International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 02 | Feb 2021 www.irjet.net p-ISSN: 2395-0072

# Genetic Algorithm for Minimizing the Makespan in Job Shop Scheduling with Machine Setup Issue

**Abdalla Omar Dagroum Ali, Houssein .M.A. Elaswad, Abulgasem Jabuda**

*Faculty of Engineering, Sabratha University, Regdaleen, Libya*
*E-mail: dagroum@hotmail.com*

---

**Abstract -** *Scheduling problems that involving machines setup times have recently attracted significant attention in many real-world industrial applications. In this paper a job-shop scheduling problem with machines dependent setup times and job sequence independent setup times is considered. The setup time is assumed to start before the corresponding job becomes available on the machine, which is known as anticipatory setup time. A genetic algorithm was applied with the aim of minimizing the total completion time (the makespan). After applying crossover and mutation operators, every solution was then evaluated with taking into consideration the machines setup times. Computational experiments were carried out for 5 adapted benchmark problems, and the obtained results used to validate the effectiveness of the proposed algorithm.*

***Key Words***: Job Shop Scheduling, Machine Setup Time, Genetic Algorithm, Makespan

## 1. INTRODUCTION

Scheduling problems that involving machines setup times have recently attracted significant attention in many real-world industrial applications. When properly incorporated into the scheduling decision, the addition of setup times has been shown to give significant improvements in reliability and accuracy of the overall scheduling system. In general, machine setup time can be classified into two main types; - machine independent setup time and machine dependent setup time, see Figure 1.

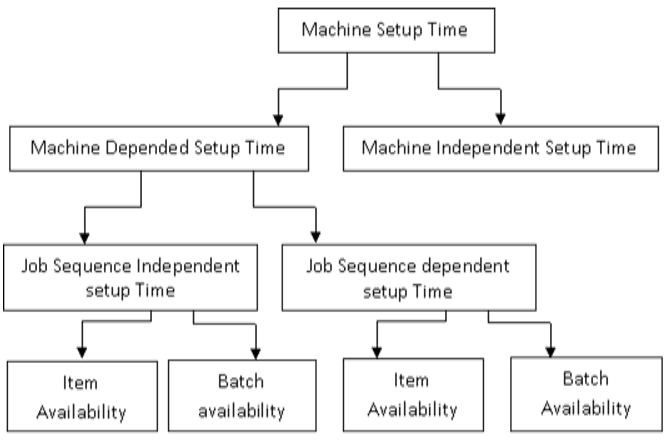


**Fig -1**: Classification of machine setup times

The case of machine independent setup time occurs when the setup time is included in the operation processing time or when there is no need for setup time. However, in the machine dependent setup time, the setup can be job sequence independent, where the preceding job does not have an effect on the duration of the setup time, or job sequence dependent, where the duration of the setup depends directly on the preceding job. These two types can be further divided into setup time with batch availability or setup time with item availability [1], [2].

This paper deals with job sequence independent setup times with item availability in the Job-Shop Scheduling Problem (JSP). In the job sequence independent and machine dependent type, a setup is required if a job is processed on a machine immediately after a job of different group [1]. The application of this problem can be found in many small and medium size manufacturing companies in which different tools are required to process operations of different jobs on the same machine such as different milling cutters in a milling machine or different cutting tools in turning machine. A comprehensive review of scheduling problems with setup times can be found in [2], [3], [4], [5]. So far, only a few papers have dealt with JSP's with job sequence independent setup time. Sotskov, Tautenhahn et al. (1996) proposed different insertion techniques combined with beam search to solve the problem, in which number of different jobs is partitioned into a number of groups. The setup takes place on a machine when the first job has to be processed on that machine and when a job has to be processed after a job of another group [1]. The present paper differs from Sotskov, Tautenhahn et al. (1996) work so that the group is made based on the operations and machines, i.e. a job can belong to one group on one machine and to a different group on another machine. Ali, Hackney et al. (2015) developed a genetic algorithm for minimizing the number of tardy jobs in job shop scheduling with machine setup issue [6]. Still, the largest part of the recent works has focused on solving JSPs where setup times are sequence dependent. Vinod and Sridharan (2008) proposed a discrete event simulation model to minimize the mean flow time, mean tardiness, mean setup time and mean number of setups, for JSPs with sequence dependent setup times [7]. Naderi, Zandieh et al. (2009) also considered JSPs where the setup times are sequence dependent with the aim of minimizing the makespan [8]. Vela, Varela et al. (2010) applied a hybrid of Genetic Algorithm (GA) and diversification mechanism,

known as a restart phase, and a simple form of Local Search (LS) to solve JSPs with sequence dependent setup times [9]. For the problem of scheduling flexible job shops with sequence dependent setup times Mousakhani (2013) proposed a metaheuristic algorithm based on iterated local search to find a schedule with minimum total tardiness [10]. González, Vela et al. (2013) applied Tabu Search (TS) to tackle the JSPs with sequence dependent setup times in order to minimize the maximum lateness [11]. Sharma and Jain (2014) developed a discrete event simulation model of a stochastic dynamic job shop manufacturing system with sequence dependent setup times to investigate the performance of some dispatching rules on makespan, mean flow time, maximum flow time, mean tardiness, maximum tardiness, number of tardy jobs, total setups and mean setup time [12]. In this paper, a GA for solving JSP with machines dependent setup times and job sequence independent setup time is developed in order to satisfy some production system requirements.

## 2. PROBLEM DESCRIPTION

In this paper, the JSP with job sequence independent and machine dependent setup requirements is considered, where a number of jobs that consist of a number of operations has to be processed on various machines. The sequence of operations for each job is predefined based on job technological requirement. Machines can only handle one job at a time and transportation time between machines is neglected. Operations of the same job cannot be processed concurrently and cannot be started until their precedence operation is finished and, if necessary, a setup completed. The setup takes place on a machine when the operation has to be processed on that machine as a first and when an operation has to be processed after an operation of another group on that machine. Taking into account that the setup can be started before the corresponding job becomes available on the machine, which known as anticipatory setup time, the objective is to find an optimal solution for minimum makespan. The problem can be formulated mathematically as follows:

**Indices**

$n$ : number of jobs.

$m$ : number of machines.

$i$ : job $i$ $(i = 1,2,\ldots,n).$

$k$ : machine $k$ $(k = 1,2,\ldots,m).$

$p_{ijk}$ : processing time for operation $j$ of job $i$ on machine $k$.

$St_{gk}$ : setup time for category of operations that belongs to group $g$ on machine $k$.

**Decision variables**

$S_{ij}$ : start time for operation $j$ of job $i$.

$C_{ij}$ : completion time for operation $j$ of job $i$.

$C_i$ : completion time for the last operation of job $i$.

$X_{ijk} = 1$ if operation $j$ of job $i$ is processed on machine $k$, otherwise $X_{ijk} = 0$.

$Y_{ijpqk} = 1$ if operation $j$ of job $i$ precedes operation $q$ of job $p$ on machine $k$, otherwise $Y_{ijpqk} = 0$.

$Z_{ijk} = 1$ if operation $j$ of job $i$ is processed on machine $k$ as the first, otherwise $Z_{ijk} = 0$.

$G_{jqk} = 0$ If two different operations $j$ and $q$ from the same setup group are processed consecutively on machine $k$, otherwise $G_{jqk} = 1$.

**Constraints**

$$(S_{ijk} - St_{gk}) * Z_{ijk} \geq 0 \quad (1)$$

$$S_{ij} \geq C_{i,j-1} \quad (2)$$

$$(C_{ij} - S_{ij}) * X_{ijk} = p_{ijk} \quad (3)$$

$$(C_{pq} - C_{ij} - p_{pqk}) * Y_{ijpqk} + St_{gk} * G_{jpk} \geq 0 \quad (4)$$

$$(C_{ij} - C_{pq} - p_{ijk}) * (1 - Y_{ijpqk}) + St_{gk} * G_{jqk} \geq 0 \quad (5)$$

$$\sum_{k=1}^{m} X_{ijk} = 1 \qquad \forall i, j \quad (6)$$

Constraint 1 ensures that the first job on the machine cannot be processed until the machine setup has been completed. Constraint 2 ensures that Operation $j$ of job $i$ cannot be started before its preceding operation $j-1$ is completed. Constraint 3 ensures that the difference between the start time and the completion time of operation $j$ on machine $k$ is equal to the required processing time of operation $j$ on machine $k$. Constraints 4 & 5 ensure that two different operations cannot be processed at the same time on the same machine and that machine setup must take place when an operation has to be processed after an operation of another group on that machine. Constraint 6 ensures that every job is processed by only one machine in each stage.

**Objective**

Minimizing the makespan ( $C_{\max}$ )

Where $C_{\max} = \max(C_1, \ldots, C_n)$ (7)

## 3. SOLUTION APPROACH

In this work the operation based representation that was proposed by Gen, Tsujimura et al. (1994) is used to represent the solution or the chromosome. Each gene in this method stands for a sequence of one operation. Each integer number in the gene represents a job type. The first occurrence of a job in a chromosome stands for the first operation of that job in the corresponding machine, while the second occurrence of a job in a chromosome stands for the second operation of that job in the corresponding machine and so on [13]. The initial population that contains

a number of different solutions is generated randomly. Solutions are evaluated based on the value of total completion time (the makespan), so that the schedule with the smallest makespan has better fitness [14]. The evaluation function takes into account the machine setup time when it is required. Here we use same technique as that used in [6] by using complex numbers in the machine matrix to represent the machine number and setup group for each operation on that machine. For example if operation $q$ of job $p$ on machine $k$ belongs to setup group $f$ then it will appear on machine matrix $(k+gi)$. Where $k$ refers to the machine required to do operation $q$, and $g$ is the setup group of job $p$ on machine $k$.

To create a new generation that hopefully can be better than the current generation; the approach of two point crossover has been used in this work. For the two-point crossover, two parents are selected randomly using crossover probability ( $p_c$ =0.7) from the top 40 percent of solution pool, as those have better fitness to make two new children. Repairing algorithm is then used in order to repair the solution from infeasible to feasible solution. Mutation usually works with a single chromosome to keep diversity in a population and to avoid the GA from trapping in local optima. Here with mutation probability ( $p_m$ =0.3) two different numbers in which each number refers to a job's operation are randomly selected and exchanged. The rest of chromosomes are regenerated randomly. These procedures are repeated until the number of generations is reached.

## 4. EXPERIMENT AND RESULTS

In this section, 5 benchmark datasets (LA01, LA02, L03, LA04, LA05) of JSPs taken from the OR-Library are used to evaluate the performance of the algorithm [15]. In order to adapt the instances for the machine setup problem, jobs are divided into different setup groups in each machine. The same set of setup groups in each machine have been used for all 5 benchmark problems in this work as shown in Table 1.

**Table -1:** Job's setup groups and times in each machine

| Setup groups | Machines | | | | |
|---|---|---|---|---|---|
| | **M1** | **M2** | **M3** | **M4** | **M5** |
| **g1** | J1,J2, J5 | J6,J7, J10 | J4,J7, J8 | J9, J10 | J1,J2,J3,J4,J5,J6, J7,J8,J9,J10 |
| $St_{1k}$ | 20 | 18 | 9 | 14 | 17 |
| **g2** | J4,J6, J7 | J3,J5, J8,J9 | J1,J3,J6, J10 | J3,J5, J6,J7 | - |
| $St_{2k}$ | 15 | 13 | 11 | 14 | - |
| **g3** | J3,J8, J9,J10 | J1,J2, J4 | J2,J5, J9 | J1,J2, J4,J8 | - |
| $St_{3k}$ | 9 | 15 | 16 | 11 | - |

In the following example, we use LA01 to show the representation of Machines $M$, processing time $P$, and setup time $St$ matrices.

$$M = \begin{bmatrix} 1+1i & 5+1i & 4+3i & 3+2i & 2+3i \\ 5+1i & 3+3i & 4+3i & 2+3i & 1+1i \\ 3+2i & 4+2i & 1+3i & 2+2i & 5+1i \\ 1+2i & 5+1i & 4+3i & 2+3i & 3+1i \\ 5+1i & 3+3i & 2+2i & 1+1i & 4+2i \\ 1+2i & 2+1i & 4+2i & 5+1i & 3+2i \\ 3+1i & 4+2i & 1+2i & 2+1i & 5+1i \\ 2+2i & 5+1i & 1+3i & 3+1i & 4+3i \\ 3+3i & 1+3i & 4+1i & 5+1i & 2+2i \\ 4+1i & 3+2i & 2+1i & 1+3i & 5+1i \end{bmatrix}$$

$$P = \begin{bmatrix} 21 & 53 & 95 & 55 & 34 \\ 21 & 52 & 16 & 26 & 71 \\ 39 & 98 & 42 & 31 & 12 \\ 77 & 55 & 79 & 66 & 77 \\ 83 & 34 & 64 & 19 & 37 \\ 54 & 43 & 79 & 92 & 62 \\ 69 & 77 & 87 & 87 & 93 \\ 38 & 60 & 41 & 24 & 83 \\ 17 & 49 & 25 & 44 & 98 \\ 77 & 79 & 43 & 75 & 96 \end{bmatrix} \quad St = \begin{bmatrix} 20 & 15 & 9 \\ 18 & 13 & 15 \\ 9 & 11 & 16 \\ 14 & 14 & 11 \\ 17 & 0 & 0 \end{bmatrix}$$

We incorporate the groups of operation setups in the machine matrix by using the complex number. The real part of the complex number represents the machine number while the imaginary part of the complex number represents the setup group of that job on that machine. For more illustration, the first operation of job 3 is on machine 3 with processing time equal to 39. This operation belongs to setup group 2 with setup time equal to 11. If operation 2 of job 10 is the immediate successor of operation 1 of job 3 on machine 3, then there is no machine setup between these two operations. However, if the immediate successor of operation 1of job 3 on machine 3 was operation 1 of job 7, then machine setup will take place between these two operations. These procedures are continually applied between any two immediate successors for any operations on the same machine. The setup can also be started before the corresponding job is available on the machine. Fig - 1 depicts the resulting schedule for the LA01 benchmark problem with the objective of minimizing the makespa ( $C_{\max}$ ).

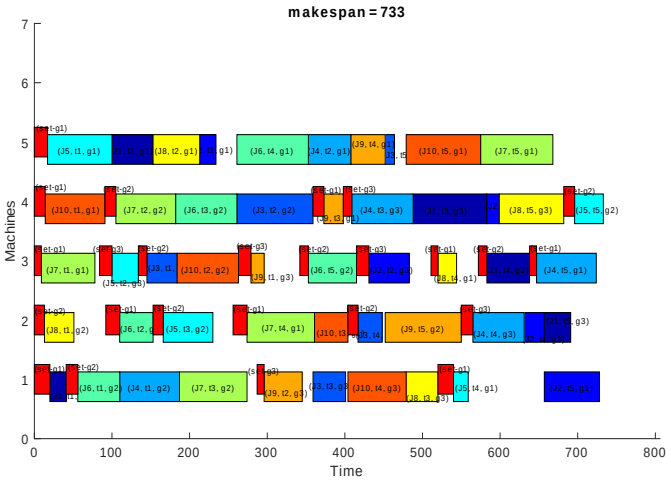


Fig - 1: Gantt chart for optimal schedule with incorporating the setup times

Table 2 shows the values of makespan ($C_{\max}$) for all 5 benchmark datasets, with and without the incorporation of setup times, and with the aims of minimizing the $C_{\max}$.

**Table -2:** $C_{\max}$ with and without setup times

| | **Minimizing the $C_{\max}$ without incorporating setup times** | **Minimizing the $C_{\max}$ with incorporating setup times** |
|---|---|---|
| **LA1** | 666 | 733 |
| **LA2** | 655 | 724 |
| **LA3** | 597 | 671 |
| **LA4** | 590 | 658 |
| **LA5** | 593 | 611 |

## 5. CONCLUSIONS

In this paper we applied a genetic algorithm for solving job shop scheduling problems with machine dependent setup times and job sequence independent setup times using complex numbers in the machine matrix to represent the machine number and setup group for each operation on that machine. The aim was to find an optimal solution for the criterion of minimizing the makespan ($C_{\max}$). The model was tested using 5 benchmark scheduling problems after they had been adapted to suit the problem in hand. The computational results show that the proposed approach was effective to solve the problem in hand in terms of solution quality and gives a feasible solution. Future work will focus on consideration and incorporation of batch availability, multi criteria optimization, alternative machine, and fuzzy processing time and fuzzy setup time into the problem.

## REFERENCES

[1] Sotskov, J. N., T. Tautenhahn and F. Werner (1996). On the Application of Insertion Techniques for Job Shop Problems with Setup Times, Otto-von-Guericke-Univ., Fak. für Math..

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

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

[4] Potts, C. N. and M. Y. Kovalyov (2000). "Scheduling with batching: A review." European Journal of Operational Research 120(2): 228-249.

[5] Sharma, P. and A. Jain (2015). "A review on job shop scheduling with setup times." Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture.

[6] Ali, A., Hackney, P., Birkett, M., & Bell, D. (2015). Genetic algorithms for minimizing the number of tardy jobs in job shop scheduling with machine setup issue. In Proceedings of the 13th International Conference on Manufacturing Research (ICMR) (pp. 63-69). University of Bath.

[7] Vinod, V. and R. Sridharan (2008). "Scheduling a dynamic job shop production system with sequence-dependent setups: An experimental study." Robotics and Computer-Integrated Manufacturing 24(3): 435-449.

[8] Naderi, B., M. Zandieh and S. M. T. Fatemi Ghomi (2009). "Scheduling job shop problems with sequence-dependent setup times." International Journal of Production Research 47(21): 5959-5976.

[9] Vela, C., R. Varela and M. González (2010). "Local search and genetic algorithm for the job shop scheduling problem with sequence dependent setup times." Journal of Heuristics 16(2): 139-165.

[10] Mousakhani, M. (2013). "Sequence-dependent setup time flexible job shop scheduling problem to minimise total tardiness." International Journal of Production Research 51(12): 3476-3487.

[11] González, M., C. Vela, I. González-Rodríguez and R. Varela (2013). "Lateness minimization with Tabu search for job shop scheduling problem with sequence dependent setup times." Journal of Intelligent Manufacturing 24(4): 741-754.

[12] Sharma, P. and A. Jain (2014). "Analysis of dispatching rules in a stochastic dynamic job shop manufacturing system with sequence-dependent setup times." Frontiers of Mechanical Engineering 9(4): 380-389.

[13] Gen, M., Y. Tsujimura and E. Kubota (1994). Solving job-shop scheduling problems by genetic algorithm. Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on.

[14] Werner, F. (2011) "Genetic algorithms for shop scheduling problems: a survey." 1- 66.

[15] Beasley, J. E. (1990). "OR-Library: Distributing Test Problems by Electronic Mail." J Oper Res Soc 41(11): 1069-1