On Scheduling a Single Machine to Minimize a Piecewise Linear Objective Function : A Compact MIP Formulation

Philippe Baptiste* Ruslan Sadykov*

July 23, 2007

Abstract

We study the scheduling situation in which a set of jobs subjected to release dates and deadlines are to be performed on a single machine. The objective is to minimize some piecewise linear objective function $\sum_j F_j$ where $F_j(C_j)$ corresponds to the cost of the completion of job J_j at time C_j. This class of function is very large and thus interesting both from a theoretical and practical point of view: It can be used to model total (weighted) completion time, total (weighted) tardiness, earliness and tardiness, *etc.* We introduce a new MIP formulation based on time interval decomposition. Our MIP is closely related to the well-known time-indexed MIP formulation but uses much less variables and constraints. This allows us to solve medium size instances of the problem.

Keywords: Mixed Integer Program, Scheduling, Earliness, Tardiness

1 Introduction

We study the scheduling situation in which a set N of jobs $\{1, 2, \ldots, n\}$ have to to be processed without preemption on a single machine. Each job $j \in N$ has a release date r_j, a positive processing time $p_j > 0$ and a deadline d_j. For each job j, we also have a cost function F_j which is a piecewise linear function of the completion time C_j of j. The objective is to minimize the overall cost $\sum_j F_j(C_j)$. This class of function is very large and thus interesting both from a theoretical and practical point of view: It can be used to model total (weighted) completion time, total (weighted) tardiness, earliness and tardiness, *etc.*

We first introduce some basic notation for the problem. Let $T = \max_{j \in N} d_j$ denote the time horizon of the problem and, without any loss of generality, we assume that there is a partition of the interval $(0, T]$ into a set $M = \{1, \ldots, m\}$ of intervals $I_u = (e_{u-1}, e_u]$ (for $u \in M$) such that

*LIX, UMR CNRS 7161, École Polytechnique, F-91128 Palaiseau, Philippe.Baptiste@polytechnique.fr

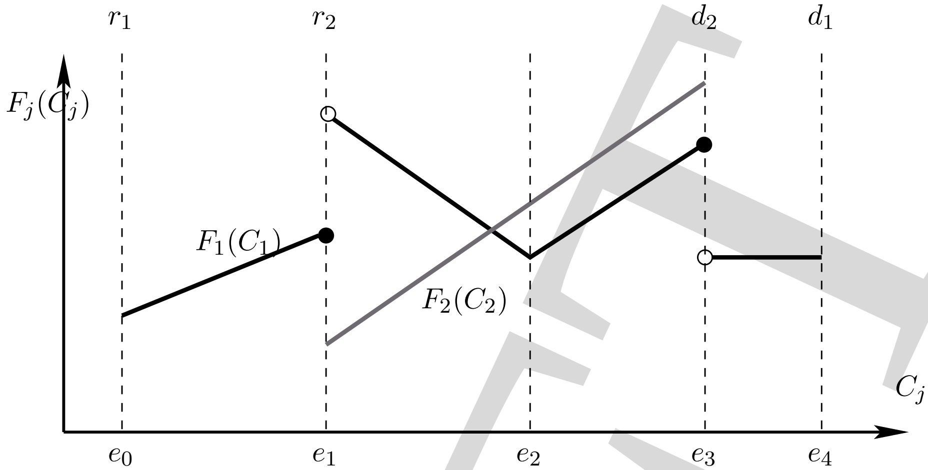


Figure 1: Linear partition of the time horizon — an example with 2 jobs

- the cost function of any job j over any interval I_u is linear, *i.e.*,

$$F_j(C_j) = f_j^u + w_j^u \cdot (C_j - e_{u-1}), \quad C_j \in I_u,\ u \in M,\ j \in N.$$

  where f_j^u, w_j^u are some constant values,

- for every job $j \in N$, $r_j = e_v$ and $d_j = e_u$ for some $v, u \in M$.

We will call a partition with such a property *linear*. See an example of such a partition in Figure 1.

The major contribution of this paper is to introduce a new MIP (Section 4) for the single machine problem. It is based on basic properties (introduced in Section 3 of linear partitions. This MIP is closely related to time-indexed MIPs (see Section 2) but it uses much less variables and constraints. A more efficient (and more complex) variant of the MIP is described in Section 5. Experimental results are reported in Section 6.

2 Literature Review

A huge amount of research has been carried on some single machine "total cost" scheduling problems. However, most of the papers are dedicated to special cases and there are few results on generic objective functions. Surprisingly, the combination of time windows (release dates and deadlines) together with a sum objective function is almost never considered in the literature.

To formulate the objective function, we introduce the lateness $L_j = C_j - d_j$, the tardiness $T_j = \max\{0, L_j\}$, the earliness $E_j = \max(0, d_j - C_j)$ and the unit penalty U_j, where $U_j = 0$ if $C_j \leq d_j$ and $U_j = 1$ otherwise. The objective functions (depicted in Figure 2) to be minimized are defined as follows:

- The *Makespan* $C_{\max} = \max_j C_j$,

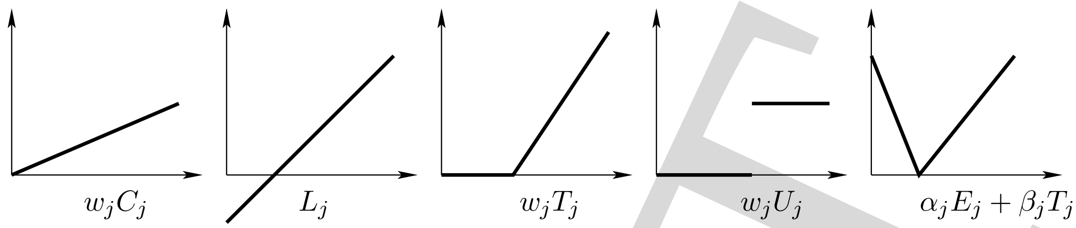


Figure 2: Classical objective functions — weighted completion time, lateness, weighted tardiness, weighted number of late jobs, weighted earliness-tardiness

- the *Maximum Lateness* $L_{\max} = \max_j L_j$,
- the *Maximum Tardiness* $T_{\max} = \max_j T_j$,
- the *Total Weighted Completion Time* $\sum w_jC_j$,
- the *Total Weighted Tardiness* $\sum w_jT_j$,
- the *Total Weighted Number of Tardy Jobs* $\sum w_jU_j$.
- the *Earliness-Tardiness* $\sum \alpha_jE_j + \beta_jT_j$.

Weights can be all equal to 1 and in this case, w_j is dropped in the above notation.

2.1 Specific Scheduling Algorithms

In this section, it is always assumed that we do not have deadlines. A lot of research has been carried on the unweighted total tardiness problem with no release date. Powerful dominance rules have been introduced by Emmons [30]. Lawler [36] has proposed a dynamic programming algorithm that solves the problem in pseudo-polynomial time. Finally, Du and Leung have shown that the problem is NP-Hard [28]. Most of the exact methods for solving the total tardiness problem strongly rely on Emmons' dominance rules. Potts and Van Wassenhove [42], Chang *et al.*[19] and Szwarc *et al.*[51], have developed Branch and Bound methods using the Emmons rules coupled with the decomposition rule of Lawler [36] together with some other elimination rules. The best results have been obtained by Szwarc, Della Croce and Grosso [51, 52] with a Branch and Bound method that efficiently handles instances with up to 500 jobs. The total weighted tardiness problem ($\sum w_iT_i$) is strongly NP-Hard [36]. For this problem, Rinnooy Kan *et al.*[47] and Rachamadugu [43] have extended the Emmons Rules [30]. Exact approaches based on Dynamic Programing and Branch and Bound have been tested and compared by Abdul-Razacq, Potts and Van Wassenhove [2]. Recently, Pan and Shi [40] have proposed a very efficient branch-and-bound algorithm which solves instances of the problem $1 \,||\, \sum w_jT_j$ with up to 100 jobs.

There are less results on the total tardiness problem with arbitrary release dates. Chu and Portmann [23] have introduced a sufficient condition for local optimality which allows them to build a dominant subset of schedules. Chu [21] has also proposed a Branch and Bound method using efficient dominance rules. This method handles instances with up to 30 jobs for the hardest instances and with up to 230 jobs for the easiest ones. More recently, Baptiste, Carlier and Jouglet [7] have described a new lower bound and some dominance rules which are used in a Branch and Bound procedure which handles instances with up to 50 jobs for the hardest instances and 500 jobs for the easiest ones. Let us also mention that exact Branch and Bound procedures have been proposed for the same problem with setup times [44, 50]. For the total weighted tardiness problem ($\sum w_i T_i$) with release dates, Akturk and Ozdemir [4] have proposed a sufficient condition for local optimality which improves heuristic algorithms. This rule is then used with a generalization of Chu's dominance rules to the weighted case in a Branch and Bound algorithm [3]. This Branch and Bound method handles instances with up to 20 jobs. Recently Jouglet et al. [33] have proposed a new Branch and Bound that solves all instances with up to 35 jobs.

For the total completion time problem ($\sum w_i C_i$), in the case of identical release dates, both the unweighted and the weighted problems can easily be solved polynomially in $O(n \log n)$ by applying the Shortest Weighted Processing Time priority rule, also called Smith's rule [49]. For the unweighted problem with release dates, several researchers have introduced dominance properties and proposed a number of algorithms [18, 27, 26]. Chu [20, 22] has proved several dominance properties and has provided a Branch and Bound algorithm. Chand, Traub and Uzsoy used a decomposition approach to improve Branch and Bound algorithms [17]. Among the exact methods, the most efficient algorithms [20, 17] can handle instances with up to 100 jobs. The weighted case with release dates is NP-Hard in the strong sense [46] even when the preemption is allowed [35]. Several dominance rules and Branch and Bound algorithms have been proposed [11, 12, 32, 45]. To our knowledge, the best results are obtained by Pan and Shi [41] with a hybrid Branch and Bound-Dynamic Programming algorithm which has been tested on instances involving up to 200 jobs.

Many exact methods have been proposed for the problem of minimizing the number of late jobs ($\sum U_i$) [8, 25, 9]. More recently, Ruslan Sadykov [48] and M'Hallah and Bulfin [39] have proposed efficient exact algorithms for solving the general case of this problem: $1 \mid r_j \mid \sum w_j U_j$. Both algorithms are able to solve instances with up to 100 jobs.

Less works have been devoted to the problem with the earliness-tardiness objective function. The recent Branch and Bound algorithm by Sourd and Kedad-Sidhoum [31] and Branch and Bound and Dynaming Programming algorithms by Yau et al. [54] can be used to solve optimally instances of the problem $1 \mid\mid \sum \alpha_j E_j + \beta_j T_j$ with up to 50 jobs.

2.2 Generic MIP Formulation

Time Indexed Formulation

When all processing times p_j of jobs are integers, the single-machine non-preemptive scheduling problem with an arbitrary cost function can be formulated as an Integer Program using time-indexed variables. Binary variable X_{jt}, $j \in N$, $t \in [0, T)$, takes value 1 if job j starts at time t, and otherwise $X_{jt} = 0$. We then have

$$\min \quad \sum_{t=0}^{T} F_j(t+p_j)X_{jt} \tag{1}$$

$$s.t. \quad \sum_{t=r_j}^{d_j-p_j} X_{jt} = 1, \quad j \in N, \tag{2}$$

$$\sum_{j\in N} \sum_{s=\max\{0,t-p_j+1\}}^{t} X_{js} \leq 1, \quad t \in [0,T), \tag{3}$$

$$X_{jt} \in \{0,1\}, \quad j \in N,\ t \in [0,T). \tag{4}$$

The constraints (2) state that each job starts exactly once within its time window. The constraints (3) guarantee that, at each time moment, only one job is processed. Release dates and deadlines can be taken into account by setting appropriate variables to zero.

The time-indexed formulation is known for more than 40 years. It was used, for example, in the works by Bowman [14], Pritsker et al. [1], Redwine and Wismer [24]. The polyhedral study of this formulation was conducted by Dyer and Wolsey [29], Sousa and Wolsey [34], Akker et al. [53]. The main advantage of this formulation is that, by solving its LP relaxation, one can obtain a very strong lower bound on the optimal solution value. In the special case where processing times are equal ($\forall i, p_i = p$), many problems turn to be polynomially solvable (see [6, 5]) and the the continuous relaxation of the time-indexed formulation is integral [16].

Another obvious advantage is the possibility to model single-machine non-preemptive problem with any objective function. Unfortunately, the formulation has one big drawback. For practical instances with a large number of jobs and large processing times, the size of the formulation becomes so large that it is difficult to solve even its LP relaxation in a reasonable time.

Disjunctive Formulation

Another way to solve some single-machine scheduling problems by Integer Programming is to use the following disjunctive formulation. Binary variable δ_{ij}, $i,j \in N$, takes value 1 if job i precedes job j, and otherwise $\delta_{ij} = 0$. Continuous variable C_j, $j \in N$, represents the completion time of job j. Now we can write the linear ordering formulation.

$$\min \quad \sum_{j\in N} F_j(C_j) \tag{5}$$

$$s.t. \quad \delta_{ij} + \delta_{ji} \leq 1, \quad i,j \in N,\ i<j, \tag{6}$$

$$\delta_{ij} + \delta_{jk} + \delta_{ki} \leq 2, \quad i,j,k \in N,\ i \neq j \neq k, \tag{7}$$

$$C_j \geq \sum_{i\in N, i\neq j} p_i\delta_{ij} + p_j, \quad j \in N, \tag{8}$$

$$C_j \geq 0, \quad j \in N, \tag{9}$$

$$\delta_{ij} \in \{0,1\}, \quad i,j \in N,\ i \neq j. \tag{10}$$

The constraints (6) state that, for every pair of jobs, one should precede the other. The constraints (7) guarantee that, for every triple of jobs $i, j, k \in N$, if i precedes j and j precedes k then i should precede k. The constraints (8) relate the variables δ and T. To be able to express the objective function $\sum_{j \in N} F_j(C_j)$ using linear constraints, $F_j(C_j)$ should be piecewise linear and convex for all $j \in N$. Another limitation is that this formulation is not directly applicable where there can be an idle time in all optimal schedules. From the other side, release dates and deadlines can be easily introduced in this formulation. This formulation is compact but its continuous relaxation is known to be rather poor. A survey on this formulation can be found in [38].

3 Basic Results

We say that job j is "started" in interval I_u if its starting time is greater than or equal to e_{u-1} and less than e_u. We say that job j is "completed" in interval I_u if its completion time is such that $e_{u-1} < C_j \le e_u$. Let Q_u denote the set of jobs started and completed in interval I_u:

$$Q_u = \Big\{ j \in N : \Big(S_j, C_j \Big] \subseteq I_u \Big\}.$$

Claim 1 *There exists an optimal schedule in which, for any interval I_u, $u \in M$, and any two jobs $i, j \in Q_u$, job i is sequenced before job j when $\frac{w_i^u}{p_i} > \frac{w_j^u}{p_j}$.*

This claim is based on a simple exchange argument between consecutive jobs. It is a straightforward adaptation of Smith rule (see for instance [15]). In the following, we denote by σ_u a permutation of jobs $\{1, 2, \ldots, n\}$ in which "long" jobs come first in any order, and "short" jobs come last according to Smith rule:

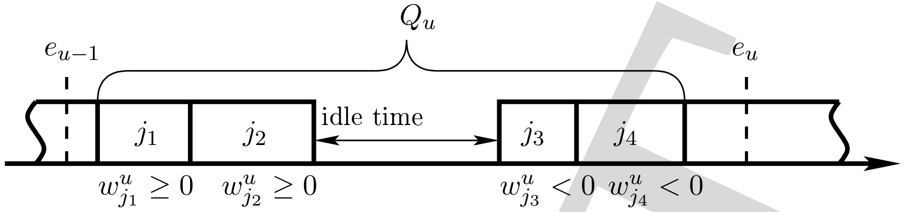
$$\left.\begin{array}{l} p_i < e_u - e_{u-1}, p_i < e_u - e_{u-1}, \frac{w_i^u}{p_i} > \frac{w_j^u}{p_j} \\ \text{or} \\ p_i \ge e_u - e_{u-1}, p_i < e_u - e_{u-1} \end{array}\right\} \Rightarrow \sigma_u(i) < \sigma_u(j), \quad \forall i, j \in N.$$

Although several permutations satisfy this condition, for each $u \in M$, only one of them is used in the remaining of the paper. The necessety of moving "long" jobs to the beginning of the permutation will be clear in Section 5.

Definition 1 *Given a linear partition $\{I_u\}_{u \in M}$, a schedule is called canonical if, for each $u \in M$,*

- *there is one idle time period (possibly of the zero length) in interval I_u,*
- *jobs in Q_u are processed according to the permutation σ_u, where jobs $j \in Q_u$ with $w_j^u \ge 0$ are processed before the idle time period in I_u and jobs $j \in Q_u$ with $w_j^u < 0$ are processed after the idle time period in I_u (see Figure 3).*

Claim 2 *There exists an optimal canonical schedule.*



$$\sigma_u(j_1) < \sigma_u(j_2) < \sigma_u(j_3) < \sigma_u(j_4)$$

Figure 3: Sequencing of jobs in a canonical schedule

Proof: Consider an optimal schedule which is not canonical and $u \in M$ such that jobs in Q_u are not processed according to σ_u. Obviously, $\mid Q_u \mid \geq 2$, meaning that $j \in Q_u \Rightarrow p_j < e_u - e_{u-1}$. Now we rearrange jobs in Q_u according to σ_u. By the Smith rule, the cost of the schedule do not increase, and the schedule remains optimal. Then, we shift jobs $j \in Q_u$, $w_j^u \geq 0$, to the left, and jobs $j \in Q_u$, $w_j^u > 0$, to the right as much as possible and as long as they remain in Q_u. By doing this, we again do not icrease the cost of the schedule and reduce the number of non-empty idle time periods in I_u to at most one. By implementing this procedure for each $u \in M$, we obtain an optimal canonical schedule. $\square$

So, we can restrict our search for an optimal solution to only canonical schedules. Therefore, our problem reduces to

- determining in which intervals jobs are started and completed;
- finding the lengths of the idle time periods in each interval.

In the paper, we rely on the following notation. Let B_j^u and A_j^u, $j \in N$, $u \in M$, be the sets of jobs which come, respectively, before and after job j in the permutation σ_u. Let also NB_u and NS_u denote the sets of "big" and "small" jobs for a given interval I_u: $NB_u = \{i \in N : p_i > e_u - e_{u-1}\}$, $NS_u = \{i \in N : p_i \leq e_u - e_{u-1}\}$. We also define the sets $AB_j^u = A_j^u \cap NB_u$, $BB_j^u = B_j^u \cap NB_u$, $AS_j^u = A_j^u \cap NS_u$, $BS_j^u = B_j^u \cap NS_u$.

4 The interval-indexed formulation

First, we introduce the *variables* of the model. The binary variable X_j^u, $j \in N$, $u \in M$, takes value 1 if job j is started in interval I_u or earlier, and otherwise $X_j^u = 0$. The binary variable Y_j^u, $j \in N$, $u \in M$, takes value 1 if job j is completed in interval I_u or earlier, and otherwise $Y_j^u = 0$. For each $j \in N$, we set $X_j^0 = 0$, $X_j^m = 1$, $Y_j^0 = 0$, $Y_j^m = 1$. The continuous variable W_u, $u \in M$, denotes the length of the idle time period in interval I_u. The continuous variables F_j^u, $j \in N$, $u \in M$, are used to compute the difference between the actual cost of job j and the minimum cost of j over the interval I_u:

$$F_j^u = \begin{cases} 0, & C_j \notin I_u \text{ or } w_j^u = 0, \\ C_j - e_{u-1}, & C_j \in I_u \text{ and } w_j^u > 0, \\ e_u - C_j, & C_j \in I_u \text{ and } w_j^u < 0, \end{cases}$$

Then, if job j is completed in interval I_u, $F_j(C_j) = f_j^u + |w_j^u| F_j^u$, and the objective function can be written as

$$\min \sum_{j \in N} \sum_{u \in M} \mid w_j^u \mid F_j^u + \sum_{j \in N} \sum_{u \in M} \min\{F_j(e_{u-1}), F_j(e_u)\}(Y_j^u - Y_j^{u-1}). \tag{11}$$

4.1 Feasibility constraints

Each canonical schedule is determined by a vector $(X, Y, W) \in \{0,1\}^{nm} \times \{0,1\}^{nm} \times \mathbb{R}_+^m$ of instantiated variables. We now give the constraints which describe the set of feasible canonical schedules.

$$Y_j^{u-1} \leq Y_j^u, \quad \forall u \in M, \ \forall j \in N, \tag{12}$$
$$X_j^{u-1} \leq X_j^u, \quad \forall u \in M, \ \forall j \in N, \tag{13}$$
$$Y_j^u \leq X_j^u, \quad \forall u \in M, \ \forall j \in N, \tag{14}$$
$$X_j^{u-1} \leq Y_j^u, \quad \forall u \in M, \ \forall j \in NS_u, \tag{15}$$
$$Y_j^u \leq X_j^{u-1}, \quad \forall u \in M, \ \forall j \in NB_u, \tag{16}$$
$$X_j^u = 0, \quad \forall j \in N, \ r_j = e_u, \tag{17}$$
$$Y_j^u = 1, \quad \forall j \in N, \ d_j = e_u, \tag{18}$$
$$\sum_{j \in N} p_j Y_j^u + \sum_{v=1}^{u} W_v \leq e_u - \sum_{i \in N} \epsilon(X_i^u - Y_i^u), \quad \forall u \in M, \tag{19}$$
$$\sum_{j \in N} p_j X_j^u + \sum_{v=1}^{u} W_v \geq e_u + \sum_{i \in N} \epsilon(X_i^u - Y_i^u), \quad \forall u \in M, \tag{20}$$
$$\sum_{i \in N} (X_i^u - Y_i^u) \leq 1, \quad \forall u \in M, \tag{21}$$
$$W_u \leq e_u - e_{u-1} - (e_u - e_{u-1}) \cdot \sum_{i \in N} \left(X_i^{u-1} - Y_i^u\right), \ \forall u \in M, \tag{22}$$
$$\sum_{i \in NB_u} (X_i^{u-1} - Y_i^u) + Y_j^u - X_j^{u-1} \leq 1, \quad \forall u \in M, \ \forall j \in NS_u, \tag{23}$$
$$Y_j^u \in \{0,1\}, \quad \forall u \in M, \forall j \in N, \tag{24}$$
$$X_j^u \in \{0,1\}, \quad \forall u \in M, \forall j \in N. \tag{25}$$

The inequalities (12) and (13) ensure that the values taken by variables Y and X are consistent with the definition of these variables. The constraints (14) state that, if a job is completed in some interval, it should be started in this interval or before. The inequalities (15) reflect the fact that, if a job is "small" for an interval, it cannot be started before the beginning of the interval and completed after the end of the interval. The inequalities (16) state that, if a job is "big" for an interval, it cannot be started and completed in this interval. Note that the constraints (15) and (16) can be omitted after a suitable modification of the inequalities (12) and (14). The constraints (17) and (18) are needed to take into account the release dates and deadlines of jobs.

The constraints (19), (20) guarantee that the sum of the processing times of jobs completed (started) in the first u intervals plus the total idle time in these intervals is not more (less) than e_u, i.e. the total length of these intervals. The terms with ϵ are used here to impose the strict conditions: if job j is started in I_u then $S_j < e_u$; if job j is completed in I_u then $C_j > e_{u-1}$. ϵ should be chosen in such a way that, for all $u \in M$, e_u/ϵ is integer, and for all $j \in N$, p_j/ϵ is integer. Note that the terms with ϵ can be omitted as long as $f_j^{u-1} + w_j^u(e_u - e_{u-1}) \leq f_j^u$ for all $u \in M$, $j \in N$ (this obviously holds for regular objective functions).

The constraints (21) state that there is at most one job that is started before time moment e_u and finished after it. The constraints (22) put the length of the idle time period in an interval to zero, if some job is started before the beginning of the interval and completed after the end of the interval. Note that the constraints (22) imply the constraints (21). The constraints (23) eliminate the possibility of "overlapping", when some job i is started before the beginning of an interval I_u and completed after the end of I_u and some job j is fully processed inside interval I_u.

We have just showed that the constraints (12)-(25) are valid. In other words, if a canonical schedule is feasible, the corresponding vector (X, Y, W) satisfy the constraints (12)-(25). We now show that these constraints describe the set of all feasible canonical schedules.

Proposition 1 *Given a linear partition $\{I_u\}_{u \in M}$, let vector (X, Y, W) satisfy the constraints (12)-(25). Then the corresponding canonical schedule is feasible.*

Proof: Let $x(j)$, $j \in N$, be the index such that $X_j^{x(j)} - X_j^{x(j)-1} = 1$ and $y(j)$, $j \in N$, be the index such that $Y_j^{x(j)} - Y_j^{x(j)-1} = 1$. By the constraints (13) and (12), $x(j)$ and $y(j)$ are defined identically. We first show that there is a permutation $(j_1, j_2, \ldots, j_n)$ of jobs which satisfies the condition

$$y(j_{k-1}) \leq x(j_k), \quad k \in \{2, \ldots, n\}. \tag{26}$$

For this, we prove that there is no pair (i, j) of jobs such that $x(i) < y(j)$ and $x(j) < y(i)$.

Consider a pair (i, j) of jobs. Suppose that $x(i) < y(j)$ and $x(j) < y(i)$. Note that, by the constraints (14), $x(i) \leq y(i)$ and $x(j) \leq y(j)$. Then, there can be two possibilities.

1. Let $x(i) < y(i)$ and $x(j) < y(j)$. We denote $x = \max\{x(i), x(j)\}$ and $y = \min\{y(i), y(j)\}$. Then we have $x < y$, and therefore $X_i^x = X_j^x = 1$ and $Y_i^x = Y_j^x = 0$. But this is impossible due to the constraints (21). Contradiction.

2. Let $x(i) = y(i)$ or $x(j) = y(j)$. Without loss of generality, assume that $x(j) = y(j)$. Then $X_j^{x(j)} = Y_j^{x(j)} = 1$, $X_j^{x(j)-1} = Y_j^{x(j)-1} = 0$, and $j \in NS_{x(j)}$, otherwise the constraints (16) would be violated. We have $x(i) < y(j) = x(j) < y(i)$, therefore $X_i^{x(j)-1} = 1$, $Y_i^{x(j)} = 0$, and $i \in NB_{x(j)}$, otherwise we would violate the constraints (15). Consequently, $X_i^{x(j)-1} - Y_i^{x(j)} = 1$ and $Y_j^{x(j)-1} - X_j^{x(j)} = 1$. But this is impossible due to the constraints (23). Contradiction.

So, there exists a permutation $\gamma = (j_1, \ldots, j_n)$ which satisfies the condition (26). We perturb γ by sorting all jobs j such that $x(j) = y(j) = u$ according to the permutation σ_u for all $u \in M$. We obtain permutation $\delta = (j_1, \ldots, j_n)$ which still satisfies (26).

Let B_j^δ and A_j^δ, $j \in N$, be the sets of jobs which come, respectively, before and after job j

in permutation δ. Now we construct schedule π by setting

$$C_j(\pi) = \sum_{i \in B_j^\delta \cup \{j\}} p_i + \sum_{v=1}^{u(j)} W_v, \quad \forall j \in N, \tag{27}$$

$$\text{where} \quad u(j) = \begin{cases} y(j)-1, & w_j^{y(j)} \geq 0 \text{ or } x(j) < y(j), \\ y(j), & w_j^{y(j)} < 0 \text{ and } x(j) = y(j). \end{cases}$$

As $u(j_1) \leq \cdots \leq u(j_n)$, we have $C_{j_k}(\pi) - p_{j_k} \geq C_{j_{k-1}}(\pi)$, $\forall k \in \{2, \ldots, n\}$. To show that π is a feasible schedule, it remains to show that $r_j + p_j \leq C_j(\pi) \leq d_j$. As the partition is linear, we have $r_j = e_\omega$ for some $\omega \in M$. Note that $\omega < x(j)$, otherwise the constraints (17) would be violated. As $X_j^\omega = 0$, $\forall i \in A_j^\delta \cup \{j\}$, and $\omega \leq u(j)$,

$$C_j(\pi) \overset{(27)}{\geq} \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{\omega} W_v + p_j \overset{(20)}{\geq} e_\omega + p_j = r_j + p_j.$$

In the same manner, using the constraints (18), we can prove that the completion times do not violate the deadlines of jobs.

We now show that, for each $j \in N$, $e_{x(j)-1} \leq S_j(\pi) < e_{x(j)}$ and $e_{y(j)-1} < C_j(\pi) \leq e_{y(j)}$.

As $x(j) - 1 \leq u(j)$ and $x(j) \leq x(i)$, $\forall i \in A_j^\delta$, we have $X_i^{x(j)-1} = 0$, $\forall i \in A_j^\delta \cup \{j\}$, and $X_i^{y(j)-1} = 0$, $\forall i \in A_j^\delta$. Therefore,

$$\begin{aligned} S_j(\pi) &= \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{u(j)} W_v \geq \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{x(j)-1} W_v \overset{(20)}{\geq} e_{x(j)-1}, \\ C_j(\pi) &\overset{(27)}{\geq} \sum_{i \in B_j^\delta \cup \{j\}} p_i + \sum_{v=1}^{y(j)-1} W_v \overset{(20)}{\geq} e_{y(j)-1} + \epsilon \sum_{i \in N} \left(X_i^{y(j)-1} - Y_i^{y(j)-1} \right). \end{aligned}$$

Suppose $C_j(\pi) = e_{y(j)-1}$, then, as $p_j > 0$, by the constraints (20), $X_j^{y(j)-1} = 1$, and $Y_j^{y(j)-1} = 0$ implying $C_j(\pi) \geq e_{y(j)-1} + \epsilon$, contradiction. Therefore, $C_j(\pi) > e_{y(j)-1}$.

Note that the constraints (22) imply

$$\sum_{v=x(j)+1}^{y(j)-1} W_v = 0, \quad \forall j \in N. \tag{28}$$

As $y(i) \leq x(j) \leq y(j)$, $\forall i \in B_j^\delta$, we have $Y_i^{x(j)} = 1$, $\forall i \in B_j^\delta$, and $Y_i^{y(j)} = 1$, $\forall i \in B_j^\delta \cup \{j\}$. Therefore,

$$C_j(\pi) \overset{(27)}{\leq} \sum_{i \in B_j^\delta \cup \{j\}} p_i + \sum_{v=1}^{y(j)} W_v \overset{(19)}{\leq} e_{y_j}.$$

Let $u(j) = y(j) - 1$, then

$$\begin{aligned} S_j(\pi) &= \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{y(j)-1} W_v = \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{x(j)} W_v + \sum_{v=x(j)+1}^{y(j)-1} W_v \\ &\overset{(19),(28)}{\leq} e_{x_j} - \epsilon \sum_{i \in N} \left(X_i^{x(j)} - Y_i^{x(j)} \right). \end{aligned}$$

Let $u(j) = y(j)$ implying $x(j) = y(j)$, then

$$S_j(\pi) \overset{(27)}{\leq} \sum_{i \in B_j^\delta} p_i + \sum_{v=1}^{x(j)} W_v \overset{(19)}{\leq} e_{x_j} - \epsilon \sum_{i \in N} \left(X_i^{x(j)} - Y_i^{x(j)} \right).$$

Suppose $S_j(\pi) = e_{x(j)}$, then, as $p_j > 0$, by the constraints (19), $Y_j^{x(j)} = 0$, and $X_j^{x(j)} = 1$, implying $S_j(\pi) \leq e_{x(j)} - \epsilon$, contradiction. Therefore, $S_j(\pi) < e_{x(j)}$.

Finally, by construction, π is a canonical schedule. $\square$

4.2 Constraints Related to the Overall Cost

Now we describe the constraints that relate variables X, Y, W and F.

$$F_j^u \geq \sum_{i \in N \setminus \{j\}} p_i Y_i^{u-1} + p_j Y_j^u + \sum_{v=1}^{u-1} W_v - e_{u-1},$$
$$\forall u \in M,\ \forall j \in NB_u,\ w_j^u > 0, \tag{29}$$

$$F_j^u \geq \sum_{i \in N \setminus \{j\}} p_i Y_i^{u-1} + p_j X_j^{u-1} + \sum_{v=1}^{u-1} W_v - e_{u-1},$$
$$\forall u \in M,\ \forall j \in NS_u,\ w_j^u > 0, \tag{30}$$

$$F_j^u \geq p_j Y_j^u + \sum_{i \in B_j^u} p_i Y_i^u + \sum_{i \in AB_j^u} p_i Y_i^u + \sum_{i \in AS_j^u} p_i X_i^{u-1} +$$
$$\sum_{v=1}^{u-1} W_v - e_{u-1} - (1 - Y_j^u + X_j^{u-1}) \cdot (e_u - e_{u-1}),$$
$$\forall u \in M,\ \forall j \in NS_u,\ w_j^u > 0, \tag{31}$$

$$F_j^u \geq \sum_{i \in N \setminus \{j\}} p_i (1 - X_i^{u-1}) + \sum_{v=u+1}^{m} W_v - (T - e_u) -$$
$$(1 - Y_j^u + Y_j^{u-1}) \cdot (e_u - e_{u-1}),$$
$$\forall u \in M,\ \forall j \in NB_u,\ w_j^u < 0, \tag{32}$$

$$F_j^u \geq \sum_{i \in N \setminus \{j\}} p_i (1 - X_i^{u-1}) + \sum_{v=u+1}^{m} W_v - (T - e_u) -$$
$$(1 - X_j^{u-1} + Y_j^{u-1}) \cdot (e_u - e_{u-1}),$$
$$\forall u \in M,\ \forall j \in NS_u,\ w_j^u < 0, \tag{33}$$

$$F_j^u \geq \sum_{i \in A_j^u \cup BB_j^u} p_i (1 - X_i^{u-1}) + \sum_{i \in BS_j^u} p_i (1 - Y_i^u) +$$
$$\sum_{v=u+1}^{m} W_v - (T - e_u) - (1 - Y_j^u + X_j^{u-1}) \cdot (e_u - e_{u-1}),$$
$$\forall u \in M,\ \forall j \in NS_u,\ w_j^u < 0, \tag{34}$$

Once the variables Y, X, W are instantiated, the constraints (29)-(31) determine the values for variables F_j^u, $j \in N$, $u \in M$, $w_j^u > 0$, and the constraints (32)-(34) determine the values for variables F_j^u, $j \in N$, $u \in M$, $w_j^u < 0$.

Assume job j is completed in interval I_u in π. We consider two cases.

1. Either job j is "big" for I_u (then j is completed first in I_u in π) or job j is "small" for I_u and j is started in I_{u-1} or earlier and completed in I_u in π.

   - $w_j^u > 0$. $C_j(\pi)$ equals the sum of p_j, the total processing time of jobs completed in interval I_{u-1} or earlier and the total idle time in the first $u-1$ intervals. Here $F_j^u = C_j(\pi) - e_{u-1}$. If j is "big" for I_u, F_j^u is instantiated by the constraint (29). If j is "small" for I_u, F_j^u is instantiated by the constraint (30).
   - $w_j^u < 0$. $C_j(\pi)$ equals T minus the sum of the total processing time of jobs in $N \setminus \{j\}$ started in interval I_u or later and the total idle time in the last $m-u$

intervals. Here $F_j^u = e_u - C_j(\pi)$. If j is "big" for I_u, F_j^u is instantiated by the constraint (32). If j is "small" for I_u, F_j^u is instantiated by the constraint (33).

2. Job j is "small" for I_u, and j is started and completed in I_u in π.
   - $w_j^u > 0$. $C_j(\pi)$ equals the sum of p_j, the total processing time of jobs in B_j^u completed in interval I_u or earlier, the total processing time of jobs in A_j^u started in interval I_{u-1} or earlier and the total idle time in the first $u-1$ intervals. Here $F_j^u = C_j(\pi) - e_{u-1}$, F_j^u is instantiated by the constraint (31).
   - $w_j^u < 0$. $C_j(\pi)$ equals T minus the sum of the total processing time of jobs in B_j^u completed in interval I_{u+1} or later, the total processing time of jobs in A_j^u started in interval I_u or later and the total idle time in the last $m-u$ intervals. $F_j^u = e_u - C_j(\pi)$, F_j^u is instantiated by the constraint (34).

To prove the correctness of the interval-based formulation, it remains to show the validity of the constraints (29)-(34). We do this in the next proposition. Let $\underline{F}_j^u$(k) be the value of the right-hand side of the constraint (k), and

$$\underline{F}_j^u = \begin{cases} \underline{F}_j^u(29), & j \in NB_u,\ w_j^u > 0, \\ \max\left\{\underline{F}_j^u(30),\ \underline{F}_j^u(31)\right\}, & j \in NS_u,\ w_j^u > 0, \\ \underline{F}_j^u(32), & j \in NB_u,\ w_j^u < 0, \\ \max\left\{\underline{F}_j^u(33),\ \underline{F}_j^u(34)\right\}, & j \in NS_u,\ w_j^u < 0. \end{cases}$$

Proposition 2 *The formulation (11)-(25), (29)-(34) is correct.*

Proof: Consider vector (X, Y, W) satisfying the constraints (11)-(25) and the corresponding canonical schedule. To prove the proposition, we show that $F_j^u = \max\{\underline{F}_j^u, 0\}$, i.e.

1. if job j is completed in interval I_u, then $w_j^u > 0$ implies $\underline{F}_j^u = C_j(\pi) - e_{u-1}$, and $w_j^u < 0$ implies $\underline{F}_j^u = e_u - C_j(\pi)$;
2. if job j is not completed in interval I_u, then $\underline{F}_j^u \leq 0$.

Case 1. Let job j is completed in interval I_u, implying $Y_j^u = 1$ and $Y_j^{u-1} = 0$. We have two sub-cases.

1.a. Either $j \in NB_u$ (then j is completed first in I_u in π) or $j \in NS_u$, j is started in

I_{u-1} or earlier and completed in I_u in π. We have

$$\begin{aligned}
w_j^u > 0: \quad C_j(\pi) &= \sum_{\substack{i\in N\setminus\{j\},\\ y(i)\le u-1}} p_i + p_j + \sum_{v=1}^{y(j)-1} W_v \\
&= \sum_{i\in N\setminus\{j\}} p_i Y_i^{u-1} + p_j Y_j^u + \sum_{v=1}^{u-1} W_v. \qquad (35)\\
w_j^u < 0: \quad C_j(\pi) &= T - \sum_{\substack{i\in N\setminus\{j\},\\ x(i)\ge u}} p_i - \sum_{v=y(j)+1}^{m} W_v \\
&= T - \sum_{i\in N\setminus\{j\}} p_i(1-X_i^{u-1}) - \sum_{v=u+1}^{m} W_v. \qquad (36)
\end{aligned}$$

Let $j \in NB_u$, $w_j^u > 0$. Then $C_j(\pi) = (35) = \underline{F}_j^u(29) + e_{u-1}$.

Let $j \in NB_u$, $w_j^u < 0$. Then $C_j(\pi) = (36) = e_u - \underline{F}_j^u(32)$.

Let $j \in NS_u$, $w_j^u > 0$. $C_j(\pi) = (35) = \underline{F}_j^u(30) + e_{u-1}$. Also, as $X_j^{u-1} = 1 = Y_j^u$, using (15), $\underline{F}_j^u(31)$ is less or equal to

$$\sum_{i\in N} p_i Y_i^u + \sum_{v=1}^{u-1} W_v - e_u \overset{(19)}{\le} 0. \qquad (37)$$

Let $j \in NS_u$, $w_j^u < 0$. $C_j(\pi) = (36) = e_u - \underline{F}_j^u(33) + e_{u-1}$. Also, as $X_j^{u-1} = 1 = Y_j^u$, using (15), $\underline{F}_j^u(34)$ is less or equal to

$$\begin{aligned}
&\sum_{i\in N\setminus\{j\}} p_i(1-X_i^{u-1}) + \sum_{v=u+1}^{m} W_v - T + e_{u-1} \qquad (38)\\
&\le -\sum_{i\in N\setminus\{j\}} p_i X_i^{u-1} - \sum_{v=1}^{u-1} W_v + e_{u-1} \overset{(20)}{\le} 0.
\end{aligned}$$

1.b. $j \in NS_u$, j is started and completed in I_u in π. Then $X_j^{u-1} = 0$, $Y_j^u = 1$, and we have

$$
\begin{aligned}
w_j^u > 0: \; C_j(\pi) &= \sum_{\substack{i \in B_j^u,\\ y(i) \le u}} p_i + \sum_{\substack{i \in A_j^u,\\ x(i) \le u-1}} p_i + p_j + \sum_{v=1}^{y(j)-1} W_v \\
&= \sum_{i \in B_j^u} p_i Y_i^u + \sum_{i \in A_j^u} p_i X_i^{u-1} + p_j Y_j^u + \sum_{v=1}^{u-1} W_v. \qquad (39) \\
w_j^u < 0: \; C_j(\pi) &= T - \sum_{\substack{i \in B_j^u,\\ y(i) \ge u+1}} p_i - \sum_{\substack{i \in A_j^u,\\ x(i) \ge u}} p_i - \sum_{v=y(j)+1}^{m} W_v \\
= T \; &- \sum_{i \in B_j^u} p_i(1 - Y_i^u) - \sum_{i \in A_j^u} p_i(1 - X_i^{u-1}) - \sum_{v=u+1}^{m} W_v. \qquad (40)
\end{aligned}
$$

Let $w_j^u > 0$. Then, using (16) and (23), $X_i^{u-1} = Y_i^u$, $\forall i \in AB_j^u$, and therefore $C_j(\pi) = (39) = \underline{F}_j^u(31) + e_{u-1}$. Also, as $X_j^{u-1} < Y_j^u$, $\underline{F}_j^u(30) < (37) \le 0$.

Let $w_j^u < 0$. Then, using (16) and (23), $X_i^{u-1} = Y_i^u$, $\forall i \in BB_j^u$, and therefore $C_j(\pi) = (40) = e_u - \underline{F}_j^u(32)$. Also, $\underline{F}_j^u(33) = (38) \le 0$.

Case 2. Let job j is not completed in interval I_u, implying $Y_j^u = Y_j^{u-1}$. We have two sub-cases.

2.a. $j \in NB_u$. Then, $\underline{F}_j^u(29)$ is equal to

$$
\sum_{i \in N} p_i Y_i^{u-1} + \sum_{v=1}^{y(j)-1} W_v - e_{u-1} \overset{(19)}{\le} 0. \qquad (41)
$$

Also, $\underline{F}_j^u(32) = (38) \le 0$.

2.b. $j \in NS_u$. Then $Y_j^{u-1} \overset{(14)}{\le} X_j^{u-1} \overset{(15)}{\le} Y_j^u = Y_j^{u-1} \Rightarrow Y_j^{u-1} = X_j^{u-1}$. Therefore, $\underline{F}_j^u(30) = (41) \le 0$. Using (15), we have $\underline{F}_j^u(31) \le (37) \le 0$ and $\underline{F}_j^u(34) \le (38) \le 0$. Finally, $\underline{F}_j^u(33) = (38) \le 0$. $\square$

Clearly, the interval-indexed formulation is compact. The number of variables do not exceed $3nm + m = O(nm)$, the number of constraints do not exceed $6nm + 3m = O(nm)$, and the number of non-zeros in the coefficients matrix do not exceed $2n^2m + nm^2 + m^2 + 22nm + m$. For the classical objective functions, we have $m = O(n)$, and the size of the formulation becomes $O(n^2) \times O(n^2)$.

4.3 Additional constraints

A usual way to strengthen a MIP formulation is to add redundant constraints which cut off some fractional solutions. In this subsection, we suggest such constraints for the interval-indexed formulation.

Consider intervals I_v and I_u, $v, u \in M$, $v \le u$, and a job $j \in N$.

- Let $p_j \leq e_u - e_v$. Then job j cannot be started before e_v and completed after e_u, therefore $Y_j^u \geq X_j^v$. This constraint is not dominated by other constraints of this type if $p_j > e_u - e_{v+1}$ and $p_j > e_{u-1} - e_v$.

- Let $p_j < e_u - e_v$. Then job j cannot be started after or at e_v and completed before or at e_u, therefore $Y_j^u \leq X_j^v$. This constraint is not dominated by other constraints of this type if $p_j \leq e_{u+1} - e_v$ and $p_j \leq e_u - e_{v-1}$.

- Let $w_j^u > 0$, $e_{v-1} + p_j \leq e_u$ and $e_{v-1} + p_j > e_u$, meaning that, once started in interval I_v, job j should be completed in I_u or later. Then, if j is completed in I_u, $F_j^u \geq p_j - (e_{u-1} - e_{v-1})$, and the constraint

  $$F_j^u \geq (p_j - e_{u-1} + e_{v-1})(Y_j^u - X_j^{v-1}) \tag{42}$$

  is valid. Moreover, if $e_v + p_j \leq e_u$, once started in I_v, j should be completed in I_u, and (42) can be strengthened to

  $$F_j^u \geq (p_j - e_{u-1} + e_{v-1})(X_j^v - X_j^{v-1}).$$

- Let $w_j^u < 0$, $e_v + p_j \leq e_u$ and $e_v + p_j > e_{u-1}$, meaning that, once started in interval I_v, job j should be completed in I_u or earlier. Then, if j is completed in I_u, $F_j^u \geq e_u - e_v - p_j$, and the constraint

  $$F_j^u \geq (e_u - e_v - p_j)(X_j^v - Y_j^{u-1}) \tag{43}$$

  is valid. Moreover, if $e_{v-1} + p_j > e_{u-1}$, once started in I_v, j should be completed in I_u, and (43) can be strengthened to

  $$F_j^u \geq (e_u - e_v - p_j)(X_j^v - X_j^{v-1})$$

Note that the overall number of the suggested constraints which are not dominated is $O(nm)$.

5 Appropriate partitions of the time horizon

In this section, we will restrict the class of canonical schedules. This will allow us to strengthen the interval-indexed formulation by

- reducing the number of feasible solutions of the formulation,
- tightening the constraints (31) and(34), as the term $-(1 - Y_j^u + X_j^{u-1}) \cdot (e_u - e_{u-1})$ will be changed to $-(1 - Y_j^u + Y_j^{u-1}) \cdot (e_u - e_{u-1})$.

Additionally, it will be possible to formulate a special case of the problem using only variables Y and F (subsection 5.3).

Remember that, in a canonical schedule, jobs in Q_u (started and completed in I_u) are sequenced according to the permutation σ_u. Let now $\bar{Q}_u$ denote the set of jobs completed, *but not necessarily started* in interval I_u:

$$\bar{Q}_u = \Big\{ j \in N : \ C_j \in I_u \Big\}.$$

Definition 2 *Given a linear partition $\{I_u\}_{u\in M}$, a schedule is called strictly canonical if, for each $u \in M$,*

- *there is one idle time period (possible of the zero length) in interval I_u,*
- *jobs in $\bar{Q}_u$ are processed according to the permutation σ_u, where jobs in $j \in Q_u$ with $w_j^u < 0$ are processed after the idle time period in I_u and other jobs in $\bar{Q}_u$ are processed before the idle time period in I_u.*

Unfortunately, the set of strictly canonical schedules does not keep the optimality property, as shown in the next example.

Example: Consider the partition $\{(0,9],(9,20]\}$ of the time horizon and the 3-job instance with data shown in Table 1. There is only one optimal schedule $\pi^* = (2,1,3)$ in which all jobs are completed in interval $I_2 = (9,20]$, but the permutation σ_2 is $(1,3,2)$. ♠

j	p_j	r_j	d_j	(f_j^1, w_j^1)	(f_j^2, w_j^2)
1	4	0	20	$(0,0)$	$(0,2)$
2	10	0	20	$(0,0)$	$(0,3.5)$
3	6	0	20	$(0,0)$	$(0,2.4)$

Table 1: The data for Example 5

So, for an arbitrary linear partition of the time horizon, there is not always an optimal strictly canonical schedule.

Definition 3 *A linear partition of the time horizon is called appropriate if there exists an optimal strictly canonical schedule for it.*

5.1 Obtaining an appropriate partition

In this subsection, we will give necessary conditions for a linear partition to be appropriate and describe how an appropriate partition can be obtained. For $u \in M$ and $i, j \in N$ such that $\sigma_u(i) < \sigma_u(j)$, we denote as T_{ij}^u the minimum time moment $t \in [e_{u-1}, e_u - p_i]$ such that, if j is the immediate precedessor of i and $C_j \geq t$ then exchanging j and i do not increase the cost of the schedule:

$$T_{ij}^u = \min_{t\in[e_{u-1},e_u-p_i]} \Big\{t:\ \forall s \in (t, e_u],\ F_{j\leftrightarrow i}(s) \leq 0\Big\},$$

$$\text{where} \quad F_{j\leftrightarrow i}(s) = F_i(s + p_i - p_j) + F_j(s + p_i) - F_j(s) - F_i(s + p_i).$$

If $e_{u-1} > e_u - p_i$, we set $T_{ij}^u = e_{u-1}$.

Now we explain how the values T_{ij}^u can be obtained. First note, that only the first term of the function $F_{j\leftrightarrow i}(s)$ is piecewise linear in interval $[e_{u-1}, e_u - p_i]$, other three terms are linear in it. Therefore, in interval $[e_{u-1}, e_u - p_i]$, $F_{j\leftrightarrow i}(s)$ is piecewise linear with inflections only possible at points $e_v + p_j - p_i$, $v \le u-1$. Knowing this, it is easy to find T^u_{ij} by checking the value of $F_{j\leftrightarrow i}$ at all inflection points and at e_{u-1}. So, the complexity of finding one value T^u_{ij} is $O(m)$.

Proposition 3 *A linear partition $\{I_u\}_{u\in M}$ is appropriate if, for each $u \in M$ and each pair of jobs $i, j \in N$ such that $\sigma_u(i) < \sigma_u(j)$, at least one of the following two conditions is true:*

$$e_u \le e_{u-1} + p_j, \tag{44}$$

$$e_{u-1} \ge T^u_{ij}. \tag{45}$$

Proof: Consider an optimal schedule which is not strictly canonical. We will transform it recursively to a strictly canonical schedule without increasing the cost. We begin with $u = m$.

Main step. First we rearrange jobs in Q_u according to σ_u and leave at most one idle time period (between jobs with $w^u_j \le 0$ and $w^u_j < 0$). This can be done without increasing the cost of the schedule. If now jobs in $\bar{Q}$ are processed according to σ_u, we set $u := u-1$ and do the main step from the beginning. If not, this means that $\sigma_u(j) > \sigma_u(i)$, where j is the job completed but not started in I_u and i is the job processed first among jobs in Q. There cannot be an idle time between j and i, otherwise $w^u_j < 0$ and shifting j to the right would decrease the cost — contradiction with the optimality of the schedule. By the construction of σ_u, $p_j < e_u - e_{u-1}$. So, (44) is violated, meaning that (45) is satisfied. Therefore, as $C_j > e_{u-1} \ge T^u_{ij}$, exchanging j and i do not increase the cost of the schedule.

Now there are two possible cases.

1. Job i still completes in I_u. Then we can rearrange jobs in Q_u according to σ_u without increasing the cost and, as $\sigma_u(i) = \min_{k\in Q_u}\{\sigma_u(k)\}$, jobs in $\bar{Q}_u$ are processed according to σ_u. We set $u := u-1$ and go to the main step.

2. Job i does not complete in I_u anymore. Then we go to the main step without decreasing u (but with less jobs in $\bar{Q}_u$).

We stop when $u = 0$. $\square$

By the definition of the problem, a linear partition of the time horizon is given. In order to check if it is appropriate, we check whether

$$H^u_{ij} = [T^u_{ij},\ e_{u-1} + p_j] \not\subset I_u,\ \forall u \in M, \quad \forall i, j \in N:\ \sigma_u(i) < \sigma_u(j).$$

If, for some $u \in M$, there exist pairs of jobs $i, j \in N$ such that $H^u_{ij} \subset I_u$, it suffices to divide the interval I_u into sub-intervals in a way that they do not strictly contain intervals H^u_{ij}. Clearly, a sub-partition of a linear partition is also linear. In Algorithm 1, we outline the procedure for finding an appropriate sub-partition for a given partition. It is easy to see that the overall procedure has a polynomial complexity. In practice, the time needed to find an appropriate linear partition is negligible in comparison with the time needed to solve the interval-indexed formulation.

Algorithm 1 A procedure for finding an appropriate sub-partition

Partition $\{I_u\}_{1\le u\le m}$ is given
Find σ_u, $1 \le u \le m$
$u := 1$
while $u \le m$ **do**
    $\mathcal{B}_u := \emptyset$; $k := 1$
    for all $(i,j) \in N: \ \sigma_u(i) < \sigma_u(j)$ **do**
        Find $H^u_{ij} = [T^u_{ij},\ e_{u-1} + p_j]$
        if $H^u_{ij} \subset I_u$ **then**
            $\mathcal{B}_u := \mathcal{B}_u \cup \{H^u_{ij}\}$
        end if
    end for
    if $\mathcal{B}_u \ne \emptyset$ **then**
        Find $(t_0, t_1, \ldots, t_k)$ such that $t_0 = e_{u-1}$, $t_k = e_u$,
            and $H \not\subset (t_{l-1}, t_l]$, $\forall 1 \le l \le k$, $\forall H \in \mathcal{B}_u$.
        Divide interval I_u into sub-intervals $\{(t_{l-1}, t_l]\}_{1\le l\le k}$
        $m := m + k - 1$; update σ_v, f^v_j, w^v_j, $\forall u \le v \le m$, $\forall j \in N$
    end if
    $u := u + k$
end while
return $\{I_u\}_{1\le u\le m}$

Example: Consider the 3-job instance of Table 1 and the initial partition $\{(0,9],(9,20]\}$. We have $\mathcal{B}_1 = \emptyset$ and

$$\mathcal{B}_2 = \Big\{H^2_{12} = [12,19], H^2_{32} = [11.75,19], H^2_{13} = [9.8,15]\Big\}.$$

We divide interval $I_2 = (9,20]$ into sub-intervals $(9,12]$ and $(12,20]$ and obtain an appropriate partition $\{(0,9],(9,12],(12,20]\}$. Now, $\sigma_1 = \sigma_3 = (2,1,3)$, $\sigma_2 = (1,3,2)$, $(f^2_1, w^2_1) = (0,2)$, $(f^2_2, w^2_2) = (0,3.5)$, $(f^2_3, w^2_3) = (0,2.4)$, $(f^3_1, w^3_1) = (6,2)$, $(f^3_2, w^3_2) = (10.5,3.5)$, $(f^3_3, w^3_3) = (7.2,2.4)$. ♠

5.2 Tightening the formulation

Once an appropriate linear partition of the time horizon is known, the constraints which determine the values for variables F^u_j, $u \in M$, $j \in NS_u$, can be strengthened: the constraints (30) and (31) can be replaced by the constraint

$$\begin{aligned} F^u_j \geq{} & p_j Y^u_j + \sum_{i \in B^u_j} p_i Y^u_i + \sum_{i \in AB^u_j} p_i Y^u_i + \sum_{i \in AS^u_j} p_i X^{u-1}_i + \\ & \sum_{v=1}^{u-1} W_v - e_{u-1} - (1 - Y^u_j + Y^{u-1}_j) \cdot (e_u - e_{u-1}), \\ & \qquad \forall u \in M,\ \forall j \in NS_u,\ w^u_j > 0, \end{aligned} \tag{46}$$

and the constraints (33) and (34) can be replaced by the constraint

$$\begin{aligned} F^u_j \geq{} & \sum_{i \in A^u_j \cup BB^u_j} p_i (1 - X^{u-1}_i) + \sum_{i \in BS^u_j} p_i (1 - Y^u_i) + \\ & \sum_{v=u+1}^{m} W_v - (T - e_u) - (1 - Y^u_j + Y^{u-1}_j) \cdot (e_u - e_{u-1}), \\ & \qquad \forall u \in M,\ \forall j \in NS_u,\ w^u_j < 0. \end{aligned} \tag{47}$$

To keep the formulation correct the constraint (23) should be replaced by the constraint

$$\sum_{i \in BB_u} (X^{u-1}_i - Y^u_i) + \sum_{i \in A^u_j} (X^{u-1}_i - Y^{u-1}_i) + Y^u_j - X^{u-1}_j \leq 1, \qquad \forall u \in M,\ \forall j \in NS_u. \tag{48}$$

Proposition 4 *Given an appropriate linear partition of the time horizon, the formulation (11)-(22), (24)-(25), (29), (32), (46)-(48) is correct.*

Proof: We first show that the constraint (48) cuts off vectors (X,Y,W) which correspond to schedules which are canonical but not strictly canonical. In such a schedule, for some $u \in M$, job $j \in NS_u$ started and completed in I_u succeeds a job $i \in A^u_j$ completed but not started in I_u. Then $Y^u_j - X^{u-1}_j = 1$, $X^{u-1}_i - Y^{u-1}_i = 1$, and the constraint (48) is violated. Moreover, (48) implies (23). So, a vector (X,Y,W) satisfying the constraints (13)-(22), (24)-(25), (48) corresponds to a feasible strictly optimal schedule.

In Proposition 2, we have showed that $F_j^u = \max\{\underline{F}_j^u(29), 0\}$, $j \in NB^u$, $w_j^u > 0$ and $F_j^u = \max\{\underline{F}_j^u(32), 0\}$, $j \in NB^u$, $w_j^u < 0$. It now suffices to show that

$$F_j^u = \begin{cases} \max\left\{\underline{F}_j^u(46), 0\right\}, & j \in NS_u,\ w_j^u > 0, \\ \max\left\{\underline{F}_j^u(47), 0\right\}, & j \in NS_u,\ w_j^u < 0. \end{cases}$$

Case 1. Let job j is completed in interval I_u, implying $Y_j^u = 1$ and $Y_j^{u-1} = 0$. We have

$$\begin{aligned} w_j^u < 0:\ C_j(\pi) &= \sum_{\substack{i \in B_j^u, \\ y(i) \le u}} p_i + \sum_{\substack{i \in A_j^u, \\ y(i) \le u-1}} p_i + p_j + \sum_{v=1}^{y(j)-1} W_v \\ &= \sum_{i \in B_j^u} p_i Y_i^u + \sum_{i \in A_j^u} p_i Y_i^{u-1} + p_j Y_j^u + \sum_{v=1}^{u-1} W_v. \qquad (49) \\ w_j^u < 0:\ C_j(\pi) &= T - \sum_{\substack{i \in B_j^u, \\ y(i) \ge u+1}} p_i - \sum_{\substack{i \in A_j^u, \\ y(i) \ge u}} p_i - \sum_{v=y(j)+1}^{m} W_v \\ = T &- \sum_{i \in B_j^u} p_i(1 - Y_i^u) - \sum_{i \in A_j^u} p_i(1 - Y_i^{u-1}) - \sum_{v=u+1}^{m} W_v. \qquad (50) \end{aligned}$$

Let $w_j^u > 0$. Then, using (48), (21) and (16), $Y_i^{u-1} = Y_i^u$, $\forall i \in AB_j^u$, and, using (48), (21) and (15), $Y_i^{u-1} = X_i^{u-1}$, $\forall i \in AS_j^u$. Therefore $C_j(\pi) = (49) = \underline{F}_j^u(46) + e_{u-1}$.

Let $w_j^u < 0$. Then, using (48) and (21), $Y_i^{u-1} = X_i^{u-1}$, $\forall i \in A_j^u$, and $Y_i^u = X_i^{u-1}$, $\forall i \in BB_j^u$. Therefore $C_j(\pi) = (50) = e_u - \underline{F}_j^u(47)$.

Case 2. Let job j is not completed in interval I_u, implying $Y_j^u = Y_j^{u-1}$. Then, using (15), $\underline{F}_j^u(46) \le (37) \le 0$ and $\underline{F}_j^u(47) \le (38) \le 0$. $\square$

5.3 Special case with regular objective function and no idle time

We now consider a special case of the problem, in which the objective function is regular, i.e. F_j is non-decreasing for all jobs $j \in N$. Additionally, we suppose that there exists an optimal schedule with no idle time. The last condition holds, for example, if

1. idle times are forbidden;
2. release dates are the same (we can put them to zero).

In this case, given an appropriate linear partition, we can "get rid" of the variables X and W and propose an interval-indexed formulation which uses only variables Y and F:

$$
\begin{aligned}
\min \quad & \sum_{j\in N}\sum_{u\in M} w_j^u F_j^u + \sum_{j\in N}\sum_{u\in M} f_j^u (Y_j^u - Y_j^{u-1}) && (51)\\
s.t. \quad & Y_j^{u-1} \le Y_j^u, \quad \forall j \in N,\ \forall u \in M, && (52)\\
& \sum_{j\in N} p_j Y_j^u \le e_u, \quad \forall u \in M, && (53)\\
& F_j^u \ge p_j Y_j^u + \sum_{i\in A_j^u} p_i Y_i^{u-1} + \sum_{i\in B_j^u} p_i Y_i^u - e_{u-1} - \\
& \qquad (1 - Y_j^u + Y_j^{u-1}) \cdot (e_u - e_{u-1}), \quad \forall j \in N,\ \forall u \in M, && (54)\\
& Y_j^u \in \{0,1\}, \quad \forall j \in N,\ \forall u \in M, && (55)\\
& F_j^u \ge 0, \quad \forall j \in N,\ \forall u \in M. && (56)
\end{aligned}
$$

Here the objective function (51) and the constraints (52)-(53) are conserved. The constraint (54) link variables F and Y: if job j is completed in interval I_u then its completion time equals the sum of processing times of job j, jobs in B_j^u completed in I_u or earlier and jobs in A_j^u completed in I_{u-1} or earlier.

6 Numerical experiments

In order to compare the time-indexed and interval-indexed formulations numerically, these formulations have been tested on instances of the problems $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$ and $1 \mid\mid \sum w_j T_j$. The experiments have been performed on a computer with a 1.8Ghz processor and 512 Mb of memory was using the *Cplex 10.1* MIP solver.

In the experiments, we were interested in the following statistics.

P_t — percentage of instances solved to optimality within time limit t.

T_{av} — average time in seconds needed to solve an instance to optimality (only for instances solved to optimality).

Nd_{av} — average number of nodes in the search tree (only for instances solved to optimality).

Gap — average integrality gap, i.e. the average difference between the best found solution and the best found lower bound, percentage wise the best found solution (only for instances which were not solved to optimality and for which at least one feasible solution was found).

XLP — average difference between the the best found solution and the lower bound at the top node of the search tree after generating standard cuts, percentage wise the optimal solution (only for instances for which the LP relaxation was solved within the time limit).

6.1 Test instances

The first group of the test instances of the problem $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$ were generated using the following standard procedure. For a given number of jobs n, the processing times of each

(n,θ)	Time-indexed					Interval-indexed				
	$P_{1000\text{s}}$	T_{av}	Nd_{av}	XLP	Gap	$P_{1000\text{s}}$	T_{av}	Nd_{av}	XLP	Gap
$(10,10)$	100%	2.7	89.5	0.7%	0.0%	100%	7.1	820.3	25.8%	0.0%
$(15,10)$	92.6%	47.3	1107.8	2.3%	1.3%	85.2%	194.3	8340.9	25.6%	3.8%
$(20,10)$	66.7%	152.8	1519.4	1.9%	2.2%	25.9%	255.6	2993.4	23.9%	13.6%
$(10,50)$	81.4%	135.0	627.1	1.3%	3.4%	100%	5.6	552.8	23.2%	0.0%
$(15,50)$	55.6%	316.1	98.3	3.3%	6.4%	85.2%	235.6	6949.8	23.5%	6.7%
$(20,50)$	22.2%	383.5	0.0	11.3%	17.4%	29.6%	394.5	4458.8	23.1%	10.7%

Table 2: Comparison of the formulations on the first group of the test instances of the problem $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$

job are first randomly drawn from the uniform distribution $U[\theta, 10\cdot\theta)$. Then the due dates are drawn from $U[d_{min}, d_{\min}+\rho P]$ where $d_{\min} = \max(0, P(\tau-\rho/2))$ and $P = \sum_{j=1}^{n} p_j$, the release dates r_j, $j \in N$, are drawn from $U[0, \phi d_j]$, and weights α_j, β_j are drawn from $U[1,5]$. The four parameters θ, τ, ρ, ϕ are respectively the *time*, *tardiness*, *range* and *release* parameters.

We generated instances for $n \in \{10, 15, 20, 30\}$, $\theta \in \{10, 50\}$, $\tau \in \{0.2, 0.5, 0.8\}$, $\rho \in \{0.2, 0.5, 0.8\}$, $\phi \in \{0.2, 0.5, 0.8\}$. For each value of $(n, \theta, \tau, \rho, \phi)$, 1 instance was generated, making 27 instances for each couple (n, θ). The results are presented in Table 2.

You can see that the time-indexed formulation performs better when the processing times are smaller, and the interval-indexed formulation is preferable when processing times are big. Note that, when $(n, \theta) = (20, 50)$, the time-indexed formulation was not able to find a feasible solution in 1000 seconds for the half of instances.

The second group of the test instances of the problem $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$ were generated using the procedure just presented but with one difference: here we limit by μ_n the number of distinct release and due dates. This allows us to decrease the number of intervals for the interval-indexed formulation. Such a restriction makes sense, as in practice, number of different release and due dates of jobs is often very limited. For generating instances, we set $\mu_n = \lceil n \cdot 2/3 \rceil$. Again, for each couple (n, θ), 27 instances were generated. The results are presented in Table 3.

On these instances, the interval-indexed formulation performs better, as the number of intervals is reduced. Though, still when the processing times are smaller and number of jobs is 30 or less, the time-indexed formulation is preferable. On instances with 40 jobs and more, the time-indexed formulation starts to have difficulties, as less and less feasible solutions can be found within the time limit. For the half of 40-job instances and for all 50-job instances, no feasible solution was found within 1000 seconds.

The test instances of the problem $1 \mid\mid \sum w_j T_j$ were generated using the following similar procedure. For a given number of jobs n, the processing times of each job are first randomly drawn from the uniform distribution $U[1, 100]$. Then the due dates are drawn from $U[d_{min}, d_{\min} + \rho P]$ where $d_{\min} = \max(0, P(\tau - \rho/2))$ and $P = \sum_{j=1}^{n} p_j$, and weights are

(n,θ)	Time-indexed $P_{1000\text{s}}$	T_{av}	Nd_{av}	XLP	Gap	Interval-indexed $P_{1000\text{s}}$	T_{av}	Nd_{av}	XLP	Gap
(10, 10)	100%	4.3	161.6	0.7%	0.0%	100%	1.3	703.2	33.2%	0.0%
(20, 10)	81.5%	201.0	1226.6	1.1%	1.3%	48.5%	393.9	26788.2	27.7%	7.7%
(30, 10)	40.7%	279.2	130.6	5.5%	8.9%	3.7%	393.9	948.1	28.7%	18.3%
(40, 10)	7.4%	508.4	170.0	10.7%	12.1%	0.0%	0.0	0.0	34.6%	30.0%
(50, 10)	-	-	-	-	-	0.0%	0.0	0.0	40.7%	39.4%
(10, 50)	92.6%	86.1	124.1	0.7%	3.5%	100%	1.5	769.5	35.8%	0.0%
(15, 50)	48.2%	350.0	42.9	4.3%	8.2%	96.3%	130.8	23214.0	29.0%	1.2%
(20, 50)	18.5%	417.4	6.4	6.4%	10.8%	51.8%	225.5	16238.3	27.8%	6.6%

Table 3: Comparison of the formulations on the second group of the test instances of the problem $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$

n	Time-indexed $P_{10\text{m}}$	T_{av}	Nd_{av}	XLP	Gap	Interval-indexed $P_{10\text{m}}$	T_{av}	Nd_{av}	XLP	Gap
10	100%	0.3	1.1	0.0%	0.0%	100%	0.1	14.1	4.7%	0.0%
20	99.2%	23.3	25.7	0.3%	0.0%	100%	1.7	338.3	6.8%	0.0%
30	80.0%	120.5	53.6	0.4%	9.9%	82.4%	30.8	5245.8	6.8%	3.0%
40	51.2%	450.6	33.0	0.5%	12.1%	64.8%	34.0	2320.0	7.0%	3.6%
50	31.2%	272.1	38.4	0.6%	7.4%	57.6%	56.5	2654.8	7.2%	3.9%

Table 4: Comparison of the formulations on the test instances of the problem $1 \mid\mid \sum w_j T_j$

drawn from $U[1, 10]$. We generated instances for $n \in \{10, 20, 30\}$, $\tau \in \{0, 0.2, 0.4, 0.6, 0.8\}$, $\rho \in \{0.2, 0.4, 0, 6, 0.8, 1\}$. For each triple (n, τ, ρ), 5 instance were generated, making 125 instances for each n. The larger test instances of the problem $1 \mid\mid \sum w_j T_j$ were taken from the OR-Library [10]. For the problem $1 \mid\mid \sum w_j T_j$, the interval-indexed formulation uses only the variables Y and F. The results are presented in Table 4. Note that the linear programming relaxation of the time-indexed formulation could not be solved within 10 minutes for 1% of the 20-job instances, 2% of the 30-job instances, 23% of the 40-job instances, and 54% of the 50-job instances

You can clearly see here that the interval-indexed formulation gives better results on these test instances.

6.2 Practical instances

Recently, Le Pape and Robert have published a library of practical instances for the planning and scheduling problems [37]. The instances of the type "NCOS" in this library can be transformed to instances of the problem $1 \mid r_j \mid \sum \alpha_j E_j + \beta_j T_j$. We have tested the both formulations on the open instances of this type.

Using the interval-indexed formulation, the following previously open 4 instances have been solved to optimality:

Name	n	Time
NCOS_04c	10	5s
NCOS_05c	15	109s
NCOS_14c	25	102m
NCOS_14d	25	51m

Additionally, using the time-indexed formulation, the following previously open 12 instances have been solved to optimality:

Name	n	Time
NCOS_11c	20	193s
NCOS_12c	24	257s
NCOS_12d	24	347s
NCOS_13c	24	53s
NCOS_15c	30	1320s
NCOS_21a	50	<1s
NCOS_21c	50	<1s
NCOS_32c	75	2s
NCOS_32d	75	2s
NCOS_51c	200	8s
NCOS_51d	200	7s
NCOS_61d	500	15s

Note that the large instances which were solved contain many identical jobs.

7 Conclusions

In this paper, we have proposed the interval-indexed formulation which is the first compact MIP formulation for the single machine scheduling problem to minimize a piecewise linear objective function. This formulation has $O(nm)$ variables and $O(nm)$ constraints, where n is the number of jobs, and m is the number of intervals in which the objective functions of all jobs are linear.

Both the time-indexed and interval-indexed formulations have advantages. The former has much less binary variables, and the latter provides very strong linear programming lower bounds. The numerical experiments showed that the choice of the formulation to use should be made based on the properties of the given instance to solve. The larger the processing times of jobs are the more likely that the interval-indexed formulation will provide better results.

Unfortunately, the experimental research showed that the direct application of the both formulations are usefull only for solving relatively small instances. On the positive side, some open practical instances were solved.

The main direction for the future research concerns the biggest disadvantage of the interval-indexed formulation — weakness of lower bounds provided by its LP relaxation. The formulation should be tightened in order to be more useful in practice.

Another research direction is an extension of the formulation to more general situations. These can be the presence of precedence relations between jobs, or the availability of several identical or unrelated machines.

Adaptation to the special cases of the problem is also a perspective direction. For example, often in practice, many jobs are fully or almost identical. Exploiting such particularities can lead to reducing the formulation size or its strengthening.

As it was mentioned, the direct application of the interval-indexed formulation is not usually efficient. As an alternative, the Dantzig-Wolfe reformulation and the column generation method can be tried. This approach would be similar to one of Bigras et al [13]. The latter also uses a partition of the time horizon. An advantage of our approach is that the partition is done taking into account properties of the problem, and this can be exploited to speed up the column generation procedure.

References

[1] P.M. Wolfe A.A.B. Pritsker, L.J.Watters. Multiproject scheduling with limited resources : a zero-one programming approach. *Manag. Sci.*, 16:93–108, 1969.

[2] T.S. Abdul-Razacq, C.N. Potts, and L.N. Van Wassenhove. A survey of algorithms for the single machine total weighted tardiness scheduling problem. *Discrete Applied Mathematics*, 26:235–253, 1990.

[3] M.S. Akturk and D. Ozdemir. An exact approach to minimizing total weighted tardiness with release dates. *IIE Transactions*, 32:1091–1101, 2000.

[4] M.S. Akturk and D. Ozdemir. A new dominance rule to minimize total weighted tardiness with unequal release dates. *European Journal of Operational Research*, 135:394–412, 2001.

[5] Ph. Baptiste. Polynomial time algorithms for minimizing the weighted number of late jobs on a single machine when processing times are equal. *Journal of Scheduling*, 2:245–252, 1999.

[6] Ph. Baptiste. Scheduling equal-length jobs on identical parallel machines. *Discrete Applied Mathematics*, 103:21–32, 2000.

[7] Ph. Baptiste, J. Carlier, and A. Jouglet. A branch and bound procedure to minimize total tardiness on one machine with arbitrary release dates. *to appear in European Journal of Operational Research*, 2002.

[8] Ph. Baptiste, C. Le Pape, and L. Peridy. Global constraints for partial csps: A case study of resource and due-date constraints. In *Proceedings of the Fourth International Conference on Principles and Practice of Constraint Programming, Pisa*, 1998.

[9] Ph. Baptiste, L. Peridy, and E. Pinson. A branch and bound to mininimze the number of late jobs on a single machine with release time constraints. *European Journal of Operational Research*, 144:1–11, 2003.

[10] J.E. Beasley. Or-library. http://people.brunel.ac.uk/ mastjjb/jeb/info.html.

[11] H. Belouadah, M.E. Posner, and C.N. Potts. Scheduling with release dates on a single machine to minimize total weighted completion time. *Discrete Applied Mathematics*, 36:213–231, 1992.

[12] L. Bianco and S. Ricciardelli. Scheduling of a single machine to minimize total weighted completion time subject to release dates. *Naval Research Logistics*, 29:151–167, 1982.

[13] L.-Ph. Bigras, M. Gamache, and G Savard. Time-indexed formulations and the total weighted tardiness problem. *INFORMS Journal on Computing*, Forthcoming.

[14] E.H. Bowman. The schedule-sequencing problem. *Oper. Res.*, 7:621–624, 1959.

[15] Peter Brucker. *Scheduling Algorithms*. Springer, 2001.

[16] Peter Brucker and Svetlana Kravchenko. Scheduling jobs with equal processing times and time windows on identical parallel machines. *to appear in Journal of Scheduling*, 2005.

[17] S. Chand, R. Traub, and R. Uzsoy. Single-machine scheduling with dynamic arrivals: Decomposition results and an improved algorithm. *Naval Research Logistics*, 43:709–716, 1996.

[18] R. Chandra. On $n/1/\overline{F}$ dynamic determistic systems. *Naval Research Logistics*, 26:537–544, 1979.

[19] S. Chang, Q. Lu, Tang G., and W. Yu. On decomposition of the total tardiness problem. *Operations Research Letters*, 17:221–229, 1995.

[20] C. Chu. A branch and bound algorithm to minimize total flow time with unequal release dates. *Naval Research Logistics*, 39:859–875, 1991.

[21] C. Chu. A branch and bound algorithm to minimize total tardiness with different release dates. *Naval Research Logistics*, 39:265–283, 1992.

[22] C. Chu. Efficient heuristics to minimize total flow time with release dates. *Operations Research Letters*, 12:321–330, 1992.

[23] C. Chu and M.C. Portmann. Some new efficient methods to solve the $n|1|r_i|\sum T_i$ scheduling problem. *European Journal of Operational Research*, 58:404–413, 1991.

[24] D.A. Wismer C.N. Redwine. A mixed integer programming model for scheduling orders in a steel mill. *J. of Optimization Theory and Applications*, 14:305–318, 1974.

[25] S. Dauzère-Pérès and M. Sevaux. A branch and bound method to minimize the number of late jobs on a single machine. Technical report, Research report 98/5/AUTO, Ecole des Mines de Nantes, 1998.

[26] D.S. Deogun. On scheduling with ready times to minimize mean flow time. *Comput. J.*, 26:320–328, 1983.

[27] M.I. Dessouky and D.S. Deogun. Sequencing jobs with unequal ready times to minimize mean flow time. *SIAM J. Comput.*, 10:192–202, 1981.

[28] J. Du and J.Y.T. Leung. Minimizing total tardiness on one processor is NP-Hard. *Mathematics of Operations Research*, 15:483–495, 1990.

[29] M. Dyer and L.A. Wolsey. Formulating the single machine sequencing problem with release dates as mixed integer program. *Disc. Applied Math.*, 26:255–270, 1990.

[30] H. Emmons. One-machine sequencing to minimize certain functions of job tardiness. *Operations Research*, 17:701–715, 1969.

[31] F. Francis Sourd and S. Kedad-Sidhoum. An efficient algorithm for the earliness-tardiness scheduling problem. *Optimisation Online*, (1205), 2005.

[32] A.M.A Hariri and C.N. Potts. An algorithm for single machine sequencing with release dates to minimize total weighted completion time. *Discrete Applied Mathematics*, 5:99–109, 1983.

[33] A. Jouglet, Ph. Baptiste, and J. Carlier. *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, chapter Branch-and-Bound Algorithms for Total Weighted Tardiness. CRC Press, 2004.

[34] L.A. Wolsey J.P. Sousa. A time-indexed formulation of non-preemptive single-macine scheduling problems. *Math. Prog.*, 54:353–367, 1992.

[35] J. Labetoulle, E.L. Lawler, J.K. Lenstra, and A.H.G Rinnooy Kan. *Progress in Combinatorial Optimization*, chapter Preemptive scheduling of uniform machines subject to release dates. Academic Press, New York, 1984.

[36] E.L. Lawler. A pseudo-polynomial algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, 1:331–342, 1977.

[37] C. Le Pape and A. Robert. Jeux de données pour l'évaluation d'algorithmes de planification et ordonnacement. In *Livre des résumés séléctionnés lors de la conférence conjointe FRANCORO V / ROADEF 2007, 20–23 fevrier, Grenoble*, 2007.

[38] A.S. Schulz M. Queyranne. Polyhedral approaches to machine scheduling. *Department of Mathematics, Technical University of Berlin*, Preprint 408, 1994.

[39] R. M'Hallah and R.L. Bulfin. Minimizing the weighted number of tardy jobs on a single machine with release dates. *European Journal of Operational Research*, 176(2):727–744, 2007.

[40] Y. Pan and L. Shi. On the equivalence of the max-min transportation lower bound and the time-indexed lower bound for single-machine scheduling problems. *Mathematical Programming, Series A*, 110(3):543–559, 2006.

[41] Y. Pan and L. Shi. New hybrid optimization algorithms for machine scheduling problems. *IEEE Trans. on Automation Science and Engineering*, Forthcoming.

[42] C.N. Potts and L.N. Van Wassenhove. A decomposition algorithm for the single machine total tardiness problem. *Operations Research Letters*, 26:177–182, 1982.

[43] R.M.V. Rachamadugu. A note on weighted tardiness problem. *Operation Research*, 35:450–452, 1987.

[44] G.L. Ragatz. A branch-and-bound method for minimum tardiness sequencing on a single processor with sequence dependent setup times. In *Twenty-fourth Annual Meeting of the Decison Sciences Institute*, 1993.

[45] G. Rinaldi and A. Sassano. On a job scheduling problem with different ready times: Some properties and a new algorithm to determine the optimal solution. *Operation Research*, 1977. Rapporto dell'Ist. di Automatica dell'Universita di Roma e del C.S.S.C.C.A.-C.N.R.R, Report R.77-24.

[46] A.H.G. Rinnooy Kan. *Machine sequencing problem: classification, complexity and computation*. Nijhoff. The Hague, 1976.

[47] A.H.G. Rinnooy Kan, B.J. Lageweg, and J.K. Lenstra. Minimizing total costs in one-machine scheduling. *Operation Research*, 23:908–927, 1975.

[48] R. Sadykov. A hybrid branch-and-cut algorithm for the one-machine scheduling problem. In *Proceedings of the International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimisation Problems*, 2004.

[49] W.E. Smith. Various optimizers for single stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

[50] A. Souissi, I. Kacem, and C. Chu. New lower bound for minimizing total tardiness on a single machine with sequence-dependent setup times. In *Proceedings of the Ninth International Conference on Project Management and Scheduling, PMS04*, 2004.

[51] W. Szwarc, F. Della Croce, and A. Grosso. Solution of the single machine total tardiness problem. *Journal of Scheduling*, 2:55–71, 1999.

[52] W. Szwarc, A. Grosso, and F. Della Croce. Algorithmic paradoxes of the single machine total tardiness problem. *Journal of Scheduling*, 4:93–104, 2001.

[53] J.M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. A polyhedral approach to single-machine scheduling problems. *Math. Prog.*, 85:541–572, 1999.

[54] H. Yau, Y. Pan, and L Shi. New solution approaches to the general single machine earliness-tardiness problem. *IEEE Trans. on Automation Science and Engineering*, Forthcoming.