J. Basic. Appl. Sci. Res., 3(10)68-75, 2013
© 2013, TextRoad Publication

ISSN 2090-4304
Journal of Basic and Applied Scientific Research
www.textroad.com

Fuzzy Optimization of Queue Length and MakeSpan in Job shop Scheduling problem by NSGA-II Algorithm

A. Asgari[1*], N. Shahsavari Pour[2], H.R.Mollayi[3], M. Pour kheradmand[4]

[1] Department of Industrial Engineering, Science and Research branch, Islamic Azad University, Kerman, Iran.
[2]Department of Industrial Management, Vali-e- Asr University, Rafsanjan, Iran
[3]Department of Industrial Management, Science and Research branch, Islamic Azad University, Kerman, Iran.
[4] Department of Industrial Engineering, Science and Research branch, Islamic Azad University, Kerman, Iran.

Received: June 26 2013
Accepted: August 2 2013

ABSTRACT

Job shop scheduling is one of the difficult combined optimization problems. In this paper it is solved in uncertainty condition. The objective function is to minimizing the makespan and queue length of jobs simultaneously. To solve problem which is a bi-objective problem, we used the NSGA-II algorithm. In this paper the duration of jobs are considered as fuzzy numbers. Finally an applicable example is provided to shown the capability of proposed method.

KEYWORDS: Job Shop Scheduling, multi-objective optimization, NSGA-II Algorithm, Makespan, Queue length, uncertainty condition, fuzzy numbers.

1. INTRODUCTION

Job shop scheduling problem is one of the difficult combined optimization problems. This problem is known as an NP-hard problem. Job shop production can be used in industries with different operation successions in every job. Therefore there is more potential for a flexible production in job shop scheduling which is extremely crucial at present situation and has turned this system to a popular one. This paper deals with minimizing the make span and the queue length of…by NSGA-II algorithm. Makespan is the necessary time to complete all jobs.

In the extremely competitive, customer-centered market of today, better and on time delivery has become a high rank of practice in order to satisfy the customers. Regarding the process of today market, ordering small sized products in various types has increased. Concerning the increase in the demand of the consumers together with the programming problems of the recent years, the decrease in the production circles has a crucial role; Thus optimization programs and programming the production being an essential step in the process of production[1]. The flexible job shop production problem was first seen by Brucker et al. [2]. A classic JSP is a combination of n different jobs and m different machines. Each job is processed in a predefined succession on each machine. Each job contains a series of operations and every operation needs a different machine. All the operations related to each job are processed according to a fixed direction and every operation has a definite processing time. Certain assumptions and limitations are made on the jobs and the machines to solve JSP. One solution is to determine the operation succession on the machines to remove certain limitations and the goal in JSP is usually minimizing the make span, the delay in operating the machines, the average of the operation time and etc. JSP with the goal of minimizing the make span is one of the most famous and most popular present programming models which is one of the most difficult combined optimization and the strongest NP-hard problems, [3]. Many researchers have succeeded in solving the JSP problem by using approaches such as Simulated Annealing (SA), [4], Taboo Search(TS), [5], Genetics Algorithm (GA), [6], Ant Colony Optimization (ACO), [7], Nervous Networks (NN), [8] , Evolutional Algorithms (EA), and other exploratory approaches, [9]. These metaheuristic algorithms could be considered as independent tools to solve this problem. Although there are a lot of multi objective approaches in order to search in continuous and discrete spaces, [10], NSGA-II has good effect in solving combined optimization problems. Job shop scheduling problems consist of delay in delivery, job rating and Make span which are made as a multi-objective function **Error! Reference source not found.**.

* **Corresponding Author:** Amir Asgari, Department of Industrial Engineering, Science and Research branch, Islamic Azad University, Kerman, Iran. Email: amirasgari_84313192@yahoo.com Tel: +98 3624242451.

Ghedjati [12] has suggested a mixed method of metaheuristic approaches based on Genetics Algorithm (GA) to solve FJS problem in order to minimize the performing of jobs. One of the characteristics of his research is taking into consideration precedence constraints between job operations in two forms of linear and nonlinear. Kacem et al. [13] has, for the first time, examined FJS problem in multi objective status in which the suggested method is a combination of fuzzy logic and evolutionary algorithms, the flexibility of the problem constraining only to the flexibility of operations. In addition, Brandimarte has also examined FJS problem in multi objective mode. He has used Tabu Search to solve the problem. Lee et al. [14] have presented a Genetics Algorithm to solve a similar problem to FJS in supply chain. In the suggested model outsourcing, alternative machine for each operation and succession of multi-function for each part have been assumed. Park et al. [15] have suggested a metaheuristic method based on a hybrid and parallel Genetics Algorithm to solve the problem. Brucker et al. [16] have studied a multi-mode Job Shop scheduling problem and have suggested a Tabu Search algorithm to solve it. Brucker et al. [16]studied flexible Job Shop problem with multi-purpose machines. This is a special mode in flexible Job-Shop problem in which the processing time of each operation is not related to the machine doing the operation. Kim et al. [18] have presented a combined algorithm for integration of process planning and JS scheduling; among the characteristics of this research, parallel search and 3 kinds of flexibilities (flexibility of operation, succession of operation and process) in process planning can be noted. Scrich et al. [19] have constrained the flexibility in JS problem to flexibility in operation. Objective function of their problem is to minimize the tardiness and have tried to solve it by using TS algorithm. Xia, et al. [20] have studied the FJS problem with a new approach. They have created a combination of PSO (Particle Swam Optimization) and SA, using PSO in allocating operations to machines and SA to determine the succession of operations.

Minimizing the Make span and the jobs' queue length on machines are the main purposes in this paper.

The paper is organized as follows. In Section 2, the problem definition is provided. This is followed by the solution procedure in Section 3. An illustrative example is provided in Section 4; Section 5then concludes this paper.

2. STARTING JOB-SHOP PROBLEM

In general, JSP has been described as different jobs n, on programmed different machines, m. Each job has an operation (n_i) shown as $O_{i1}, O_{i2}, \ldots, O_{in}$. In this paper JSP is a series of jobs $Job = \{ j_1, j_2, \ldots, j_n \}$ and a series of machines $Machine = \{ M_1, M_2, \ldots, M_M \}$. Each job consists of a series of operations being processed on the machines. There is a series of machines for each operation O_{ij} that are able to do it and the operation O_{ij} should be done by one of them. In order to optimize certain scheduling criteria the succession of operations on each machine is determined. There are some assumptions about the job and the machines in this problem.

Problem Assumptions:

1-Each job is processed by only one machine in a given time.

2-All jobs are done only once by each machine.

3-There is constraint in priority of the operations for different jobs.

In this problem, the objective is to determine the succession of operation on the machines in order to minimize Make span and the queue time of jobs on the machines.

2.1. Multi Objective Mathematical Model

For each waiting time for each job an expense is allocated in this problem. For example, the expense of the waiting time of a given time of the ith job is AC_i . Considering the waiting expense of each job, a coefficient (α) is given to each job; that is the coefficient of the ith job will be (αi). (Formula 1)

$$\alpha_i = \frac{AC_i}{\sum_{i=1}^{n} AC_i} \tag{1}$$

In order to have an easier decision, a coefficient β is given to each job considering the tardiness expense. For example, the coefficient of the ith job is $\left(\beta_i\right)$ (Formula 5).Using this coefficient and considering tardiness expenses of each job and the importance of on time delivery and also the extent of the expenses and the penalties, the importance and the priority of the job can be specified. This is also true for the waiting expense coefficient $\left(\alpha\right)$. Regarding the above relations, the total tardiness of each job is calculated by Formula 2.

$$\beta_i = \frac{TC_i}{\sum_{i=1}^{n} TC_i} \tag{2}$$

It should be taken into consideration that the tardiness of the ith job is T_i and the finishing time of the ith job is C_i. The tardiness of each job on each machine is calculated in the following way:

$$Queue = \sum_{i=1}^{n} \beta_i T_i \tag{3}$$

In this section we use mixed integer linear programming (MILP) model for Job-Shop problem. This is formulized in the following way: The problem consists of job N and machine M. Each work consists of different chains of operations which should be processed on a specific machine in a series of machines. Each job has its own fixed time.

$$\min z_1 = \left(\max\left(C_{ik}\right)\right) \tag{4}$$

$$\min z_2 = \sum_{i=1}^{n} \beta_i T_i \tag{5}$$

$$s.t \quad c_{ik} - t_{ik} + M(1 - x_{ihk}) \geq c_{ih}, \tag{6}$$

$$c_{jk} - c_{ik} + M(1 - a_{ihk}) \geq t_{jk}, \tag{7}$$

$$c_{ik} \geq 0 \tag{8}$$

$$a_{ihk} = 0,1 \tag{9}$$

$$x_{ijk} = 0,1 \tag{10}$$

In which M is a huge quantity.

cik: the finishing time of the ith job (i=1,2,…,n) on the kth machine (k= 1,2,…,n)
tik: the processing time of the ith job (i=1,2,…,n) on the kth machine (k= 1,2,…,n)

aihk= 1: if the job i is processed on the device (h=1,2,…,n) before the machine K; otherwise aihk= 0
xihk= 1 if the ith job is processed before the jth job (j= 1,2,…,n) on the machine k; if vice versa, xihk=0
The equation no.4 shows the minimization of the finishing time of the job on the machine. Equation no.5 notes the minimization of the total queue length. Equation no.6 guaranteed that each job will be done according to thesequence matrix.
Equation no.7 expresses the constraints of the machinery that is each device can process only one job at a given time.

2.2. Fuzzy Multi Objective Mathematical Model

$$\min z_1 = \left(\max\left(\tilde{C}_{ik}\right)\right) \tag{11}$$

$$\min z_2 = \sum_{i=1}^{n} \beta_i \tilde{T}_i \tag{12}$$

$$s.t \quad \tilde{c}_{ik} - \tilde{t}_{ik} + M(1 - a_{ihk}) \geq \tilde{c}_{ih}, \tag{13}$$

$$\tilde{c}_{jk} - \tilde{c}_{ik} + M(1 - x_{ihk}) \geq \tilde{t}_{jk}, \tag{14}$$

$$\tilde{c}_{ik} \geq 0 \tag{15}$$

$$a_{ihk} = 0,1 \tag{16}$$

$$x_{ijk} = 0,1 \tag{17}$$

$\tilde{C}_{ik}$ **Is the fuzzy finishing time of work *i* on the machine *m*,** $(i = 1,2,\ldots,n), (k = 1,2,\ldots,m)$

$\tilde{t}_{ik}$ **Is the fuzzy processing time of work *i* on the machine *m*,** $(i = 1,2,\ldots,n), (k = 1,2,\ldots,m)$

3. Solution procedure

3.1 NSGA-II method

The NSGA method was suggested by Deb et al. [21]. In this method the chromosomes of the first population $\boldsymbol{P_t}$ are first used applying the cross over operator. The new chromosomes form the population $\boldsymbol{Q_t}$. These two populations are merged together to produce the population R_t with 2N chromosomes.

In the first stage non post ranking is applied on the population $\boldsymbol{R_t}$. The results that fall in fronts with level one up to the level needed to provide N chromosomes in the population are selected and other chromosomes are removed. It happens in many cases that the last front consists of more chromosomes needed for the completion of the population. In this case several chromosomes should be removed.

The procedure of this algorithm is summarized below:

First a primary population P_0 is produced accidentally according to the needs of the problem. Front ranking is made and a suit equivalent to the rank of each front (level) is specified for its chromosomes. For example the best chromosomes which are in level one, will have the suit 1. (As the problem is a minimizing one, the population Q_0 is produced by applying cross over and mutation operators.) Other stages are as follows:

1. The populations of the parents and the offspring are merged: $R_t = P_t \cup Q_t$

2. Non post ranking is done on the population R_t the chromosomes are classified in F_i fronts (i=1, 2, …)

3. The new population $P_{t+1} = \varnothing$ is produced. The *i* counter is supposed to start from 1.

4. Until $|P_{t+1}| + |F_i| < N$, the operations $P_{t+1} = P_{t+1} \bigcup F_i$ and $i = i + 1$ are applied.

5. The ranking is done on the basis of crowd distance $(F_i < c)$ and the results showing the most dispersion $(\mathrm{N} - |\mathrm{P}_{t+1}|)$ in the results put into F_i , are put into P_{t+1} .

6. The new chromosomes in the population Q_{t+1} are produced applying the tournament selected operators according to the crowd distance, cross over and mutation on the population P_{t+1} .

In stage 5, using the crowd distance parameter which will be discussed in the next chapter, the ranking of the chromosomes in the ith front (Non-Dominated) that cannot appear perfectly in the population is done. Based on this parameter, the chromosomes are arranged in descending order and then the tournament selected operator according to the crowd distance is applied on them. The advantage of NSGA-II method is using crowd distance to keep the variation of the results in pareto front; although calculating crowd distance and classifying the chromosomes in different fronts will add a huge calculation load to the algorithm.

3.1. Crowd Distance

This idea was suggested to keep the variety of results in the optimized pareto front. In order to calculate the crowd distance related to each point on a certain front, the previous and the next points are selected in respect of target functions of the problem and a rectangle is formed (in a two dimensional level) as shown in figure (1). It is obvious that if the target functions are more than 2, the points will form a cube.

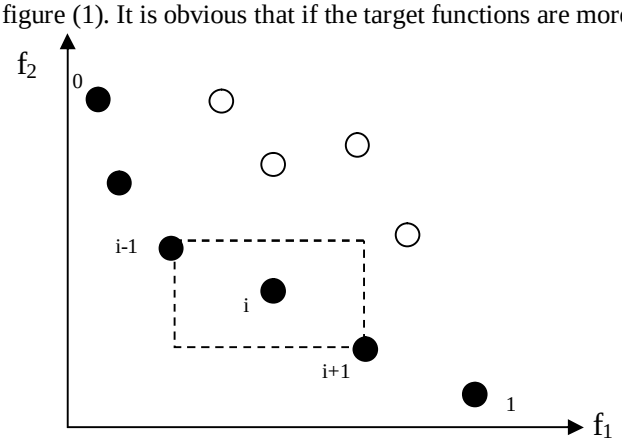

Figure 1: Calculating Crowd Distance in a Minimizing bi-objective problem

As shown in this figure, the numbers in the discussed front are illustrated with solid circles while other results are shown with hollow circles. In order to calculate the crowd distance of the ith result in this figure, a rectangle is drawn the corners of which the previous and the next results regarding the two target functions are located. The crowd distance for this point is equivalent to the average of the rectangle sides. The less the crowd distance, the denser will be the results. It is to be mentioned that in the case where the problem has more than two targets, the points $i-1$ and $i+1$ will not be the same for all functions.

The calculating procedure of crowd distance for the results in front F is as follows:

1. Calculate the number of results in the front F and call it l $(|F| = l)$. Assume the primary amount of crowd distance for each i in this series to be zero $(d_i = 0)$.

2. Arrange the results based on each target function, $m = 1, 2, \ldots, M$.

3. For each target function m, allocate a crowd distance to the results on the front border (the first and the last points) $\left(d_{I_l^m} = d_{I_l^m} = \infty\right)$ and to calculate this indicator for other results use the following relation:

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{\left(I_{j+1}^l\right)} - f_m^{\left(I_{j+1}^l\right)}}{f_m^{\max} - f_m^{\min}} \tag{11}$$

In the relation above, I_j^m indicates the ith result in the arranged list of results based on the target function m. The right numerator of the fraction above shows the difference of the mth target function for the two results next to the result j. The denominator of the fraction shows the difference between the least and the most amount of the mth target function in the population.

3.2. Presenting the Chromosome & Code Breaking

In this paper, each chromosome contains m*n genes and each gene is an integer. The number of different integers equals the number of jobs (n). A job is a series of operations that is scheduled on m machines. Each job appears in a chromosome for *m* times. In each chromosome, from left to right, the jth occurrence of a number of jobs is allocated to the jth technical succession operation of that job. For example when [1,2,3] chromosome is given, the code will be broken as {j1,j2,j3 } jobs respectively. There are 3 different integers each being repeated 3times. From left to right, the first gene means showing the first 3 operation of the third job which is processed on the parallel machine. Then, the second gene means showing the first 2 operations of the second job. Therefore, the chromosome is shown as in figure 2.

[3,3,2,1,2,3,1,2,1]

$$\{O_{31}, O_{32}, O_{21}, O_{11}, O_{22}, O_{33}, O_{12}, O_{23}, O_{13}\}$$

O_{ij} is the sign of jth operation of the i th job.

Figure2- Chromosome Code Breaking

3.3. Crossover Method

In crossover, first 2 parents are randomly selected and then the new generation is generated by changing genetic data between parents. Crossover process is described as in Figure 3. We use uniform crossover in this paper [22]. A random homogenous mask chromosome is randomly selected from 2 points between which the genes will be transferred Figure 4. It should be noted that the mutation rate (F (m)) will be reduced as in equation (12) so that it will reach zero in the last generation.

$$F_{(m)} = 1 - \frac{n_g}{G} \tag{12}$$

Parent1	3	3	2	1	1	2	2	1	3
Parent 2	2	3	1	3	2	1	1	3	2

Figure 3- Parent Chromosomes Sample

Mask 1	1	0	0	0	0	1	0	1	1
Mask2	0	0	0	1	1	1	1	0	0

Figure 3-1-Mask Chromosome Sample

Child1	3	3	1	1	2	2	1	3	2
Child 2	3	3	2	3	2	1	1	1	2

Figure 3-2-The Child Generated by Crossover

Before Mutation	3	3	2	3	2	1	1	1	2
After Mutation	3	1	2	3	2	1	3	1	2

Figure 4-The Chromosome Before and After Mutation

4. NUMERICAL EXAMPLE

A numerical solution of 3 jobs and 3 machines has been considered in this paper. This model has been programmed by visual basic programs (VBA) in Microsoft Excel 2010 software. NSGA-II parameters are as follows: G=100, N=50, crossover rate=0.85, mutation rate=0.1.

Table 1–Processing Time Matrix of each job on each machine

J1	20	50	80	30	40	80	30	50	80
J2	20	40	60	30	40	70	20	40	90
J3	30	50	60	30	50	90	20	40	80

Table 2 –Sequence Matrix of each job on each machine

J1	1	2	3
J2	2	3	1
J3	3	2	1

5. Conclusion

Our suggestion in this paper is the optimization of a multi objective mathematical model for Job-Shop problem scheduling. NSGA-II is applied to solve this problem. The high speed of the algorithm and its fast convergence makes it a proper solution for the Job-Shop problem in order to decrease the makespan and the queue length of jobs on each machine. However, this problem is categorized as NP-hard and finding optimized solutions for bigger problems is very hard using Integer Programming (IP) model, Therefore using NSGA-II Algorithm mentioned in this paper is a good effective solution for this problem.

6. Acknowledgment

I thank all my good friends and teachers Dr. Ghaffari and Mr. Zanjani who have always been so kind as to guide me all the way long and in every aspect.

7. REFRENCES

[1] Hoitomt, D. J., Luh, P. B., & Pattipati, K. R. 1993 ,Practical approach to job-shop scheduling problems, Intl. J. IEEE Transactions on Robotics and Automation., 9(1): 1–13.

[2] Brucker, P., Schlie, R., 1990 ,Job-Shop Scheduling with Multipurpose Machines, Intl. J. Computing., 45: 369–375.

[3] M.R. Garey, D.S. Johnson, R. Sethi, 1996 ,The complexity of flow shop and job-shop scheduling, Intl. J. Mathematics of Operations Research., 1 (2): 117–129.

[4] M. Kolonko, 1999 ,Some new results on Simulated Annealing applied to the Job Shop Scheduling Problem, Intl. J. European Journal of Operational Research., 113 (1): 123–136.

[5] F. Pezzella, E. Merelli, 2000 ,A Tabu Search method guided by shifting bottleneck for the Job Shop Scheduling Problem, Intl. J. European Journal of Operational Research., 120 (2): 297–310.

[6] J.F. Goncalves, et al., 2005 ,A hybrid genetic algorithm for the Job Shop Scheduling Problem, Intl. J. European Journal of Operational Research., 167 (1): 77–95.

[7] K.L. Huang, C.J. Liao, 2008 ,Ant colony optimization combined with taboo search for the job shop scheduling problem, Intl. J. Computers and Operations Research.,35 (4): 1030–1046.

[8] D.J. Fonseca, D. Navaresse, 2002 ,Artificial neural networks for job shop simulation, Intl. J. Advanced Engineering Informatics., 16 (4): 241–246.

[9] H. Chen, P.B. Luh, 2003 ,An alternative framework to Lagrangian relaxation approach for Job Shop Scheduling, Intl. J. European Journal of Operational Research., 149 (3): 499–512.

[10] Coello, C. A. C. 2005 ,Recent trends in evolutionary multi objective optimization. In Ajith Abraham, Lakhmi Jain, & Robert Goldberg (Eds.), Evolutionary multi objective optimization, Intl. J. Theoretical advances and applications. pp7–32.

[11] Fulya, A., Mitsuo, G., Lin, L., & Turan, P.., 2006 ,A genetic algorithm approach for multi-objective optimization of supply chain networks. Intl. J. Computers & Industrial Engineering., 51(1): 196–215.

[12] Ghedjati, F., 1999 ,Genetic Algorithms for the Job-Shop Scheduling Problem with Unrelated Parallel Constraints: Heuristic Mixing Method Machines and Precedence, Intl. J. Computer & Industrial Engineering., 37: 39-42.

[13] Kacem, I., Hammadi, S., Borne, P., 2002 ,Pareto-Optimality Approach for Flexible Job-Shop Scheduling Problems: Hybridization of eVolutionary Algorithms and Fuzzy Logic, Intl. J. Mathematics and Computers in Simulation., 60: 245-276.

[14] Lee, Y.H., Jeong, C.S., Moon, C., 2002 ,Advanced Planning and Scheduling with Outsourcing in Manufacturing Supply Chain, Intl. J. Computer & Industrial Engineering., 43: 351-374.

[15] Park, B.J., Choi, H.R., Kim, H.S., 2003 ,A Hybrid Genetic Algorithm for the Job Shop Scheduling Problems, Intl. J. Computers & Industrial Engineering., 45 : 597–613.

[16] Brucker, P., Neyer, J., 1998 ,Tabu Search for the Multi-Mode Jobshop Problem, OR-Spektrum., 20: 21–28.

[17] Brucker, P., Drexl, A., Mohring, R., Neumann, K., Pesch, E., 1999 Resource-Constrained Project Scheduling: Notation, Classification, Models, and Methods, Intl. J. European Journal of Operational Research., 112: 3- 41

[18] Kim, Y.K., Park, K., Ko, J., 2003 ,A Symbiotic eVolutionary Algorithm for the Integration of Process Planning and Job Shop Scheduling, Intl. J. Computers & Operations Research., 30: 1151-1171.

[19] Scrich, C.A., Armentano, V.A., Laguna, M., 2004 ,Tardiness Minimization in a Flexible Job Shop: A Tabo Search Approach, Intl. J. Journal of Intelligent Manufacturing., 15: 103-115.

[20] Xia, W., Wu, Z., 2005 , An Effective Hybrid Optimization Approach for Multi-Objective Flexible Job-Shop Scheduling Problems, Intl. J. Computers & Industrial Engineering., 48: 409-425.

[21] Deb K., S. Agrawal, A.Pratab, and T. Meyarivan, 2000. A Fast Elitist NonDominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Proceeding of the Parallel problem Solving From Natural VI Conference, pp: 849-858.

[22] O. Hansancebi, F. Erbatur, 2000.Evaluation of crossover techniques in genetic algorithm based optimum structural design, Intl. J. computer and structures, 78, pp: 437-448.