frac{\mathcal{K}^l - K}{\mathcal{K}^l - \mathcal{K}^n}.$$

Otherwise since function $\mathcal{K}$ decreases monotonely in $[C^* - 1, C^*]$ and the maximum decrease is equal to $B = \sum_{i=1}^n b_i$, an optimal value of $C_{\max}$ can be found by a binary search procedure in that interval. If ε is a given error bound, then interval $[C^* - 1, C^*]$ can be split into B/ε equal parts such that for two consecutive points C' and C'', $C' - C'' = \varepsilon/B$, and the change in the K-value $K'' - K'$ is not larger than ε. It means that the binary search procedure checks $O(\log B/\varepsilon)$ values of K, each of which is obtained by solving problem P2 in $O(n^3)$ time, i.e., problem P3 can be solved in $O(n^3(\log n + \log B/\varepsilon))$ time.

6. Unit Compression Costs

In this section, we discuss how the algorithms developed in the previous sections can be simplified for the problem with equal compression costs ($b_i = b, \ i = 1, \dots, n$).

6.1. PROBLEM P1

In the bicriterion problem, we may assume without loss of generality that the compression costs are equal to 1, i.e., to consider the problem $1|p_i = 1, \overline{r}_i - x_i|\mathcal{C}_{\max}, \sum x_i$.

THEOREM 3. *If s^g is a Pareto optimal schedule and there are two jobs i and j, $\overline{r}_i < \overline{r}_j$, processed in time-slots τ_1, τ_2, such that* $\max\{\underline{r}_i, \underline{r}_j\} \leqslant \tau_1 < \tau_2$, *then job i is processed in τ_1 and job j is processed in τ_2.*

Proof. Suppose in schedule s^g job i is processed in time-slot τ_2 and job j is processed in time-slot τ_1. Then interchanging jobs i and j leads to a modified schedule $\tilde{s}^g$ such that

$$\mathcal{K}(\tilde{s}^g) - \mathcal{K}(s^g) = \begin{cases} (\min\{\overline{r}_i, \tau_2\} - \tau_1) - (\min\{\overline{r}_j, \tau_2\} - \tau_1) < 0, & \text{if } \overline{r}_j > \tau_1, \\ 0, & \text{otherwise, i.e., if } \overline{r}_i < \overline{r}_j \leqslant \tau_1. \end{cases}$$

□

If in the initial schedule s^0 all jobs are processed contiguously in one block, then the algorithm constructing the first n Pareto optimal points is similar to steps 5–8 of algorithm “Last Points” . It is easy to check that the correctness of this approach follows from Theorem 3. Since there are no more than n first points and each of them is obtained in $O(n)$ time, all first points can be constructed in $O(n^2)$ time and this is also the time complexity of constructing the last points. Thus the overall complexity of solving problem P1 with integer compression amounts is $O(n^2)$.

Theorem 3 holds for arbitrary (not necessarily integer) starting times. It means that if T^g, T^{g+1} are starting times of two Pareto optimal schedules s^g and s^{g+1} with integer starting times, then any schedule with noninteger starting time T, $T^{g+1} < T < T^g$, has either the same job sequence as s^g, or the same as s^{g+1}. In this case the efficient frontier for all schedules with starting times from $[T^{g+1}, T^g]$ is a segment with endpoints $(\mathcal{C}^g, \mathcal{K}^g)$, $(\mathcal{C}^{g+1}, \mathcal{K}^{g+1})$ if the job sequences in s^g and s^{g+1} are the same, or it consists of two line segments, otherwise: the slope of the line segments starting in $(\mathcal{C}^g, \mathcal{K}^g)$ is given by the total number of the jobs whose release dates are being compressed while shifting schedule s^g earlier, and the slope of the line segments starting in (C^{g+1}, K^{g+1}) is given by the total number of jobs whose release dates are being decompressed while shifting schedule s^{g+1} later.

The arguments from the previous section justify that the complexity bound $O(n^2)$ holds for the more general cases when the initial schedule s^0 consists of several blocks,

6.2. PROBLEMS P2 AND P3

First we consider the problem with makespan as a constraint: $1|p_i = 1, \overline{r}_i - x_i, \mathcal{C}_{\max} \leqslant C| \sum x_i$. It is easy to check that Theorem 3 holds for problem P2, and the the algorithm can be formulated as follows.

```
    ALGORITHM "Schedule with $\mathcal{C}_{\max} \leqslant C$"
    Input:   Constant $C$
    Output:  An optimal schedule for problem P2
1.  Number the jobs such that $\overline{r}_1 \leqslant \cdots \leqslant \overline{r}_n$.
2.  FOR $i = 1$ TO $n$
3.      Assign job $i$ to the first available time-slot $\tau$ which satisfies: $\tau \geqslant \underline{r}_i, \tau \in [C - n, C - 1]$
    END FOR
    The complexity of algorithm "Schedule with $\mathcal{C}_{\max} \leqslant C$" is $O(n \log n)$.
```

To solve the inverse problem P3, $1|p_i = 1, \overline{r}_i - x_i, \sum x_i \leqslant K|\mathcal{C}_{\max}$, with compression cost as a constraint, we again can avoid constructing the whole set of Pareto optimal points. It can be done in $O(n \log n)$ time by modifying Algorithm "Schedule with $\sum b_i x_i \leqslant K$" from Section 5. Steps 1–2 of the modified algorithm use the $O(n \log n)$ algorithm "Schedule with $\sum_{\max} \leqslant C$" to solve problem P2 with $C = \overline{C} + n$ and $C = \underline{C} + n$. Step 3 constructs two integer schedules s^g, s^{g+1} such that $\mathcal{K}(s^g) \leqslant K \leqslant \mathcal{K}(s^{g+1})$ using binary search over n first integer points or over n last integer points. As optimal schedule s^* is obtained by left shifting schedule s^g or by right shifting schedule s^{g+1}. Since problem P2 is solved $O(\log n)$ times in the binary search procedure, the complexity of Step 3 is $O(n(\log n)^2)$, and this is the overall complexity to solve problem P3.

7. Minsum Criteria

In this section, we prove that problem P_3 with the total weighted completion time criterion is NP-hard in the ordinary sense.

THEOREM 4. *The problem* $1|p_i = 1, \overline{r}_i - x_i, \sum b_i x_i \leqslant K| \sum w_i C_i$ *is* NP*-hard in the ordinary sense.*

Proof. We construct a reduction from the PARTITION problem which is known to be NP-complete: given z different positive integers $e_1, \ldots, e_z$ and $E = (1/2) \sum_{i=1}^{z} e_i$, do there exist two disjoint subsets A_1 and A_2 such that $\sum_{i \in A_1} e_i = \sum_{i \in A_2} e_i = E$? The reduction is based on the following instance of the problem $1|p_i = 1, \overline{r}_i - x_i, \sum b_i x_i \leqslant K| \sum w_i C_i$ with $n = 2z$ jobs. The release dates of the first z jobs are incompressible and they are given by $\overline{r}_i = \underline{r}_i = iE$, $i = 1, \ldots, z$. The release dates of the next z jobs can be compressed and they are given by $\overline{r}_{z+i} = iE + 1$, $\underline{r}_{z+i} = iE - 1$, $i = 1, \ldots, z$. Job weights and compression costs are determined as $b_i = e_i$, $w_i = e_i$, $b_{z+i} = e_i$, $w_{z+i} = e_i$, $i = 1, \ldots, z$.

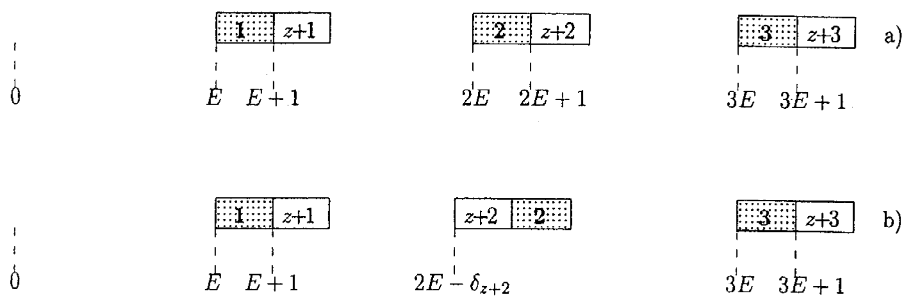


Figure 5. Fragment of a schedule with a) incompressed or b) compressed release dates.

We show that there exists a schedule s^* with

$$\begin{aligned}\sum_{i=1}^{n} w_i C_i &\leqslant D-2E,\\ \sum_{i=1}^{n} b_i x_i &\leqslant 2E\end{aligned} \tag{3}$$

if and only if the PARTITION problem has a solution. Here D is a constant given by $D=\sum_{i=1}^{z} e_i(2iE+3)$.

First we analyze a schedule with the minimum total weighted completion time if all release dates are uncompressed. In such a schedule, all jobs start at their maximum release dates (see Fig. 5,a). We have: $\sum_{i=1}^{n} w_i C_i = D,\ \sum_{i=1}^{n} b_i x_i = 0$.

Suppose first that schedule s^* satisfying (3) exists. Then the release dates of some jobs from $\{z+1,\dots,2z\}$ have to be compressed (otherwise we have a schedule with uncompressed release dates and $\sum_{i=1}^{n} w_i C_i = D$ violating (3)). If job $i \in \{z+1,\dots,2z\}$ has compressed release date, then its compression amount can be represented in a form $x_i = 1+\delta_i$, where $0<\delta_i \leqslant 1$ because smaller compression amount $x_i \leqslant 1$ does not decrease $\sum w_i C_i$, but increases $\sum b_i x_i$. A fragment of such a schedule with release date of job $z+2$ compressed by $x_{z+2} = 1+\delta_{z+2}$ is presented in Fig. 5(b).

Let the set of jobs with compressed release dates in schedule s^* be I. First we show that $\delta_i = 1$, $i \in I$. Indeed,

$$\begin{aligned}\sum_{i=1}^{n} w_i C_i &= D-2\sum_{i\in I} e_i\delta_i,\\ \sum_{i=1}^{n} b_i x_i &= \sum_{i\in I} e_i(1+\delta_i).\end{aligned}$$

Since schedule s^* satisfies conditions (3),

$$\begin{aligned}-2\sum_{i\in I} e_i\delta_i &\leqslant -2E,\\ \sum_{i\in I} e_i(1+\delta_i) &\leqslant 2E.\end{aligned} \tag{4}$$

Summing the last two inequalities, we obtain: $\sum_{i\in I} e_i(1-\delta_i) \leqslant 0$, which implies that $\delta_i = 1, i \in I$. Substituting $\delta_i = 1$ in (4), we obtain:

$$\begin{aligned} -\sum_{i\in I} e_i &\leqslant -E, \\ \sum_{i\in I} e_i &\leqslant \quad E. \end{aligned}$$

It means that I forms a solution to the PARTITION problem.

Suppose now that sets A_1 and A_2 form a solution to the PARTITION problem. Then for $i \in A_1$, we compress the release dates of the jobs $z+i$ by 2 and construct schedule s^* with

$$\begin{aligned} &\sum_{i=1}^{n} w_i C_i = D - 2\sum_{i\in A_i} e_i = D - 2E, \\ &\sum_{i=1}^{n} b_i x_i = 2\sum_{i\in A_i} e_i = 2E, \end{aligned}$$

This completes the proof of Theorem 4. □

It follows from the proof of Theorem 4 that problems P1 and P3 with total weighted completion time criterion are also NP-hard.

8. Conclusions

In this paper, we have investigated the single-machine scheduling problem with unit-time operations and controllable release dates which is a special case of the strongly NP-hard problem with arbitrary processing times. The results obtained are summarized in Table 1. In this table, the special case with integer compression amounts is denoted by $[x_i]$ and the general case with arbitrary compression amounts is denoted by x_i.

Problem P1 can be solved in $O(n^4)$ time if compression amounts are integer or in $O(n^4\lceil B/\varepsilon\rceil)$ time if compression amounts are arbitrary. Problem P2 can be solved in $O(n^3)$ time in case of integer or arbitrary compression amounts. Problem P3 can be solved in $O(n \log n)$ and $O(n^3(\log n + \log B/\varepsilon))$ time in case of integer or arbitrary compression amounts, respectively. If the compression costs are equal, then the time complexity of solving problem P1 reduces to $O(n^2)$ and to $O(n \log n)$ for problems P2 and P3.

Observe that the algorithm developed for problem P1 with arbitrary compression is not polynomial. In fact other known ε-approximation algorithms for bicriterion scheduling jobs with controllable processing times $p_i - y_i, 0 \leqslant y_i \leqslant \overline{y}, i = 1, \ldots, n$ have similar nonpolymonial complexities:

Table 1. The complexity of the unit-time problems with controllable release dates

	$\mathcal{C}_{\max}, \sum x_i$		$\mathcal{C}_{\max}, \sum b_i x_i$		$\sum C_i, \sum x_i$
	$[x_i]$	x_i	$[x_i]$	x_i	
Bicriterion problem P_1	$O(n^2)$	$O(n^2)$	$O(n^4)$	$O(n^4 \lceil B/\varepsilon)$	NP-hard
Single criterion problem P_2 with $\mathcal{C}_{\max}$ as constraint	$O(n \log n)$	$O(n \log n)$	$O(n^3)$	$O(n^3)$	NP-hard
Single criterion problem P_3 with $\mathcal{K}$ as constraint	$O(n \log n)$	$O(n(\log n)^2)$	$O(n^3 \log n)$	$O(n^3(\log n + \log B/\varepsilon))$	NP-hard

- Tuzilov (1984) suggested an $O(n^2 G/\varepsilon)$ algorithm to solve problem $1|p_i - y_i| f_{\max}, \sum b_i y_i$, where $f_{\max}(y_1, \dots, y_n) = \max_{i=1,\dots,n} f_i(C_i, y_1, \dots, y_n)$, f_i is an arbitrary nondecreasing function depending on completion time of job i and compression amounts $y_i, \dots, y_n$, and $G = f_{\max}(0, \dots, 0) - f_{\max}(\overline{y}_1, \dots, \overline{y}_n)$;
- Nowicki and Zdrzalka (1985) developed $O(n \log n + \lfloor H/\varepsilon \rfloor nm)$ algorithm to solve the bicriterion problem $Q|Pmtn, p_i - y_i|\mathcal{C}_{\max}, \sum b_i y_i$ of preemptive scheduling n jobs by m uniform machines, where $H = \overline{C} - \underline{C}$, $\overline{C}$ is makespan of the optimal schedule with incompressed processing times p_i and $\underline{C}$ is makespan of the optimal schedule with fully compressed processed times $p_i - \overline{y}_i, i = 1, \dots, n$.

Problems P1, P2, and P3 with the total weighted completion time criterion $\sum w_i C_i$ are NP-hard. An interesting direction for future research is investigating the problems with the minsum criterion with either equal job weights $w_i = w, i = 1, \dots, n$, or equal compression costs $b_i = b, i = 1, \dots, n$.

Acknowledgements

The authors are grateful to the anonymous referees whose comments have contributed to improving the paper.

References

1. Ahuja, R.K., Magnanti, T.L. and Orlin J.B. (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, New Jersey.
2. Chen, C.L. and Bulfin, R.L. (1990), Scheduling Unit Processing Time Jobs on a Single Machine with Multiple Criteria, *Computers and Operations Research* 17, 1–7.
3. Cheng, T.C.E. and Janiak, A. (1994), Resource Optimal Control in Single Machine Scheduling with a Completion Time Constraint, *IEEE Transactions on Automatic Control* 39, 1243–1246.

4. Graham, R.L., Lawler, E.L., Lenstra J.K. and Rinnooy Kan, A.H.G. (1979), Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey, *Annals of Discrete Mathematics* 5, 287–326.
5. Horn, W.A. (1974), Some Simple Scheduling Algorithms, *Naval Research Logistics Quarterly* 21, 177–185.
6. Janiak, A. (1986), Time-Optimal Control in a Single Machine Problem with Resource Constraints, *Automatica* 22, 745–747.
7. Janiak, A. (1991), Single Machine Scheduling Problem with a Common Deadline and Resource Dependent Release Dates, *European Journal of Operational Research* 53, 317–325.
8. Janiak, A. (1998), Single Machine Sequencing with Linear Models of Release Dates, *Naval Research Logistics* 45, 99–113.
9. Li, C.L. (1994), Scheduling with Resource-Dependent Release Dates – a Comparison of Two Different Resource Consumption Functions, *Naval Research Logistics* 41, 807–819.
10. Li, C.L. (1995), Scheduling to Minimize Resource Consumption with a Constraint on the Sum of Completion Times, *European Journal of Operational Research* 80, 381–388.
11. Nowicki, E. and Zdrzalka, S. (1990), A Survey of Results for Sequencing Problems with Controllable Processing Times, *Discrete Applied Mathematics* 26, 271–287.
12. Nowicki, E. and Zdrzalka, S. (1995), A Bicriterion Approach to Preemptive Scheduling of Parallel Machines with Controllable Job Processing Times, *Discrete Applied Mathematics* 63, 237–256.
13. Tuzikov, A.V. (1984), On Two-Criterion Scheduling Problem with Controllable Processing Times, *USSR Computational Mathematics and Mathematical Physics* 24, 191-194.
14. Williams, T.J. (1986), *Analysis and Design of Hierarchical Control Systems: with Special Reference of Steel Plant Operations*, North-Holland, Amsterdam.