한국경영과학회지
제39권 제3호
2014년 9월
http://dx.doi.org/10.7737/JKORMS.2014.39.3.083

순서 독립적인 셋업타임을 가진 동일작업의 병렬기계 배치스케줄링*

최병천[1] · 박명주[2†]
[1]충남대학교 경영학부, [2]경희대학교 산업경영공학과

Parallel Machine Scheduling with Identical Jobs and Sequence-Independent Setup Times

Byung-Cheon Choi[1] · Myoung-Ju Park[2]
[1]Department of Business Administration, Chungnam National University
[2]Department of Industrial and Management Systems Engineering, Kyung Hee University

■ Abstract ■

We consider the problem of scheduling identical jobs with sequence-independent setup times on parallel machines. The objective is to minimize total completion times. We present the pseudopolynomial-time algorithm for the case with a fixed number of machines and an efficient approximation algorithm for our problem with identical setup times, which is known to be NP-hard even for the two-machine case.

Keywords : Batch Scheduling, Setup Times, Parallel Machine

1. Problem Definition

Batch scheduling problems with setup times have been studied extensively [2, 3]. In this paper, we consider a particular batch scheduling problem that can be stated as follows. Suppose we have a set of n jobs to be scheduled on m parallel machines, where each job belongs to some batch. Batch scheduling problems are characterized by a setup time that is only required

논문접수일 : 2014년 06월 20일 논문게재확정일 : 2014년 08월 06일
논문수정일(1차 : 2014년 08월 05일)
* This study was financially supported by the research fund of Chungnam National University in 2014.
† 교신저자 pmj0684@khu.ac.kr

between jobs from different batches. Each batch g has its own set of n_g jobs, $J^g = \{J_{g,1}, J_{g,2}, \cdots, J_{g,n_g}\}$, $g = 1, 2, \cdots, b$. Note that $n = \Sigma_{g=1}^{b} n_g$. Let $p_{g,j}$ be the processing time of $J_{g,j}$, $j = 1, 2, \cdots, n_g$, $g = 1, 2, \cdots, b$. In our problem, the processing time of each job is identical, that is, $p_{g,j} = p$, $j = 1, 2, \cdots, n_g$, $g = 1, 2, \cdots, b$. Let $\sigma = (\sigma_1, \sigma_2, \cdots, \sigma_m)$ be the schedule such that σ_i is the subsequence of jobs assigned to machine i, $i = 1, 2, \cdots, m$. Let $\sigma_i(j)$ be the j-th job in σ_i, $i = 1, 2, \cdots, m$. Let $C_{g,j}(\sigma)$ be the completion time of $J_{g,j}$ in σ. Let s_g be the setup time required to process a job in batch g following a job in a different batch. Note that if a job follows a member of the same batch, then a setup time is not required. The objective is to find a schedule σ^* to minimize total completion times, $z(\sigma^*) = \Sigma_{g=1}^{b} \Sigma_{j=1}^{n_g} C_{g,j}(\sigma^*)$. Let this problem be referred to as *Problem P*.

Cheng and Chen [5] showed that Problem P is NP-hard even for the two-machine case with unit length jobs, that is, $p_{g,j} = 1$. Webster [7] showed that Problem P is unary NP-hard even for the case in which each job of the same batch has the same processing time, that is, $p_{g,j} = p_g$. Liu et al. [6] considered the two-machine case of Problem P and presented a pseudopolynomial-time algorithm for the case with unit length jobs and an NP-hardness proof for the case with unit length jobs and identical setup times, that is, $s_g = s$. Webster and Azzioglu [8] presented two dynamic programming algorithms for Problem P with arbitrary processing times whose objective is to minimize the total weighted flow time. In this paper, we present a pseudopolynomial-time algorithm with better complexity than that in [8] for Problem P with a fixed number of machines and an efficient approximation algorithm for Problem P with identical setup times.

2. Problem P

In this section, we introduce an optimality condition and present a pseudopolynomial-time algorithm for Problem P with a fixed number of machines.

2.1 Optimality Condition

In this subsection, we present an optimality condition that is used later to develop a pseudopolynomial-time algorithm.

First, we introduce some terminology and the known result. Let batch g be referred to as a *split* batch if it has at least two setups and let the schedule with no split batches be referred to as a *group technology (GT)* schedule. Note that in the GT schedule, each batch has exactly one setup. Consider a schedule $\sigma = (\sigma_1, \sigma_2, \cdots, \sigma_m)$ such that for $i = 1, 2, \cdots, m$:

- Let α_i be the number of batches allocated to σ_i;
- Let $\pi_i = (\pi_i(1), \pi_i(2), \cdots, \pi_i(\alpha_i))$ be the sequence of the batches allocated to σ_i;
- Let $\sigma_i = (J^{i,\pi_i(1)}, J^{i,\pi_i(2)}, \cdots, J^{i,\pi_i(\alpha_i)})$, where $J^{i,\pi_i(j)}$ is the set of jobs in batch $\pi_i(j)$ in σ_i.

Proposition 1 *[4] There exists an optimal schedule σ^* for the single-machine case of Problem P such that σ^* is a GT schedule and*

$$\frac{s_{\pi^*(1)}}{|J^{\pi^*(1)}|} \le \frac{s_{\pi^*(2)}}{|J^{\pi^*(2)}|} \le \cdots \le \frac{s_{\pi^*(b)}}{|J^{\pi^*(b)}|},$$

where $|J^{\pi^*(j)}|$ is the cardinality of $J^{\pi^*(j)}$. Note that since this is single-machine case, for simplicity, the subscripts of π are deleted.

Following [4], henceforth, we consider only a schedule σ with no split job on each machine such that, for $i=1, 2, \cdots, m$,

$$\frac{s_{\pi_i(1)}}{|J^{i,\pi_i(1)}|} \le \frac{s_{\pi_i(2)}}{|J^{i,\pi_i(2)}|} \le \cdots \le \frac{s_{\pi_i(\alpha_i)}}{|J^{i,\pi_i(\alpha_i)}|}, \quad (1)$$

where $|J^{i,\pi_i(j)}|$ is the cardinality of $J^{i,\pi_i(j)}$. Note that Proposition 1 does not imply that an optimal schedule is a GT schedule. Then, $z(\sigma)$ can be expressed as

$$z(\sigma) = \sum_{i=1}^{m}\sum_{g=1}^{\alpha_i} s_{\pi_i(g)} \sum_{j=g}^{\alpha_i} |J^{i,\pi_i(j)}| \quad (2)$$
$$+\frac{p}{2}\sum_{i=1}^{m}(\sum_{j=1}^{\alpha_i}|J^{i,\pi_i(j)}|)(1+\sum_{j=1}^{\alpha_i}|J^{i,\pi_i(j)}|).$$

It is observed from equation (2) that if the total number of jobs allocated to each machine is fixed, $z(\sigma)$ is determined by the combination of the number of jobs processed after each setup time.

Lemma 1 *Let $G^{\sigma}=(M, N, E^{\sigma})$ be the bipartite graph corresponding to a feasible schedule σ defined as follows :*

- *$M=\{1, \cdots, m\}$ is the set of machines and $N=\{1, \cdots, b\}$ is the set of batches;*
- *$\{u,v\}\in E^{\sigma}$ if some job of batch u is processed on machine v in σ.*

Then, Problem P has an optimal schedule σ with no cycle in G^{σ}.

Proof Suppose that an optimal schedule σ has a cycle C in G^{σ}. Without loss of generality, the cycle C can be represented as

$$C:=1-k_1-2-k_2-\cdots-l-k_l-1,$$

where $i\in M$ and $k_i\in N$, $i=1, \cdots, l$. Let $J^{i,k_{i-1}}$ (J^{i,k_i}) be the set of jobs in $J^{i,k_{i-1}}$ (J^{k_i}) allocated to σ_i, $i=1, 2, \cdots, l$. For consistency of notation, let $k_0=k_l$. Let σ^1 be a schedule identical to σ except that the last job in J^{i,k_i} is moved immediately after the last job in J^{i+1,k_i}, $i=1, 2, \cdots, l$. Let σ^2 be a schedule identical to σ except that the last job in J^{i+1,k_i} is moved immediately after the last job in J^{i,k_i}, $i=1, 2, \cdots, l$. Note that, for simplicity, let $J^{l+1,k_l}=J^{1,k_l}$. Then, we can show that $z(\sigma^1)\le z(\sigma)$.

To do so, we introduce the following additional notation :

- Let S_i be the set of batches between batches k_{i-1} and k_i in σ_i, $i=1, 2, \cdots, l$, respectively;
- Under σ_i, let $n_{i,k_{i-1}}$ and n_{i,k_i} be the number of jobs after the last job in $J^{i,k_{i-1}}$ and J^{i,k_i}, $i=1, 2, \cdots, l$, respectively;
- For $i=1, 2, \cdots, l$, let

$$\delta_{i,k_i}=\begin{cases}(n_{i,k_i}+1)s_{k_i} & \text{if} |J^{i,k_i}|=1 \text{ and } k_i\to k_{i-1},\\ n_{i,k_i}s_{k_i} & \text{if} |J^{i,k_i}|=1 \text{ and } k_{i-1}\to k_i,\\ 0 & \text{otherwise}\end{cases}$$

and

$$\delta_{i,k_{i-1}}=\begin{cases}(n_{i,k_{i-1}}+1)s_{k_{i-1}} & \text{if} |J^{i,k_{i-1}}|=1 \text{ and } k_{i-1}\to k_i,\\ n_{i,k_{i-1}}s_{k_{i-1}} & \text{if} |J^{i,k_{i-1}}|=1 \text{ and } k_i\to k_{i-1},\\ 0 & \text{otherwise}\end{cases}$$

where $k_{i-1}\to k_i$ means that batch k_{i-1} is processed before batch k_i. For $i=1, 2, \cdots, l$,

$$\rho_i=\begin{cases}1 & \text{if } k_{i-1}\to k_i,\\ 0 & \text{if } k_i\to k_{i-1}.\end{cases}$$

Then,

$$z(\sigma^1) = z(\sigma) + \Delta^1 \ \text{ and } \ z(\sigma^2) = z(\sigma) + \Delta^2,$$

where

$$\Delta^1 = \sum_{i=1}^{l}\Big(-\delta_{i,k_i} + \rho_i(-\sum_{g \in S_i} s_g - s_{k_i}) + (1-\rho_i)(\sum_{g \in S_i} s_g + s_{k_{i-1}})\Big)$$

and

$$\Delta^2 = \sum_{i=1}^{l}\Big(-\delta_{i,k_{i-1}} + \rho_i(\sum_{g \in S_i} s_g + s_{k_i}) + (1-\rho_i)(-\sum_{g \in S_i} s_g - s_{k_{i-1}})\Big).$$

Since σ is an optimal schedule, the following inequalities should be satisfied :

$$\Delta^1 \geq 0 \ \text{ and } \ \Delta^2 \geq 0. \qquad (3)$$

Let $\overline{\Delta} = \Sigma_{i=1}^{l}\big(\rho_i(-\Sigma_{g \in S_i} s_g - s_{k_i}) + (1-\rho_i)(\Sigma_{g \in S_i} s_g + s_{k_{i-1}})\big)$. Then, by inequalities (3) and the definitions of δ_{i,k_i} and $\delta_{i,k_{i-1}}$,

$$0 \leq \sum_{i=1}^{l} \delta_{i,k_i} \leq \overline{\Delta} \leq -\sum_{i=1}^{l} \delta_{i,k_{i-1}} \leq 0.$$

Since $\overline{\Delta} = 0$,

$$z(\sigma^1) = z(\sigma) - \sum_{i=1}^{l} \delta_{i,k_i} \leq z(\sigma).$$

By repeatedly applying the argument used for σ^1, we can construct a new schedule $\overline{\sigma}$ such that $z(\overline{\sigma}) \leq z(\sigma)$ and $G^{\overline{\sigma}}$ does not contain C. The proof is complete. ■

2.2 Pseudopolynomial-time Algorithm

In this subsection, we develop a pseudopolynomial-time algorithm for Problem P with a fixed number of machines. First, we consider the problem of finding an optimal schedule among GT schedules. Let this problem be referred to as *Problem PGT*.

Lemma 2 *Problem PGT can be solved in time* $O(bmn^m)$.

Proof For simplicity, let the batches be indexed in non-decreasing order of $\frac{s_g}{n_g}$, that is,

$$\frac{s_1}{n_1} \leq \frac{s_2}{n_2} \leq \cdots \leq \frac{s_b}{n_b}.$$

When c_i jobs are processed on machine i while the schedule is being constructed, let batch g be processed before the first job on machine i. Then, it is observed that

- The completion time of job $J_{g,j}$ is $s_g + jp$;
- The total completion time of jobs in batch g is $n_g s_g + \frac{p}{2} n_g(n_g+1)$.
- The total completion time of jobs after batch g is increased by $c_i(s_g + n_g p)$.

Based on these observations, we reduce Problem PGT into the shortest path problem in an acyclic graph. Let $N(g; c_1, c_2, \cdots, c_m)$ be the node that represents the following :

- The machines on which the batches in $\{b, b-1, \cdots, g\}$ are processed have been determined;
- c_i is the number of jobs allocated to machine i, $i = 1, 2, \cdots, m$.

Let s $:= N(b+1; \underbrace{0, \cdots, 0}_{m})$ and t be the source and sink nodes, respectively. For $g = 1, \cdots, b$ and $i = 1,$

$\cdots, m$, let $N(g+1; c_1, \cdots, c_m)$ be connected to $N(g, \bar{c}_1, \cdots, \bar{c}_m)$ with weight $((n_g s_g + \frac{p}{2} n_g (n_g+1) + c_i (s_g + n_g p))$, if $\bar{c}_i = c_i + n_g$ and $\bar{c}_{i'} = c_{i'}$ for each $i' \in \{1, 2, \cdots, b\} \setminus \{i\}$. This edge denotes that batch g is processed before the first job on machine i. Let $N(1; c_1, c_2, \cdots, c_m)$ be connected to t with weight 0.

It is clear that the $s-t$ shortest path of the reduced graph represents an optimal schedule for Problem PGT. Since the reduced graph is acyclic and the number of edges is $O(bmn^m)$, the $s-t$ shortest path can be found in $O(bmn^m)$ by the algorithm in [1]. The proof is complete. ■

It is observed from Lemma 1 that, in Problem P, there exists an optimal schedule with at most $(m-1)$ split batches, each of which can be processed on at most m machines. Let $(\omega_1, \omega_2, \cdots, \omega_{m-1})$ be the combination such that $\omega_h = (\omega_{h,1}, \omega_{h,2}, \cdots, \omega_{h,m})$ is the vector of sub-batches of batch ω_h. Let $a_{h,i}$ be the number of the jobs in sub-batch ω_h allocated to machine i. Note that $a_{h,i}$ can become zero for some i. This implies that no jobs in batch ω_h are allocated to machine i. For each combination $(\omega_1, \omega_2, \cdots, \omega_h)$, we can construct Problem PGT, where sub-batches are regarded as different batches. Note that if $i \neq i'$, then sub-batches $\omega_{g,i}$ and $\omega_{g,i'}$ are regarded as different batches in the Problem PGT. Let b' be the number of batches in Problem PGT. Then, by Lemma 1,

$$b' \leq b + m(m-1) \leq bm^2. \tag{4}$$

It is observed that the optimal schedule of Problem IP is identical to the schedule with the minimum total completion times among the optimal schedules of each combination. Based on this observation, we can construct the following algorithm.

Algorithm ALG

Step 1 For each combination $(\omega_1, \omega_2, \cdots, \omega_{m-1})$, construct the corresponding Problem PGT.

Step 2 For each Problem PGT, obtain an optimal schedule by using the approach in Lemma 2.

Step 3 Select the schedule with the minimum total completion time.

Note that since the number of combinations $(\omega_1, \omega_2, \cdots, \omega_{m-1})$ is $\binom{b}{m-1} = O(b^{m-1})$ and the number of combinations $(a_{h,1}, a_{h,2} \cdots, a_{h,m})$ is $O(n_{\omega_h}^{m-1})$ for each batch ω_h, $h = 1, 2, \cdots, m-1$, the total number of combinations can be calculated as follows :

$$O(b^{m-1} \prod_{h=1}^{m-1} n_{\omega_h}^{m-1}) = O(b^{m-1} (\frac{n}{m-1})^{(m-1)^2}).$$

Furthermore, by Lemma 2 and inequality (4), each Problem PGT can be solved in $O(b'mn^m)$. Thus, Algorithm ALG terminates in $O\left(b^m n^{(m-1)^2+1} m^{3-(m-1)^2}\right)$.

Theorem 1 *Problem P can be solved in pseudo-polynomial-time when the number of machines is fixed.*

Proof To encode Problem P, we just need the setup time of each batch, the number of jobs belonging to each batch and the processing time. Thus, the order of the input size is $(b \log n_{\max} + b \log s_{\max} + \log p)$, where $n_{\max} = \max\{n_1, \cdots, n_b\}$ and $s_{\max} = \max\{s_1, \cdots, s_b\}$. The complexity of Algorithm ALG is pseudopolynomial when the number of machines is fixed. ■

Remark 1 *When we apply dynamic programming algorithms [8] for Problem P, their complexities are* $O(mb^{m-b+1}n^{m+b-1})$ *and* $O(mb^{m-b+2}n^{b}(P+S)^{m})$*, respectively, where* $P = np$ *and* $S = \Sigma_{g=1}^{b} s_g$*. Since they are pseudopolynomial-times only if the numbers of machines and batches are fixed, Algorithm ALG is more efficient.*

3. Problem P with Identical Setup Times

In this section, we consider Problem P with identical setup times, that is, $s_g = s,\ g = 1, 2, \cdots, b$. Since Problem P is NP-hard even for the two-machine case with identical setup times and unit processing times [6], we propose an approximation algorithm for Problem P with identical setup times. Without loss of generality, assume that the batches are indexed in non-increasing order of n_g, that is,

$$n_1 \geq n_2 \geq \cdots \geq n_b.$$

Since the setup times are identical, equation (2) can be rewritten as

$$z(\sigma) = s\sum_{i=1}^{m}\sum_{g=1}^{\alpha_i}\sum_{j=g}^{\alpha_i}|J^{i,\pi_i(j)}| + \frac{p}{2}\sum_{i=1}^{m}\gamma_i(\gamma_i+1),$$

where $\gamma_i = \Sigma_{j=1}^{\alpha_i}|J^{i,\pi_i(j)}|$. Since $\Sigma_{i=1}^{m}\Sigma_{g=1}^{\alpha_i}\Sigma_{j=g}^{\alpha_i}|J^{i,\pi_i(j)}| = \Sigma_{i=1}^{m}\Sigma_{j=1}^{\alpha_i} j|J^{i,\pi_i(j)}|$, however, $z(\sigma)$ can be rewritten as

$$z(\sigma) = s\sum_{i=1}^{m}\sum_{j=1}^{\alpha_i} j|J^{i,\pi_i(j)}| + \frac{p}{2}\sum_{i=1}^{m}\gamma_i(\gamma_i+1). \quad (5)$$

Note that the objective function (5) consists of two parts. Let

$$f(\sigma) = s\sum_{i=1}^{m}\sum_{j=1}^{\alpha_i} j|J^{i,\pi_i(j)}| \text{ and } h(\sigma) = \frac{p}{2}\sum_{i=1}^{m}\gamma_i(\gamma_i+1).$$

To develop an approximation algorithm, we introduce additional notation. Let k and r be the quotient and remainder, respectively, when b is divided by m, that is, $b = km + r$. Consider a GT schedule $\tau = (\tau_1, \cdots, \tau_m)$ as follows :

$$\tau_i = \begin{cases} (J^i, J^{m+i}, \cdots, J^{km+i}) & \text{for } i = 1, \cdots, r, \\ (J^i, J^{m+i}, \cdots, J^{(k-1)m+i}) & \text{for } i = r+1, \cdots, m. \end{cases} \quad (6)$$

Let q and u be the quotient and remainder, respectively, when n is divided by m, that is, $n = qm + u$. Let

$$L_i = \begin{cases} q+1 & \text{for } i = 1, 2, \cdots, u, \\ q & \text{for } i = u+1, u+2, \cdots, m. \end{cases}$$

We present an approximation algorithm for Problem P with identical setup times. The underlying idea is to modify τ into a schedule such that the number of jobs processed on machine i is exactly $L_i,\ i = 1, 2, \cdots, m$.

Algorithm APP

Step 1 Sort the batches by the decreasing order of the number of jobs and let $Q = \varnothing$.

Step 2 Construct a schedule $\tau = (\tau_1, \cdots, \tau_m)$, defined in (6).

- Let $\bar{\gamma}_i$ be the number of jobs on machine i in $\tau,\ i = 1, 2, \cdots, m$.
- Let $\bar{r}$ be the index such that $\bar{\gamma}_i > L_i,\ i = 1, 2, \cdots \bar{r}$ and $\bar{\gamma}_i \leq L_i,\ i = \bar{r}+1, \bar{r}+2, \cdots, m$.

Step 3 For $i = 1, 2, \cdots, \bar{r}$, move the first $(\bar{\gamma}_i - L_i)$ jobs from τ_i into Q and sort the jobs by increasing batch index.

Step 4 For $i=\bar{r}+1,\ \bar{r}+2,\ \cdots,\ m$, sequence the first $(L_i-\bar{\gamma}_i)$ jobs from Q before the first job in τ_i.

Step 5 Output a new schedule.

Note that since sorting batches in Step 1 and Steps 2~4 requires $O(b\log b)$ and $O(b+m)$ times, respectively, Algorithm APP terminates in $O(b\log b+m)$ time.

Lemma 3 $f(\tau)\le f(\sigma)$.

Proof Consider two cases.

i) σ is a GT schedule

Suppose that $\tau\neq\sigma$. Let $vm+i$ be the smallest index in σ such that batch $vm+i$ is not processed on machine i. Without loss of generality, assume that batch $vm+i$ is processed on machine i' in σ. Note that $i'\neq i$. It is observed from relation (1) that batch $vm+i$ is processed at the $(v+1)$-th position or later on machine i' in σ. Let batch g be the $(v+1)$-th batch on machine i in σ. We can make a new schedule σ^1 by exchanging the positions of batches $vm+i$ and g. Since $n_{vm+i}\ge n_g$, it is observed that $f(\sigma^1)\le f(\sigma)$. By repeatedly applying the argument above, we can attain a schedule τ and thus $f(\tau)\le f(\sigma)$.

ii) σ is not a GT schedule

Suppose batch g is processed on machines i and i' in σ. Let batch g be the k_i-th and $k_{i'}$-th batches in σ_i and σ_i', respectively. Without loss of generality, assume that $k_i\ge k_{i'}$. Then, we construct a new schedule σ^2 by moving all the jobs of batch g on machine i immediately after batch g on machine i'. Then,

$$f(\sigma^2)=f(\sigma)-(k_i-k_{i'})|J^{i,g}|-\sum_{j=k_i+1}^{\alpha_i}|J^{i,\pi_i(j)}|\le f(\sigma).$$

By repeatedly applying the argument above, we can attain a GT schedule $\hat{\sigma}$ and thus $f(\tau)\le f(\hat{\sigma})\le f(\sigma)$by case i).

By cases i) and ii), the proof is complete. ■

Theorem 2 *Let ρ be the schedule obtained by Algorithm APP. Then,*

$$\frac{z(\rho)}{z(\rho^*)}\le 1+\frac{2s(m-1)}{2sm+p(n+m)}.$$

Proof For i in $i\in\{\bar{r}+1,\bar{r}+2,\cdots,m\}$, let $(\bar{\pi}(1),\cdots,\bar{\pi}_i(l_i))$ be the subsequence of the batches moved to machine i by Step 4 of Algorithm APP.

Claim $\Sigma_{i=\bar{r}+1}^{m}(l_i-1)\le\bar{r}-1$.

Proof It is observed from the construction of τ that $\bar{\gamma}_1\ge\bar{\gamma}_2\ge\cdots\ge\bar{\gamma}_m$ and

$$L_i\ge\bar{\gamma}_m\ge\bar{\gamma}_i-n_{i,}\quad i=1,\cdots,\bar{r}. \tag{7}$$

Inequality (7) implies the following :

- Since the first $(\bar{\gamma}_i-L_i)$ jobs of τ_i belong to batch i, $i=1,2,\cdots,\bar{r}$, the set of batches in Q after Step 3 is $\{1,2,\cdots,\bar{r}\}$;
- Since batch 1 is always sequenced at the first position, it does not belong to $\{\bar{\pi}_i(2),\cdots,\bar{\pi}_i(l_i)\}$ for $i=\bar{r}+1,\cdots,m$.

Furthermore, it is observed from the way to sequence jobs in Step 4 that if $i\neq i'$, then $\{\bar{\pi}_i(2),\cdots,\bar{\pi}_i(l_i)\}$ and $\{\bar{\pi}_{i'}(2),\cdots,\bar{\pi}_{i'}(l_{i'})\}$ are disjoint. By the implications above and this observation,

$$\sum_{i=\bar{r}+1}^{m}(l_i-1)=\sum_{i=\bar{r}+1}^{m}|\{\bar{\pi}_i(2),\cdots,\bar{\pi}_i(l_i)\}|$$

$$=|\bigcup_{i=\bar{r}+1}^{m}\{\bar{\pi}_i(2),\cdots,\bar{\pi}_i(l_i)\}|\le\bar{r}-1.$$

The proof is complete. □

By Claim,

$$\sum_{i=\bar{r}+1}^{m}(l_i-1)L_i\le\sum_{i=\bar{r}+1}^{m}(l_i-1)L_{\bar{r}+1}\le(\bar{r}-1)L_{\bar{r}+1}\le\sum_{i=2}^{\bar{r}}L_i. \quad (8)$$

We, henceforth, introduce four relations to derive the bound.

i) When τ is transformed into ρ by Algorithm APP, $f(\tau)$ is increased by at most $s\Sigma_{i=\bar{r}+1}^{m}l_iL_i$. Thus, by inequality (8),

$$f(\rho)-f(\tau)\le s\sum_{i=\bar{r}+1}^{m}l_iL_i=s\left\{\sum_{i=\bar{r}+1}^{m}L_i+\sum_{i=\bar{r}+1}^{m}(l_i-1)L_i\right\}$$
$$\le s\left\{\sum_{i=\bar{r}+1}^{m}L_i+\sum_{i=2}^{\bar{r}}L_i\right\}=s(n-L_1)\le s\frac{m-1}{m}n. \quad (9)$$

ii) By Lemma 3, $h(\sigma^*)\ge\frac{p}{2}\Sigma_{i=1}^{m}L_i(L_i+1)$ and the way to construct ρ,

$$z(\sigma^*)=f(\sigma^*)+h(\sigma^*)\ge f(\tau)+\frac{p}{2}\sum_{i=1}^{m}L_i(L_i+1) \quad (10)$$
$$=f(\tau)+h(\rho).$$

iii) Since $\Sigma_{i=1}^{m}L_i(L_{i+1})\ge\Sigma_{i=1}^{m}\frac{n}{m}(\frac{n}{m}+1)$,

$$h(\rho)=\frac{p}{2}\sum_{i=1}^{m}L_i(L_i+1)\ge\frac{p}{2}\sum_{i=1}^{m}\frac{n}{m}(\frac{n}{m}+1) \quad (11)$$
$$=\frac{pn(n+m)}{2m}.$$

iv) Let α'_i be the number of batches allocated to τ_i and let $\pi'_i=(\pi'_i(1),\pi'_i(2),\cdots,\pi'_i(\alpha'_i))$. be the sequence of batches allocated to τ_i, $i=1,2,\cdots,m$. Then,

$$f(\tau)=s\sum_{i=1}^{m}\sum_{j=1}^{\alpha'_i}j|J^{\pi'_i(j)}|\ge s\sum_{i=1}^{m}\sum_{j=1}^{\alpha'_i}|J^{\pi'_i(j)}|=sn. \quad (12)$$

Then, by inequalities (9)~(12),

$$\frac{z(\rho)}{z(\sigma^*)}\le\frac{f(\tau)+s\frac{m-1}{m}n+h(\rho)}{f(\tau)+h(\rho)}$$
$$=1+\frac{s\frac{m-1}{m}n}{f(\tau)+h(\rho)}\le1+\frac{s\frac{m-1}{m}n}{sn+\frac{pn(n+m)}{2m}}$$
$$=1+\frac{2s(m-1)}{2sm+p(n+m)}.$$

The proof is complete. ■

References

[1] Ahuja, R.A., K. Mehlhorn, and J.B. Orlin, "Faster algorithm for the shortest path problem," *Journal of the Association for Computing Machinery*, Vol.37(1990), pp.213-223.

[2] Allahverdi, A., J.N.D. Gupta, and T. Aldowaisan, "A review of scheduling research involving setup considerations," *Omega*, Vol.27(1999), pp.219-239.

[3] Allahverdi, A., C.T. Ng, T.C.E. Cheng, and M.Y. Kovalyov, "A survey of scheduling problems with setup times or costs," *European Journal of Operational Research*, Vol.187(2008), pp.985-1032.

[4] Baker, K.R. and D. Trietsch, *Principles of Scheduling and Sequencing*, John Wiley and

Sons, Inc, 2009.

[5] Cheng, T.C.E. and Z.L. Chen, "Parallel machine scheduling with batch setup times," *Operations Research*, Vol.42(1994), pp.1171-1174.

[6] Liu, Z., W. Yu, and T.C.E. Cheng, "Scheduling groups of unit length jobs on two identical parallel machines," *Information Processing Letters*, Vol.69(1999), pp.275-281.

[7] Webster, S.T., "The complexity of scheduling job families about a common due date," *Operations Research Letters*, Vol.20(1997), pp. 65-74.

[8] Webster, S.T. and M. Azzioglu, "Dynamic programming algorithms for scheduling parallel machines with family setup times," *Computers and Operations Research*, Vol.28(2001), pp.127-137.