A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms

Ehsan Ahmadi[a], Mostafa Zandieh[b], Mojtaba Farrokh[c], Seyed Mohammad Emami[d]

[a*]Industrial and Systems Engineering Department, PhD Student, Ohio University, Athens., Ohio, USA

[b] Department of Industrial Management, Associate Professor, Shahid Beheshti University, G.C., Tehran, Iran

[c] Department of Industrial Management, PhD Student, University of Tehran, Tehran, Iran

[d] Industrial and Systems Engineering Department, Graduate Student, Rochester Institute of Technology, Rochester, NY, USA

Abstract

This paper addresses the stable scheduling of multi-objective problem in flexible job shop scheduling with random machine breakdown. Recently, numerous studies are conducted about robust scheduling; however, implementing a scheme which prevents a tremendous change between scheduling and after machine breakdown (preschedule and realized schedule, respectively) can be critical for utilizing available resources. The stability of the schedule can be detected by a slight deviation of start and completion time of each job between preschedule and realized schedule under the uncertain conditions. In this paper, two evolutionary algorithms, NSGA-II and NRGA, are applied to combine the improvement of makespan and stability simultaneously. A simulation approach is used to evaluate the state and condition of the machine breakdowns. After the introduction of the evaluation criteria, the proposed algorithms are tested on a variety of benchmark problems. Finally, through performing statistical tests, the algorithm with higher performance in each criterion is identified.

Keywords: Flexible job shop problem, Multi-objective, Makespan, Stability, Machine Breakdown, Simulation

1. Introduction

A proper scheduling has always been one of the success factors for production systems. Job shop scheduling problem (JSSP) is a branch of production planning which includes a set of hardest problems in combinatorial optimization [39]. A JSSP is a multistage system in which each job comprises of several operations that are processed according to a predetermined order. Also, in this problem, it is assumed that only one machine is capable of performing each operation and during the machine break down periods, the production operations are blocked. To deal with this issue, a flexible job shop problem (FJSP) is suggested which is a developed form of the JSSP. In a JSSP, the objective is finding the best sequence of manufacturing operations on a machine, while in a FJSP, in addition to the sequencing of the operations, the machine's task assignments (routing) are also considered. It has been also demonstrated that the JSSP is a NP-Hard problem [12]. Since the FJSP is a developed version of JSSP, therefore, it is also considered to be a NP-Hard problem. In order to achieve a practical solution for the FJSP, various conflicting objectives have to be considered. In most of the previous studies, scheduling problems are generally solved by using a meta-heuristic algorithm, with a single goal including several criteria. However, the combination of criteria as an objective is not a practical approach for real situation.

* Corresponding author. Ehsan Ahmadi, Email: ea162814@ohio.edu

The number of available research on the multi-objective FJSP (MO-FJSP) in literature is less than the single-objective FJSP. The use of a multi-objective evolutionary algorithm approach is essential to cope several objectives simultaneously.

In MO-FJSP, most of the literature used aggregated single-objective algorithms. However, these algorithms usually show a lower level of performance, as compared to new algorithms called Pareto-based multi-objective evolutionary algorithms (MOEAs) [8]. Over the past decade many research projects have been implemented on the number of the MOEAs [11, 33, 6, 30]. Among the MOEAs, Genetic algorithm (GA) has proved to be one of the best evolutionary algorithms that can solve the component optimization problems with a better computational performance than other algorithms like random search and particle swarm optimization. The non-dominated sorting genetic algorithm (NSGA), which is classified as one of MOGA techniques, is one of the MOEAs technics proposed by Srinivas and Deb [31]. To overcome the high dependency of the NSGA to the fitness sharing and other parameters, Deb [8] introduced a new version of the NSGA called NSGA-II, which is usually used for solving large multi-objective problems. NSGA-II is one of the most well-known multi-objective algorithms used in this area which has three specific features that includes (1) an approach with a speed non-dominated sorting, (2) a procedure that can quickly estimate the crowding distance in a timely manner and (3) an operator with simple crowded comparison [9]. These features have made the NSGA-II very effective in solving the FJSP. In this paper, NSGA-II is used for solving this problem, as it has recently been applied to solve several different problems [10,19,27]. Another well-known MOEA, a modified version of the NSGA-II, is called non-dominated ranking genetic algorithm (NRGA) proposed by Al Jadaan et al. [3]. The major difference of the NSGA-II and NRGA is in selection procedure, where NSGA-II uses tournament selection algorithm, while NRGA applies Roulette-wheel algorithm.

In recent studies, several research has shown that the scheduling in real production environments is often influenced by uncertain or stochastic factors such as the resources shortage and machine breakdown. In this study, we considered the uncertainty in the FJSP as a stochastic FJSP. When it comes to the issue of stochastic environment, robust scheduling should be taken into account. According to the literature in robust scheduling methodologies, robustness is mainly grouped into quality robustness and solution robustness [15]. The quality robustness refers to the insensitivity of the scheduling performance such as makespan and total tardiness in the presence of uncertainty. When there is a machine breakdown, actual scheduling may be shifted away from the baseline schedule. The property that start and completion of each job should be as close as possible to its baseline schedule is known as the solution robustness and it is usually considered as a "stability" measurement of the schedule [5]. One of the objectives of this paper is the scheduling stability in the FJSP, in which machine breakdowns are expected. Although the simulation of machine breakdowns can result in a scheduling solution with higher stability, more robustness, and closer to reality, it has not been investigated in most studies. In our proposed algorithm the machines will be subjected to the breakdown by using simulation process to calculate the stability index, where the algorithm will improve the makespan and stability index of the schedule simultaneously.

The structure of this paper is organized as follows: in section 2 of this paper, a number of studies on multi-objective optimization, stable scheduling, and evolutionary algorithm are reviewed. In the next section, problem definitions of FJSP, stability, and simulation are explained. In section 4, different multi-objective evolutionary algorithms are presented, and the result of experimental test is reported in section 5. Finally, conclusion and future possible works related to this study are covered in section 6.

2. Literature review

Kodali et al. [24] and Jianling [20] used optimization technique of NSGA-II to simulate a number of test problems from previous studies. They claimed that the performance of this technique is better than Pareto-archived evolution strategy (PAES) and strength Pareto EA (SPEA) in terms of converging near the true Pareto-optimal set and finding a diverse set of solutions.

Chiang and Lin [6] developed a MOEA that used effective genetic operators. The principal feature of this algorithm is its simplicity in which the problem is solved in a Pareto manner. Rahmati et al. [30] proposed a methodology to investigate a multi-objective FJSP. The objectives that they have addressed in their study are makespan, critical machine work load and total work load of machine under deterministic environment. While in our study, we considered makespan and stability within stochastic environment. Zhang et al. [39] developed a hybrid algorithm which combined two models of PSO and TS. In their model the PSO deals with routing and sequencing simultaneously, while the TS uses a neighborhood function, introduced by Mastrolilli and Gambardella [28], on the sequencing part of the problem. Also, Lei [26] developed a simplified multi-objective genetic algorithm called SMGA for the FJSP, with the objectives of minimizing makespan and total tardiness ratio simultaneously.

Xiong et al. [35] proposed robust scheduling for a FJSP with random machine breakdowns (They simultaneously considered two objectives of makespan and robustness). Two surrogate measures for robustness are suggested by utilizing the available information about machine breakdowns; the first surrogate measure considers the probability of machine breakdowns, while the second considers the location of float times and machine breakdowns. Yuan and Xu [36] proposed memetic algorithm which is developed by incorporating a novel local search algorithm into the NSGA-II for the multi-objective flexible job shop scheduling problem. The objective of their proposed model is defined as the minimization of makespan, total workload, and critical workload. He and Sun [16] proposed an approach in which two strategies of right-shift scheduling and route changing scheduling are used to improve the robustness and stability of rescheduling in the FJSP in subject to the machine breakdown. Xiong et al. [34] suggested a hybrid multi objective evolutionary approach (H-MOEA) for solving the multi-objective FJSP by using development of well-design chromosome representation and genetic operators. They also applied a local search procedure in H-MOEA based on the critical path theory to improve the convergence of the algorithm.
Al-Hinai and ElMekkawy [1] studied the robust and stable FJSP with random machine breakdowns by using a two-stage genetic algorithm. In the first stage, the makespan is minimized and in the second stage, cost function is converted to a bi-objective function and integrates machine assignments and operations sequencing with the expected machine breakdown. Goren and Sabuncuog [14] proposed two surrogate measures for robust and stable schedules with respect to random disruptions, which generate schedules in a single-machine environment that are

subjected to machine breakdowns. They considered both busy and repair time distributions are embedded in a tabu-search-based scheduling algorithm.

Although many studies are conducted in robust and stable scheduling, some of them investigate single-machine, while the rest take multi-criteria approach. In addition, their simulation strategy to evaluate stability measure are quite simplistic and far from reality. To be much more applicable in manufacturing environment, we used an extended simulation in this paper and we also adopted NSGA-II and NRGA algorithms for solving MO-FJSP in subject to the makespan and stability measures. More details are presented in following section.

3. Problem definition

3.1 Flexible job shop scheduling

The scheduling problem in the FJSP is divided into two sub-problems of routing and sequencing. In a routing sub-problem, each operation will be assigned to a capable machine, while in a sequencing problem, the operation position will depend on the order of the assigned operations for each machine. In the literature, some of the studies applied the two sub-problems separately on a multi-objective flexible job shop problem (MO-FJSP), where in other group of studies, they used the two sub-problems simultaneously.

Also in so called hierarchical approach, a hard problem is divided into two simpler problems. However, in integrated approach, keeping the complexity of the problem usually results in higher quality of solutions in a real world situation.

Assumptions of a FJSP are as following:

- A set of ***n*** jobs $\boldsymbol{J} = \{\boldsymbol{J}_1, \boldsymbol{J}_2, ..., \boldsymbol{J}_{\boldsymbol{n}}\}$ and a set of ***m*** machines $M = \{M_1, M_2, ..., M_m\}$.
- Each job consists of a predetermined sequence of operations $\boldsymbol{O} = \{\boldsymbol{O}_{1\boldsymbol{j}}, \boldsymbol{O}_{2\boldsymbol{j}}, ..., \boldsymbol{O}_{\boldsymbol{ij}}\}$
- Each operation requires one machine that is selected from a set of available machines
- All jobs and machines are available at time 0, and each machine can only execute one operation at a given time
- Machines never breakdown and they are always available.
- The processing time of an operation ($\boldsymbol{O}_{\boldsymbol{ij}}$) on machine k is predetermined.
- The setup time of any operation is sequence independent and it is included in its processing time.
- Preemption is not allowed (A started operation cannot be interrupted).

Also, the flexibility of a FJSP can be categorized into total flexibility (T-FJSP) or partial flexibility (P-FJSP). It is considered to be in total flexibility when each operation can be processed with any of the machines and it is assumed to be a partial flexibility when each operation can be processed only on one or a subset of the machines [21].

3.2 Stability measure

In the literature, a schedule is assumed to be stable schedule if it has less deviation in the makespan and sequence between the preschedule and the schedule after breakdown named realized schedule [25].

For calculating the stability, we use the stability measurement model that is proposed by Al-Hinai and ElMekkawy [1]. In this model, the stability is calculated as an average of difference between the completion times of the predicted schedule and the realized completion time:

$$\boldsymbol{STB} = \min \frac{\sum_{j=1}^{n} \sum_{i=1}^{q_j} \left| \boldsymbol{CT_{ijP}} - \boldsymbol{CT_{ijR}} \right|}{\sum_{j=1}^{n} \boldsymbol{O_j}} \quad (1)$$

In this equation, n is the number of jobs, q_j the number of operations of job $\boldsymbol{j}$, CT_{ijP} the predicted completion time of operation $\boldsymbol{i}$ of job $\boldsymbol{j}$, CT_{ijR} the realized completion time of operation $\boldsymbol{i}$ of job $\boldsymbol{j}$, and $\boldsymbol{O_j}$ the total number of operations of job $\boldsymbol{j}$.

3.3 Machine breakdown

In a stochastic environment, to evaluate the effect of machine breakdown using simulation is inevitable. Liu, Gu, and Xi [25] mentioned that exhausted simulations may have large time requirements. However, to address the stability of the schedule, we implement a simulation approach to evaluate the effects of the machine breakdown. In a case of machine breakdown, the interval between every two breakdown occurrences is considered to follow an exponential distribution with ***MTBF***[1] as mean parameter. Furthermore, the repair times follow an exponential distribution with ***MTTR***[2] as mean parameter. In addition, all machines have the same mean values for ***MTTR*** and ***MTBF***.

After a breakdown occurrence, a reactive repair of the predictive schedule should be performed. Rescheduling only the operations which are directly and indirectly affected by a disruption is a better alternative to a total

affected operations and preserve the stability of the schedule. Also, we assumed that the processing is resumed after a machine breakdown.

To illustrate the effect of machine breakdown on makespan and stability, suppose we have an FJSP with three machines and three jobs. The process time for each operation on the different machines is as shown in Table 1.

---

[1] MTBF Stands For mean time between failure
[2] MTTR Stand for mean time to repair

Table 1. Processing times table for a FJSP with 3 jobs and 3 machines.

J	O_{ij}	$M1$	$M2$	$M3$
J_1	O_{11}	3	4	1
	O_{12}	4	1	0
J_2	O_{21}	1	1	5
	O_{22}	1	0	2
	O_{23}	0	2	2
J_3	O_{31}	2	2	2
	O_{32}	1	2	2
	O_{33}	0	1	1

Two schedules are shown in Fig. 1. The left-hand-side of the figure shows two Gantt-charts of two possible schedules for this FJSP. The makespan of Fig. 1(A) is 4 time unit and 5 for Fig. 1(C). If the objective was just minimum makespan, then the schedule Fig. 1(A) would be more appropriate to be selected. But when the probability of machine breakdown is considered into the problem, then a schedule that can absorb the effect of this disruption would be selected. For example, suppose machine 1 and machine 2 are subject to the same disruption specified by the cross. For both preschedule, rescheduling, as shown in the right-hand-side of Fig. 1, is performed. The operations that are affected by machine breakdown are displayed by red lines. After rescheduling, both of the machines have the same makespan. But, regarding to stability measure, since schedule Fig. 1(B) has a stability value of 0.375 and Fig. 1(D) has a stability value of 0, schedule of Fig. 1(D) is more preferable. In this condition decision maker would release schedule Fig. 1(C) to the shop floor.

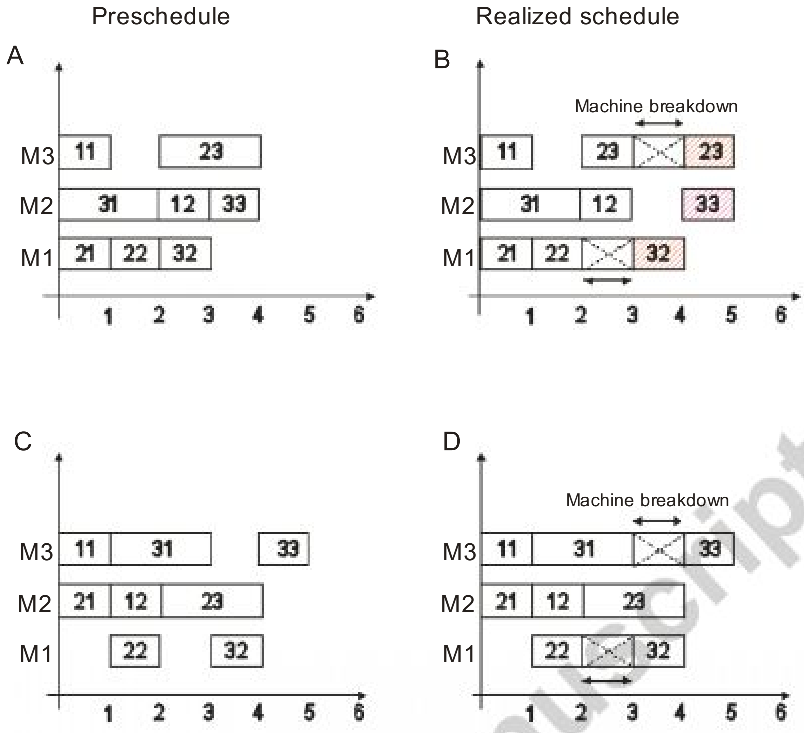

Figure 1. Two preschedule (A, C) and rescheduling after machine breakdown named realized schedule (B, D).

As described in this example, it is very important for decision maker to have a trade-off between makespan, stability and robustness in which the decision to release schedule on the shop floor will be affected.

In the following sections, simulation algorithm and structure of proposed evolutionary algorithm are presented.

3.4 Simulation algorithm

Since in this paper our objective is to investigate the effect of machine breakdowns on the scheduling scheme under randomly disruption, we require a simulation to reproduce the situation that allows the machines to disrupt due to random breakdowns.

Simulation algorithm used in this paper was proposed by Zandieh and Gholami [38]. They have incorporated a simulation model into an immune algorithm to schedule a hybrid flow shop with sequence-dependent setup times and machines with random breakdowns.

Since this problem has a probabilistic nature, it is essential to replicate the computation of the simulation several times (N***sim***) for each sequence while all the features of the problem remain constant.

4. Multi-objective evolutionary algorithms

Multi-objective evolutionary algorithms (MOEAs), which are classified as one of Meta heuristic algorithms, are also used for solving multi objective optimization problems. In the study, among different MOEAs algorithms, the NSGA-II and NRGA algorithms are used for solving a MO-FJSP, with an efficient procedure to generate the initial population.

4.1 Non-dominated sorting genetic algorithm

Non-dominated sorting genetic algorithm II (NSGA-II) is one of MOEA techniques that optimize the problem objectives simultaneously, without being affected by any other solution [37]. The computational complexity of the NSGA algorithm provoked from existing complexity of a non-dominated sorting procedure in every generation, results in an expensive procedure as compared to the NSGA-II algorithm for large population sizes [9]. NSGA-II, a type of the GA, includes the same type of GA's operators. Since the performance of a GA depends on its selected operators, we apply operators whose performances have been verified in literature. In this study, some of the operators such as crossover and mutation, have a strong similarity to Al-Hinai and ElMekkawy's algorithm [2].

In the following section, chromosome representation, initialization and operators of NSGA-II are introduced.

4.1.1 Chromosome representation

Chromosome representation is an important issue for the GA in regards of computational time. It can be concluded from the works of Ho et al. [17] and Mattfeld [29] that the search space of an operation-based representation covers the whole solution space and any permutation of operators can correspond to a feasible schedule.

In this paper, the permutation-based chromosome representation proposed by Kacem, Hammadi, and Brone [22] has been used. This permutation-based chromosome composes of a matrix whose each row consist of triples (j, i, k) forms the chromosome, in which the

- j is current job number.
- i is operation number within job j.
- k is machine assigned to the operation.

For instant following matrix represent an example of one chromosome for three jobs and three machines:

$$\begin{bmatrix} 1 & 1 & 3 \\ 3 & 1 & 2 \\ 2 & 1 & 1 \\ 1 & 2 & 2 \\ 2 & 2 & 1 \\ 3 & 2 & 1 \\ 3 & 3 & 2 \\ 2 & 3 & 3 \end{bmatrix}$$

Figure 2. Example of chromosome representation.

4.1.2 Initialization

For the initial population, we used Ini-PopGen heuristic suggested by Al-Hinai and ElMekkawy [2] that randomly assigns the priority to jobs. Consequently an operation is assigned to the machine which can finish processing sooner than the rest of appropriate machines. This procedure considers both the processing time and the work load on each machine. This heuristic algorithm is very effective in generating the initial population in subject to the makespan objective.

4.1.3 Genetic operators

Achieving a high performance of genetic algorithms is highly dependent on the performance of the genetic operators that are used in these algorithms [13]. Therefore, using appropriate operators is a fundamental factor for extending any GA algorithm. Furthermore, performing genetic operators may produce infeasible schedule. In this situation we have to perform a repair mechanism that could be time consuming. Thus, it is more practical to design the operators that maintain the feasibility of the schedule and avoid the repair mechanisms.

Selected operators in this study consist of precedence preserving order-based crossover (POX) for crossover and a modified Position Based Mutation (PBM) as well as Machine Based Mutation (MBM) for performing the mutation [1]. The main advantage of these operators is that no infeasible chromosome is produced. Thus, no repairing mechanism is required.

Figure 3 shows the described procedures for the two chromosomes randomly produced in problem 3×3 of Table 1.

$$\begin{array}{c} \\ \rightarrow \\ \\ \\ \rightarrow \\ \\ \\ \end{array}\begin{bmatrix}3&1&3\\1&1&1\\2&1&2\\3&2&1\\1&2&2\\2&2&1\\2&3&2\end{bmatrix} \quad \begin{bmatrix}0&0&0\\1&1&1\\0&0&0\\0&0&0\\1&2&2\\0&0&0\\0&0&0\end{bmatrix} \quad \begin{array}{c}\otimes \\ \\ \otimes \\ \\ \\ \\ \\ \end{array}\begin{bmatrix}1&1&3\\3&1&2\\1&2&2\\2&1&1\\2&2&2\\3&2&1\\2&3&2\end{bmatrix} \quad \begin{bmatrix}3&1&2\\1&1&1\\2&1&1\\2&2&2\\1&2&2\\3&2&1\\2&3&2\end{bmatrix}$$

Donor parent   Offspring (first step)   Receptor parent   Offspring (second step)

Figure 3. POX operator.

Suppose gene $\begin{bmatrix}1&1&1\end{bmatrix}$ has been randomly selected from a donor parent. Therefore, all the genes of the donor parents comprising of the operations of one job should be selected. The two selected genes in the donor parents have been shown as arrows and then, the same operations in receptor parent have been displayed as $\otimes$ that should be removed. Likewise, operations of the jobs are similarly transferred from parents to offspring. Thus, with satisfying

be needed.

In MBM mutation operator, a number of operations within a chromosome are randomly selected and later they are assigned to another machine. For example, suppose that there are two target genes as shown in Figure 4 by an arrow; in the next step, the operations of the jobs are assigned to other machines. In this example, operation 2 from job 3, after a processed on machine 1 is transferred to machine 2; but operation 3 from job 2 can only be processed on machine 2, hence remain in same condition

$$\begin{array}{c} \\ \\ \\ \\ \\ \rightarrow \\ \rightarrow \end{array}\begin{bmatrix}3&1&2\\1&1&1\\2&1&1\\2&2&2\\1&2&2\\3&2&1\\2&3&2\end{bmatrix} \quad \begin{bmatrix}3&1&2\\1&1&1\\2&1&1\\2&2&2\\1&2&2\\3&2&2\\2&3&2\end{bmatrix}$$

Figure 4 MBM operator.

PBM mutation operator begins by randomly selecting an operation from the parents and then reinserting it into another position in a new offspring. The remaining operations are copied to the new offspring, without violating the sequence constrains. Fig. 5 displays the mutation procedure in which the selected genes and the new position are displayed by arrows and $\otimes$ mark, respectively.

$$\begin{array}{c} \\ \otimes \\ \\ \\ \\ \\ \rightarrow \end{array}\begin{bmatrix} 3 & 1 & 2 \\ 1 & 1 & 1 \\ 2 & 1 & 1 \\ 2 & 2 & 2 \\ 1 & 2 & 2 \\ 3 & 2 & 2 \\ 2 & 3 & 2 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 3 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \\ 2 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \\ 3 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 3 & 1 & 2 \\ 2 & 1 & 1 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 2 & 3 & 2 \\ 1 & 2 & 2 \\ 3 & 2 & 2 \end{bmatrix}$$

Figure 5. PBM operator.

4.1.4 NSGA-II flow chart

Figure 6 presents flow chart of NSGA-II. The algorithm terminates after reaching a maximal number of generations.

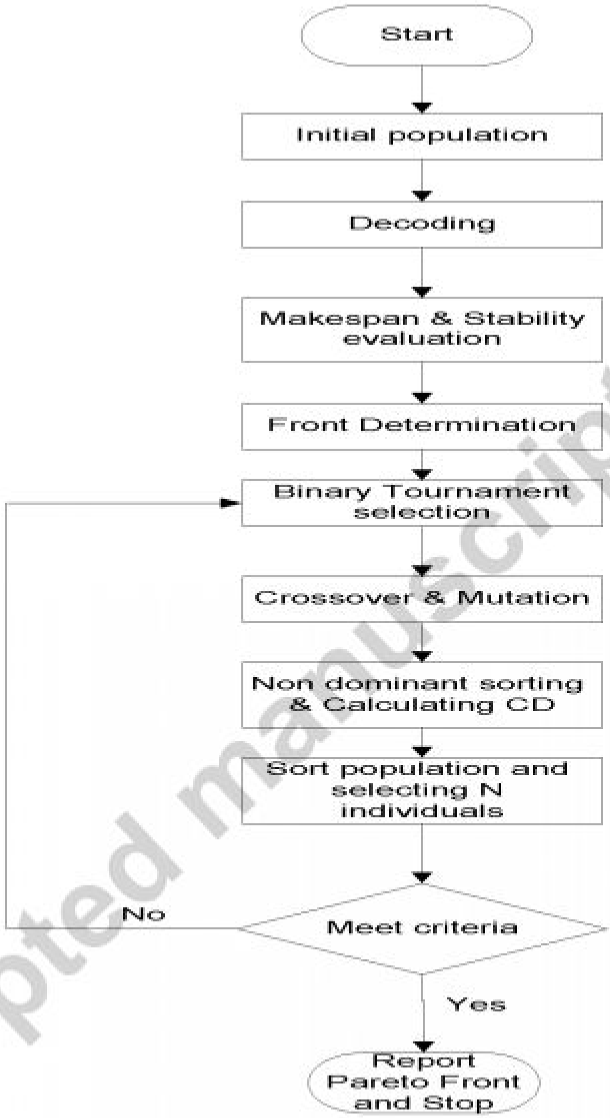

Figure 6. Flow chart of NSGAII algorithm.

4.2 Non-dominated ranking genetic algorithm

Among of all type of MOEAs, the MOEA algorithms in class of Pareto-based approach are the most appropriate algorithms in which it's emphasize is on moving toward the true Pareto-optimal region in the selection process [7]. According to several studies, the two well-known algorithms of this class are NRGA and NSGA-II which are used in this study. The simulation results on benchmark test problems show that both of these algorithms have outperformed many of the classical state-of-the-art algorithms [8,9]. Therefore, using both of the MOEAs is critically appropriate in the coming study of the MO-FJSP.

5. Experimental design

5.1 Parameter setting

All algorithms are coded and executed by using MATLAB R2009a on an Intel® Core™ i5 CPU @2.27GHz with 3.86 GB RAM. The chosen parameter values for both NSGA-II and NRGA algorithms are as follows:

Npop is considered 100, crossover and mutation's rates are 0.8 and 0.3 respectively, and the number of gene for mutation is considered 3.

Nsim is determined based on experience in which the value of 10 results in a higher value. Stopping criteria is determined based on the number of generations which is chosen as 200.

5.2 Benchmark problem and data generation

Currently, there is no standard benchmarking for the scheduling of a stochastic flexible job shop, therefore we have to select deterministic FJSPs for our experiment and modify them into stochastic problems by performing simulation. To cover most range of problems we have considered two examples of T-FJSP: I3 consisting of 10 ×10 and I4 consisting of 10×15 that is taken from Kacem, Hammadi, and Brone [22], and ten examples of P-FJSP: MK01, MK02, MK03, MK04, MK05, MK06, MK07, MK08, MK09, MK10 taken from Brandimarte [4]. They are categorized in Table 2.

Table 2. Benchmark problems.

Type of the problem	Problem	Size	Number of Operation	Source
T-FJSP	I3	10 ×10	30	Kacem, Hammadi, and Brone (2002a)
	I4	10×15	56	
P-FJSP	MK01	10 ×6	56	Brandimarte (1993)
	MK02	10 ×6	58	
	MK03	15 ×8	150	
	MK04	15 ×8	90	
	MK05	15 ×4	106	
	MK06	10 ×15	150	
	MK07	20 ×5	100	
	MK08	20 ×10	225	
	MK09	20 ×10	250	
	MK10	20 ×15	240	

The value of ***MTBF*** and ***MTTR*** is chosen based on $A_g = MTTR/(MTBF + MTTR)$ which denotes the breakdown level of the shop [18]. Holthaus [18] suggested that MTTR is related to $\bar{p}$, where $\boldsymbol{MTTR} \in \{\bar{p}, 5\bar{p}, 10\bar{p}\}$ and $\bar{p}$ denotes the mean processing time of an operation. In this study we have supposed the constant breakdown level for all problems. We consider $A_g = 0.05$ and $MTTR = \bar{p}$.

For instance, in MK05 problem $A_g = 0.05$, $\bar{p} = 6.79$, $MTTR = \bar{p} = 6.79$, $MTBF = (6.79/0.05) - 6.79 = 129.01$.

Thus, on an average of 129.01 time units, a machine is available and the breakdown of a machine occurs with a mean time to repair 6.79 time units

5.3 Performance measures

In this research, we have used six criteria for evaluating and comparing the results of two meta-heuristic algorithms, NSGA-II and NRGA, in scheduling area. These criteria include:

5.3.1 Number of Pareto solution (NPS)

This criterion is used as an indicator for the quality of the algorithm results; higher values of NPS imply that more options are available for managers in decision making situations and administrators have access to more alternative solutions.

5.3.2 Spacing

Spacing is another criterion for the evaluation of the quality of the results. Spacing is used to display the consistency of distance between solutions in a Pareto front. Lower values of Spacing indicate the consistency of spacing between solutions is higher or in other words, higher quality of results [32].

$$\boldsymbol{S} = \sqrt{\frac{1}{\boldsymbol{n}-1}\sum_{i=1}^{n}(\boldsymbol{d}_i - \bar{\boldsymbol{d}})^2} \tag{2}$$

Where

$$\boldsymbol{d}_i = \min_{k \in n, k \neq i} \sum_{m=1}^{2} \left| \boldsymbol{f}_m^i - \boldsymbol{f}_m^k \right| \tag{3}$$

$\boldsymbol{f}_m{}^i$, $\boldsymbol{f}_m{}^k$ denotes the ***i***th and ***k***th value of non-dominated solution in a Pareto front which are makespan for ***m***=1 and stability for ***m***=2

$$\bar{\boldsymbol{d}} = \frac{1}{\boldsymbol{n}}\sum_{i=1}^{n} \boldsymbol{d}_i \qquad \boldsymbol{n} = \text{number of non-dominated solutions in a Pareto front} \tag{4}$$

5.3.3 Diversity

The Diversity criterion specifies the Euclidean distance between the first and the last solutions in a Pareto front. Higher values of Diversity imply greater quality of the set's results [40].

$$\boldsymbol{D} = \sqrt{\sum_{m=1}^{2}(\max_i \boldsymbol{f}_m^i - \min_i \boldsymbol{f}_m^i)^2} \tag{5}$$

5.3.4 Modified Mean Ideal Distance (MMID)

The Mean Ideal Distance is used for the evaluation of the efficiency and convergence power of an algorithm [23]. This criterion presents the distance of Pareto front solutions from an ideal point which is usually the point (0,

0) in a two dimensional graph. Since in this research the objective functions are minimized, lower values of MMID indicates the algorithm's performance are being improved.

MID is calculated as follows:

$$MID = \frac{1}{n}\sum_{i=1}^{n}\sqrt{MSP_i^2 + STB_i^2} \tag{6}$$

Where n denotes the number of non-dominated solution and MSP_i and STB_i are the value of ith non-dominated solution for makespan and stability, respectively. Because of the different scale between objective functions, the range of makespan index is much wider than the range of stability index; for example, in MK9 instance, the range of MSP changes is between 340 and 450 whereas the changes of stability index are between 10 and 40. One unit of change in stability index has a large impact on the MSP which shifts the MID index towards MSP course of change. A suggestion for this situation is the normalization for each of MSP indices and STB indices before the calculation of the MID. The following formulas have been projected for this purpose:

$$NMSP_i = \frac{MSP_i - MinMSP}{MaxMsp - MinMSP} \tag{7}$$

$$NSTB_i = \frac{STB_i - MinSTB}{MaxSTB - MinSTB} \tag{8}$$

In these formulas, MinMSP and MaxMSP are constant values that specify the upper and lower levels of MSP index. These numbers are determined through multiple executions (5 times) for each of the problems as a single-objective problem and within the set of solutions, the minimum and maximum of indices are defined.

The normalized MID (which is the MMID) is calculated as follows:

$$MMID = \frac{1}{n}\sum_{i=1}^{n}\sqrt{NMSP_i^2 + NSTB_i^2} \tag{9}$$

5.3.5 Time

The last criteria which is used to compare between NSGA-II and NRGA, is the Time of running algorithms. Since all parameters in two algorithms are the same, this criterion is an appropriate comparative index among these algorithms.

5.4 Experimental result

In this part the quality of the results of the two implemented algorithms is evaluated by the five discussed criteria. In this study, the null hypothesis (H_0) is the equivalency of the examined criterion of the solutions of the two algorithms, and the alternative hypothesis (H_1) is the unequal criterion of the solutions.

$$H_0 : \mu_1(A) = \mu_2(A)$$
$$H_1 : \mu_1(A) \neq \mu_2(A) \qquad (10)$$

In the above hypothesis, $\mu_1(A)$ is the average of the criterion A in algorithm 1, and $\mu_2(A)$ is the average of the criterion A in the second algorithm. In the following calculations, the t-student test has been used to determine the statistical significance of null hypothesis. The significance level of the test is set to $\alpha = 2\%$ which denotes P-Values with values more than 1% $(\alpha / 2)$ would reject the null hypothesis. In Figure 7 and Figure 8, a set of optimized Pareto solutions and the trend of their improvements in each generation of the two algorithms (NSGA-II and NRGA) are displayed.

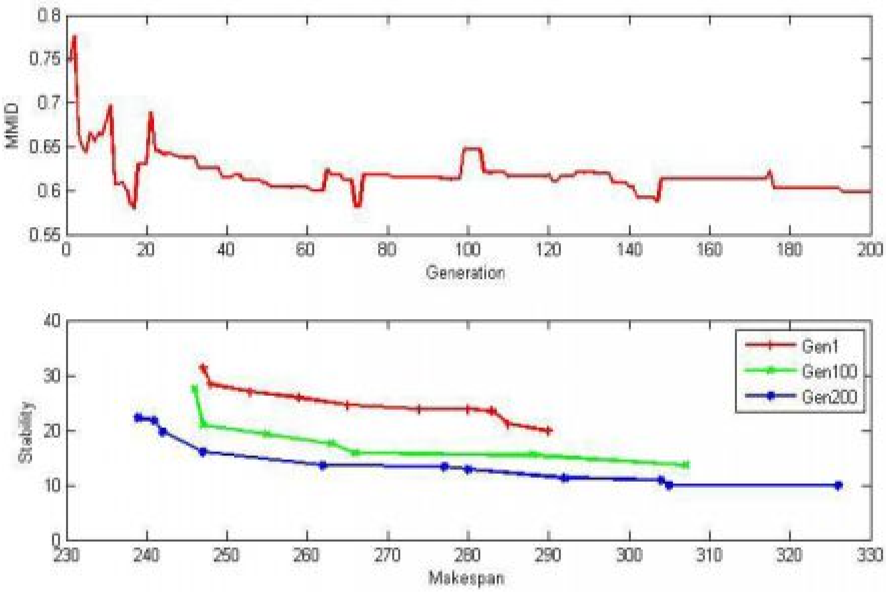

Figure 7. Convergence plot (upper) and Pareto front (lower) of NSGAII for MK10 problem.

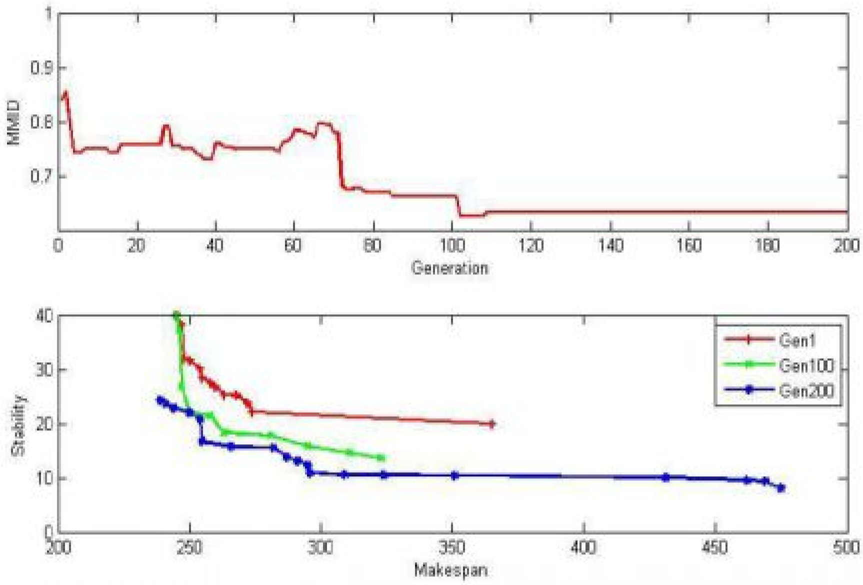

Figure 8. Convergence plot (upper) and Pareto front (lower) of NRGA for MK10 problem.

Overall, we examined the two algorithms for 120 runs (2 algorithms * 12 examples * 5 run per example); the results of these testes are presented in Table 3. In this Table, each row represents an example, and its columns indicate the average values of each criterion.

Table 3. Computational result for twelve test cases.

	NSGAII					NRGA				
Problem	NPS	Diversity	Spacing	MMID	Time	NPS	Diversity	Spacing	MMID	**Time**
I3	2.60	1.17	2.80	0.18	945.16	2.8	10.31	5.85	0.59	1433.85
I4	8.00	16.73	5.43	0.40	1307.80	4.6	22.23	8.23	0.65	1363.07
MK01	4.40	7.75	2.72	0.49	1417.19	4.8	27.95	11.18	0.72	3194.05
MK02	7.80	15.70	6.13	0.58	1395.12	7	42.52	14.32	0.64	2027.26
MK03	8.00	77.32	23.25	0.62	2958.40	4.8	69.64	25.57	0.58	2182.20
MK04	7.20	26.89	8.21	0.50	1963.58	7.4	60.76	17.56	0.63	3042.14
MK05	6.40	25.55	9.44	0.57	2370.14	6.2	113.32	40.13	0.54	2027.76
MK06	12.80	35.76	10.20	0.66	2970.60	3.2	60.95	15.21	0.45	4445.42
MK07	5.40	32.12	10.62	0.53	2178.00	6.4	105.93	42.13	0.54	5194.08
MK08	6.60	43.38	16.66	0.45	4589.68	12.6	153.86	15.21	0.61	4877.03
MK09	7.00	47.13	16.84	0.54	5412.96	2	5.26	3.92	0.33	965.955
MK10	12.20	75.08	23.10	0.63	4906.11	5	19.41	6.75	0.38	1361.24
Mean	**7.37**	**33.72**	**11.28**	**0.51**	**2701.23**	**5.57**	**57.68**	**20.00**	**0.56**	2676.18

5.4.1 Analysis of performance measures

In this part, statistical hypothesis tests over five mentioned indices are performed on the results of the two algorithms (NRGA and NSGA-II). For this purpose, the paired samples student's t-test is performed to compare the average results of these two algorithms. The results of these tests are shown in Table 4.

Table 4. Summary of t-student test for different measures.

Measure	P-Value	Result
Diversity	0%	**H_0 Is Rejected**
Spacing	0%	**H_0 Is Rejected**
MMID	0.1%	**H_0 Is Rejected**
NPS	0%	**H_0 Is Rejected**
Time	23.8%	**H_0 Is Not Rejected**

The determined P-Values for 4 evaluation criteria (Diversity, Spacing, MMID, and NPS) are less than the significant level of the test which is 1%. In this case, these criteria imply that the null hypothesis (equivalent average of both sets) cannot be accepted. However, the Time criterion has a higher value of 1% in which the null hypothesis is not rejected and no statistical significance is found.

In the next few parts, different indices of criteria are examined and in each criterion, the algorithm with higher performance is determined.

5.4.1.1 Diversity

As it can be observed in Figure 9, the NRGA algorithm has a superior quality as compared to the NSGA-II. Also, since the average of the Diversity for NRGA is equal to 57.68 and NSGA-II has an average of 33.72 (from Table 3), NRGA algorithm shows higher quality compared to NSGA-II.

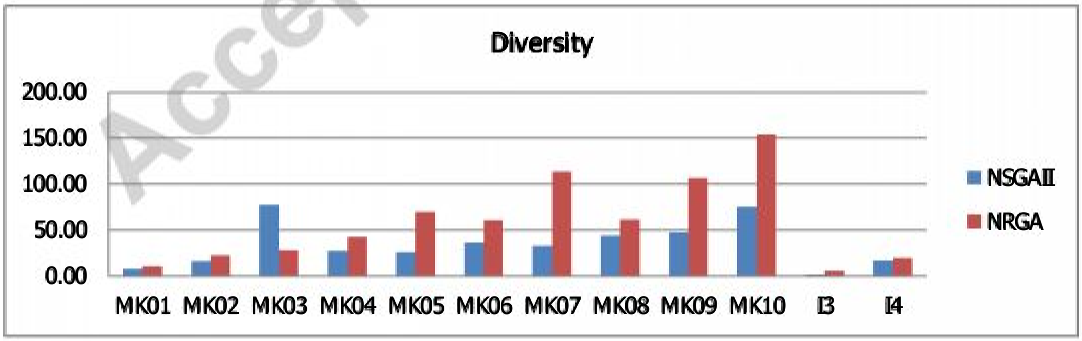

Figure 9. Assessing Diversity criterion of the two algorithms on 12 test problems.

In Figure 10, the Spacing criterion comparison between the two algorithms is plotted. Since lower values of Diversity imply a higher quality, in examples MK03 and MK08, NRGA has a better performance. But since the average of the Diversity criterion for NRGA and NSGA-II are 20 and 11.28, respectively. The NSGA-II algorithm delivers a better quality when compared to the NRGA.

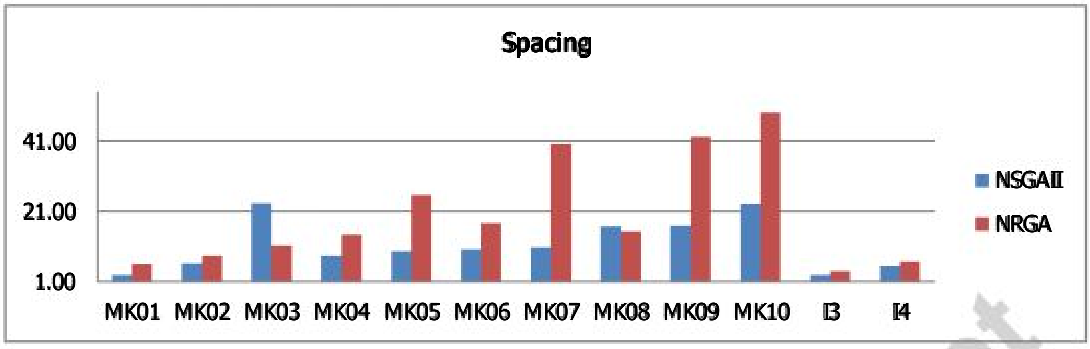

Fig. 10: Assessing spacing criterion of the two algorithms on 12 test problems

5.4.1.3 MMID

According to Table 3, the NSGA-II shows a better performance in seven instances based on the MMID criterion. Comparing the average of the two algorithms also indicates that NSGA-II has a lower value and therefore a better quality of performance.

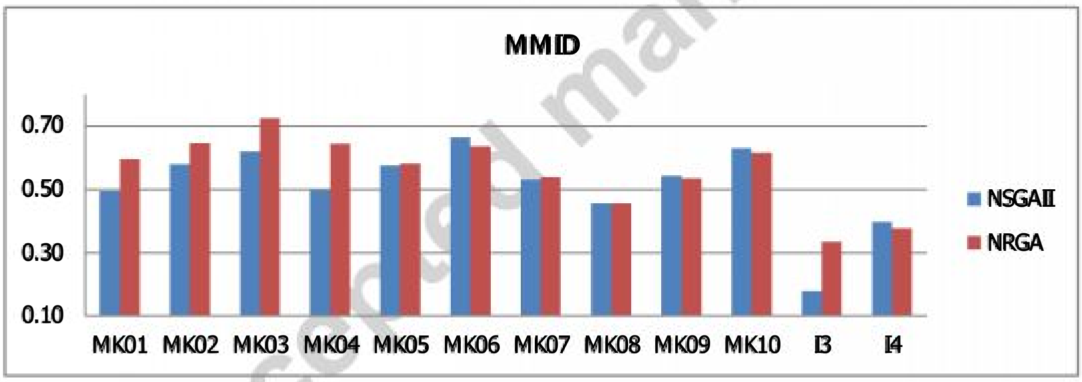

Figure 11. Assessing MMID criterion of the two algorithms on 12 test problems.

5.4.1.4 NPS

In Figure 12, NSGA-II resulted in higher values of NPS in ten cases whereas in other two cases, NRGA generates slightly better results. Overall, NSGA-II with an average of 7.37 produced higher quality of solutions when compared to NRGA with an average of 5.57.

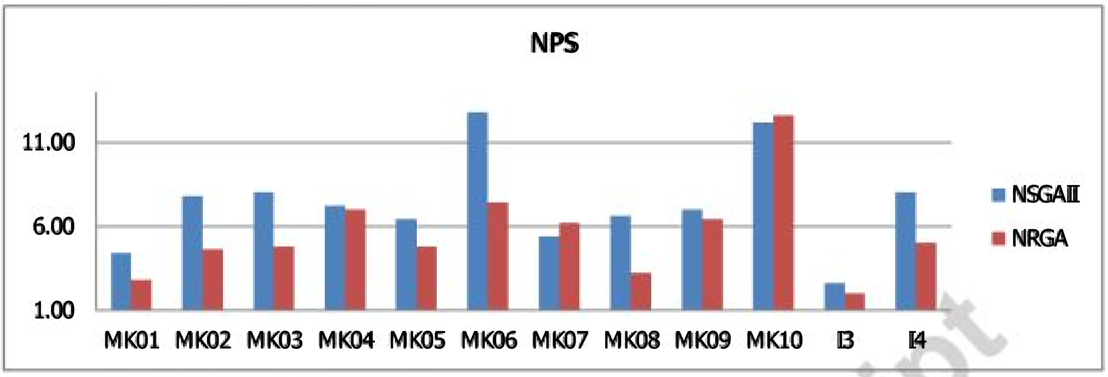

Figure 12. Assessing NPS criterion of the two algorithms on 12 test problems.

5.4.1.5 Time

As discussed earlier, there is no significance statistical difference between the results of the two algorithms based on the Time criterion. In Figure 13, the values of these algorithms are plotted, and merely a trivial difference of results can be observed (However, the average of the NRGA suggest a slightly better result).

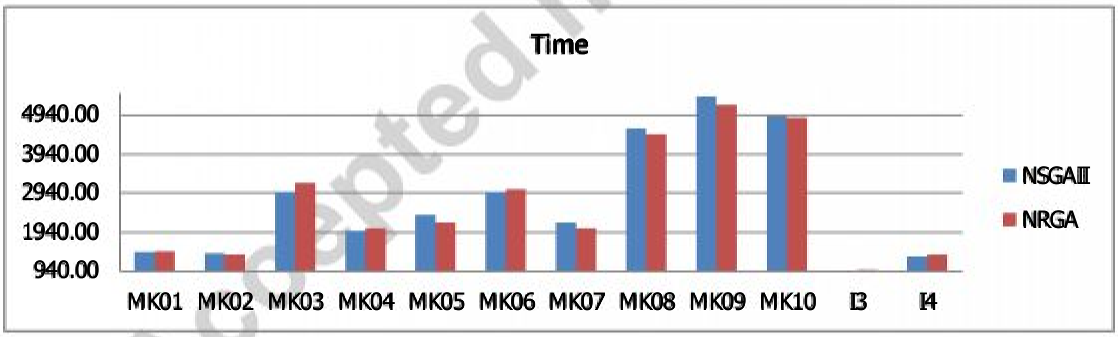

Figure 13. Assessing Time criterion of the two algorithms on 12 test problems.

6. Conclusions and future work

This study evaluates a methodology of optimizing both makespan and stability objectives for dealing with machine breakdown in FJSP. In literature, most of the research in scheduling problems is assessed with a single objective or a single solution for a linear combination of objectives known as Multi Criteria methodology. However, considering several objectives in parallel for scheduling problems is more realistic in manufacturing environment. Also, in most cases, scheduling problems are evaluated under a deterministic environment, although interruption of machine equipment function can happen. Random machine breakdown is a condition where the deterministic state of the problem could be changed into a stochastic problem. Therefore, we proposed a multi-objective methodology for the FJSP scheduling problem in machine breakdown situations. The two algorithms, NRGA and NGSA-II, are applied on multiple cases and also simulation is used to evaluate the state and condition of the machine breakdowns. A set of Pareto solutions are suggested for two objectives of stability and makespan in which these solutions provide a range of solution for making a suitable decision.

Finally, as it is depicted in the previous sections, the main difference between NSGA-II and NRGA is in their selection strategy in which the former uses Tournament selection and the later applies Roulette-wheel algorithm strategy. This differential selection method would be reflected in the quality of solutions. The efficiency of the two algorithms was compared based on the Diversity, Spacing, MMID, NPS, and Time criteria. Statistical hypothesis was used to evaluate the algorithm with higher performance. A summary of the results is shown in Table 5.

Table 5. The leading algorithms in each criterion.

Criteria	Leading Algorithm
Diversity	NRGA
Spacing	NSGAII
MMID	NSGAII
NPS	NSGAII
Time	No Statistical Difference

In future studies, other uncertainties such as job cancellation and arrival of new jobs can be examined. Also the evaluation of other meta-heuristic algorithms against the NSGA-II and NRGA algorithms is another potential future study that can be investigated. Moreover, the use of different types of GA operators and simultaneous measurement of robustness and stability are other possible areas that can be investigated.