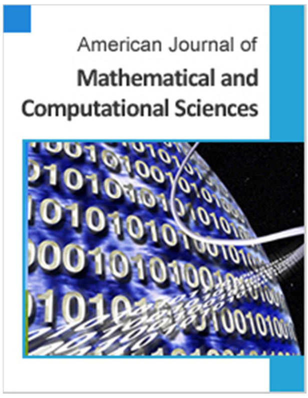
American Journal of Mathematical and Computational Sciences
2018; 3(1): 31-36
http://www.aascit.org/journal/ajmcs

American Association for Science and Technology



Mixed Integer Programming for Job Shop Scheduling Problem with Separable Sequence-Dependent Setup Times

Emna Driss[1, *], Racem Mallouli[2], Wafik Hachicha[3]

[1]Nord American University, International Institute of Technology, Sfax, Tunisia
[2]Umm Al-Qura University, Al-Qunfudhah Engineering College, Khalidiah, Saudi Arabia
[3]University of Sfax, Higher Institute of Management of Sfax, Sfax, Tunisia

Email address

driss.emna@iit-nau.com (E. Driss), racem.mellouli@yahoo.fr (R. Mallouli), wafik_hachicha@yahoo.fr (W. Hachicha)

[*]Corresponding author

Citation

Emna Driss, Racem Mallouli, Wafik Hachicha. Mixed Integer Programming for Job Shop Scheduling Problem with Separable Sequence-Dependent Setup Times. *American Journal of Mathematical and Computational Sciences.* Vol. 3, No. 1, 2018, pp. 31-36.

Keywords

Job Shop Problem,
Setup Times,
Makespan,
Mixed Integer Programming

Received: April 22, 2017
Accepted: October 26, 2017
Published: January 20, 2018

Abstract

In scheduling practice, the setup times are often important and no one can neglect them. That is why many research studies which deal with job shop problems (JSP) with separable sequence-dependent setup times (SDST) have been developed according to various objectives and hypotheses. Several complex programming formulations are proposed in the literature in order to minimize the makespan. Because of this complexity, there are few studies using exact approaches and the current trend is towards heuristics and metaheuristic based-methodologies. Therefore, it would be useful to develop a new model solvable by any optimizer directly. In this context, a new mixed integer programming (MIP) is developed through this paper for the case of SDST. This new model was designed to consider fewer variables and constraints than the existing formulations. Finally, a simple problem from the literature including four jobs and four machines is resolved in order to verify the model.

1. Introduction

Scheduling problems exist practically everywhere in the industrial word contexts. Production scheduling in the manufacturing system consists on determining the execution sequence of set of jobs on a set of machines over time in order to achieve some objectives under the constraints of production. Indeed, proper scheduling leads to increased efficiency and machine capacity utilization, reduced time required to complete operations, and, consequently, increased performance of an organization.

One of the most recognized pragmatic scheduling problems is the job shop problems which consist in programming the processing of n jobs (j=1,…, n) on m machines (k=1,…, m). Indeed, each job j consists on a sequence of m' operations or sub-jobs (O_{j1}, …, $O_{jm'}$ with $m' \leq m$) which must be run in this order, and each operation on a specific machine. Operations running orders and its specific processing machine are known in advance. The scheduling literature shows that the Job Shop Problems are NP-hard (Garey et al. [1]) and in the strong sense for specific conditions (Garey et al. [2]). That is why various approaches and methods are proposed in the last five decades to deal with these problems with their different variances.

One of the standard assumptions of the job shop problems is that setup times which have been considered negligible and consequently ignored, or considered as part of the processing times. While this possibly justified for some scheduling problems, many other situations call for separable setup time consideration. According to Allahverdi et al. [3] the literature is categorized on two types of separable setup problem. The first is called sequence-independent setup times problem because setup depends only on the job to be processed. In the second type, setup depends on both the job to be processed and the immediately preceding job, hence it is called sequence-dependent setup times problem (SDST). The separable SDST problem as the focus of this paper is more reasonable in real-life scheduling situation. Indeed, there are numerous industrial systems that serve as examples for which the amount of setup time varies considerably depending on the processing sequence of the jobs. These include the cutting and stitching machines in printing firm, the changing color in plastic injection process, the stamping operation in plastics manufacturing, die changing in a metal processing, *etc*.,

The separable SDST problem was addressed by many researchers in order to minimize the maximum completion time using either exact or approximate methods. Chen et al. [4] treated $J/ST_{sd}/Cmax$ and developed a computer-based methodology in order to integrate the scheduling algorithm benefits with the expert knowledge through a graphic and interactive program so as to find the optimal solution. Moghaddas et al. [5] also proposed a heuristic model to deal with large size problems based on priority rules. The heuristic performance indicates a strong ability to solve this problem in reasonable computational time. In order to solve $J/ST_{sd}/Cmax$ Artigues et al. [6] used branch and bound method to develop a new exact solution where they searched a relaxation of the problem attached to the traveling salesman problem with time windows. Branch and bound was involved to generate a feasible solution and this problem relaxation was involved to compute a first lower bound. Focacci et al. [7] equally developed two co-operative heuristics which lead to an optimal solution. Camino et al. [8] developed a hybrid algorithm to resolve SDST. They combined genetic algorithm with local research to obtain satisfactory result. Other searchers equally involve heuristic approaches in their research such as Cheung et al. [9], Choi et al. [10], Zhou et al. [11], Yang [12], Shen [13], and Allahverdi [14]. In order to solve flexible job shop problems with sequence-dependent setup times different metaheuristics are developed in many researches in the recent years such as Shen et al. [15], Knopp et al.[16], and Abdelmaguid [17].

The remainder of this paper is organized as follows. Section 2 describes the various existing MIP formulations for SDST. Section 3 proposes a new MIP formulation for the studied problem and gives a performance comparison with these existing formulations. Section 4 contains an illustrative example from the literature including four jobs and four machines. Finally, Section 5 concludes with brief remarks and perspectives.

2. Existing MIP Formulations

The most significant separable SDST problem assumptions are the following.

a. Each job is an entity: Although the job is composed of distinct operations, no two operations of the same job may be processed simultaneously.
b. No preemption: Each operation, once started, must be completed before another operation may be started on that machine.
c. No cancellation. Each job must be processed to completion.
d. The processing times are independent of the schedule.
e. The times to move jobs between machines are negligible.
f. In-process inventory is allowed, jobs may wait for their next machine to be free.
g. There is only one of each type of machine in the workshop. This assumption eliminates, amongst others, the case where certain machines have been duplicated to avoid bottlenecks.
h. Machine may be idle.
i. Machine capacity equal to 1: No machine may process more than one operation at a time (disjunctive case).
j. Machines never breakdown and are available throughout the scheduling period.
k. There is no randomness. The numbers of jobs, the number of machines, the processing times and the setup times are known and fixed.
l. No re-entrant flows are required; each job visits at most once each machine (a general case assumption).

Three MIP formulations are identified to be mentioned in literature overviews about separable SDST: the model of Moghaddas et al. [5], the model of Choi et al. [18] and the model of Chinyao et al. [19]. The objective is to propose a new formulation involving some subtlety that make the MIP model capable to outperform these three existing referential models which will be detailed in the remaining of this section.

2.1. Moghaddas et al.'s Model

Moghaddas et al. [5] proposed a mixed integer model to solve JSP-SDST problem using notations as defined follows:

m: number of machines

i, j: notation used for operations

k: notation used for machines

D: The set of operations in the floor

M_i: the specific machine that operation i requires

S_{ij}: the setup time between operations i and j, if operation j is processed just after operation i

t_i: processing time of operation i

F_i: starting time of operation i

C_i: completing time of operation i

M: very large positive number

X_{ijk} ($i \neq j$): 1, if operation j is processed just after operation i on machine k

0, otherwise

R_k: The dummy operation that describes the first operation on machine k

$$\text{Minimize max } (C_i = F_i + t_i) \tag{1}$$

Subject to:

$$F_i + t_i \leq F_j \ \forall \ i, j \in S_{ij} \tag{2}$$

$$F_i + t_i + S_{ij} \leq F_j + M\,(1 - X_{ijk}) \ \forall \ (i, j) \in D, (i, j, k \mid k = M_i = M_j) \tag{3}$$

$$\textstyle\sum_{i \in D} X_{ijk} = 1 \ \forall \ j \in D, (j, k \mid k = M_i = M_j) \tag{4}$$

$$\textstyle\sum_{j \in D} X_{ijk} = 1 \ \forall \ i \in D, (i, k \mid k = M_i = M_j) \tag{5}$$

$$t_{Rk} = 0 \ \forall \ k \in M \tag{6}$$

$$S_{Rk,i} = 0 \ S_{i,Rk} = 0 \ \forall \ i \in D, (i, k \mid k = M_i) \tag{7}$$

Equation 1 presents the objective function that attempts to reduce the completion time of all operations. Constraint set 2 ensures that an operation could not start until its preceding operation is done. Constraint set 3 insists that the starting time of each operation on one machine should be larger than the completion time of the operation that performs just before this operation considering the setup times between the pervious operation and current operation. Constraints set 4 and 5 force the job scheduling to have a unique sequence on each machine. In Constraints set 6 and 7, processing time and the relationship of setup times between dummy operations and other operations, which require the same machine, are set to zero.

It is almost the same model proposed by Choi et al. [18]. The main limit of these models is that they did not consider the first and the last operations processed on each machine. In fact, constraint 4 and constraint 5 insist that each operation is preceded and pursued by another operation on all machines which is cannot be the case for the first and the last operations. The authors believe that despite of the supposing of dummy operation it did not solve efficiently the problem for both models. So it would be useful to develop a new model which deals effectively with this limit.

2.2. Chinyao et al.'s Model

For this model proposed by Chinyao et al. [19], different notations are used. These notations will be used for the rest of the paper as follows:

n: number of jobs

i, j: notation used for jobs

E_k: set of jobs that are processed on the machine k

m_k: number of jobs which are processed on the machine k

n_i: operations number of job i

P_{ik}: processing time of job i on machine k

JT_{ik}: starting time of job i on machine k

Cmax: maximum completion time or makespan

S_{ijk}: the setup time between jobs i and j on machine k, if operation j is processed just after operation i

S_{0ik}: the starting setup time, if machine k starts with operation i

U_i: Set of operation pairs, two consecutive operations, of job i

ST_{ik}: Starting time of setup of job i on machine k

e_{0k}: a dummy operation numbered 0 in the set E_k

X_{ijk} ($i \neq j$): 1, if operation j is processed just after operation i on machine k

0, otherwise

Y_{ijk} ($i \neq j$): 1, if operation j is processed after operation i on machine k

0, otherwise

Each job i has a fixed machine sequence $\sigma_i = (\sigma_1^i, \sigma_2^i, .., \sigma_l^i)$ where σ_k^i represents the kth machine (operation) that the job i must is processed on.

The mixed integer programming proposed by Chinyao et al. [19], supposing that there aren't re-entrant flows and the objective consists just in minimizing the makespan, is written as follows:

$$\text{Minimize Cmax} \tag{8}$$

Subject to

$$JT\,\sigma_l^i + P\,\sigma_l^i \leq \text{Cmax} \ \forall \ i \in \{1,.., n\} \tag{9}$$

$$JT\,\sigma_k^i + P\,\sigma_k^i \leq JT\,\sigma_h^i \ \forall i \in \{1,.., n\}, \forall \ \sigma_k^i, \sigma_h^i \in U_i \tag{10}$$

$$JT_{ik} = ST_{ik} + \textstyle\sum_{j \in Ek \cup \{e0\}} X_{jik} S_{jik} \ \forall \ k \in \{1,.., m\}, i \in E_k \tag{11}$$

$$\textstyle\sum_{j \in Ek \cup \{e0\}} X_{jik} = 1 \ \forall \ k \in \{1,.., m\}, i \in E_k \tag{12}$$

$$\textstyle\sum_{i \in Ek} X_{e0ik} = 1 \ \forall \ k \in \{1,.., m\} \tag{13}$$

$$\textstyle\sum_{j \in Ek} X_{ijk} \leq 1 \ \forall \ k \in \{1,.., m\}, i \in E_k \tag{14}$$

$$JT_{ik} + P_{ik} - M\,(1 - Y_{ijk}) \leq ST_{jk} \ \forall \ k \in \{1,.., m\}, i, j \in E_k \tag{15}$$

$$JT_{jk} + P_{jk} - MY_{ijk} \leq ST_{ik} \ \forall \ k \in \{1,.., m\}, i, j \in E_k \tag{16}$$

$$Y_{ijk} - X_{ijk} \geq 0 \ \forall \ k \in \{1,.., m\}, i, j \in E_k \tag{17}$$

$$Y_{ijk} + X_{jik} \leq 1 \ \forall \ k \in \{1,.., m\}, i, j \in E_k \tag{18}$$

Equation 8 presents the objective function of the model. Constraint set 9 defines the makespan of a system. The makespan must be larger than the completion times of all jobs. Constraint set 10 presents the precedence relationship between two consecutive operations. Constraint set 11 defines the relation between the starting processing epoch and setup epoch. Constraint set 12 stipulates that there is only one preceding operation for each operation. Constraint set 13 stipulates that only one operation can be the first for processing machine k. Constraint set 14 stipulates that each operation has at most one successive operation on each machine. Constraints set 15 and 16 are the constraint of the operational sequence of the operations that are processed on the same machine k. Constraints set 17 and 18 define the relation between the two variables, Y_{ijk} and X_{ijk}.

Although, this model handles with the limit of Moghaddas et al.[5]'s model and Choi et al.[18]'s model, in fact it deals well with the first and the last operations by adding a dummy operation on each machine, but it uses a huge number of variables that generates a huge number of constraints likely to the other models existing in literature. For this reason, authors are motivated to propose a new model which considers fewer variables and constraints.

3. Proposed MIP Model and Comparisons

3.1. Proposed Mixed Integer Program

For the new model the same notations described in sub-section 2.2 are used but the new MIP is built differently the MIP:

$$\text{Minimize Cmax} \tag{19}$$

Subject to

$$\sum_{j \in Ek} X_{ijk} \leq 1 \ \forall\, k \in \{1,.., m\},\ i \in E_k \tag{20}$$

$$\sum_{j \in Ek} X_{jik} \leq 1 \ \forall\, k \in \{1,.., m\},\ i \in E_k \tag{21}$$

$$JT_{jk} \geq JT_{ik} + P_{ik} + M\,(X_{ijk}\text{-}1) + S_{ijk} \ \forall\, k \in \{1,.., m\},\ i, j \in E_k \tag{22}$$

$$\sum_{i \in Ek} \sum_{j \in Ek} X_{ijk} \geq mk\text{-}1 \ \forall\, k \in \{1,.., m\} \tag{23}$$

$$JT_{ik} \geq S_{0ik} - M\sum_{j \in Ek} X_{jik} \ \forall\, k \in \{1,.., m\},\ i \in E_k \tag{24}$$

$$JT\ \sigma_l^i + P\ \sigma_l^i \leq \text{Cmax} \ \forall\, i \in \{1,.., n\} \tag{25}$$

$$JT\ \sigma_k^i + P\ \sigma_k^i \leq JT\ \sigma_h^i \ \forall i \in \{1,.., n\},\ \forall\, \sigma_k^i, \sigma_h^i \in U_i \tag{26}$$

$$\text{Cmax} \geq 0 \tag{27}$$

$$JT_{ik} \geq 0 \ \forall\, k \in \{1,.., m\},\ i \in E_k \tag{28}$$

$$X_{ijk} \in \{0, 1\} \ \forall\, k \in \{1,.., m\},\ i, j \in E_k \tag{29}$$

Equation 19 presents the objective function. Constraint set 20 stipulates that each operation has at most one successive operation on all machines. Constraint set 21 stipulates that each operation is preceded at most by one operation on all machines. Constraint set 22 imposes that the starting time of each operation on each machine should be larger than the starting time of the operation that performs just before this operation considering the processing time of the previous operation as well as the setup times between the previous operation and the current operation. Constraint set 23 stipulates that the job sequence on machine k is defined by the number of operations executed on machine k minus one binary variables equal to one. By dint of constraint set 22, which forces the MIP to minimize the number of binary variables equal to one in order to minimize the objective function and prohibits the partial grouping of operations on each machine, the authors could put the constraint set 23 in form of inequality. Constraint set 24 ensures that the starting time of a staring operation on each machine is larger than its starting setup time. Constraint set 25 ensures that the makespan is larger than the completion time of all jobs. Constraint set 26 presents the precedence relationship between each two consecutives operations for all jobs. Constraints set 27 and 28 represent the non-negativity restrictions for Cmax and JT_{ik}. Constraint set 29 represents the binary restrictions for X_{ijk}.

Unlike the existing models, the proposed MIP did not add a dummy operation for each machine in order to deal with the first and the last operations, so the authors considered them in the formulation of the constraints.

If all starting setup time ST_{ik} are equal to 0, which is often the case in practice, constraint set (24) can be eliminated.

3.2. Comparison Between Models

Table 1 shows the difference in term of characteristics number between the existing models and the proposed MIP.

Table 1. *Comparison between the proposed MIP and the existing models.*

Characteristics		Number of binary variables	Number of continuous variables	Number of constraints
New proposed model		$\sum_{k=1}^{m} mk(mk-1)$	$\sum_{k=1}^{m} mk+1$	$n+m+3\sum_{k=1}^{m} mk+\sum_{k=1}^{m} mk(mk-1)+\sum_{i=1}^{n}(ni-1)$
Chinyao and al.'s model	Number	$\sum_{k=1}^{m} mk\,(2mk-1)$	$2\sum_{k=1}^{m} mk+1$	$n+2m+7\sum_{k=1}^{m} mk+\sum_{i=1}^{n}(ni-1)2\sum_{k=1}^{m} mk(mk-1)$
	Profit	$\sum_{k=1}^{m} mk^2$	$\sum_{k=1}^{m} mk$	$4\sum_{k=1}^{m} mk+\sum_{k=1}^{m} mk(mk-1)+m$
Moghaddas et al.'s model	Number	$\sum_{k=1}^{m} mk(mk+1)$	$\sum_{k=1}^{m} mk+1+m$	$\sum_{k=1}^{m} mk(mk+1)+n+\sum_{i=1}^{n}(ni-1)+4\sum_{k=1}^{m}(mk+1)$
	Profit	$2mk$	M	$3\sum_{k=1}^{m} mk\text{-}m+4$

According to Table 1 it's clear that the proposed MIP permits to reduce considerably the number of binary variables, number of continuous variables and number of constraints comparing to Chinyao et al.[19]'s model. Moreover Table 1 shows Moghaddas et al.[5]'s model not only doesn't deal with the first and the last operations on each machine but also contains more variables and constraints than the proposed model.

The proposed MIP uses the same number of binary variables and continuous variables as the disjunctive model proposed by Manne [20] which is considered the best MIP for simple JSP According to Ku et al. [21]. Although the proposed MIP uses more constraints but the two models are equivalent for large size problems.

4. Experimental Model Implementation and Validation

In order to validate the proposed mixed integer programming, authors apply it in well-known example from the literature selected from Moghaddas et al. [5]. This example is characterized by four machines and four jobs. Table 2 shows the machine sequence of each job.

$$m=4 \tag{30}$$

$$n=4 \tag{31}$$

$$E_k=\{1, 2, 3, 4\} \ \forall\ k \in \{1,.., m\} \tag{32}$$

$$m_k=4 \ \forall\ k \in \{1,.., m\} \tag{33}$$

$$n_i=4 \ \forall\ i \in \{1,.., n\} \tag{34}$$

Equations (30), (31), (32), and (34) describe the example. Table 3 contains the processing times P_{ik} of each operation. Tables 4, 5, 6 and 7 represent the setup times S_{ijk} between each two operations on each machine. For this example all starting setup time are considered equal to 0.

Table 2. *The machine sequencing of jobs.*

σ_i	Sequencing			
σ_1	4	3	2	1
σ_2	4	1	2	3
σ_3	3	2	4	1
σ_4	1	2	3	4

Table 3. *The processing times of each operation.*

Job	Processing time			
1	**2**	**3**	**2**	**3**
2	3	2	7	2
3	4	3	6	4
4	10	3	4	5

Table 4. *The setup times on machine 1.*

S_{ij1}				
Job	**1**	**2**	**3**	**4**
1	0	1	2	0
2	1	0	1	0
3	1	0	0	1
4	1	0	2	0

Table 5. *The setup times on machine 2.*

S_{ij2}				
Job	**1**	**2**	**3**	**4**
1	0	0	1	1
2	0	0	1	0
3	0	2	0	0
4	0	2	1	0

Table 6. *The setup times on machine 3.*

S_{ij3}				
Job	**1**	**2**	**3**	**4**
1	0	0	2	1
2	1	0	1	1
3	0	2	0	1
4	0	2	0	0

Table 7. *The setup times on machine 4.*

S_{ij4}				
Job	**1**	**2**	**3**	**4**
1	0	3	1	3
2	2	0	2	2
3	1	4	0	3
4	1	1	2	0

Based on the proposed model, authors used Lindo 6.1 to solve this problem and got $Cmax_{optimal}$=24 and the results of starting times of operations and sequence order are given in Tables 8 and 9:

Table 8. *The starting times of operations.*

JT_{ik} k i	1	2	3	4
1	21	13	9	0
2	13	15	22	5
3	16	4	0	10
4	0	10	13	19

Table 9. *The values of the binary variables.*

X_{ijk} ij k	12	13	14	21	23	24	31	32	34	41	42	43
1	0	0	0	0	1	0	1	0	0	0	1	0
2	1	0	0	0	0	0	0	0	1	1	0	0
3	0	0	1	0	0	0	1	0	0	0	1	0
4	1	0	0	0	1	0	0	0	1	0	0	0

Figure 1 illustrates the Gantt of the optimal solution of the described problem.

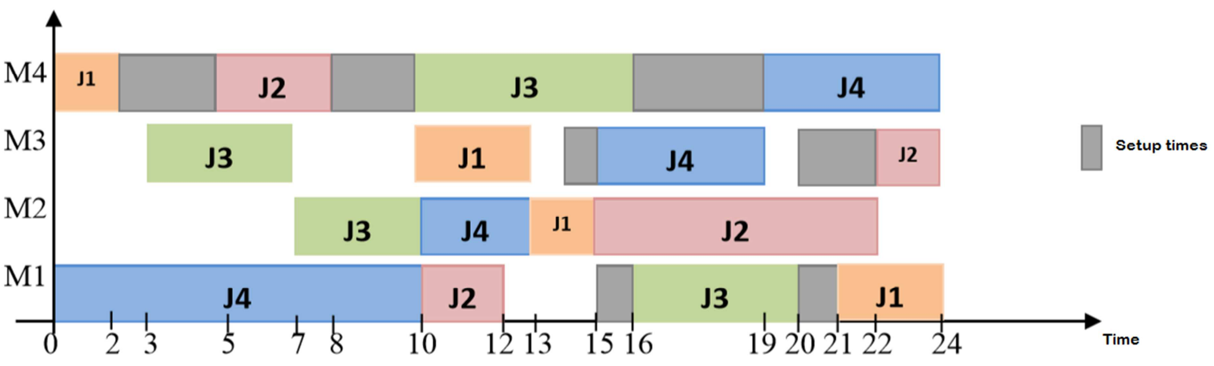


Figure 1. *Gantt of the optimal solution for the described problem.*

5. Conclusion

The main contribution of this paper is the proposition of a new MIP which deals with separable SDST in order to minimize the makespan. The model was designed to be subtly faster than existing one in literature. The proposed model was compared it to the main existing models in literature handling with SDST. Authors found that it considers less number of binary and continuous variables, in addition less number of constraints then potentially better convergence. To validate the model and its waited performance, it was implemented on a well-known example from literature including four jobs and four machines.

An additional interest of the new proposed MIP is that it is possible to adapt it easily for others objective functions such as min ∑Ci, min Lmax, min ∑Li, min ∑Ti, min Tmax or add equally others constraints such as release time constraints, time lags constraints, precedence constraints, reentrant flows.

References

[1] Garey M. R. and Johnson D. S. (1979), *Computers and intractability: a guide to the theory of NP completeness*, Series of Books in the Mathematical Sciences, W. H. Freeman and company, New York.

[2] Garey M. R., Johnson D. S. and Sethi R. (1976), The complexity of flow shop and job shop scheduling, *Mathematics of Operations Research*, Vol. 1 No. 2, pp. 117-129.

[3] Allahverdi A., Gupta J. N. D. and Aldowaisan T. (1999), A review of scheduling research involving setup considerations, *International Journal of Management*, Vol. 27 No. 2, pp. 219-239.

[4] Chen Z. and Pius E. J. (1989), Scheduling in manufacturing shop with sequence-dependent setups, *Robotics and Computer-Integrated Manufacturing*, Vol. 5 No. 1, pp. 73-81.

[5] Moghaddas R. and Houshmand M. (2008), Job shop scheduling problem with sequence dependent setup times, *Proceedings of the International Multiconference of Engineers and Computers Scientists*, Vol. 2, Hong Kong, China.

[6] Artigues C., Belmokhtar S. and Feillet D. (2004), A new exact solution algorithm for the job shop problem with sequence time setup times, *International Conference on Integration of Artificial Intelligence and Operations Research Techniques in Constraint Programming*, Vol. 3011, Berlin, Heidelberg, pp. 37-49.

[7] Focacci F., laborie P. and Nuijten W. (2000), Solving scheduling problems with setup times and alternatives resources, *Proceedings International Conference on Artificial Intelligence Planning Systems*, Vol. 5, Breckenridge, CO, USA, pp. 92-101.

[8] Camino R. V., Ramiro V. and Miguel A. G. (2010), Local search and genetic algorithm for the job shop scheduling problem with sequence-dependent setup times, *Journal of Heuristics*, Vol. 16 No. 2, pp. 139-165.

[9] Cheung W. and Zhou H. (2001), Using genetic algorithms and heuristics for job shop scheduling with sequence dependent setup times, *Annals of Operations Research*, Vol. 107 No. 1, pp. 65-81.

[10] Choi I. C. and Choi D. (2002), A local search algorithm for job shop scheduling problems with alternative operations and sequence-dependent setups, *Computers and Industrial Engineering,* Vol. 42 No. 1, pp. 43-58.

[11] Zhou Y., Li B. and Yang J. (2006), Study on job shop scheduling with sequence dependent setup times using biological immune algorithm, *The International Journal of Advanced Manufacturing Technology*, Vol. 30 No. 1, pp. 105-111.

[12] Yang Y. (2001), *A study on compound agent and its application to the enterprise management information system*, Dissertation, Donghua University, Shanghai, China.

[13] Shen L. (2014), A tabu search algorithm for the job shop problem with sequence dependent setup times, *Computers and Industrial Engineering,* Vol. 78, pp. 95-106.

[14] Allahverdi A. (2015), The third comprehensive survey on scheduling problems with setup times/costs, *European Journal of Operational Research,* Vol. 246 No. 2, pp. 345-378.

[15] Shen L., Dauzère-Pérès S. and Neufeld J. S. (2017), Solving the flexible job shop scheduling problem with sequence-dependent setup times, *European Journal of Operational Research,* Available online.

[16] Knopp S., Dauzère-Pérès S. and Yugma C. (2017), A batch-oblivious approach for Complex Job-Shop scheduling problems, *European Journal of Operational Research,* Vol. 263 No. 1, pp. 50-61.

[17] Abdelmaguid T. F., neighborhood search function for flexible job shop scheduling with separable sequence-dependent setup times, *Applied Mathematics and Computation,* Vol. 260, pp. 188-203.

[18] Choi I. C. and Korkmaz O. (1997), Job shop scheduling with separable sequence-dependent setups, *Annals of Operations Research*, Vol. 70 No. 0, pp. 155-170.

[19] Chinyao L., Tai-His W. and Chih-Ming H. (2005), Mathematical modeling of multi-objective job shop scheduling with dependent setups and re-entrant operations, *The International Journal of Advanced Manufacturing Technology*, Vol. 27 No. 1, pp. 181-189.

[20] Manne A. S. (1960), On the job-shop scheduling problem, *Operations Research*, Vol. 8 No. 2, pp. 219-223.

[21] Ku W. Y. and Christopher Beck J. (2016), Mixed Integer Programming Models for Job Shop Scheduling: A Computational Analysis, *Computers and Operations Research*, Vol. 73, pp. 165-173.