Lin, Meng-Ye; Kuo, Yarlin

**Article**

# Efficient mixed integer programming models for family scheduling problems

Operations Research Perspectives

**Provided in Cooperation with:**
Elsevier

*Suggested Citation:* Lin, Meng-Ye; Kuo, Yarlin (2017) : Efficient mixed integer programming models for family scheduling problems, Operations Research Perspectives, ISSN 2214-7160, Elsevier, Amsterdam, Vol. 4, pp. 49-55, https://doi.org/10.1016/j.orp.2017.03.001

This Version is available at:
https://hdl.handle.net/10419/178280

Operations Research Perspectives 4 (2017) 49–55

Contents lists available at ScienceDirect

Operations Research Perspectives

journal homepage: www.elsevier.com/locate/orp

# Efficient mixed integer programming models for family scheduling problems

Meng-Ye Lin, Yarlin Kuo*

*Department of Industrial Engineering and Management, Yunlin University of Science and Technology, Yunlin, Taiwan*

ARTICLE INFO

*Article history:*
Received 9 November 2016
Revised 3 March 2017
Accepted 15 March 2017
Available online 23 March 2017

*Keywords:*
Family scheduling
Sequence independent setup
Total weighted completion time
Maximum lateness

ABSTRACT

This paper proposes several mixed integer programming models which incorporate optimal sequence properties into the models, to solve single machine family scheduling problems. The objectives are total weighted completion time and maximum lateness, respectively. Experiment results indicate that there are remarkable improvements in computational efficiency when optimal sequence properties are included in the models. For the total weighted completion time problems, the best model solves all of the problems up to 30-jobs within 5 s, all 50-job problems within 4 min and about 1/3 of the 75-job to 100-job problems within 1 h. For maximum lateness problems, the best model solves almost all the problems up to 30-jobs within 11 min and around half of the 50-job to 100-job problems within 1 h.



## 1. Introduction

Batching jobs that share the same setup on a machine to increase productivity is a common practice in manufacturing. Such batching of jobs is often classified as a family in scheduling. When processing two jobs belonging to different families consecutively, a setup is required between them. The batching operation often leads to job lateness and results in poor delivery performance. How to trade-off productivity and delivery performance is a tricky scheduling problem. This problem is called batch/family/group scheduling with setups in the literature. For past research on family scheduling see Refs. [12,16,23].

The solution approaches to solve family scheduling problems include mathematical programming [1,3,8,13], branch and bound [7,18,19], dynamic programming [6,9,11,17] and heuristic/meta heuristics [4,5,14,21,24,25]. The last approach, heuristic/meta heuristics, is often designed to find good solutions for large scale problems. Except for mathematical programming, time consuming and complicated coding is required to find optimal/good sequences. On the other hand, commercial solvers such as LINGO and CPLEX are available to solve mathematical models. Thus, the start-up task for mathematical programming is much less than those of the other three solution approaches. Often the mathematical programming approach is applied to find an optimal schedule with limited CPU time, say 1 h, and is often used to solve small size problems. Thus, the mathematical programming approach is applicable for small-size enterprise scheduling problems or as a basis for rolling-horizon based scheduling techniques.

Webster and Baker [23] give an overview of optimal properties on family scheduling on a single machine. Very often these optimal properties are included in the branch and bound, dynamic programming and heuristic solution approaches to achieve better computational efficiency. In this study, we include some optimal properties of the family scheduling problem into mathematical programming models and investigate the effects of this inclusion on computational efficiency. In solving scheduling problems with mathematical programming, there are several well-known mixed integer programming (MIP) formulations proposed in the literature, which include: (1) the disjunctive formulation developed by Manne [10] which contains precedence variables that define the precedence order of any two jobs and disjunctive constraints that relate the completion times between any two jobs (2) the time indexed formulation proposed by Sousa and Wolsey [20] which defines time variables that relate jobs to the corresponding processing starting times in a finite discrete time horizon, (3) the linear ordering formulation developed by Potts [15] which adds triangle inequalities among the precedence variables of any three jobs and (4) the sequence position formulation developed by Wagner [22] which contains sequence position variables that relate jobs to corresponding positions in a sequence. For the single machine scheduling problem with release dates and sequence dependent setup and the objectives of total weighted tardiness and total weighted completion time, respectively, Nogueira and Carvalho (2014) compares the performance of six MIP formulations,

* Corresponding author.
*E-mail addresses:* M10221311@yuntech.edu.tw (M.-Y. Lin), kuoyl@yuntech.edu.tw (Y. Kuo).

http://dx.doi.org/10.1016/j.orp.2017.03.001

which include the four formulations above and two improved formulations and finds that the disjunctive formulation solves a greater number of problems.

In this study, we adopt the mathematical programming approach to find the optimal sequence of a single machine family scheduling problem with family sequence independent setup time and the objectives of total weighted completion time (TWC) and maximum lateness ($L_{\max}$), respectively. Three MIP formulations which include the optimal sequence properties are proposed. The performance of the proposed three MIP formulations is compared with two MIP models that do without the optimal sequence properties by computational experiments under different operating scenarios. The rest of this paper is organized as follows. Section 2 gives problem statements and presents MIP formulations. Section 3 gives the computational results and the paper concludes in Section 4.

## 2. Problem statements and MIP formulations

Consider the problem of scheduling N jobs belonging to F families on a single machine. A family sequence independent setup is required when a machine switches from idle to busy and from processing jobs in one family to jobs in another family. The objectives are to minimize total weighted completion time (TWC) and maximum lateness, respectively. Before proposing the MIP models, we give two optimal sequence properties of the family scheduling problems first.

**Bruno and Sethi's optimal property:** For the TWC problems, there is an optimal sequence in which jobs in the same family are ordered by SWPT (shortest weighted processing time first) [2].

**Monma and Potts's optimal property:** For the $L_{\max}$ problems, there is an optimal sequence in which jobs in the same family are ordered by EDD (earliest due date first) [11].

We include these two optimal properties in the constraints of MIP formulations and investigate the effects of these inclusions. Five MIP formulations are proposed to solve this problem, they are (1) family linear ordering formulation, FLO, (2) FLO with jobs in the same family sequenced in SWPT for TWC problems, $\text{FLO}_{\text{swpt}}$, and in EDD for $L_{\max}$ problems, $\text{FLO}_{\text{edd}}$, (3) ordered linear ordering formulation, OLO, where jobs in the same family are sequenced in SWPT for TWC problems and in EDD for $L_{\max}$ problems, (4) disjunctive formulation, DJ and (5) DJ with jobs in same family sequenced in SWPT for TWC problems, $\text{DJ}_{\text{swpt}}$, and in EDD for $L_{\max}$ problems, $\text{DJ}_{\text{edd}}$. The FLO and DJ formulations are proposed for the purpose of comparison. The notation and parameters used in the MIP formulations are as follows.

**Notation and parameters**

$a_{(i,j)}$: job $j$ of family $i$.
$F$: set of all families.
$N$: set of all jobs.
$s_i$: setup time of family $i$.
$s_{(i,k)}$: setup time from family $i$ to family $k$, $s_{(i,k)} = s_k$, $s_{(i,k)} = 0$ if $i = k$.
$p_{(i,j)}$: processing time of $a_{(i,j)}$.
$w_{(i,j)}$: weight of $a_{(i,j)}$.
$d_{(i,j)}$: due date of $a_{(i,j)}$.
$f$: total number of families.
$n_i$: total number of jobs in family $i$.
$n$: total number of jobs.
$M$: a big number.

**Decision variables**

$$\delta_{(i,j)(k,l)} = \begin{cases} 1 \text{ if } a_{(i,j)} \text{ is scheduled before} a_{(k,l)}. \\ 0 \text{ otherwise.} \end{cases}$$

$$x^i_{j'j} = \begin{cases} 1 \text{ if } a_{(i,j')} \text{ is } \textit{scheduled immediately before } a_{(i,j)}. \\ 0 \text{ otherwise.} \end{cases}$$

$$y_{(i,j)} = \begin{cases} 1 \text{ if a setup occurs immediately before } a_{(i,j)} \\ 0 \text{ otherwise.} \end{cases}$$

$$z_{(i,j)(k,l)} = \begin{cases} 1 \text{ if } a_{(i,j)} \text{ is } \textit{scheduled before } a_{(k,l)} \textit{ and} \text{ a setup} \\ \quad s_i \text{ occurs immediately before } a_{(i,j)}. \\ 0 \text{ otherwise.} \end{cases}$$

$C_{(i,j)}$ : completion time of $a_{(i,j)}$.

### 2.1. FLO formulation

The linear ordering formulation is to solve a single machine scheduling problem without setups. The problem studied here is a family scheduling problem with setups, and thus a family linear ordering formulation, FLO, is proposed. In the FLO formulation, jobs in each family are arbitrarily numbered. The constraints of FLO are:

$$\delta_{(i,j)(k,l)} + \delta_{(k,l)(i,j)} = 1 \ \forall (i,j), (k,l) \in N, \ (i,j) \neq (k,l) \tag{A1}$$

$$\delta_{(i,j)(k,l)} + \delta_{(k,l)(o,p)} + \delta_{(o,p)(i,j)} \leq 2$$
$$\forall (i,j), (k,l), (o,p) \in N \quad \text{and} \quad (i,j) \neq (k,l) \neq (o,p) \tag{A2}$$

$$\left(1 - x^i_{j'j}\right) \leq \left(\sum_{k=1}^{f}\sum_{l=1}^{n_k} \delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o} \delta_{(o,p)(i,j')} - 1\right) + M_1\left(1 - \delta_{(i,j')(i,j)}\right) \quad \forall (i,j), \left(i,j'\right) \in N, j < j' \tag{A3}$$

$$\left(\sum_{k=1}^{f}\sum_{l=1}^{n_k} \delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o} \delta_{(o,p)(i,j')} - 1\right) \leq M_2\left(1 - x^i_{j'j}\right) \quad \forall (i,j), \left(i,j'\right) \in N, j < j' \tag{A4}$$

$$x^i_{j'j} \leq \delta_{(i,j')(i,j)} \quad \forall (i,j), \left(i,j'\right) \in N, j < j' \tag{A5}$$

$$y_{(i,j)} = 1 - \sum_{j'=1}^{n_i} x^i_{j'j} \quad \forall (i,j) \in N \tag{A6}$$

$$z_{(i,j)(k,l)} \geq \delta_{(i,j)(k,l)} + y_{(i,j)} - 1 \quad \forall (i,j), \ (k,l) \in N \tag{A7}$$

$$C_{(i,j)} = \sum_{k=1}^{f}\sum_{l=1}^{n_k} \left(p_{(k,l)}\delta_{(k,l)(i,j)} + s_k z_{(k,l)(i,j)}\right) + p_{(i,j)} + s_i y_{(i,j)} \quad \forall (i,j) \in N \tag{A8}$$

$$C_{(i,j)} \geq 0 \quad \forall (i,j) \in N \tag{A9}$$

$$\delta_{(i,j)(i,j)} = 0 \quad \forall (i,j) \in N \tag{A10}$$

$$\delta_{(i,j)(k,l)} \in \{0,1\} \quad \forall (i,j),\ (k,l) \in N \tag{A11}$$

$$x^{i}_{j'j} \in \{0,1\} \quad \forall (i,j),\ (i,j') \in N \tag{A12}$$

$$y_{(i,j)} \in \{0,1\} \quad \forall (i,j) \in N \tag{A13}$$

$$z_{(i,j)(k,l)} \in \{0,1\} \quad \forall (i,j),\ (k,l) \in N \tag{A14}$$

(A1) and (A2) are the original constraints proposed by Potts [15]. (A1) ensures that $a_{(i,j)}$ is scheduled before $a_{(k,l)}$ or $a_{(k,l)}$ is scheduled before$a_{(i,j)}$. (A2) is the transitivity relation among any three jobs. (A3, A4) and (A5) determine the value of $x^{i}_{j'j}$ for any two jobs, $a_{(i,j')}$ and $a_{(i,j)}$ in family $i$: If $a_{(i,j')}$ is scheduled immediately before $a_{(i,j)}$ $x^{i}_{j'j} = 1$, otherwise $x^{i}_{j'j} = 0$. $(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1)$ calculates the number of jobs scheduled between $a_{(i,j')}$ and $a_{(i,j)}$. When $a_{(i,j')}$ is scheduled before $a_{(i,j)}$ $(\delta_{(i,j')(i,j)} = 1)$, there are two possible conditions:

I $a_{(i,j')}$ and $a_{(i,j)}$ are processed consecutively.
II $a_{(i,j')}$ and $a_{(i,j)}$ are processed inconsecutively.

If condition I is satisfied, $(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1) = 0$, and (A3, A4) and (A5) are simplified to:

$$\left(1 - x^{i}_{j'j}\right) \le 0 \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A3a}$$

$$0 \le M_2\left(1 - x^{i}_{j'j}\right) \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A4a}$$

$$x^{i}_{j'j} \le 1 \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A5a}$$

Thus (A3) leads to $x^{i}_{j'j} = 1$. If condition II is satisfied, $0 < (\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1) \le n-2$, and (A3, A4) and (A5) are simplified to:

$$\left(1 - x^{i}_{j'j}\right) \le \left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1\right) \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A3b}$$

$$\left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1\right) \le M_2\left(1 - x^{i}_{j'j}\right) \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A4b}$$

$$x^{i}_{j'j} \le 1 \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A5b}$$

Thus (A4) leads to $x^{i}_{j'j} = 0$. In order not to violate (A4), $M_2 \ge n-2$. When $a_{(i,j')}$ is scheduled after $a_{(i,j)}$, $\delta_{(i,j')(i,j)}$=0 and (A3, A4) and (A5) are simplified to:

$$\left(1 - x^{i}_{j'j}\right) \le \left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1\right) + M_1 \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A3c}$$

$$\left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1\right) \le M_2\left(1 - x^{i}_{j'j}\right) \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A4c}$$

**Table 1**
Value of $x^{i}_{j'j}$ with respect to relation of $a_{(i,j')}$ and $a_{(i,j)}$.

| | $x^{i}_{j'j}$=1 | $x^{i}_{j'j}$=0 |
|---|---|---|
| $\delta_{(i,j')(i,j)} = 1$ | Consecutive $a_{(i,j')}$, $a_{(i,j)}$ | Inconsecutive $a_{(i,j')}$, $a_{(i,j)}$ |
| $\delta_{(i,j')(i,j)} = 0$ | Impossible | No constraint is imposed |

$$x^{i}_{j'j} \le 0 \quad \forall (i,j),\ (i,j') \in N,\ j < j' \tag{A5c}$$

Thus, (A5) leads to $x^{i}_{j'j} = 0$. Because $-n \le (\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j')} - 1) < 0$, in order not to violate (A3) and (A5), $M_1 \ge n+1$ needs to be added to the right-hand side of (A3). Table 1 summarizes possible values of $x^{i}_{j'j}$ under different $a_{(i,j')}$ and $a_{(i,j)}$ relations.

(A6) examines whether the job scheduled immediately before $a_{(i,j)}$ belongs to family $i$ or not. If this job is from family $i$ then $y_{(i,j)} = 1$, otherwise $y_{(i,j)} = 0$. (A7) finds the value of $z_{(i,j)(k,l)}$. If $\delta_{(i,j)(k,l)}$=1 and $y_{(i,j)}$=1, then $z_{(i,j)(k,l)} = 1$, otherwise the minimization of $C_{(i,j)}$ related objective function will make $z_{(i,j)(k,l)} = 0$. (A8) finds the completion time of $a_{(i,j)}$ which is the sum of the processing times of all the jobs scheduled before $a_{(i,j)}$ plus its processing time and all needed setup times. (A9) ensures that the completion times are positive. (A10) is trivial. (A11, A12, A13) and (A14) are integrality constraints.

For the TWC problems, the objective is minimizing $\Sigma w_{(i,j)}C_{(i,j)}$, and constraints (A1)-(A14) are required. For the maximum lateness problems, besides (A1)-(A14), the following constraints (A15) and (A16) are required, the objective is minimizing $L_{\max}$.

$$L_{\max} \ge C_{(i,j)} - d_{(i,j)} \quad \forall (i,j) \in N \tag{A15}$$

$$L_{\max} \ge 0 \tag{A16}$$

### 2.2. $FLO_{swpt}/FLO_{edd}$ formulation

To include Bruno and Sethi's optimal property and Monma and Potts' optimal property into the FLO formulation of TWC problems and $L_{\max}$ problems, respectively, jobs within each family are numbered in SWPT for the TWC problems, and in EDD for the $L_{\max}$ problems first and then the following constraints are added to the FLO formulation to derive the $FLO_{swpt}/FLO_{edd}$ formulation.

$$\delta_{(i,j)(i,j+1)} = 1 \quad \forall (i,j) \in N,$$
$$i = 1,2,3,\ldots f \text{ and } j = 1,2,3,..(n_i - 1)$$

### 2.3. OLO formulation

If jobs in each family are numbered in SWPT order for the TWC problems and in EDD order for the $L_{\max}$ problems, variables $x^{i}_{j'j}$ can be deleted. We can determine whether a family $i$ setup is required or not based on whether there are jobs scheduled between $a_{(i,j-1)}$ and $a_{(i,j)}$ or not. If there are jobs scheduled between $a_{(i,j-1)}$ and $a_{(i,j)}$, a family $i$ setup is needed. Replace (A3)-(A6) of the FLO formulation with (B3)-(B6) below and we have the OLO formulation.

$$\delta_{(i,j)(i,j+1)} = 1 \quad \forall (i,j) \in N,\ i = 1,2,3,\ldots f \text{ and } j = 1,2,3,..(n_i - 1) \tag{B3}$$

$$y_{(i,1)} = 1 \quad \forall i \in F,\ i = 1,2,3,\ldots f \tag{B4}$$

$$y_{(i,j)} \le \left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)} - \sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j-1)} - 1\right) \quad \forall (i,j) \in N \tag{B5}$$

$$\left(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)}-\sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j-1)}-1\right)\le M_3 y_{(i,j)}\quad \forall(i,j)\in N \tag{B6}$$

(B3) ensures that jobs within each family are processed in SWPT for the TWC problems and in EDD for the $L_{\max}$ problems. (B4) ensures that a setup is implemented before processing the first job within each family. In (B5), $0\le(\sum_{k=1}^{f}\sum_{l=1}^{n_k}\delta_{(k,l)(i,j)}-\sum_{o=1}^{f}\sum_{p=1}^{n_o}\delta_{(o,p)(i,j-1)}-1)\le n-n_i$ finds the number of jobs scheduled between $a_{(i,j-1)}$ and $a_{(i,j)}$. (B5) and (B6) determine the value of $y_{(i,j)}$. If there are jobs scheduled between $a_{(i,j-1)}$ and $a_{(i,j)}$, $y_{(i,j)}=1$, otherwise $y_{(i,j)}=0$. $M_3\ge n-n_i$ is required in order not to violate (B6).

### 2.4. DJ formulation constraints

For the DJ formulation, jobs within each family are arbitrarily numbered. The following formulation is adapted from Manne (1989) which solves a scheduling problem with family sequence dependent setups, thus, we change family $k$'s setup time $s_k$ to $s_{(i,k)}=s_k$ for all $i$:

$$C_{(k,l)}\ge C_{(i,j)}+s_{(i,k)}+p_{(k,l)}-M_4\left(1-\delta_{(i,j)(k,l)}\right)\quad \forall(i,j),(k,l)\in N \text{ and } (i,j)\ne(k,l) \tag{C1}$$

$$\delta_{(i,j)(k,l)}+\delta_{(k,l)_{(i,j)}}=1\quad \forall(i,j),(k,l)\in N \text{ and } (i,j)\ne(k,l) \tag{C2}$$

$$C_{(k,l)}\ge p_{(k,l)}+s_{(0,k)}\quad \forall(k,l)\in N \tag{C3}$$

$$C_{(k,l)}\ge 0\quad \forall(k,l)\in N \tag{C4}$$

$$\delta_{(i,j)(k,l)}\in\{0,1\}\quad \forall(i,j),(k,l)\in N \text{ and } (i,j)\ne(k,l) \tag{C5}$$

(C1) ensures that the completion time of $a_{(k,l)}$ is greater or equal to the sum of the completion time of $a_{(i,j)}$, the required setup time $s_{(i,k)}=s_k$ (from family $i$ to family $k$) and the processing time of $a_{(k,l)}$ if $a_{(i,j)}$ is scheduled before $a_{(k,l)}$. $M_4=\sum_{(i,j)}p_{(i,j)}+n(\max_i(s_i))$ [13]. (C2) ensures that $a_{(i,j)}$ is scheduled before $a_{(k,l)}$ or $a_{(k,l)}$ is scheduled before $a_{(i,j)}$. (C3) ensures that the completion time of $a_{(k,l)}$ is greater or equal to its processing time plus its initial setup time $s_{(0,k)}$. (C4) ensures that the completion times are positive. (C5) is the integrality constraints. For the TWC problem, (C1)-(C5) are necessary and for the $L_{\max}$ problem, two additional (C6) and (C7) are needed:

$$L_{\max}\ge C_{(i,j)}-d_{(i,j)}\quad \forall(i,j)\in N \tag{C6}$$

$$L_{\max}\ge 0 \tag{C7}$$

### 2.5. $DJ_{swpt}/DJ_{edd}$ formulation

The $DJ_{swpt}/DJ_{edd}$ formulation incorporates the optimal sequence properties into the DJ formulation. To apply the $DJ_{swpt}/DJ_{edd}$ formulation, jobs within each family are numbered in SWPT for the TWC problems and in EDD for the $L_{\max}$ problems. The following constraints are added to the DJ formulation.

$$\delta_{(i,j)(i,j+1)}=1\quad \forall(i,j)\in N,\ i=1,2,3,\ldots f \text{ and } j=1,2,3,..(n_i-1)$$

**Table 2**
Parameters generations.

| Input data | Value |
|---|---|
| Processing time | *Uniform*(1, $\alpha_1$50) |
| Setup time | *Uniform*(1, $\alpha_2$10) |
| Weight | *Uniform*(1, $n$) |
| Due date | *Uniform*($\min_{(i,j)}(p_{(i,j)})$, $\frac{2h'}{\alpha_3}$) |
| Number of families | *Uniform*(2, *min*($n$, $\alpha_4$4)) |

## 3. Computational experiments and results

To compare the performance of the five MIP formulations proposed, we designed the parameter selection similar to those of Nogueira et al. [13] to generate six operating scenarios (classes). The values of the parameters are all integral and generated according to Table 2
where $h'=\sum_{(i,j)}p_{(i,j)}+n(\max_i(s_i))$ and $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$ are scale parameters. $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$ determine the six operation classes/scenarios. The minimum values of $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$ are all 1 and the maximum values are all 4 except $\alpha_2$ which has a maximum value 10. These six classes are

Class 1: all scale parameters have minimum values; (a basic class)
Class 2: $\alpha_1$ has maximum value (4) and the others have minimum values; (a large processing time class)
Class 3: $\alpha_2$ has maximum value (10) and the others have minimum values; (a large setup time class)
Class 4: $\alpha_3$ has maximum value (4) and the others have minimum values; (a tight due date class)
Class 5: $\alpha_4$ has maximum value (4) and the others have minimum values; (a large number of families class)
Class 6: all scale parameters have maximum values (a complex class).

Class 4 investigating the due date effects is not applicable to TWC problems. 20 independent instances are randomly generated for each class and each job size $n\in\{10, 20, 30, 50, 75, 100\}$. Thus, there are 720 instances for $L_{\max}$ problems and 600 instances for TWC problems.

All MIP formulations are coded with Ilog Cplex 12.6 with the default setting and run on a computer with a 2.9 Ghz processor and 16 GB memory. Each instance is run for 1 h and its optimality gap is recorded.

### 3.1. Computational results for TWC problems

For the TWC problems, based on the mean optimality gap (Fig. 1), the performance of OLO is the best among five MIP models for each class. The next one is $FLO_{swpt}$ followed by FLO and DJ is the worst.

Table 3 gives the mean CPU time per solved problem, and Table 4 gives the number of solved problems for the TWC problems. Based on Tables 3 and 4, OLO has the best performance, which solves all problems up to 30-jobs within 5 s, and solves all 50-job problems within 200 s. The second best model is $FLO_{swpt}$, which solves all problems up to 30-jobs within 15 s. The worst model is DJ. Considering problems of all classes and all job-sizes, FLO and DJ solve 34.17% and 16.67% of problems, respectively and those of $FLO_{swpt}$, $DJ_{swpt}$ and OLO are 64.83, 34.33% and 77.83%, respectively. Furthermore, the mean CPU times per solved problem for $FLO_{swpt}$, $DJ_{swpt}$ and OLO are much better than those of FLO and DJ, respectively. It is clear that the performances of $FLO_{swpt}$, $DJ_{swpt}$ and OLO are much better than those of FLO and DJ, respectively.

In conclusion, for TWC problems when Bruno and Sethi's optimal property is included into MIP models, there is remarkable

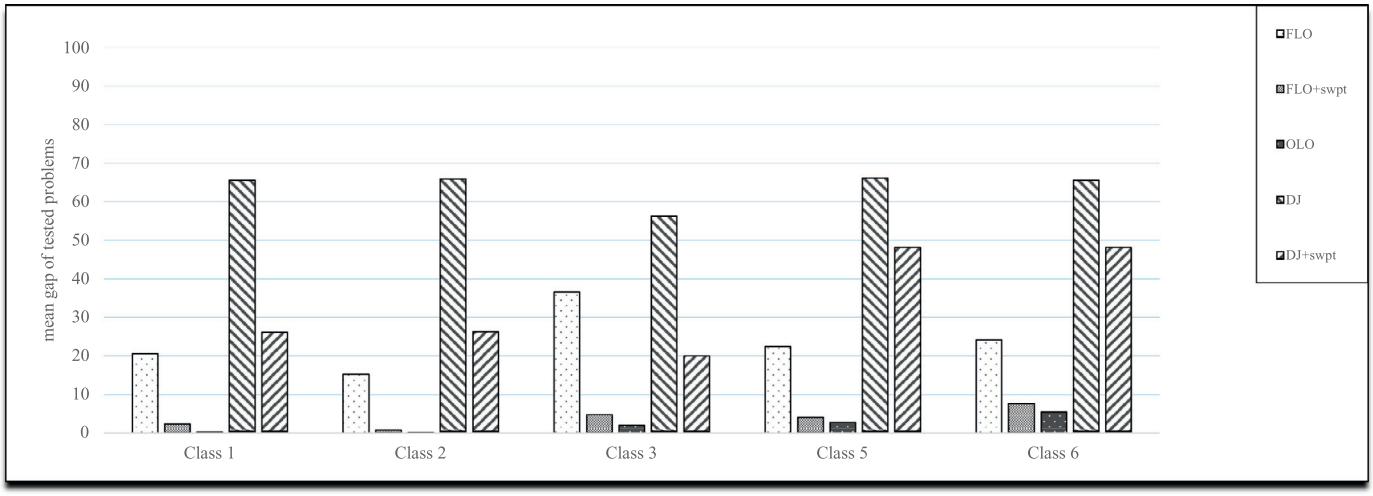


**Fig. 1.** Mean optimality gap% for TWC problems.

**Table 3**
Mean CPU time per solved problem for TWC.

| mean CPU | Class 1: basic class | | | | | Class 2: large processing time | | | | | Class 3: large setup time | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| number of jobs | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT |
| 10 | 0.39 | 0.07 | 0.13 | 5.14 | 0.32 | 0.15 | 0.04 | 0.09 | 5.27 | 0.21 | 1.85 | 0.11 | 0.07 | 1.03 | 0.07 |
| 20 | 609.84 | 0.69 | 0.33 | – | 213.55 | 26.92 | 0.44 | 0.32 | – | 704.24 | 1725.8 | 0.68 | 0.21 | – | 24.83 |
| 30 | – | 9.77 | 2.4 | – | 223.36 | 1417.1 | 4.03 | 1.28 | – | 1012.96 | – | 8.88 | 1.84 | – | 809.94 |
| 50 | – | 730.05 | 102.94 | – | – | – | 712.1 | 36.88 | – | – | – | 675.23 | 130.09 | – | 538.17 |
| 75 | – | – | 1110.46 | – | – | – | 1202.46 | 687.65 | – | – | – | 1681.6 | 1348.53 | – | 1725.1 |
| 100 | – | – | 2696.27 | – | – | – | – | 1482.75 | – | – | – | – | 2417.4 | – | – |

| mean CPU | Class 5: large number of families | | | | | Class 6: most complex | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| number of jobs | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT |
| 10 | 0.19 | 0.02 | 0.07 | 7.01 | 0.81 | 0.25 | 0.08 | 0.05 | 3.88 | 1.12 |
| 20 | 108.2 | 0.48 | 0.41 | – | 289.19 | 208.64 | 0.48 | 0.35 | – | 1167.9 |
| 30 | 259.62 | 6.53 | 2.48 | – | 126.91 | 895.86 | 11.41 | 4.06 | – | 834.41 |
| 50 | – | 714.1 | 117.02 | – | – | – | 474.76 | 196.27 | – | – |
| 75 | – | 3109.3 | 1367.41 | – | – | – | – | 2223.62 | – | – |
| 100 | – | – | 2716.68 | – | – | – | – | – | – | – |

**Table 4**
Number of solved problems for TWC.

| # solved | Class 1: basic class | | | | | Class 2: large processing time | | | | | Class 3: large setup time | | | | | Class 5: large number of families | | | | | Class 6: most complex | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| number of jobs | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT | FLO | FLO SWPT | OLO | DJ | DJ SWPT |
| 10 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 20 | 18 | 20 | 20 | 0 | 16 | 20 | 20 | 20 | 0 | 17 | 3 | 20 | 20 | 0 | 20 | 20 | 20 | 20 | 0 | 5 | 19 | 20 | 20 | 0 | 5 |
| 30 | 0 | 20 | 20 | 0 | 6 | 8 | 20 | 20 | 0 | 9 | 0 | 20 | 20 | 0 | 19 | 10 | 20 | 20 | 0 | 2 | 7 | 20 | 20 | 0 | 3 |
| 50 | 0 | 9 | 20 | 0 | 0 | 0 | 17 | 20 | 0 | 0 | 0 | 6 | 20 | 0 | 3 | 0 | 16 | 20 | 0 | 0 | 0 | 15 | 20 | 0 | 0 |
| 75 | 0 | 0 | 10 | 0 | 0 | 0 | 4 | 17 | 0 | 0 | 0 | 3 | 10 | 0 | 1 | 0 | 1 | 4 | 0 | 0 | 0 | 0 | 7 | 0 | 0 |
| 100 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| sum | 38 | 69 | 94 | 20 | 42 | 48 | 81 | 106 | 20 | 46 | 23 | 69 | 95 | 20 | 63 | 50 | 77 | 85 | 20 | 27 | 46 | 75 | 87 | 20 | 28 |

improvement in computational efficiency. OLO has the best performance and DJ has the worst performance among the five MIP models with respect to both the number of solved problems, mean CPU time per solved problem and mean optimality gap for all classes and all job-sizes. OLO solves all problems up to 50-jobs and around 50% of 75-job problems and near 20% of 100-job problems under various operating scenarios. On the easiness of problem solving in different classes/scenarios, class 2, the large processing times scenario, problems are the easiest to solve for all formulations.

### 3.2. Computational results for $L_{max}$ problems

For the $L_{\max}$ problem, with the inclusion of Monma and Potts' optimal property into MIP models, based on the mean optimality gap (Fig. 2), for all classes the performance of $FLO_{edd}$ and OLO are better than that of FLO; and the performance of $DJ_{edd}$ is better than that of DJ.

Table 5 gives the mean CPU time per solved problem, and Table 6 gives the number of solved problems for the $L_{\max}$ problems. The best model is OLO again, but the performance of OLO for the $L_{\max}$ problems is worse than that of the TWC problems. For classes 1 to 4 problems, OLO solves all problems up to 30-jobs within 20 s. For class 5 problems OLO solves 93% of the problems up to 30-jobs within 3 min. On the number of solved problems (Table 6), considering problems of all classes and all job-sizes, FLO and DJ solve 39.83% and 50.33% of the problems, respectively and those of $FLO_{edd}$, $DJ_{edd}$ and OLO are 84.83, 70.83% and 87.83%, respectively. The performance of $FLO_{edd}$, $DJ_{edd}$ and OLO are better than those of FLO and DJ, respectively and these improvements are more profound than those of the TWC problems.

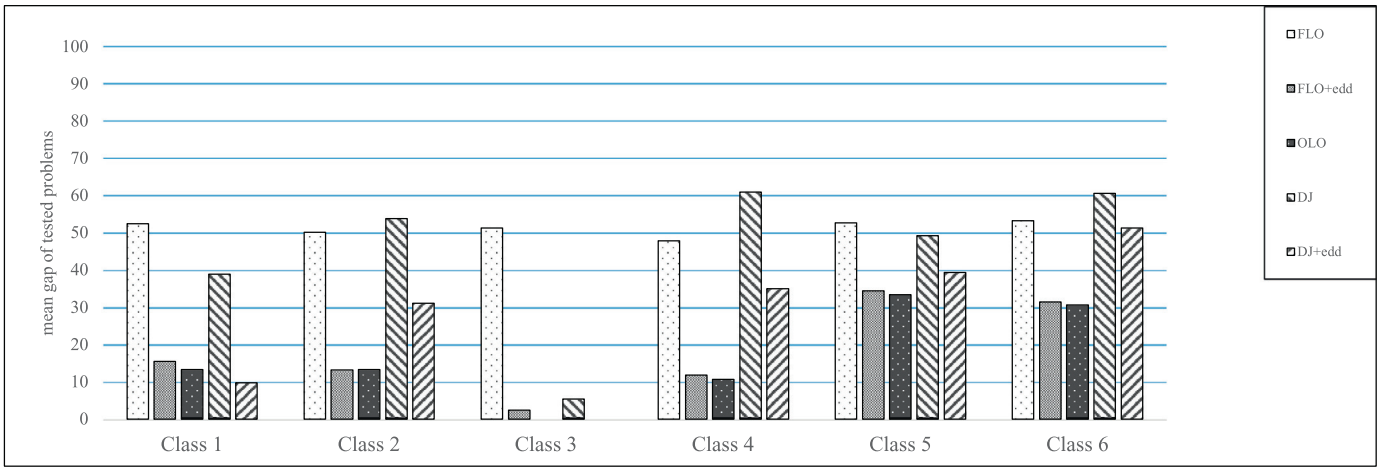


**Fig. 2.** Mean optimality gap % for $L_{max}$ problems.

**Table 5**
Mean CPU time per solved problem for $L_{max}$ problems.

| mean CPU | Class 1: basic class | | | | | Class 2: large processing time | | | | | Class 3: large setup time | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| job size | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD |
| 10 | 0.51 | 0.09 | 0.14 | 0.3 | 0.17 | 0.68 | 0.1 | 0.21 | 0.49 | 0.27 | 0.11 | 0.03 | 0.02 | 0.02 | 0.06 |
| 20 | 67.16 | 1.21 | 0.95 | 76.65 | 0.85 | 307.57 | 2.07 | 2.34 | 349.99 | 238.28 | 51.27 | 0.23 | 0.24 | 0.31 | 0.12 |
| 30 | 757.07 | 8.62 | 6.67 | 38.87 | 1.85 | 1108.59 | 29.73 | 16.95 | 342.26 | 111.94 | 179.77 | 2.66 | 1.45 | 1.13 | 0.28 |
| 50 | – | 167.08 | 157.07 | 344.83 | 121.19 | – | 655.33 | 730.79 | – | 73.06 | 1.05 | 22.21 | 24.03 | 19.62 | 0.89 |
| 75 | – | 130.42 | 389.1 | – | 24.37 | – | 72.48 | 285.34 | – | 21.64 | 3.3 | 393.9 | 210.51 | 567.27 | 3.14 |
| 100 | – | 856.79 | 251.21 | – | 124.84 | – | 331.58 | 897.75 | – | 241.89 | – | 449.43 | 542.56 | 550.54 | 1.97 |
| mean CPU | Class 4: small due dates | | | | | Class 5: large number of families | | | | | Class 6: most complex | | | | |
| job size | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD |
| 10 | 1.58 | 0.1 | 0.12 | 3.22 | 0.18 | 0.27 | 0.06 | 0.16 | 0.13 | 0.2 | 1.6 | 0.38 | 0.35 | 3.25 | 1.62 |
| 20 | – | 1.92 | 1.59 | – | 269.93 | 184.5 | 8.26 | 5.64 | 171.82 | 228.41 | 1553.13 | 175.44 | 301.03 | – | 856.53 |
| 30 | – | 26.19 | 18.82 | – | 644.27 | 1200.37 | 232.53 | 166.34 | 472.17 | 55.27 | – | 555.57 | 631.08 | – | 70.7 |
| 50 | – | 686.49 | 751 | – | – | – | 692.15 | 1163.08 | 589.59 | 132.94 | – | 10.22 | 25.26 | – | – |
| 75 | – | 773.18 | 482.36 | – | – | – | 20.56 | 289.76 | – | 63.27 | – | 2307.5 | 963.55 | – | – |
| 100 | – | 1867.66 | 1632.17 | – | – | – | 45.01 | 17.22 | – | 53.63 | – | – | – | – | – |

**Table 6**
Number of solved problems for $L_{max}$ problems.

| # solved | Class 1: basic class | | | | | Class 2: large processing time | | | | | Class 3: large setup time | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Number of Jobs | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD |
| 10 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 20 | 20 | 20 | 20 | 20 | 20 | 16 | 20 | 20 | 10 | 20 | 19 | 20 | 20 | 20 | 20 |
| 30 | 12 | 20 | 20 | 19 | 19 | 7 | 20 | 20 | 8 | 15 | 15 | 20 | 20 | 20 | 20 |
| 50 | 0 | 18 | 18 | 7 | 16 | 0 | 13 | 14 | 0 | 5 | 1 | 20 | 20 | 20 | 20 |
| 75 | 0 | 11 | 14 | 0 | 14 | 0 | 6 | 6 | 0 | 4 | 2 | 20 | 20 | 15 | 20 |
| 100 | 0 | 9 | 9 | 0 | 14 | 0 | 8 | 9 | 0 | 5 | 0 | 16 | 20 | 16 | 20 |
| sum | 52 | 98 | 101 | 66 | 103 | 43 | 87 | 89 | 38 | 69 | 57 | 116 | 120 | 111 | 120 |
| # solved | Class 4: small due dates | | | | | Class 5: large number of families | | | | | Class 6: most complex | | | | |
| Number of Jobs | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD | FLO | FLO EDD | OLO | DJ | DJ EDD |
| 10 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 20 | 0 | 20 | 20 | 0 | 20 | 20 | 20 | 20 | 16 | 17 | 1 | 20 | 20 | 0 | 8 |
| 30 | 0 | 20 | 20 | 0 | 6 | 6 | 16 | 16 | 8 | 9 | 0 | 11 | 12 | 0 | 3 |
| 50 | 0 | 17 | 19 | 0 | 0 | 0 | 5 | 8 | 3 | 5 | 0 | 1 | 1 | 0 | 0 |
| 75 | 0 | 7 | 8 | 0 | 0 | 0 | 2 | 3 | 0 | 3 | 0 | 3 | 4 | 0 | 0 |
| 100 | 0 | 5 | 5 | 0 | 0 | 0 | 1 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 |
| sum | 20 | 89 | 92 | 20 | 46 | 46 | 64 | 68 | 47 | 56 | 21 | 55 | 57 | 20 | 31 |

Comparatively, for the $L_{max}$ problems, the performance of OLO is the best among all formulations. OLO solves all problems up to 20-jobs, 90% of 30-job problems, around 50% of the 50-job, 75-job, and 100-job problems. Finally, on the easiness of problem solving in different operating classes (Table 6) for all the MIP models the large setup time problems (class 3) are easier to solve than other classes, the next one is the class 1 problems.

## 4. Conclusion

In this paper, we study the computational efficiency of including optimal sequence properties into MIP models for family scheduling problems. Two sets of MIP models are constructed: the first set contains FLO and DJ formulations that do without the optimal sequence properties and the second set contains

the $FLO_{swpt/edd}$, $DJ_{swpt/edd}$ and OLO models that include optimal sequence properties. Based on the experiment results we find that the computational efficiencies of the set of MIP models embedded with the optimal sequence properties are much better than those MIP models that do without optimal sequence properties in various operating scenarios. Among the five MIP models studied, the best model for the TWC problems is OLO, which solves all of the problems up to 50-jobs within 4 min and around 50% of the 75-job problems and near 20% of the 100-job problems within 1 h. For $L_{max}$ the best model is OLO again, which solves all problems up to 20-jobs, 90% of the 30-job problems within 11 min around 50% of the problems containing more than 50 jobs within 1 h. Thus for almost all small size problems and some medium size problems applying easy-to-construct improved MIP models such as OLO, practitioners can find the optimal job sequence within an hour.

## References


[1] Baker KR, Keller B. Solving the single-machine sequencing problem using integer programming. Comput Ind Eng 2010;59(4):730–5.
[2] Bruno J, Sethi R. Task sequencing in a batch environment with setup times. In: Proceedings of the international workshop organized by the commission of the European communities on modelling and performance evaluation of computer systems. North-Holland Publishing Co; 1976. p. 81–8.
[3] Costa A, Cappadonna FA, Fichera S. Joint optimization of a flow -shop group scheduling with sequence dependent set -up times and skilled workforce assignment. Int J Prod Res 2014a;52(9):2696–728.
[4] Costa A, Cappadonna FA, Fichera S. A hybrid metaheuristic approach for minimizing the total flow time in a flow shop sequence dependent group scheduling problem. Algorithms 2014b;7(3):376–96.
[5] Crauwels HAJ, Potts CN, Van Wassenhove LN. Local search heuristics for single machine scheduling with batch set-up times to minimize total weighted completion time. An Oper Res 1997;70:261–79.
[6] Ghosh JB, Gupta JN. Batch scheduling to minimize maximum lateness. Oper Res Lett 1997;21(2):77–80.
[7] Jordan C, Drexl A. Discrete lotsizing and scheduling by batch sequencing. Manage Sci 1998;44(5):698–713.
[8] Keha AB, Khowala K, Fowler JW. Mixed integer programming formulations for single machine scheduling problems. Comput Ind Eng 2009;56(1):357–67.
[9] Liao CJ, Liao LM. Single facility scheduling with major and minor setups. Comput Oper Res 1997;24(2):169–78.
[10] Manne AS. On the job-shop scheduling problem. Oper Res 1960;8(2):219–23.
[11] Monma CL, Potts CN. On the complexity of scheduling with batch setup times. Oper Res 1989;37(5):798–804.
[12] Neufeld JS, Gupta JND, Buscher U. A comprehensive review of flowshop group scheduling literature. Comput Oper Res 2016;70:56–74.
[13] Nogueira TH, de Carvalho CRV, Ravetti MG. Analysis of mixed integer programming formulations for single machine scheduling problems with sequence dependent setup times and release dates. Optimization Online 2014.
[14] Nowicki E, Zdrzałka S. Single machine scheduling with major and minor setup times: a tabu search approach. J Oper Res Soc 1996;47(8):1054–64.
[15] Potts CN. An algorithm for the single machine sequencing problem with precedence constraints. In: Combinatorial Optimization II. Springer; 1980. p. 78–87.
[16] Potts CN, Kovalyov MY. Scheduling with batching: a review. Eur J Oper Res 2000;120(2):228–49.
[17] Psaraftis HN. A dynamic programming approach for sequencing groups of identical jobs. Oper Res 1980;28(6):1347–59.
[18] Schaller JE, Gupta JN. Single machine scheduling with family setups to minimize total earliness and tardiness. Eur J Oper Res 2008;187(3):1050–68.
[19] Schutten JMJ, Van de Velde SL, Zijm WHM. Single-machine scheduling with release dates, due dates and family setup times. Manage Sci 1996;42(8):1165–74.
[20] Sousa JP, Wolsey LA. A time indexed formulation of non-preemptive single machine scheduling problems. Math Prog 1992;54(1-3):353–67.
[21] Uzsoy R, Velásquez JD. Heuristics for minimizing maximum lateness on a single machine with family-dependent set-up times. Comput Oper Res 2008;35(6):2018–33.
[22] Wagner HM. An integer linear-programming model for machine scheduling. Naval Res Logist Q 1959;6(2):131–40.
[23] Webster S, Baker KR. Scheduling groups of jobs on a single machine. Oper Res 1995;43(4):692–703.
[24] Williams D, Wirth A. A new heuristic for a single machine scheduling problem with set-up times. J Oper Res Soc 1996;47(1):175–80.
[25] Yin N, Kang L, Wang XY. Single-machine group scheduling with processing times dependent on position, starting time and allotted resource. Appl Math Model 2014;38(19):4602–13.