The single machine batching problem with family setup times to minimize maximum lateness is strongly NP-hard

T.C.E. CHENG[1*], C.T. NG[1] and J.J. YUAN[1,2]

[1]Department of Management, The Hong Kong Polytechnic University,
Hung Hom, Kowloon, Hong Kong, People's Republic of China

[2]Department of Mathematics, Zhengzhou University,
Zhengzhou, Henan 450052, People's Republic of China

ABSTRACT

In this paper, we consider the single machine batching problem with family setup times to minimize maximum lateness. While the problem was proved to be binary NP-hard in 1978, whether the problem is strongly NP-hard is a long-standing open question. We show that this problem is strongly NP-hard.

Keywords: Scheduling; Batching; Due-dates; Maximum lateness; Multi-operation jobs

1 Introduction and Problem Formulation

In the single machine, family jobs, batch scheduling problem (see [1, 6]), we have n jobs $J_1, J_2, ..., J_n$ and F family of jobs $\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_F$, which partition the job set $\{J_1, J_2, ..., J_n\}$. Each job J_j has a processing time p_j and a due date d_j, and each family $\mathcal{F}_f$ has an associated setup time s_f. The jobs in a family are processed in batches and each batch of family $\mathcal{F}_f$ will incur a setup time s_f. In the literature, the problem is denoted by $1|s_f|V$, where V is the objective function to be minimized.

*Corresponding author

The maximum lateness scheduling problem $1|s_f|L_{\max}$ was first researched by Bruno and Downey [1] in 1978. The binary NP-hardness proof for the problem $1|s_f|L_{\max}$ was given by Bruno and Downey [1]. By Bruno and Downey [1], $1|s_f|L_{\max}$ is NP-hard even for either two distinct due dates, two jobs per family, or three distinct due dates, three jobs per family, and equal setup times; however, it is pseudo-polynomially solvable for a fixed number of due dates. The best algorithm for the problem $1|s_f|L_{\max}$ is a dynamic programming algorithm given by Ghosh and Gupta [4] with a time bound $O(F^2N^F)$, where $N = \frac{1}{F}\sum_{1\leq f\leq F}|\mathcal{F}_f| + 1$. Correspondingly, it is shown by Gerodimos, Glass, Potts and Tautenhahn [3] that, the problem $1|s_f,\ \textit{assembly}|L_{\max}$ is binary NP-hard, and can be solved by applying the dynamic programming algorithm of Ghosh and Gupta [4] with a time bound $O(F^2n^F)$. Bruno and Downey [1] first posed the question of whether the problem $1|s_f|L_{\max}$ is strongly NP-hard in 1978. They wrote "One issue that we have not been able to resolve is whether the general problem ($1|s_f|\sum U_j = 0$) is NP-complete when the task lengths, setup times and/or change-over costs are not allowed to be exponentially large with respect to the number of tasks." Ghosh and Gupta [4] pointed out that the long-standing question as to whether $1|s_f|L_{\max}$ is strongly NP-hard had remained open.

To clarify the arguments, we will use the notation of the single machine, multi-operation jobs, assembly scheduling problem for our discussion.

As introduced by Gerodimos, Glass, Potts and Tautenhahn [3], the single machine, multi-operation jobs, assembly scheduling problem arises in a food manufacturing environment. It can be stated as follows: Let n multi-operation jobs $J_1, J_2, ..., J_n$ and a single machine that can handle only one job at a time be given. Each job consists of several operations that belong to different families. There are F families $\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_F$. We assume that each job has at most one operation in each family. If job J_j has an operation in family $\mathcal{F}_f$, then we denote this operation by (f, j) and its associated processing time by $p_{(f,j)} > 0$. The processing time of each job J_j is defined by $p_j = \sum_{(f,j)\in\mathcal{F}_f} p_{(f,j)}$. Each family $\mathcal{F}_f$ has an associated setup time s_f. If in a schedule the operations of a family $\mathcal{F}_f$ are processed in batches, then each batch will incur a setup time s_f. A job completes when all of its operations have been processed. Hence, the completion time of the job J_j under a schedule π is

$$C_j(\pi) = \max\{C_{(f,j)}(\pi) : (f, j) \text{ is an operation of job } J_j\},$$

where $C_{(f,j)}(\pi)$ is the completion time of the operation (f, j). Suppose that the due-date of each job J_j is d_j, $1 \leq j \leq n$. For a given schedule π, we define $U_j(\pi) = 0$ if $C_j(\pi) \leq d_j$, and $U_j(\pi) = 1$ if $C_j(\pi) > d_j$, $1 \leq j \leq n$. Hence, a job J_j is tardy if and only if $U_j = 1$. We also define the lateness of a job J_j as $L_j(\pi) = C_j(\pi) - d_j$, $1 \leq j \leq n$. The objective considered in this paper is to find a schedule π that minimizes the maximum lateness $L_{\max}(\pi) = \max_{1\leq j\leq n} L_j(\pi)$. We call this problem the single machine, multi-operation jobs, maximum lateness scheduling problem. Following [3], we denote the problem by

$$1|s_f,\ \textit{assembly}|L_{\max},$$

where the term "assembly" is used to describe the fact that a job completes when it becomes available for assembly, i.e., when all of its operations have been processed. The related feasible problem, denoted by $1|s_f,\ \textit{assembly}|\sum U_j = 0$, asks whether there is a feasible schedule π such that all jobs are on time. If each job has a single operation, then $1|s_f,\ \textit{assembly}|L_{\max}$ degenerates into the standard single machine, family jobs, maximum lateness scheduling problem, $1|s_f|L_{\max}$ (see [1, 6]). In [3], the equivalence between $1|s_f,\ \textit{assembly}|L_{\max}$ and $1|s_f|L_{\max}$ is established.

We show in this paper that the feasibility problem $1|s_f,\ \textit{assembly}|\sum U_j = 0$ is strongly NP-hard. Hence, the feasibility problem $1|s_f|\sum U_j = 0$ is also strongly NP-hard. Consequently, both problems $1|s_f,\ \textit{assembly}|L_{\max}$ and $1|s_f|L_{\max}$ are strongly NP-hard.

2 NP-hardness Proof

As implied in [3], we have

Lemma 2.1 If the problem

$$1|s_f,\ \textit{assembly}|\sum U_j = 0$$

is feasible, then there is a feasible schedule π such that, within each family, the operations of jobs are sequenced in the earliest due date (EDD) order under π, i.e., for two operations (f,i) and (f,j) contained in a family $\mathcal{F}_f$, $1 \le f \le F$, (f,i) is processed before (f,j) under π if and only if $d_i \le d_j$.

We need the following strongly NP-complete 3-partition problem.

3-Partition Given a set of $3t$ integers $a_1, a_2, ..., a_{3t}$, where $1 \le a_i \le B-1$, such that $\sum_{i=1}^{3t} a_i = tB$, is there a partition of the a_i's into t groups of 3, each summing exactly to B?

By Garey and Johnson [2], we have

Lemma 2.2 The 3-partition problem is strongly NP-complete.

Theorem 2.3 The feasibility problem

$$1|s_f,\ \textit{assembly}|\sum U_j = 0$$

is strongly NP-complete.

Proof: The feasibility problem is clearly in NP. To prove the NP-completeness, we use the strongly NP-complete 3-partition problem for our reduction.

For a given instance of the 3-partition problem with $a_1, a_2, ..., a_{3t}$, where $\frac{1}{t}\sum_{i=1}^{3t} a_i = B$, we construct an instance of the feasibility problem with t jobs $J_1, J_2, ..., J_t$ and $4t$ families $\mathcal{F}_1, \mathcal{F}_2, ...\mathcal{F}_{4t}$ as follows:

$n = t$;

$F = 4t$;

$\mathcal{F}_f = \{(f, j) : 1 \le j \le t\}$, for $1 \le f \le 3t$;

$\mathcal{F}_{3t+i} = \{(3t+i, i), (3t+i, i+1)\}$, for $1 \le i \le t-1$;

$\mathcal{F}_{4t} = \{(4t, t)\}$;

$p_{(f,j)} = Z + a_f$, for $1 \le j \le t$ and $1 \le f \le 3t$, where $Z = t(t+2)B$;

$s_f = Y + a_f$, for $1 \le f \le 3t$, where $Y = (3t^2+1)Z$;

$p_{(3t+i,i)} = p_{(3t+i,i+1)} = s_{3t+i} = p_{(4t,t)} = s_{4t} = X$, for $1 \le i \le t-1$, where $X = 6tY$;

$d_j = (3j-1)X + 3(t+j-1)Y + \frac{3t(t+1)}{2}Z + \frac{3(j-1)(2t-j)}{2}Z + (j-1)(t+2)B + \frac{(t-j+1)(t+j+2)}{2}B$, for $1 \le j \le t$.

Clearly, the construction can be done in polynomial time. We show in the sequel that the instance of the 3-partition problem has a solution if and only if the instance of our scheduling problem is feasible, i.e., there is a schedule such that every job is on time.

The following are some observations about the instance of our feasibility problem.

Observation 1 $d_1 < d_2 < ... < d_t$.

Observation 2 For each job J_j, $1 \le j \le t$,

$$\begin{aligned} d_j &< (3j-1)X + 3(t+j-1)Y + \tfrac{3t(t+1)}{2}Z + \tfrac{3(j-1)(2t-j)}{2}Z + Z \\ &\le (3j-1)X + (3t+3j-2)Y \\ &< 3jX. \end{aligned}$$

Observation 3 $p_1 = X + 3tZ + tB$, and $p_r = 2X + 3tZ + tB$ for $2 \le r \le t$.

If the 3-partition problem has a solution, we can re-lable the indices of $a_1, a_2, ..., a_{3t}$ such that

$$a_{3i-2} + a_{3i-1} + a_{3i} = B, \text{ for } 1 \le i \le t.$$

We construct a schedule π of our feasibility problem as follows.

Each family $\mathcal{F}_f$ with $3t-2 \le f \le 4t$ acts as a batch. Each family $\mathcal{F}_f$ with $1 \le f \le 3(t-1)$ is divided into two batches $\mathcal{B}_f$ and $\mathcal{A}_f$ such that

$$\mathcal{B}_f = \{(f, i) : 1 \le i \le \lceil \frac{1}{3} f \rceil\}$$

and

$$\mathcal{A}_f = \{(f, i) : \lceil \frac{1}{3} f \rceil < i \le t\}.$$

The batches are processed according to the following order under π:

$$\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, ..., \mathcal{B}_{3i-2}, \mathcal{B}_{3i-1}, \mathcal{B}_{3i}, ..., \mathcal{B}_{3t-5}, \mathcal{B}_{3t-4}, \mathcal{B}_{3(t-1)}, \mathcal{F}_{3t-2}, \mathcal{F}_{3t-1}, \mathcal{F}_{3t},$$

$$\mathcal{F}_{3t+1}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, ..., \mathcal{F}_{3t+i}, \mathcal{A}_{3i-2}, \mathcal{A}_{3i-1}, \mathcal{A}_{3i}, ..., \mathcal{F}_{4t-1}, \mathcal{A}_{3t-5}, \mathcal{A}_{3t-4}, \mathcal{A}_{3(t-1)}, \mathcal{F}_{4t}$$

The operations in each batch are sequenced by the EDD order under π.

It is not hard to verify that, under the above schedule π, $C_j(\pi) = d_j$ for $1 \le j \le n = t$, and so $\sum_{j=1}^{t} U_j(\pi) = 0$. Hence, our problem is feasible.

Now suppose that our problem is feasible. We need to show that the 3-partition problem has a solution. By Lemma 2.1 and Observation 1, we have the following claim.

Claim 1 There is a schedule π of our feasibility problem such that

(1) each job is on time under π;

(2) for every two operations (f, i) and (f, j) of any family $\mathcal{F}_f$ with $i < j$, $C_{(f,i)}(\pi) < C_{(f,j)}(\pi)$;

(3) the job indices in each batch are consecutive, i.e., if $\mathcal{H}$ is a batch of the family $\mathcal{F}_f$ under π, then for every two operations $(f, i), (f, j) \in \mathcal{H}$ with $i < j$, $\{(f, k) : i \le k \le j\} \subseteq \mathcal{H}$.

Let π be a feasible schedule of our problem that satisfies the properties in Claim 1. We need more properties of π.

Claim 2 Each family $\mathcal{F}_{3t+i}$ with $1 \le i \le t$ acts as a batch under π.

Suppose, to the contrary, that a certain family $\mathcal{F}_{3t+i}$ with $1 \le i \le t-1$ is divided into two batches under π. Because $s_{3t+i} = X$ for $1 \le i \le t$, we have at least $t+1$ batches each of which has a setup time X. Since each family $\mathcal{F}_{3t+i}$ with $1 \le i \le t-1$ has two operations, each with processing time X, and the family $\mathcal{F}_{4t}$ has one operation with processing time X, the makespan $C_{\max}$ is greater than $(t+1)X + (2t-1)X > d_t = \max_{1 \le i \le t} d_i$, where the inequality is obtained from Observation 2. This contradicts the fact that π is a feasible solution of our problem. The proof of Claim 2 is completed.

Claim 3 $C_j(\pi) = C_{(3t+j,j)}(\pi)$, i.e., $(3t + j, j)$ is the final operation of job J_j under π, for $1 \le j \le t-1$.

Otherwise, there is a job J_j with $1 \le j \le t-1$ such that the processing of every operation in the families $\mathcal{F}_{3t+i}$ with $1 \le i \le j$ is completed on or before the time $\max_{1 \le i \le j} C_i(\pi)$. Then, $\max_{1 \le i \le j} C_i(\pi) \ge 3jX > d_j$, where the latter inequality is obtained from Observation 2. This contradicts the fact that π is a feasible schedule of our problem. The proof of Claim 3 is completed.

For the reason that each of the $3t + 1$ families $\mathcal{F}_1, ..., \mathcal{F}_{3t}, \mathcal{F}_{3t+1}$ contains an operation of job J_1, and the family $\mathcal{F}_{3t+1}$ is processed in a single batch under π, we can suppose that $\mathcal{B}_f$ is a batch of family $\mathcal{F}_f$ under π, such that the operation $(f, 1) \in \mathcal{B}_f$, $1 \le f \le 3t$. Furthermore, we re-lable the indices of $\mathcal{F}_1, ..., \mathcal{F}_{3t}$ such that

$$|\mathcal{B}_1| \le |\mathcal{B}_2| \le ... \le |\mathcal{B}_{3t}|.$$

Write $|\mathcal{B}_f| = b_f$, for $1 \le f \le 3t$. Then, by Claim 1(3),

$$\mathcal{B}_f = \{(f, 1), (f, 2), ..., (f, b_f)\}, \text{ for } 1 \le f \le 3t.$$

Claim 4 $b_{3r-2} \geq r$, for $1 \leq r \leq t$.

Otherwise, there is a certain r with $2 \leq r \leq t$, such that $b_{3r-2} \leq r-1$. Then, $b_f \leq r-1$ and $(f,r) \notin \mathcal{B}_f$, for $1 \leq f \leq 3r-2$. For each f with $1 \leq f \leq 3r-2$, let $\mathcal{A}_f$ be the batch under π such that $(f,r) \in \mathcal{A}_f$. The setup time of each batch in $\{\mathcal{B}_f : 1 \leq f \leq 3t\} \cup \{\mathcal{A}_f : 1 \leq f \leq 3r-2\}$ is greater than Y. With the batches $\mathcal{F}_{3t+1}, ..., \mathcal{F}_{3t+r}$ being considered, the maximum completion time $\max_{1\leq j\leq r} C_j(\pi)$ of the jobs in $\{J_1, J_2, ..., J_r\}$ is greater than $(3t+3r-2)Y + (3r-1)X$. By Observation 2, $\max_{1\leq j\leq r} C_j(\pi) > d_r$, contradicting the fact that π is a feasible schedule of our problem. This completes the proof of Claim 4.

Claim 5 $b_{3r} \leq r$, for $1 \leq r \leq t$.

Otherwise, there is a certain r with $1 \leq r \leq t-1$, such that $b_{3r} \geq r+1$. By Claim 3, the operations in $\mathcal{B}_i$ with $1 \leq i \leq 3t$ and the operation $(3t+1, 1)$ are processed on or before the time $C_1(\pi)$. By Claim 4 and by the fact $b_i \leq b_{i+1}$ for $1 \leq i \leq 3t-1$, we have

$$b_{3k-2}, b_{3k-1}, b_{3k} \geq k, \text{ for } 1 \leq k \leq t.$$

Since each operation in $\mathcal{B}_1 \cup \mathcal{B}_2 \cup ... \cup \mathcal{B}_{3t}$ has a processing time greater than Z, the total processing time of the operations in $\mathcal{B}_1 \cup \mathcal{B}_2 \cup ... \cup \mathcal{B}_{3t}$ is greater than $Z + \frac{3}{2}t(t+1)Z$. The setup time of each batch $\mathcal{B}_i$ with $1 \leq i \leq 3t$ is greater than Y. Both the setup time of the family $\mathcal{F}_{3t+1}$ and the processing time of the operation $(3t+1, 1)$ are X. Hence,

$$d_1 \geq C_1(\pi) > Z + \frac{3}{2}t(t+1)Z + 3tY + 2X.$$

This contradicts Observation 2 and completes the proof of Claim 5.

Combining Claim 4 and Claim 5 and by the fact $b_i \leq b_{i+1}$ for $1 \leq i \leq 3t-1$, we deduce

Claim 6 $b_{3r-2} = b_{3r-1} = b_{3r} = r$, for $1 \leq r \leq t$.

It is implied in Claim 6 that each family $\mathcal{F}_i$, $1 \leq i \leq 3(t-1)$, is divided into at least two batches under π.

Claim 7 Each family $\mathcal{F}_i$ with $1 \leq i \leq 3(t-1)$ is divided into just two batches under π.

Otherwise, there are at least $3t+3(t-1)+1 = 6t-2$ batches (under π), each of which consists of the operations in $\cup_{1\leq i\leq 3t}\mathcal{F}_i$, and so each of which has a setup time greater than Y. Then the makespan of π is

$$C_{\max}(\pi) > (3t-1)X + (6t-2)Y > d_t,$$

where the second inequality is obtained from Observation 2. This contradicts the fact that π is a feasible solution of our problem. The proof of Claim 7 is completed.

Set $\mathcal{A}_i = \mathcal{F}_i \setminus \mathcal{B}_i$, for $1 \leq i \leq 3(t-1)$. By Claim 2, Claim 6 and Claim 7, each family $\mathcal{F}_i$ with $1 \leq i \leq 3(t-1)$ is divided into two batches $\mathcal{B}_i$ and $\mathcal{A}_i$ under π, and each family

$\mathcal{F}_i$ with $3t-2 \le i \le 4t$ acts as a batch under π. Furthermore, from Claim 1(3) and Claim 6, we have

$$\mathcal{B}_i = \{(i,j) : 1 \le j \le \lceil \frac{i}{3} \rceil\}, \text{ for } 1 \le i \le 3t,$$

and

$$\mathcal{A}_i = \{(i,j) : \lceil \frac{i}{3} \rceil + 1 \le j \le t\}, \text{ for } 1 \le i \le 3(t-1).$$

Now, for $1 \le r \le t$, write $\alpha_r = a_{3r-2} + a_{3r-1} + a_{3r}$. Then $\sum_{r=1}^{t} \alpha_r = tB$. We establish the value of $\max_{1\le j\le r} C_j(\pi)$.

By Claim 3 and the above discussion, $\max_{1\le j\le r} C_j(\pi)$ is greater than or equal to the value obtained by summing up the processing times of the operations in

$$\cup_{1\le i\le 3t}\mathcal{B}_i \cup \cup_{1\le i\le 3(r-1)}\mathcal{A}_i \cup \cup_{1\le i\le r-1}\mathcal{F}_{3t+i},$$

the processing time of the operation $(3t+r, r)$ and the setup times of the batches in

$$\{\mathcal{B}_i : 1 \le i \le 3t\} \cup \{\mathcal{A}_i : 1 \le i \le 3(r-1)\} \cup \{\mathcal{F}_{3t+i} : 1 \le i \le r\}.$$

Hence,

$$\begin{aligned} & \max_{1\le j\le r} C_j(\pi) \\ \ge\ & (3r-1)X + 3(t+r-1)Y + \tfrac{3t(t+1)}{2}Z + \tfrac{3(r-1)(2t-r)}{2}Z \\ & +(t+2)\textstyle\sum_{k=1}^{r-1}\alpha_k + \sum_{k=r}^{t}(1+k)\alpha_k. \end{aligned}$$

Because $\max_{1\le j\le r} C_j(\pi) \le d_r$ and

$$\begin{aligned} d_r =\ & (3r-1)X + 3(t+r-1)Y + \tfrac{3t(t+1)}{2}Z + \tfrac{3(r-1)(2t-r)}{2}Z \\ & +(r-1)(t+2)B + \tfrac{(t-r+1)(t+r+2)}{2}B \\ =\ & (3r-1)X + 3(t+r-1)Y + \tfrac{3t(t+1)}{2}Z + \tfrac{3(r-1)(2t-r)}{2}Z \\ & +(t+2)\textstyle\sum_{k=1}^{r-1}B + \sum_{k=r}^{t}(1+k)B, \end{aligned}$$

it follows that

$$(t+2)\sum_{k=1}^{r-1}\alpha_k + \sum_{k=r}^{t}(1+k)\alpha_k \le (t+2)\sum_{k=1}^{r-1}B + \sum_{k=r}^{t}(1+k)B$$

holds for each r with $1 \le r \le t$. From the fact that $\sum_{k=1}^{t}\alpha_k = tB$, we deduce

$$\sum_{k=r}^{t}(t+1-k)\alpha_k \ge \sum_{k=r}^{t}(t+1-k)B, \text{ for } 1 \le r \le t,$$

or equivalently, we have t inequalities (I_r), $1 \le r \le t$, as follows:

$$(I_r): \quad \sum_{k=1}^{r} k\alpha_{t+1-k} \ge \sum_{k=1}^{r} kB = \frac{1}{2}r(r+1)B, \text{ for } 1 \le r \le t.$$

Set $\lambda_k = \frac{1}{k} - \frac{1}{k+1} = \frac{1}{k(k+1)}$ for $1 \le k \le t-1$, and set $\lambda_t = \frac{1}{t}$. It is easy to check that

$$\sum_{r=k}^{t} \lambda_r = \frac{1}{k}, \text{ for } 1 \le k \le t.$$

Because each λ_k is positive, the convex linear combination of the above t inequalities (I_r), $1 \le r \le t$, yields the following inequality $(*)$.

$$(*): \quad \sum_{r=1}^{t} \lambda_r \sum_{k=1}^{r} k\alpha_{t+1-k} \ge \sum_{r=1}^{t} \lambda_r \sum_{k=1}^{r} kB.$$

Now, the left hand side of the inequality $(*)$ is

$$\begin{aligned} & \textstyle\sum_{r=1}^{t} \lambda_r \sum_{k=1}^{r} k\alpha_{t+1-k} \\ = & \textstyle\sum_{k=1}^{t} (\sum_{r=k}^{t} \lambda_r) k\alpha_{t+1-k} \\ = & \textstyle\sum_{k=1}^{t} \alpha_{t+1-k} \\ = & \textstyle\sum_{r=1}^{t} \alpha_r. \end{aligned}$$

The right hand side of the inequality $(*)$ is

$$\sum_{r=1}^{t} \lambda_r \frac{r(r+1)}{2} B = \sum_{r=1}^{t-1} \frac{1}{2} B + \frac{t+1}{2} B = tB.$$

By the fact that $\sum_{r=1}^{t} \alpha_r = tB$, we deduce that equality always holds for the inequality $(*)$. Since the inequality $(*)$ is a convex linear combination of the t inequalities (I_r), $1 \le r \le t$, we deduce that equality always holds for each of the t inequalities (I_r), $1 \le r \le t$; that is,

$$\sum_{k=1}^{r} k\alpha_{t+1-k} = \sum_{k=1}^{r} kB, \text{ for } 1 \le r \le t.$$

Finally, we can trivially deduce

$$\alpha_r = B, \text{ for } 1 \le r \le t.$$

Hence, the 3-partition problem has a solution. The result follows. □

Acknowledgements

This research was supported in part by the Research Grant Council of Hong Kong under grant number PolyU 5191/01E. The last author was also supported by the National Natural Science Foundation of China and the Huo Ying Dong Education Foundation of China.

References

[1] J. Bruno and P. Downey, Complexity of task sequencing with deadlines, setup times and changeover costs, *SIAM Journal on Computing*, **7**(1978), 393-404.

[2] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the theory of NP-Completeness*, Freeman, San Francisco, CA, 1979.

[3] A.E. Gerodimos, C.A. Glass, C.N. Potts and T. Tautenhahn, Scheduling multi-operation jobs on a single machine, *Annals of Operations Research*, **92**(1999), 87-105.

[4] J.B. Ghosh and J.N.D. Gupta, Batch scheduling to minimize maximum lateness, *Operations Research Letters*, **21**(1997), 77-80.

[5] J.K. Lenstra, A.H.G. Rinnooy Kan and P. Brucker, Complexity of machine scheduling problems, *Annals of Discrete Mathematics,* **1**(1977), 343-362.

[6] C.L. Monma and C.N. Potts, On the complexity of scheduling with batch setup times, *Operations Research*, **37**(1988), 798-804.