A NOVEL B AND B ALGORITHM FOR A UNRELATED PARALLEL MACHINE SCHEDULING PROBLEM TO MINIMIZE THE TOTAL WEIGHTED TARDINESS

R. Tavakkoli-Moghaddam*and M. Aramon-Bajestani

Department of Industrial Engineering and Center of Excellence for Intelligence Based Experimental Mechanics
College of Engineering, University of Tehran
P.O. Box 11155-4563, Tehran, Iran
tavakoli@ut.ac.ir - maliheha@gmail.com

*Corresponding Author

(Received: October 5, 2008 – Accepted in Revised Form: February 19, 2009)

Abstract This paper presents a scheduling problem with unrelated parallel machines and sequence-dependent setup times that minimizes the total weighted tardiness. A new branch-and-bound (B and B) algorithm is designed incorporating the lower and upper bounding schemes and several dominance properties. The lower and upper bounds are derived through an assignment problem and the composite dispatching rule (ATCS), respectively. We carry out computational experiments and the related results are reported.

Keywords Parallel Machines Scheduling, Sequence-Dependent Setup Times, Total Weighted Tardiness, Branch-And-Bound Algorithm

چکیده این مقاله مسئلهٔ زمان‌بندی ماشین‌های موازی نامرتبط با زمان‌های آماده‌سازی وابسته به توالی را بررسی می‌کند که کل دیرکرد وزنی را کاهش می‌دهد. الگوریتم شاخه و کران جدیدی شامل رویکردهایی برای یافتن حد پایین و حد بالا و قوانین مبتنی‌بر ویژگی‌های مسئله طراحی شده است. حد پایین و حد بالا به‌ترتیب ازطریق حل مسئله تخصیص و به‌کمک یکی از قانون‌های دیسپچینگ مرکب (ATCS) حاصل شده است. در انتها نتایج حاصل‌از اجرای الگوریتم پیشنهاد شده عرضه شده است.

1. INTRODUCTION

Production scheduling is a decision-making process that plays a vital role in industries. Scheduling problems are often categorized by the machine configuration and constraints. Among the various types of machine configurations, parallel machine models are important in practice, in that such models bring about more system capacity and flexibility, while processing different jobs. Parallel machine scheduling problems aim to allocate a set of jobs on a number of parallel machines, so that the customer's requirements can be met. In scheduling problems, machines often have to be prepared between jobs. This process is considered as a setup. Should the setups depend on both the job to be processed and one just completed, setup times are sequence-dependent.

Although there are many studies on scheduling problems, developing exact algorithms especially for the problems involving sequence-dependent setup times is so scarce. We focus on some studies addressing either designing exact algorithms or considering sequence-dependent setup times. Dessouk [1] designed a branch-and-bound procedure to schedule identical jobs with unequal ready times on uniform parallel machines that minimized the maximum lateness. An initial solution as an upper bound was established by using some heuristic procedures. Azizoglu, et al [2] addressed the unrelated parallel machine scheduling problem to minimize the total weighted flow time through designing a branch-and-bound algorithm incorporating some properties of optimal solution and a lower bound scheme. Liaw, et al [3] developed an efficient lower and upper bound to minimize the total tardiness in the unrelated parallel machine scheduling problem. The upper bound was derived

from a two-phase heuristic method, while the solution of an assignment problem resulted in the lower bound. Rabadi, et al [4] proposed a branch-and-bound (B and B) algorithm to solve a single machine early/tardy scheduling problem with unrestricted common due date and sequence-dependent setup times. Luo, et al [5] proposed a B and B algorithm including the implementation of lower and upper bounding procedures, and dominance rules to solve a single machine scheduling problem with sequence-dependent setup times that minimized the total tardiness. Shim, et al [6] suggested a B and B algorithm by developing several dominance rules and lower bound for an unrelated parallel machine scheduling problem with the objective of minimizing the total tardiness.

Pfund, et al [7] extended the composite dispatching rule proposed by Lee, et al [8] to minimize the total weighted tardiness on identical parallel machines with sequence dependent setup times. First, they developed a grid approach which considered multiple values for scaling parameters, and then used the experimentation to develop regression equations to predict the scaling parameters. Park, et al [9] considered the identical parallel machine scheduling problem with sequence dependent setup times. The objective of schedule was to minimize total weighted tardiness. They introduced a new factor, in addition to four factors proposed by Lee, et al [10], to get more accurate values for look-ahead parameters. They designed a neural network to predict the look-ahead parameters. Logendran, et al [11] considered the problem of unrelated parallel machine scheduling with both the machine and sequence-dependent setup times by including dynamism for the job release and machine availability. They also considered six different search algorithms based on tabu search to minimize the weighted tardiness, and investigated the efficacy and efficiency of the proposed algorithm through an extensive statistical analysis.

Anghinolfi, et al [12] addressed a parallel machine scheduling problem with non-zero ready times and sequence-dependent setup times and designed a hybrid meta-heuristic (HMH) approach that combines several aspects from a subset of well-known meta-heuristics. The experimental analysis showed that such an approach was suitable to face the total tardiness criteria. Tavakkoli-Moghaddam, et al [13] proposed a novel multi-objective model of a scheduling problem of unrelated parallel machines with different speeds that minimizes the number of tardy jobs and total completion time of all jobs. They considered sequence-dependent setup times, and some precedence relations between jobs with non-identical due dates and ready times. Tavakkoli-Mogahddam, et al [14] presented an integer-linear programming (ILP) model for an identical parallel-machine scheduling problem with family setup times that minimizes the total weighted flow time (TWFT). They proposed genetic algorithms to solve this model.

Nessah, et al [15] constructed a B and B algorithm which incorporates the heuristic, the lower bound, and the dominance properties to minimize the total completion time in an identical parallel machine scheduling problem with sequence-dependent setup times and release dates. Rocha, et al [16] designed a B and B algorithm to minimize the additive functions including the makespan and weighted tardiness criteria in an unrelated parallel scheduling problem with sequence and machine-dependent setup times. The dominance rules have not been incorporated in the proposed exact algorithm.

According to the literature review, there is no research work regarding the design of a B and B algorithm for the unrelated parallel machine scheduling with sequence-dependent setup times. Hence, in this paper by development of lower and upper bounding schemes and dominance properties, a B and B algorithm is designed to minimize the total weighted tardiness. The main contributions of this paper can be summarized as follows:

- Presenting new dominance properties considering sequence-dependent setup times in unrelated parallel machines scheduling.
- Developing the lower bound and upper bound scheme based on the assignment problem and composite dispatching rule (ATCS), respectively, applied for the first time for an unrelated parallel machine scheduling problem with sequence-dependent setup times.
- Presenting the new concept of partial pruning, in which only the children of the parent node possessing specific features are to be pruned.

The reminder of paper is organized as follows. The mathematical formulation is provided in Section 2. Section 3 presents several dominance properties. The lower bound scheme and upper bound procedure are presented in the succeeding sections. The proposed Band B algorithm is developed in Section 6. The associated computational results are given in Section 7. Finally, the remarking conclusion is provided in Section 8.

2. PROBLEM DESCRIPTION

2.1. Notations

i, j Job indices ($i, j \in J$; $i, j = 1, 2, ..., n$)
k Machine index ($k \in M$; $k = 1, 2, ..., m$)
t Position index ($t \in P$; $t = 1, 2, ..., n$)
S_p Set of scheduled jobs in the partial schedule p
S_u Set of unscheduled jobs
S_v Set of unfilled positions
P_{ik} Processing time of job i on machine k
d_i Due date of job i
w_i Priority (relative importance) of job i
S_{ij} Setup time to switch from job i to job j
AP^k_{ij} The adjusted processing time for job j on machine k when job i precedes job j
C^t_{ik} The completion time of job i on machine k scheduled in the t-th position

$$x^t_{ik} = \begin{cases} 1 & \text{If job } i \text{ scheduled in the t - th position on machine } k \\ 0 & \text{Otherwise} \end{cases}$$

The processing and setup times are combined since it makes the problem more appropriate to be studied. So, the adjusted processing time matrix for any machine k, $[AP^k]$ is defined as follows:

$$AP^k_{ij} = P_{jk} + S_{ij};\ (i, j = 1, 2, ..., n)$$

Rabadi, et al [4] also used a similar approach in a single machine environment.

2.2. Assumptions

- Each job requires an operation that can be processed on each machine.
- The setup time between two distinct jobs depends on jobs sequence.
- All jobs become available for processing simultaneously.
- Processing times of jobs on different machines may be varied.
- Preemptions and machine breakdown are not allowed.
- All processing times and due dates are deterministic.
- There is the triangular inequality among setup times, that is $S_{ij} + S_{jk} \geq S_{ik}$

2.3. Mathematical Model Considering the above-mentioned assumptions and notations, the problem is formulated as the following integer programming problem (P). The approach is similar to the models presented in [2 and 3].

Problem P:

$$Z_p = \min \sum_{i=1}^{n} \sum_{k=1}^{m} \sum_{t=1}^{n} f(C^t_{ik}) x^t_{ik} \quad (1)$$

s.t.

$$\sum_{k=1}^{m} \sum_{t=1}^{n} x^t_{ik} = 1, \quad \forall \text{i} \quad (2)$$

$$\sum_{i=1}^{n} x^t_{ik} \leq 1, \quad \forall \text{t,k} \quad (3)$$

$$C^t_{ik} = \sum_{\substack{l=1 \\ l \neq i}}^{n} \sum_{\substack{q=1 \\ q \neq i}}^{n} \sum_{s=2}^{t-1} AP^k_{lq} x^{s-1}_{lk} x^s_{qk} + \sum_{l=1}^{n} AP^k_{ll} x^1_{lk} + \sum_{\substack{l=1 \\ l \neq i}}^{n} AP^k_{li} x^{t-1}_{lk} \quad (4)$$

$$f(C^t_{ik}) = w_i \max\{0, C^t_{ik} - d_i\} \quad (5)$$

$$x^t_{ik} \in \{0, 1\}, \quad \forall \text{i,k,t} \quad (6)$$

Equation 2 ensures that each job is only assigned to one position on one machine. Constraint (3) implies that at most one job can be assigned to each position on each machine. The completion time of job i scheduled in the t-th position on machine k is given by Equation 4. Equation 5

describes the considered objective function, called total weighted tardiness. Finally, Equation 6 shows the integrality constraint.

3. DOMINANCE PROPERTIES

In this section, we present several properties of an optimal schedule leading to the substantial improvements in the algorithm performance. The following proposition generalizes the one given by Azizoglu, et al [2] considering sequence-dependent setup times.

3.1. Proposition There exists an optimal schedule in which the sum of processing times of the jobs assigned on machine k does not exceed

$$A_k = \frac{1}{m}\left[\begin{array}{l} \sum_{j\in J} \max_{r\in M} (\max_{i\in J} AP_{ij}^r) + \\ \sum_{r\in M} \max_{j\in J} (\max_{i\in J} AP_{ij}^r) \\ r\neq k \end{array}\right] \quad (7)$$

Proof Let l_k be the last job to be processed on machine k, and C_{l_k} be the completion time of l_k in an optimal schedule. For any other machine r ($r\neq k$), it must be

$$\begin{aligned} &C_{l_r} + AP_{l_r,l_k}^r \geq C_{l_k}, \quad \forall r \\ &C_{l_r} \geq C_{l_k} - AP_{l_r,l_k}^r, \quad \forall r \end{aligned} \quad (8)$$

With l_k having been taken from machine k and placed on machine r, the total cost (i.e., the total weighted tardiness) would decrease or not change if Equation 8 did not hold. By summing Equation (8) over all machines $r \neq k$, we obtain

$$\sum_{r\in M} C_{l_r} - C_{l_k} \geq (m-1)C_{l_k} - \sum_{r\neq k} AP_{l_r,l_k}^r$$

This implies that

$$C_{l_k} \leq \frac{1}{m}\left[\sum_{r\in M} C_{l_r} - \sum_{r\neq k} AP_{l_r,l_k}^r\right]$$

Taking into account that

$$\sum_{r\in M} C_{l_r} \leq \sum_{j\in J} \max_{r\in M} (\max_{i\in J} AP_{ij}^r)$$

and

$$AP_{l_r,l_k}^r \leq \max_{j\in J} (\max_{i\in J} AP_{ij}^r)$$

Consequently

$$C_{l_k} \leq \frac{1}{m}\left[\begin{array}{l} \sum_{j\in J} \max_{r\in M} (\max_{i\in J} AP_{ij}^r) + \\ \sum_{r\in M} \max_{j\in J} (\max_{i\in J} AP_{ij}^r) \\ r\neq k \end{array}\right]$$

For any machine k, the value of A_k can represent an upper bound on the total processing times of the jobs assigned to machine k. Thus, the node associated with the partial schedule p will be pruned in the case the completion time for a job, say job i, in a partial schedule is greater than or equal to the A_k value; that is $C_{ik}(p) \geq A_k$. The following proposition is the direct result of Proposition 1.

3.2. Proposition There exists an optimal schedule in which job j processed last on any one of the machines if

$$d_j = \max_{i\in J} d_i, \text{ and } d_j + \min_{i\in J} AP_{ij}^k \geq A_k \quad \text{for all } k$$

Proof According to Corollary 1 in Shim, et al [6] developed to minimize the total tardiness in the unrelated parallel machine scheduling, if there is a job j whose due date is the largest one and $d_j + P_{jk} \geq A_k$ for all k, there is an optimal schedule in which job j can be processed last on any machine. Replacing P_{jk} with $\min_{i\in J} AP^k{}_{ij}$ proves Proposition 2.

The following two properties are the generalization of the ones developed by Luo, et al [5] for a single machine tardiness problem. In the following propositions, job, [i] refers to the index of the job scheduled in the i-th position and the following notations is used.

$J(\pi,k)$ Set of jobs in the partial schedule π assigned on machine k.

$(\pi,k)|u$ New partial schedule obtained by assigning job u on machine k preceded by the given partial schedule π on machine k.

$\sum(\pi,k)|u$ Completed schedule composed of $(\pi,k)|u$, in which the jobs belonging to J-$J((\pi,k)|u)$ assigned to any machine except k.

3.3. Proposition

If $\exists i,[i] \in J(\pi,k)$,

$$\Delta_2 = S_{[i-1][l]} + P_{[l]k} + S_{[l][i+1]} - S_{[i-1][i]} - P_{[i]k} - S_{[i][i+1]} \le 0,$$

$$\Delta_3 = S_{[i-1][l]} + S_{[l][i+1]} + S_{[l-1][i]} + S_{[i]u} - S_{[i-1][i]} - S_{[i][i+1]} - S_{[l-1][l]} - S_{[l]u} \le 0,$$

and

$\Delta_T = T'_{[i]} - T_{[i]} + T'_{[l]} - T_{[l]} \le 0,$ there is a schedule that dominates $\sum(\pi,k)|u$.

Proof The proposition is proved by illustrating that the total weighted tardiness decreases or remains unchangeable by exchanging job [i] and job [l] in the sequence for machine k. Let the schedule be $S = \sum(\pi,k)|u$. Consider another schedule S' obtained by interchanging the position of job [i] and job [l] (see Figure 1).

In Part 1: $C_1 = C'_1, T_1 = T'_1$

In Part 2: The completion time of job [j], [i+1] ≤ [j] ≤ [l-1] has a change of Δ_2. Given that $\Delta_2 \le 0$; that is, $C_{[j]} \ge C'_{[j]}$. We consider the following two cases of $d_{[j]} \ge C_{[j]}$ and $d_{[j]} \le C_{[j]}$. Then, we have

Case 1: If $d_{[j]} \ge C_{[j]}$, then $d_{[j]} \ge C'_{[j]}$. So,

$$\Delta T_{[j]} = w_{[j]} \max(0, C'_{[j]} - d_{[j]}) - w_{[j]} \max(0, C_{[j]} - d_{[j]}) = w_{[j]}(0) + w_{[j]}(0) = 0$$

Case 2: If $d_{[j]} \le C_{[j]}$, then we could consider two sub cases: (a) $d_{[j]} \le C'_{[j]}$ and (b) $d_{[j]} \ge C'_{[j]}$

(a) If $d_{[j]} \le C'_{[j]}$; that is, $C'_{[j]} - d_{[j]} \ge 0$, then

$$\Delta T_{[j]} = w_{[j]}(C'_{[j]} - d_{[j]}) - w_{[j]}(C_{[j]} - d_{[j]}) = w_{[j]}(C'_{[j]} - C_{[j]}) \le 0$$

(b) If $d_{[j]} \ge C'_{[j]}$; that is, $C'_{[j]} - d_{[j]} \le 0$, then

$$\Delta T_{[j]} = w_{[j]}(0) - w_{[j]}(C_{[j]} - d_{[j]}) = w_{[j]}(d_{[j]} - C_{[j]}) \le 0$$

Therefore, because $\Delta_2 \le 0$, the weighted tardiness of no job [j] increased.

In Part 3, given that $\Delta_3 \le 0$, $C_u \ge C'_u$, following exactly the same procedure as that for Part 2 proves that the weighted tardiness of job u does not increase. With respect to jobs [i] and [l], the change of the weighted tardiness is $\Delta_T = T'_{[i]} - T_{[i]} + T'_{[l]} - T_{[l]} \le 0$. If $\Delta_T \le 0$, the sum of weighted tardiness of jobs [i] and [l] decreased or did not changed. Moreover, the weighted tardiness of each of other jobs does not increase since $\Delta_2 \le 0$ and $\Delta_3 \le 0$. So, the total weighted tardiness of jobs does not increase by this interchanging. This completes the proof.

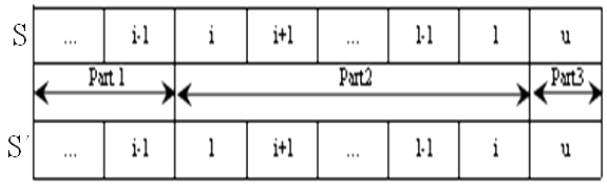

Figure 1. Interchanging the position of jobs [i] and [l].

3.4. Proposition

If $\exists i,[i] \in J(\pi,k)$,

$$\Delta_2 = S_{[i-1][i+1]} - S_{[i-1][il]} - S_{[i][i+1]} - P_{[i]k} \le 0,$$

$$\Delta_3 = S_{[i-1][i+1]} + S_{[l][i]} + S_{[i]u} - S_{[i-1][i]} - S_{[i][i+1]} - S_{[l]u} \le 0,$$

and

$\Delta_{Ti} = T'_{[i]} - T_{[i]} \le 0$, there is a schedule that dominates $\sum(\pi,k)|u$.

Proof Consider the schedule $S = \sum(\pi,k)|u$. We construct another schedule S' by putting job [i] between job [l] and job u (see Figure 2).

The completion time of job $[j]$, $[i+1] \le [j] \le [l]$, has a change of $\Delta_2 = S_{[i-1][i+1]} - S_{[i-1][i1]} - S_{[i][i+1]} - P_{[i]k}$. Since the assumption of the triangle inequality is given among setup times, it is clear that $\Delta_2 \le 0$. Following the same procedure used for Part 1 of the proposition 3 proves that the weighted tardiness of job [j] is decreased or unchanged.

Let consider job u, because $\Delta_3 \le 0$, we have $C_u \ge C'_u$. Thus, the weighted tardiness of job u is not increased. With respect to job [i], the change of weighted tardiness is $\Delta_{Ti} = T'_{[i]} - T_{[i]}$. If $\Delta T_i \le 0$, the weighted tardiness of job [i] is decreased or unchanged. To summarize, since the weighted tardiness of each job does not increase, the $\sum(\pi,k)|u$ is dominated.

4. LOWER BOUND

By solving an assignment problem, a lower bounding procedure is provided in this section. This approach was proposed by Azizoglu, et al [2]. We generalize this approach to find a lower bound for unrelated parallel machine with sequence-dependent setup times.

Figure 2. Putting jobs [i] and [l].

Replacing the exact completion times (C^t_{ik}) with the rough ones $(\hat{C}^t_{ik})$ is the main feature of this lower bounding scheme that results into defining an assignment problem as follows:

Problem AP:

$$Z_{Ap} = \min \sum_{i=1}^{n} \sum_{k=1}^{m} \sum_{t=1}^{n} f(\hat{C}^t_{ik}) x^t_{ik} \tag{9}$$

s.t.

$$\sum_{k=1}^{m} \sum_{t=1}^{n} x^t_{ik} = 1, \quad \forall i \tag{10}$$

$$\sum_{i=1}^{n} x^t_{ik} \le 1, \quad \forall t,k \tag{11}$$

$$f(\hat{C}^t_{ik}) = w_i \max\{0, \hat{C}^t_{ik} - d_i\} \tag{12}$$

$$x^t_{ik} \in \{0,1\}, \quad \forall i,k,t \tag{13}$$

Provided that the rough completion times $(\hat{C}^t_{ik})$ are smaller than or equal to the exact completion times (C^t_{ik}) for all i, k, and t, the optimal solution of the above assignment problem will be a lower bound on the original problem. The following definition of $(\hat{C}^t_{ik})$ supports the conditions.

$$\hat{C}^1_{ik} = AP^k_{ii}, \quad \forall i,k$$

$$\hat{C}^t_{ik} = \min_{j \ne i} \left(AP^k_{ji} + C^{t-1}_{jk} \right), \quad \forall i,k,t \text{, and } t \ne 1$$

To find a lower bound, we adhere to the same procedure used by Azizoglu, et al [2]. We solve the assignment problem only at the root node and use the dual values at others nodes. The dual problem of AP is defined as follows:

$$Z_{Ap} = \max \sum_{i=1}^{n} u_i + \sum_{t=1}^{n} \sum_{k=1}^{m} v_{tk} \tag{14}$$

s.t.

$$u_i + v_{tk} \le w_i \hat{T}^t_{ik}, \quad \forall i,t,k \tag{15}$$

$$v_{tk} \le 0, \quad \forall t,k \tag{16}$$

u_i is unrestricted in sign, $\forall i$ (17)

Where u_i and v_{tk} are the dual variables corresponded to Constraints (10) and (11), respectively. Consider $A(\sigma)$ as an assignment problem related to partial schedule σ, the optimal solution of which $Z_A(\sigma)$ is a lower bound on the total weighted tardiness of unscheduled jobs. Since $\hat{T}^t_{ik}(\sigma)$ uses more information in regard to the partial schedule, never will $\hat{T}^t_{ik}(\sigma)$ be smaller than $\hat{T}^t_{ik}$ for all unscheduled jobs and unfilled positions. Hence, $u^*_i + v^*_{tk} \le w_i \hat{T}^t_{ik} \le w_i \hat{T}^t_{ik}(\sigma)$. This implies that the optimal solution at the root node provides a feasible solution to the dual problem corresponding to $A(\sigma)$. As the dual problem is a maximization one, a feasible dual solution is a lower bound on the total weighted tardiness of unscheduled jobs provided by the following equation:

$$\sum_{i \in E(\sigma)} u^*_i + \sum_{(t,k) \in F(\sigma)} v^*_{tk} =$$

$$Z_A(\sigma) - \sum_{i \notin E(\sigma)} u^*_i - \sum_{(t,k) \notin F(\sigma)} v^*_{tk}$$

Where $E(\sigma)$ and $F(\sigma)$ denote the set of unscheduled jobs and the set of unfilled positions, respectively. Given the partial schedule σ, with an unscheduled job j added to the first available position, say position q, on machine r, the lower bound on the total weighted tardiness of all possible complete schedules will be

$$LB =$$

$$\sum_{i \notin E(\sigma)} w_i T_i(\sigma) + w_j \max\left(AP^r(\sigma) + AP^r_{l_r j} - d_j, 0\right) +$$

$$Z_{AP} - \sum_{i \in E(\sigma)} u^*_i - \sum_{(t,k) \in F(\sigma)} v^*_{tk} - u^*_j - v^*_{qr}$$

Where $T_i(\sigma)$ is tardiness of job i in partial schedule σ, $AP^r(\sigma)$ is the sum of the adjusted processing times of the jobs processed on machine r in the partial schedule σ, and l_r is the last job assigned on machine r in σ.

As the earliest possible completion time for each unscheduled job $i \in E(\sigma)$ is as follows:

$$EPC_i = \min_{1 \le k \le m} \left\{ AP^k(\sigma) + AP^k_{l_k i} \right\}$$

and the lower bound can be modified as follows:

$$LB =$$

$$\sum_{i \notin E(\sigma)} w_i T_i(\sigma) + w_j \max\left(AP^r(\sigma) + AP^r_{l_r j} - d_j, 0\right) +$$

$$\max\left\{ Z_{AP} - \sum_{i \in E(\sigma)} u^*_i - \sum_{(t,k) \in F(\sigma)} v^*_{tk} - u^*_j - v^*_{qr}, \; \sum_{i \in E(\sigma)} w_i \max\{0, EPC_i - d_i\} \right\}$$

5. UPPER BOUND

As a good upper bound has a direct impact on the performance of the B and B algorithm, a two-phase procedure is designed to compute the upper bound. In the first phase, we generalize the composite dispatching rule, namely apparent tardiness cost and setup (ATCS), proposed by Lee, et al [10], while in the second phase we improve the schedule obtained in the first phase by a local improvement procedure. The steps of the upper bounding scheme are summarized as follows:

Step 1. Consider U as the set of unscheduled job, and AP^k as the sum of adjusted processing times of the jobs that have already been processed on machine k. Set the initial value of U = {1, 2,...,n} and $AP^k = 0$ for k=1, 2, ..., m.

Step 2. Specify the first available machine; that is $AP^{k^*} = \min_{1 \le k \le m} \{AP^k\}$

Step 3. Determine the unscheduled job j^* such that $I_{j^*k^*} = \max_{j \in U} \{I_{jk^*}\}$

$$I_{jk^*} =$$

$$\frac{w_j}{P_{jk^*}} \exp\left(\frac{\max\left\{0, d_j - AP^{k^*} - P_{jk^*}\right\}}{k_1 \overline{P}_{k^*}} \right) \exp\left(\frac{S_{l_{k^*} j}}{k_2 \overline{S}} \right)$$

l_{k^*} The last job scheduled on machine k^*

$\bar{P}_{k^*} = \sum_{i=1}^{n} \frac{P_{jk^*}}{n}$ The average job processing times of machine k^*.

$\bar{S}$ The average of all setup time combinations.

Step 4. Schedule job j^* in the first available position on machine k^* and update

$$AP^{k^*} = AP^{k^*} + AP^{k}_{l_k j}$$

Step 5. If all jobs are scheduled, go to Step 6; otherwise, go back to Step 2.

Step 6. Improve the given schedule on each machine applying adjacent pair-wise interchanges.

K_1 and k_2 are look-ahead parameters. We use the equations suggested by Lee, et al [10] to quantify them.

6. B AND B ALGORITHM

The implementation of the proposed B and B algorithm to minimize the total weighted tardiness is elaborated in this section. The main procedures through which the B and B algorithm should be designed are the choice of a lower bound, the utilization of an initial solution as an upper bound, the incorporation of dominance rules, and the specification of branching mechanism (Baker [17]).

In the search tree, each node, say a node in the i-th level corresponds to a partial schedule in which i jobs are scheduled. In most B and B algorithm for parallel machine, each parent node can be partitioned into $m(n\text{-}i)$ chilled nodes, each of which includes the parent node plus the assignment of an unscheduled job to the first available position on one machine. To avoid generating redundant schedules, we use the branching scheme devised by Shim, et al [6]. In this branching scheme, only the child nodes associated with machines, the indices of which are not less than the machine index associated with parent node are generated from each parent node. This scheme gives rise to substantial reduction in the number of nodes generated in the branch-and-bound algorithm.

Before the execution of the B and B algorithm, the number of jobs to be considered should be reduced using Proposition 2. The job which satisfies the condition of this proposition can be scheduled last in an optimal solution. Whenever one job is eliminated, the value of A_k is updated for all machines and the condition is again checked. Then, we utilize an initial solution as an upper bound generalizing the composite dispatching rule (i.e., ATCS) at the root node of the search tree. Whenever a complete schedule is found, the upper bound is updated providing that the last complete schedule is better than the initial solution. To select a node to generate branches from, the best-first branching rule is applied. The best-first branching rule leads the algorithm to branch the active nodes with the smallest lower bound. When the child nodes are generated, the dominance propositions are employed to check whether the schedule associated with the child nodes are dominated or not. Not only the nodes corresponding to the dominates schedules, but the nodes whose lower bound is greater than or equal to upper bound also are pruned. By using dominance propositions, either full pruning or partial pruning can be occurred. In full pruning, all children of the parent node are to be pruned, while in partial pruning only the children of the parent node associated with the machines whose indices are greater than the index of machine associated with the parent node are to be pruned. Full pruning results from using Proposition 1, while Propositions 3 and 4 lead to partial pruning.

7. COMPUTATIONAL EXPERIMENTS

The branch-and-bound algorithm has been compiled in Matlab 2006a. The performance of the proposed algorithm is reported on randomly generated problems. Generally 144 test problems are generated, one instance of each combination of six levels for the number of jobs (n=6, 8, 10, 14, 18, 20), two levels for the number of machines (m=2, 4), two levels for the tardiness factor TF = 0.2, 0.9, three levels for the due date ranges (DDR=0.2, 0.6,1), and two levels for the setup times. Processing times are randomly chosen from the uniform distribution between 5 and 200. For the setup times, two categories are defined. In the first category setup times are small and generated from U[25,50], while in the second category setup times are large and generated from U[25,150]. Due dates are randomly

generated from a uniform distribution whose mean is set to $P_m \times N \times (1\text{-}TF)$ and whose range is set to $DDR \times N \times P_m$, where P_m is the mean processing times. The overall performance of our proposed B and B algorithm is illustrated in Table 1.

By considering six levels for the number of jobs and two levels for the number of machines, twelve different size groups are produced. In each group, twelve test problems characterized with different tardiness factor, due date range, and setup times exist. The result in Table 1 is concerned just with different size. As can be seen, the percentage of problems solved just using Proposition 2 is so considerable for all job sizes, and the average percentage of problem solved using this proposition has been increased with the increasing the number of machines. As it has been mentioned before, Preposition 2 can schedule the jobs satisfying the condition before starting the B and B algorithm. Therefore, the problems solved just using this preposition have not undergone the B and B algorithm. When the number of jobs and the number of machines is increased, test problems either cannot be solved or are solved just using Proposition 2. In this case, the average times required for solving the problems is decreased as this average is calculated only for the jobs solved with 15 minutes. So the small amount of the average processing time reported in Table 1 for large-sized problems is because of solving most problems just using Proposition 2. Table 2 shows the percentage of the solved problems for two various categories defined for the setup times. It can be seen that the value of setup times does not have a significant effect on the percentage of solved problems.

With respect to the results in Table 1, we find out when the tardiness factor is set to the largest value (i.e. 0.9), and the due date ranges is set to the smallest value (i.e. 0.2), most test problems cannot be solved. Since the levels which have been chosen for the tardiness factor and due date ranges often leads to test problems with tight due dates especially for large-sized problems, we generate 36 new test problems, three instances of each combination of six levels for the number of jobs (n=6, 8, 10, 14, 18, 20), two levels for the number of machines (m=2, 4), one level for the tardiness factor TF = 0.7, one level for the due date ranges (DDR=0.3), and one level for the small setup times. The detailed results are provided in Table 3.

The proposed B and B algorithm performs well with up to four machines and 10 jobs. However, it seems that the performance of the algorithm is greatly affected by the input values of the given problems. For example, when the number of jobs is equal to 8 and the number of machines equals to 4, the gap between execution times is so large. It is clear that for obtaining more solid results more numerical experiments should be carried out. It seems that the uniform distribution proposed for generating due dates is not appropriate. In future research, we are going to complete the computational results and clarify the influence of each factor on the performance of the B and B algorithm.

8. CONCLUSION

This paper has developed a new branch-and-bound (B and B) algorithm for unrelated parallel machine scheduling problems with sequence-dependent setup time. Several dominance properties have been provided. The lower bound has been derived through assignment problem. The upper bound has been resulted from generalizing the composite dispatching rule, namely ATCS. Our proposed B and B algorithm has incorporated all these procedures to minimize the total weighted tardiness. We have observed that Proposition 2 often leads to the final solution with increasing the size of problems, and also the suggested algorithm gives the optimal solution for problems with up to 10 jobs and 4 machines in all combination of tardiness factor and due date ranges. In following we aim to carry out more numerical experiments to achieve solid results.

9. ACKNOWLGEMENT

This study was partially supported by the University of Tehran under the research grant No. 8106043/1/11. The first author is grateful for this financial support.

10. APPENDIX

The result of Table 1 is the summarization of data provided in Table 4 to 9.

TABLE 1. The Overall Performance of the Algorithm.

Size	Time (s.)	PP2	Solved
(2,6)	5.58	50	100
(2,8)	35.79	33.33	83.33
(2,10)	2.22	41.67	75
(2,14)	2.26	41.67	83.33
(2,18)	4.02	33.33	66.67
(2,20)	3.96	66.67	83.33
(4,6)	68.01	83.33	100
(4,8)	3.75	50	91.67
(4,10)	2.28	58.33	91.67
(4,14)	2.27	75	91.67
(4,18)	3.93	75	83.33
(4,20)	4.16	66.67	83.33

PP2: Denotes the percentage of problems solved completely just using proposition 2.

TABLE 2. Influence of Setup Times on the Percentage of the Solved Problems.

No. of Jobs	Setup	Solved
6	small	100.00
	Large	100.00
8	small	100.00
	Large	83.33
10	small	83.33
	Large	83.33
14	small	83.33
	Large	91.67
18	small	83.33
	Large	66.67
20	small	83.33
	Large	83.33

TABLE 3. The Performance of the Algorithm at *TF*=0.3 and *DDR*=0.7.

No. of Jobs	No. of Machines	Time (s.)	NS
6	2.00	4.02	12.00
	2.00	7.40	347.00
	2.00	7.11	313.00
	4.00	16.44	950.00
	4.00	4.00	24.00
	4.00	20.27	1304.00
8	2.00	18.33	1318.00
	2.00	441.09	14572.00
	2.00	257.83	9458.00
	4.00	554.97	12234.00
	4.00	3600.00	31341.00
	4.00	4.07	32.00
10	2.00	34.10	3646.00
	2.00	1419.48	46571.00
	2.00	902.36	32173.00
	4.00	Not Solved	
	4.00	2.24	0.00
	4.00	3600.00	38885.00
14	2.00	Not Solved	
	2.00	Not Solved	
	2.00	Not Solved	
	4.00	Not Solved	
	4.00	5400.00	59380.00
	4.00	5400.00	56881.00
18	2.00	Not Solved	
	2.00	Not Solved	
	2.00	Not Solved	
	4.00	Not Solved	
	4.00	Not Solved	
	4.00	Not Solved	
20	2.00	Not Solved	
	2.00	Not Solved	
	2.00	Not Solved	
	4.00	Not Solved	
	4.00	Not Solved	
	4.00	Not Solved	

NS: Denotes the number of sub problems considered for the problem.

TABLE 4. The Performance of the Algorithm at *N*=6.

No. Job	No. Machine	Time (s.)	No. Nodes	DDR	TF	Setup
6	2	2.29	13	0.2	0.2	Large
6	2	3.84	0	0.6	0.9	Large
6	2	3.78	4	1	0.2	Large
6	2	14.83	368	0.2	0.9	Large
6	2	3.88	4	0.6	0.2	Large
6	2	3.61	0	1	0.9	Large
6	2	3.75	12	0.2	0.2	Small
6	2	3.60	0	0.6	0.9	Small
6	2	3.71	0	1	0.2	Small
6	2	16.18	1092	0.2	0.9	Small
6	2	3.72	0	0.6	0.2	Small
6	2	3.76	0	1	0.9	Small
6	4	3.61	0	0.2	0.2	Large
6	4	3.62	0	0.6	0.9	Large
6	4	3.64	0	1	0.2	Large
6	4	775.65	13148	0.2	0.9	Large
6	4	3.70	0	0.6	0.2	Large
6	4	3.70	0	1	0.9	Large
6	4	3.69	0	0.2	0.2	Small
6	4	3.68	0	0.6	0.9	Small
6	4	3.58	0	1	0.2	Small
6	4	3.84	16	0.2	0.9	Small
6	4	3.72	0	0.6	0.2	Small
6	4	3.68	0	1	0.9	Small

TABLE 5. The Performance of the Algorithm at *N*=8.

No. Job	No. Machine	Time (s.)	No .Nodes	DDR	TF	Setup
8	2	2.31	16	0.2	0.2	Large
8	2	3600.00	34318	0.6	0.9	Large
8	2	3.69	0	1	0.2	Large
8	2	294.08	10176	0.2	0.9	Large
8	2	3.81	14	0.6	0.2	Large
8	2	3.68	6	1	0.9	Large
8	2	4.09	16	0.2	0.2	Small
8	2	35.29	2034	0.6	0.9	Small
8	2	3.64	0	1	0.2	Small
8	2	977.54	18667	0.2	0.9	Small
8	2	3.73	0	0.6	0.2	Small
8	2	3.60	0	1	0.9	Small
8	4	3.84	0	0.2	0.2	Large
8	4	3.88	4	0.6	0.9	Large
8	4	3.64	0	1	0.2	Large
8	4	3600.00	30484	0.2	0.9	Large
8	4	3.88	12	0.6	0.2	Large
8	4	3.80	4	1	0.9	Large
8	4	3.64	0	0.2	0.2	Small
8	4	3.71	8	0.6	0.9	Small
8	4	3.67	0	1	0.2	Small
8	4	3.97	32	0.2	0.9	Small
8	4	3.61	0	0.6	0.2	Small
8	4	3.64	0	1	0.9	Small

TABLE 6. The Performance of the Algorithm at *N*=10.

No. Job	No. Machine	Time (s.)	No. Nodes	DDR	TF	Setup
10	2	3600.00	41432	0.2	0.2	Large
10	2	2.30	0	0.6	0.9	Large
10	2	2.28	0	1	0.2	Large
10	2	3600.00	39161	0.2	0.9	Large
10	2	2.23	20	0.6	0.2	Large
10	2	2.29	11	1	0.9	Large
10	2	2.25	20	0.2	0.2	Small
10	2	2.21	18	0.6	0.9	Small
10	2	2.16	0	1	0.2	Small
10	2	3600.00	37609	0.2	0.9	Small
10	2	2.10	0	0.6	0.2	Small
10	2	2.10	0	1	0.9	Small
10	4	2.24	0	0.2	0.2	Large
10	4	2.31	16	0.6	0.9	Large
10	4	2.27	0	1	0.2	Large
10	4	2.37	40	0.2	0.9	Large
10	4	2.33	0	0.6	0.2	Large
10	4	2.24	0	1	0.9	Large
10	4	2.26	0	0.2	0.2	Small
10	4	2.31	0	0.6	0.9	Small
10	4	2.23	0	1	0.2	Small
10	4	3600.00	43872	0.2	0.9	Small
10	4	2.21	20	0.6	0.2	Small
10	4	2.27	16	1	0.9	Small

TABLE 7. The Performance of the Algorithm at *N*=14.

No. Job	No. Machine	Time (s.)	No. Nodes	DDR	TF	Setup
14	2	2.51	28	0.2	0.2	Large
14	2	2.21	0	0.6	0.9	Large
14	2	2.12	0	1	0.2	Large
14	2	3600.00	48741	0.2	0.9	Large
14	2	2.27	16	0.6	0.2	Large
14	2	2.29	6	1	0.9	Large
14	2	2.43	28	0.2	0.2	Small
14	2	2.10	0	0.6	0.9	Small
14	2	2.18	0	1	0.2	Small
14	2	3600.00	45033	0.2	0.9	Small
14	2	2.07	0	0.6	0.2	Small
14	2	2.39	4	1	0.9	Small
14	4	2.26	0	0.2	0.2	Large
14	4	2.24	0	0.6	0.9	Large
14	4	2.19	0	1	0.2	Large
14	4	2.73	56	0.2	0.9	Large
14	4	2.16	0	0.6	0.2	Large
14	4	2.23	0	1	0.9	Large
14	4	2.26	0	0.2	0.2	Small
14	4	2.29	0	0.6	0.9	Small
14	4	2.11	0	1	0.2	Small
14	4	3600.00	53577	0.2	0.9	Small
14	4	2.23	4	0.6	0.2	Small
14	4	2.31	0	1	0.9	Small

TABLE 8. The Performance of the Algorithm at *N*=18.

No. Job	No. Machine	Time (s.)	No. Nodes	DDR	TF	Setup
18	2	3600.00	35634	0.2	0.2	Large
18	2	4.05	28	0.6	0.9	Large
18	2	4.18	0	1	0.2	Large
18	2	3600.00	35804	0.2	0.9	Large
18	2	3.93	32	0.6	0.2	Large
18	2	3600.00	27686	1	0.9	Large
18	2	4.73	34	0.2	0.2	Small
18	2	3.90	28	0.6	0.9	Small
18	2	4.08	0	1	0.2	Small
18	2	3600.00	35600	0.2	0.9	Small
18	2	3.63	0	0.6	0.2	Small
18	2	3.70	0	1	0.9	Small
18	4	3.92	0	0.2	0.2	Large
18	4	3.83	0	0.6	0.9	Large
18	4	3.64	0	1	0.2	Large
18	4	3600.00	43796	0.2	0.9	Large
18	4	4.01	0	0.6	0.2	Large
18	4	3.83	0	1	0.9	Large
18	4	4.10	0	0.2	0.2	Small
18	4	3.98	0	0.6	0.9	Small
18	4	4.34	12	1	0.2	Small
18	4	3600.00	43592	0.2	0.9	Small
18	4	3.73	0	0.6	0.2	Small
18	4	3.90	0	1	0.9	Small

TABLE 9. The Performance of the Algorithm at *N*=20.

No. Job	No. Machine	Time (s.)	No. Nodes	DDR	TF	Setup
20	2	4.56	40	0.2	0.2	Large
20	2	3.78	0	0.6	0.9	Large
20	2	3.96	0	1	0.2	Large
20	2	3600.00	36710	0.2	0.9	Large
20	2	4.57	32	0.6	0.2	Large
20	2	3.86	0	1	0.9	Large
20	2	3.72	0	0.2	0.2	Small
20	2	3.86	0	0.6	0.9	Small
20	2	3.84	0	1	0.2	Small
20	2	3600.00	36710	0.2	0.9	Small
20	2	3.74	0	0.6	0.2	Small
20	2	3.72	0	1	0.9	Small
20	4	5.98	0	0.2	0.2	Large
20	4	3.84	0	0.6	0.9	Large
20	4	3.87	0	1	0.2	Large
20	4	3600.00	45224	0.2	0.9	Large
20	4	4.10	4	0.6	0.2	Large
20	4	4.07	16	1	0.9	Large
20	4	3.85	0	0.2	0.2	Small
20	4	4.10	0	0.6	0.9	Small
20	4	3.76	0	1	0.2	Small
20	4	3600.00	44768	0.2	0.9	Small
20	4	4.07	0	0.6	0.2	Small
20	4	3.93	0	1	0.9	Small

11. REFERENCES

1. Dessouk, M.M., "Scheduling Identical Jobs with Unequal Ready Times on Uniform Parallel Machines to Minimize the Maximum Lateness", ***Computers and Industrial Engineering***, Vol. 34, (1998), 793-806.
2. Azizoglu, M. and Kirca, O., "Scheduling Jobs on Unrelated Machines to Minimize Regular Total Cost Function", ***IIE Transaction***, Vol. 31, (1999), 153-159.
3. Liaw, C.F., Lin, Y.K., Cheng, C.Y. and Chen, M., "Scheduling Unrelated Parallel Machines to Minimize Total Weighted Tardiness", ***Computers and Operations Research***, Vol. 30, (2003), 1777-1789.
4. Rabadi, G., Mollaghasemi, M. and Anagnostopoulos, G.C., "A Branch-and-Bound Algorithm for the Early/Tardy Machine Scheduling Problem with a Common Due-Date and Sequence-Dependent Setup Time", ***Computers and Operations Research***, Vol. 31, (2004), 1727-1751.
5. Luo, X. and Chu, F., "A Branch And Bound Algorithm Of The Single Machine Schedule With Sequence Dependent Setup Times for Minimizing Total Tardiness", ***Applied Mathematics and Computation***, Vol. 183, (2006), 575-588.
6. Shim, S.O. and Kim, Y.D., "Minimizing Total Tardiness in an Unrelated Parallel Machine Scheduling Problem", ***Journal of the Operational Research Society***, Vol. 58, (2007), 346-354.
7. Pfund, M., Fowler, J.W., Gadkari, A. and Chen, Y., "Scheduling Jobs on Parallel Machines with Setup Times and Ready Times", ***Computers and Industrial Engineering***, Vol. 54, (2008), 764-782.
8. Lee, Y.H. and Pinedo, M., "Scheduling Jobs on Parallel Machines with Sequence Dependent Setup Times", ***European Journal of Operational Research***, Vol. 100, (1997), 464-474.
9. Park, Y., Kim, S. and Lee, Y.H., "Scheduling Jobs on Parallel Machines Applying Neural Network and Heuristic Rules", ***Computers and Industrial Engineering***, Vol. 38, (2000), 189-202.
10. Lee, Y.H., Bhaskaran, K. and Pinedo, M., "A Heuristic to Minimize the Total Weighted Tardiness with Sequence-Dependent Setups", ***IIE Transactions***, Vol. 29, (1997), 45-52.
11. Logendran, R., McDonell, B. and Smucker, B., "Scheduling Unrelated Parallel Machines with Sequence-Dependent Setups", ***Computers and Operations Research***, Vol. 34, (2007), 3420-3438.
12. Anghinolfi, D. and Paolucci, M., "Parallel machine total Tardiness Scheduling with a New Hybrid Meta-Heuristic Approach", ***Computers and Operations Research***, Vol. 34, (2007), 3471-3490.
13. Tavakkoli-Moghaddam, R., Taheri, F. and Bazzazi, M., "Multi-Objective Unrelated Parallel Machines Scheduling with Sequence-Dependent Setup Times and Precedence Constraints", ***IJE Transactions A: Basics***, Vol. 21, No. 3, (September 2008), 269-278.
14. Tavakkoli-Moghaddam, R. and Mehdizadeh, E., "A New ILP Model for Identical Parallel-Machine Scheduling with Family Setup Times Minimizing the Total Weighted Flow Time by a Genetic Algorithm", ***IJE Transactions A: Basics***, Vol. 20, No. 2, (June 2007) 183-194.
15. Nessah, R., Chu, C. and Yalaoui, F., "An Exact Algorithm for $Pm/sds,r_i/\sum_{i=1}^{n} C_i$ Problem", ***Computers and Operations Research***, Vol. 34, (2007), 2840-2848.
16. Rocha, P.L., Ravetti, M.G., Mateus, G.R. and Pardalos, P.M., "Exact Algorithms for A Scheduling Problem with Unrelated Parallel Machines and Sequence-Dependent Setup Times", ***Computers and Operations Research***, Vol. 35, (2008), 1250-1264.
17. Baker, K.R., "Introduction to Sequence and Scheduling", John Wiley and Sons, New York, U.S.A., (1974).