Asia-Pacific Journal of Operational Research
Vol. 25, No. 2 (2008) 169–186

AN EXACT APPROACH FOR THE SINGLE MACHINE SCHEDULING PROBLEM WITH LINEAR EARLY AND QUADRATIC TARDY PENALTIES

JORGE M. S. VALENTE
LIAAD, Faculdade de Economia
Universidade do Porto, Portugal
jvalente@fep.up.pt

Received 29 December 2006
Accepted 11 March 2007

In this paper, we consider the single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. We propose a lower bounding procedure based on the relaxation of the jobs' completion times. Optimal branch-and-bound algorithms are then presented. These algorithms incorporate the proposed lower bound, as well as an insertion-based dominance test.

The branch-and-bound procedures are tested on a wide set of randomly generated problems. The computational results show that the branch-and-bound algorithms are capable of optimally solving, within reasonable computation times, instances with up to 20 jobs.

Keywords: Scheduling; single machine; linear earliness; quadratic tardiness; lower bound; branch-and-bound.

1. Introduction

In this paper, we consider a single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. Single machine scheduling environments actually occur in many practical operations (for a specific example in the chemical industry, see Wagner *et al.* (2002)). Also, the performance of production environments is frequently determined by a single bottleneck machine. Single processor models are then quite useful for scheduling such a machine. Moreover, results and insights obtained for single machine problems can often be applied to more complex scheduling environments, such as flow shops or job shops.

Scheduling models with earliness and tardiness penalties are compatible with a recent trend in industry, namely the adoption of supply chain management by many organizations. In this approach, customers and suppliers try to integrate the flow of materials, in order to improve the efficiency of the supply chain and provide a better service to the end user. The adoption of supply chain management has caused

organizations to view both early and tardy deliveries as undesirable. Early/tardy scheduling models are also suited to the philosophy of just-in-time (JIT) production. In fact, the JIT production philosophy emphasizes producing goods only when they are needed. Therefore, in a JIT setting an ideal schedule is one in which all jobs are completed exactly on their due dates.

In this paper, we consider linear earliness and quadratic tardiness costs. On the one hand, early deliveries or early completions of jobs result in unnecessary inventory that ties up cash, as well as space and resources required to maintain and manage the inventory. These costs tend to be proportional to the quantity of inventory held, and therefore a linear penalty is used for early jobs.

Late deliveries, on the other hand, can result in lost sales and loss of customer goodwill, and may also cause disruptions in stages further down the production line or supply chain. We consider a quadratic tardiness penalty, instead of the more usual linear tardiness or maximum tardiness functions. As described in Sun *et al.* (1999), the quadratic penalty may be selected over these two other tardiness measures for the following reasons.

First, the maximum tardiness performance measure does not distinguish between schedules where tardiness occurs for all jobs, or only one, as long as the maximum tardiness is the same. Second, when the linear tardiness function is used, it is possible that a single or only a few jobs contribute the majority of the cost, without regard to how the overall tardiness is distributed. In fact, the linear tardiness criterion does not differentiate between sequences where all jobs are only a little tardy, or a single job is extremely late, providing the total cost is equal. The quadratic penalty overcomes these problems, and provides a more robust performance measure. Moreover, a quadratic tardiness penalty is also appropriate in practice. In fact, tardiness represents an important attribute of service quality. Also, as proposed in the loss function of Taguchi (1986), a customer's dissatisfaction tends to increase quadratically with the tardiness.

We assume that no machine idle time is allowed. This assumption is actually appropriate for many production settings. In fact, when the capacity of the machine is limited when compared with the demand, the machine must indeed be kept running in order to satisfy the customers' orders. Also, the assumption of no idle time is justified when the machines have high operating costs. Furthermore, idle time must also be avoided when starting a new production run involves high setup costs or times. Some specific examples of production settings where the no idle time assumption is appropriate, have been given by Korman (1994) and Landis (1993).

Formally, the problem we consider can be stated as follows. A set of n independent jobs $\{J_1, J_2, \ldots, J_n\}$ has to be scheduled on a single machine that can handle at most one job at a time. The machine is assumed to be continuously available from time zero onwards, and preemptions are not allowed. Job $J_j, j = 1, 2, \ldots, n$, requires a processing time p_j and should ideally be completed on its due date d_j. For a given schedule, the earliness and tardiness of J_j are respectively defined as $E_j = \max\{0, d_j - C_j\}$ and $T_j = \max\{0, C_j - d_j\}$, where C_j is the completion time of J_j. The objective is then to find a schedule that minimizes the sum of linear

earliness and quadratic tardiness costs $\sum_{j=1}^{n}(E_j + T_j^2)$, subject to the constraint that no machine idle time is allowed.

This problem has been previously considered by Valente (2006). He presented several dispatching heuristics, and analyzed their performance on a wide range of instances. Schaller (2004) considered the corresponding problem with inserted idle time. He presented a timetabling procedure to optimally insert idle time in a given sequence, as well as a branch-and-bound procedure and simple and efficient heuristics.

The single machine early/tardy problem with linear earliness and tardiness costs $\sum_{j=1}^{n}(E_j + T_j)$ has also been previously considered by Garey *et al.* (1988) and Kim and Yano (1994). Garey *et al.* (1988) showed that the problem is NP-hard, and proposed a timetabling procedure. Kim and Yano (1994) presented some properties of optimal solutions, and developed both optimal and heuristic algorithms.

The minimization of the quadratic lateness $\sum_{j=1}^{n} L_j^2$, where the lateness of J_j is defined as $L_j = C_j - d_j$, has also been previously considered. Gupta and Sen (1983) presented a branch-and-bound algorithm and a heuristic rule for the problem with no idle time. Su and Chang (1998) and Schaller (2002) considered the insertion of idle time, and proposed timetabling procedures and heuristic algorithms. Sen *et al.* (1995) presented a branch-and-bound algorithm for the weighted problem $\sum_{j=1}^{n} w_j L_j^2$ where idle time is allowed only prior to the start of the first job. Baker and Scudder (1990) provide an excellent survey of scheduling problems with earliness and tardiness penalties, while Kanet and Sridharan (2000) give a review of scheduling models with inserted idle time that complements our focus on a problem with no machine idle time.

In this paper, we first develop a lower bounding procedure. An optimal branch-and-bound algorithm that incorporates this lower bound, as well as an insertion-based fathoming test, is then proposed. The branch-and-bound procedure is then tested on a wide set of randomly generated problems.

The remainder of this paper is organized as follows. The lower bounding procedure is described in Sec. 2. In Sec. 3, we discuss the implementation details of the branch-and-bound algorithm. The computational results are presented in Sec. 4. Finally, some concluding remarks are given in Sec. 5.

2. Lower Bounding

In this section, we first propose a lower bounding procedure based on a relaxation of the completion times. Then, we present special cases in which the lower bounding procedure provides the optimum objective function value. Finally, we discuss the generalization and applicability of the lower bounding technique to other objective functions.

2.1. *The lower bounding procedure*

In this section, we propose a lower bound for the linear earliness/quadratic tardiness problem. For convenience, and without loss of generality, it will be assumed

throughout this section that we wish to calculate a lower bound for a sequence that starts at time $t = 0$. This makes the presentation of the lower bound easier and clearer, and the extension of the lower bound to partial sequences that start at any time $t > 0$ is straightforward.

Let Z denote the objective function of the considered problem, i.e. $Z = \sum_{j=1}^{n}(E_j + T_j^2) = \sum_{j=1}^{n}\max(d_j - C_j, 0) + \sum_{j=1}^{n}[\max(C_j - d_j, 0)]^2$. Also, let $[j]$ denote the job in the jth position in a sequence. In order to derive the lower bound, we consider a relaxation of the completion times C_j. Let C_l^{LPT} be the sum of the processing times of the first l jobs, when the jobs are ordered in longest processing time (LPT) order (i.e. $p_{[j]} \geq p_{[k]}$ for $j < k$). Similarly, let C_l^{SPT} be the sum of the processing times of the first l jobs, when the jobs are ordered in shortest processing time (SPT) order (i.e. $p_{[j]} \leq p_{[k]}$ for $j < k$).

We can now define a modified objective function Z' by replacing the completion times C_j in Z with C_l^{LPT} and C_l^{SPT}. More precisely, the modified objective function Z' is defined as $Z' = \sum_{j=1}^{n}\max(d_{[j]} - C_j^{\mathrm{LPT}}, 0) + \sum_{j=1}^{n}[\max(C_j^{\mathrm{SPT}} - d_{[j]}, 0)]^2$. For any given sequence, we have $Z' \leq Z$, since $C_j^{\mathrm{LPT}} \geq C_{[j]}$ and $C_j^{\mathrm{SPT}} \leq C_{[j]}$. Let d_j^{EDD} denote the due date of the jth job, when the jobs are ordered in earliest due date (EDD) order (i.e. $d_{[j]} \leq d_{[k]}$ for $j < k$). Finally, let $Z'_{\mathrm{EDD}} = \sum_{j=1}^{n}\max(d_j^{\mathrm{EDD}} - C_j^{\mathrm{LPT}}, 0) + \sum_{j=1}^{n}[\max(C_j^{\mathrm{SPT}} - d_j^{\mathrm{EDD}}, 0)]^2$.

Theorem 2.1. *$Z'_{\mathrm{EDD}} \leq \min Z$, i.e. Z'_{EDD} is a lower bound for the optimal objective function value of the linear earliness/quadratic tardiness problem.*

Proof. As previously mentioned, we have $Z' \leq Z$ for any specific sequence. Therefore, it is sufficient to prove that $Z'_{\mathrm{EDD}} = \min Z'$, since we then have $Z'_{\mathrm{EDD}} \leq Z$ for all possible sequences. This will be done by contradiction. Consider a schedule S in which jobs j and i are scheduled in positions l_1 and l_2, respectively, with $l_1 < l_2$ and $d_j > d_i$. Therefore, jobs i and j are not scheduled in EDD order. Also consider a schedule S' that is identical to S, except for the fact that the positions of jobs i and j have been interchanged. We must show that $Z'(S') \leq Z'(S)$.

Since only the positions of jobs i and j are different in schedules S and S', all other jobs k, with $k \neq i, j$, occupy the same positions in both schedules. The contribution of those jobs k to Z' is therefore identical in schedules S and S'. Hence, it suffices to compare the contributions of jobs i and j to Z'.

We first consider the change that may occur in the first part of Z', i.e. the change in the term $\sum_{j=1}^{n}\max(d_{[j]} - C_j^{\mathrm{LPT}}, 0)$. Let $E(S)$ denote the sum of the contributions of jobs i and j to the term $\sum_{j=1}^{n}\max(d_{[j]} - C_j^{\mathrm{LPT}}, 0)$ in schedule S. Therefore, we have $E(S) = \max(d_j - C_{l_1}^{\mathrm{LPT}}, 0) + \max(d_i - C_{l_2}^{\mathrm{LPT}}, 0)$ and $E(S') = \max(d_i - C_{l_1}^{\mathrm{LPT}}, 0) + \max(d_j - C_{l_2}^{\mathrm{LPT}}, 0)$. Also let $\Delta E = E(S') - E(S)$. The following three cases must then be considered.

Case 1. $C_{l_2}^{\mathrm{LPT}} < d_i < d_j$. We have $E(S) = d_j - C_{l_1}^{\mathrm{LPT}} + d_i - C_{l_2}^{\mathrm{LPT}}$ and $E(S') = d_i - C_{l_1}^{\mathrm{LPT}} + d_j - C_{l_2}^{\mathrm{LPT}}$. Therefore, $\Delta E = 0$.

Case 2. $d_i \leq C^{\mathrm{LPT}}_{l_2} \leq d_j$. In this case, we have $E(S) = d_j - C^{\mathrm{LPT}}_{l_1}$ and $E(S') = \max(d_i - C^{\mathrm{LPT}}_{l_1}, 0) + d_j - C^{\mathrm{LPT}}_{l_2}$. We then have $\Delta E = \max(d_i - C^{\mathrm{LPT}}_{l_1}, 0) + C^{\mathrm{LPT}}_{l_1} - C^{\mathrm{LPT}}_{l_2} = \max(d_i, C^{\mathrm{LPT}}_{l_1}) - C^{\mathrm{LPT}}_{l_2}$. Therefore, $\Delta E \leq 0$, since $d_i \leq C^{\mathrm{LPT}}_{l_2}$ and $C^{\mathrm{LPT}}_{l_1} < C^{\mathrm{LPT}}_{l_2}$.

Case 3. $d_i < d_j < C^{\mathrm{LPT}}_{l_2}$. We have $E(S) = \max(d_j - C^{\mathrm{LPT}}_{l_1}, 0)$ and $E(S') = \max(d_i - C^{\mathrm{LPT}}_{l_1}, 0)$. Consequently, $\Delta E = \max(d_i - C^{\mathrm{LPT}}_{l_1}, 0) - \max(d_j - C^{\mathrm{LPT}}_{l_1}, 0)$. Therefore, we have $\Delta E \leq 0$, since $d_i < d_j$.

We now consider the change in the second part of Z', i.e. the change in the term $\sum_{j=1}^{n}[\max(C^{\mathrm{SPT}}_j - d_{[j]}, 0)]^2$. Let $T(S)$ denote the sum of the contributions of jobs i and j to the term $\sum_{j=1}^{n}[\max(C^{\mathrm{SPT}}_j - d_{[j]}, 0)]^2$ in schedule S. We then have $T(S) = [\max(C^{\mathrm{SPT}}_{l_1} - d_j, 0)]^2 + [\max(C^{\mathrm{SPT}}_{l_2} - d_i, 0)]^2$ and $T(S') = [\max(C^{\mathrm{SPT}}_{l_1} - d_i, 0)]^2 + [\max(C^{\mathrm{SPT}}_{l_2} - d_j, 0)]^2$. Also let $\Delta T = T(S') - T(S)$. The following three cases must be considered.

Case 1. $C^{\mathrm{SPT}}_{l_2} < d_i < d_j$. In this case, we have $T(S') = T(S) = 0$, so $\Delta T = 0$.

Case 2. $d_i \leq C^{\mathrm{SPT}}_{l_2} \leq d_j$. We have $T(S) = (C^{\mathrm{SPT}}_{l_2} - d_i)^2$ and $T(S') = [\max(C^{\mathrm{SPT}}_{l_1} - d_i, 0)]^2$. Consequently, $\Delta T = [\max(C^{\mathrm{SPT}}_{l_1} - d_i, 0)]^2 - (C^{\mathrm{SPT}}_{l_2} - d_i)^2$. Therefore, we have $\Delta T \leq 0$, since $C^{\mathrm{SPT}}_{l_1} < C^{\mathrm{SPT}}_{l_2}$ and $d_i \leq C^{\mathrm{SPT}}_{l_2}$.

Case 3. $d_i < d_j < C^{\mathrm{SPT}}_{l_2}$. In this case, we have $T(S) = [\max(C^{\mathrm{SPT}}_{l_1} - d_j, 0)]^2 + (C^{\mathrm{SPT}}_{l_2} - d_i)^2$ and $T(S') = [\max(C^{\mathrm{SPT}}_{l_1} - d_i, 0)]^2 + (C^{\mathrm{SPT}}_{l_2} - d_j)^2$. We then have $\Delta T = [\max(C^{\mathrm{SPT}}_{l_1} - d_i, 0)]^2 + (C^{\mathrm{SPT}}_{l_2} - d_j)^2 - [\max(C^{\mathrm{SPT}}_{l_1} - d_j, 0)]^2 - (C^{\mathrm{SPT}}_{l_2} - d_i)^2$. In order to simplify the analysis, this case can now be divided in three further subcases.

Case 3a. $C^{\mathrm{SPT}}_{l_1} < d_i < d_j < C^{\mathrm{SPT}}_{l_2}$. In this situation, we have $\Delta T = 0 + (C^{\mathrm{SPT}}_{l_2} - d_j)^2 - 0 - (C^{\mathrm{SPT}}_{l_2} - d_i)^2$. Therefore, we have $\Delta T < 0$, since $d_i < d_j < C^{\mathrm{SPT}}_{l_2}$.

Case 3b. $d_i \leq C^{\mathrm{SPT}}_{l_1} \leq d_j < C^{\mathrm{SPT}}_{l_2}$. We now have $\Delta T = (C^{\mathrm{SPT}}_{l_1} - d_i)^2 + (C^{\mathrm{SPT}}_{l_2} - d_j)^2 - 0 - (C^{\mathrm{SPT}}_{l_2} - d_i)^2$. Let $A = C^{\mathrm{SPT}}_{l_2} - d_j$, $B = d_j - C^{\mathrm{SPT}}_{l_1}$ and $C = C^{\mathrm{SPT}}_{l_1} - d_i$. We then have $\Delta T = C^2 + A^2 - (A + B + C)^2$. Therefore, $\Delta T \leq 0$, since with $A, B, C \geq 0$ we have $(A + B + C)^2 \geq C^2 + A^2$.

Case 3c. $d_i < d_j < C^{\mathrm{SPT}}_{l_1} < C^{\mathrm{SPT}}_{l_2}$. We now have $\Delta T = (C^{\mathrm{SPT}}_{l_1} - d_i)^2 + (C^{\mathrm{SPT}}_{l_2} - d_j)^2 - (C^{\mathrm{SPT}}_{l_1} - d_j)^2 - (C^{\mathrm{SPT}}_{l_2} - d_i)^2$. Let $A = C^{\mathrm{SPT}}_{l_2} - C^{\mathrm{SPT}}_{l_1}$, $B = C^{\mathrm{SPT}}_{l_1} - d_j$ and $C = d_j - d_i$. We then have $\Delta T = (B+C)^2 + (A+B)^2 - B^2 - (A+B+C)^2$, with $A, B, C > 0$. Expanding the squared terms and simplifying, we obtain $\Delta T = -2AC < 0$.

For all the situations considered, we have $\Delta E \leq 0$ and $\Delta T \leq 0$. Hence, scheduling jobs i and j in EDD order gives a lower value for Z'. Consequently, we have $Z'_{\mathrm{EDD}} = \min Z'$ and $Z'_{\mathrm{EDD}} \leq \min Z$, which concludes the proof. □

2.2. *Special cases*

In this section, we present three special cases in which the lower bound is equal to the optimum objective function value. Let C^*_j and d^*_j denote the completion time

and the due date, respectively, of the job scheduled in the jth position in an optimal sequence. Also, let $Z^* = \sum_{j=1}^{n} \max(d_j^* - C_j^*, 0) + \sum_{j=1}^{n} [\max(C_j^* - d_j^*, 0)]^2$ denote the optimum objective function value. The following theorem provides three cases in which $Z'_{\text{EDD}} = Z^*$.

Theorem 2.2. *$Z'_{\text{EDD}} = Z^*$, i.e. the lower bounding procedure gives the optimum objective function value, for each of the following cases*:

Case 1. the EDD sequence is optimal and $p_j = p$, for all j;
Case 2. the EDD sequence is optimal, contains no tardy jobs, and is identical to the LPT sequence;
Case 3. the EDD sequence is optimal, contains no early jobs, and is identical to the SPT sequence.

Proof. For each of the three cases, we must show that the value provided by the lower bounding procedure is indeed equal to the objective function value of the optimal sequence.

Case 1. Given that the EDD sequence is optimal, we then have $d_j^{\text{EDD}} = d_j^*$, for all j. Also, since the processing times are identical for all jobs, we have $C_j^{\text{LPT}} = C_j^{\text{SPT}} = C_j^*$, for all j. Therefore, $Z'_{\text{EDD}} = \sum_{j=1}^{n} \max(d_j^{\text{EDD}} - C_j^{\text{LPT}}, 0) + \sum_{j=1}^{n} [\max(C_j^{\text{SPT}} - d_j^{\text{EDD}}, 0)]^2 = \sum_{j=1}^{n} \max(d_j^* - C_j^*, 0) + \sum_{j=1}^{n} [\max(C_j^* - d_j^*, 0)]^2 = Z^*$, so the lower bound is equal to the optimum objective function value.

Case 2. Since the EDD sequence is optimal, we again have $d_j^{\text{EDD}} = d_j^*$, for all j. Furthermore, the EDD and LPT sequences are identical, so $C_j^{\text{LPT}} = C_j^*$. Given that the optimal sequence contains no tardy jobs, we then have $\sum_{j=1}^{n} [\max(C_j^* - d_j^*, 0)]^2 = 0$ and $Z^* = \sum_{j=1}^{n} \max(d_j^* - C_j^*, 0) = \sum_{j=1}^{n} \max(d_j^{\text{EDD}} - C_j^{\text{LPT}}, 0)$. Also, we have $\sum_{j=1}^{n} [\max(C_j^{\text{SPT}} - d_j^{\text{EDD}}, 0)]^2 = \sum_{j=1}^{n} [\max(C_j^{\text{SPT}} - d_j^*, 0)]^2 = 0$, since $\sum_{j=1}^{n} [\max(C_j^* - d_j^*, 0)]^2 = 0$ and $C_j^{\text{SPT}} \leq C_j^{\text{LPT}} = C_j^*$. Therefore, $Z'_{\text{EDD}} = \sum_{j=1}^{n} \max(d_j^{\text{EDD}} - C_j^{\text{LPT}}, 0) = Z^*$.

Case 3. The EDD sequence is once more optimal, so we have $d_j^{\text{EDD}} = d_j^*$, for all j. Furthermore, since the EDD and SPT sequences are identical, we have $C_j^{\text{SPT}} = C_j^*$. Given that the optimal sequence contains no early jobs, we then have $\sum_{j=1}^{n} \max(d_j^* - C_j^*, 0) = 0$ and $Z^* = \sum_{j=1}^{n} [\max(C_j^* - d_j^*, 0)]^2 = \sum_{j=1}^{n} [\max(C_j^{\text{SPT}} - d_j^{\text{EDD}}, 0)]^2$. Also, we have $\sum_{j=1}^{n} \max(d_j^{\text{EDD}} - C_j^{\text{LPT}}, 0) = \sum_{j=1}^{n} \max(d_j^* - C_j^{\text{LPT}}, 0) = 0$, since $\sum_{j=1}^{n} \max(d_j^* - C_j^*, 0) = 0$ and $C_j^{\text{LPT}} \geq C_j^{\text{SPT}} = C_j^*$. Therefore, $Z'_{\text{EDD}} = \sum_{j=1}^{n} [\max(C_j^{\text{SPT}} - d_j^{\text{EDD}}, 0)]^2 = Z^*$, which concludes the proof. □

2.3. *Applicability and extensions*

The lower bounding procedure creates a modified objective function, based on a relaxation of the completion times. For any specific sequence, the value of the modified objective function value is less than or equal to the value of the original function.

The EDD ordering for the due dates is then shown to minimize the modified objective function, therefore providing a lower bound for the original objective. In this section, we present previous applications of this technique, and its generalization and applicability to other objective functions.

This lower bounding technique was previously used by Azizoglu *et al.* (1991) and Valente (2007) for two other early/tardy scheduling problems with no idle time. Azizoglu *et al.* (1991) applied this approach to derive a lower bound for the problem $\sum_{j=1}^{n}[(1-q)E_j + qT_j]$, with $0 < q < 1$, which is equivalent to the linear early/tardy problem with job-independent penalties $\sum_{j=1}^{n}(hE_j + wT_j)$. Valente (2007) instead considered the problem with job-dependent penalties and quadratic costs $\sum_{j=1}^{n}(h_jE_j^2 + w_jT_j^2)$. However, in order to derive a lower bound, the job penalties were first relaxed, in order to obtain a modified problem $\sum_{j=1}^{n}(h_{\min}E_j^2 + w_{\min}T_j^2)$, with $h_{\min} = \min\{h_j; j = 1,\ldots,n\}$ and $w_{\min} = \min\{w_j; j = 1,\ldots,n\}$. Therefore, the lower bounding technique was used to derive a lower bound for this modified problem with job-independent penalties. A somewhat similar approach was also used by Schaller (2004) for the $\sum_{j=1}^{n}(E_j + T_j^2)$ problem with inserted idle time. Therefore, and even though we focus on problems with no idle time, the lower bounding technique can also be extended to situations where the insertion of idle time is allowed.

The results given in this paper, together with those presented by Azizoglu *et al.* (1991) and Valente (2007), show that this technique provides a lower bound for each of the terms $\sum_{j=1}^{n}E_j$, $\sum_{j=1}^{n}E_j^2$, $\sum_{j=1}^{n}T_j$ and $\sum_{j=1}^{n}T_j^2$. Therefore, this approach can then be used to generate a lower bound for problems whose objective function is a convex combination of any of these four terms. This includes scheduling problems minimizing a sum of linear and/or quadratic earliness and tardiness, as well as problems with job-independent penalties.

The lower bounding technique is actually even more general since it can be used to obtain a lower bound for cubic and higher powers of the earliness and/or tardiness. Indeed, this approach yields a lower bound for any power $a \geq 1$. Let $E^a = \sum_{j=1}^{n}E_j^a$ and $T^a = \sum_{j=1}^{n}T_j^a$. Also, let $E_{\text{EDD}}^a = \sum_{j=1}^{n}[\max(d_j^{\text{EDD}} - C_j^{\text{LPT}}, 0)]^a$ and $T_{\text{EDD}}^a = \sum_{j=1}^{n}[\max(C_j^{\text{SPT}} - d_j^{\text{EDD}}, 0)]^a$. Therefore, in the problem considered in this paper we have $Z = E^1 + T^2$ and $Z'_{\text{EDD}} = E_{\text{EDD}}^1 + T_{\text{EDD}}^2$. The following theorem shows that the lower bounding technique can be generalized to all powers $a \geq 1$.

Theorem 2.3. *$E_{\text{EDD}}^a \leq \min E^a$ and $T_{\text{EDD}}^a \leq \min T^a$ for $a \geq 1$, i.e. E_{EDD}^a and T_{EDD}^a provide a lower bound for the optimal objective function value of the $\sum_{j=1}^{n}E_j^a$ and $\sum_{j=1}^{n}T_j^a$ functions, respectively.*

Proof. The results previously given in this paper, as well as those developed by Azizoglu *et al.* (1991) and Valente (2007), have already showed that the theorem is valid for $a = 1, 2$. A proof similar to those previously presented for these linear and quadratic functions could also be used to show that the same applies to other values of a. For brevity, however, we will remark that all the cases and subcases that would have to be considered in the general proof can be classified into one of

three categories. In the remainder of this proof, we first illustrate these categories with an example from the proof we previously presented for Theorem 2.1. More specifically, we will use as examples some of the cases considered for the quadratic tardiness component of our objective function. Then, we show that the proof given for that case is also valid for a general power $a \geq 1$.

Category 2.1. In this category, the proof of the case is obvious from the relations between the due dates and the completion times. For example, consider the proof of Case 2. In this case, we have $d_i \leq C_{l_2}^{\mathrm{SPT}} \leq d_j$, and the proof for $a = 2$ requires that $[\max(C_{l_1}^{\mathrm{SPT}} - d_i, 0)]^2 - (C_{l_2}^{\mathrm{SPT}} - d_i)^2 \leq 0$. Therefore, the proof for a general $a \geq 1$ would require that $[\max(C_{l_1}^{\mathrm{SPT}} - d_i, 0)]^a - (C_{l_2}^{\mathrm{SPT}} - d_i)^a \leq 0$. This inequality does hold, since $C_{l_1}^{\mathrm{SPT}} < C_{l_2}^{\mathrm{SPT}}$ and $d_i \leq C_{l_2}^{\mathrm{SPT}}$.

Category 2.2. In this category, the case can be proved by showing that $C^a + A^a - (A+B+C)^a \leq 0$. Case 3b gives an example of this category, since its proof requires that $C^2 + A^2 - (A + B + C)^2 \leq 0$. Since we have $A, B, C \geq 0$, it is then clear that we have $C^a + A^a - (A + B + C)^a \leq 0$.

Category 2.3. In this final category, the case can be proved by showing that $(B+C)^a + (A+B)^a - B^a - (A+B+C)^a \leq 0$. Case 3c provides an example of this category. We first recall that $A, B, C \geq 0$. Newton's binomial theorem can then be used to expand the expression $(B+C)^a+(A+B)^a-B^a-(A+B+C)^a$. It is then clear, from the expanded expression, that we have $(B+C)^a+(A+B)^a-B^a-(A+B+C)^a \leq 0$. For the sake of brevity, we omit the details. □

The lower bounding technique, however, cannot be used to obtain a lower bound for some other earliness and/or tardiness cost functions. In fact, simple counterexamples show that the approach fails for objective functions with powers $0 < a < 1$. Also, as previously mentioned, the technique is appropriate for objective functions that minimize a simple sum of earliness and tardiness costs, as well as for problems with job-independent penalties. However, this approach is not suited to the minimization of objective functions with job-dependent penalties, such as $\sum_{j=1}^{n} h_j E_j^2$ or $\sum_{j=1}^{n} w_j T_j$. As described above, Valente (2007) considered the problem with job-dependent penalties $\sum_{j=1}^{n} (h_j E_j^2 + w_j T_j^2)$. Nevertheless, a relaxation of the job penalties was first performed, and the lower bounding technique was then applied to a modified problem with job-independent penalties.

3. Branch-and-Bound Procedure

In this section, we discuss the implementation details of the branch-and-bound algorithm. We use a forward-sequencing branching rule, where a node at level l of the search tree corresponds to a sequence with l jobs fixed in the first l positions. The depth-first strategy is used to search the tree, and ties are broken by selecting the node with the smallest value of the associated partial schedule cost plus the associated lower bound for the unscheduled jobs. The tie-breaking strategy therefore

chooses the node that seems most promising, i.e. the node that will most likely lead to a complete schedule with a lower objective function value.

The initial upper bound on the optimum schedule cost is calculated using the EQTP–EXP dispatching rule. The EQTP–EXP heuristic calculates a priority index for each remaining job every time the machine becomes available, and the job with the highest priority is then selected to be processed next. Let $I_j(t)$ denote the priority index of job J_j at time t. The EQTP–EXP dispatching rule then uses the following priority index $I_j(t)$:

$$I_j(t) = \begin{cases} (1/p_j)[\overline{p} + 2(t + p_j - d_j)] & \text{if } s_j \leq 0 \\ (\overline{p}/p_j)\exp[-(\overline{p}+1)s_j/k\overline{p}] & \text{if } 0 < s_j < [\overline{p}/(\overline{p}+1)]k\overline{p} \\ (1/p_j)^{-2}[(\overline{p}/p_j) - (1/p_j)(\overline{p}+1)s_j/k\overline{p}]^3 & \text{if } [\overline{p}/(\overline{p}+1)]k\overline{p} \leq s_j < k\overline{p} \\ -(1/p_j) & \text{otherwise,} \end{cases}$$

where $s_j = d_j - t - p_j$ is the slack of job J_j, $\overline{p}$ is the average processing time of the remaining unscheduled jobs and k is a lookahead parameter. The EQTP_EXP procedure provided the best results among all the heuristics analyzed in Valente (2006). The upper bound value is then updated whenever a feasible schedule with a lower cost is found during the branching process.

Two fathoming tests are used to reduce the number of nodes in the search tree. In the first test, an insertion-based procedure is used to eliminate dominated nodes. This procedure inserts the job most recently added to the node's partial sequence before a certain number of the previously scheduled jobs. If an improving sequence is found, the node is then fathomed.

We considered six alternative versions of the branch-and-bound procedure. These versions differ only in the number (denoted by INS) of previously scheduled jobs that are considered in the insertion fathoming test. In the version INS = *single*, only one insertion is performed. In the version INS = 0.10 (respectively, 0.25, 0.50, 0.75 and 1.00), on the other hand, the number of insertions that are considered is equal to 10% (respectively, 25%, 50%, 75% and 100%) of the number of previously scheduled jobs.

When the node is not eliminated by the previous test, a lower bound is then calculated for the unscheduled jobs. If the lower bound plus the cost of the associated partial schedule is larger than or equal to the current upper bound, the node is discarded.

4. Computational Results

In this section, we first describe the set of test problems used in the computational experiments. Then, we analyze the performance of the lower bounding procedure. Finally, we present computational results for the branch-and-bound algorithms. Throughout this section, and in order to avoid excessively large tables, we will sometimes present results only for some representative cases.

4.1. *Experimental design*

The computational tests were performed on a set of problems with 7, 10, 12, 15, 17 and 20 jobs. These problems were randomly generated as follows. For each job J_j, an integer processing time p_j was generated from one of the two uniform distributions $[1, 10]$ and $[1, 100]$, in order to obtain a low (L) and a high (H) range, respectively, for the processing time values. For each job J_j, an integer due date d_j is generated from the uniform distribution $[P(1 - T - R/2), P(1 - T + R/2)]$, where P is the sum of the processing times of all jobs, T is the tardiness factor, set at 0.0, 0.2, 0.4, 0.6, 0.8 and 1.0, and R is the range of due dates, set at 0.2, 0.4, 0.6 and 0.8.

For each combination of problem size n, processing time range (rng), T and R, 50 instances were randomly generated. Therefore, a total of 1200 instances was generated for each combination of problem size and processing time range. All the algorithms were coded in Visual C++ 6.0, and executed on a Pentium IV–2.8 GHz personal computer.

4.2. *Lower bound results*

In this section, we analyze the performance of the lower bound. In Table 1, we present the average of the relative deviations from the optimum, calculated as $(O - LB)/O \times 100$, where O and LB represent the optimum objective function value and the initial lower bound value (i.e. the lower bound at the root node), respectively. From the results given in this table, we can see that the lower bound value is, on average, 25% to 30% below the optimum. Also, the performance of the lower bounding procedure is somewhat better when the range of processing times is low.

The effect of the T and R parameters on the relative deviation from the optimum is given in Table 2. The lower bound performs better when $T = 0.0$ or $T = 1.0$. Therefore, the lower bound value is closer to the optimum when nearly all jobs are early ($T = 0.0$) or tardy ($T = 1.0$). The performance of the lower bound is particularly good for instances where most jobs are early. In fact, the lower bound is usually quite close to the optimum for instances with $T = 0.0$, especially when the range of the due dates R is low.

The lower bound's relative deviation from the optimum then increases as the tardiness factor T approaches its intermediate values. Therefore, the performance of

Table 1. Relative deviation from the optimum.

n	Low rng	High rng
7	24.63	25.01
10	26.41	27.58
12	26.87	27.64
15	27.14	28.97
17	26.92	28.57
20	27.58	29.22

Table 2. Effect of the tardiness factor and the range of due dates on the relative deviation from the optimum.

n	T	Low rng				High rng			
		$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$	$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$
10	0.0	1.47	2.62	5.39	7.03	1.99	4.62	7.41	13.77
	0.2	18.33	24.46	28.82	37.44	15.38	32.93	29.65	41.77
	0.4	15.63	33.97	44.08	74.60	19.43	30.68	41.10	70.31
	0.6	17.74	29.40	42.60	54.73	18.52	29.28	45.83	53.36
	0.8	16.11	27.11	31.52	39.54	12.19	27.96	38.00	45.85
	1.0	10.47	19.03	23.89	27.77	10.65	17.73	24.76	28.84
15	0.0	0.90	2.33	3.85	5.23	1.55	2.68	5.31	6.79
	0.2	17.96	22.12	30.91	26.81	18.36	32.59	32.84	40.54
	0.4	20.02	34.44	51.57	77.10	22.33	33.33	42.23	71.11
	0.6	17.44	29.84	44.66	51.79	21.68	32.25	45.13	61.56
	0.8	16.33	26.89	39.39	46.17	17.19	27.97	38.50	50.32
	1.0	11.13	18.68	26.53	29.28	11.23	21.63	27.04	31.06
20	0.0	0.75	2.15	3.86	5.44	0.99	2.92	5.63	6.92
	0.2	16.89	20.39	24.19	30.12	21.24	30.38	29.10	30.43
	0.4	23.10	32.10	50.08	72.58	21.52	34.02	46.71	72.54
	0.6	20.83	34.37	42.61	59.09	20.44	31.56	47.97	64.53
	0.8	16.34	26.99	39.04	47.05	16.99	29.39	42.33	50.55
	1.0	12.12	20.75	27.59	33.43	11.52	22.72	28.05	32.72

the lower bounding procedure deteriorates when there is a greater balance between the number of early and tardy jobs.

The range of due dates parameter also has a noticeable impact on the effectiveness of the lower bound. In fact, the relative deviation from the optimum becomes higher as R increases. Consequently, the lower bound performs better when the due dates are quite close, and its performance deteriorates as the due dates become more widely spread.

4.3. *Branch-and-bound results*

The results for the branch-and-bound algorithms will be presented in this section. In Table 3, we give the branch-and-bound average computation times, in seconds. The branch-and-bound procedures are capable of solving, within reasonable computation times, problems with up to 20 jobs. The range of the processing times has a significant effect on the runtimes since the branch-and-bound procedures are much faster for instances with a high processing time range.

For the smaller instances, the computation times are similar for the various INS values. The difference in the runtimes, however, becomes much clearer for the larger instances with 17 and 20 jobs. For these instance sizes, the branch-and-bound procedures with INS = single or INS = 0.10 require a substantially higher computation time. The runtimes for the remaining INS values are somewhat close even though the best performance is given by the INS = 0.50 and INS = 0.75 algorithms. When the INS parameter is higher, the insertion fathoming test

Table 3. Branch-and-bound runtimes (in seconds).

rng	n	INS					
		single	0.10	0.25	0.50	0.75	1.00
L	7	0.000	0.000	0.000	0.001	0.001	0.001
	10	0.007	0.005	0.004	0.007	0.007	0.007
	12	0.042	0.029	0.027	0.039	0.038	0.041
	15	0.779	0.748	0.646	0.641	0.632	0.660
	17	7.534	7.638	5.967	5.370	5.344	5.503
	20	266.843	253.358	180.464	159.420	157.162	159.991
H	7	0.001	0.001	0.001	0.001	0.001	0.001
	10	0.006	0.007	0.006	0.006	0.006	0.006
	12	0.031	0.032	0.031	0.030	0.031	0.031
	15	0.551	0.558	0.474	0.429	0.432	0.446
	17	3.569	3.625	2.884	2.629	2.670	2.755
	20	83.140	80.036	52.758	48.010	49.241	50.761

will usually eliminate more nodes, but it will also require additional computation time. Therefore, increasing the number of insertions has two opposite effects on the efficiency of the branch-and-bound algorithm. The best results were then achieved when the number of insertions is relatively high (INS between 0.50 and 0.75), but still below its maximum possible value.

In Table 4, we present several additional statistics for the computation times, namely the minimum (min) and maximum (max) values, the coefficient of variation (cov), and the percentiles 25, 50, 75, 95 and 99 (p25, p50, p75, p95 and p99, respectively) of the distribution of the runtimes. The results in Table 4 once again show that the procedures with INS = single or INS = 0.10 require higher computation times. The remaining values of the INS parameter (particularly the values 0.50 and 0.75) provide superior results and are also significantly more consistent, since the variability in their runtimes is lower, as indicated by the cov values.

The improvement in performance provided by the higher values of the parameter INS becomes larger as the instance difficulty increases. For the easier instances (min, p25 and p50), which require low computation times, the runtimes are close for all INS values. As the instance difficulty, and correspondingly the runtime, increase, the branch-and-bound procedures with INS equal or greater than 0.25 become increasingly more efficient, since the increase in the runtime is much slower for these procedures. For the most difficult instances (p95, p99 and max), the computation times are substantially lower for the algorithms with a INS value of at least 0.25.

The effect of the T and R parameters on the computation times for instances with 20 jobs is given in Table 5. The results presented in this table show that the tardiness factor T has a clear and significant effect on the runtimes. Indeed, the branch-and-bound procedures are quite fast when $T = 0.0$ or $T = 1.0$. The runtimes increase significantly, however, as T approaches its intermediate values. Therefore,

Table 4. Branch-and-bound runtimes statistics.

rng	n	INS	min	p25	p50	p75	p95	p99	max	cov
L	15	single	0.000	0.063	0.187	0.735	3.578	8.227	16.375	208.0
		0.10	0.000	0.062	0.172	0.705	3.495	8.087	15.674	206.9
		0.25	0.000	0.061	0.170	0.629	3.044	6.399	11.729	193.0
		0.50	0.000	0.078	0.187	0.594	2.774	6.187	13.516	192.1
		0.75	0.000	0.075	0.186	0.589	2.660	6.189	13.527	191.1
		1.00	0.000	0.078	0.203	0.617	2.937	6.438	14.672	190.7
	20	single	0.000	2.672	10.196	96.945	1110.895	4408.280	22466.100	439.4
		0.10	0.000	2.632	10.343	99.732	1084.160	4128.345	18986.600	418.6
		0.25	0.000	2.562	8.693	76.031	770.061	3033.090	11926.600	402.0
		0.50	0.000	2.687	8.555	68.478	669.931	2478.330	9360.400	388.0
		0.75	0.000	2.812	9.166	66.734	703.789	2434.775	8724.430	372.2
		1.00	0.000	2.867	9.352	69.101	727.679	2472.930	8913.630	368.2
H	15	single	0.000	0.047	0.172	0.547	2.429	4.834	14.421	193.6
		0.10	0.000	0.047	0.172	0.562	2.470	5.001	14.953	194.6
		0.25	0.000	0.047	0.172	0.516	1.992	3.711	12.609	181.0
		0.50	0.000	0.047	0.157	0.469	1.758	3.406	12.516	181.1
		0.75	0.000	0.047	0.172	0.484	1.688	3.493	12.875	182.3
		1.00	0.000	0.062	0.172	0.500	1.758	3.626	13.328	182.4
	20	single	0.000	1.188	7.734	53.453	372.126	1410.945	3184.880	305.5
		0.10	0.000	1.196	7.742	52.624	384.672	1392.995	3135.770	297.0
		0.25	0.000	1.188	6.336	39.687	223.420	841.878	2327.460	287.2
		0.50	0.000	1.219	5.805	36.281	221.044	783.361	1779.420	287.7
		0.75	0.000	1.281	5.898	36.562	224.779	811.123	1921.410	292.4
		1.00	0.000	1.328	6.000	37.874	230.710	836.464	1980.080	292.8

Table 5. Branch-and-bound runtimes (in seconds) for instances with 20 jobs.

INS	T	Low rng				High rng			
		$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$	$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$
single	0.0	1.379	2.968	7.181	7.834	0.046	0.178	0.469	0.667
	0.2	144.480	54.097	14.172	10.206	19.489	10.494	2.346	2.020
	0.4	1675.648	602.005	257.300	108.419	63.151	157.273	206.859	63.196
	0.6	1815.821	1149.630	137.527	39.781	447.947	370.219	196.340	62.631
	0.8	266.418	45.660	17.196	16.536	232.023	81.376	34.557	12.363
	1.0	3.905	7.627	10.898	7.535	3.268	12.396	9.107	6.941
0.10	0.0	1.626	2.966	7.384	7.936	0.048	0.181	0.475	0.671
	0.2	143.606	53.186	14.338	10.307	19.284	10.128	2.336	2.020
	0.4	1509.562	588.645	258.094	108.411	60.043	148.263	191.502	62.291
	0.6	1706.668	1110.604	135.878	39.939	416.307	358.686	192.187	62.343
	0.8	271.429	46.051	17.221	16.463	235.718	80.196	33.937	12.219
	1.0	3.950	7.774	10.977	7.573	3.275	12.588	9.236	6.935
0.25	0.0	1.450	2.707	7.557	8.120	0.047	0.184	0.495	0.679
	0.2	145.280	50.477	14.465	10.295	19.351	9.500	2.198	1.984
	0.4	1160.361	499.310	205.845	90.866	54.412	118.689	158.936	50.635
	0.6	1047.229	740.272	88.983	32.016	263.000	224.767	117.488	39.794
	0.8	152.211	26.654	11.991	11.869	111.886	45.393	16.974	8.455
	1.0	3.803	5.856	6.976	6.541	3.018	7.893	5.755	4.659

Table 5. (*Continued*)

INS	T	Low rng				High rng			
		$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$	$R = 0.2$	$R = 0.4$	$R = 0.6$	$R = 0.8$
0.50	0.0	2.317	3.811	8.472	8.709	0.048	0.188	0.499	0.702
	0.2	158.499	50.847	15.077	10.809	19.619	9.753	2.241	2.039
	0.4	1045.781	486.379	187.088	77.886	54.738	117.973	160.013	47.774
	0.6	876.977	617.390	74.817	26.632	253.962	201.571	100.069	30.670
	0.8	116.064	19.067	9.257	9.953	80.746	34.797	11.159	6.872
	1.0	3.343	4.807	5.969	6.138	2.223	5.632	4.641	4.303
0.75	0.0	2.575	4.188	9.186	9.340	0.054	0.202	0.530	0.743
	0.2	176.349	53.486	15.972	11.474	20.361	10.243	2.357	2.154
	0.4	1040.547	502.557	190.703	77.972	56.982	123.203	169.323	49.786
	0.6	845.262	573.299	73.784	23.989	262.623	211.588	101.038	28.623
	0.8	103.792	17.531	9.053	9.396	75.960	30.699	11.193	6.868
	1.0	3.562	5.101	6.247	6.532	2.292	5.634	4.854	4.470
1.00	0.0	1.847	3.741	9.361	9.710	0.057	0.208	0.559	0.778
	0.2	184.757	56.006	16.632	11.913	20.876	10.627	2.432	2.219
	0.4	1067.915	518.150	196.666	79.596	59.031	127.459	175.826	51.451
	0.6	852.607	567.286	74.987	24.976	269.587	217.613	104.450	29.956
	0.8	103.458	18.285	9.472	9.747	76.624	31.253	11.803	7.235
	1.0	3.821	5.406	6.556	6.895	2.437	5.953	5.120	4.716

the problem becomes harder to solve when there is a greater balance between the number of early and tardy jobs.

In Table 6, we present the average number of nodes generated by the branch-and-bound algorithm (NG), as well as the average percentage of these nodes that were eliminated by the two fathoming tests (%EL). We also provide some data on the relative importance of these tests, namely the average percentage of nodes fathomed by the lower bound test (%LB) and the insertion-based procedure (%INS). The number of nodes generated is higher for the instances with a low processing time range. Additionally, this number decreases with the value of the INS parameter. These results are in line with the computation times previously presented in Table 3.

The proportion of nodes eliminated by the fathoming tests increases with the instance size, and is marginally higher for instances with a high processing time range. Also, the percentage of eliminated nodes increases very slightly with the INS parameter. The relative importance of the lower bound fathoming test is lower for the instances with a high processing time range, and decreases with the instance size, while the proportion of nodes eliminated by the insertion procedure correspondingly increases. The relative importance of the insertion-based test also increases with the INS parameter. This is to be expected since more insertions are performed for the higher INS values.

In Table 7, we present the effect of the T and R parameters on the average number of nodes generated and the percentage of nodes eliminated by the lower

Table 6. Average number of nodes and relative importance of the fathoming tests.

n	INS	Low rng				High rng			
		NG	%EL	%LB	%INS	NG	%EL	%LB	%INS
10	single	1683	83.08	49.65	50.35	1635	83.25	44.29	55.71
	0.10	1683	83.08	49.65	50.35	1635	83.25	44.29	55.71
	0.25	1667	83.14	49.01	50.99	1620	83.31	43.69	56.31
	0.50	1596	83.27	47.56	52.44	1554	83.44	42.39	57.61
	0.75	1555	83.32	46.89	53.11	1516	83.50	41.81	58.19
	1.00	1552	83.33	46.83	53.17	1515	83.50	41.78	58.22
15	single	202219	87.78	43.61	56.39	143699	88.19	38.12	61.88
	0.10	202171	87.78	43.58	56.42	143633	88.20	38.08	61.92
	0.25	176435	87.96	41.44	58.56	122348	88.42	35.73	64.27
	0.50	157738	88.07	40.04	59.96	107782	88.52	34.47	65.53
	0.75	151430	88.09	39.60	60.40	103124	88.55	34.12	65.88
	1.00	150659	88.09	39.54	60.46	102803	88.55	34.08	65.92
20	single	64614975	90.27	40.55	59.45	19979272	90.76	35.24	64.76
	0.10	60581125	90.32	39.90	60.10	19039709	90.81	34.59	65.41
	0.25	43370259	90.51	37.13	62.87	12725387	91.05	31.64	68.36
	0.50	36809014	90.56	36.05	63.95	10986024	91.11	30.72	69.28
	0.75	34645609	90.58	35.74	64.26	10590399	91.12	30.52	69.48
	1.00	34182075	90.58	35.69	64.31	10550631	91.12	30.50	69.50

Table 7. Nodes generated and importance of lower bound test for INS = 0.50.

rng	n	T	$R = 0.2$		$R = 0.4$		$R = 0.6$		$R = 0.8$	
			NG	%LB	NG	%LB	NG	%LB	NG	%LB
L	10	0.0	215	87.49	383	84.78	533	78.35	603	75.06
		0.2	1654	43.57	1287	53.34	1101	56.82	958	59.17
		0.4	3683	27.19	3263	32.10	2318	39.38	1091	43.37
		0.6	3545	38.60	2931	38.10	2390	35.81	1479	37.27
		0.8	2136	43.84	1688	36.88	1217	41.45	1124	39.54
		1.0	1018	53.57	1128	43.61	1271	41.61	1294	37.67
	15	0.0	15432	84.81	17071	78.77	20627	75.11	17329	69.79
		0.2	347280	29.02	70254	45.58	70331	45.82	52411	55.71
		0.4	565606	20.75	515549	26.11	263003	33.89	117987	37.20
		0.6	457500	36.42	423822	32.64	262144	28.06	56996	29.07
		0.8	134916	36.42	113028	29.74	75948	24.28	53906	24.43
		1.0	22164	42.18	37504	31.97	39427	29.89	35472	29.84
	20	0.0	440300	82.77	770813	75.06	1948321	68.64	2003348	61.48
		0.2	48999699	23.65	13395740	44.10	4097621	46.52	2565839	45.50
		0.4	258166460	23.26	111282106	26.91	44457661	30.37	18573874	38.23
		0.6	169508421	34.51	145896177	25.25	19655283	23.54	7054485	22.41
		0.8	20650575	33.20	4317269	24.21	2415883	21.04	2693626	20.63
		1.0	610871	34.08	1000351	26.66	1376567	22.48	1535043	19.82

Table 7. (*Continued*)

rng	n	T	$R = 0.2$		$R = 0.4$		$R = 0.6$		$R = 0.8$	
			NG	%LB	NG	%LB	NG	%LB	NG	%LB
H	10	0.0	164	81.35	340	74.08	446	66.35	632	60.62
		0.2	1822	25.68	1398	37.18	790	50.20	1079	46.81
		0.4	3163	21.12	3033	24.80	2420	28.55	1839	34.62
		0.6	3163	37.22	2806	34.04	2416	33.87	1456	33.71
		0.8	1424	48.41	1960	37.22	1592	33.26	1441	33.82
		1.0	652	55.12	940	47.24	1217	43.53	1092	42.30
	15	0.0	2229	78.74	2236	73.81	7031	63.58	9086	58.15
		0.2	128102	8.94	67324	25.97	28272	34.16	37252	38.71
		0.4	231050	19.04	245601	23.92	213025	23.60	109675	29.95
		0.6	336058	34.21	339098	28.52	218147	23.63	87756	20.93
		0.8	167745	35.67	84906	28.25	78040	26.37	68863	23.18
		1.0	17755	43.53	37175	31.47	36066	28.50	34288	29.33
	20	0.0	10327	79.53	43650	65.55	131308	56.71	185452	52.43
		0.2	8320655	5.26	3078434	22.36	658317	33.89	561120	35.87
		0.4	13618711	20.48	28010326	24.62	39339572	22.61	12955845	24.55
		0.6	47595664	33.30	44118348	27.65	25162764	21.68	7908324	19.16
		0.8	15643648	31.26	7958865	22.42	2802132	20.04	1887911	17.22
		1.0	398030	37.00	1218970	23.40	1049279	23.28	1006924	21.30

bound test for INS = 0.50. The tardiness factor T has a clear and significant effect on the number of nodes generated. In fact, this number is much lower when $T = 0.0$ or $T = 1.0$, and then increases significantly as T approaches its intermediate values. Once more, this result is in accordance with the runtimes previously given in Table 5. The lower bound test eliminates a large percentage of nodes for instances with $T = 0.0$. This is to be expected since the lower bound performs particularly well when most jobs are early, as mentioned before. For the remaining parameter values, however, the insertion-based test usually fathoms a larger proportion of the nodes.

5. Conclusion

In this paper, we considered the single machine scheduling problem with linear earliness and quadratic tardiness costs, and no machine idle time. We developed a lower bound based on the relaxation of the jobs' completion times. An optimal branch-and-bound algorithm was then presented. This algorithm incorporates the proposed lower bound, as well as an insertion-based fathoming test.

The branch-and-bound procedures were tested on a wide set of randomly generated problems. These algorithms were capable of solving, within reasonable computation times, instances with up to 20 jobs. The best results were obtained by setting the INS parameter required by the insertion dominance procedure to 0.50 or 0.75. In fact, these values not only provided the lowest average computation times, but were also more consistent. Also, as the instance difficulty, and correspondingly the runtime, increased, the branch-and-bound procedures with INS equal to 0.50 or

0.75 became increasingly more efficient, since the increase in the runtime was much slower for these procedures.

Acknowledgment

The author would like to thank the anonymous referees and an associate editor for several, and most useful, comments and suggestions that were used to improve this paper.

References

Azizoglu, M, S Kondakci and O Kirca (1991). Bicriteria scheduling problem involving total tardiness and total earliness penalties. *International Journal of Production Economics*, 23, 17–24.

Baker, KR and GD Scudder (1990). Sequencing with earliness and tardiness penalties: A review. *Operations Research*, 38, 22–36.

Garey, MR, RE Tarjan and GT Wilfong (1988). One-processor scheduling with symmetric earliness and tardiness penalties. *Mathematics of Operations Research*, 13, 330–348.

Gupta, SK and T Sen (1983). Minimizing a quadratic function of job lateness on a single machine. *Engineering Costs and Production Economics*, 7, 187–194.

Kanet, JJ and V Sridharan (2000). Scheduling with inserted idle time: Problem taxonomy and literature review. *Operations Research*, 48, 99–110.

Kim, YD and CA Yano (1994). Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates. *Naval Research Logistics*, 41, 913–933.

Korman, K (1994). A pressing matter. *Video*, 46–50.

Landis, K (1993). Group technology and cellular manufacturing in the Westvaco Los Angeles VH department. Project report in IOM 581, School of Business, University of Southern California.

Schaller, J (2002). Minimizing the sum of squares lateness on a single machine. *European Journal of Operational Research*, 143, 64–79.

Schaller, J (2004). Single machine scheduling with early and quadratic tardy penalties. *Computers & Industrial Engineering*, 46, 511–532.

Sen, T, P Dileepan and MR Lind (1995). Minimizing a weighted quadratic function of job lateness in the single machine system. *International Journal of Production Economics*, 42, 237–243.

Su, L-H and P-C Chang (1998). A heuristic to minimize a quadratic function of job lateness on a single machine. *International Journal of Production Economics*, 55, 169–175.

Sun, X, JS Noble and CM Klein (1999). Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness. *IIE Transactions*, 31, 113–124.

Taguchi, G (1986). *Introduction to Quality Engineering*. Asian Productivity Organization, Tokyo, Japan.

Valente, JMS (2006). Heuristics for the single machine scheduling problem with early and quadratic tardy penalties. *European Journal of Industrial Engineering*, 1, 431–448.

Valente, JMS (2007). An exact approach for single machine scheduling with quadratic earliness and tardiness penalties. Working Paper 238, Faculdade de Economia, Universidade do Porto, Portugal.

Wagner, BJ, DJ Davis and H Kher (2002). The production of several items in a single facility with linearly changing demand rates. *Decision Sciences*, 33, 317–346.

Jorge M. S. Valente is Assistant Professor in the Management Department of the Faculty of Economics, University of Porto (Portugal). He holds a Ph.D. degree in Management Science from the University of Porto. He has published in Asia-Pacific Journal of Operational Research, Computers & Industrial Engineering, Computers & Operations Research, International Journal of Production Economics, Journal of Manufacturing Systems and in the Journal of the Operational Research Society, among others. His current research interests include production scheduling, combinatorial optimization, heuristic techniques and agent-based computational economics.