Integrated Approach for Flexible Job Shop Scheduling Using Multi-objective Genetic Algorithm

Sunita Nayak, Anoop Kumar Sood, and Abhishek Pandey

Abstract The study proposes a genetic algorithm-based integrated approach where the selection of suitable machine and sequencing of operations in machines are performed simultaneously in a flexible job shop-based environment. For this, chromosome representation chosen is simple to code and decode and always results in a feasible solution on the application of genetic operators. In true sense to real production problems, multiple objectives are considered which will reduce the total production time of the entire batch with efficient machine utilization. The proposed solution is evaluated on test data, and it was shown that the algorithm is giving equivalent and in some cases better results in comparison with existing methodologies.

Keywords FJSP · GA · Pareto front · Makespan · Maximum machine workload · Total machine workload

S. Nayak
Department of Mechanical Engineering, Bhubanananda Odisha School of Engineering, Cuttack, Odisha 753007, India

A. K. Sood (✉)
Department of Manufacturing Engineering, National Institute of Foundry and Forge Technology, Ranchi 834003, India

A. Pandey
Department of Mechanical Engineering, ABES Engineering College, Ghaziabad, Uttar Pradesh 201009, India
e-mail: abhishek.pandey@abes.ac.in

K. Govindan et al. (eds.), *Advances in Mechanical and Materials Technology*,
Lecture Notes in Mechanical Engineering,
https://doi.org/10.1007/978-981-16-2794-1_35

1 Introduction

Flexible job shop scheduling (FJS) problem arises in a job shop-based production environment where multiple machines are available for performing a particular operation. This additional flexibility demands two types of decisions to be made, namely selection of appropriate machine from the available machines for performing an operation on a job and then finding an appropriate sequence of operations of same or different jobs to be performed in a particular machine to achieve some desire objectives. The first type of decision comes under the routing subproblems, and the second type of decision falls under the scheduling subproblems of production planning domain. In line with its predecessor job shop scheduling problem (JSP), which only concerns the scheduling of operations, it is combinatorial NP-hard problem, but the presence of routing makes it more difficult to solve in comparison with JSP.

Several alternative approaches have been suggested in the literature to tackle FJS problems (FJSP) which can be characterized as heuristic and meta-heuristic solution methodologies. Heuristic approaches may guarantee the optimal solution but that may not be true if large size problem instances are considered. Whereas high-level strategies known as meta-heuristics have produced optimal solutions within a reasonable computational time for a different size of problem instances [1]. These meta-heuristics use either a hierarchical or a combination approach to solve FJSP. The hierarchical approach decomposes the FJSP into routing and sequencing subproblems. Routing can be done using simple dispatching rules or a combination of them, and then some other higher-order heuristic can be used to solve the sequencing subproblem. In a combination approach, both types of subproblems are combined and solved simultaneously [2]. Although integration is difficult to implement, the quality of solutions is better than the hierarchical approach. In this direction, GA [2], PSO [3, 4], and ACO [5] are effective meta-heuristics which can exploit a lot of domain knowledge and implemented to solve the FJSP. GA is the most popular meta-heuristics technique as it explores solution space more diversely by using the population of solutions and provides resistance to premature convergence on local minima. Recently, more and more papers are concentrating on GA [6] or hybrid GA [7] with different population representation and generation strategies.

Most of these approaches concentrated on determining an optimum schedule which satisfies a single objective [4, 7]. The real-world production scenario demands production of a complete batch in minimum time and proper utilization of all the available machines so that no single machine is under-utilized or over-utilized. The presence of these multiple objectives put additional difficulty in FJSP solution methodology [8–10].

In this regard, the present study proposes the use of simple GA to solve multi-objective FJSP. As per the combinatorial nature of FJSP, solution coding and encoding are incorporated using problem-specific knowledge to obtain feasible solutions. In the true sense of many-objective behaviour of the problem, no artificial fix up is used. Instead, the Pareto-optimal approach [9] is used. In this approach, superiority of one

solution over others is not established for all the objectives considered simultaneously. Through this algorithm will try to find out a diverse set of solutions close to Pareto front. The main aim of developed GA is to obtain a diverse set of solutions as close as possible to the actual Pareto-optimal front.

2 Problem Formulation

Multi-objective FJSP addressed in this study consider n jobs (indexed by i) and M machines (indexed by k). Each job has a total of Q_i operations, and operation sequence is given by O_{ij} for $j = 1, 2, \ldots, Q_i$. Machines for each O_{ij} are represented by M_{ijk}. If $M_{ijk} \subset M$, it is a case of partial flexibility, and if $M_{ijk} = M$, it is a case of full flexibility. Processing time of M_{ijk} is predefined and given by p_{ijk}. The goal is to determine schedule which results in the minimization of makespan (F_1); maximal workload (F_2), and the total workload of the machines (F_3).

It is assumed that all machines and jobs are available and ready to start at time $t = 0$. Jobs are independent of each other, and machines never breakdown and process single operation at a time with a non-preemption condition. Precedence constraint between the operations of the same job is known and maintained. Each job could visit the machine more than once. The setup time and job transportation time are included in the corresponding processing time.

3 Solution Procedure

3.1 Coding and Decoding

In FJSP, potential solutions known as individuals or chromosome consist of two genes representing scheduling and routing subproblems, respectively. Hence, chromosome representation consists of two parts, namely operation sequence (OP) and machine assignment (MS), as shown in Fig. 1. In OP, integer i indicates job number and its jth repetition indicates O_{ij} operation. In MS, machines are indicated by integer values and arranged in order of jobs. That is first we arrange the machines for first job operations, then for second job operations and so on.

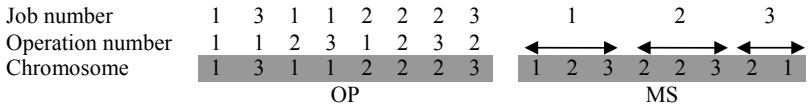


Fig. 1 Structure of proposed chromosome

To reduce the large feasible search space resulting chromosome is mapped to the corresponding phenotype using an active schedule. In an active schedule, an operation is scheduled at the earliest available time in a machine.

3.2 Non-domination Sorting

In this step, the population is sorted in ranks according to an ascending level of non-domination [10]. The non-dominating solutions from the entire population are assigned rank 1, followed by assigning rank 2 to those solutions which are only dominated by rank 1 solutions and so on. For determining the diversity among individuals of the same front crowding distance is calculated. It is a Euclidian distance between the members of the same rank calculated using their function value in the m dimensional hyperspace. The crowding distance of members at the extreme is taken equal to infinity.

3.3 Initial Population

The efficiency of the present approach is increased by selecting suitable chromosomes located at a proper region in search space so that algorithm reaches the Pareto front in the least possible time and without endanger of trapping in local optima. To achieve this, 50% of the initial population is generated randomly and the remaining 50% population is generated using a problem-specific approach. For random population generation, alleles of OP gene are generated randomly with the restriction that each i appear Q_i times. For MS sub-string, each i in OP is decoded to corresponding job and operation, then for ith job jth operation set of available machines M_{ijk} is identified and one machine is assigned randomly. The second part of the initial population is generated using a problem-specific heuristic. For this, OP is generated randomly as in random population generation, but for elements of MS, the machine which has the minimum processing time for performing selected operation from OP is selected.

3.4 Selection

Selection of individual chromosomes for generating next-generation children is an important criterion to balance exploitation and exploration capability of an algorithm. For this tournament-selection procedure is adopted. In this method, two distinct individual solutions are selected randomly and compare. The comparison is done on the bases of rank, and the solution with the least rank is selected. If ranks are same, then individual with maximum crowding distance is selected. Between individuals of the

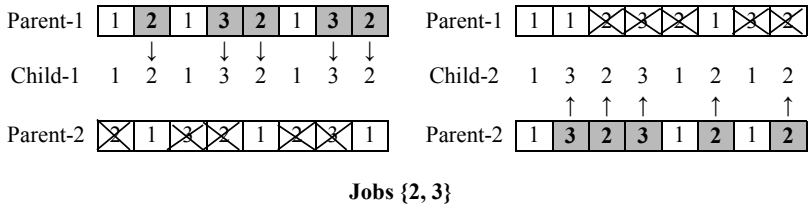
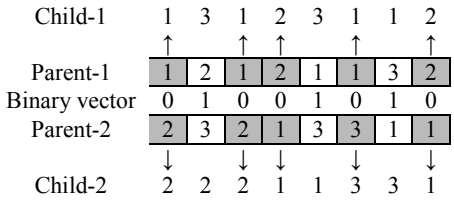


Fig. 2 POX crossover for OS part

Fig. 3 MPX crossover for MS sub-string



same rank, the one with maximum crowding distance is selected else random selection is done. The selected individual is added to the mating pool for the generation of next-generation individuals.

3.5 Crossover

Pair of distinct individuals are selected randomly from the mating pool for crossover operation. For crossover of OP, sub-string of chromosome precedence preserving order-based crossover (POX) is used, and for MS sub-string multi-point preservative (MPX) crossover is used. In POX, as shown in Fig. 2, two jobs are selected randomly from a set of available jobs and where it occurs in parent-1; at the same position, it is added in child-1 and deleted from parent-2. Remaining entries in parent-2 are added in child-1 empty positions in a sequence as in a parent-2. Then roles of two parents are reversed, and child-2 is formed as explained for child-1. In MPX crossover, as shown in Fig. 3, a binary vector of length equals to MS length is generated with the element as 0 or 1. At a particular position of this vector if 0 is present then for child-1, allele value at that position in parent-1 is added and for child-2 allele value at that position of parent-2 is added; otherwise, parent-1 and parent-2 are reversed.

3.6 Mutation

The generated children will replace their parents in population, and then the entire population is subjected to mutation. The individual for mutation is selected randomly. For an OP part of selected individual two positions are selected at random and sub-string is inverted between these two positions. For MS sub-string of the same

selected individual, any job is selected randomly and the machines associated with each operation of the respective job are replaced randomly from the set of available machines.

3.7 Elitism

Better solutions of the previous generation known as elites of the population are merged with the solutions of the current generation to find the elites of the present generation. Non-domination sorting is performed on this merged population, and solutions having rank one are preserved as an elite of present generation. Out of these generated elites, only those which are not present in the current generation are added in the population and worst solutions are removed to keep population size constant. Complete steps followed in this algorithm are represented in Fig. 4.

4 Results

To illustrate the efficiency and effectiveness of the proposed GA for optimization of multi-objective FJSP, five Kacem instances, namely 4×5, 8×8, 10×7, 10×10 and 15×10, are used. An algorithm is coded in MATLAB and run in Intel™ core7 computer having 8 GB RAM. Algorithm parameters are given in Table 1.

For comparison purpose, results given in [1, 3, 5, 11] are combined and non-dominating sorting is performed to find out the optimal solutions. Results from the proposed methodology are compared with those from the literature and are shown in Table 2. From the obtained result, it is seen that for 4×5, 8×8 and 10×10 problem our algorithm gives the same result as discovered by other researchers. But for the case of 10×7 and 15×10 problems, proposed algorithm can locate better Pareto front as shown in Fig. 5.

5 Conclusions

In this study, a simple genetic algorithm is proposed to solve the multi-objective flexible job shop scheduling problems. The main highlight of the work and associated advantages are as follows.

- Solutions known as chromosomes are represented using an integrated approach.
- In this, single chromosome consists of two parts known as operation sequence and machine sequence. Operation sequence represents the routing subproblem, and machine sequence represents the scheduling subproblem of FJSP.

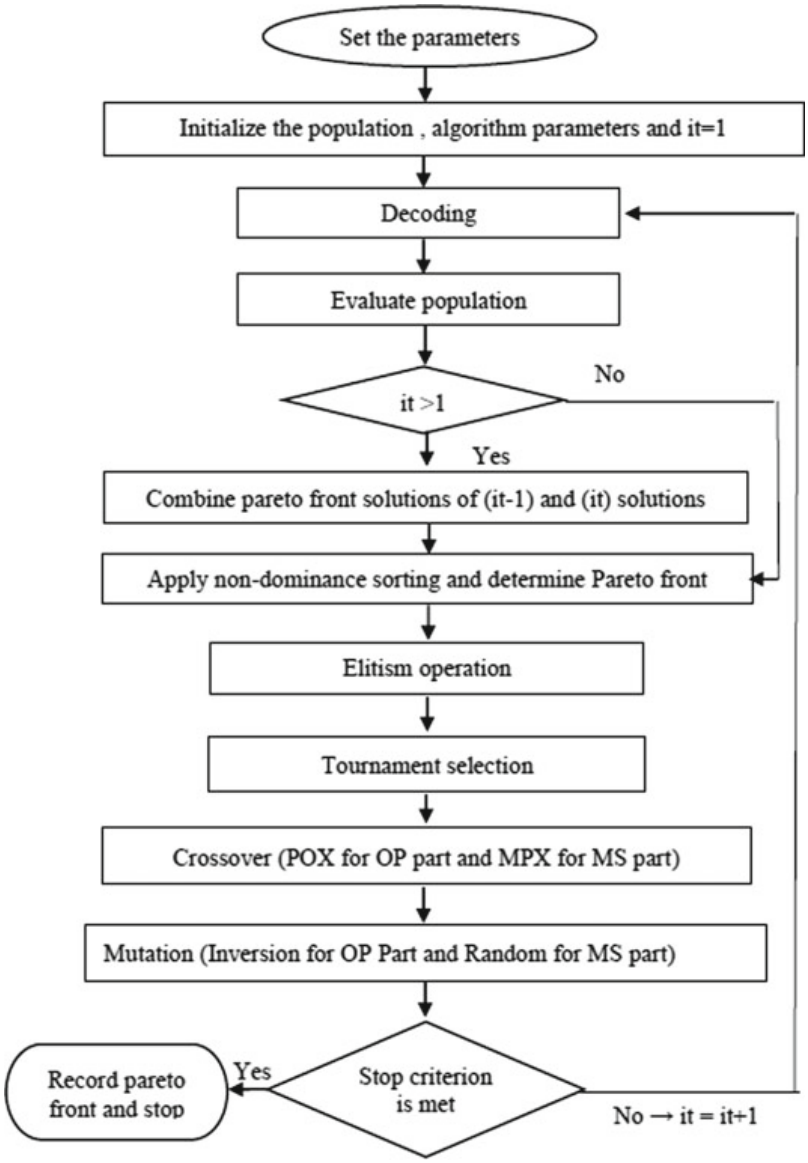


Fig. 4 Flowchart of multi-objective genetic algorithm for FJSP (it = iteration number)

Table 1 Algorithm parameters

Parameters	Value
Generations	200
Total population	10 × length of chromosome
Percentage of population generated randomly	10%
Crossover probability	90%
Mutation probability	1%

Table 2 Comparison of results of proposed algorithm with those from literature

Problem	Length of chromosome	Pareto front from literature			Obtained Pareto front		
		F_1	F_2	F_3	F_1	F_2	F_3
4 × 5	24	11	9	34	11	9	34
		11	10	32	11	10	32
		12	8	32	12	8	32
		13	7	33	13	7	33
8 × 8	54	14	12	77	14	12	77
		15	12	75	15	12	75
		16	11	77	16	11	77
		16	13	73	16	13	73
10 × 7	58	11	10	62	11	10	62
		12	12	60	11	11	60
		11	11	61	12	10	61
					12	11	59
					12	12	58
10 × 10	60	7	5	43	7	5	43
		7	6	42	7	6	42
		8	5	42	8	5	42
		8	7	41	8	7	41
15 × 10	112	11	10	93	11	10	93
		11	11	91	11	11	91
					13	10	90

- Efficient chromosome representation adopted in this work simplifies the generation of active schedule and requires no repair mechanism during genetic alteration and hence always results in a feasible schedule.
- Chromosome coding and decoding mechanism adopted is suitable for both full and partial flexible problems.
- The performance of the approach is evaluated with the results obtained from other authors' algorithms, and it was shown that the algorithm is giving equivalent and in some cases better results.
- In future, work will be extended to incorporate the uncertainty in scheduling like machine breakdown or fuzziness in the planning horizon.