Crossover and Mutation Strategies applied in Job Shop Scheduling Problems

J. Vilma Roseline¹, Dr. D. Saravanan²

¹ Asst Professor, Department of Mathematics, SRM IST, Ramapuram Campus, Chennai, India. Research Scholar, Bharathiyar University, Coimbatore, India
² Professor of Mathematics, Karpaga Vinayaga College of Engineering and Technology, Chennai, India.

*Corresponding author: vilmaroselinej@gmail.com

Abstract: This paper reviews modern genetic algorithm based approaches for solving job shop scheduling problems. This paper elaborates the types of crossover methods such as Partially Mapped crossover (PMX), Order One Crossover (OX), etc involved in Genetic Algorithm (GA) that are used for Job Shop Scheduling problems. Job Shop Scheduling represents one of the hardest combinatorial optimization problems where number of possible schedules drastically increases with the number of operations and machines. Normal crossover operators will often lead to inadmissible solution. Many specialized combining order or adjacency information from the two parents. The crossover method operators builds an offspring by choosing a subsequence of elements from one parent and preserving the order and position of as many elements as possible from the other parent. A subsequence of elements is selected by choosing two random cut points, which serve as boundaries for the swapping operations. There are few distinct mutation operators widely used for JSS problems such as swap, inversion, insertion (shift) and displacement mutation. Here some modern genetic algorithm-based approaches from the literature are also discussed aswell.

Keywords: genetic operators, job shop scheduling, process planning, makespan, optimization

1. Introduction

Production scheduling represents the process of making the best possible utilization of resources by allocating them over some time period with the purpose of performing some tasks and satisfying certain criteria [1,2]. This type of problem belongs to the group of combinatorial optimization problems and goal is to find best solution within a discrete set of alternative solutions [1]. A significant number of published papers are related to the optimization of production scheduling problems like the ones that will be discussed later. In this paper we are focused on job shop scheduling problem which is the most popular problem of this group. Job shop environment consists of a set of different resources/machines (lathes, milling machines, drilling machines etc.) that are able to perform operations of some jobs where jobs represent parts to be produced [3]. One of the simplest definitions of job shop scheduling (JSS) problem is provided by authors [4]: Given n jobs which have to be processed on m machines, each job consists of a sequence of task operations. Each of these operations requires to be processed without interruption for a given period of time on a given machine (no pre-emption). All machines in the shops are capable of processing at most one job at a time. Tasks of the same
job cannot be processed simultaneously & each of them needs to meet each machine only once and the aim is to generate a time schedule to perform the operations on machines. One of the most attractive and mostly considered objectives is to find a scheduling plan that minimizes the makespan, that is, the time needed to complete all the jobs [4]. These assumptions are usually considered in static and deterministic models of classical JSS problems which are widely considered in the literature. During the years, metaheuristics became very popular approaches which are capable of solving instances of problems that are believed to be hard in general, by exploring the usually large solution search space of these instances [5]. Genetic algorithms (GA) are widely known metaheuristics which have been successfully applied to many different optimization problems in last few decades. They belong to the class of evolutionary algorithms which are based on the principles of natural evolution [6] and are very applicable to different combinatorial problems such as job shop scheduling. This paper reviews genetic operators as implementation methods for solving job shop scheduling problems. Many different types of GA components are mentioned and briefly discussed and some modern examples from the literature are analysed. Besides that, integrated process planning and scheduling approach is shortly mentioned and some examples are shown.

2. Job Shop Scheduling

The JSSP can be formulated as follows: For "n" jobs, each one is composed of several operations that must be executed on "m" machines. For each operation uses only one machine for a fixed duration. Each machine can process at most one operation at a time and once an operation starts processing on a given machine, it must complete processing on that same machine without any interruption. The operations of a given job have to be processed in a given set of order [(7)]. The problem’s main aim is to schedule the activities or operations of the jobs on the machines such that the total time to finish all jobs, that is the makespan (Cmax), is minimized. The term makespan refers to the cumulative time to complete all the operations of all jobs on all machines. It is a measure of the time period from the starting time of the first operation to the ending time of the last operation. In the literature authors often consider basic assumptions while solving JSS problem [12]:

- Job cannot visit one machine more than once;
- Simultaneous performing of operations not possible;
- Machine can process only one job at current time;
- No specifications of release and due dates;
- No pre-emptions of operations.

During the years, research in the field of JSS problems has expanded towards flexible job shop scheduling problems (FJSS) in which more than one machine for performing each operation are considered. This problem exhibit additional complexity in comparing to the classical JSS. Two different types of FJSS are considered in the literature: Total FJSS, where each operation can be processed on any machine on the shop floor, and partial FJSS, where each operation can be processed on some machine of subset of few existing machines on the shop floor.

3. Genetic Algorithms And Approaches For Improvement

Like the other algorithms in the class of evolutionary algorithms, genetic algorithm is based on population of individuals which assumes that population of possible solutions evolve through iteration process. As population based method, GA is exploration oriented, meaning that algorithm will ensure that all regions of search space are visited [5]. Genetic algorithm was firstly introduced by John Holland in 1975[9] and belongs to the group of evolutionary algorithms, like genetic programming among others. Since then, GA was applied on large number of various types of optimization problems including job shop scheduling. Davis [10] makes the point that although GAs work very well across a range of different problems, they can never be considered the best optimization method for any particular problem. Based on some disadvantages, like slow convergence and lack of exploitation ability (reduced possibility of finding best possible individual in visited regions, i.e. bad local search abilities) [11], there are three different hybridization approaches that improve efficiency of GA [2,10]:

Many authors classify GA representations in two basic ones: direct and indirect. Main distinction is that for direct representations, schedules are directly encoded into chromosomes, while, on the other side, for indirect representations that is not the case. For job
shop scheduling problems several GA representations are used. Two main categories are model-based and algorithm based GA representations according to the authors [6,13,14]. Different types of these representations are illustrated on Figure 1.

![Figure 1. GA types of representations for JSS problem [13]](image)

For model-based representations, the structure of the genotype (configuration of genes) is based on the decision variables of particular job shop mathematical model and the genotype can be directly interpreted to feasible or infeasible solution. If infeasible solution is obtained, it has to be transformed into a feasible one. On the other side, in algorithm based representations genotype used to store information which can be used by an algorithm to generate feasible solutions. Model based representation are divided into three groups based on decision variables that are considered; those are binary, processing sequence and integer variables as shown on Figure 1 [6,13]. After evaluating fitness values for each individual in population, standard procedures for selecting best individuals (schedules) are considered. According to author [6] fitness-proportional selection (popular as roulette wheel) and tournament selection are mostly used for job shop problems.

### 3.1 Crossover

The crossover operates on two chromosomes at a time and generates offspring by combining features of both chromosomes. In general, the performance of the GAs depends, to a great extent, on the performance of the crossover operator used. During the past two decades, various crossover operators have been proposed for literal permutation encodings for JSSP, such as partial-mapped crossover (PMX), order crossover (OX), cycle crossover (CX), position-based crossover, order-based crossover, etc.[(20)]. In this paper, we use a modified OX that proposed by Gao and et al. [(21)], where they used OX in operation sequence; here we used it in job sequence for every machine.

The modified OX can be described as follows:

1. **Step 1:** Select a subsection of job sequence for a machine from one parent at random.
2. **Step 2:** Produce offspring by copying the substring of job sequence for a machine into the corresponding positions.
3. **Step 3:** Delete the jobs that are already in the substring from the second parent. The resulted sequence of jobs contains jobs that the offspring needs.
4. **Step 4:** Place the jobs into the unfixed positions of the offspring from left to right according to the order of the sequence in the second parent. The modified OX for one machine is illustrated in Figure.

| Chromosome | Gender | Generation t | Generation t+1 |
|------------|--------|--------------|----------------|
| C₁         | Male   | Female       |
| C₂         | Female | Male         |
| C₃         | Male   | Female       |
| C₄         | Female | Male         |
| C₅         | Male   | Female       |

The modified OX can be described as follows:

**Step 1:** Select a subsection of job sequence for a machine from one parent at random.

**Step 2:** Produce offspring by copying the substring of job sequence for a machine into the corresponding positions.

**Step 3:** Delete the jobs that are already in the substring from the second parent. The resulted sequence of jobs contains jobs that the offspring needs.

**Step 4:** Place the jobs into the unfixed positions of the offspring from left to right according to the order of the sequence in the second parent. The modified OX for one machine is illustrated in Figure.
3.2 Sexual Selection

In a standard GA, chromosomes reproduce asexually, that is, any two chromosomes may be the parents in crossover. Gender division and sexual selection inspired a model of gendered GA in which crossover takes place only between chromosomes of opposite sex. Since GA mimics natural evolution, then gender is one of the crucial elements in a GA. Inspired by the nongenetic sex determination system prevalent in some reptile species where sex is determined by the temperature at which the egg is incubated, the population are divided such that the male and female would be selected in an alternate way. The layout of the male and female chromosomes in each generation is different. In other words, when the number of generation is even, the chromosomes and are the male and female, respectively, and vice versa when the number of the generation is odd. For instance, suppose that a population size is five, the layout of males and females from this population is shown in Table 1.

3.3 Mutation

Mutation is just used to produce small perturbations on chromosomes in order to maintain the diversity of population. During the last decade, there are several mutation operators have been proposed such as inversion, insertion, displacement, reciprocal exchange mutation, and shift mutation [(21)]. Inversion mutation selects two positions within a chromosome at random and then inverts the substring between these two positions. In this paper we advanced inversion mutation, by choosing two or more positions in the jobs sequence for one machine and invert them as showed in Figure 3. These steps are applied to other machines, to get a new Offspring.

3.4 New Generation
In this step, the new generation composed by migration initial population with and offspring population. Then we take the best individuals of initial population with generation gap percentage and offspring population.

3.5 Termination Test
The algorithm is terminated either the maximum number of generations is achieved, or when the individuals of the population converge, convergence occurs when all individual positions in the population are identical. In this case, crossover will have no further effect.

4. Conclusion
In the paper the brief overview of genetic algorithms for job shop problems is introduced. This is extending the knowledge of reproduction scenario which we have learnt in all school days. Different types of representations, selection strategies, genetic operators (crossover and mutation) and termination criteria of GA for successful implementation in the search for optimal scheduling plans are also discussed.

References
[1] Kumar, R., Sankha, D.: A Genetic Algorithm Based Approach for Optimization of Scheduling in Job Shop Environment, Journal of Advanced Manufacturing Systems, Vol. 10, p.p. 223-240, 2011.
[2] Gen, M., Cheng, R.: Genetic Algorithms and Engineering Optimization, John Wiley & Sons, Inc., New York, 2000.
[3] Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., Weglarz, J.: Handbook on Scheduling: From Theory to Applications, Springer Berlin Heidelberg, p.p. 345-396, Berlin, 2007.
[4] Shafia, M. A., Pourseyed Aghaee, M., Jamili, A.: A new mathematical model for the job shop scheduling problem with uncertain processing times, International Journal of Industrial Engineering Computation, Vol. 2, p.p. 295-306, 2011.
[5] El-Ghazali, T.: Metaheuristics: From Design to Implementation, John Wiley & Sons, Inc., New York, 2009.
[6] Werner, F.: Genetic Algorithms for Shop Scheduling Problems: A Survey, chapter 8, in: Siarry, P.: Heuristics: Theory and Applications, Nova Science Publishers, Inc., New York, 2013.
[7] Goncalves J.F., Mendes J.J.M., Resende M.G.C., 2005, A hybrid genetic algorithm for the job shops scheduling problem, European Journal of Operational Research, 167, 77–95.
[8] Holland, J.A.: Adaptation in natural and artificial systems, MIT Press, USA, 1975.
[9] Partridge, D.: A review of “Handbook of Genetic Algorithms” from Davis, L.D., Taylor & Francis Group, Vol. 3, p.p. 446-448, 1991.
[10] Beasley, D., Bull, D.R., Martin, R.R.: An Overview of Genetic Algorithms: Part 1, Fundamentals, University Computing, Vol.15, p.p. 58-69, 1993.
[11] Bagchi, T.: Multiobjective Scheduling By Genetic Algorithm, Springer US, New York, 1999. M. Young, The Technical Writer’s Handbook, Mill Valley, CA: University Science, 1989.
[12] Grosan, C., Abraham, A.: Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews, Studies in Computational Intelligence, Vol. 75, p.p. 1–17, 2007.
[13] Maghfiroh, M.F.N., Darmawan, A., Yu, V.F.: Genetic Algorithm for Job Shop Scheduling Problem: A Computational Study, J. Software Engineering & Applications, Vol. 3, p.p. 1155-1162, December 2010.
scheduling approach, 12th International Scientific Conference MMA 2015-Flexible Technologies, p.p. 117-120, Novi Sad, 25-26 September, 2015.

[18] Shao, X., Li, X., Gao, L., Zhang, C.: Integration of process planning and scheduling – A modified genetic algorithm-based approach, Computers & Operations Research, Vol. 36, pp. 2082–2096, July 2008.

[19] Zhang, L., Wong, T.N.: An object-coding genetic algorithm for integrated process planning and scheduling, European Journal of Operational Research, Vol. 244, p.p. 434-444, January 2015.

[20] Cheng R., Gen M., Tsujimura Y., 1999, A tutorial survey of job-shop scheduling problems using genetic algorithms, Computers & Industrial Engineering, 36, 343-364.

[21] Gao J., Sun L., Gen M., 2008, A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems, Computers & Operations Research, 35, 2892 – 2907.