# A branch and bound approach for earliness-tardiness scheduling problems with different due dates

Haoxun Chen, Chengbin Chu, Jean-Marie Proth

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *A branch and bound approach for earliness-tardiness scheduling problems with different due dates*

Haoxun CHEN - Chengbin CHU - Jean-Marie PROTH

N° 2495
Février 1995

PROGRAMME 5

# A BRANCH AND BOUND APPROACH FOR EARLINESS-TARDINESS SCHEDULING PROBLEMS WITH DIFFERENT DUE DATES

## Une méthode par séparation et évaluation pour minimiser les avances/retards avec délais différents

Haoxun Chen
Xi'an Jiaotong University, Xi'an, People's Republic of China and
visiting professor at INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France

Chengbin Chu
INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France

Jean-Marie Proth
INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France and
Institute for Systems Research, University of Maryland,
College Park, MD, USA

**Abstract.** The problem of scheduling $n$ jobs on a single machine in order to minimize the weighted sum of earliness and tardiness is NP-complete when jobs have different due dates. In most of the papers dedicated to this problem, authors assume that there is no idle time between two consecutive jobs. However, as indicated by several authors, this assumption is not consistent with the earliness-tardiness criterion. It is the reason why we do not make this assumption in this paper. To reach an optimal solution, we propose a branch-and-bound approach which takes advantage of some dominance properties and lower bounding procedures. Numerical experiments show that the algorithm can solve this problem with up to twenty jobs in a reasonable amount of time.

**Keywords.** Scheduling, Earliness, Tardiness, Lower Bounds, Dominance Properties, Branch and Bound, Dynamic Programming.

**Résumé.** Le problème d'ordonnancement des tâches sur une machine en vue de minimiser la somme pondérée des avances et des retards est NP-difficile lorsque les délais des tâches sont différents. Dans la plupart des articles dédiés à ce type de problèmes, les auteurs supposent qu'il n'y a pas de temps mort entre deux tâches consécutives. En revanche, comme l'ont indiqué plusieurs auteurs, ceci n'est pas cohérent avec le critère avances retards. C'est pourquoi nous ne faisons pas cette hypothèse dans ce papier. Pour obtenir une solution optimale, nous utilisons la méthode par séparation et évaluation en exploitant des propriétés de dominance. Les expériences numériques montrent que cet algorithme permet de résoudre des problèmes comportant jusqu'à vingt tâches, la plus grande taille de problèmes résolus à ce jour, en un temps raisonable.

**Mots clefs.** Ordonnancement, Avances, Retards, Bornes inférieures, Propriétés de dominance, Séparation et évaluation, Programmation dynamique.

# 1. Introduction

The study of earliness and tardiness penalties in scheduling models is a quite new area of research. For many years, scheduling researches focused on single performance measures, referred to as *regular* measures, which are nondecreasing with respect to job completion times. However, this emphasis changed with the current interest in Just-In-Time (JIT) production [9], for which earliness, as well as tardiness, should be taken into account. The scheduling models (problems) with both earliness and tardiness penalties are referred to as E/T models (problems).

For a comprehensive review of machine scheduling with E/T models, see Baker and Scudder [2]. Kanet [13] studied a special class of E/T problems which minimize the total (unweighted) sum of earliness and tardiness for jobs having a common due date. He introduced an unrestricted version and a restricted version of the problems, the latter being used when the common due date is too tight for a sufficient quantity of the jobs to be completed before it. For the unrestricted version, it has been proved that there exists an optimal V-shaped schedule with no idle time between two successive jobs such that one job is completed exactly at the due date. The algorithm to solve this problem is polynomial (Emmons [5]). The restricted version is solved by enumeration. A very efficient pseudopolynomial dynamic programming algorithm has been proposed by Hall and Posner [11], Hall, Kubiak and Sethi [10].

The E/T problem tends to be very difficult when jobs have different due dates. Abdul-Razaq and Potts [1] solved this type of problems, but they considered only schedules with no inserted idle time. Their solution method is a branch-and-bound scheme, and they used a *relaxed dynamic programming* procedure to obtain good bounds. Their computational results suggest that problems with more than 25 jobs may lead to excessive computation times. More recently, Ibaraki and Nakamura [12] improved the procedure: they proposed an SSDP (Successive Sublimation Dynamic Programming) method to solve the problem. Computational experiments show that the SSDP method can solve problems with up to 35 jobs. However, they also considered only schedules with no inserted idle time. It might be meaningful to study this version of the model, but, as indicated in Baker and Scudder [2], the assumption of no idle time is not consistent with the E/T criterion.

Assuming that the job sequence is given, the optimal schedule when idle times are allowed can be obtained by applying a simple procedure proposed by Garey, Tarjan and Wilfong [8]. The complexity of the algorithm is $O(n\log n)$. Similar schemes have been developed by Davis and Kanet [3], by Fry, Darby-Dowman and Armstrong [7] and by Kim and Yano [14].

Given that the idle time can easily be optimized for a specified job sequence, the remaining task is to determine the best sequence. Fry, Armstrong and Blackstone [6] described a solution procedure which is based on local search through adjacent pairwise interchange neighborhoods until a local optimum is discovered. Their computational tests indicate that this heuristic yields solutions within 2% deviation from optimum on the average. Fry, Darby-Dowman and Armstrong [7] describe a branch-and-bound procedure for minimizing the mean sum of earliness and tardiness. They indicate that the algorithm runs into difficulty in an attempt to solve problems with more than 20 jobs. Kim and Yano [14] consider the special case of the problem when the criterion is the total (unweighted) sum of earliness and tardiness, and they propose a branch-and-bound algorithm that solve problems with up to 20 jobs.

In this paper, we consider the E/T problem with jobs having different due dates, and different earliness and tardiness penalties. In addition, we do not consider the assumption of no

inserted idle time. This problem has not been considered in the literature, since all the work cited above make assumption either on common due dates, or with particular earliness and tardiness penalties, or without inserted idle times. We exploit several optimality conditions of the optimal schedules and propose a branch and bound algorithm to solve the problem. The branch and bound algorithm enumerates completion time windows of jobs and is based on dominance properties and on two lower bounding procedures. Computational results for problems with up to 20 jobs are given.

## 2. Model

The general E/T scheduling problem is defined as follows: $n$ jobs have to be scheduled on a single machine. Each job $j$ ($j = 1, 2, \ldots, n$) has a *positive* integer processing time $p_j$ and a *positive* integer due date $d_j$. The jobs are assumed to be available at time 0, i.e., the release date of all the jobs is zero. If $C_j$ is the completion time of job $j$, the earliness and tardiness of job $j$, respectively denoted by $E_j$ and $T_j$, are defined as follows:

$$E_j = \max(0, d_j - C_j) = (d_j - C_j)^+,$$

$$T_j = \max(0, C_j - d_j) = (C_j - d_j)^+.$$

The objective is to find a schedule $\sigma = \{C_1, C_2, \ldots, C_n\}$ that minimizes the sum of weighted earliness and tardiness of the jobs, that is:

$$\min_{\sigma \in \Pi} f(\sigma) = \sum_{j=1}^{n} (\alpha_j E_j + \beta_j T_j) \tag{1}$$

where $\Pi$ denotes the set of all possible nonpreemptive single machine schedules, $\alpha_j$ and $\beta_j$ are the weights associated respectively with earliness and tardiness of job $j$. The constraints to be verified are:

$$C_j \geq p_j, \qquad \forall 1 \leq j \leq n. \tag{2}$$

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset, \quad \forall j < k \leq n, 1 \leq j \leq n. \tag{3}$$

Constraints (2) say that any job cannot start processing before time 0. Constraints (3) are capacity constraints to make sure that the machine process at most one job at a time.

Let $m$ be the number of distinct due dates. It is obvious that $1 \leq m \leq n$. Let $\bar{d}_i$ be the $i$th smallest distinct due date. We then have $\bar{d}_1 < \bar{d}_2 < \cdots < \bar{d}_m$. By convention, we set $\bar{d}_0 = 0$ and $\bar{d}_{m+1} = +\infty$. In what follows, the time interval $(\bar{d}_i, \bar{d}_{i+1}]$ is referred to as the $i$th time window ($i = 0, 1, \ldots, m$). Note that for any $i = 0, 1, \ldots, m$, there is no $j$ such that $\bar{d}_i < d_j < \bar{d}_{i+1}$.

## 3. Optimality Conditions

In this section, we describe some dominance properties of the E/T problem. These properties will allow us to reduce the search for an optimal solution. From these properties, we deduce a new branching scheme in our branch and bound algorithm.

In the remainder of this section, we consider an *optimal* schedule $\sigma = \{C_1, C_2, \ldots, C_n\}$ and analyze its properties. This will allow us to consider only schédules satisfying these properties. For this purpose, we first define the following sets for each time window $i$ ($i = 0, 1, \ldots, m$):

$F_i = \{j | \bar{d}_i < C_j \leq \bar{d}_{i+1}\}$: set of jobs completed (or finished as indicated by "$F$") within the $i$th time window.

$X_i = \{j \in F_i | C_j - p_j < \bar{d}_i\}$: set of jobs started before $\bar{d}_i$ and finished within the $i$th time window. It is obvious that $X_i$ contains at most one job. If it is not empty, this job is called *cross-due-date job* of the $i$th time window.

$F_i^t = \{j \in F_i | d_j \leq \bar{d}_i\}$: set of jobs completed within the $i$th time window and being tardy.

$F_i^e = \{j \in F_i | d_j > \bar{d}_i\}$: set of jobs completed within the $i$th time window and being early or on time.

Since $\forall j, d_j \geq \bar{d}_1$, all jobs in $F_0$ are early or on time, the following remark is true.

**Remark 1.** $F_0^t = \emptyset$.

On the other hand, since $\forall j, d_j \leq \bar{d}_m$, all jobs in $F_m$ are tardy, the following remark holds.

**Remark 2.** $F_m^e = \emptyset$.

In the remainder, the set operation "$A \setminus B$" means "$A \setminus (A \cap B)$". Thus, it is possible that $(A \setminus B) \cup B \neq A$.

We now prove that, since $\sigma$ is an optimal schedule, it should have the following proerties. All the proofs use a reduction at absurdum method.

**Property 1.** *For any $i = 0, 1, \ldots, m$, and for any pair of jobs $j, k \in F_i \setminus X_i$, if $j \in F_i^t$ and $k \in F_i^e$, then job $j$ precedes job $k$.*

**Proof.** Assume that this property is not true, that is, there are an $i$ and two jobs $j, k \in F_i \setminus X_i$ such that $j \in F_i^t$, $k \in F_i^e$ and job $k$ precedes job $j$. Between jobs $k$ and $j$ (including themselves), there are certainly two *adjacent* jobs $r, s \in F_i \setminus X_i$, $r \in F_i^t$ and $s \in F_i^e$, such that jobs $s$ precedes immediately job $r$. As a consequence, $C_s \leq C_r - p_r$. If we interchange jobs $r$ and $s$, the schedule of the other jobs remains unchanged, and we obtain a new schedule $\sigma'$ with $C'_r = C_s - p_s + p_r \leq C_r - p_s$, $C'_s = C_r \geq C_s + p_r$. According to the assumption of the property, we have $C'_r \geq \bar{d}_i + p_r \geq d_r$ and $C'_s \leq \bar{d}_{i+1} \leq d_s$. This means that in $\sigma'$, job $r$ is still tardy and job $s$ is still early. Therefore, we have

$$f(\sigma') - f(\sigma) = \alpha_s(C_s - C'_s) + \beta_r(C'_r - C_r) \leq -\alpha_s p_r - \beta_r p_s < 0.$$

This means that schedule $\sigma$ is not optimal, which is in contradiction with the assumption. □

From this property we can see that, except for the cross-due-date job (if any), jobs in $F_i^t$ are scheduled together and those of $F_i^e$ are scheduled together, for any $i = 0, 1, \ldots, m$.

**Remark 3.** *Property 1 does not mean that $j$ precedes $k$, $\forall j \in F_i^t, \forall k \in F_i^e$, since it is possible that $X_i \subseteq F_i^e$.*

**Property 2.** *For any $i = 0, 1, \ldots, m$ and for any job $j \in F_i^t$, if $b_j$ is the completion time of the job preceding immediately job $j$ (if $j$ is the first job in schedule $\sigma$, $b_j$ is set to 0), we have $C_j = \max(b_j + p_j, \bar{d}_i)$.*

**Proof.** From the definition of the set $F_i^t$ and the feasibility of schedule $\sigma$, we know that, for any $i = 0, 1, \ldots, m$ and for any job $j \in F_i^t$, we should have $C_j \geq \max(b_j + p_j, d_j)$. Suppose that the property is not true. In this case, then there are an $i$ and a $j \in F_i^t$ such that $C_j > \max(b_j + p_j, \bar{d}_i)$. If we construct a schedule $\sigma'$ such that $C'_j = \max(b_j + p_j, \bar{d}_i)$, the schedule is still feasible and we have $C_j > C'_j > d_j$, which implies that $f(\sigma') - f(\sigma) < 0$. This means that schedule $\sigma$ is not optimal, which is in contradiction with the assumption. □

This property means that, for any $i = 0, 1, \ldots, m$, jobs in $F_i^t$ are shifted to the left, and thus there is no idle time between them.

**Property 3.** *For any $i = 0, 1, \ldots, m$ and for any job $j \in F_i^e$, if $a_j$ is the starting time of the job following immediately job $j$ (if $j$ is the last job in schedule $\sigma$, $a_j$ is set to $+\infty$), we have $C_j = \min(a_j, \bar{d}_{i+1})$.*

**Proof.** Similar to the proof of Property 2. □

This Property means that, for any time window $i = 0, 1, \ldots, m$, jobs in $F_i^e$ are right shifted, and thus there is no idle time between them.

**Property 4.** *For any $i = 0, 1, \ldots, m$, and for any pair of* adjacent *jobs $j, k \in F_i^t \setminus X_i$ ($k$ follows $j$), we have $p_j/\beta_j \leq p_k/\beta_k$.*

**Proof.** From Property 1, we know that jobs in $F_i^t$ are scheduled together, for all $i = 0, 1, \ldots, m$. Assume that Property 4 does not hold. In this case, there are some $i$ and a pair of *adjacent* jobs $j, k \in F_i^t \setminus X_i$ ($k$ follows $j$) such that $p_j/\beta_j > p_k/\beta_k$. We construct another schedule $\sigma'$ by permuting these two jobs. In $\sigma'$, we have $C'_j = C'_k > C_j > d_j$ and $C'_k = C_j - p_j + p_k \geq \bar{d}_i + p_k \geq d_k + p_k > d_k$, which implies

$$f(\sigma') - f(\sigma) = \beta_j p_k - \beta_k p_j < 0.$$

This means that $\sigma$ is not an optimal schedule, which is in contradiction with the assumption. □

**Property 5.** *For any $i = 0, 1, \ldots, m$ and for any pair of* adjacent *jobs $j, k \in F_i^e \setminus X_i$ ($k$ follows $j$), we have $p_j/\alpha_j \geq p_k/\alpha_k$.*

**Proof.** Similar to the proof of Property 4. □

From Property 1, we can deduce that, in an optimal schedule jobs are scheduled in the the following order: $F_0^e, X_1, F_1^t \setminus X_1, F_1^e \setminus X_1, X_2, F_2^t \setminus X_2, \ldots, F_{i-1}^e \setminus X_{i-1}, X_i, F_i^t \setminus X_i, F_i^e \setminus X_i, \ldots, X_m, F_m^t \setminus X_m$. According to properties 2 and 3, jobs in $F_{i-1}^e \setminus X_{i-1}$ and those in $F_i^t \cup X_i$ ($i = 1, 2, \ldots, m$) are scheduled contiguously, that means that there is no idle time between them: They form a block.

From Properties 4 and 5, we can see that in an optimal schedule jobs completed on the left side of $\bar{d}_i$ (or before $d_i$), that is jobs in $F_{i-1}^e \setminus X_{i-1}$, are scheduled in nonincreasing order of $p_j/\alpha_j$ while those completed on the right side of $\bar{d}_i$ (or after $d_i$), except the cross-due-date one if any, that is jobs in $F_i^t \setminus X_i$, are scheduled in non decreasing order of $p_j/\beta_j$. Therefore each block is $V$-shaped, according to Kanet's terminology ([13]) (decreasing on the left and increasing on the

right). Since an optimal schedule is composed of several blocks, each of which is V-shaped, this schedule is then piece-wise V-shaped.

Other optimality conditions concern two jobs at the extremities of two successive time windows.

**Property 6.** *For any $i = 1, 2, \ldots, m$, if $j$ is the last job belonging to $F_{i-1} \setminus X_{i-1}$ and $k$ is the first job belonging to $F_i$, then following properties hold*

*(1). If $d_j \geq \bar{d}_{i+1}$ then $d_k > \bar{d}_{i-1}$.*

*(2). If $d_j \geq \bar{d}_{i+1}$ and $d_k = \bar{d}_i$, then $p_j/\alpha_j \geq p_k/\alpha_k$.*

*(3). If $d_j = d_i$ and $d_k \leq \bar{d}_{i-1}$, then $p_j/\beta_j \leq p_k/\beta_k$.*

It should be noticed that jobs $j$ and $k$ are adjacent jobs. It is then possible to maintain the schedule of other jobs unchanged by interchanging these jobs.

**Proof of (1).** Assume that (1) is not true, that is, there are an $i$ and two jobs $j, k$ such that $j$ (resp. $k$) is the last (resp. first) job in $F_{i-1} \setminus X_{i-1}$ (resp. $F_i$), $d_j \geq d_{i+1}$ and $d_k \leq d_{i-1}$. According to the assumptions of the Property, $C_j \leq d_i < \bar{d}_{i+1} \leq d_j$ and $C_k > d_i > d_{i-1} \geq d_k$. These latter relations imply that $j$ is early and $k$ is tardy. According to Properties 3 and 2, there is no idle time between them. This means that $C_k = C_j + p_j$. If we construct another schedule $\sigma'$ by interchanging jobs $j$ and $k$, the completion times of $j$ and $k$ are respectively $C'_j = C_k$ and $C'_k = C_j - p_j + p_k$. Since $j \in F_{i-1} \setminus X_{i-1}$, we have $C_j - p_j \geq \bar{d}_{i-1}$. With these relations, we obtain $C'_j \leq \bar{d}_{i+1} \leq d_j$ and $C'_k > C_j - p_j \geq \bar{d}_{i-1} \geq d_k$. This means that $j$ is still early and $k$ is still tardy. However, compared to schedule $\sigma$, the earliness of $j$ is reduced by $\alpha_j p_k$ while the tardiness of $k$ is reduced by $\beta_k p_j$. This means that $\sigma'$ gives a better solution than $\sigma$, which is in contradiction with the assumption that $\sigma$ is an optimal schedule. □

**Proof of (2).** Assume that (2) is not true, that is, there are an $i$ and two jobs $j, k$ such that $j$ (resp. $k$) is the last (resp. first) job in $F_{i-1} \setminus X_{i-1}$ (resp. $F_i$), $d_j \geq \bar{d}_{i+1}$, $d_k = d_i$ and $p_j/\alpha_j < p_k/\alpha_k$. With the same reasoning as in the proof of (1), we can see that in schedule $\sigma$, $j$ is early and $k$ is tardy. Consequently, according to Properties 3 and 2, there is no idle time between them. We now construct another schedule $\sigma'$ by interchanging these two jobs. We obtain $C'_j = C_k$ and $C'_k = C_j - p_j + p_k = C_k - p_j$. We know that $C'_j \leq d_{i+1} \leq d_j$. This last relation means that $j$ is still early. Compared to schedule $\sigma$, the earliness of $j$ is reduced by $\alpha_j p_k$. It is necessary to examine two cases: the case where job $k$ is tardy, and the case where it is early.

If $k$ is tardy in $\sigma'$, that is, $C'_k > d_k$, the tardiness of $k$ is reduced by $\beta_k p_j$. Considering the reduction of earliness of job $j$, $\sigma'$ provides a better solution than $\sigma$, which is in contradiction with the assumption that $\sigma$ is an optimal solution.

If $k$ is early in $\sigma'$, that is, $C'_k \leq d_k$, the difference between the tardiness in $\sigma'$ and earliness in $\sigma$ is $\alpha_k(d_k - C'_k) - \beta_k(C'_k - d_k) = \alpha_k p_j - (\alpha_k + \beta_k)(C'_k - d_k)$. Considering the fact that $k \in F_i$, which implies that $C'_k > \bar{d}_i = d_k$, this difference is less than $\alpha_k p_j$. Considering furthermore the reduction of earliness of $j$, the difference of criterion values of $\sigma'$ and $\sigma$ is $f(\sigma') - f(\sigma) \leq -\alpha_j p_k + \alpha_k p_j$. With the assumption that $p_j/\alpha_j < p_k/\alpha_k$, we obtain $f(\sigma') - f(\sigma) < 0$. This means that $\sigma$ is not an optimal schedule, which is in contradiction with the assumption. □

**Proof of (3).** The proof of (3) is quite similar to that of (2). Assume that (3) is not true, that is, there are an $i$ and two jobs $j, k$ such that $j$ (resp. $k$) is the last (resp. first) job in $F_{i-1} \setminus X_{i-1}$ (resp. $F_i$), $d_j = \bar{d}_i$, $d_k \leq \bar{d}_{i-1}$ and $p_j/\beta_j > p_k/\beta_k$. From the assumptions, we know that in $\sigma$, $j$ is early and $k$ is tardy. According to Properties 3 and 2, there is no idle time between them. When we construct another schedule $\sigma'$ by interchanging job $j$ and $k$, we have $C'_j = C_k$ and $C'_k = C_j - p_j + p_k$. Therefore, $C'_j > \bar{d}_i = d_j$ and $C'_k > \bar{d}_{i-1} + p_k > d_{i-1} = d_k$. The difference between the criterion values of $\sigma'$ and $\sigma$ is then

$$\begin{aligned} f(\sigma') - f(\sigma) &= \beta_j(C'_j - d_j) - \alpha_j(d_j - C_j) - \beta_k p_j \\ &< \beta_j(C_k - d_j) - \beta_k p_j. \end{aligned}$$

From Property 2 and the fact that $k$ is tardy in $\sigma$ and $k$ is the first job in $F_i$, we know that $C_k - p_k \leq \bar{d}_i = d_j$. As a consequence, $f(\sigma') - f(\sigma) < \beta_j p_k - \beta_k p_j$. The assumption $p_j/\beta_j > p_k/\beta_k$ leads to $f(\sigma') - f(\sigma) < 0$. This relation implies that $\sigma$ is not an optimal schedule, which is in contradiction with the assumption. □

This property means that in an optimal schedule, jobs are likely to be scheduled around their due dates.

**Property 7.** *If $\alpha_j = \beta_j = p_j$, $j = 1, 2, \ldots, n$ and if there exists an optimal schedule such that at least one job is processed within each time window, then there is an optimal schedule such that jobs with the same due date are scheduled around this due date.*

This property is a consequence of Properties 1—6.

# 4. Optimal Schedule for Given Completion Time Windows of Jobs: A Dynamic Programming Algorithm

In the previous section, we proved that an optimal schedule should be piece-wise V-shaped. From these results, we know that as soon as $F_i$ (i.e. the set of jobs completed within the $i$th time window) is given for each $i = 0, 1, \ldots, m$, it is possible to compute $F_i^t$ and $F_i^e$, where $F_i^t = \{j \in F_i | d_j \leq d_i\}$ and $F_i^e = \{j \in F_i | d_j \geq \bar{d}_{i+1}\}$. Consequently, if the cross-due-date jobs $X_i$ ($i = 1, 2, \ldots, m$) are given, we can decide partial precedence relations between jobs. Jobs in $F_i^t \setminus X_i$ should precede those in $F_i^e \setminus X_i$. Furthermore, jobs in $F_i^e \setminus X_i$ should be scheduled in non increasing order of $p_j/\alpha_j$. Jobs in $F_i^t \setminus X_i$ should be scheduled in non decreasing order of $p_j/\beta_j$. Therefore, if cross-due-date jobs are specified, the sequence is fully specified, and we can use the algorithm of Garey, Tarjan and Wilfong [8] or Davis and Kanet([4]) to insert idle times. In this section, we show that as soon as $F_i$ is given for each $i = 0, 1, \ldots, m$, it is possible to obtain an optimal schedule using a dynamic programming approach, without knowing the cross due date jobs *a priori*.

In the dynamic programming approach, we use the results obtained in the previous section saying that jobs in $F_{i-1}^e \setminus X_{i-1}$ and those in $F_i^t \cup X_i$ ($i = 1, 2, \ldots, m$) form a block, i.e. there is no idle time between these jobs.

Before giving the details of the dynamic programming algorithm, we have to solve the following subproblem: It consists of scheduling a block composed of $F_{i-1}^e \setminus X_{i-1}$ and $F_i^t \cup X_i$ for given $X_{i-1} \subseteq F_{i-1}, X_i \subseteq F_i$ and $\tau$, where $\tau + \bar{d}_i$ is the completion time of the last job of the block.

Using Properties 4 and 5, we know that jobs in $F'_{i-1} \setminus X_{i-1}$ are scheduled in non decreasing order of $p_j/\alpha_j$ while those in $F'_i \setminus X_i$ are scheduled in non decreasing order of $p_j/\beta_j$. Therefore, if a feasible solution exists for the given $X_{i-1}, X_i$ and $\tau$, the minimal total earliness-tardiness of jobs in this block is well defined. Let $z(i, \tau, u, v)$ denote this minimal value when $X_{i-1} = \{u\}$ and $X_i = \{v\}$. We also introduce a fictitious job 0 with $p_0 = 0$. For any $i$ such that $X_i = \emptyset$, we artificially set $X_i = \{0\}$. In order to simplify the explanations, we introduce the following notations related to the block composed of $\{v\} \cup F'_i \cup (F'_{i-1} \setminus \{u\})$.

$\rho_i(v) = \sum_{j \in F'_i \setminus \{v\}} p_j$: Total processing time of jobs completely processed on the right side (as indicated by "$\rho$") of $d_i$;

$\zeta_i(u, v) = \rho_i(v) + p_v + \sum_{j \in F'_{i-1} \setminus \{u\}} p_j$: Total processing time of the block.

The following notations are related to the $i$th time window

$\lambda_i = \max_{j \in F_i} p_j$: Longest processing time of jobs completed within the $i$th time window;

$P_i = \sum_{j \in F_i} p_j$: Total processing time of the jobs completed within the $i$th time window.

Knowing $i$, $u$, and $v$, we examine the conditions which should be verified by $\tau$ in order that a feasible solution exists. If $v \neq 0$, since $v$ should start before $\bar{d}_i$ and be completed after $d_i$, we should have,

$$\rho_i(v) < \tau < \rho_i(v) + p_v. \tag{4}$$

If $v = 0$ and $F'_i \neq \emptyset$, since the first job in $F'_i$ should start exactly at $\bar{d}_i$, it is necessary that

$$\tau = \rho_i(v). \tag{5}$$

If $v = 0$ and $F'_i = \emptyset$, since jobs in $F_{i-1} \setminus \{u\}$ should be completed within the $(i-1)$th time window, we should have,

$$P_{i-1} - p_u - \bar{d}_i + \bar{d}_{i-1} < \tau \leq 0. \tag{6}$$

If $v \neq 0$ or $F'_i \neq \emptyset$, the following condition must be satisfied, because the jobs in $F'_i \cup \{v\}$ have to be completed within the $i$th time window.

$$0 \leq \tau \leq \bar{d}_{i+1} - \bar{d}_i. \tag{7}$$

To sum up, $\tau$ should verify the following conditions in order that $z(i, \tau, u, v)$ exists.

$$\tau \in (\rho_i(v), \min\{\bar{d}_{i+1} - \bar{d}_i, \rho_i(v) + p_v + 1\}], \quad \text{if } v \neq 0, \tag{8}$$

$$\tau \in [\rho_i(v), \min\{\rho_i(v), \bar{d}_{i+1} - \bar{d}_i\}], \quad \text{if } v = 0 \text{ and } F'_i \neq \emptyset, \tag{9}$$

$$\tau \in (P_{i-1} - p_u - \bar{d}_i + d_{i-1}, 0], \quad \text{if } v = 0 \text{ and } F'_i = \emptyset. \tag{10}$$

If these conditions are not satisfied, no solution exists for the given $i, \tau, u$ and $v$. In that case, $z(i, \tau, u, v)$ is set to $+\infty$.

We now explain the dynamic programming algorithm. Informally speaking, the blocks are scheduled one after another. Let $q$ be the number of $i$'s ($1 \leq i \leq m$) such that $F_i \cup F'_{i-1} \neq \emptyset$ and $h_k$ ($k = 1, 2, \ldots, q$) be the $k$th smallest $i$ ($1 \leq i \leq m$) such that $F_i \cup F'_{i-1} \neq \emptyset$. $F_i \cup F'_{i-1}$ is

the union of the set of jobs completed within the $i$th time window and the set of jobs completed within the $(i-1)$th time window and which are early or on time. With these notations, we have $F_{h_1-1} = \emptyset$.

In the remainder of the section, $g(k, \tau, v)$ denotes the minimal total earliness+tardiness of jobs in $\{v\} \cup F^t_{h_k} \cup (\cup_{0 \le i < h_k} F_i)$, given that, the cross-due-date job of the $h_k$th time window is $X_{h_k} = \{v\}$, and the completion time of the last one of these jobs is $\tau + d_{h_k}$. With this definition, we obtain the following boundary condition

$$g(1, \tau, v) = z(1, \tau, 0, v). \tag{11}$$

because $F_{h_1-1} = \emptyset$.

For a solution to exist for the given $k, \tau$ and $v$, or, in otherwords, in order that $g(k, \tau, v)$ exists, it is necessary that there exists a $u$ such that $z(h_k, \tau, u, v) < +\infty$. Therefore, if $v \neq 0$ or $F^t_{h_k} \neq \emptyset$, relations (4) and (5) should be verified by replacing $i$ with $h_k$. Thus,

$$\tau \in (\rho_{h_k}(v), \min\{\bar{d}_{h_k+1} - \bar{d}_{h_k}, \rho_{h_k}(v) + p_v + 1\}]. \qquad \text{if } v \neq 0, \tag{12}$$

$$\tau \in [\rho_{h_k}(v), \min\{\rho_{h_k}(v), \bar{d}_{h_k+1} - \bar{d}_{h_k}\}]. \qquad \text{if } v = 0 \text{ and } F^t_{h_k} \neq \emptyset. \tag{13}$$

If $v = 0$ and $F^t_{h_k} = \emptyset$, there must be a $u$ such that $z(h_k, \tau, u, v) < +\infty$. As a consequence,

$$0 \ge \tau > P_{h_k-1} - \lambda_{h_k-1} - \bar{d}_{h_k} + d_{h_k-1}. \tag{14}$$

From the definition of $g$, the following recurrence formula holds:

$$g(k+1, \tau, v) = \min_{u \in F_{h_k} \cup \{0\}} \left[ \min_{\theta \in \Theta_k(\tau, u, v)} g(k, \theta, u) + z(h_{k+1}, \tau, u, v) \right], \tag{15}$$

where the set $\Theta_k(\tau, u, v)$ is defined hereafter. In this formula, it is possible that $u \notin F_{h_k-1}$ (in the case where $h_{k-1} < h_k - 1$, we should have $F_{h_k-1} = \emptyset$). In that case, without loss of generality, we set $z(i, \tau, u, v) = z(i, \tau, 0, v)$.

Since there should be no overlapping between the last job of the block and the first job of the next block, the following inequality holds:

$$\theta \le \eta_k(\tau, u, v), \tag{16}$$

where $\eta_k(\tau, u, v) = \tau + d_{h_k+1} - \bar{d}_{h_k} - \zeta_{h_k+1}(u, v)$.

To summarize, $\Theta_k(\tau, u, v)$ is defined as

$$(\rho_{h_k}(u), \min\{d_{h_k+1} - d_{h_k}, \rho_{h_k}(u) + p_u + 1, \eta_k(\tau, u, v)\}]. \qquad \text{if } u \neq 0, \tag{17}$$

$$[\rho_{h_k}(u), \min\{\rho_{h_k}(u), \bar{d}_{h_k+1} - d_{h_k}, \eta_k(\tau, u, v)\}]. \qquad \text{if } u = 0 \text{ and } F^t_{h_k} \neq \emptyset. \tag{18}$$

$$(P_{h_k-1} - \lambda_{h_k-1} - d_{h_k} + \bar{d}_{h_k-1}, \min\{0, \eta_k(\tau, u, v)\}]. \qquad \text{if } u = 0 \text{ and } F^t_{h_k} = \emptyset. \tag{19}$$

A little more should be said about the case where $u = 0$ and $F^t_{h_k} = \emptyset$. In this case, jobs in $F^e_{h_k-1}$ are early. The closer $\theta$ is to 0, the smaller the earliness of these jobs. In fact we can construct a schedule such that $\theta$ is as close to 0 as possible, since if there is a feasible schedule such that $\theta$ is not close to 0, we can obtain another feasible schedule by reducing $\theta$, the schedule of the jobs preceding jobs in $F^e_{h_k-1}$ being unchanged. In this latter schedule, the criterion value

in also reduced. Therefore, $\theta \leq 0$ should be as close to 0 as possible. From this remark, if $u = 0$ and $F^t_{h_k} = \emptyset$, $\Theta_k(\tau, u, v)$ is

$$\begin{cases} \emptyset, & \text{if } P_{h_k-1} - \lambda_{h_k-1} - d_{h_k} + d_{h_k-1} \geq \min\{0, \eta_k(\tau, u, v)\}. \\ \{\min[0, \eta_k(\tau, u, v)]\}. & \text{otherwise.} \end{cases} \tag{20}$$

From (17), (18) and (20), we can see that whatever the case, we have

$$|\Theta_k(\tau, u, v)| \leq p_{\max}. \tag{21}$$

where $p_{\max} = \max_{1 \leq j \leq n} p_j$.

We can see that it is possible that $\Theta_k(\tau, u, v) = \emptyset$. In this case, no solution is feasible for the given $k, \tau, u$ and $v$, and we set, by convention, $\min_{\theta \in \Theta_k(\tau,u,v)} g(k, \theta, u) = +\infty$.

The optimal criterion value for the given $\Phi = \{F_i | i = 0, 1, \ldots, m\}$, denoted by $ET(\Phi)$, is obtained when the last block is scheduled. We obtain,

$$ET(\Phi) = \min_{v \in F_{h_q} \cup \{0\}} \left[ \min_{\theta \in \Theta_q(+\infty, v, 0)} g(q, \theta, v) \right] \tag{22}$$

where,

$$\Theta_q(+\infty, v, 0) = \left( \rho_{h_q}(v), \min\left\{ d_{h_q+1} - d_{h_q}, \rho_{h_q}(v) + p_v \right\} \right). \tag{23}$$

if $v \neq 0$, and

$$\Theta_q(+\infty, v, 0) = \left\{ \rho_{h_q}(v) \right\}. \tag{24}$$

if $v = 0$.

Concerning the complexity of the algorithm, we can see that for each $k$ $(0 \leq k \leq m-1)$, the number of $v$'s (i.e. the number of cross due date jobs) is at most $|F_{h_{k+1}}|$ and the number of $\tau$'s is at most $p_{\max}$ (see (21)). Therefore the number of $g(k+1, \tau, v)$'s is at most $p_{\max} \sum_{k=1}^{m} |F_{h_k}| = np_{\max}$. For the computation of each $g(k+1, \tau, v)$, there are at most $n$ choices of $u$. The choices of $\theta$ can be avoided by memorizing $\min_{\theta \in \Theta_k(\tau,u,v)} g(k, \theta, u)$ during the computation of $g(k, \theta, u)$'s. Therefore, the complexity of the dynamic programming algorithm is $O(n^2 p_{\max})$.

## 5. Lower Bounds

In this section, we describe two methods to compute a lower bound of a complete schedule constructed from a partial solution.

**Note.** In this section, problems are denoted by a P followed by the equation numbers in parentheses. These equations define the objective function and the set of constraints of the problem. For instance, P(26 – 29) denotes the problem defined by criterion and constraints referred by (26) to (29).

Let us denote by $w_j$ the time window in which job $j$ should be completed, i.e. $w_j = i$ if and only if $d_i < C_j \leq d_{i+1}$.

According to Properties introduced in Section 3 and the dynamic programming approach described in Section 4, a complete schedule can be found using the dynamic programming approach proposed in Section 4, as soon as the set $\{w_j, j = 1, 2, \ldots, n\}$ is given. Similarly, a given partial solution can be represented by $\tilde{W}_l = \{\tilde{w}_j, j = 1, 2, \ldots, l\}$ with $1 \leq l < n$. From this partial solution, we can define the following sets, which are similar to those of Sections 3 and 4.

$\hat{F}_i^e = \{1 \le j \le l | \hat{w}_j = i, d_j > \bar{d}_i\}$: set of jobs assigned to the $i$th time window and being early or on time;

$\hat{F}_i^t = \{1 \le j \le l | \hat{w}_j = i, d_j \le \bar{d}_i\}$: set of jobs assigned to the $i$th time window and being tardy;

$J_i = \{l < j \le n | d_j = d_i\}$: set of jobs with due date $d_i$ and to which no time window is assigned;

$S_i = J_i \cup \hat{F}_i^t \cup \hat{F}_{i-1}^e$.

With these notations, the following equalities hold:

$$\cup_{1 \le i \le m} S_i = \{1, 2, \ldots, n\}, \cup_{1 \le i \le m} J_i = \{l+1, l+2, \ldots, n\}. \tag{25}$$

A partial solution being given, the problem of minimizing total earliness/tardiness can be formulated as follows.

$$\underset{\{C_j | 1 \le j \le n\}}{\text{Minimize}} \sum_{j=1}^{n} \max\{\alpha_j(d_j - C_j), \beta_j(C_j - d_j)\} \tag{26}$$

subject to

$$C_j \ge p_j. \qquad \forall 1 \le j \le n. \tag{27}$$

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset. \quad \forall j < k \le n, \forall 1 \le j \le n. \tag{28}$$

$$\bar{d}_{\hat{w}_j} < C_j \le \bar{d}_{\hat{w}_j + 1}. \qquad \forall 1 \le j \le l. \tag{29}$$

Constraints (27) and (28) are the same as (2) and (3). Constraints (29) are introduced to take into account the partial solution.

The remainder of the section provides two methods to obtain lower bounds of this problem.

## 5.1. First Method

In this subsection, we give a method based on the idea to separate the jobs to which time windows are assigned by the partial solution, i.e. the jobs $1 \le j \le l$, and those to which no time window is assigned, taking into account the influence of the previous jobs.

Before giving the details of the method, we introduce a quite obvious lemma.

**Lemma 1.** *Consider a problem defined as follows*

$$\left\{ x \left| \xi_k(x) \ge 0, k \in \bigcup_{1 \le i \le \mu} \Omega_i \right. \right\} \quad \underset{}{\text{Minimize}} \quad \sum_{i=1}^{\mu} \varphi_i(x).$$

*and its $\mu$ subproblems,*

$$\underset{\{x | \xi_k(x) \ge 0, k \in \Omega_i\}}{\text{Minimize}} \quad \varphi_i(x).$$

*The sum of the optimal criterion values of the $\mu$ subproblems is a lower bound of the optimal criterion value of the initial problem.*

We introduce other decision variables denoted by $w_j$ $(1 \le j \le n)$. Variable $w_j$ is the time window assigned to job $j$. Furthermore, $X_i$ still represents the set of cross due date jobs related to the $i$th time window. From $w_j$'s, we can also define the sets $F_i^t$ and $F_i^e$ $(0 \le i \le m)$ in the

same way as the definition of $\hat{F}_i^l$ and $\hat{F}_i^e$ $(0 \le i \le m)$ from $\hat{w}_j$'s at the beginning of the section, that is, $F_i^e = \{1 \le j \le n | w_j = i, d_j > \bar{d}_i\}$. $F_i^l = \{1 \le j \le n | w_j = i, d_j \le \bar{d}_i\}$. The set of constraints related to these new decision variables are the following

$$w_j = \hat{w}_j, \quad \forall j \in \{\omega \in S_l | 1 \le \omega \le l\}, \forall 1 \le i \le m. \tag{30}$$

$$w_j = i, \quad \forall j \in X_i, \forall 1 \le i \le m. \tag{31}$$

$$0 \le w_j \le m, \quad \forall j \in S_i, 1 \le i \le m. \tag{32}$$

where contraints (30) take into account the given partial solution.

The following constraints are derived from the definitions of the decision variables:

$$\bar{d}_{w_j} < C_j \le \bar{d}_{w_j+1}, \quad \forall j \in \{\omega | w_\omega = i\}, 1 \le i \le m. \tag{33}$$

$$\bar{d}_{w_j} < C_j < \bar{d}_{w_j} + p_j, \quad \forall j \in X_i, \forall 1 \le i \le m. \tag{34}$$

$$j \in X_i, \quad \forall j \in \{\omega | w_\omega = i, d_i < C_\omega < d_i + p_\omega\}, \forall 1 \le i \le m. \tag{35}$$

Since constraints (29) can be derived from constraints (30) and (33), they are neglected in the remainder of this subsection.

It should be noticed that the optimal criterion values of problem P(26—29) and of problem P(26—28, 30—35) are the same, because from an optimal solution $\{C_j | 1 \le j \le n\}$ of problem P(26—29), we can derive the decision variables $w_j$'s and $X_i$'s that verify all constraints of problem P(26—28, 30—35), and vice versa.

From Properties 4 and 5, we know that an optimal solution should verify the following conditions,

$$C_j < C_k, \quad \forall k \in \{\omega \in F_{i-1}^e \setminus X_{i-1} | \omega \ne j, p_\omega/\alpha_\omega < p_j/\alpha_j\}, \quad \forall j \in F_{i-1}^e \setminus X_{i-1}, \quad \forall 1 \le i \le m.$$
$$C_j < C_k, \quad \forall k \in \{\omega \in F_i^l \setminus X_i | \omega \ne j, p_\omega/\beta_\omega > p_j/\beta_j\}, \quad \forall j \in F_i^l \setminus X_i, \quad \forall 1 \le i \le m.$$

Therefore, the following constraints can be added, since they are a subset of the previous conditions,

$$C_j < C_k, \quad \forall k \in \{\omega \in L_i | \omega \ne j, p_\omega/\alpha_\omega < p_j/\alpha_j\}, \forall j \in L_i, \forall 1 \le i \le m. \tag{36}$$

$$C_j < C_k, \quad \forall k \in \{\omega \in R_i | \omega \ne j, p_\omega/\beta_\omega > p_j/\beta_j\}, \forall j \in R_i, \forall 1 \le i \le m. \tag{37}$$

where $L_i = S_i \cap (F_{i-1}^e \setminus X_{i-1})$ (resp. $R_i = S_i \cap (F_i^l \setminus X_i)$) is the set of jobs in $S_i$ completely processed within the time window on the left (resp. right) of $\bar{d}_i$, that is, the $(i-1)$th (resp. $i$th) time window. It should be noticed that all jobs in $R_i$ are tardy and all jobs in $L_i$ are early or on time.

We now establish the lower bounds for the problem P(26—28, 30—37). From this problem, we construct two subproblems: The first one is

$$\underset{\{W_j, X_i, C_j\}}{\text{Minimize}} \sum_{j=1}^{l} \max\{\alpha_j(d_j - C_j), \beta_j(C_j - d_j)\}$$

subject to (27), (30)—(37) and the following capacity constraints which constitute a subset of constraints (28)

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset, \quad \forall j < k \le l, \forall 1 \le j \le l.$$

The other one is:

$$\underset{\{F_i, X_i, C_j\}}{\text{Minimize}} \sum_{i=1}^{m} \sum_{j \in J_i} \max\{\alpha_j(d_j - C_j), \beta_j(C_j - d_j)\} \tag{38}$$

subject to (27)—(28) and (30)—(37).

From Lemma 1, we know that the sum of the optimal criterion values of these two problems provides a lower bound to problem P(26—28, 30—37). The first problem is just the scheduling problem of jobs in $\{j | 1 \leq j \leq l\}$ given the time window of each job. This problem can be solved by the dynamic programming approach described in Section 4.

The second problem, that is P(38, 27—28, 30—37), is still NP-hard, since it is a generalization, because of constraints (30), of the original problem (i.e. the scheduling problem to minimize total weighted earliness/tardiness with different due dates), which is known to be NP-hard. Therefore, we have to relax additional constraints in order to obtain a lower bound of this problem. Contraints (27) are removed. The capacity constraints (28) are relaxed for jobs belonging to different $S_i$'s. They then become:

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset. \qquad \forall k \in \{\omega \in S_i | \omega > j\}, \forall j \in S_i, \forall 1 \leq i \leq m. \tag{39}$$

Constraints (33) are relaxed to the following ones

$$C_j \geq \bar{d}_i, \quad \forall j \in F_i', \forall 1 \leq i \leq m. \tag{40}$$

$$C_j \leq \bar{d}_i, \quad \forall j \in F_{i-1}'', \forall 1 \leq i \leq m. \tag{41}$$

It is *very important* to notice that with this relaxation, some decision variables lose their original physical meaning. For instance, $w_j$ does not mean any more the time window in which job $j$ should be completed, because even if $w_j = i$, it is possible that $C_j > d_{i+1}$ or $C_j \leq d_i$. $w_j$ now means that job $j$ should be completed after or at $\bar{d}_{w_j}$ (i.e. $C_j \geq \bar{d}_{w_j}$) if $d_j \leq \bar{d}_{w_j}$ and it should be completed before or at $d_{w_j}$ (i.e. $C_j \leq d_{w_j}$) if $d_j > \bar{d}_{w_j}$. Consequently, $L_i$ and $R_i$ lose their physical meaning. $L_i$ (resp. $R_i$) is just the set of jobs defined by $L_i = \{j \in S_i | w_j = i - 1, j \notin X_{i-1}, d_j > \bar{d}_{i-1}\}$ (resp. $R_i = \{j \in S_i | w_j = i, j \notin X_i, d_j \leq \bar{d}_i\}$). However, considering constraints (40) and (41), we always have the remark that all jobs in $R_i$ are tardy and all jobs in $L_i$ are early or on time.

The relaxed problem becomes therefore P(38, 30—32, 34—37, 39—41). For this problem, neglecting the physical meaning of decision variables, the following property holds.

**Property 8.** *There is an optimal solution for the problem P(38, 30—32, 34—37, 39—41) such that for any $i$ ($1 \leq i \leq m$) and for any job $j \in S_i$, we have $w_j \in \{i - 1, i\}$.*

**Proof.** The proof can be found in Appendix A. □

Furthermore, we can easily prove that $X_i = \emptyset, \forall 1 \leq i \leq m$, since constraints (27) are removed as in the case of the unrestricted version of common due date problem. Therefore, constraints (31), (34) and (35) can be removed. As a consequence of Property 8, there is an optimal solution such that $F_{i-1}'' = L_i = \{j \in S_i | w_j = i - 1\}$, $F_i' = R_i = \{j \in S_i | w_j = i\}$ and $L_i \cup R_i = S_i$. The problem then becomes P(38, 30, 32, 36—37, 39—41) with an additional set of constraints: $w_j \in \{i - 1, i\}, \forall j \in S_i, \forall 1 \leq i \leq m$. The criterion and the constraints of this problem are decomposable. The problem can then be decomposed into $m$ independent subproblems. Each of them is related to a due date $d_i$ ($i = 1, 2, \ldots, m$), and the sum of optimal criterion values of

these problems gives the optimal criterion value of problem P(38, 30 – 32, 34 – 37, 39 – 41). The subproblem related to the due date $d_i$ is as follows:

$$\underset{\{C_j, L_i\}}{\text{Minimize}} \sum_{j \in J_i} \max\{\alpha_j(d_i - C_j), \beta_j(C_j - d_i)\} \tag{42}$$

subject to

$$j \in L_i, \qquad \forall j \in S_i \text{ such that } \hat{w}_j = i - 1, \tag{43}$$

$$j \in R_i, \qquad \forall j \in S_i \text{ such that } \hat{w}_j = i, \tag{44}$$

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset, \quad \forall k \in \{\omega \in S_i | \omega > j\}, \forall j \in S_i, \tag{45}$$

$$C_j \leq \bar{d}_i, \qquad \forall j \in L_i, \tag{46}$$

$$C_j \geq \bar{d}_i, \qquad \forall j \in R_i, \tag{47}$$

$$C_j < C_k, \qquad \forall k \in \{\omega \in L_i | \omega \neq j, p_w/\alpha_w < p_j/\alpha_j\}, \forall j \in L_i, \tag{48}$$

$$C_j < C_k, \qquad \forall k \in \{\omega \in R_i | \omega \neq j, p_w/\beta_w > p_j/\beta_j\}, \forall j \in R_i, \tag{49}$$

where $R_i = S_i \setminus L_i$. This problem is identical to the problem of minimizing total weighted earliness/tardiness with an *unrestricted* common due date, except for the objective function (42) and constraints (43) and (44), where constraints (48) and (49) mean that the optimal schedule should be V-shaped. Therefore all available algorithms to solve general unrestricted version of common due date E/T problems apply to solve the problem after slight modifications. However, no algorithm exists to solve the problem with arbitrary earliness and tardiness penalties. The complexity about this problem is still open. But when the jobs ordering according to the ratios $p_j/\alpha_j$ is the same as the job ordering according to the ratios $p_j/\beta_j$, for instance $\beta_j = \gamma\alpha_j$, for $1 \leq j \leq n$, where $\gamma > 0$, the algorithm of Hall and Posner [11] is applicable to solve problem P(42—49) with slight modifications taking into account objective function (42) and constraints (43) and (44).

## 5.2. Second Method

In this subsection, another method to obtain a lower bound of problem P(26 – 28) is presented. Unlike what has been done in subsection 5.1, jobs to which a time window is assigned as well as the remaining jobs are considered simultaneously in the objective function.

The lower bound is also obtained by relaxing constraints. As in the first method, constraints (27) are relaxed and the capacity constraints (28) are relaxed for the jobs belonging to different $S_i$'s. They become:

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset, \qquad \forall k \in \{\omega \in S_i | \omega > j\}, \forall j \in S_i, \forall 1 \leq i \leq m. \tag{50}$$

Constraints (29) are replaced by the following relaxed ones:

$$C_j \geq \bar{d}_i, \quad \forall j \in \hat{F}_i^t, \forall 1 \leq i \leq m. \tag{51}$$

$$C_j \leq \bar{d}_i, \quad \forall j \in \hat{F}_{i-1}^t, \forall 1 \leq i \leq m. \tag{52}$$

The relaxed problem is then P(26, 50– 52). Since $\cup_{1 \leq i \leq m} S_i = \{1, 2, \ldots, n\}$ and $S_{i'} \cap S_i = \emptyset, \forall i < i' \leq m, \forall 1 \leq i \leq m$, the objective function (26) can be rewritten as

$$\underset{\{C_j\}}{\text{Minimize}} \sum_{i=1}^{m} \sum_{j \in S_i} \max\{\alpha_j(d_j - C_j), \beta_j(C_j - d_j)\} \tag{53}$$

The objective function and the constraints of the problem P(50--53) can be decomposed into $m$ parts, each of them being related to a due date $\bar{d}_i$ $(1 \leq i \leq m)$. Therefore, the problem can be also decomposed into $m$ independent subproblems, each of them being related to a due date $\bar{d}_i$ $(1 \leq i \leq m)$. The sum of the optimal criterion values of these subproblems is the optimal criterion value of problem P(50--53). The subproblem related to the due date $\bar{d}_i$ is as follows.

$$\underset{\{C_j\}}{\text{Minimize}} \sum_{j \in S_i} \max\{\alpha_j(d_j - C_j), \beta_j(C_j - d_j)\} \tag{54}$$

subject to

$$[C_j - p_j, C_j) \cap [C_k - p_k, C_k) = \emptyset, \quad \forall k \in \{\omega \in S_i | \omega > j\}, \forall j \in S_i, \tag{55}$$

$$C_j \geq d_i, \qquad \forall j \in \hat{F}_i^t, \forall 1 \leq i \leq m, \tag{56}$$

$$C_j \leq d_i, \qquad \forall j \in \hat{F}_{i-1}^e, \forall 1 \leq i \leq m. \tag{57}$$

Constraints (56) and (57) can be integrated into the objective function (54) by applying a large penalty. The new objective function becomes:

$$\begin{aligned} \underset{\{C_j\}}{\text{Minimize}} \quad & \sum_{j \in J_i} \max\{\alpha_j(d_i - C_j), \beta_j(C_j - d_i)\} + \sum_{j \in \hat{F}_{i-1}^e} \max\{\alpha_j(d_i - C_j), M(C_j - \bar{d}_i)\} + \\ & \sum_{j \in \hat{F}_i^t} \max\{M(d_i - C_j), \beta_j(C_j - \bar{d}_i)\} + ET \end{aligned} \tag{58}$$

where $M$ is a very large number, and $ET = \sum_{j \in \hat{F}_i^t} \beta_j(\bar{d}_i - d_j) + \sum_{j \in \hat{F}_{i-1}^e} \alpha_j(d_j - \bar{d}_i)$.

Therefore subproblem P(54--57) becomes P(58, 55). Since $ET$ is constant (i.e. independent of decision variables), this latter problem is simply the scheduling problem to minimize total weighted earliness-tardiness with a common due date, where, for jobs in $\hat{F}_i^t$ (resp. $\hat{F}_{i-1}^e$), the earliness (resp. tardiness) penalty is $M$. As mentioned in the first method, when the job ordering in the ratio $p_j/\alpha_j$ is the same as the job ordering in the ratio $p_j/\beta_j$, for instance $\beta_j = \gamma\alpha_j$, for $1 \leq j \leq n$, where $\gamma > 0$, the algorithm of Hall and Posner [11] is applicable to solve this problem.

These two lower bounds are used in the branch and bound algorithm presented in the next section.

## 6. Branch and Bound Algorithm

We now present the main features of our branch and bound algorithm. The algorithm implicitly enumerates completion time windows of jobs as described by $\{w_j, j = 1, 2, \ldots, l\}$ in the previous section.

An initial upper bound on the cost of an optimal schedule is generated by a greedy heuristic. In this heuristic, jobs are scheduled one after another. At each iteration $j$, job $j$ is scheduled taking into account the partial schedule composed of jobs $1, 2, \ldots, j-1$. At this iteration, we expect that job $j$ is scheduled around its due date, that is if $d_j = \bar{d}_i$ it is expected to be completed either within the $(i-1)$th time window or within the $i$th time window. This gives rise to two choices. For each of the choices, we know the expected completion time window for each job in $\{1, 2, \ldots, j\}$. Therefore, we can construct sets $F_i^t$'s and $F_i^e$'s. By Assuming $X_i = \emptyset, \forall 1 \leq i \leq m$, we can completely specify the sequence of these according to Properties 1 to 5. Idle times then

Table 1: Computational results for the branch and bound algorithm

| No. | E1/A | E1/B | E2/A | E2/B | E3/A | E3/B | E4/A | E4/B | E5/A | E5/B |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 23 | * | 10 | 271 | 2 | 46 | 1 | * | 10 | 195 |
| 2 | 138 | 181 | 241 | 153 | 192 | 107 | * | 125 | 27 | 140 |
| 3 | 108 | * | * | * | 204 | * | 97 | 240 | 8 | * |
| 4 | 26 | 121 | 261 | 242 | * | 184 | 298 | 163 | 36 | 152 |
| 5 | 22 | 47 | 28 | 51 | 14 | 156 | 3 | * | 39 | 126 |
| 6 | 78 | * | 38 | 141 | 29 | 60 | 100 | 245 | 119 | 134 |
| 7 | 31 | 85 | 35 | 90 | 50 | 18 | 13 | 175 | 53 | 148 |
| 8 | 45 | 81 | 12 | 31 | 4 | 21 | 9 | 250 | 27 | * |
| 9 | 81 | * | 20 | * | 27 | * | 3 | 202 | 14 | 124 |
| 10 | * | 240 | 280 | 107 | * | 51 | 91 | 118 | * | 162 |

A: lower bound computed by the first method;
B: lower bound computed by the second method;
E1: examples for $\beta_j = \gamma\alpha_j$ with $\gamma = 2$;
E2: examples for $\beta_j = \gamma\alpha_j$ with $\gamma = 1$;
E3: examples for $\beta_j = \gamma\alpha_j$ with $\gamma = 0.5$;
E4: examples for $\beta_j = \alpha_j = p_j$;
E5: examples for $\alpha_j = 1, \beta_j = 10$;
*: examples for which the computation time exceeds 300 seconds.

can be inserted using the algorithm of Garey, Tarjan and Wilfong [8]. This provides a cost for each choice of expected completion time window for job $j$. The final expected completion time window of job $j$ is chosen so that the corresponding cost is the least.

Other aspects of the branch and bound algorithm are the following. For nodes that are not eliminated in the enumeration tree of the algorithm, a lower bound is computed using one of the methods presented in the previous section. In the algorithm, we use the depth-first-search plus best lower bound rule to select a node to be examined next.

## 7. Computational experience

The branch and bound algorithm was tested on earliness-tardiness scheduling problems with different due dates with $n = 20$. $m$ was randomly generated using the uniform distribution on [2,10]. Processing times of jobs were randomly generated using the uniform distribution on [1,10]. $\alpha_j$'s and $\beta_j$'s are generated with three different schemes: 1) $\beta_j = \gamma\alpha_j$, for $j = 1, 2, \ldots, n$ where $\alpha_j, j = 1, 2, \ldots, n$ were randomly generated from the uniform distribution [1,5] and $\gamma$ was taken from the set $\{0.5, 1, 2\}$; 2) $\alpha_j = \beta_j = p_j$ for $j = 1, 2, \ldots, n$; 3) $\alpha_j = 1, \beta_j = 10$ for $j = 1, 2, \ldots, n$. Due dates $d_i, i = 1, 2, \ldots, m$ were randomly generated using the uniform distribution $\sum_{k=1}^{i-1} A_k + U[0.5A_i, A_i]$, where $U[.,.]$ denotes the uniform distribution. $A_i$ is the total processing time of jobs with due date $\bar{d}_i$. The algorithm was coded in C language and run on an IBM 6091-19 microcomputer. Ten problems were generated for each type of $\alpha_j$'s and $\beta_j$'s. Whenever a problem could not be solved within the time limit of 300 seconds (5 minutes), computation was abandoned for that problem. Computation times in seconds in order to obtain optimal solutions for these problems are given in Table 1. The influence of the lower bounds needs more comments. The computational experience shows that the computation of lower

bounds using the first method requires a little more time than that with the second method, but the lower bound provided by the first method is frequently a little better than the one provided by the second method. As a result, the average computation time for the algorithm using the first method is less than the computation time when using the second method.

## 8. Conclusion

In this paper, some properties about optimal solutions of the earliness-tardiness scheduling problems with different due dates are exploited and a branch and bound algorithm is proposed for solving the problems. Computational results show that the branch and bound algorithm can solve the problem up to $n = 20$ with modest run times.

Since weighted earliness-tardiness scheduling problems tend to be very difficult when jobs have different due dates, much investigation is needed further to develop more efficient algorithms to attack large size problems.

## A. Proof of Property 8

Consider an *optimal* schedule which does not verify Property 8. There is at least one $i$ such that the set $B_i \cup A_i \neq \emptyset$, where $B_i = \{j \in S_i | 0 \leq w_j < i - 1\}$ and $A_i = \{j \in S_i | i < w_j \leq m\}$.

Consider the case where $A_i \neq \emptyset$. It should be noticed that $A_i \subseteq J_i$, since otherwise constraints (30) would be violated. Then, for any $j \in A_i$, $d_j = \bar{d}_i \leq d_{w_j}$, which implies that $j \in F'_{w_j}$. According to (40), we have,

$$C_j \geq \bar{d}_{w_j} > \bar{d}_i = d_j. \qquad \forall j \in A_i. \tag{59}$$

Let $k \in A_i$ be the job with the smallest completion time. We consider two cases.

**Case 1.** $p_k/\beta_k \geq p_j/\beta_j, \forall j$ such that $j \in R_i$ and $C_j < C_k$.

We construct another solution identical to the one considered currently, except that $w_k = i$ in the new solution. This solution is feasible, since all the constraints are verified. Furthermore, this new solution does not increase the criterion value. However in the new solution, the set $A_i$ is reduced by one element ($k$ does not belong to $A_i$ any more).

**Case 2.** The hypothesis of Case 1 does not hold

In this case, it is possible to find a job $j \in R_i$ with the smallest completion time such that $C_j < C_k$ and $p_j/\beta_j > p_k/\beta_k$. Let $Q \subseteq S_i$ be the set of jobs whose completion times are between $C_j$ and $C_k$, including job $j$. From the notations, we have, $w_\omega = i, \forall \omega \in Q$. As a consequence, $Q \subseteq R_i$ and from (37) $p_\omega/\beta_\omega \geq p_j/\beta_j > p_k/\beta_k, \forall \omega \in Q$. Construct another solution such that $w'_k = i$, $C'_\omega = C_\omega + p_k, \forall \omega \in Q$ and $C'_k = C_j - p_j + p_k \geq d_i + p_k = \bar{d}_{w'_k} + p_k > d_{w'_k}$ (since $j \notin X_i$). This new solution is feasible, since all the constraints are verified. Furthermore, in this new solution, $A_i$ is reduced by one element ($k$ does not belong to $A_i$ any more). Considering relations $\forall \omega \in Q \subseteq R_i, C_\omega > d_\omega$, the fact that $C'_k > d_k$ (since $k \in A_i$, see (59)) and $C'_k \geq d_i + p_k > d_i = d_k$ (since $k \in A_i$, see (59)), if $\Delta$ denotes the difference between the criterion values of the new and the previous solutions, we obtain,

$$\Delta \leq -\beta_k \sum_{\omega \in Q} p_\omega + p_k \sum_{\omega \in Q, \omega > i} \beta_\omega$$

$$\leq \sum_{\omega \in Q, \omega > i} (p_i - p_\omega)$$
$$\leq 0.$$

which means that the criterion value is not increased.

Therefore, whatever the case is, we can obtain a schedule, without increasing the criterion value, and in which the set $A_i$ is reduced by one element. We can repeat the same process until $A_i = \emptyset, \forall 1 \leq i \leq m$.

If there is an $i$ such that $B_i \neq \emptyset$, we can use a similar process to reach a solution, without increasing the criterion value and in which $B_i = \emptyset, \forall 1 \leq i \leq m$.

Consequently, from the initial solution, we can derive a solution verifying the property, without increasing the criterion value. This means that either the new solution is also optimal (in case the criterion value remains unchanged) or the initial solution is not optimal (if the new criterion value is strictly less than the initial one), which is in contradiction with the assumption.

## References


[1] T. Abdul-Razaq, C. Potts, "Dynamic programming state-space relaxation for single-machine scheduling", *J. Opnl. Res. Soc.*, Vol. **39**, 1988, pp. 141-152.

[2] K.R. Baker, G.D. Scudder, "Sequencing with earliness and tardiness penalties: A review", *Operations Research*, Vol. **38**, No.1, 1990, pp. 22-36.

[3] J. Davis, J. Kanet, "Single machine scheduling with a nonregular convex performance measure", Working paper, Department of Management, Clemson University, Clemson, S.C.

[4] J. Davis, J. Kanet, "Single machine scheduling with early and tardy completion costs", *Nav. Res. Logist.*, Vol. **40**, 1993, pp. 85-101.

[5] H. Emmons, "Scheduling to a common due date on parallel common processors", *Naval Res. Logist. Quart.* Vol. **34**, 1987, pp. 803-810.

[6] T. Fry, R. Armstrong and J. Blackstone, "Minimizing weighted absolute derivation in single machine scheduling", *IIE Transactions*, Vol. **19**, 1987, pp. 445-450.

[7] T. Fry, K. Darby-Dowman and R. Armstrong, "Single machine scheduling to minimize mean absolute lateness", Working paper, College of Business Administration, University of South Carolina, Columbia.

[8] M. Garey, R. Tarjan and G. Wilfong, "One-processor scheduling with symmetric earliness and tardiness penalties", *Math. Opns. Res.*, Vol. **13**, 1988, pp. 330-348.

[9] D.Y. Golhar, C.L. Stamm, "The JIT philosophy: A review", *Int. J. of Production Research*, Vol. **29**, No.4, 1991, pp. 657-676.

[10] N. Hall, W. Kubiak, S. Sethi, "Earliness-tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date", *Operations Research*, Vol. **39**, No.5, 1991, pp. 847-856.

[11] N. Hall, M. Posner, "Earliness-tardiness scheduling problems, I: Weighted deviation of completion times about a common due date", *Operations Research*, Vol. **39**, No. 5, 1991, pp. 836-846.

[12] T. Ibaraki, Y. Nakamura, "A dynamic programming method for single machine scheduling", *European J. of Opnl. Res.*, Vol. **76**, 1994, pp. 72-84.

[13] J. Kanet, "Minimizing the average derivation of job completion times about a common due date", *Naval Res. Logist. Quart.* Vol. **28**, 1981, pp. 643-651.

[14] Y. Kim, C. Yano, "Minimizing mean tardiness and earliness in single machine scheduling problems with unequal due dates", *Naval Research Logistics*, Vol. **41**, 1994, pp. 913-933.

Unité de recherche INRIA Lorraine
Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)

Unité de recherche INRIA Rennes - IRISA, Campus universitaire de Beaulieu 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes - 46, avenue Félix Viallet - 38031 Grenoble Cedex 1 (France)
Unité de recherche INRIA Rocquencourt - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis - 2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)

ISSN 0249 - 6399