COMPUTATIONAL & APPLIED MATHEMATICS

Volume 29, N. 2, pp. 107–124, 2010

ISSN 0101-8205
www.scielo.br/cam

# The single machine earliness and tardiness scheduling problem: lower bounds and a branch-and-bound algorithm*

DÉBORA P. RONCONI and MÁRCIO S. KAWAMURA

Department of Production Engineering, EPUSP, University of São Paulo
Av. Prof. Almeida Prado, 128, Cidade Universitária, 05508-900, São Paulo SP, Brazil

E-mails: dronconi@usp.br / marcio.kawamura@poli.usp.br

**Abstract.** This paper addresses the single machine scheduling problem with a common due date aiming to minimize earliness and tardiness penalties. Due to its complexity, most of the previous studies in the literature deal with this problem using heuristics and metaheuristics approaches. With the intention of contributing to the study of this problem, a branch-and-bound algorithm is proposed. Lower bounds and pruning rules that exploit properties of the problem are introduced. The proposed approach is examined through a computational comparative study with 280 problems involving different due date scenarios. In addition, the values of optimal solutions for small problems from a known benchmark are provided.

**Mathematical subject classification:** 90C11, 62P30, 90B35.

**Key words:** single machine, common due date, earliness and tardiness, lower bound, branch-and-bound.

## 1 Introduction

Scheduling problems involving both earliness and tardiness costs have received significant attention in recent years. This type of problem became more important with advent of lean production principles, including the just-in-time (JIT)

#CAM-107/09. Received: 18/VI/09. Accepted: 26/XI/09.

*This research has been financially supported by "Fundação de Amparo à Pesquisa do Estado de São Paulo" – FAPESP (Grant 06/03496-3) and by "Conselho Nacional de Desenvolvimento Científico e Tecnológico" – CNPq (Grants 486124/2007-0 and 307399/2006-0).

concept. According to JIT, earliness and tardiness are considered harmful to profitability and, for this reason, must be minimized: tardiness causes loss of customer goodwill and damage reputation, as well as delay of payments, while earliness causes inventory carrying costs and possible loss of product quality. Probably based on this motivation, many authors have considered the scheduling problem aiming to minimize earliness and tardiness in the delivery of goods. Comprehensive surveys on the common due date assignment and scheduling problems can be found in [8, 2].

The problem of scheduling jobs with a common due date in a single machine has been studied by several authors. From a practical point of view, according to [7], customer orders with a combination of goods to be delivered at a specified time, export shipping and chemical or physical mixtures containing some ingredients with a short half-life period are some examples where jobs are to be scheduled in a single machine and be delivered on a common due date.

In this problem, there are $n$ jobs available at time zero to be processed on a single machine and to be delivered on a common due date $d$. Each job $i$ requires exactly one operation and its processing time $p_i$ is known. If a job $i$ is completed before the due date, its earliness is given by $E_i = d - C_i$, where $C_i$ is the completion time of job $i$. Conversely, if a job $i$ is completed after the desired date, its tardiness is given by $T_i = C_i - d$. Each job $i$ has its own unit earliness penalty $\alpha_i$ and unit tardiness penalty $\beta_i$. Preemption is not allowed and the initial processing time is not necessarily at time zero, when all jobs are available. The objective of the problem is to obtain an optimal schedule that minimizes the sum of earliness and tardiness penalties.

The common due date can be restrictive or unrestrictive. A due date is called unrestrictive if its optimal value has to be calculated or if its given value does not influence the optimal schedule. In the cases where the given due date is greater than or equal to the sum of processing times of all jobs available, this due date is unrestrictive [7]. In [9] it was demonstrated that this scheduling problem is NP-hard even with the unrestrictive due date and $\alpha_i = \beta_i$. In [15] the author addressed the particular unrestrictive case in which $\alpha_i = \beta_i = 1$ for all jobs, that can be solved by a polynomial algorithm of $O(n \log n)$ complexity. The case in which penalties are independent of the jobs ($\alpha_i = \alpha$ and $\beta_i = \beta$ for

all $i$) can also be treated by polynomial algorithms [21]. For the general case (no restrictions on the penalties), a branch-and-bound algorithm that is capable of solving instances with up to 15 jobs was proposed in [6]. By that time, a 0-1 quadratic model for this problem was presented and solved with a specialized branch-and-bound algorithm [5]. Another exact approach that combined column generation with a Lagrangean relaxation algorithm was presented in [1]. This strategy solves problems with up to 125 jobs.

The restrictive version of this problem is NP-hard even with $\alpha_i = \beta_i = 1$ [10, 12]. Due to its complexity, many authors addressed this problem using heuristic and metaheuristic approaches (see, for example, [13, 17, 7, 11, 18]). Most of these methods based their search strategies on the following properties.

For the restrictive common due date case with general penalties, there is an optimal solution with these properties:

1. No idle times are inserted between consecutive jobs [4];
2. The schedule is V-Shaped, that is, jobs that complete on or before the due date are sequenced in a non-increasing order of the $p_i/\alpha_i$ ratio. The jobs that start on or after the due date are sequenced in a non-decreasing order of the $p_i/\beta_i$ ratio. Note that there may be a straddling job, i.e., a job whose processing is started before and finished after the due date (see [12, 3]);
3. There is an optimal schedule in which either the processing time of the first job starts at time zero or one job is completed on the due date. The proof is similar to the one presented in [12].

The development of exact algorithms for the restrictive common due date case with special characteristics was considered by some authors. A pseudopolynomial dynamic programming algorithm for this problem with $\alpha_i = \beta_i$ for all jobs was presented in [12], while a pseudopolynomial algorithm for the constraint problem (machine idle time is forbidden) with $\alpha_i = \alpha$ and $\beta_i = \beta$ was proposed by [14]. In [19] the constraint problem with the general penalties case and different due dates related to each job was considered. The author presented a branch-and-bound algorithm that makes use of a lower bound based on Lagrangean relaxation. More recently, algorithms based on dynamic programming and branch-and-bound schemes for the unconstrained problem with uniform penalties $\alpha$ and $\beta$ were presented [20].

Aiming to contribute with the study of this problem with restrictive common due date and general penalties, this paper addresses the development of a specific branch-and-bound algorithm. Lower bounds and pruning rules that exploit the properties of the problem are introduced. Computational tests are presented and the performance of the proposed algorithm is analysed through a comparative study with 280 problems involving different due date scenarios. In addition, the values of optimal solutions for small problems from a known benchmark are provided. Similar approaches were successfully applied in [22] and [23] to the flowshop environment with blocking aiming to minimize the makespan and the tardiness criteria, respectively.

This paper is organized as follows: the next section presents a mathematical model. Section 3 describes the search strategies of the branch-and-bound algorithm and the proposed lower bounds. Section 4 shows the computational results, while the last section summarizes the main results.

**Notation.** For all $v \in \mathbb{R}$ we denote $v^+ = \max\{v, 0\}$.

## 2 Mathematical model

The following mixed integer linear programming (MILP) formulation, slightly modified from the model presented in [3], can be used to obtain optimal solutions to this problem.

**Parameters:**

$d$: common due date;

$\alpha_i$: earliness penalty of job $i$ per time unit;

$\beta_i$: tardiness penalty of job $i$ per time unit;

$p_i$: processing time of job $i$;

$R$: sufficiently large number.

**Variables:**

$x_{ik}$: 1, if job $i$ is sequenced (not necessarily directly) prior to job $k$. 0, otherwise;

$C_i$: completion time of job $i$;

$E_i$: earliness of job $i$;

$T_i$: tardiness of job $i$.

**Model:**

$$\text{Min } z = \sum_{i=1}^{n} \alpha_i \; E_i + \sum_{i=1}^{n} \beta_i \; T_i \tag{1}$$

subject to

$$T_i - E_i = C_i - d, \qquad i = 1, 2, \ldots, n, \tag{2}$$

$$C_i \leq C_k - p_k + R\,(1 - x_{ik}), \qquad i = 1, 2, \ldots, n-1, k = i+1, \ldots, n, \tag{3}$$

$$C_k \leq C_i - p_i + R\,x_{ik}, \qquad i = 1, 2, \ldots, n-1, k = i+1, \ldots, n, \tag{4}$$

$$C_i - p_i \geq 0, \qquad i = 1, 2, \ldots, n, \tag{5}$$

$$T_i \geq 0, \qquad i = 1, 2, \ldots, n, \tag{6}$$

$$E_i \geq 0, \qquad i = 1, 2, \ldots, n, \tag{7}$$

$$x_{ik} \in \{0, 1\}, \qquad i = 1, 2, \ldots, n-1, k = i+1, \ldots, n. \tag{8}$$

Equation (1) represents the objective function to be minimized, i.e., the sum of tardiness and earliness penalties. In [3] the tardiness and earliness are calculated through the following restrictions:

$$T_i \geq d - C_i, i = 1, 2, \ldots, n,$$

$$E_i \geq C_i - d, i = 1, 2, \ldots, n.$$

Alternatively, we calculate tardiness and earliness through constraints of form (2). Note that the presented model has $n$ restrictions less than the original one. Constraints of form (3) and (4) indicate the completion time of each job: if job $i$ is sequenced prior to job $k$, $x_{ik} = 1$ and, consequently, restriction (3) gives $C_i \leq C_k - p_k$ and, due the addition of constant $R$, restriction (4) is not restrictive. On the other hand, if $x_{ik} = 0$, restriction (4) becomes $C_k \leq C_i - p_i$ and restriction (3) is not restrictive. Restriction (5) assures that initial time of each job $i$ is not negative. The set of restrictions (6) and (7) defines the non-negativity of variables $T_i$ and $E_i$, while restriction (8) defines the variable $x_{ik}$ as binary.

## 3 Branch-and-bound algorithm

The proposed branch-and-bound algorithm is composed of two diverse strategies. This separation is based on Property 3 (see Section 1), which states that there is an optimal solution in which either the processing of the first job starts at time zero or one job is completed on the due date.

In the first strategy (Search Strategy 1), schedules that have a job being completed exactly on the due date are explored, while in the second one (Search Strategy 2), schedules that start at time zero are investigated. These strategies will be applied in a sequential form and their combined execution guarantees that the algorithm covers the entire solution space. The initial incumbent solution is provided by a constructive heuristic, HRM, proposed in [11].

### *3.1 Search Strategy 1*

This search strategy looks for the best solution that has a job being completed exactly on the due date; the starting time of this solution may be different from time zero. The sequence of jobs is treated as two subsequences: in one subsequence, the last job finishes on the due date and, in the other, the first job starts on the due date.

The first level of the search tree corresponds to the number of jobs that finish their processing before or on the due date ($n^{\text{fb}}$). For the remaining levels, only two nodes will be generated representing the relative position of a job in relation to the due date. These positions are indicated by the label *Before* (the analyzed job finishes its processing before or on the due date) or *After* (the job starts its processing after or on the due date). It was assumed that level 2 corresponds to job 1; level 3 corresponds to job 2, and so on. When the node *Before* is created, the algorithm checks if there is enough space to allocate the new job. The order of the jobs in each subsequence follows the V-shape (Property 2).

Figure 1 illustrates an example of the proposed search tree with four jobs. Note that, in the first level, solutions with $n^{\text{fb}} = 4$ and $n^{\text{fb}} = 0$ are not part of the tree. If feasible schedules with all non-tardy jobs can be obtained, the considered due date is unrestrictive. On the other hand, a schedule with all late jobs always can be improved with a left shift of the sequence.

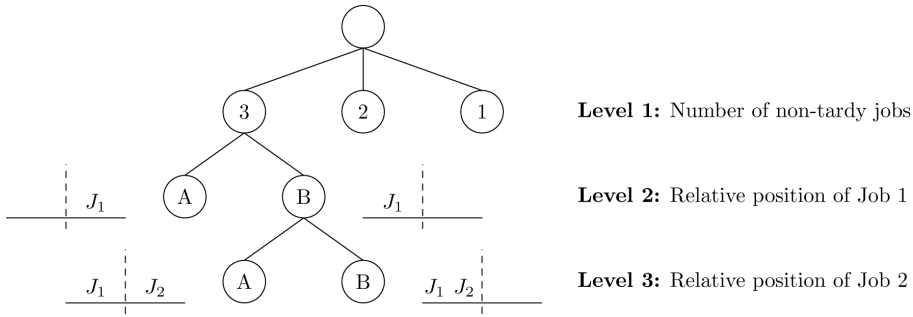


Figure 1 – Example of a partial tree with 4 jobs generated by Search Strategy 1. In the picture, "A" means *After* and "B" means *Before*.

In the first level of the tree, the father node will be the one with the largest number of non-tardy jobs. This criterion aims to foster the elimination of solutions that have the minimum sum of $n^{\text{fb}}$ processing times greater than the period of time available before the due date. The subtree below the chosen node will be investigated before the selection of another node in this level.

For each father node in the first level, a lower bound is calculated. The proposed lower bound, $LB_{1a}$, is presented in the following proposition:

**Proposition 1.** *Consider a set $J$ of $n$ jobs that is composed of two subsequences, one with $n^{\text{fb}}$ unknown jobs starting before the due date, and another with $n - n^{\text{fb}}$ unknown jobs completed after the due date. Assume that there is no idle time between consecutive jobs and the job in position $n^{\text{fb}}$ is completed on the due date. Then, the weighted sum of earliness and tardiness penalties of the jobs is greater than or equal to:*

$$LB_{1a} = \min_{k\in J}\{\alpha_k\} \sum_{i=1}^{n^{\text{fb}}-1} (n^{\text{fb}} - i)\ \ p_i^{\min} + \min_{k\in J}\{\beta_k\} \sum_{i=1}^{n-n^{\text{fb}}} (n - n^{\text{fb}} + 1 - i)\ \ p_i^{\min},$$

*where $p_q^{\min}$ is the $q$-th smallest processing time among jobs in $J$.*

**Proof.** Consider a known sequence of a set $J$ with $n$ jobs where $n^{\text{fb}}$ jobs are completed before or on the due date and $n - n^{\text{fb}}$ jobs are completed after the due date. The weighted sum of earliness and tardiness penalties of this

sequence is given by:

$$\sum_{i\in J}\alpha_i(d-C_i)^+ + \beta_i(C_i-d)^+$$
$$= \sum_{i=1}^{n^{\mathrm{fb}}-1}\left(\alpha_{\pi(i)}\sum_{g=i+1}^{n^{\mathrm{fb}}} p_{\pi(g)}\right) + \sum_{i=n^{\mathrm{fb}}+1}^{n}\left(\beta_{\pi(i)}\sum_{g=n^{\mathrm{fb}}+1}^{i} p_{\pi(g)}\right),$$

where $\pi(y)$ is the job that is allocated in position $y$.

It can be observed that:

$$\begin{aligned}
&\sum_{i=1}^{n^{\mathrm{fb}}-1}\left(\alpha_{\pi(i)}\sum_{g=i+1}^{n^{\mathrm{fb}}} p_{\pi(g)}\right) + \sum_{i=n^{\mathrm{fb}}+1}^{n}\left(\beta_{\pi(i)}\sum_{g=n^{\mathrm{fb}}+1}^{i} p_{\pi(g)}\right)\\
&\quad\geq \min_{k\in J}\{\alpha_k\}\sum_{i=1}^{n^{\mathrm{fb}}-1}\sum_{g=i+1}^{n^{\mathrm{fb}}} p_{\pi(g)} + \min_{k\in J}\{\beta_k\}\sum_{i=n^{\mathrm{fb}}+1}^{n}\sum_{g=n^{\mathrm{fb}}+1}^{i} p_{\pi(g)}\\
&\quad\geq \min_{k\in J}\{\alpha_k\}\sum_{i=1}^{n^{\mathrm{fb}}-1}\sum_{g=1}^{n^{\mathrm{fb}}-i} p_g^{\min} + \min_{k\in J}\{\beta_k\}\sum_{i=1}^{n-n^{\mathrm{fb}}}\sum_{g=1}^{i} p_g^{\min}\\
&\quad= \min_{k\in J}\{\alpha_k\}\sum_{i=1}^{n^{\mathrm{fb}}-1}(n^{\mathrm{fb}}-1)\ p_i^{\min} + \min_{k\in J}\{\beta_k\}\sum_{i=1}^{n-n^{\mathrm{fb}}}(n-n^{\mathrm{fb}}+1-i)\ p_i^{\min}\\
&\quad= LB_{1a}.
\end{aligned}$$

Since $LB_{1a}$ is smaller than or equal to the weighted sum of earliness and tardiness penalties of an arbitrary sequence defined as in the statement of the proposition, we conclude that the thesis holds. □

In the next levels, the node to be branched is the one that contains the subsequence that is closest to be completed. This proximity is measured by the smallest number of jobs to be positioned in each subsequence. It is calculated for each node using the following expression:

$$\text{proximity} = \min\left\{n^{\mathrm{fb}} - x_b,\, n - n^{\mathrm{fb}} - x_a\right\},$$

where $x_b$ and $x_a$ are the number of jobs already positioned before and after the due date, respectively. Note that, if all jobs in any side of the due date have been

fixed, the sequence of the remaining jobs is also determined by Property 2. In case of ties, the algorithm selects the node with the maximum lower bound $LB_{1b}$ which considers the jobs already positioned and it is evaluated according to the following expression:

$$LB_{1b} = \max\left\{LB_{1a}, LB'_{1b}\right\},$$

where

$$LB'_{1b} = \sum_{i=1}^{x_a+x_b} \left[\alpha_{\pi(i)}(d - C_{\pi(i)})^+ + \beta_{\pi(i)}(C_{\pi(i)} - d)^+\right].$$

A node is fathomed if its lower bound is greater than or equal to the current incumbent solution value.

### *3.2 Search Strategy 2*

This strategy addresses solutions whose initial time is zero. The algorithm builds a branch-and-bound (b&b) tree where each node represents a partial sequence. When a node is branched, one or more nodes are generated by adding one more job to the partial sequence associated with the node being branched. This sequence is constructed from end to beginning, i.e., schedule construction starts by positioning the last job to be processed and it continues fixing jobs adjacently until it reaches the first job to be processed. Note that the first fixed job will always finish its processing in time instant equivalent to the sum of all processing times, while the last fixed job will start at time zero. According to [16], this approach delivers better results than the natural one (constructing it from beginning to end). Figure 2 illustrates this construction order.

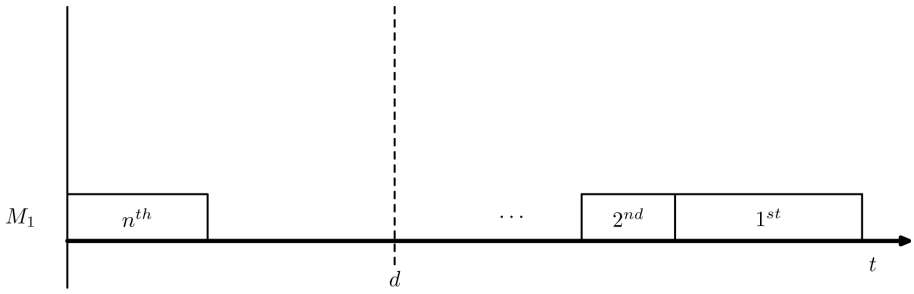


Figure 2 – Sequence construction order in Search Strategy 2.

To reduce the search space, only schedules whose jobs are positioned in V-shape (Property 2) and without idle time between consecutive jobs (Property 1) are assembled.

The node that will be branched (father node) is the one with the largest sum of processing times of the already fixed jobs. This criterion aims to favor partial sequences whose starting time is closer to the due date. In case of ties, the algorithm chooses the node with the largest lower bound. When the starting time of a partial sequence is equal to or smaller than the due date, the best position of the remaining jobs is determined by Property 2 (V-shape) and the node is fathomed.

For each generated node, a lower bound on the weighted sum of earliness and tardiness penalties is computed. A node is fathomed if this estimate is greater than or equal to the current incumbent solution.

The proposed lower bound ($LB_2$) considers the penalties caused by the known jobs in the partial sequence. It also considers the estimates of the penalty generated by the job that will finish its processing at the starting time of the partial sequence and by the subsequence of unknown jobs that start its processing at time zero. The following lemma is needed to establish the lower bound.

**Lemma 1.** *Let $A = a_1, a_2, \ldots, a_t$ and $B = b_1, b_2, \ldots, b_t$ sequences of positive integer numbers and define $Z(A, B) = \sum_{\ell=1}^{t} a_\ell b_\ell$. If $a_1 \geq a_2 \geq \cdots \geq a_t$ and $b_1 \leq b_2 \leq \cdots \leq b_t$ then $Z(A, B)$ is minimized.*

**Proof.** Assume that $Z$ is minimized by sequences $A$ and $B$ that do not satisfy the hypothesis. We will show that there is another pair of sequences $A'$ and $B'$ satisfying the hypothesis that also minimizes $Z$. As $A$ and $B$ do not satisfy the hypothesis then there exists at least one index $v$ such that $a_v < a_{v+1}$ or $b_v > b_{v+1}$. We are left to three different cases:

i) $a_v < a_{v+1}$ and $b_v \leq b_{v+1}$,

ii) $a_v < a_{v+1}$ and $b_v > b_{v+1}$,

iii) $a_v \geq a_{v+1}$ and $b_v > b_{v+1}$.

Interchanging $a_v$ and $a_{v+1}$ in case i), interchanging $a_v$ and $a_{v+1}$ and $b_v$ and $b_{v+1}$ in case ii), or interchanging $b_v$ and $b_{v+1}$ in case iii), it is easy to see that

the associated value of $Z$ evaluated at the modified sequences does not increase (in fact, it remains unchanged in case ii)). By continuing this pairwise interchange we can obtain ordered sequences $A'$ and $B'$ satisfying the hypothesis that minimize $Z$, as we wanted to prove. □

**Proposition 2.** *Consider n jobs and a known subsequence R of these jobs that starts its processing at* $S = \sum_{i=1}^{n} p_i - \sum_{i\in R} p_i > d$ *and finishes at* $\sum_{i=1}^{n} p_i$. *Let D be the set of the remaining jobs. The weighted sum of earliness and tardiness penalties of the complete schedule is greater than or equal to:*

$$LB_2 = \sum_{a=1}^{|D|} \alpha_a^{\min} \left( d - \sum_{b=1}^{a} p_b^{\max} \right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d),$$

*where* $|D|$ *is the cardinality of D and, among jobs in D,* $\alpha_q^{\min}$ *is the q-th smallest earliness penalty and* $p_b^{\max}$ *is the b-th largest processing time.*

**Proof.** Consider a known sequence $W$ with $n$ jobs that starts at time zero. The weighted sum of earliness and tardiness penalties of this sequence is given by:

$$f(W) = \sum_{i\in W} \left[\alpha_i (d - C_i)^+ + \beta_i (C_i - d)^+\right].$$

Let $R$ and $D$ be defined as in the statement of this proposition and let $\pi(y)$ be the job that is allocated in position $y$ and $u$ the job that finishes its processing at instant $S$. Then $f(W)$ can be expressed as:

$$f(W) = \sum_{i\notin R} \alpha_i (d - C_i)^+ + \sum_{\substack{i\notin R \\ i\neq u}} \beta_i (C_i - d)^+ + \beta_u (S-d) + \sum_{i\in R} \beta_i (C_i - d).$$

As all penalties are non-negative and $\beta_u \geq \min_{k\in D}\{\beta_k\}$, the following relations can be established:

$$\begin{aligned} f(W) &\geq \sum_{i\notin R} \alpha_i (d - C_i)^+ + \beta_u (S-d) + \sum_{i\in R} \beta_i (C_i - d) \\ &\geq \sum_{i\notin R} \alpha_i (d - C_i)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d) \\ &= \sum_{a=1}^{|D|} \alpha_{\pi(a)} \left( d - \sum_{b=1}^{a} p_{\pi(b)} \right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d). \end{aligned}$$

Since $\sum_{b=1}^{a} p_b^{\max} \geq \sum_{b=1}^{a} p_{\pi(b)}$ implies that

$$\left(d - \sum_{b=1}^{a} p_{\pi(b)}\right)^+ \geq \left(d - \sum_{b=1}^{a} p_b^{\max}\right)^+,$$

it follows that:

$$\sum_{a=1}^{|D|} \alpha_{\pi(a)} \left(d - \sum_{b=1}^{a} p_{\pi(b)}\right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d)$$

$$\geq \sum_{a=1}^{|D|} \alpha_{\pi(a)} \left(d - \sum_{b=1}^{a} p_b^{\max}\right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d).$$

We know that:

$$\left(d - \sum_{b=1}^{1} p_b^{\max}\right)^+ \geq \left(d - \sum_{b=1}^{2} p_b^{\max}\right)^+ \geq \cdots \geq \left(d - \sum_{b=1}^{|D|} p_b^{\max}\right)^+.$$

Using Lemma 1 (with $a_\ell = \left(d - \sum_{b=1}^{\ell} p_b^{\max}\right)^+$ and $b_\ell = \alpha_\ell^{\min}$, for $\ell = 1, \ldots, |D|$) we have that:

$$\sum_{a=1}^{|D|} \alpha_{\pi(a)} \left(d - \sum_{b=1}^{a} p_b^{\max}\right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d)$$

$$\geq \sum_{a=1}^{|D|} \alpha_a^{\min} \left(d - \sum_{b=1}^{a} p_b^{\max}\right)^+ + \min_{k\in D}\{\beta_k\}(S-d) + \sum_{i\in R} \beta_i (C_i - d) = LB_2.$$

Since $LB_2$ is smaller than or equal to the weighted sum of earliness and tardiness penalties of an arbitrary sequence defined as in the statement of the proposition we conclude that the thesis holds. □

## 4 Computational experiments

The proposed b&b algorithm was applied to two different sets of problems. In the first experiment, the instances were generated according to [3], so that they do not depend on the computer used. The authors of [3] were the pioneers in

considering solutions starting at time instants different from zero. Seven different numbers of jobs $n \in \{5, 10, 15, 20, 25, 30, 35\}$ and four restrictive factors $h \in \{0.2, 0.4, 0.6, 0.8\}$ were considered. The factor $h$ indicates how jammed the production line is at the beginning of the schedule and it is used in the definition of the common due date, according to the expression: $d = \lfloor h \sum_{i=1}^{n} p_i \rfloor$.

The processing times are integers uniformly distributed in [1, 20], the earliness penalties in [1, 10] and the tardiness penalties in [1, 15]. There are 10 instances to be tested for each problem size and each restrictive factor, totaling $7 \times 4 \times 10 = 280$ problems. The computer code was written using C language and the experiments were run on an Intel Core 2 Duo with a 2.40 GHz processor and 2.0 Gb of RAM memory. To prevent excessive computation time, the algorithm was stopped after 1 hour of CPU time for each problem. All instances are available at [25].

Table 1 presents the results when the b&b algorithm was applied to this first set of problems. The columns *Min*, *Max* and *Average* represent, respectively, the minimum, maximum and average CPU time obtained in each combination of number of jobs and restrictive factor. The column *Number of solved problems* indicates the quantity of instances solved within the time limit. As it can be seen, all the problems with $n \leq 25$ jobs were solved in an average time of less than 23 seconds.

In order to evaluate the proposed b&b we solve the same problem set using the MILP model described in Section 2 with the solver CPLEX 11.0 with its default parameter values. The computation time for each test instance was also limited to 1 hour. Analyzing Table 1 it can be noted that the CPLEX solver was not able to prove optimality within the allowed execution time in all instances with more than 10 jobs while the b&b algorithm found the optimal solution for all instances with up to 30 jobs. Moreover, it can be observed that in all instances solved by both methods, the CPU time of the b&b was smaller than the one presented by the CPLEX solver. These results indicate the efficiency of the proposed algorithm in reducing the search space. This good performance was expected since the lower bounds and pruning rules proposed in this work are specific for the single machine earliness and tardiness scheduling problem. These essential components of the presented b&b are able to exploit properties

| $n$ | $h$ | b&b | | | | CPLEX | |
|---|---|---|---|---|---|---|---|
| | | CPU Time (s) | | | Number of solved problems | Average CPU Time (s) | Number of solved problems |
| | | Min | Max | Average | | | |
| 5 | 0.2 | 0.0 | 0.0 | 0.0 | 10 | 0.5 | 10 |
| | 0.4 | 0.0 | 0.0 | 0.0 | 10 | 0.5 | 10 |
| | 0.6 | 0.0 | 0.0 | 0.0 | 10 | 0.5 | 10 |
| | 0.8 | 0.0 | 0.0 | 0.0 | 10 | 0.5 | 10 |
| 10 | 0.2 | 0.0 | 0.0 | 0.0 | 10 | 5.5 | 10 |
| | 0.4 | 0.0 | 0.0 | 0.0 | 10 | 4.6 | 10 |
| | 0.6 | 0.0 | 0.0 | 0.0 | 10 | 2.6 | 10 |
| | 0.8 | 0.0 | 0.0 | 0.0 | 10 | 2.4 | 10 |
| 15 | 0.2 | 0.0 | 0.0 | 0.0 | 10 | 3600[1] | 0 |
| | 0.4 | 0.0 | 0.0 | 0.0 | 10 | 3600[1] | 0 |
| | 0.6 | 0.0 | 0.0 | 0.0 | 10 | 3600[1] | 0 |
| | 0.8 | 0.0 | 0.0 | 0.0 | 10 | 3600[1] | 0 |
| 20 | 0.2 | 0.0 | 0.4 | 0.1 | 10 | 3600[1] | 0 |
| | 0.4 | 0.1 | 0.4 | 0.2 | 10 | 3600[1] | 0 |
| | 0.6 | 0.1 | 1.0 | 0.4 | 10 | 3600[1] | 0 |
| | 0.8 | 0.2 | 1.1 | 0.5 | 10 | 3600[1] | 0 |
| 25 | 0.2 | 0.6 | 6.2 | 3.0 | 10 | 3600[1] | 0 |
| | 0.4 | 4.1 | 14.4 | 9.0 | 10 | 3600[1] | 0 |
| | 0.6 | 7.5 | 35.5 | 20.0 | 10 | 3600[1] | 0 |
| | 0.8 | 8.7 | 45.2 | 22.8 | 10 | 3600[1] | 0 |
| 30 | 0.2 | 12.7 | 332.0 | 75.2 | 10 | 3600[1] | 0 |
| | 0.4 | 84.5 | 616.5 | 232.9 | 10 | 3600[1] | 0 |
| | 0.6 | 152.2 | 1197.1 | 646.3 | 10 | 3600[1] | 0 |
| | 0.8 | 256.5 | 1311.4 | 636.1 | 10 | 3600[1] | 0 |
| 35 | 0.2 | 223.9 | 2100.2 | 865.4 | 10 | 3600[1] | 0 |
| | 0.4 | 1169.9 | 3600.0 | 2707.3[1] | 5 | 3600[1] | 0 |
| | 0.6 | 996.3 | 3600.0 | 3339.6[1] | 1 | 3600[1] | 0 |
| | 0.8 | 968.9 | 3600.0 | 3327.8[1] | 2 | 3600[1] | 0 |

Table 1 – Performance of the b&b algorithm.

[1] Lower bounds on the mean values as there are unsolved problems in these sets.

of the problem, as, for example, the fact that if a sequence of jobs before or after the common due date is determined the complete schedule is already defined.

Aiming to contribute to the study of this problem, in a second experiment, the b&b algorithm was applied to instances with 10 and 20 jobs from the benchmark problems presented in [3]. These benchmark values are frequently used in literature (see, for example [11] and [18]). For the 10-jobs instances, the benchmark provides optimal solutions as well as the b&b algorithm. Considering problems with 20 jobs, the proposed algorithm obtained optimal solutions within the time limit for all problems. Table 2 shows the solution values for the 40 instances generated by the b&b algorithm.

We also computed the percentage difference of the optimal value ($F_o$) in relation to the benchmark value of Biskup and Feldman ($F_{BF}$), as follows:

$$\%\text{Diff} = 100\frac{F_{BF} - F_o}{F_o}.$$

Table 2 shows the %Diff for each considered problem. It can be observed that, in the worst case scenario, the benchmark value is 11.37% greater than the optimal value. These optimal solutions are interesting for future comparison purposes.

| $h = 0.2$ | | $h = 0.4$ | | $h = 0.6$ | | $h = 0.8$ | |
|---|---|---|---|---|---|---|---|
| Optimal value | %Diff | Optimal value | %Diff | Optimal value | %Diff | Optimal value | %Diff |
| 4394 | 0.84 | 3066 | 0.00 | 2986 | 0.00 | 2986 | 0.00 |
| 8430 | 1.63 | 4847 | 1.03 | 3206 | 1.68 | 2980 | 0.00 |
| 6210 | 1.95 | 3838 | 1.17 | 3583 | 0.47 | 3583 | 0.47 |
| 9188 | 3.16 | 5118 | 0.08 | 3317 | 0.57 | 3040 | 0.00 |
| 4215 | 2.97 | 2495 | 3.05 | 2173 | 1.52 | 2173 | 1.52 |
| 6527 | 3.66 | 3582 | 0.53 | 3010 | 0.20 | 3010 | 0.20 |
| 10455 | 6.18 | 6238 | 1.91 | 4126 | 1.19 | 3878 | 0.57 |
| 3920 | 7.22 | 2145 | 0.28 | 1638 | 0.00 | 1638 | 0.00 |
| 3465 | 1.88 | 2096 | 0.05 | 1965 | 1.37 | 1965 | 1.37 |
| 4979 | 11.37 | 2925 | 9.13 | 2110 | 0.28 | 1995 | 0.00 |

Table 2 – Solution values of the b&b for the Biskup and Feldman's benchmark.

## 5 Final remarks

This paper considered the single machine scheduling problem with restrictive common due date involving tardiness and earliness penalties. This type of problem became more important with the advent of the lean production principles, including the just-in-time (JIT) concept. Due to its complexity, most of the authors addressed this problem using heuristic and metaheuristic approaches.

In this study, a branch-and-bound algorithm was proposed to find optimal solutions to this problem. In the development of the algorithm, the use of problem properties was important for the development of new lower bounds and pruning rules that have enhanced the efficiency of the proposed method.

An implementation of the method was tested in 280 problems generated as presented in [3]. The proposed b&b outperformed the CPLEX optimization software. This software was unable to prove optimality in all instances with $n \geq 15$ (200 instances) within the time limit, while the b&b algorithm found the optimal solution in 92% of the considered instances. These results indicate the efficiency of the algorithm, mainly due to the elimination of inferior quality solutions through the use of the proposed lower bounds that exploit characteristics of the considered problem.

In addition, the optimal solution values obtained for the benchmark problems suggested in [3] revealed that these reference values can be improved by up to 11.37%. These results can be used to evaluate the performance of heuristics and meta-heuristics developed for this problem.

As an extension of this study, we suggest the use of properties (see [24]) in the development of a lower bound for more general cases, such as in the flowshop problem with multiple machines.

**Acknowledgements.** The authors would like to thank the anonymous referees whose comments helped a lot to improve this paper.

### REFERENCES

[1] M. van den Akker, H. Hoogeveen and S. van de Velde, *Combining column generation and Lagrangean relaxation to solve a single-machine common due date.* INFORMS Journal on Computing, **14** (2002), 37–51.

[2] K.R. Baker and G.D. Scudder, *Sequencing with earliness and tardiness penalties: A review.* Operations Research, **38** (1990), 22–36.

[3] D. Biskup and M. Feldmann, *Benchmarks for scheduling on a single-machine against restrictive and unrestrictive common due dates.* Computers and Operations Research, **28** (2001), 787–801.

[4] T.C.E. Cheng and H.G. Kahlbacher, *A proof for the longest-job-first policy in one-machine scheduling.* Naval Research Logistics, **38** (1991), 715–720.

[5] P. De, J.B. Ghosh and C.E. Wells, *Solving a generalized model for CON due date assignment and sequencing.* International Journal of Production Economics, **34** (1994), 179–185.

[6] P. Dileepan, *Common due date scheduling problem with separate earliness and tardiness penalties.* Computers and Operations Research, **20** (1993), 179–184.

[7] M. Feldmann and D. Biskup, *Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches.* Computers and Industrial Engineering, **44** (2003), 307–323.

[8] V. Gordon, J.M. Proth and C. Chu, *A survey of the state-of-art of common due date assignment and scheduling research.* European Journal of Operational Research, **139** (2002), 1–25.

[9] N.G. Hall and M.E. Posner, *Earliness-tardiness scheduling problems, I: weighted deviation of completion times about a common due date.* Operations Research, **39** (1991), 836–846.

[10] N. Hall, G.W. Kubiak and S.P. Sethi, *Earliness-tardiness scheduling problems, II: deviations of completion times about a restrictive common due date.* Operations Research, **39** (1991), 847–856.

[11] C.M. Hino, D.P. Ronconi and A.B. Mendes, *Minimizing earliness and tardiness penalties in a single-machine problem with a common due date.* European Journal of Operational Research, **160** (2005), 190–201.

[12] J.A. Hoogeven and S.L. van de Velde, *Scheduling around a small common due date.* European Journal of Operational Research, **55** (1991), 237–242.

[13] R.J.W. James, *Using tabu search to solve the common due date early/tardy machine scheduling problem.* Computers and Operations Research, **24** (1997), 199–208.

[14] H.G. Kahlbacher, *Scheduling with monotonous earliness and tardiness penalties.* European Journal of Operational Research, **64** (1993), 258–277.

[15] J.J. Kanet, *Minimizing the average deviation of job completion times about a common due date.* Naval Research Logistics Quartely, **28** (1981), 643–651.

[16] M.S. Kawamura and D.P. Ronconi, *Aplicação do método branch-and-bound na programação de tarefas em uma única máquina com data de entrega comum sob penalidades de adiantamento e atraso.* Technical Report 0606, Escola Politécnica, Universidade de São Paulo (2006).

[17] C.Y. Lee and S.J. Kim, *Parallel genetic algorithm for the earliness-tardiness job scheduling problem with general penalty weights.* Computers and Industrial Engineering, **28** (1995), 231–243.

[18] C-J. Liao and C-C. Cheng, *A variable neighborhood search for minimizing single machine weighted earliness and tardiness with common due date.* Computers and Industrial Engineering, **52** (2007), 404–413.

[19] C.-F. Liaw, *A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem.* Computers and Operations Research, **26** (1999), 679–693.

[20] S.A. Mondal and A.K. Sen, *Single machine weighted earliness-tardiness penalty problem with a common due date.* Computers and Operations Research, **28** (2001), 649–669.

[21] S.S. Panwalkar, M.L. Smith and A. Seidmann, *Common due date assignment to minimize total penalty for one machine problem.* Operations Research, **30** (1982), 391–399.

[22] D.P. Ronconi, *A Branch-and-Bound Algorithm to Minimize the Makespan in a Flowshop with Blocking.* Annals of Operations Research, **138** (2005), 53–65.

[23] D.P. Ronconi and V.A. Armentano, *Lower Bounding Schemes for Flowshops with Blocking In-Process.* Journal of the Operational Research Society, **52** (2001), 1289–1297.

[24] C.S. Sakuraba, D.P. Ronconi and F. Sourd, *Scheduling in a two-machine flowshop for the minimization of the mean absolute deviation from a common due date.* Computers and Operations Research, **36** (2009), 60–72.

[25] *http://www.poli.usp.br/pro/docentes/dronconi/*.