ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20211022164333
Original scientific paper

Multi-Objective Flexible Job Shop Scheduling Using Genetic Algorithms

Attia BOUDJEMLINE, Imran Ali CHAUDHRY*, Amer Farhan RAFIQUE, Isam A-Q ELBADAWI, Mohamed AICHOUNI, Mohamed BOUJELBENE

Abstract: Flexible Job Shop Scheduling is an important problem in the fields of combinatorial optimization and production management. This research addresses multi-objective flexible job shop scheduling problem with the objective of simultaneous minimization of: (1) makespan, (2) workload of the most loaded machine, and (3) total workload. A general-purpose, domain independent genetic algorithm implemented in a spreadsheet environment is proposed for the flexible job shop. Spreadsheet functions are used to develop the shop model. Performance of the proposed algorithm is compared with heuristic algorithms already reported in the literature. Simulation experiments demonstrated that the proposed methodology can achieve solutions that are comparable to previous approaches in terms of solution quality and computational time. Flexible job shop models presented herein are easily customizable to cater for different objective functions without changing the basic genetic algorithm routine or the spreadsheet model. Experimental analysis demonstrates the robustness, simplicity, and general-purpose nature of the proposed approach.

Keywords: flexible job shop scheduling; genetic algorithms; makespan; multi-objective; scheduling; spreadsheet

1 INTRODUCTION

Job shop scheduling problem (JSSP) is an important practical problem that is considered to be NP-hard, and it is the most typical one of the production scheduling problems. Due to its importance and application researchers have developed various efficient methods. In a general JSSP, j jobs are to be scheduled on m machines; each job has a set of tasks that must be performed on a different machine for a different processing time following a specified routing. Flexible job shop scheduling problem (FJSSP) is an extension of classical job shop and is regarded as *NP*-hard in the strong sense. In a FJSSP, a particular operation can be processed on more than one machine whereby meaning that alternative machines exist to perform the operation. FJSSP therefore comprises two problems. First is to select a suitable machine for an operation from a set of candidate machines, while the second problem is to determine the order in which allocated operations are to be processed on each of the machines.

In the recent years FJSSP has received considerable attention due to its inherent complexity and application. FJSSP was first addressed by Brucker and Schlie [5]. Since then, efficient meta-heuristics have been proposed for the problem.

This research aims to solve multi-objective FJSSP with simultaneous minimization of multiple objective functions namely: (1) makespan, (2) workload of the most loaded machine, and (3) total workload through a general-purpose domain independent GA.

FJSSP models in this research have been developed in Microsoft® Excel™ spreadsheet while the GA routine is implemented thru a proprietary add-in software. It has been shown that proposed methodology is easy to implement and robust. This is considered to be a key aspect of this research. The main advantage of the proposed methodology is that it is general-purpose and domain independent with only minimal changes being required in the spreadsheet model to suit any manufacturing environment rather than the GA routine itself. Similarly, a change in objective function does not warrant a change of the GA routine or the spreadsheet model. Due to the information presented in linked cells, the user of the proposed methodology can easily carry out what-if analysis. This enables a decision maker to look at the effect of change on the schedule due to change in various parameters in a real time environment. Production managers on the shop floor also welcome the spreadsheets as they are quite familiar with these kinds of tools. Additionally, we must consider that the scheduling solutions provided will maybe also be used by non-skilled workers. It is worth mentioning here that in all the problems presented later in the manuscript, the basic GA routine remains the same with the exception of only a slight modification to the shop model in the spreadsheet.

The rest of the paper is organized as follows: problem definition and assumptions are presented in Section 2. Section 3 covers a review of recent literature. Section 4 provides a brief introduction of GA and implementation methodology in this research. Section 5 presents the computational analysis followed by the conclusions in Section 6.

2 PROBLEM DEFINITION & ASSUMPTIONS

FJSSP is more challenging than a classical JSSP as each operation is required to be assigned to a suitable machine from a set of candidate machines that can process the operation and then determine the sequence of operations for each machine. FJSSP addressed in this research is described as follows [32-34]:

1. Manufacturing / production environment has n number of independent jobs to be processed.
2. Each job J_i has a sequence of operations, O_{i1}, O_{i2}, …, O_{in_i} to be performed in a given order one after the other (precedence constraints). All jobs are available at the beginning of the planning horizon.
3. There are m machines to process different operations. Machines are also independent of each other and are available at the beginning of the planning horizon.
4. In terms of flexibility, the FJSSP can be grouped into two categories i.e. partial flexibility and total flexibility [17]. If an operation is only performed on a subset of m machines, it is termed as partial flexibility, whereas in case of total flexibility any of the m machines can process all operations.
5. Other assumptions for the FJSSP problem are as follows:

- Processing times of operations are known and fixed.
- At any given time, one and only one machine can perform an operation.
- An operation once started cannot be interrupted.
- Processing times of operations of a job are machine dependent.
- Machine setting up and job movement time from one machine to another is insignificant and included in processing time.
- Machines do not break down.
- Operations of different jobs do not have precedence constraints.

6. Objective. Assignment of each operation to a machine from candidate machines and then to determine the order of operations on each machine to simultaneously minimize:
   - C_{max}: makespan or maximum completion time of the schedule. Makespan indicates general throughput of the system. Lower values of makespan means that the scheduler is providing good and efficient planning of tasks to the resources.
   - W_M: workload of the most loaded machine or critical machine workload. This objective prevents too much work assignment by balancing the workload among all machines.
   - W_T: total workload or the total working time of all machines. This objective improves economic efficiency by assigning the machine with relatively small processing time for each of the operation.

Tackling multi-objective optimization problems can be classified in three broad categories [15]:

1. Weighted sum approach. This transforms multi-objective optimization problem into a mono-objective optimization problem by assigning weights to all objectives based on their importance.
2. Non-Pareto approach. Deals with different objectives in a separated way.
3. Pareto approach. Based on Pareto optimality concept.

When dealing with multi-objective optimization, weighted sum approach offers advantage of easy understanding, convenience in application and modification [32]. Weighted sum multi-objective approach has also been used for scheduling applications by other researchers, e.g. Karthikeyan et al. [18] and Rajkumar et al. [21]. In this research, we follow a weighted sum approach and assign weights to the three above mentioned objectives to transform them into a mono-objective problem i.e.:

$$OBJ = 0.5\ C_{max} + 0.3\ W_M + 0.2\ W_T \quad (1)$$

In this study weights have been selected as suggested by Xing et al. [32]. Weights demonstrate relative importance of each of the components of objective function. Importance for each component can be categorised into three classes: very important, important and unimportant. The more attention required for a certain objective, the larger the weight of that objective.

As mentioned by Xing et al. [32], C_{max}, W_M and W_T are assigned weights of 0.5, 0.3 and 0.2 respectively based on their importance.

3 LITERATURE REVIEW

Brucker and Schlie [5] first addressed two-job FJSSP and proposed a polynomial algorithm. A detailed review of application of various techniques has been presented by Chaudhry and Khan [7] and Türkyılmaz et al. [27].

Table 1 Literature review of recent research articles

Literature	Problem Characteristics	Type of Objective Function	Objective Function	Optimization Method
Zhu and Zhou [37]	FJSSP with due dates and uncertain processing time	Multi-objective	Simultaneous minimization of makespan, average tardiness and total machine workload	Parallel Multi-objective multi-micro-swarm leadership hierarchy-based optimization algorithm
Xu et al. [35]	Distributed FJSSP	Multi-objective	Minimization of makespan, quality, costs and carbon emission	GA hybridized with tabu search method and use of fuzzy AHP to transform multi-objective into a single objective function
Wei et al. [30]	Resource constrained FJSSP	Single	Minimization of makespan / makespan expectation / operation delay / cycle delay	Multi-objective GA combined with whale optimization algorithm
Wang et al. [29]	FJSSP with transportation times preventive maintenance	Multi-objective	Minimization of makespan and total energy consumption	Multi-region division sampling strategy integrated with a GA and a differential evolution algorithm
Sun et al. [23]	Low carbon FJSSP	Multi-objective	Minimization of makespan, carbon emission, and machine load	Coevolutionary non-dominated sorting GA - III with multi-crossover operator and natural selection
Sun et al. [22]	FJSSP with transportation and setup times	Multi-objective	Minimization of critical machine workload, total workload and makespan	Many-objective evolutionary algorithm hybridized and tabu search
Hongyu and Xiuli [14]	Dual resource sustainable FJSSP	Multi-objective	Simultaneous minimization of makespan, energy consumption and ergonomic risk	Improved survival duration-guided NSGA - III algorithm
Caldeira and Gnanavel Babu [6]	FJSSP	Multi-objective	Minimization of critical machine workload, total workload and makespan	Discrete Jaya algorithm
Bissoli et al. [4]	FJSSP	Multi-objective	Minimization of critical machine workload, total workload and makespan	Clustering algorithm using lexicographic classification of the objectives
Abderrabi et al. [1]	FJSSP with setup times and job splitting	Single	Total flow time	Two different methods: genetic algorithm & iterative local search

Due to large amount of literature available on the topic, only some of the recent papers have been reviewed. Reviewed literature is presented in Tab. 1.

Manuscripts reviewed above have used different algorithms. However, each algorithm has its own advantages and disadvantages. There is not any single optimizer which is suitable for all classes and types of problems [31]. None of the metaheuristics guarantee a global optimal solution, unless near infinite iterations are made. Therefore, the idea to use GA in this research was its ease of implementation in the spreadsheet environment.

In this research, flexible job shop scheduling problem is solved through a general-purpose domain-independent genetic algorithm. The main distinction between the approaches reviewed above and this manuscript is in terms of ease of use and presentation of schedules in a tabular form, that is easily understandable by a practitioner on a shop floor. Further, as mentioned earlier, the basic GA routine or objective function does not require a change with a change in the shop environment and also the ability of the proposed methodology to carry out easy what-if analysis. EXCEL® VBA can also be used to develop Gantt Charts, thus giving a graphical representation for a shop floor worker.

3 GENETIC ALGORITHMS (GA)

GA is a stochastic search method used for optimization problems and derived from natural evolution process based on the mechanisms of natural selection. Since first introduced by Holland [13], GAs have been successfully used by numerous researchers to provide good solutions to many complex problems in different fields. GAs are particularly suitable if the objective function being optimized is multi-modal or there are irregular search spaces as GAs are quite robust to transverse a large search space quickly.

Davis [8] first applied GA to a simple job shop problem. Since then, several researchers have applied GAs to numerous problems in the manufacturing environment.

Built-in functions and formulae of Microsoft EXCEL™ spreadsheet are used to develop the FJSSP model. The GA is then run to find the optimized schedule for a given objective function.

3.1 Chromosome Representation

For the FJSSP in this research, chromosome has two parts. First part of the chromosome represents the operations, where the chromosome length is a vector equal to the sum of operations of all the jobs while the machine associated with each operation is given in the second part of the chromosome. The order of the machines in the second part of the chromosome corresponds to the order of the operations in the first part. Permutation representation is used for the operations. Any sequence not meeting the requirement of preceding constraints is repaired automatically by the GA to produce a valid schedule.

For each gene of machine assignment block of the chromosome an integer number between 1 and total number of given machines is randomly generated for each related operation. This is demonstrated by a 4-job and 4-machine example given in Tab. 2.

The resulting chromosome for the sample data is as given in Fig. 1. The sample data in Fig. 1, 31, 32 and 33 represent the first, second and third operations of job 3 respectively. The first and second parts of the chromosome are linked to each other through spreadsheet formulas. For example, job 2 - operation 1 represented by 21 is to be processed on machine 3; similarly job 4 - operation 3 represented by 43 is to be processed on machine 3. Example in Tab. 2 is a case of partially FJSSP where some of the operations can be processed only on a subset of machines. A partial FJSSP is transformed to a total flexible FJSSP by assigning a very high processing time for the machines that cannot perform a particular operation. For example, machines 1 and 4 cannot process job 2-operation 1 (denoted by 21); therefore, to handle this situation a large process time of 99 would be assigned to machines 1 and 4 to prevent the GA to select this operation.

3.2 Reproduction/Selection

A steady state approach is used in the GA uses for reproduction. As compared to other approaches, steady state approach produces only one child solution after the crossover. The worst performing organism of the population is replaced by the child solution, if the fitness of the child solution is better than the organisms already present in population. If not, then the child solution is discarded thus replicating survival of the fittest strategy. Measure of fitness for a chromosome is the value of the objective function i.e., makespan etc.

Parents for crossover operation are chosen by a rank-based mechanism [2]. Rank-based mechanism is based on probabilities. Selection probabilities are defined by the relative position or rank of a chromosome in the population. Rank-based selection offers a smoother selection probability curve.

Table 2 A 4-job, 4-machine flexible job shop problem

Jobs	1			2		3			4		
Operations	1	2	3	1	2	1	2	3	1	2	3
Machines available to perform operation	M1-M3	M1-M4	M3-M4	M2-M3	M1-M2	M2-M3-M4	M1-M2	M1-M2-M3	M1-M2-M4	M2-M3-M4	M2–M3

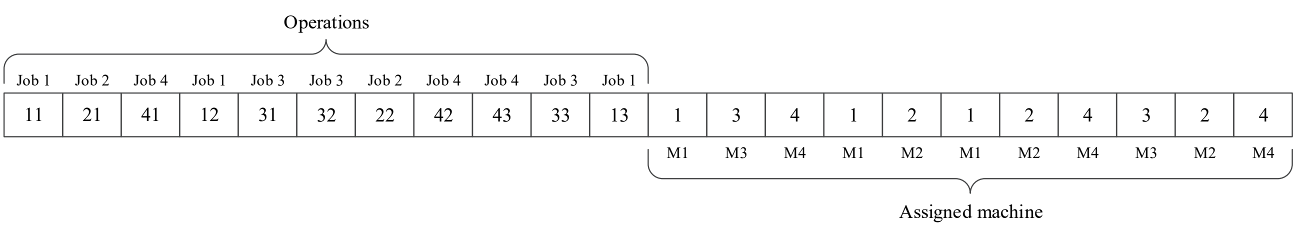

Figure 1 Chromosome representation for sample data

3.3 Crossover Operator

Crossover operator takes the best characteristics from each of the parents to produce a new solution. Crossover operation occurs according to a user-defined crossover rate which is set by the user in the run settings dialog box before the start of the simulation. Crossover rate can have any value between 0 and 1.

For the operations part of the chromosome, an order crossover [9] is used. Order crossover works well with permutation representation as it preserves the relative order of the genes without violating the precedence constraints. Order crossover uses a zero-one template to determine the genes that would be contributed by each of the parent to make the offspring. Wherever the binary template contains "1", the genes are copied from Parent 1 in the offspring at the same position as they appear in Parent 1. While genes in Parent 1 associated with "0" in the binary template appear in the same order in offspring as they appear in Parent 2. An example order crossover is given in Fig. 2.

In Fig. 2, the genes C, D, E, G and H at position 3, 4, 5, 7 and 8 respectively in P1 are associated with binary digit "1". These are inherited by the child solution in the same positions as they appear in P1. Remaining genes associated with digit "0" are A, B, F and I at positions 1, 2, 6 and 9 respectively. These genes appear in the child solution in the same order as they appear in P2.

Position:	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
Parent 1 (P1):	A	B	C	D	E	F	G	H	I
Binary Template:	*0*	*0*	*1*	*1*	*1*	*0*	*1*	*1*	*0*
Parent 2 (P2):	H	B	C	E	D	F	A	G	I
Offspring (O):	B	F	C	D	E	A	G	H	I

Figure 2 Order-based crossover

For machine assignment case i.e. last eleven genes of the chromosome in Fig. 1, uniform crossover (Syswerda [24]) is implemented. In this case, crossover rate or mixing ratio determines the parent that will contribute to each gene in the child solution. Consider the two parents in Fig. 3. P1 is coloured green while P2 is coloured yellow. If crossover rate is 0.6, approximately sixty percent of the genes in the offspring will come from Parent 1 while the other forty percent will be contributed by Parent 2. Corresponding to the crossover rate, a random mask of 0-1 is generated below the P2. If the corresponding bit is "0", the gene is taken from P1, or from P2 if corresponding bit is "1".

Parent P1	0	0	1	1	0	1	0	1
Parent P2	1	1	0	1	1	0	0	0
Mask	0	1	1	0	1	0	0	1
Child	0	1	0	1	1	1	0	0

Figure 3 Uniform crossover

The new solution may be modified by the GA routine to generate a valid solution if it does not meet the precedence constraints.

3.4 Mutation Operator

With each successive operation the population loses diversity in the solutions. In order to tackle such situation, mutation operation is carried out so that diversity is maintained thus preventing the population from getting trapped at a local optimum. Mutation is performed by randomly swapping the positions of two tasks for the operations block of the chromosome. In case swapping of tasks violates any precedence constraint, the chromosome is repaired by the GA routine to adhere to precedence constraints. Within the machine assignment block a random number is generated between 1 and m machines for each block or variable. Swapping of jobs in operations block and random number generation for machine block depends on the mutation rate that is defined before the start of the simulation.

4 COMPUTATIONAL ANALYSIS

Effectiveness and performance of the proposed methodology is illustrated by solving test problem instances from the literature. Following parameters of GA are used for the simulation experiments: population size = 65, crossover rate = 0.65 and mutation rate = 0.05. All problem instances in subsequent sections were run with the same set of parameters.

4.1 Instance 1

Instance 1 consists of three problems taken from Thomalla [26]. Problem 1 has 3-jobs and 3-machines where each job has 3-operations. All operations can be processed on all available machines with different speeds. Problem 2 is a 4-job, 3-machine problem, each job having 4-operations. While problem 3 is 6-job, 10-machine problem with each job having 6-operations. Two objective functions considered for these three problems are sum of tardiness and sum of quadratic tardiness. Sum of quadratic tardiness is given by:

$$Z^{*} = \min \left\{ \sum_{i=1}^{N_{\text{job}}} \omega_i T_i^2 \right\} \tag{2}$$

where,
ω_i = weight of job A_i
T_i = tardiness of job A_i

Weights of the jobs have been set equal to 1, while the due dates are set to the start of the planning horizon. The comparative result of proposed GA with previous approaches by Thomalla [26] for sum of tardiness and by Thomalla [26] for sum of quadratic tardiness are given in Tab. 3 and Tab. 4 respectively.

For minimization of quadratic sum of tardiness, the proposed GA produced the same results for Problem 1 and 2. For problem 3, the proposed GA found better result for sum of quadratic tardiness value. The proposed GA found the sum of quadratic tardiness value of 467 093 with a root mean square (RMS) of 279 as compared to heuristic solution of 514 306 with RMS value of 292.7 thus providing a better solution.

4.2 Instance 2

Instance 2 is taken from Baskak and Erol [3]. This instance is a real-world problem taken from an electrical motor factory. The problem has 12 jobs to be processed on 9 machines. Machines are grouped into 6 work centres. Machine grouping by work centres is shown in Tab. 5. Previously, forward scheduling was being used, so that heavy jobs were sequentially processed first on work centre A, B, ..., F. Baskak and Erol used GA to find the makespan. The proposed methodology found the makespan value of 336 mins as compared to 474 and 374 found by forward scheduling and GA by Baskak and Erol [3]. Job processing order and work centre/machine assignment is given in Tab. 6 for the three approaches.

4.3 Instance 3

Instance 3 is a small-scale problem with 4-jobs and 5-machines (12 operations) taken from [17]. Instance is a total flexibility case where all five machines can process all jobs.

4.4 Instance 4

Instance 4 is an 8-job, 8-machine problem having total of 27 operations [17]. This instance is a case of partial flexible FJSSP. For the current methodology, partial flexibility is converted to total flexibility by assigning a large processing time to an operation, such as 999 thus ensuring that the machine is not selected by the GA for this operation.

4.7 Instance 7

Instance 7, a case of total flexibility, has 15-jobs to be processed on 10-machines with 56 operations [17].

4.8 Comparison of Previous Solution Approaches for Instance 3-7 with Proposed GA

The performance of proposed GA methodology is compared with previously reported approaches for the minimization of objective function given in Eq. (1). The comparative analyses for undermentioned meta-heuristics are given in Tab. 7.

- A-1: Hybrid tabu search algorithm [19]
- A-2: Newton-based heuristic algorithm [20]
- A-3: Hybrid Immune Algorithm & SA [12]
- A-4: Teaching – Learning-based hybrid GA & PSO (Huang et al. [16])
- A-5: Hybrid algorithm based on shuffled frog leaping algorithm & PSO [11]
- A-6: Improved NSGA-II algorithm [36]
- A-7: Mixed integer programming model [25]
- A-8: Discrete firefly algorithm [25]
- A-9: GA by Tessaro Lunardi et al. [25]
- A-10: Multi-objective memetic algorithm based on decomposition [28]
- A-11: Bee evolutionary guiding NSGA-II [10]

For instances 3, 4 and 5, the proposed GA produced the same solution as other approaches. For instance 6, the solution obtained by proposed GA was worse by 0.74%. Only for instance 7, proposed GA was not able to find better solution as compared to other approaches.

Table 3 Comparison of results for minimization of sum of tardiness objective

Problem	Exact B&B	Lagrange method	Slack rule	Heuristic B&B1	Weighted loss of slack	Earliest due date	Heuristic B&B2/3	Proposed GA	Optimum
Problem 1	328	328	388	332	347	342	328	328	328
Problem 2	369	369	491	417	410	410	369	369	369
Problem 3	?	1 754	2 402	1 913	2 209	2 032	1 754	1 557	?

Table 4 Comparison of results for minimization of quadratic sum of tardiness objective

Problem	Proposed GA	Heuristic Value	Lagrange Value	Sum of tardiness	Avg. tardiness	Max. tardiness	RMS
Problem 1	36 424	36 424	36 416.46	328	109.33	120	110,19
Problem 2	36 679	36 679	35 733.13	369	92.25	127	95.76
Problem 3	467 093	514 306	489 102.3	1 557/1 754	259.5/ 92.3	335/316	279/292.7

Table 5 Work centers and Machines

<table>
<tr><td rowspan="3">Work Center A</td><td>Machine 1</td><td rowspan="2">Work Center B</td><td>Machine 4</td><td>Work Center D</td><td>Machine 7</td></tr>
<tr><td>Machine 2</td><td>Machine 5</td><td>Work Center E</td><td>Machine 8</td></tr>
<tr><td>Machine 3</td><td>Work Center C</td><td>Machine 6</td><td>Work Center F</td><td>Machine 9</td></tr>
</table>

Table 6 Comparative machine assignment and process ordering for three approaches

Work Centers	Machines	Job Processing Order by Forward Scheduling	Job Processing Order by Baskak et al. (2005)	Job Processing Order by Proposed GA
A	1	3-9-2-6-11	1-4-10-11-2	10-1-6-9-2
	2	8-10-12-1	3-12-9-8	3-4-8
	3	4-5	5-6	5-12-11
B	4	2-7-6-12-1-9-3-8	7-2-3-12-9-8	11-2-3-4-12-9
	5	11-4	11-1-4-6	6-1-7-8
C	6	9-4-7-10-2-3-8-11-5-12	9-7-3-4-10-11-8-2-5-12	10-9-3-11-4-5-8-2-7-12
D	7	6-3-8-11-10-1-2-5-9-12	6-3-11-1-10-5-8-12-9-2	6-10-3-11-5-1-8-2-12-9
E	8	5-1-7-12-11-4-6-10-9-3	5-1-11-10-7-4-12-9-6-3	5-10-6-1-11-4-12-3-7-9
F	9	5-7-1-3-4-6-8-10-11	5-3-10-11-1-4-7-8-6	5-10-7-6-3-1-4-8-11
Solution Value / mins		**474**	**374**	**336**

Table 7 Comparative results of various heuristics and Proposed GA

Instance	Obj.	A-1	A-2	A-3	A-4	A-5	A-6	A-7	A-8	A-9	A-10	A-11	GA
4 × 5	C_{max}	**12**	**12**	11	-	-	-	**12**	11	**12**	**12**	**12**	**12**
	W_T	**32**	**32**	32	-	-	-	**32**	32	**32**	**32**	**32**	**32**
	W_M	**8**	**8**	10	-	-	-	**8**	10	**8**	**8**	**8**	**8**
	OBJ	**14.8**	**14.8**	14.9	-	-	-	**14.8**	14.9	**14.8**	**14.8**	**14.8**	**14.8**
8 × 8	C_{max}	**14**	**14**	15	14	16	16	**14**	**14**	**14**	**14**	**14**	**14**
	W_T	**77**	**77**	75	77	73	73	**77**	**77**	**77**	**77**	**77**	**77**
	W_M	**12**	**12**	12	12	13	12	**12**	**12**	**12**	**12**	**12**	**12**
	OBJ	**26**	**26**	26.1	27	26.5	26.2	**26**	**26**	**26**	**26**	**26**	**26**
10 × 7	C_{max}	**11**	11	-	-	-	-	**11**	**11**	**11**	**11**	**-**	**11**
	W_T	**62**	61	-	-	-	-	**62**	**62**	**62**	**62**	**-**	**62**
	W_M	**10**	11	-	-	-	-	**10**	**10**	**10**	**10**	**-**	**10**
	OBJ	**20.9**	21	-	-	-	-	**20.9**	**20.9**	**20.9**	**20.9**	**-**	**20.9**
10 × 10	C_{max}	**7**	7	7	7	7	-	**7**	7	**7**	**7**	**7**	7
	W_T	**43**	42	42	42	42	-	**43**	42	**43**	**43**	**43**	42
	W_M	**5**	6	6	6	6	-	**5**	6	**5**	**5**	**5**	6
	OBJ	**13.6**	13.7	13.7	13.7	13.7	-	**13.6**	13.7	**13.6**	**13.6**	**13.6**	13.7
15 × 10	C_{max}	**11**	12	12	12	**11**	-	11	12	11	**11**	**11**	14
	W_T	**91**	92	92	91	**91**	-	93	93	93	**91**	**91**	95
	W_M	**11**	12	11	11	**11**	-	11	12	11	**11**	**11**	13
	OBJ	**27**	28	27.7	27.5	**27**	-	27.4	28.2	27.4	**27**	**27**	29.9

5 CONCLUSIONS

In this paper, a general-purpose domain independent GA methodology was presented to solve FJSSP. The shop model and the proposed GA are coded within a spreadsheet environment which can address both total and partial flexibility conditions. The proposed GA approach can be used to find any objective function without any change in the shop model or basic GA routine. This aspect was demonstrated by attempting various problems taken from the literature. Different objective functions were used to demonstrate the effectiveness and general-purpose & domain independent nature of the proposed GA. For instance 1 to 3, GA was able to find better or same solution as compared to previous approaches. All problems attempted in this paper have been compared with already reported heuristics. For small to medium sized problems the proposed GA method was able to obtain optimal solutions while for larger problems though the GA was not able to obtain better results, however, it produced results better than some of the already reported heuristics. The shop model is easily customizable to cater for additional constraints that may be imposed on the shop. Furthermore, spreadsheet environment also enables a person to carry out what-if analysis, enabling a decision maker to look at the effect of change on the schedule due to change in various parameters in a real time environment.

The final experimental results demonstrate that the proposed methodology is an effective and feasible methodology for the multi-objective FJSSP. The key advantage of GA presented in this methodology is that it provides a general-purpose solution to the scheduling problem that is not problem specific, with the peculiarities of any particular scenario being accounted for in fitness function without disturbing the logic of the standard optimization routine. Spreadsheet based environment also makes this approach well suited for production managers and shop floor workers.

Acknowledgements

This research has been funded by Scientific Research Deanship at University of Ha'il - Saudi Arabia through Project number RG - 20 028.

6 REFERENCES

[1] Abderrabi, F., Godichaud, M., Yalaoui, A., Yalaoui, F., Amodeo, L., Qerimi, A., & Thivet, E. (2021). Flexible Job Shop Scheduling Problem with Sequence Dependent Setup Time and Job Splitting: Hospital Catering Case Study. *Applied Sciences*, *11*(4). https://doi.org/10.3390/app11041504

[2] Baker, J. E. (1985). Adaptive Selection Methods for Genetic Algorithms. *Proceedings of the 1st International Conference on Genetic Algorithms*, 101-111.

[3] Baskak, M. & Erol, V. (2005). A genetic algorithm approach for solving flexible job shop scheduling problems. *35th International Conference on Computers & Industrial Engineering*, 245-250.

[4] Bissoli, D. C., Zufferey, N., & Amaral, A. R. S. (2021). Lexicographic optimization-based clustering search metaheuristic for the multiobjective flexible job shop scheduling problem. *International Transactions in Operational Research*, *28*(5), 2733-2758. https://doi.org/10.1111/itor.12745

[5] Brucker, P. & Schlie, R. (1990). Job-shop scheduling with multi-purpose machines. *Computing*, *45*(4), 369-375. https://doi.org/10.1007/BF02238804

[6] Caldeira, R. H. & Gnanavel Babu, A. (2021). A Pareto based discrete Jaya algorithm for multi-objective flexible job shop scheduling problem. *Expert Systems with Applications*, 170. https://doi.org/10.1016/j.eswa.2021.114567

[7] Chaudhry, I. A. & Khan, A. A. (2016). A research survey: review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, *23*(3), 551-591. https://doi.org/10.1111/itor.12199

[8] Davis, L. (1985). Job Shop Scheduling with Genetic Algorithms. *Proceedings of the 1st International Conference on Genetic Algorithms*.

[9] Davis, L. (1991). *Handbook of Genetic Algorithms*. New York, USA, Van Nostrand Reinhold.

[10] Deng, Q., Gong, G., Gong, X., Zhang, L., Liu, W., & Ren, Q. (2017). A Bee Evolutionary Guiding Nondominated Sorting Genetic Algorithm II for Multiobjective Flexible Job-Shop Scheduling. *Computational Intelligence and Neuroscience*, *2017*, 1-21. https://doi.org/10.1155/2017/5232518

[11] Gou, M., Gao, Q., & Yang, S. (2018). Improved Shuffled Frog Leaping Algorithm for Multi-objection Flexible Job-Shop Scheduling Problem. *Intelligent Computing and Internet of Things (ICSEE 2018, IMIOT 2018).* https://doi.org/10.1007/978-981-13-2384-3_1

[12] Habibi, Y., Swalaganata, G., & Yustita, A. D. (2017). Penyelesaian Multi-Objective Flexible Job Shop Scheduling Problem Menggunakan Hybrid Algoritma Imun. *Jurnal Teknosains*, *6*(2), 79-94. https://doi.org/10.22146/teknosains.22901

[13] Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI, University of Michigan Press.

[14] Hongyu, L. & Xiuli, W. (2021). A survival duration-guided NSGA-III for sustainable flexible job shop scheduling problem considering dual resources. *IET Collaborative Intelligent Manufacturing*, *3*(2), 119-130. https://doi.org/10.1049/cim2.12003

[15] Hsu, T., Dupas, R., Jolly, D., & Goncalves, G. (2002). Evaluation of mutation heuristics for solving a multiobjective flexible job shop by an evolutionary algorithm. *IEEE International Conference on Systems, Man and Cybernetics*, 6. https://doi.org/10.1109/ICSMC.2002.1176444

[16] Huang, X., Guan, Z., & Yang, L. (2018). An effective hybrid algorithm for multi-objective flexible job-shop scheduling problem. *Advances in Mechanical Engineering*, *10*(9), 1-14. https://doi.org/10.1177/1687814018801442

[17] Kacem, I., Hammadi, S., & Borne, P. (2002). Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic. *Mathematics and Computers in Simulation*, *60*(3-5), 245-276. https://doi.org/10.1016/S0378-4754(02)00019-8

[18] Karthikeyan, S., Asokan, P., & Nickolas, S. (2014). A hybrid discrete firefly algorithm for multi-objective flexible job shop scheduling problem with limited resource constraints. *The International Journal of Advanced Manufacturing Technology*, *72*(9-12), 1567-1579. https://doi.org/10.1007/s00170-014-5753-3

[19] Li, J. Q., Pan, Q. K., & Liang, Y. C. (2010). An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, *59*(4), 647-662. https://doi.org/10.1016/j.cie.2010.07.014

[20] Pérez, M. A. F. & Raupp, F. M. P. (2016). A Newton-based heuristic algorithm for multi-objective flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing*, *27*(2), 409-416. https://doi.org/10.1007/s10845-014-0872-0

[21] Rajkumar, M., Asokan, P., Anilkumar, N., & Page, T. (2011). A GRASP algorithm for flexible job-shop scheduling problem with limited resource constraints. *International Journal of Production Research*, *49*(8), 2409-2423. https://doi.org/10.1080/00207541003709544

[22] Sun, J., Zhang, G., Lu, J., & Zhang, W. (2021). A hybrid many-objective evolutionary algorithm for flexible job-shop scheduling problem with transportation and setup times. *Computers & Operations Research*, 132. https://doi.org/10.1016/j.cor.2021.105263

[23] Sun, X., Wang, Y., Kang, H., Shen, Y., Chen, Q., & Wang, D. (2021). Modified Multi-Crossover Operator NSGA-III for Solving Low Carbon Flexible Job Shop Scheduling Problem. *Processes*, 9(1), 1-20. https://doi.org/10.3390/pr9010062

[24] Syswerda, G. (1989). Uniform Crossover in Genetic Algorithms. *Proceedings of the 3rd International Conference on Genetic Algorithms*, 9.

[25] Tessaro Lunardi, W., Cherri, L. H., & Voos, H. (2018). A Mathematical Model and a Firefly Algorithm for an Extended Flexible Job Shop Problem with Availability Constraints. *Artificial Intelligence and Soft Computing (ICAISC 2018)*. https://doi.org/10.1007/978-3-319-91253-0_51

[26] Thomalla, C. S. (2001). Job shop scheduling with alternative process plans. *International Journal of Production Economics*, *74*(1-3), 125-134. https://doi.org/10.1016/S0925-5273(01)00119-0

[27] Türkyılmaz, A., Şenvar, Ö., Ünal, İ., & Bulkan, S. (2020). A research survey: heuristic approaches for solving multi objective flexible job shop problems. *Journal of Intelligent Manufacturing*, *31*(8), 1949-1983. https://doi.org/10.1007/s10845-020-01547-4

[28] Wang, C., Ji, Z., & Wang, Y. (2017). A Novel Memetic Algorithm Based on Decomposition for Multiobjective Flexible Job Shop Scheduling Problem. *Mathematical Problems in Engineering*, 1-20. https://doi.org/10.1155/2017/2857564

[29] Wang, H., Sheng, B., Lu, Q., Yin, X., Zhao, F., Lu, X., Fu, G. et al. (2021). A novel multi-objective optimization algorithm for the integrated scheduling of flexible job shops considering preventive maintenance activities and transportation processes. *Soft Computing*, *25*(4), 2863-2889. https://doi.org/10.1007/s00500-020-05347-z

[30] Wei, F. F., Cao, C. Y., & Zhang, H. P. (2021). An Improved Genetic Algorithm for Resource-Constrained Flexible Job-Shop Scheduling. *International Journal of Simulation Modelling*, *20*(1), 201-211. https://doi.org/10.2507/IJSIMM20-1-CO5

[31] Wolpert, D. H. & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82. https://doi.org/10.1109/4235.585893

[32] Xing, L.-N., Chen, Y.-W., & Yang, K.-W. (2009). Multi-objective flexible job shop schedule: Design and evaluation by simulation modeling. *Applied Soft Computing*, *9*(1), 362-376. https://doi.org/10.1016/j.asoc.2008.04.013

[33] Amjad, M. K., Butt, S. I., Anjum, N., Chaudhry, I. A., Faping, Z., & Khan, M. (2020). A layered genetic algorithm with iterative diversification for optimization of flexible job shop scheduling problems. *Advances in Production Engineering & Management*, *15*(4), 377-389. https://doi.org/10.14743/apem2020.4.372

[34] Ma, D. Y., He, C. H., Wang, S. Q., Han, X. M., & Shi, X. H. (2018). Solving fuzzy flexible job shop scheduling problem based on fuzzy satisfaction rate and differential evolution. *Advances in Production Engineering & Management*, *13*(1), 044-056. https://doi.org/10.14743/apem2018.1.272

[35] Xu, W., Hu, Y., Luo, W., Wang, L., & Wu, R. (2021). A multi-objective scheduling method for distributed and flexible job shop based on hybrid genetic algorithm and tabu search considering operation outsourcing and carbon emission. *Computers & Industrial Engineering*, 157. https://doi.org/10.1016/j.cie.2021.107318

[36] Yang, T., Li, J., & Chen, H. (2018). Multi-objective Flexible Job-shop Scheduling Problems with Limited Resource Constraints Using Nondominated Sorting Genetic Algorithm II. *2018 IEEE International Conference on Information and Automation (ICIA)*. https://doi.org/10.1109/ICInfA.2018.8812599

[37] Zhu, Z. & Zhou, X. (2021). A multi-objective multi-micro-swarm leadership hierarchy-based optimizer for uncertain flexible job shop scheduling problem with job precedence constraints. *Expert Systems with Applications*, 182. https://doi.org/10.1016/j.eswa.2021.115214

Contact information:

Attia BOUDJEMLINE, Assistant Professor
Industrial Engineering Department, College of Engineering
University of Ha'il, Saudi Arabia
E-mail: a_boudjemline@hotmail.com

Imran Ali CHAUDHRY, Professor
(Corresponding author)
Industrial Engineering Department, College of Engineering,
University of Ha'il, Saudi Arabia
E-mail: i.chaudhry@uoh.edu.sa

Amer Farhan RAFIQUE, Associate Professor
King Abdulaziz University,
Jeddah, Saudi Arabia
E-mail: afrafique@gmail.com

Isam A-Q ELBADAWI, Professor
Industrial Engineering Department, College of Engineering,
University of Ha'il, Saudi Arabia
E-mail: isam149@gmail.com

Mohamed AICHOUNI, Professor
Industrial Engineering Department, College of Engineering,
University of Ha'il, Saudi Arabia
E-mail: m.aichouni@uoh.edu.sa

Mohamed BOUJELBENE, Associate Professor
Industrial Engineering Department, College of Engineering,
University of Ha'il, Saudi Arabia
E-mail: mboujelbene@yahoo.fr