# A BRANCH-AND-BOUND ALGORITHM WITH LAGRANGIAN DECOMPOSITION FOR PARALLEL MACHINE SCHEDULING

**Shunji Tanaka** *, **Mituhiko Araki** *

* *Graduate School of Electrical Engineering, Kyoto University*
*Kyotodaigaku-Katsura, Nishikyo-ku, Kyoto 615–8510 JAPAN*

Abstract: The purpose of this study is to propose a new branch-and-bound algorithm for a class of scheduling problems to minimize total tardiness on identical parallel machines. In this algorithm, Lagrangian decomposition is applied to obtain better lower bounds. In addition, the job dominance conditions for the single machine tardiness problem are utilized for both restricting branches and improving the lower bounds. As will be shown by computational experiments, instances with up to 25 jobs and any number of machines are optimally solved within acceptable computational times. *Copyright © 2005 IFAC*.

Keywords: Manufacturing, Scheduling algorithms, optimization, relaxation, decomposition.

## 1. INTRODUCTION

In this study a class of scheduling problems to minimize total tardiness on identical parallel machines ($P||\sum T_j$) is considered. The single machine total tardiness problem ($1||\sum T_j$) has been extensively studied so far. Emmons (1969) first showed that some precedence relations of jobs hold in an optimal schedule for $1||\sum T_j$, which are referred to as Emmons' dominance conditions. Lawler (1977) proposed a pseudopolynomial algorithm to solve $1||\sum T_j$ based on his theorem called Lawler's decomposition theorem. After the Lawler's research, solution algorithms have been improved by several researchers, and Szwarc *et al.* (1999) reported that their algorithm can handle instances with up to 300 jobs.

As for $P||\sum T_j$, strict solution algorithms have been proposed (Root, 1965; Elmaghraby and Park, 1974; Barnes and Brennan, 1977; Azizoglu and Kirca, 1998; Liaw *et al.*, 2003; Tanaka and Araki, 2004), but only the last three can be applied to general $P||\sum T_j$ (Liaw *et al.* treated the more general problem $R||\sum w_j T_j$). Moreover, it is difficult to handle larger problem instances by these algorithms, although instances with up to 20 jobs and 3 machines were optimally solved (Tanaka and Araki, 2004).

In this study the result by Tanaka and Araki (2004) is improved so that larger problem instances can be solved. The primary improvement is the introduction of Lagrangian decomposition for lower bound calculation in the branch-and-bound algorithm. Since it is known that Lagrangian decomposition yields good lower bounds for scheduling problems (e.g. Luh *et al.* (1990)), it can be expected that the efficiency of the branch-and-bound algorithm improves. Indeed, Babu *et al.* (2004) successfully applied Lagrangian decomposition to lower bound calculation in the branch-and-bound algorithm for the single machine weighted tardiness problem ($1||\sum w_j T_j$). Another improvement of the proposed algorithm is the utilization of the Emmons' dominance conditions for both restricting branches and improving lower bounds. These improvements enable us to handle instances with up to 25 jobs and any number of machines.

## 2. THE TOTAL TARDINESS PROBLEM ON IDENTICAL PARALLEL MACHINES

Consider that a set of $n$ jobs $\mathcal{J} = \{\mathrm{J}_1, \ldots, \mathrm{J}_n\}$ are to be processed on $m$ identical parallel machines $\{\mathrm{M}_1, \ldots, \mathrm{M}_m\}$. Each job $\mathrm{J}_j$ is given the integer processing time $p_j$ and the integer duedate $d_j$. All the jobs are available at time zero, and no job preemption is allowed. The tardiness $T_j$ of $\mathrm{J}_j$ is given by $T_j = \max(C_j - d_j, 0)$, where $C_j$ is the completion time of $\mathrm{J}_j$. The objective here is to search an optimal schedule that minimizes the total tardiness $\sum_{j=1}^{n} T_j$. This problem is referred to as $P||\sum T_j$ according to the standard classification of scheduling problems.

## 3. LAGRANGIAN DECOMPOSITION OF $P||\sum T_J$

Lagrangian decomposition is to decompose an original problem into relatively easy subproblems by relaxing "coupling" constraints via Lagrangian multipliers. In this section, Lagrangian decomposition is applied to $P||\sum T_j$ according to Luh *et al.* (1990).

The problem $P||\sum T_j$ can be formulated by the following binary integer programming.

$$(\mathrm{P}): F = \min_{x} \sum_{j=1}^{n} \sum_{t=0}^{S_j} W_{jt} x_{jt}, \tag{1}$$

subject to

$$x_{jt} \in \{0, 1\} \quad (1 \le j \le n,\ 0 \le t \le S_j), \tag{2}$$

$$\sum_{t=0}^{S_j} x_{jt} = 1 \quad (1 \le j \le n), \tag{3}$$

$$\sum_{j=1}^{n} \sum_{s=\max(t-p_j+1,0)}^{\min(t,S_j)} x_{js} \le M_t \quad (0 \le t \le S_{\max}). \tag{4}$$

Here, $x_{jt}$ are decision variables such that

$$x_{jt} = \begin{cases} 1 & \text{If } \mathrm{J}_j \text{ is started at } t, \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

and the constants $S_j$, $W_{jt}$ and $M_t$ are given by

$S_j$: The latest possible starting time of $\mathrm{J}_j$ ($S_j := (\sum_{i=1}^{n} p_i - p_j)/m$),
$S_{\max}$: $S_{\max} = \max_{1 \le j \le n}(S_j + p_j - 1)$,
$W_{jt}$: The tardiness of $\mathrm{J}_j$ when started from $t$ ($W_{jt} = \max(t + p_j - d_j, 0)$),
$M_t$: The number of machines available at $t$ ($M_t = m$).

The machine resource constraints (4) in (P) are relaxed by introducing nonnegative Lagrangian multipliers $\mu_t$ ($0 \le t \le S_{\max}$). Then, the relaxed problem becomes:

$$(\mathrm{R}): F^*(\mu) = \min_{x} F(\mu), \tag{6}$$

$$\text{subject to (2) and (3)}, \tag{7}$$

where

$$\begin{aligned} F(\mu) &= \sum_{j=1}^{n} \sum_{t=0}^{S_j} W_{jt} x_{jt} \\ &\quad - \sum_{t=0}^{S_{\max}} \mu_t \left( M_t - \sum_{j=1}^{n} \sum_{s=\max(t-p_j+1,0)}^{\min(t,S_j)} x_{js} \right) \\ &= - \sum_{t=0}^{S_{\max}} \mu_t M_t + \sum_{j=1}^{n} F_j(\mu), \\ F_j(\mu) &= \sum_{t=0}^{S_j} W_{jt} x_{jt} + \sum_{t=0}^{S_{\max}} \mu_t \sum_{s=\max(t-p_j+1,0)}^{\min(t,S_j)} x_{js} \\ &= \sum_{t=0}^{S_j} (W_{jt} + \sum_{s=t}^{s=t+p_j-1} \mu_s) x_{jt}. \end{aligned} \tag{8}$$

The dual (D) of (R) is given by

$$\begin{aligned} (\mathrm{D}): \underline{F} &= \max_{\mu} F^*(\mu) \\ &= \max_{\mu} \left( - \sum_{t=0}^{S_{\max}} \mu_t M_t + \min_{x} \sum_{j=1}^{n} F_j(\mu) \right), \end{aligned} \tag{9}$$

$$\text{subject to (2) and (3)}. \tag{10}$$

The minimization in (9) can be performed separately with regard to $j$, and the following trivial subproblems corresponding to $\mathrm{J}_j$ ($1 \le j \le n$) are obtained by noting the constraints (2) and (3).

$$(\mathrm{R}_j): F_j^*(\mu) = \min_{0 \le t_j \le S_j} (W_{jt_j} + \sum_{s=t_j}^{s=t_j+p_j-1} \mu_s). \tag{11}$$

Therefore, $F^*(\mu)$ for a fixed set of Lagrangian multipliers can be easily calculated. To solve (D), subgradient optimization is applied as many other researches. More specifically, the set of the multipliers at $(n+1)$th iteration, $\mu^{n+1}$, is determined from $\mu^n$ as follows:

$$\mu^{n+1} = \mu^n + \alpha^n g(\mu^n) \tag{12}$$

where $g(\mu)$ is a subgradient of $F^*(\mu)$ with respect to $\mu$ and $\alpha^n$ is the $n$th step size. The step size is chosen as

$$\alpha^n = \lambda \frac{\overline{F} - F^*(\mu^n)}{g(\mu^n)^{\mathrm{T}} g(\mu^n)}. \tag{13}$$

Here, $\overline{F}$ is an upper bound of $F$ and $\lambda$ is the step size parameter satisfying $0 < \lambda < 2$.

It is known that tight lower bounds are obtained for parallel machine scheduling problems by Lagrangian

decomposition (e.g. Luh *et al.* (1990)). This lower bounds are utilized in the proposed branch-and-bound algorithm. In the next section, the outline of the algorithm will be shown.

## 4. THE OUTLINE OF THE BRANCH-AND-BOUND ALGORITHM

Since the total tardiness is a nondecreasing function of job completion times, there exists an optimal schedule in that no idle times are inserted between jobs. It follows that an optimal schedule can be constructed by assigning jobs on the earliest available machines one by one according to a priority list if it is appropriately chosen. Therefore, an optimal priority list is searched by a depth-first branch-and-bound algorithm as in the previous researches (Azizoglu and Kirca, 1998; Tanaka and Araki, 2004). In the following, a procedure to assign jobs on the earliest available machines according to a priority list $P_{\mathrm{r}}$ is denoted by $\mathcal{D}(P_{\mathrm{r}})$.

### *4.1 Initial upper and lower bounds*

First, how to calculate the initial upper and lower bounds at the root node of the branch-and-bound algorithm will be stated. As explained in the preceding section, Lagrangian decomposition gives tight lower bounds for $P||\sum T_j$. Indeed, there are no duality gaps for most problem instances as will be shown in Section 5. Therefore, a good feasible schedule (an upper bound) and a lower bound are searched at the root node so that an optimal schedule could be obtained without branching.

To search good upper and lower bounds, the following procedure is applied.

$0^\circ$ If

$$d_j \geq \frac{1}{m}\sum_{i=1}^{n} p_i + \frac{m-1}{m} p_j, \tag{14}$$

is satisfied, $\mathrm{J}_j$ is always non-tardy in any schedule constructed by $\mathcal{D}(\cdot)$ (Tanaka and Araki, 2004). Therefore, such non-tardy jobs are removed to reduce the size of the problem. Redefine $\mathcal{J}$ by the set of the remaining jobs and $n$ by the number of the remaining jobs.

$1^\circ$ Construct a schedule by $\mathcal{D}(P_{\mathrm{r}}^{\mathrm{SPT}}(\mathcal{J}))$. Here, $P_{\mathrm{r}}^{\mathrm{SPT}}(\mathcal{J})$ denotes the priority list of the jobs belonging to $\mathcal{J}$ sequenced in SPT (Shortest-Processing-Time-first) order. If all the jobs are tardy in this schedule, the procedure is terminated because it is optimal (Koulamas, 1997). Otherwise, a better schedule is searched from this schedule by a local search explained later. Denote the total tardiness of the obtained schedule by $\overline{F}^{\mathrm{SPT}}$.

$2^\circ$ Construct a schedule by the KPM heuristics (Koulamas, 1994). Then, the local search is applied. Denote the total tardiness of the obtained schedule by $\overline{F}^{\mathrm{KPM}}$.

$3^\circ$ Let $\overline{F} := \min(\overline{F}^{\mathrm{SPT}}, \overline{F}^{\mathrm{KPM}})$. Apply Lagrangian decomposition and maximize $F^*(\mu)$ by the subgradient optimization as explained in Section 3. The step size parameter $\lambda$ in (13) is set to be $\lambda := 2.0$ initially. At each step of the subgradient optimization, a schedule is constructed from a solution corresponding to $F^*(\mu^n)$ by resolving conflicts heuristically. The local search is applied to this schedule and $\overline{F}$ is updated if necessary.

If $F^*(\mu^n)$ is not updated for $50$ successive iterations, $\lambda$ is scaled by $\lambda := 0.8\lambda$. The subgradient optimization is terminated if $F^*(\mu^n) = \overline{F}$ or if $F^*(\mu^n)$ is not updated for $100$ successive iterations, or if $\lambda < 10^{-4}$. Denote the obtained multipliers by $\mu^*$ and let $\underline{F} := F^*(\mu^*)$.

Although there are several types of simple heuristics proposed for $P||\sum T_j$ (Wilkerson and Irwin, 1971; Ho and Chang, 1991; Koulamas, 1994), the KPM heuristics seems to be the best among them according to Koulamas (1994). It motivates us to adopt the KPM heuristics to generate an initial upper bound. It can be also expected that $\mathcal{D}(P_{\mathrm{r}}^{\mathrm{SPT}}(\mathcal{J}))$ generates a good schedule since the schedule is optimal if all the jobs are tardy in that schedule as mentioned in $1^\circ$.

However, these two schedules themselves are not good enough to be adopted as an initial upper bound. Thus, a simple local search is applied to improve these schedules. The neighborhood structure of this search is given by

Insertion: A job is moved into another position, on the same machine or on another machine.
Exchange: Two jobs on the same machine or on different machines are exchanged.

If the schedule cannot be improved by either the insertion nor the exchange, the local search is terminated.

Next, by using the obtained upper bound $\overline{F}$, the subgradient optimization is applied to the Lagrangian dual (D). As suggested by Fisher (1981), the step size parameter $\lambda$ is reduced when the solution is not updated. The number of iterations for reducing $\lambda$ and the number of iterations for terminating the subgradient optimization are determined by preliminary computational experiments.

At each step of the subgradient optimization, $F^*(\mu^n)$, a solution of (R) is obtained. Since (R) is a relaxed problem of (P), it is, in general, not feasible for (P). However, it can be used for constructing a feasible solution of (P) heuristically. The heuristics in this study is a modified version of the one proposed by Luh *et al.* (1990). Let us denote by $t_j^n$ the solution (the starting time of $\mathrm{J}_j$) that minimizes $F_j(\mu^n)$ in the decomposed subproblem ($\mathrm{R}_j$). Jobs are scheduled on the earliest available machines one by one in the

nondecreasing order of $t_j^n$ if no conflict occurs ($t_j^n$ is not lesser than the earliest machine release time). If some conflict occurs, a job is chosen from amongst the conflicting jobs by the following rule.

(a) If there exists at least one conflicting job that is tardy when scheduled on the earliest available machine, a tardy job that can be completed as close as possible to its duedate is chosen.
(b) If every conflicting job is non-tardy when scheduled on the earliest available machine, a job that can be completed as early as possible is chosen.

The local search is also applied to the schedule obtained by this heuristics. Then, the upper bound $\overline{F}$ is updated if necessary.

### *4.2 Branching*

Branching is performed by fixing the elements of the priority list from the first to the last. Thus, a subproblem corresponding to a node at depth $l$ is to determine the last $(n-l)$ elements of the priority list.

Branches are restricted by introducing dominance conditions. First one is the simple global dominance condition given by Azizoglu and Kirca (1998).

**Branch restriction via the global dominance condition.**

If two jobs $\mathrm{J}_i$ and $\mathrm{J}_j$ satisfy $p_i = p_j$ and $d_i < d_j$, $\mathrm{J}_i$ should precede $\mathrm{J}_j$ in an optimal priority list.

Next, the Emmons' dominance conditions (Emmons, 1969) are utilized as Tanaka and Araki (2004). Let us denote by $L^l$ a partial priority list of length $l$ corresponding to a node at depth $l$, and by $S^l$ the partial schedule constructed by $\mathcal{D}(L^l)$. The set of the $(n-l)$ unscheduled jobs is denoted by $\mathcal{U}^l$. Let us assume that the processing order of the jobs on $\mathrm{M}_k$ $(1 \le k \le m)$ in the partial schedule $S^l$ is given by $\mathrm{J}_{r_{k1}}, \ldots, \mathrm{J}_{r_{k,v_k^l}}$, and denote the total processing time on $\mathrm{M}_k$ in $S^l$ by

$$c_k^l = \sum_{j=1}^{v_k^l} p_{r_k,j}. \tag{15}$$

Let us further define $m^l = \arg\min_k c_k^l$. Since the candidates for the $(l+1)$th job in the priority list are to be scheduled on $\mathrm{M}_{m^l}$, they should satisfy the dominance conditions for the single machine total tardiness problem $(1||\sum T_j)$ on $\mathrm{M}_{m^l}$. Therefore, by checking the Emmons' dominance conditions, branches can be restricted as follows.

**Branch restriction via the Emmons' dominance conditions.**

If for some $j$ $(1 \le j \le v_{m^l}^l)$, $\mathrm{J}_u \in \mathcal{U}^l$ satisfies at least one of the following two conditions, $\mathrm{J}_u$ cannot be a candidate for the $(l+1)$th job in the priority list.

(1) $p_u < p_{r_{m^l,j}}$, $d_u \le \max(\sum_{i=1}^{j} p_{r_{m^l,i}},\ d_{r_{m^l,j}})$.
(2) $p_u > p_{r_{m^l,j}}$, $d_u < d_{r_{m^l,j}}$, $c_{m^l}^l + p_u < d_{r_{m^l,j}} + p_{r_{m^l,j}}$.

These two types of branch restriction are already used by Tanaka and Araki (2004). Here, another type of branch restriction is proposed. The basic idea is as follows. Since all the unscheduled jobs are to be scheduled on some machines after all, they should satisfy the dominance conditions on the machines where they are scheduled. It follows that if an unscheduled job is known to break at least one dominance condition on all the machines, it cannot be scheduled to any machines. Therefore, the candidates for the $(l+1)$th job in the priority list should be such that there is no unscheduled job that breaks a dominance condition on all the machines. It is summarized as follows.

**Another type of branch restriction via the Emmons' dominance conditions.**

Consider that $\mathrm{J}_u \in \mathcal{U}^l$ is chosen as a candidate for the $(l+1)$th job in the priority list, and that the processing order of the jobs on $\mathrm{M}_k$ $(1 \le k \le m)$ in the partial schedule $S^{l+1}$ (the partial schedule obtained by adding $\mathrm{J}_u$ to $S^l$) is given by $\mathrm{J}_{r_{k1}}, \ldots, \mathrm{J}_{r_{k,v_k^{l+1}}}$. If there exists $\mathrm{J}_w \in \mathcal{U}^l \backslash \{\mathrm{J}_u\}$ such that for all $k$ $(1 \le k \le m)$ and for some $j$ $(1 \le j \le v_k^{l+1})$, at least one of the following conditions is satisfied, $\mathrm{J}_u$ cannot be a candidate for the $(l+1)$th job in the priority list.

(1) $p_w < p_{r_{kj}}$, $d_w \le \max(\sum_{i=1}^{j} p_{r_{ki}},\ d_{r_{kj}})$.
(2) $p_w > p_{r_{kj}}$, $d_w < d_{r_{kj}}$, $S_w + p_w < d_{r_{kj}} + p_{r_{kj}}$.

Here, $S_w := (\sum_{j=1}^{n} p_j - p_w)/m$ is the latest possible starting time of $\mathrm{J}_w$.

### *4.3 Fathoming test by SPT*

If all the jobs belonging to $\mathcal{U}^l$ are tardy in the schedule constructed by applying $\mathcal{D}(P_{\mathrm{r}}^{\mathrm{SPT}}(\mathcal{U}^l))$ to $S^l$, this schedule is optimal under the condition that $S^l$ is fixed (Tanaka and Araki, 2004). Therefore, in such a case the node is fathomed and the incumbent solution is updated if necessary.

### *4.4 Lower bound calculation*

In the proposed algorithm, two types of lower bounds for the total tardiness of the unscheduled jobs $\mathcal{U}^l$ are considered. The one is based on the Lagrangian decomposition explained in Section 3. The subproblem to minimize the total tardiness of the unscheduled jobs $\mathcal{U}^l$ is formulated by

$$(\mathrm{SP}^l):\ \min \sum_{\mathrm{J}_j \in \mathcal{U}^l} \sum_{t=E_j^l}^{S_j} W_{jt} x_{jt}, \tag{16}$$

subject to

$$x_{jt} \in \{0,1\} \quad (\mathrm{J}_j \in \mathcal{U}^l,\ E_j^l \le t \le S_j), \tag{17}$$

$$\sum_{t=0}^{S_j} x_{jt} = 1 \quad (\mathrm{J}_j \in \mathcal{U}^l), \tag{18}$$

$$\sum_{\mathrm{J}_j \in \mathcal{U}^l} \sum_{s=\max(t-p_j+1, E_j^l)}^{\min(t,S_j)} x_{js} \le M_t^l \quad (E_{\min}^l \le t \le S_{\max}). \tag{19}$$

Here, $M_t^l$ and $E_j^l$ respectively denote the number of machines available at $t$ and the earliest possible starting time of $\mathrm{J}_j$ corresponding to the partial schedule $\mathcal{S}^l$, and $E_{\min}^l$ is defined by $E_{\min}^l := \min_{\mathrm{J}_j \in \mathcal{U}^l} E_j^l$. A lower bound of the total tardiness of the jobs in $\mathcal{U}^l$ can be obtained by solving the Lagrangian dual corresponding to ($\mathrm{SP}^l$). However, it takes several numbers of iterations for the convergence of subgradient optimization. For this reason, in the previous researches that utilize Lagrangian decomposition in branch-and-bound algorithms (Fisher, 1981; Babu *et al.*, 2004) subgradient optimization is performed for a small number of iterations where multipliers are initialized by those obtained at the parent nodes. On the other hand, in this study the subgradient optimization is performed only at the root node. For every ($\mathrm{SP}^l$), a lower bound is calculated by fixing the multipliers to those obtained at the root node.

To improve this lower bound, $E_j^l$, the earliest possible starting time of $\mathrm{J}_j$, is restricted by a dominance condition. Recall the dominance condition (2) explained in "branch restriction via the Emmons' dominance conditions."

(2) $p_u > p_{r_{m^l,j}}$, $d_u < d_{r_{m^l,j}}$, $c_{m^l}^l + p_u < d_{r_{m^l,j}} + p_{r_{m^l,j}}$.

By noting that $c_{m^l}^l$ corresponds to the stating time of $\mathrm{J}_u$ when it is scheduled on $\mathrm{M}_{m^l}$, the third equation can be interpreted as

$$(\text{The starting time of } \mathrm{J}_u) < d_{r_{m^l,j}} + p_{r_{m^l,j}} - p_u. \tag{20}$$

Therefore, this condition requires that for $\mathrm{J}_u$ to be scheduled on $\mathrm{M}_{m^l}$, the starting time of $\mathrm{J}_u$ on $\mathrm{M}_{m^l}$ should satisfy

$$(\text{The starting time of } \mathrm{J}_u) \ge d_{r_{m^l,j}} + p_{r_{m^l,j}} - p_u \tag{21}$$

for any job $\mathrm{J}_{r_{m^l,j}}$ ($1 \le j \le v_{m^l}^l$) satisfying $p_u > p_{r_{m^l,j}}$ and $d_u < d_{r_{m^l,j}}$. Thus, the earliest possible starting time of $\mathrm{J}_u$ on $\mathrm{M}_{m^l}$ is given by

$$\max \left\{ \max_{\substack{p_u > p_{r_{m^l,j}} \\ d_u < d_{r_{m^l,j}}}} (d_{r_{m^l,j}} + p_{r_{m^l,j}} - p_u),\ c_{m^l}^l \right\}. \tag{22}$$

Since this relation holds on the other machines, the earliest possible starting time $E_u^l$ of $\mathrm{J}_u$ is given by

$$\min_{1 \le k \le m} \max \left\{ \max_{\substack{p_u > p_{r_{kj}} \\ d_u < d_{r_{kj}}}} (d_{r_{kj}} + p_{r_{kj}} - p_u),\ c_k^l \right\} \tag{23}$$

If the dominance condition is not taken into account, $E_u^l$ is given simply by

$$\min_{1 \le k \le m} c_k^l. \tag{24}$$

Since (23) is not less than (24), $E_u^l$ is restricted by the dominance condition, and hence the lower bound is expected to be improved.

The other lower bound is based on the SPT optimality condition. As already mentioned in the preceding subsection, if all the jobs are tardy in the schedule constructed by $\mathcal{D}(P_{\mathrm{r}}^{\mathrm{SPT}}(\cdot))$, this schedule is optimal. Therefore, a job set $\mathcal{T}^l \subset \mathcal{U}^l$ is searched such that all the jobs in $\mathcal{T}^l$ are tardy in the schedule constructed by applying $\mathcal{D}(P_{\mathrm{r}}^{\mathrm{SPT}}(\mathcal{T}^l))$ to $\mathcal{S}^l$. Then, the total tardiness of the jobs in $\mathcal{T}^l$ in this schedule is used as a lower bound of the total tardiness of the jobs in $\mathcal{U}^l$. The detailed procedure is as follows.

0° $c_i^{\mathrm{B}} := c_i^l$ $(1 \le i \le m)$, $\mathcal{U}^{\mathrm{B}} := \mathcal{U}^l$ and $\mathrm{LB} := 0$.
1° Remove a job with the smallest processing time from $\mathcal{U}^{\mathrm{B}}$. Denote this job by $\mathrm{J}_j$.
2° $k := \arg\min_i c_i^{\mathrm{B}}$. If $c_k^{\mathrm{B}} + p_j < d_j$, go to 4°.
3° $\mathrm{LB} := \mathrm{LB} + c_k^{\mathrm{B}} + p_j - d_j$, $c_k^{\mathrm{B}} := c_k^{\mathrm{B}} + p_j$.
4° If $\mathcal{U}^{\mathrm{B}} \neq \phi$, go to 1°. Otherwise, output LB as a lower bound and terminate.

## 5. COMPUTATIONAL EXPERIMENTS

The efficiency of the proposed algorithm is examined by computational experiments. Computation is performed on a personal computer with a Pentium4 2.4GHz.

Problem instances are generated by the Fisher's standard method (Fisher, 1976). First, the integer processing times $p_j$ $(1 \le j \le n)$ are generated by the uniform distributions in $[1, 100]$. Then, let $P = \sum_{j=1}^n p_j$ and the integer duedates $d_j$ $(1 \le j \le n)$ are generated by the uniform distributions in $[P(1 - \tau - R/2)/m, P(1 - \tau + R/2)/m]$. The number of jobs $n$, the number of machines $m$, the tardiness factor $\tau$ and the range of duedates $R$ are changed by $n = 20, 25$, $m = 2, 3, 4, 5, 6, 7, 8, 9, 10$, $\tau = 0.2, 0.6, 1.0$, and $R = 0.2, 0.5, 0.8$. For every combination of $n$, $m$, $\tau$ and $R$, 10 problem instances are generated. Thus, for each combination of $n$ and $m$, 90 problem instances are generated.

The results are shown in Table 1. In this table, the average and maximum computational times are given in seconds. From this table, it can be seen that instances with larger numbers of machines are easier to solve because almost all the instances have no duality gaps

Table 1. Computational results

| | | $m=2$ | $m=3$ | $m=4$ | $m=5$ | $m=6$ | $m=7$ | $m=8$ | $m=9$ | $m=10$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $n=20$ | ave. | 1.546 | 1.157 | 0.524 | 0.270 | 0.318 | 0.287 | 0.423 | 0.314 | 0.289 |
| | max. | 5.867 | 8.910 | 3.350 | 2.734 | 1.889 | 2.797 | 2.393 | 2.297 | 2.268 |
| | $N_1/N_2$ | 46/46 | 52/52 | 73/73 | 83/84 | 79/79 | 84/85 | 82/83 | 90/90 | 89/89 |
| $n=25$ | ave. | 4.572 | 30.990 | 7.336 | 43.878 | 0.736 | 0.626 | 0.681 | 3.753 | 0.627 |
| | max. | 15.971 | 822.746 | 312.422 | 2603.932 | 4.375 | 4.203 | 4.473 | 261.707 | 3.291 |
| | $N_1/N_2$ | 32/32 | 39/39 | 62/62 | 65/65 | 74/74 | 76/76 | 77/77 | 79/80 | 81/82 |

$N_1$: The number of instances solved at the root node. $N_2$: The number of instances without duality gaps.

($N_2$ is large) and are solved at root nodes. However, it is possible that the subgradient optimization does not work well for instances with smaller numbers of machines due to the increase of the number of multipliers (it is given by $(\sum_{j=1}^{n} p_j + (m-1)\max_j p_j)/m$), although they have less or no duality gaps in reality. Since the convergence of the subgradient optimization depends on the step size parameter $\lambda$, it would be necessary to examine how to adjust $\lambda$. Nonetheless, all the instances with 25 jobs are solved by the proposed algorithm within acceptable computational times.

## 6. CONCLUSION

In this study a new branch-and-bound algorithm is proposed for a class of scheduling problems to minimize total tardiness on identical parallel machines. Computational experiments showed that most problem instances have no duality gaps and can be solved at the root node without branching. Even when there is a duality gap, the proposed algorithm can find an optimal solution efficiently. Indeed, it can handle instances with up to 25 jobs and any number of machines. To improve its efficiency, it would be necessary to consider better choices of the step size parameter for the subgradient optimization. It would be also necessary to apply other types of Lagrangian decomposition such as the one proposed in Babu *et al.* (2004).

## REFERENCES


Azizoglu, M. and O. Kirca (1998). Tardiness minimization on parallel machines. *International Journal of Production Economics* **55**, 163–168.

Babu, P., L. Peridy and E. Pinson (2004). A branch and bound algorithm to minimize total weighted tardiness on a single processor. *Annals of Operations Research* **129**, 33–46.

Barnes, J.W. and J.J. Brennan (1977). An improved algorithm for scheduling jobs on identical machines. *AIIE Transactions* **9**, 25–31.

Du, J. and J.Y.T. Leung (1990) . Minimizing total tardiness on one machine is NP-hard. *Mathematics of Operations Research* **15**, 483–495.

Elmaghraby, S.E. and S.H. Park (1974). Scheduling jobs on a number of identical machines. *AIIE Transactions* **6**, 1–13.

Emmons, H. (1969). One-machine sequencing to minimize certain functions of job tardiness. *Operations Research* **17**, 701–715.

Fisher, M.L. (1976). A dual algorithm for the one-machine scheduling problem. *Mathematical Programming* **11**, 229–251.

Fisher, M.L. (1981). The Lagrangian relaxation method for solving integer programming problems. *Management Science* **27**, 1–18.

Gupta, J.N.D. and A.R. Maykut (1973). Scheduling jobs on parallel processors with dynamic programming. *Decision Sciences* **4**, 447–457.

Ho, J.C. and Y.-L. Chang (1991). Heuristics for minimizing mean tardiness for $m$ parallel machines. *Naval Research Logistics* **38**, 367–381.

Koulamas, C. (1994). The total tardiness problem: Review and extensions. *Operations Research* **42**, 1025–1041.

Koulamas, C. (1997). Polynomially solvable total tardiness problems: Review and extensions. *Omega* **25**, 235–239.

Lawler, E.L. (1977). A "pseudopolynomial" algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics* **1**, 331–342.

Liaw, C.-F., Y.-K. Lin, C.-Y. Cheng and M. Chen (2003) Scheduling unrelated parallel machines to minimize total weighted tardiness. *Computers & Operations Research* **30**, 1777–1789.

Luh, P.B., D.J. Hoitomt, E. Max and K.R. Pattipati (1990). Schedule generation and reconfiguration for parallel machines. *IEEE Transactions on Robotics and Automation* **6**, 687–696.

Root, J.G. (1965). Scheduling with deadlines and loss functions on $k$ parallel machines. *Management Science* **11**, 460–475.

Szwarc, W., F.D. Croce and A. Grosso (1999). Solution of the single machine total tardiness problem. *Journal of Scheduling* **2**, 55–71.

Tanaka, S. and M. Araki (2004). Two types of branch-and-bound algorithms for the scheduling problem to minimize total tardiness on identical parallel machines. *International Symposium on Scheduling 2004* pp. 90–93.

Wilkerson, L.J. and J.D. Irwin (1971). An improved method for scheduling independent tasks. *AIIE Transactions* **3**, 239–245.