Global Journal of Researches in Engineering: A
Mechanical and Mechanics Engineering
Volume 18 Issue 1 Version 1.0 Year 2018
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN:2249-4596 Print ISSN:0975-5861

Optimization of the Flexible Job Shop Scheduling Problem for Economic Sustainability

Md. Riyad Hossain, Md. Kamruzzaman Rasel, Md. Isanur Shaikh &
Utpal Kumar Dey

Khulna University of Engineering & Technology

Abstract- The flexible job-shop scheduling problem (FJSP) is one of the challenging optimization problems as they occupy very large search space. Solving this kind of problems with conventional methods are obsolete now as the Internet of Things (IoT) has changed scheduling platform by means of cloud computing and advanced data analytics. Genetic Algorithms (GAs) is a popular modern tool for machine scheduling problems and in this work, a scheduling algorithm has been developed to minimize total tardiness and make span time of parallel machines which is promoting overall economic sustainability. The algorithm consists of a machine selection module (MSM) that helps to select the right machine on the right time with the help of global selection (GS) technique by generating high quality initial population. To represent an optimized solution of the FJSP, an improved chromosome representation is used while adopting uniform crossover and mutation operator.

Keywords: economic sustainability, flexible job-shop scheduling, genetic algorithm, machine selection module, global search technique.

GJRE-A Classification: FOR Code: 091399, 290502p

OPTIMIZATION OFTHEFLEXIBLEJOBSHOPSCHEDULINGPROBLEMFORECONOMICSUSTAINABILITY

Strictly as per the compliance and regulations of:

Optimization of the Flexible Job Shop Scheduling Problem for Economic Sustainability

Md. Riyad Hossain [α], Md. Kamruzzaman Rasel [σ], Md. Isanur Shaikh [ρ] & Utpal Kumar Dey [ω]

Abstract- The flexible job-shop scheduling problem (FJSP) is one of the challenging optimization problems as they occupy very large search space. Solving this kind of problems with conventional methods are obsolete now as the Internet of Things (IoT) has changed scheduling platform by means of cloud computing and advanced data analytics. Genetic Algorithms (GAs) is a popular modern tool for machine scheduling problems and in this work, a scheduling algorithm has been developed to minimize total tardiness and make span time of parallel machines which is promoting overall economic sustainability. The algorithm consists of a machine selection module (MSM) that helps to select the right machine on the right time with the help of global selection (GS) technique by generating high quality initial population. To represent an optimized solution of the FJSP, an improved chromosome representation is used while adopting uniform crossover and mutation operator. The result showed that proposed algorithm is much more effective and efficient for solving flexible job-shop scheduling problem which is helping to reduce the overall downtime significantly.

Keywords: economic sustainability, flexible job-shop scheduling, genetic algorithm, machine selection module, global search technique.

I. Introduction

Nowadays sustainable manufacturing is an important issue for every industry and industries are aiming to achieve triple bottom line (TBL) sustainability for maintaining their competitive benchmark. The triple bottom line sustainability means achieving sustainability in every sector and most specifically achieving sustainability in economic, social and environmental sectors (Islam & AlGeddawy, 2017) (D. Chen, Thiede, Schudeleit, & Herrmann, 2014). Economic sustainability is one of the major pillars of TBL sustainably which is an expression of economic viability of a company (D. Chen et al., 2014; Islam & AlGeddawy, 2017; Kannegiesser & Günther, 2014). In this scheduling problem, the importance of economic sustainability has been highlighted and its mutual relationship between the optimized result has been analyzed.

Scheduling is involved with the proper distribution and availability of resources for accomplishing a set of tasks over a specified time (Roshanaei, Azab, & ElMaraghy, 2013; Shen, Dauzère-Pérès, & Neufeld, 2018). In 1985, Davis first applied Genetic Algorithms (GAs) to scheduling problems (L. Davis, 1991; T. E. Davis, 1991; Kelly Jr & Davis, 1991; Montana & Davis, 1989). While applying GAs to Scheduling problems the main problem is to find a suitable chromosome representation and genetic operators in order to create feasible schedules (Ak & Koc, 2012). The classical method of solving job-shop scheduling problem (JSP) is time consuming and sometimes impossible. The JSP consists of a set of n jobs processed by a set of m machines with the objective to minimize certain criteria (Pezzella, Morganti, & Ciaschetti, 2008). At time zero each machine is fully available, processing only one operation at a time. Each job contains a pre-specified processing order and specific time on the machines that are fixed and known in advance. In modern manufacturing plant, a machine may have the flexible capability to be set up to process more than one type of operations. This leads to a modified version of JSP called flexible JSP (FJSP) (Gen, Gao, & Lin, 2009; Xia & Wu, 2005; Yazdani, Amiri, & Zandieh, 2010). There are two types of FJSP (Brucker & Schlie, 1990) . For type I FJSP, jobs have alternative operation sequences and alternative identical or non-identical machines for each operation (Chan, Wong, & Chan, 2006; J. Chen, Chen, Wu, & Chen, 2008; Fattahi, Mehrabad, & Jolai, 2007). The problem is to select operation sequences for jobs and determine job processing orders on machines (Jahromi & Tavakkoli-Moghaddam, 2012). For type II FJSP, jobs can have only fixed operation sequences but alternative identical or non-identical machines for each operation. The problem is to arrange jobs to machines according to their operation sequences (Chan et al., 2006; J. C. Chen, Wu, Chen, & Chen, 2012; Gen et al., 2009; Xia & Wu, 2005).

In this article, an effective GA is proposed to solve the FJSP. Global Selection (GS) and Local Selection (LS) are designed to generate high-quality initial population in the initialization stage which could accelerate convergent speed (Zhang, Gao, & Shi, 2011). An improved chromosome representation method ''Machine Selection and Operation Sequence'' has been introduced to assist the initialization method and to assure the algorithm is performing well (Carlier &

Author α σ ρ: Department of Industrial Engineering and Management, Khulna University of Engineering & Technology, Khulna-9203, Bangladesh. e-mail: riyad35@gmail.com
Author ω: Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna-9203, Bangladesh.

Pinson, 1989; Moon, Lee, Seo, & Lee, 2002; Prakash, Tiwari, & Shankar, 2008). This method helps to find an efficient coding scheme of the individuals which respects all constraints of the FJSP. At the same time, different strategies for crossover and mutation operator are employed. After that, the computational results show that the proposed algorithm could get good solutions.

II. Problem Definitions

The FJSP consists in performing a set of n jobs $\{J_1, J_2, J_3\ldots., j_i, \ldots\ldots J_n\}$ on a set of m machines $\{M_1, M_2, M_3\ldots., M_k, \ldots\ldots M_m\}$. A job J_i is formed by a sequence of operations $\{O_{i1}, O_{i2}, O_{i3}\ldots., O_{ij}\}$ to be performed one after the other according to the given sequence. Each operation requires one machine selected out of a set of available machines. All jobs and machines are available at time 0, and a machine can only execute one operation at a given time. Preemption is not allowed, i.e., each operation must be completed without interruption once it starts. Machine Setup time and movement time between operations are negligible. The processing time of operation O_{ij} on machine M_m is $P_{ijk} > 0$. Flexibility of FJSP can be categorized into partial flexibility and total flexibility. A system will be called partial if every machine cannot perform every other operation because of some constraint or extremely high cost. Data associated with a partial flexible system are given in Table 1, in which rows stand for operations time and columns is for machines. In this Table, symbol 0 means that a machine cannot be executed the corresponding operation. Problem can be formulated by using following nodes and symbols:

Z is the set of all machines
n is the number of the total jobs
m is the number of the total machines
i is the index of the i^{th} job
j is the index of the j^{th} operation of job J_i
k, x is the index of alternative machine set
O_{ij} is the j^{th} operation of job J_i
P_{ijk} is the processing time of operation O_{ij} on machine k
S_{ijk} is the start time of operation O_{ij} on machine k
E_{ijk} is the end time of operation O_{ij} on machine k

Assumptions made in FJSP as follows:

- All machines are available at time t = 0;
- All jobs are released at time t = 0;
- For each job, the sequence of operations is predetermined and cannot be modified;
- Each machine can only execute one operation at a time;
- Each operation Oij must be processed without interruption on one of a set of available machines;
- Recirculation occurs when a job could visit a machine more than once;
- The objective of the FJSP is to minimize the total completion time;

A basic representation of flexible job shop scheduling is as follows in Table 1.

Job	*O/P*	M_1	M_2	M_3	M_4	M_5
J_1	O_{11}	2	6	5	3	4
	O_{12}	0	8	0	4	0
J_2	O_{21}	3	0	6	0	5
	O_{22}	4	6	5	0	0
	O_{23}	0	7	11	5	8

There are 2 jobs and 5 machines, where rows correspond to operations and columns correspond to machines. Each cell denotes the processing time of that operation on the corresponding machine. In the table, the ‘‘0” means that the machine cannot execute the corresponding operation, i.e., it does not belong to the alternative machine set of the operation. So this instance is a Partial FJSP.

III. Algorithm for Flexible Job Scheduling

The proposed algorithm is developed according to the concept of the genetic algorithm. The detailed procedure of the proposed algorithm is as follows:

a) Determining operation sequence

The sequence of operation should be arranged on that way so that every other operation of an individual job must be accomplished before processed to next immediate operation. For example, to perform the job j_2 of table 1 the sequence of operations O_{21} 0_{22} should be maintained i.e. $O_{21} > O_{22}$

b) Global selection

We define that a stage is the process of selecting a suitable machine for an operation. Thus this method records the sum of the processing time of each machine in the whole processing stage. Then the machine which has the minimum processing time in every stage is selected. In particular, the first job and next job are randomly selected. Detailed steps are as follows:

1. Create a new array to record all machines' processing time, initialize each element to 0;
2. Select a job randomly and insure one job to be selected only once, then select the first operation of the job;
3. Add the processing time of each machine in the available machines and the corresponding machine's time in the time array together;
4. Compare the added time to find the shortest time, and then select the index k of the machine which has the shortest time. If there is the same time among different machines, a machine is selected randomly among them;
5. Set the allele which corresponds to the current operation in the MS part to k;

6. Add the current selected machine's processing time and its corresponding allele in the time array together in order to update the time array;
7. Select the next operation of the current job, and execute Step 3 to Step 6 until all operations of the current job are selected, then go to Step 8;
8. Go to step 2 until all jobs are all selected once.

The implementation of GS is given in Fig. 1.We assume that the sequence of job is $J_1 > j_2 > j_3$ from table 1, as mentioned earlier processing time '0' means the machine is unable to perform the operation. So, we easily see that the processing time on M_4 is the shortest in the alternative machine set of operation O_{11}. Hence, machine M_1 is selected to process the operation O_{11} of job J_1, and set corresponding allele in MS to the index of M_1.Then the processing time is added to the corresponding position in time array. The selection process continued till finding a new chromosome as described above 8 steps.

Time array	0	0	0	0	0
Operation			O_{11}		
Available machines	M_1	M_2	M_3	M_4	M_5
Processing time	2	6	5	3	4
Added time	2	6	5	3	4
Shortest time	2				
Selected machine	M_1				
Update time array	2	0	0	0	0
Machine Selection	1				

(a) O_{11}

Time array	2	0	0	0	0
Operation			O_{12}		
Available machines		M_2		M_4	
Processing time		8		4	
Added time		8		4	
Shortest time				4	
Selected machine				M_4	
Update time array	2	0	0	4	0
Machine Selection	1	4			

(b) O_{12}

Time array	0	0	12	13	0
Operation			O_{21}		
Available machines	M_1		M_3		M_5
Processing time	3		6		5
Added time	5		6		5
Shortest time	5				
Selected machine	M_1				
Update time array	5	0	0	4	0
Machine Selection	1	4	1		

(c) O_{21}

Fig.1: The global selection technique for operation (a) O_{11} (b) O_{12} (c) O_{21}

Finally, the selected machines of operations O_{11} O_{12} O_{21} may be M_1-M_4-M_1 and chromosome is 1-4-1.

c) *Generate the Initial Population*

As shown in Fig 1, the structure of the chromosome used in this paper consists of two components. The first component of the chromosome represents the list of machines while the second component contains the sequence of operations to be processed used in executing the operations in the first component.

For example, consider the 3-job, 5-machine problem given in Table 1. Initial chromosomes are randomly created as shown in Fig 1. Each chromosome contains 10 genes. The numbers 1 and 2 which appear in the first component of the chromosome stand for jobs J_1 and J_2 respectively. According to the chromosome in Fig 2, the second component is [1 1 2 2 2 ...]. The first gene in the second component 1 means that the 1st operation of the 1st job is to be processed by the machine M1. Here, Ch. = chromosome

Ch.	Machine selection					Operation sequence				
	1	4	1	3	2	1	1	2	2	2

Fig. 2: Schematic representation of the chromosome structure

d) *Crossover operation*

The goal of the crossover is to obtain better chromosomes to improve the result by exchanging information contained in the current good ones. In our work we carried out uniform crossover operator to generate new chromosomes.

i. *Machine Selection part*

The crossover operation of MS is performed on two Machine Selection parts and generates two new Machine Selection parts each of which corresponds to a new allocation of operations to machines. Each machine of the new Machine Selection parts must be effective, i.e., the machine must be included in the alternative machine set of the corresponding operation. We adopt uniform crossover (Gao, Sun & Gen, 2008). MS crossover operator only changes some alleles, while their location in each individual i.e., their preceding constraints are not changed. Therefore, the individuals after crossover are also feasible. The procedure could be illustrated in Fig. 3.

ii. *Operation Sequence part*

The crossover operation of OS is different from that of MS. During the past decades, several crossover operators have been proposed for permutation representation. Here we apply a Precedence preserving order-based crossover (POX) for the Operation Sequence. Detailed implementing steps of POX are as follows:

1. Generate two sub-job set Js1/Js2 from all jobs and select two parent individuals as p1 and p2;
2. Copy any allele in p1/ p2 that belong to Js1/Js2 into two child individuals c1/c2, and retain in the same position in c1/c2;
3. Delete the alleles that are already in the sub-job Js1/Js2 from p2/ p1;
4. Orderly fill the empty position in c1/c2 with the alleles of p2/p1 that belongs to in their previous sequence. In Table 1, there are only four jobs. So it is difficult to present the process of POX clearly.

P_1	1	4	1	3	2	1	1	2	2	2
P_2	1	3	4	2	4	2	2	1	2	15
C_1	1	3	1	2	2	2	1	1	2	15
C_2	1	4	4	3	4	1	2	2	2	2

Fig. 3: MS uniform crossover operator

e) *Mutation operator*

Mutation introduces some extra variability into the population to enhance the diversity of population. Usually, mutation is applied with small probability. Large probability may destroy the good chromosome.

i. *Machine Selection part*

The machine selection mutation operation is performed by following steps:

1. MS mutation operator only changes the assignment property of the chromosomes. We select the shortest processing time from alternative machine set to balance the workload of the machines. Taking the chromosome from Fig. 3 for example, MS mutation is described as follows: Select one individual from the population;
2. Read the chromosomes of the individual from left to right and generate a probability value randomly; if all the chromosomes have been read, then end the procedure;
3. If the probability value is less than or equal to the mutation probability then go to Step 4; otherwise, go to Step 2;
4. Select the shortest processing time from the alternative machine set and assign it to the mutation position;

An illustrative instance is shown in Fig. 4. Suppose the mutative operation is O_{23}, before the mutation, O_{23} is processed on M_2, which is the fourth machine in the alternative machine set, so the allele is 2. In the mutation, the rule that selecting the machine of the shortest processing time is obeyed, so M4 is selected, and the allele in the chromosome changes into 4.

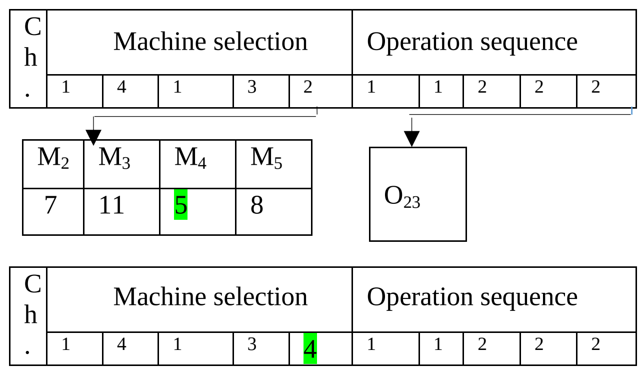

Fig. 4: MS after mutation operation.

IV. Numerical Illustration And Computational Result

The genetic algorithm was coded in ANSI C++ programming language using only simple array data structure and implemented on a PC (Intel(R) Pentium(R), CPU B960 Intel processor of 2.20 GHz).

Table 2: Shows a sample output data set for a flexible job shop scheduling problem

Generation Number	Average Objective Function value (min)	Best Objective Function value (min)
1	113.433	107
2	111.76	103
3	111.375	103
4	111.1	103
5	115.88	103
6	108.21	103
7	107.35	103
8	105.21	103
9	104.83	103
10	104.83	103
11	104.83	103
12	104.16	103
13	103.5	103
14	102.83	101
15	102.85	101
20	101	101

Firstly, we tested the performance of the algorithm using GS method. In Fig. 6 we draw the generation vs time completion curve that shows the decreased average time required to complete each and every operation. The optimal solution for a 4×5 problem can be achieved at 17th generation. In order to obtain meaningful results, we ran our algorithm for several times on different instance. The parameters used in this GA are chosen experimentally in order to get a satisfactory solution in an acceptable time span.

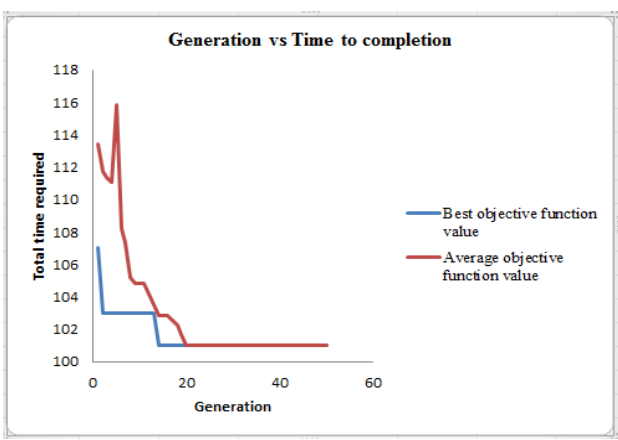

Fig. 6: Decreasing of the total time required

V. Conclusion And Future Study

In this paper, a genetic algorithm for solving the flexible job-shop scheduling problem (FJSP) is proposed. A new chromosome representation scheme is proposed and an effective decoding method interpreting each chromosome into a feasible active schedule is designed. In order to enhance the quality of initial solution, a new initial assignment method i.e., global search technique is used to generate high-quality initial population integrating different strategies to improve the convergence speed and the quality of final solutions. Then different strategies for crossover, selection and mutation operator are adopted. This makes it possible to solve the problem of trade-off resources allocation. Some realistic problems are solved by using this new algorithm. The computational results show that the proposed genetic algorithm leads to an effective scheduling considering time and quality compared with other genetic algorithms which is an indicator of gaining economic profit. The proposed algorithm can enhance the convergence speed and the quality of the solution. This algorithm can be used in solving large size flexible job shop scheduling problem with a focus on achieving economic sustainability throughout the system. In future, it will be interesting to investigate a better search technique which can be developed along with global search technique to generate better results. Also, there is more scope of adjusting an appropriate relation between crossover and mutation probability to enhance the chance of better solution.

Author Declaration

Authors declare that we don't have any conflict of interest. This work has been funded by the Department of Industrial Engineering and Management of Khulna University of Engineering and Technology. All the data and information were collected through the existing lab facilities from this department.

References Références Referencias

1. Ak, B., & Koc, E. (2012). A guide for genetic algorithm based on parallel machine scheduling and flexible job-shop scheduling. *Procedia-Social and Behavioral Sciences, 62*, 817-823.
2. Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multi-purpose machines. *Computing, 45*(4), 369-375.
3. Carlier, J., & Pinson, É. (1989). An algorithm for solving the job-shop problem. *Management science, 35*(2), 164-176.
4. Chan, F., Wong, T., & Chan, L. (2006). Flexible job-shop scheduling problem under resource constraints. *International Journal of Production Research, 44*(11), 2071-2089.
5. Chen, D., Thiede, S., Schudeleit, T., & Herrmann, C. (2014). A holistic and rapid sustainability assessment tool for manufacturing SMEs. *CIRP Annals-Manufacturing Technology, 63*(1), 437-440.
6. Chen, J., Chen, K., Wu, J., & Chen, C. (2008). A study of the flexible job shop scheduling problem

with parallel machines and reentrant process. *The International Journal of Advanced Manufacturing Technology, 39*(3-4), 344-354.
7. Chen, J. C., Wu, C.-C., Chen, C.-W., & Chen, K.-H. (2012). Flexible job shop scheduling with parallel machines using Genetic Algorithm and Grouping Genetic Algorithm. *Expert Systems with Applications, 39*(11), 10016-10021.
8. Davis, L. (1991). Handbook of genetic algorithms.
9. Davis, T. E. (1991). *A simulated annealing like convergence theory for the simple genetic algorithm.* Paper presented at the Proc. the 4th International Conference on Genetic Algorithms.
10. Fattahi, P., Mehrabad, M. S., & Jolai, F. (2007). Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. *Journal of Intelligent Manufacturing, 18*(3), 331-342.
11. Gen, M., Gao, J., & Lin, L. (2009). Multistage-based genetic algorithm for flexible job-shop scheduling problem *Intelligent and evolutionary systems* (pp. 183-196): Springer.
12. Islam, M. M., & AlGeddawy, T. (2017). *An Integrated Framework For Assessment of Sustainable Manufacturing.* Paper presented at the Proceedings of the International Annual Conference of the American Society for Engineering Management.
13. Jahromi, M., & Tavakkoli-Moghaddam, R. (2012). A novel 0-1 linear integer programming model for dynamic machine-tool selection and operation allocation in a flexible manufacturing system. *Journal of Manufacturing Systems, 31*(2), 224-231.
14. Kannegiesser, M., & Günther, H.-O. (2014). Sustainable development of global supply chains-part 1: sustainability optimization framework. *Flexible Services and Manufacturing Journal, 26*(1), 24-47. doi: 10.1007/s10696-013-9176-5
15. Kelly Jr, J. D., & Davis, L. (1991). *A Hybrid Genetic Algorithm for Classification.* Paper presented at the IJCAI.
16. Montana, D. J., & Davis, L. (1989). *Training Feedforward Neural Networks Using Genetic Algorithms.* Paper presented at the IJCAI.
17. Moon, C., Lee, M., Seo, Y., & Lee, Y. H. (2002). Integrated machine tool selection and operation sequencing with capacity and precedence constraints using genetic algorithm. *Computers & industrial engineering, 43*(3), 605-621.
18. Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. *Computers & Operations Research, 35*(10), 3202-3212.
19. Prakash, A., Tiwari, M. K., & Shankar, R. (2008). Optimal job sequence determination and operation machine allocation in flexible manufacturing systems: an approach using adaptive hierarchical ant colony algorithm. *Journal of Intelligent Manufacturing, 19*(2), 161-173.
20. Roshanaei, V., Azab, A., & ElMaraghy, H. (2013). Mathematical modelling and a meta-heuristic for flexible job shop scheduling. *International Journal of Production Research, 51*(20), 6247-6274. doi: 10.1080/00207543. 2013.827806
21. Shen, L., Dauzère-Pérès, S., & Neufeld, J. S. (2018). Solving the flexible job shop scheduling problem with sequence-dependent setup times. *European Journal of Operational Research, 265*(2), 503-516. doi: https://doi. org/10.1016/j.ejor.2017.08.021
22. Xia, W., & Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers & industrial engineering, 48*(2), 409-425.
23. Yazdani, M., Amiri, M., & Zandieh, M. (2010). Flexible job-shop scheduling with parallel variable neighborhood search algorithm. *Expert Systems with Applications, 37*(1), 678-687.
24. Zhang, G., Gao, L., & Shi, Y. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications, 38*(4), 3563-3573.