Int. J. Nonlinear Anal. Appl. 13 (2022) No. 1, 217-246
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/injaa.2021.24335.2719

Solving a bi-objective flexible flow shop problem with transporter preventive maintenance planning and limited buffers by NSGA-II and MOPSO

Meysam Kazemi Esfeh[a], Amir Abass Shojaei[a,*], Hassan Javanshir[a], Kaveh Khalili Damghani[a]

[a]*School of Industrial Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran*

(Communicated by Madjid Eshaghi Gordji)

Abstract

This study deals with a bi-objective flexible flow shop problem (BOFFSP) with transportation times and preventive maintenance (PM) on transporters via considering limited buffers. The PM actions on transporters are a missing part of the literature of the flexible flow shop problem (FFSP) which before the breakdown occurs, each transporter at each stage is stopped and the PM action is performed on it. The capacity of each intermediate buffer is limited and each job has to wait in the intermediate buffers. By including all these features in the proposed BOFFSP, not only processing times affect the objective functions, but also, the transportation times of jobs, the waiting time of jobs in the intermediate buffers, and availability of transporters in the system are considered in the model and make it a sample of a real-world FFSP. The presented BOFFSP has simultaneously minimized the total completion time and the unavailability of the system. As the problem is NP-hard, a non-dominated sorting genetic algorithm II (NSGA-II) and a multi-objective particle swarm optimization (MOPSO) is proposed to solve the model for large size problems. The experimental results show that the proposed MOPSO relatively outperforms the presented NSGA-II in terms of five different metrics considered to compare their performance. Afterwards, two one-way ANOVA tests are performed. It can be observed MOPSO achieves relatively better results than NSGA-II. Finally, sensitivity analysis is conducted to investigate the sensitivity of the objective functions to the number of jobs and their transportation time at each stage.

Keywords: Flexible flow shop problem, Transportation times, Preventive maintenance, Limited buffers, Multi-objective algorithms.

*Corresponding author
Email addresses: maysam.kazemi60@gmail.com (Meysam Kazemi Esfeh), amir@ashojaie.com (Amir Abass Shojaei), hgavanshir@yahoo.co.uk (Hassan Javanshir), kaveh.khalili@gmail.com (Kaveh Khalili Damghani)

Received: August 2021 *Accepted:* November 2021

1. Introduction

The flexible flow shop problem (FFSP) consists of some different stages in each of which there are many identical machines. There are also some jobs which have to be processed on machines. The machine's configuration is parallel at each stage and the stages are series. All jobs have to visit a machine only once at each stage. In recent decades, researchers have introduced different objective functions for FFSP. For example, minimizing the total completion times (Makespan), the total weighting times, total tardiness or earliness, production costs, and the total unavailability of the system. FFSP researchers generally have focused on makespan in their models and other objective functions have less commonly been investigated. This study presents a bi-objective flexible flow shop problem (BOFFSP) in which two objective functions are proposed to minimize total completion time and the total unavailability of the system (which means the unavailability of total transporters in all stages). Transportation time, the parameter that has been considered by very few researchers is assumed in the presented study. The transportation time occurs if a job transfers from one machine to another. Another assumption that is a missing part in previous researches is considering preventive maintenance (PM) actions on transporters. Most recent studies concentrated on PM actions on machines, but the PM actions on transporters is an important point that according to our knowledge has not been assumed in any study. According to the PM actions, before the breakdown occurs, each transporter at each stage is stopped and the PM action is performed on it. Also, the limited intermediate buffer is considered between each stage to represent a more realistic FFSP. By considering all these factors, the proposed bi-objective flexible flow shop problem (BOFFSP) can be considered as a sample of a real-world FFSP. This study introduces a BOFFSP with transportation times, PM actions on transporters, and limited intermediate buffers. Since the FFSP is NP-hard [17], a non-dominated sorting genetic algorithm II (NSGA-II) and a multi-objective particle swarm optimization (MOPSO) is proposed to deal with a large size problem.
The rest of this study is organized as follows: a literature review is presented in Section 2. Problem description and mathematical model are described in Section 3. In Section 4, the proposed metaheuristic algorithms, including an NSGA-II and a MOPSO are explained. The computational experiments and the results obtained are analyzed and reported in Section 5. Statistical tests and sensitivity analysis are performed in this section, as well. At last, conclusions and future research are offered in Section 6.

2. Literature review

During the last two decades, a wide range of studies has been performed on scheduling problems, some of them focused on single objective functions trying to optimize them and others propose different multi-objective problems. Here, the literature is investigated and a summary of what has been done with bi-objective or multi-objective scheduling problems is presented. Naderi et al. studied improved simulated annealing (SA) for hybrid flow shops with sequence-dependent setup and transportation times. The objective functions of their model were to minimize total completion times and total tardiness. The performance of their proposed algorithm was compared to the other metaheuristics in the literature and demonstrates the superiority of their algorithm [20]. Another effort was about Berrichi et al., which presented a bi-objective optimization algorithm in parallel machine problems on joint production and maintenance schedules. Two objective functions of their model were minimization of makespan and system unavailability. For solving the model, two metaheuristic algorithms, including a non-dominated sorting genetic algorithm II (NSGA-II) and a weighted sum genetic algorithm (WSGA) was developed and the computational results showed that NSGA-II significantly outperforms WSGA [4]. Naderi et al. proposed an electromagnetism-like

mechanism (EM) and a simulated annealing (SA) algorithm for the flow shop scheduling problem with transportation times. The objective functions of their model were to minimize the makespan and total weighted tardiness. The computational results indicated that the proposed EM achieved better quality solutions [19]. Berrichi et al. developed a multi-objective ant colony optimization (MOACO) approach to optimize production and maintenance scheduling problems. The objective functions of the model were to minimize total completion times and unavailability of the system. The quality of solutions of their proposed MOACO was compared to the quality of the solutions of the non-dominated sorting genetic algorithm II (NSGA-II) and the strength of Pareto evolutionary algorithm II (SPEA-II). Results showed the better performance of their proposed MOACO [5]. Mohammadi et al. presented a bi-objective mixed-integer linear programming model in a multi-stage production system in real automotive industries in France. Their model contains two inter-related decisions: which quality characteristic needs what kind of inspections and when the inspection of these characteristics should be performed. They utilized different evolution algorithms combined with Monte Carlo methods [18]. Khalili and Tavakkoli-Moghaddam proposed a multi-objective electromagnetism algorithm (MOEA) and multi-objective simulated annealing (MOSA) for bi-objective flow shop scheduling. The performance of their proposed metaheuristic algorithms was compared to the other metaheuristics in the literature. The computational results of solving different test problems showed that the proposed MOEA had better quality solutions, comparing with other metaheuristics [15]. Tavakkoli-Moghaddam et al. investigated multi-objective job shop scheduling problems with sequence-dependent setup times. Their objective functions were minimizing the total weighted mean completion time and the sum of weighted tardiness and earliness. They proposed a particle swarm optimization (PSO) with genetic operators, including crossover and mutation to update particles and improve particles by variable neighborhood search. Their algorithm was based on a Pareto archive PSO, in which the crowding measure-based archive updating method was updated with the global best position selected. The proposed PSO algorithm was compared to a prominent multi-objective genetic algorithm (MOGA) and NSGA-II. The experimental results showed that their proposed PSO had the ability to attain better quality solutions than MOGA and NSGA-II, especially for large-sized problems [32]. Also, another effort was offered by Amin-Tahmasebi and Tavakkoli-Moghaddam on a bi-objective flow shop scheduling problem with sequence-dependent setup times. The objective functions were minimizing the total completion time and the total tardiness/earliness of all jobs. A multi-objective immune system (MOIS) was introduced to solve the model and compared to well-established multi-objective genetic algorithms named SPEA2+ and sub-population genetic algorithm (SPGA). The experimental results indicated the superiority of the proposed MOIS over SPGA and SPEA2+ [2]. Tavakkoli-Moghaddam et al. (2011) developed a multi-objective Pareto archive combining with particle swarm optimization (PSO) and variable neighborhood search (VNS) algorithms for a bi-objective job shop scheduling problem with sequence-dependent setup times and ready times. The objective functions of their model were minimizing the weighted mean flow time and total penalty of tardiness and earliness. The output of the proposed algorithms were compared with the non-dominated sorting genetic algorithm II (NSGA-II) and the strength Pareto evolutionary algorithm II (SPEA-II). The experimental results indicated that their proposed algorithms achieved better solutions, especially for large-scale problems [31]. Choi and Wang studied a decomposition-based approach (DBA) in which a reactive approach and a reactive-proactive approach were combined [6]. Wang studied a bi-objective optimization problem for production scheduling and preventive maintenance in a single machine environment with sequence-dependent setup times. Two objective functions of the proposed model were minimizing the total expected completion times and maximum expected time failures of the machine at the same time. The NSGA-II and SPEA-II were developed to solve the model and the computational experiments showed that

NSGA-II reached better quality solutions than SPEA-II [34]. Naderi-Beni et al. developed a fuzzy bi-objective parallel machine scheduling problem with machine eligibility restriction and sequence-dependent setup times. Also, a fuzzy multi-objective particle swarm optimization (FMOPSO) and a fuzzy non-dominated sorting genetic algorithm (FNSGA-II) were developed to solve the model. According to the gained results, FNSGA-II was more effective than FMOPSO, especially in large-scale problems [21]. A multi-objective hybrid flexible flow shop problem with limited waiting times and machine sequence-dependent setup time constraints were considered by Attar et al. The objective functions of their model were to minimize the total weighted tardiness and maximum completion times. Two metaheuristic algorithms based on SPEA-II and MOPSO were proposed to solve the model and the evaluation results implied that MOPSO reached better quality solutions, comparing to SPEA-II in both small and large-sized problems [3]. Jolai et al. offered a bi-objective no-wait two-stage flexible flow shop scheduling problem. The objective functions of the model were minimizing makespan and tardiness of jobs. Three bi-objective methods based on classical weighted simulated annealing (CWSA), normalized weighted simulated annealing (NWSA), and fuzzy simulated annealing (FSA) were proposed to solve the model [14]. Ghodratnama et al. presented a multi-objective multi-route flexible flow line problem. The objective functions of their model simultaneously minimize the weighted delay, the capital for the purchase of the processors at stations, and capital dedicated to select the optimum processing route of parts. Also, to solve the model, NSGA-II and MOPSO were proposed and the results denoted that in most of the times, the MOPSO had produced more Pareto solutions compared to the NSGA-II [11]. Multi-objective optimization for parallel machine scheduling integrated with multi-resources preventive maintenance planning was introduced by Wang and Liu introduced. The objective functions had the ability to simultaneously minimize the makespan, the unavailability of the machine system, and the unavailability of the mold system. A multi-objective integrated optimization method with NSGA-II adaptation was proposed to solve the model [35]. Shahidi-Zadeh et al. expressed a bi-objective unrelated parallel batch-processing machine-scheduling problem to optimize the makespan, tardiness/earliness penalties, and the purchasing costs of machines, at the same time. As the problem is strongly NP-hard, they proposed a multi-objective harmony search (MOHS) algorithm and compared the results to three well-known metaheuristic algorithms including NSGA-II, MOPSO, and MOACO. According to the experimental results, their proposed MOHS had better performances compared to the other metaheuristics [27]. Zandieh et al. studied multi-objective production scheduling and maintenance planning in a hybrid flow shop system. The objective functions of their model were minimizing makespan and unavailability of the system. Two metaheuristic algorithms have been developed based on NSGA-II and hybridized NSGA-II (HNSGA-II) to solve the model. The computational results proved the better performances of HNSGA-II than NSGA-II [39]. Noori-Darvish and Tavakkoli-Moghaddam addressed bi-objective open shop scheduling problem with sequence-dependent setup times. The objective functions to be optimized were makespan and total tardiness of jobs and to achieve Pareto optimal sets of different test problems. The improved NSGA-II was developed and compared to SPEA-II. The computational results specified the efficiency of the proposed NSGA-II [22]. Amelian et al. examined multi-objective optimization for stochastic failure-prone job shop scheduling problems. The objective functions of their model were to determine the best sequence of jobs, optimal production rate, and optimum preventive maintenance period for simultaneous optimization of the three criteria of the sum of earliness and tardiness, system reliability, and energy consumptions. Their solution methodology was a hybrid of simulation and NSGA-II algorithm. The experimental results of solving test problems with different sizes revealed the effectiveness of the proposed algorithm [1]. Moghaddam explored multi-objective maintenance and replacement scheduling in a repairable multi-workstation manufacturing system. The objective functions to be optimized were

total operational costs, overall reliability, and system unavailability. They proposed a hybrid Monte Carlo simulation and goal programming procedure to solve the problem and the evaluation of results proved the effectiveness of the proposed solution methodology [17]. Rooeinfar et al. presented a stochastic flexible flow shop scheduling problems with limited buffers and fixed interval preventive maintenance. Their solution methodologies were combining three metaheuristic algorithms including GA, SA, and PSO with simulation software named HSIM-META. The simulation software was used as an initial population for the tuned metaheuristic parameters to gain better quality solutions by investigating different approaches. The computational results showed that their proposed HSIM-META had better quality performances in terms of accuracy and speed in solving the problems [26]. Raissi et al. offered stochastic flexible flow shop scheduling problem with preventive maintenance and budget constraint. They utilized three well-known metaheuristic algorithms that had efficient quality solutions in the literature. Both proposed algorithms were based on combining PSO with PSA methods under different random generation policies. Also, the third one was based on merging GA with PSA. By solving different problems, the gained results revealed that combining PSO with SA outperforms other metaheuristic algorithms in terms of makespan and CPU time [24]. A flexible flow shop scheduling problem with machine-dependent processing stages was proposed by Hasani and Hosseini. They conducted the FFSSP considering production costs and energy consumption as two conflict objective functions. The NSGA-II was used to solve the problem and compared to SPEA2 and the results showed the effectiveness of their proposed method [12]. Another study recently was about Gheisariha et al. that enhanced the multi-objective harmony search (EMOHS) algorithm and a Gaussian mutation to solve the flexible flow shop scheduling problem with the sequence-based setup time, transportation time, and probable rework. They designed a simulation-optimization framework for implementing the rework process. Their proposed algorithm had four attractive and distinctive features. The EMOHS was compared with five well-known algorithms including NSGA-II, MOPSO, PESA II, MOHS, and MOIWO, and demonstrate the performance of EMOHS [10]. To the best of our knowledge, a flexible flow shop problem considering transportation times, limited intermediate buffers, and PM actions on transporters that try to minimize the total completion time and the unavailability of the system at the same time have never been investigated before. Furthermore, minimizing the unavailability of the system has not been studied in flexible flow shop problems.

3. Problem description and mathematical model

At first, the problem with suggested considerations is defined. Assume an FFSP with J jobs, M machines, K intermediate buffers, R transporters, and S working stages as displayed in Fig.1. The preventive maintenance (PM) activities can be performed on each transporter between any stages, where M_{ns} represents machine n at stage s and R_{ns} is transportation n at stage s. Also, there is an intermediate buffer between two consecutive stages. The objective functions of the model are minimizing (1) the total completion time and (2) the unavailability of the system. A sample of the proposed FFSP is depicted in Fig.2 with 3 stages and 5 jobs. There are 1, 3, and 2 machines allocated to stages 1, 2, and 3, respectively. Also, each stage has three transporters to transfer the jobs. As it is shown in Fig.2, after transferring job 1 to machine 1, the process of job 1 is started on machine 1. When the process of job 1 is finished, it transfers to stage 2 by transporter 1 from time 5 to time 7. The PM activity occurs on this transporter and job 1 has to wait in a buffer in stage 2 from time 8 to 9. After that, job 1 is processed on machine 1 in stage 2 from time 10 to 12. Then, job 1 is transferred to stage 3 by transporter 1 from time 13 to 16. Also, between times 17 to 19, the PM activity is performed on transporter 1, and job 1 should be waiting in the buffer in stage 3 at the same time. Finally, the process of job 1 is done from time 20 to 22. Sometimes more than one job

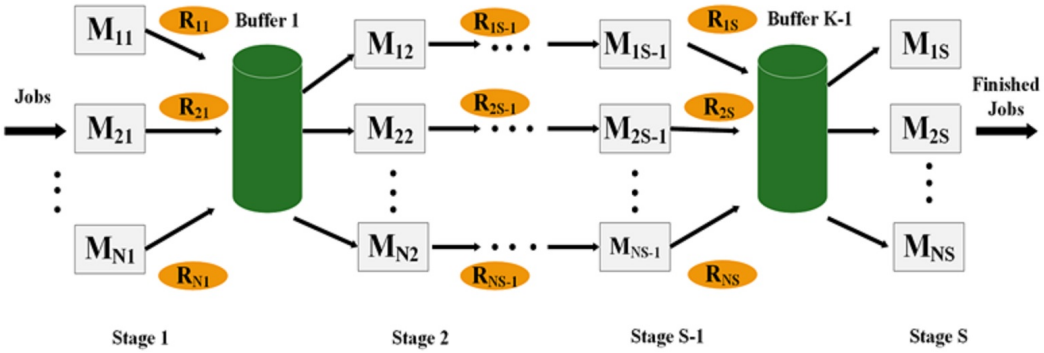

Figure 1: The schematic view of the proposed FFSP

waits in the buffers. For example, jobs 2 and 3 are placed in a buffer of stage 2 from time 11 to 12, or jobs 3 and 4 are placed in a buffer of stage 2 from time 13 to 14.

The other assumptions of the proposed model are as following:

- There is a set of jobs indicated by $J(j = 1, 2, \ldots, J)$ that are accessible at the starting time and no job has permission to be canceled before completion.
- There is a set of transporters indexed by $R(r = 1, 2, \ldots, R)$ to transfer any jobs between two consecutive stages.
- Each stage has identical parallel machines which showed by $(m = 1, 2, \ldots, M_s)$.
- The set of K, depicted by $(k = 1, 2, \ldots, K)$ shows the number of intermediate buffers with the capacity of Z_k.
- Each job has to be processed only on one machine at each stage and consecutively through all stages.
- All jobs at each stage have to be processed continuously with no preemption until they are finished at that stage.
- All machines and all transporters are available at time zero. On the other hand, no machines/transporters are deteriorating at the starting time.
- Each machine should process only one job at a time, and each job could not be visited by any machine more than once.
- The preventive maintenance activities are executed on each transporter by the fixed intervals L_{rs}.
- After the preventive maintenance activity is done on the transporter, there is no probability of a succeeding transporter breakdown.
- Each transporter can transfer only one job between two consecutive stages at the same time.
- Each machine at each stage has a specific transporter so that each job has to be transferred to the next stage by that specific transporter.
- The total capacity of each intermediate buffer is independent of the job type.

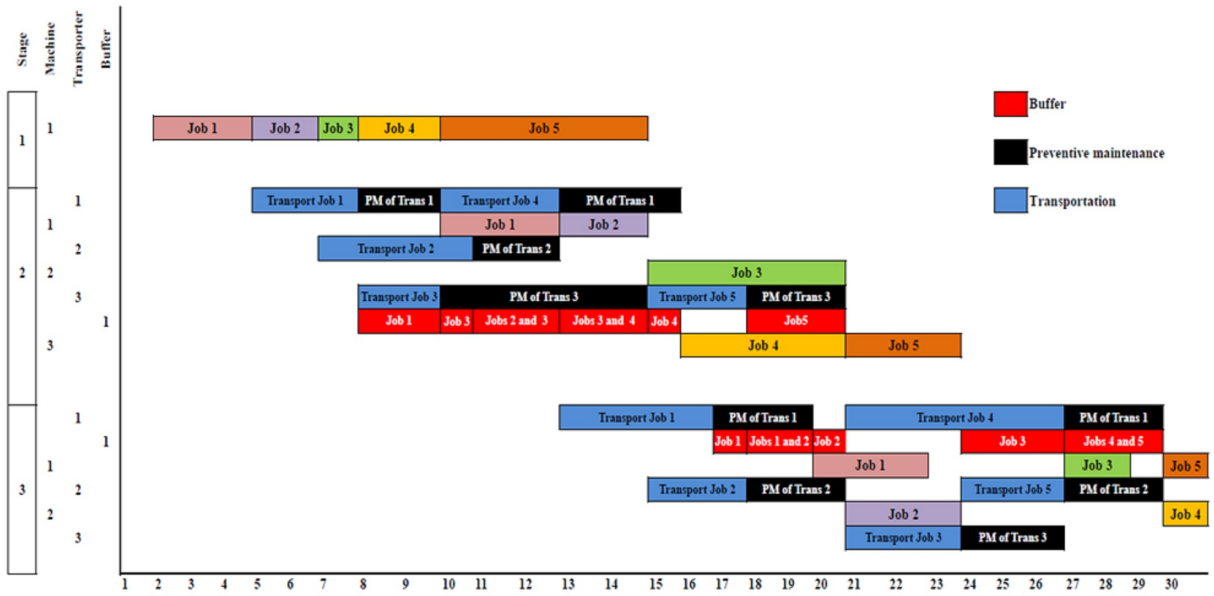

Figure 2: An example of the proposed FFSP

3.1. Notations

To present the model using mathematical relations, the following notations are described:

Indices

j Index of jobs $\{j, d = 1, 2, \ldots, J\}$

s Index of stages $\{s = 1, 2, \ldots, S\}$

m Index of machines at stage s $\{m = 1, 2, \ldots, M_s\}$

r Index of transporters $\{r = 1, 2, \ldots, R\}$

k Index of intermediate buffers $\{k = 1, 2, \ldots, K\}$

o Index of job sequence $\{O, U = 1, 2, \ldots, O_m\}$

n Index of maintenance activity which is done on transporter r $\{n = 1, 2, \ldots, V_{ms}\}$

t Index of time period $\{t = 1, 2, \ldots, T\}$

Parameters

P_{js} The processing time of job j at stage s

T_{js} The transportation time of job j at stage s

Z_{jk} The capacity of intermediate buffer k for job j

λ_{rs} The failure rate of transporter r at stage s

μ_{rs} The repair rate of transporter r at stage s

D_{rs} The duration of maintenance activity of transporter r at stage s

Q_{jk} The maximum capacity of job j at intermediate buffer k

I_{ms} The processing speed of machine m at stage s

V_{jrs} The transportation speed of transporter r to carry job j at stage s

W_{jk} The waiting time of job j in buffer k

α The minimum availability of the system

M An arbitrary large positive number

Dependent decision variables

C_{js} Completion time of job j at stage s

L_{rs} The time between two consecutive maintenance activities on transporter r at stage s

n_{rs} The number of maintenance activities on transporter r at stage s

$$U_{jk}(t) = \begin{cases} 1, & \text{if job } j \text{ is in intermediate buffer } k \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases}$$

T_{rs} The completion time of the last PM activity on transporter r at stage s

$A_{rs}(t)$ The availability of transporter r at stage s at time t

$A_s(t)$ The unavailability of stage s at time t

$A_{sys}(t)$ The unavailability of the system at time t

$$W = \begin{cases} 1, & \text{one of the two associated constraint is activated and the other is neutralized;} \\ 0, & \text{vice versa.} \end{cases}$$

Independent decision variables

$$X_{joms} = \begin{cases} 1, & \text{if job } j \text{ is processed in seqeunce } o \text{ on machine } m \text{ at stage } s; \\ 0, & \text{otherwise.} \end{cases}$$

$$X_{jj'os} = \begin{cases} 1, & \text{if job } j \text{ is processed in seqeunce } o \text{ before job } j' \text{ at stage } s; \\ 0, & \text{otherwise.} \end{cases}$$

$$Y_{nrs} = \begin{cases} 1, & \text{if } n\text{th maintenance activity on transporter } r \text{ at stage } s \text{ is performed;} \\ 0, & \text{otherwise.} \end{cases}$$

$$Y_{jos} = \begin{cases} 1, & \text{if job } j \text{ is transferred by } o\text{th operation;} \\ 0, & \text{otherwise.} \end{cases}$$

$$Z_{joks} = \begin{cases} 1, & \text{if job } j \text{ is waited in sequence } o \text{ in buffer } k \text{ at stage } s; \\ 0, & \text{otherwise.} \end{cases}$$

$$Z_{jj'os} = \begin{cases} 1, & \text{if job } j \text{ is waited in sequence } o \text{ before job } j' \text{ at stage } s; \\ 0, & \text{otherwise.} \end{cases}$$

3.2. Mathematical model

The availability of transporter r depends on the failure rate of transporter r at stage s (λ_{rs}) and the repair rate of transporter r at stage s (μ_{rs}). Also, both failure and repair rates of each transporter r are assumed to be generated according to the exponential distribution. The availability of transporter r in stage s at time t under PM activities can be calculated according to the following equation presented by Villemeur [33]:

$$A_{rs}(t) = \frac{\mu_{rs}}{\mu_{rs}+\lambda_{rs}} + \frac{\lambda_{rs}}{\mu_{rs}+\lambda_{rs}} exp[-(\mu_{rs}+\lambda_{rs})(t-T_{rs})] \tag{3.1}$$

Regarding Eq.(3.1), the availability $A_{rs}(t)$ is a time decreasing function, and unavailability $1 - A_{rs}(t)$ is a time increasing function. Hence, if no PM activity is done on transporter r at stage s, its unavailability increases. Also, the configuration of stages is a series in which there are identical parallel machines at each stage, and transporters transfer each job between two consecutive stages. A specific stage is unavailable if all of its transporters are unavailable. Therefore, the unavailability of stage s and unavailability of the total system is calculated by Eq.(3.2) and Eq.(3.3) respectively.

$$A_s(t) = \prod_{r=1}^{R}(1 - A_{rs}(t)) \tag{3.2}$$

$$A_{sys}(t) = 1 - \prod_{s=1}^{S}(1 - A_s(t)) \tag{3.3}$$

Also, the waiting time of job j in buffer k (W_{jk}) is calculated as the following Eq.(3.4):

$$W_{jk} = C_{j,s} - X_{joms}(T_{js} * V_{jrs} + P_{js} * I_{ms}) - n_{rs}L_{rs}Y_{nrs} - D_{rs} - C_{js-1} \tag{3.4}$$

The proposed mathematical model which is an extension of the basic model proposed by Zabihzadeh and Rezaeian [37] is presented as below:

$$OF1 = \{\min\{\max(C_{js})\}\} \tag{3.5}$$

$$OF2 = \left\{\min\{A_{sys}(t) = 1 - \prod_{s=1}^{S}(1 - A_s(t))\}\right\} \tag{3.6}$$

St:

$$\sum_{o=1}^{O_m}\sum_{m=1}^{N_S} X_{joms} = 1 \qquad \forall j \in J; s \in S \tag{3.7}$$

$$\sum_{j=1}^{J}\sum_{o=1}^{O_m} X_{joms} = 1 \qquad \forall m \in M_s; s \in S \tag{3.8}$$

$$\sum_{o=1}^{O_m} X_{joms} = 1 \qquad \forall j \in J; \forall m \in M_s; s \in S \tag{3.9}$$

$$\sum_{j=1}^{J}\sum_{s=1}^{S} X_{joms} = 1 \qquad \forall o \in O_m; \forall m \in M_s \tag{3.10}$$

$$C_{js}+M(2-(X_{joms}+X_{jom's-1}+Z_{joks}+Z_{jok-1s-1}))\geq C_{js-1}+T_{js}*V_{jrs}+P_{js}*I_{ms}+n_{rs}L_{rs}Y_{nrs}+D_{rs}$$
$$\forall j\in J;k\in K;\forall s\in S;\forall o,o'\in O_m;\forall m,m'\in M_s;\forall r\in R \quad (3.11)$$

$$C_{j's-1}+M(4-(X_{joms}+X_{j'o'ms}+X_{j'o'm's-1}+X_{jj'os}+Z_{joks}+Z_{j'o'ks}+Z_{j'o'k-1s-1}+Z_{jj'os}))$$
$$\geq C_{js}+T_{js}*V_{jrs}+P_{js}*I_{ms}+n_{rs}L_{rs}Y_{nrs}+D_{rs}$$
$$\forall j,j'\in J,j\neq j';\forall k\in K;\forall s\in S;\forall o,o'\in O_m;\forall m,m'\in M_s;\forall r\in R \quad (3.12)$$

$$C_{js-1}+M(3-(X_{joms}+X_{j'o'ms}+X_{jom's-1}-X_{jj'os}+Z_{joks}+Z_{j'o'ks}+Z_{jok-1s-1}-Z_{jj'os}))$$
$$\geq C_{j's}+T_{j's}*V_{j'rs}+P_{js}*I_{ms}+n_{rs}L_{rs}Y_{nrs}+D_{rs}$$
$$\forall j,j'\in J,j\neq j';\forall k\in K;\forall s\in S;\forall o,o'\in O_m;\forall m,m'\in M_s;\forall r\in R \quad (3.13)$$

$$\sum_{o=1}^{O_m}X_{joms}-M(1-X_{jom's-1})\leq 0 \qquad \forall j\in J;\forall s\in S;\forall m,m'\in M_s \quad (3.14)$$

$$X_{jom's-1}-M(2-(X_{joms}+X_{j'o'ms}+Z_{joks}+Z_{j'o'ks}+X_{jj'os}-Z_{jj'os}))\leq 0$$
$$\forall j,j'\in J,\ j\neq j';\forall k\in K;\forall s\in S;\forall o,o'\in O_m;\forall m,m'\in M_s \quad (3.15)$$

$$X_{j'o'm's-1}-M(4-(X_{j'o'ms}+X_{joms}+Z_{j'o'ks}+Z_{joks}-X_{jj'os}-Z_{jj'os}))\leq 0$$
$$\forall j,j'\in J,\ j\neq j';\forall k\in K;\forall s\in S;\forall o,o'\in O_m;\forall m,m'\in M_s \quad (3.16)$$

$$C_{js-1}-M\left(1-U_{jk}(t)\right)\leq t\leq C_{js}+M\left(1-U_{jk}(t)\right) \qquad \forall j\in J;s\in S;\forall k\in K \quad (3.17)$$

$$C_{js-1}+(1-X_{joms})\geq C_{j's}-(1-X_{j'o'ms})\,M \quad \forall j,j'\in J,\ j\neq j';\forall o,o'\in O_m;\forall m\in M_s;\forall s\in S \quad (3.18)$$

$$(n_{rs}L_{rs}Y_{nrs}-C_{js})X_{joms}\geq -M(1-W) \quad \forall j\in J;\forall s\in S;\forall r\in R;\forall m\in M_s;\forall o\in O_m;n\in V_{ms} \quad (3.19)$$

$$(C_{js}-n_{rs}L_{rs}Y_{nrs}-T_{js}*V_{jrs}-P_{js}*I_{ms}-D_{rs})X_{joms}\geq -M(W)$$
$$\forall j\in J;\forall k\in K;\forall s\in S;\forall r\in R;\forall m\in M_s;\forall o\in O_m;n\in V_{ms} \quad (3.20)$$

$$\sum_{j=1}^{J}U_{jk}(t)Z_{jk}\leq\sum_{j=1}^{J}Q_{jk} \qquad \forall k\in K \quad (3.21)$$

$$T_{rs} \leq n_{rs}.L_{rs} \qquad s \in S; \forall r \in R \tag{3.22}$$

$$1 - A_{sys}(t) \geq \alpha \tag{3.23}$$

$$C_{js}, P_{js}, T_{js}, D_{rs}, W_{js} \geq 0 \qquad \forall j \in J; \forall s \in S; \forall r \in R \tag{3.24}$$

$$X_{joms}, X_{jj'os}, Y_{nrs}, Z_{joks}, Z_{jj'os} \in \{0, 1\}$$
$$\forall j, j' \in J; j \neq j'; \forall o \in O_m; \forall m \in M_s; \forall k \in K; \forall s \in S; n \in V_{ms}; \forall r \in R \tag{3.25}$$

Equations (3.5) and (3.6) represent the objective functions of the model which are minimizing the total completion time and minimizing the unavailability of the system, respectively. Constraint (3.7) ensures that each job should be assigned to only one machine at each stage. Constraint (3.8) implies that the first operation of each job can be allocated to only one machine at each stage. Constraints (3.9) and (3.10) confirm that each transporter can carry only one job at a specific time. Constraint (3.11) guarantees that each job can be processed at each stage if its process has been finished at the previous stage. Constraints (3.12) and (3.13) state that two jobs cannot be processed on one machine at the same time. Constraint (3.14) shows that each job can be transferred to the next stage if the transportation of the previous stage is finished. Constraints (3.15) and (3.16) state that the transport sequence is the same as the sequence of jobs on a machine at each stage. Constraints (3.17) and (3.18) imply that no intervention should be taken among jobs on a specific transporter at any stage if the transporter is available (failure on the transporter has not happened). Constraints (3.19) and (3.20) confirm that no overlap should be taken among the transportation of jobs and maintenance activities on transporters. Constraint (3.21) ensures that the capacity of an intermediate buffer should not exceed the maximum capacity of jobs at that buffer. Associated together, constraints (3.22) and (3.23) control the availability of the system. Constraint (3.24) states that the continuous variables should take positive values, and constraint (3.25) specifies the binary decision variables.

4. Metaheuristic solution approaches

4.1. The proposed metaheuristics

Most NP-hard combinatorial optimization problems like flexible flow shop scheduling cannot be solved in real-world large size applications in a reasonable time. Hence, metaheuristic approaches are employed to solve these problems and achieve near-optimum solutions in a reasonable computational time. Therefore, to solve the described bi-objective flexible flow shop scheduling problem, a non-dominated sorting genetic algorithm II (NSGA-II) and a multi-objective particle swarm optimization (MOPSO) is put into practice to deal with this issue.

4.2. Non-dominated sorting genetic algorithm II (NSGA-II)

NSGA-II is one of the best known, fast sorting, and elite multi-objective optimization algorithms. NSGA-II has the ability to optimize each objective function without affected by any other result at the same time. The algorithm has three special characteristics: fast non-dominated sorting approach, fast crowded distance estimation procedure, and simple crowded comparison operator [36]. NSGA-II normally consists of six different steps as follows: Step 1 - Population initialization: in which the population is initialized considering the problem range and constraints; Step 2 – Non-dominated

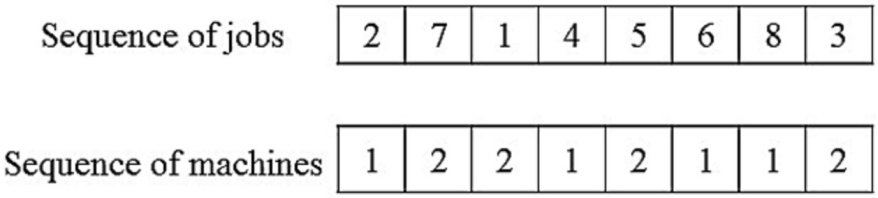

Figure 3: A sample of a chromosome

sorting: where the sorting process is carried out based on non-domination criteria of the population. Step 3 - Crowding distance: when the sorting is finished, the crowding distance value is assigned front wise. Rank and crowding distance is used to select individuals from the population. Step 4 – Selection: a binary tournament selection with a crowded-comparison operator is applied to perform the selection of individuals. Step 5 - Genetic Operators: crossover and mutation operators are generally employed. Step 6 - Recombination and selection: where the current generation population is combined with the offspring and the populations of the next generation are selected [8]. Afterwards, the new generation is filled by each front as long as the new population size is larger than the current population size [28]. The following are the components of the proposed NSGA-II.

4.2.1. Solution representation

In this section, the applied approach to display a random solution is described in detail. The first step concerning the algorithms is to relate the main problem to the structure of an algorithm, that is, to build a bridge between the problem and the solution space in which evolution takes place. First, a method must be selected to display the solutions (chromosomes). In a classical manner, a chromosome of the solution population can be a list of integers for discrete combinatorial optimization problems, a vector of real numbers for continuous problems, a string of binary digits for Boolean problems, and even a combination of representations, if necessary. Therefore, choosing an appropriate representation method is one of the most important steps to design and implement a metaheuristic. For the proposed problem, the designed chromosome consists of two parts. The first part specifies the sequence of jobs to be performed and the second part is the assignment of machines to jobs that are randomly determined according to the number of each machine at each stage. The jobs are assigned to machines at each stage such that the jobs (according to the sequence in the first part) are assigned to each machine (according to the sequence in the second part) until all jobs are assigned. Fig.3 displays a chromosome with 8 jobs and 2 machines.

4.2.2. Generating an initial solution

The initial step for implementing either NAGS-II or MOPSO is to establish a function that generates random solutions. To generate an initial solution in this research, sequences of jobs and machines are randomly placed next to each other.

4.2.3. Generating neighbor solutions and offspring

Depending on the type of problem that can be discrete, binary, or continuous, there are several operators to generate a random neighbor solution or an offspring. Therefore, considering that the proposed problem is a discrete permutation one, the operators of these two types of problems are used to generate the neighbor solution. The following are the operators applied in this study. It should be noted that these operators are separately employed for both parts of the chromosome.

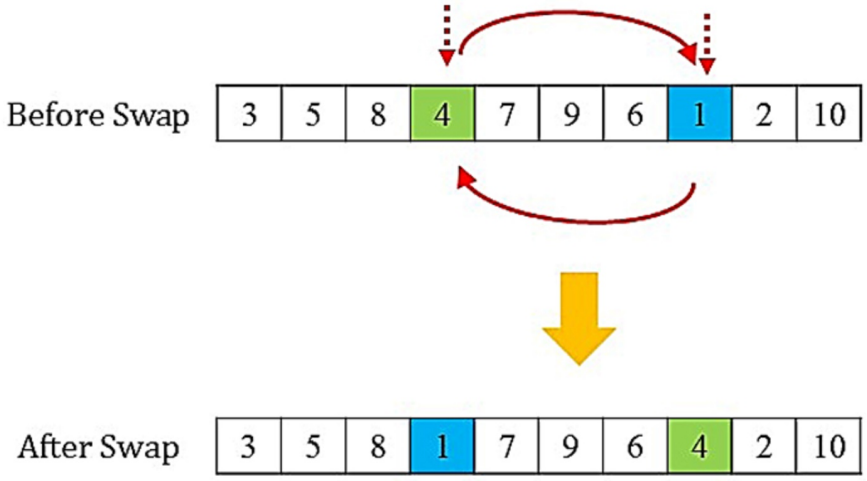

Figure 4: The swap operator

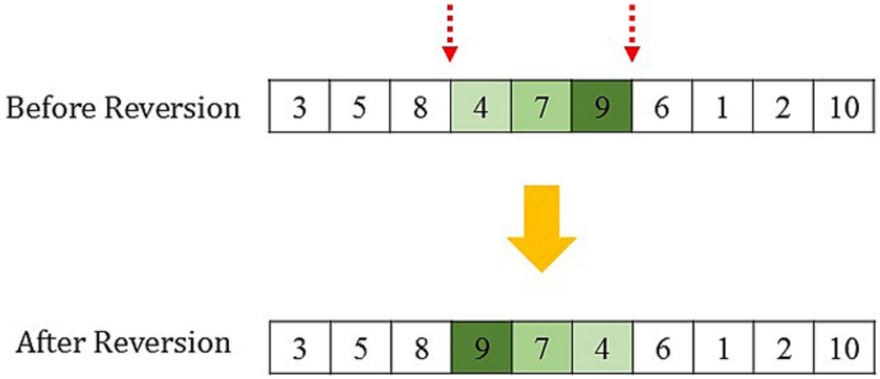

Figure 5: The reversion operator

4.2.3.1.Mutation operators for the first part

Swap

The Mutation operator corresponds to generate a new solution that is from its different from parent. The main purpose of mutation is to increase the diversity of the population. The mutation operator is working in which first, two columns are randomly selected from a two-dimensional chromosome and then replaced with each other. Fig.4 shows how this operator performs.

Reversion

For reversion, at first, two genes are selected and then the genes between them are reversed. Fig.5 displays how the reversion operator performs.

Insertion

To increase the diversification of the algorithms, first, two genes are randomly selected and then the first selected gene is placed next to the second selected gene. Fig.6 shows how the insertion operator performs.

4.2.3.2.Mutation operator for the second part

The mutation operator for the second part of the chromosome that is discrete is performed as follows: first, a gene (or genes) is randomly selected from a chromosome, and then a number is selected between the pre-determined upper and lower limit, which is not equal to the existing number. The selected number is replaced by the original one. For example, in Fig.7 which is an example of a

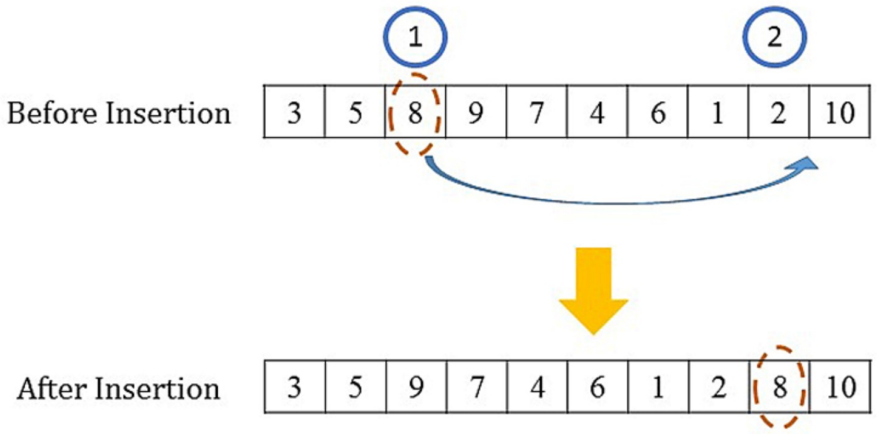

Figure 6: The insertion operator

Figure 7: The mutation operator for the second part of the chromosome

problem with 3 machines, each gene must take an integer from the range [20, 19].

4.2.3.3.Crossover operators for the first part

Single crossover for permutation problems

In permutation problems, it is not possible to employ a single crossover that is used in binary, integers, and real numbers problems. To deal with this issue, first, the crossover operator is applied, then, by holding the second part of the offspring chromosome constant, the duplicate genes (numbers) in the first part are removed and the numbers that have not been used in the chromosome are put in the empty genes (Fig.8).

Double crossover for permutation problems

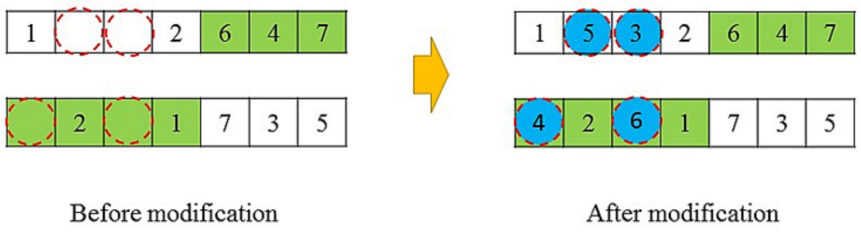

Figure 8: A representation of the single crossover operator in permutation problems

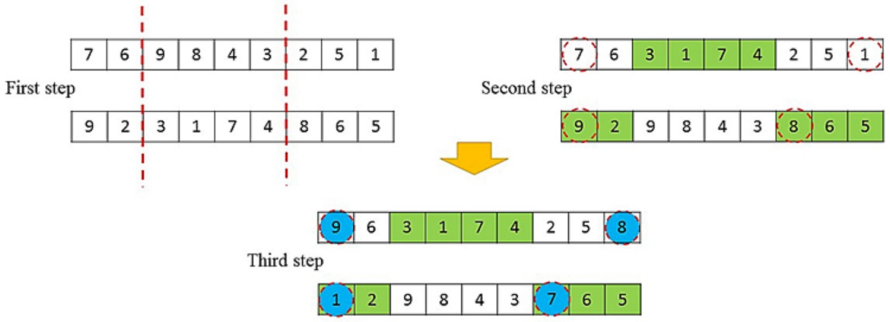

Figure 9: A representation of the crossover operator in permutation problems

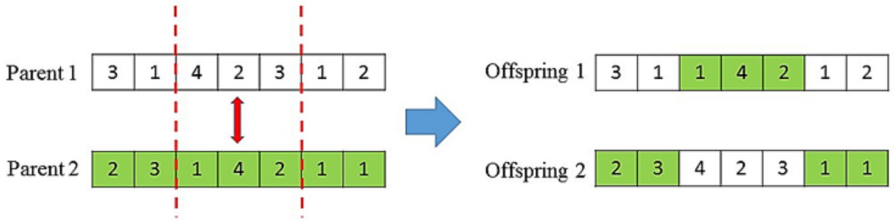

Figure 10: A representation of the crossover operator in discrete problems

For a double crossover, like a single crossover, after selecting two points from the parent chromosomes and transferring the genes between those two points to the offspring chromosomes, the genes on either side of the offspring chromosomes that are repeated twice are found. By replacing the duplicate genes with the ones not previously used in the offspring chromosomes, the double crossover is completed (Fig.9).

4.2.3.4.Crossover operators for the second part

The crossover operator in discrete problems acts like binary problems and all four types of crossover operators in binary problems can be used here as well (Fig.10).

4.3. Multi-objective particle swarm optimization (MOPSO)

Inspired by the choreography of a bird flock, particle swarm optimization (PSO) was first introduced by Kennedy and Eberhart [8]. The method is considered as a distributed behavioral algorithm that performs a multidimensional search. It employs the concept of population and a measure of performance like the fitness value applied by evolutionary algorithms [28]. An important difference between PSO and traditional evolutionary algorithms is that to accelerate convergence, this method introduces the use of flying potential solutions through hyperspace [9]. Another significant difference is that PSO permits individuals to benefit from their past experiences, while a traditional evolutionary algorithm, normally considers the current population as the only memory used by the individuals [40]. PSO has the capability to apply for multi-objective problems, in which the objective function comparison takes Pareto supremacy into account when moving the PSO particles and non-dominated solutions are stored so as to approximate the Pareto front [7]. Because of the high speed of convergence, PSO appears particularly suitable for multi-objective optimizations [16]. The particle swarm algorithm was originally designed to solve continuous problems. To use the algorithm

2.54	1.79	2.33	1.64	3.28	2.31

Figure 11: A randomly generated solution vector

for discrete problems, especially permutation problems, changes must be made in the type of solution representation. The following method which has been employed by some researchers in recent years for flexible and hybrid flow shop problems (Kurz and Askin [23], and Zandieh et al. [38]) is applied in this study to generate a solution vector. A real number is assigned to each job so that the machine number to which the job is assigned is represented by integer part and sorting the jobs assigned to each machine is performed using the fractional part. For example, there are 6 jobs and 3 machines in a stage, so a vector is randomly generated to arrange jobs in this stage. Fig.11 displays that the 2nd and 4th jobs are assigned to machine number 1, the 1st, 3rd, and 6th jobs are assigned to machine number 2 and the 5th one is assigned to machine number 3 [13]. It is important to note that this vector is used in the first stage to assign jobs to machines and their sequence. For subsequent stages, it is only used to assign jobs to machines, and their sequence depends on the completion time of each job in the previous stage [25].

5. Computational experiments

In this section, some numerical experiments are conducted to evaluate the efficiency of the proposed algorithms and to compare their performance with each other for all test problems and with the results obtained from ε constraint method for small size problems in terms of solution quality and computational times. For this purpose, the parameters of both algorithms are tuned using the Taguchi method. Then, a number of test problems are generated and the computational results are presented subsequently. Afterwards, a one-way ANOVA test is carried out to compare the results obtained by the proposed algorithms based on the considered metrics, and eventually, sensitivity analysis is performed.

5.1. Parameters tuning

Parameters tuning plays an important role to keep a metaheuristic efficient. Hence, finding the best values for the control parameters of the algorithm is the main step in implementing a metaheuristic. Therefore, a series of calibration tests are usually performed to find the optimal combination of different values of the algorithm controller parameters. Since the time and cost of the test grow exponentially as the number of test levels and the number of parameters increase, to determine the best values for parameters, a design of experiments based on Taguchi's method is employed [30]. Control parameters for NSGA-II is population size (N_{pop}), crossover probability (P_c), mutation probability (P_m) and maximum number of iterations ($\max_{it}$) and for MOPSO are number of particles (N_{rep}), maximum number of iterations (max_{it}), learning factors (C_1, C_2) and inertia weight factor (w). The quality of a heuristic optimization algorithm and its computational time is strongly dependent on control parameter values. For instance, if the initial population for the NSGA-II is assumed to be large, it results in a larger solution space which may take more time to achieve a solution. On the other hand, if it is assumed small, the algorithm may stock in a local optimal. Several experiments are carried out and the effective range of each of the control parameters of the algorithms is obtained accordingly. Afterwards, to study the interactions between

Table 1: Levels of the control parameters

Algorithm	**Parameter**	**Levels**
NSGA-II	N_{pop}	40, 50, 60
	$\max_{it}$	75, 100, 125
	P_e	0.5, 0.6, 0.7
	P_m	0.2, 0.3, 0.4
MOPSO	N_{rep}	30, 40, 50
	C_1	0.5, 1, 1.5
	C_2	0.5, 1, 1.5
	$\max_{it}$	100, 120, 150
	w	0.97, 0.98, 0.99

Table 2: Ranking of the parameters of NSGA-II according to means

Level \ Parameter	N_{pop}	P_c	P_m	$\max_{it}$
1	534.3	536.1	536.3	534.1
2	535.4	535.6	535.4	535.7
3	538.2	536.2	536.2	538.0
Delta	3.9	0.6	0.9	3.9
Rank	2	4	3	1

these parameters and in order to find the best combination of them, a series of experiments applying the Taguchi method is performed. Each of the parameters is tested at 3 levels (Table 1). Note that, the proposed problem is bi-objective and each algorithm obtains several solutions that do not dominate each other after one run, and these solutions are evaluated by several metrics. However, since the Taguchi experiment deals with only one output metric, to use the Taguchi method all the solutions must be turned into one. To do so, Mean Ideal Distance (MID) and computational time are employed as the output metrics in the current study. The experiments required investigating the various combinations of the control parameters and the corresponding results are presented in Tables 2 and Fig.12 and 13. MINITAB 19 is used to analyze the data.

In order to determine the degree of importance of each parameter, response rows are ranked according to the metrics of the mean and standard deviation of the parameters examined. The average response values for each row of each parameter for NSGA-II are listed in Tables 2 and 3. The tables show the ranking of the parameters, in both of which, the prioritization of the parameters is similar and is equal to the maximum number of iterations (max_{it}), population size (N_{pop}), mutation probability (P_m) and crossover probability (P_c), respectively.

To find the best levels of each parameter for NSGA-II, the interactions of them are analyzed. According to the results shown in Fig.12 and 13, population size N_{pop} on its first level with a value of 40, crossover probability (P_c) on its second level with a value of 0.6, mutation probability (P_m) on its second level with a value of 0.3, and the maximum number of iterations (max_{it}) on its first level with a value of 75 minimize the mean and maximize the signal-to-noise (S/N) ratio ($S/Nratio = -10 \times \log_{10}(\frac{1}{n}\sum_{i=1}^{n} y_i^2)$ where y_i is the normalized deviation of the values of objective function from the least obtained value for each test condition (i) [29]). Therefore, the best values of the control parameters of the NSGA-II are obtained.

Table 3: Ranking of the parameters of NSGA-II according to standard deviations

Level \ Parameter	N_{pop}	P_c	P_m	$\max_{it}$
1	728.0	725.4	725.2	728.2
2	726.4	726.2	726.4	726.0
3	722.5	725.3	725.2	722.7
Delta	5.5	0.9	1.2	5.5
Rank	2	4	3	1

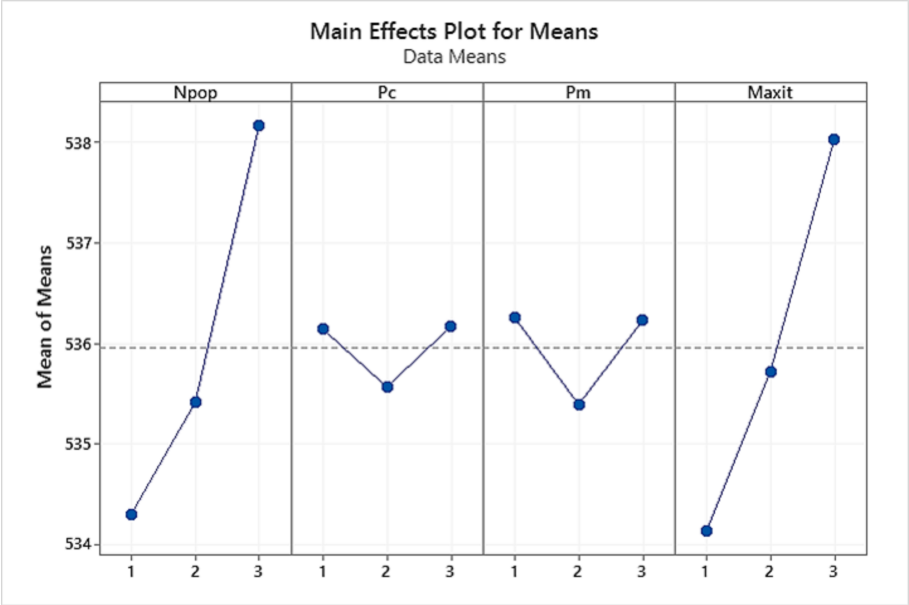

Figure 12: The results of the best means for NSGA-II parameters

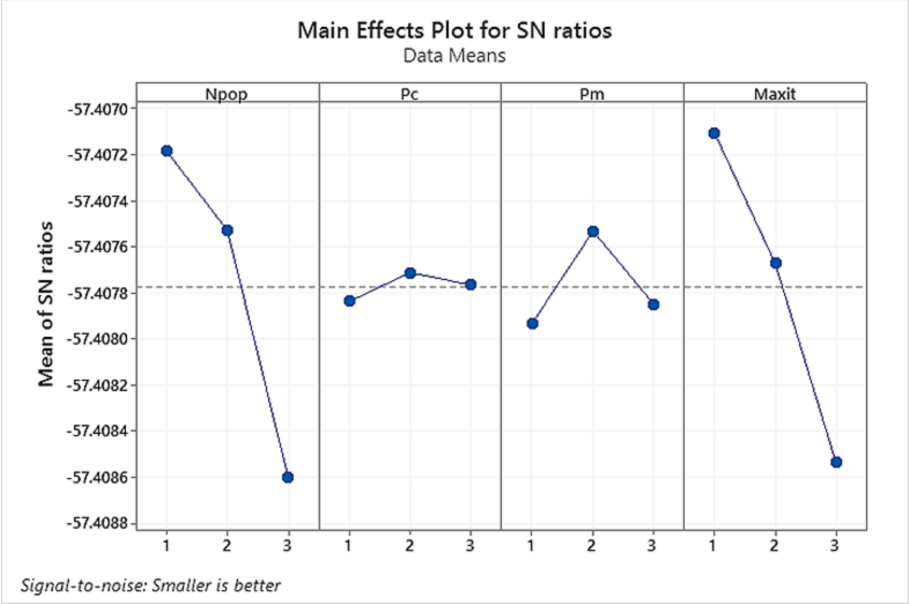

Figure 13: The results of the best S/N ratios for NSGA-II parameters

Table 4: Ranking of the parameters of MOPSO according to means

Level \ Parameter	C_1	C_2	w	$\max_{it}$	N_{rep}
1	532.5	532.4	532.4	531.0	530.3
2	532.2	532.2	532.4	532.3	532.4
3	532.2	532.2	532.4	532.3	532.4
Delta	0.3	0.3	0.2	2.6	3.9
Rank	3	4	5	2	1

Table 5: Ranking of the parameters of MOPSO according to standard deviations

Level \ Parameter	C_1	C_2	w	$\max_{it}$	N_{rep}
1	730.5	730.6	730.7	732.6	733.6
2	731.0	731.0	730.7	730.9	730.6
3	730.9	730.9	731.0	728.9	728.1
Delta	0.5	0.4	0.3	3.7	5.5
Rank	3	4	5	2	1

In order to design multi-factor experiments for the control parameters of MOPSO, the same method used for NSGA-II is applied. For MOPSO, 5 parameters and 3 levels are considered. The proposed combinations for employing the Taguchi method and their results are presented in Tables 4 and 5. These tables show the ranking of the parameters, in both of which, the priority of the parameters is similar and are as follows: number of particles (N_{rep}), the maximum number of iterations (max_{it}), learning factors (C_1) and (C_2) respectively and inertia weight factor (w).

According to Fig.14 and 15, the best combination of the parameters of the proposed algorithm is as the following: max_{it} on its first level with a value of 100, N_{rep} on its first level with a value of 30, C_1 and C_2 on their second level with a value of 1, and w in its third level with a value of 0.99 minimize the mean and maximize the S/N ratio.

5.2. Algorithms evaluation

In this section, the performance of the two proposed algorithms is compared. The values of the parameters according to Table 6 are randomly generated using uniform distributions and used as input data. To evaluate the quality and diversification of multi-objective metaheuristic algorithms, due to the solutions that do not have priority over each other, there are several metrics that differ from the comparison metrics for single-objective metaheuristic algorithms. The metrics considered in this research include the number of Pareto solutions (NPS), mean ideal distance (MID), diversification metric (DM), the spread of non-dominance solution (SNS), and computational time (Time). To make the comparisons, 15 test problems in two scales of small and large size are considered. The results obtained from solving the test problems by the proposed algorithms are presented in Tables 7 and 8. The proposed algorithms are coded using MATLAB R2016a and run on a computer Intel (R) Core i7 2 Duo CPU, 4.4 GHz CPU with 8 GB RAM.

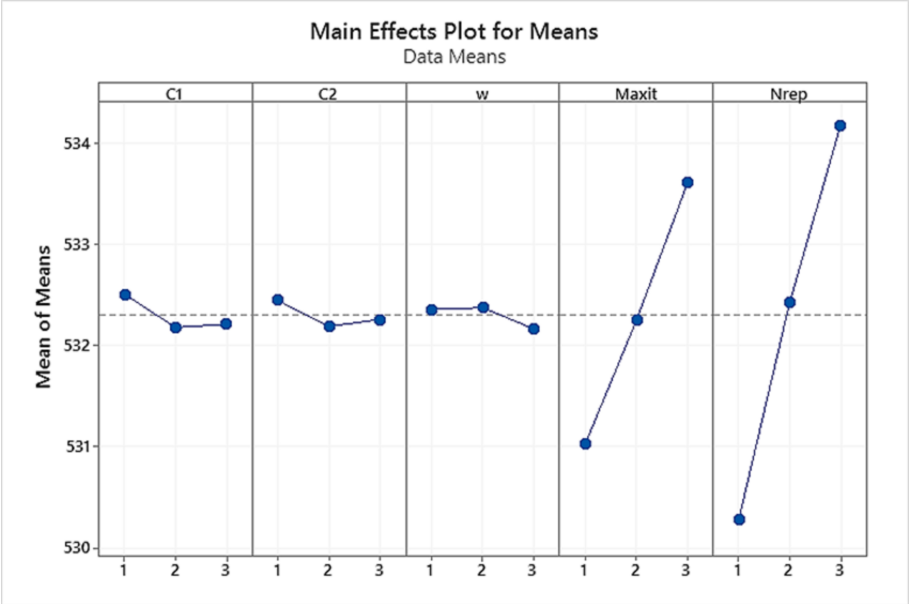

Figure 14: Results for the best means of MOPSO parameters

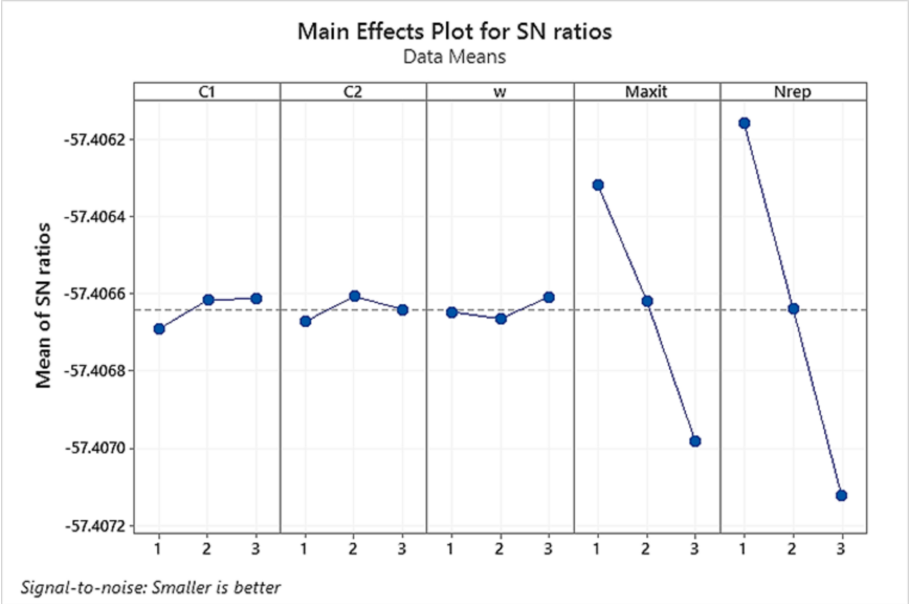

Figure 15: Results of the best S/N ratios for MOPSO parameters

Table 6: The input data of the test problems

Capacity of intermediate buffer	Uniform [10, 25]
Process times (Sec)	Uniform [50, 100]
Transportation times (Sec)	Uniform [10, 50]
The failure rate of transporters (Sec)	Uniform [250, 450]
The repair rate of transporters (Sec)	Uniform [10, 100]
The minimum of availability	0.95

5.2.1. Small size test problems

In addition to the proposed algorithms coded in MATLAB, for small-size test problems, applying the ε constraint method, the proposed mathematical model is coded and run using the exact solver CPLEX. A comparison of the results obtained from all three methods is listed in Table 7.

As Table 7 shows, the average number of Pareto solutions (NPS) obtained for the ε constraint method is more. Also, the MID metric is on average lower than the other two algorithms for the ε constraint method. For DM and SNS metrics, the proposed MOPSO performs relatively better. However, for the SNS metric, the results of the ε constraint method are very close to the ones obtained from MOSPS and the least computational time (Time) belongs to the ε constraint method.

5.2.2. Large size test problems

Since the ε constraint method cannot achieve any solution for large size problems in a period of 6000 seconds, the results of the proposed metaheuristic algorithms are compared with each other in Table 8.

According to Table 8, for one of the metrics, that is NPS, NSGA-II performs relatively better and for the other four metrics, including MID, DM, SNS, and Time, MOPSO relatively outperforms. Overall, by considering the different metrics represented in Tables 7 and 8 for both small and large size problems, it can be concluded that the proposed MOPSO performs relatively better than the presented NSGA-II.

5.2.3. Statistical test

In this section, two one-way ANOVA tests for small and large size problems are performed to compare the results of the proposed methods in terms of the metrics applied. The applied ANOVA analysis tests the difference between the means of the results achieved by the proposed methods in order to find out whether it is statistically significant or not. The procedure also displays descriptive statistics for each method. The one-way ANOVA is carried out for all test problems using SPSS 16.0 as follows:

$$\begin{cases} H_0 : \mu_{\varepsilon constraint} = \mu_{NSGA-II} = \mu_{MOPSO} \\ H_1 : Means\ are\ not\ all\ equal \end{cases}$$

where the confidence interval for the mean difference between the values is considered 95%. The results are displayed in Tables 9101112. Note that, for large size problems H_0 is $\mu_{NSGA-II} = \mu_{MOPSO}$.

5.3. Sensitivity analysis

In this section, sensitivity analysis is carried out to investigate the sensitivity of the objective functions to the number of jobs and their transportation time at each stage. Table 13 shows the sensitivity of the objective functions because of changing the values of a number of jobs for a problem with the size of two stages, three scenarios, and two transporters.

As the results of the sensitivity analysis demonstrate, changes in the number of jobs have a direct impact on both objective functions, so that in addition to increasing the maximum completion time, the unavailability of machines increases, which can be due to the more operation of the machines. Table 14 lists the sensitivity of the objective functions because of changing the values of transportation times for a problem with the size of four jobs, two stages, three scenarios, and two transporters.

According to Table 14, the second objective function does not change much with respect to changes in transportation time. The effect of transportation time on the maximum completion time is much greater than the percentage increase in transportation time, which indicates the high sensitivity of the objective function of this parameter.

Table 7: Computational results of the small size problems

		NPS			MID			DM			SNS			Time(s)		
J	*S*	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO
4	2	5	4	5	4233	4265	4474	458	245	661	158	107	288	4	125	113
	3	8	4	2	3765	3892	3862	325	323	220	185	158	156	25	165	137
	4	6	6	4	4528	5105	4959	725	535	798	310	257	329	48	176	163
6	2	7	5	3	2243	2648	2638	325	220	133	246	89	124	121	289	231
	3	10	11	11	3364	3566	3883	1401	820	1378	568	286	495	237	365	328
	4	8	9	4	4865	5199	5169	324	1137	563	168	419	272	578	437	418

Table 8: Computational results of the large size problems,(∗NA: Not Available (out of CPU time))

		NPS			MID			DM			SNS			Time(s)		
J	*S*	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO	ε constraint	NSGA-II	MOPSO
20	5	NA*	8	6	NA	5105	4872	NA	667	1397	NA	704	764	NA	865	812
	7	NA	5	4	NA	4595	3948	NA	1160	2863	NA	961	1263	NA	849	798
	9	NA	6	8	NA	5086	4162	NA	989	817	NA	1324	1246	NA	968	945
50	5	NA	7	7	NA	4837	4213	NA	782	2861	NA	1536	1692	NA	1561	1634
	7	NA	9	8	NA	6548	5896	NA	974	2172	NA	974	1286	NA	1739	1705
	9	NA	4	5	NA	7097	6693	NA	2336	2769	NA	1038	1126	NA	1863	1812
100	5	NA	12	10	NA	6523	6256	NA	496	1052	NA	729	832	NA	3284	3148
	7	NA	9	6	NA	7698	6398	NA	262	812	NA	426	954	NA	3525	3279
	9	NA	6	8	NA	7132	6942	NA	1564	923	NA	1089	862	NA	3329	3267

Table 9: Descriptive statistics for the small size problems

Metric/Method	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
NPS								
Epsilon Constraint	6	7.3333	1.75119	.71492	5.4956	9.1711	5.00	10.00
NSGAII	6	6.5000	2.88097	1.17615	3.4766	9.5234	4.00	11.00
MOPSO	6	4.8333	3.18852	1.30171	1.4872	8.1795	2.00	11.00
Total	18	6.2222	2.73443	.64451	4.8624	7.5820	2.00	11.00
MID								
Epsilon Constraint	6	3.8330E3	944.53523	3.85605E2	2841.7711	4824.2289	2243.00	4865.00
NSGAII	6	4.1125E3	967.55749	3.95004E2	3097.1107	5127.8893	2648.00	5199.00
MOPSO	6	4.1642E3	920.54232	3.75810E2	3198.1167	5130.2166	2638.00	5169.00
Total	18	4.0366E3	899.65629	2.12051E2	3589.1670	4483.9441	2243.00	5199.00
DM								
Epsilon Constraint	6	5.9300E2	425.32858	1.73640E2	146.6450	1039.3550	324.00	1401.00
NSGAII	6	5.4667E2	366.35538	1.49564E2	162.2003	931.1331	220.00	1137.00
MOPSO	6	6.2550E2	449.03574	1.83318E2	154.2659	1096.7341	133.00	1378.00
Total	18	5.8839E2	391.27276	92.22387	393.8135	782.9643	133.00	1401.00
SNS								
Epsilon Constraint	6	2.7250E2	155.63772	63.53883	109.1682	435.8318	158.00	568.00
NSGAII	6	2.1933E2	125.77546	51.34762	87.3401	351.3266	89.00	419.00
MOPSO	6	2.7733E2	132.98371	54.29037	137.7755	416.8912	124.00	495.00
Total	18	2.5639E2	133.07766	31.36671	190.2109	322.5669	89.00	568.00
Time								
Epsilon Constraint	6	1.6883E2	217.62851	88.84647	-59.5538	397.2204	4.00	578.00
NSGAII	6	2.5950E2	124.49859	50.82634	128.8467	390.1533	125.00	437.00
MOPSO	6	2.3167E2	119.79928	48.90785	105.9450	357.3883	113.00	418.00
Total	18	2.2000E2	155.66858	36.69144	142.5878	297.4122	4.00	578.00

Table 10: One-way ANOVA for the small size problems

Metric	Type	Sum of Squares	df	Mean Square	F	Sig.
NPS	Between Groups	19.444	2	9.722	1.354	.288
	Within Groups	107.667	15	7.178		
	Total	127.111	17			
MID	Between Groups	380922.111	2	190461.056	.214	.810
	Within Groups	1.338E7	15	891904.156		
	Total	1.376E7	17			
DM	Between Groups	18835.444	2	9417.722	.055	.947
	Within Groups	2583768.833	15	172251.256		
	Total	2602604.278	17			
SNS	Between Groups	12428.111	2	6214.056	.323	.729
	Within Groups	288636.167	15	19242.411		
	Total	301064.278	17			
Time(s)	Between Groups	25886.333	2	12943.167	.503	.615
	Within Groups	386069.667	15	25737.978		
	Total	411956.000	17			

Table 11: Descriptive statistics for the large size problems

Metric/Method	N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Minimum	Maximum
					Lower Bound	Upper Bound		
NPS								
NSGAII	9	7.3333	2.44949	.81650	5.4505	9.2162	4.00	12.00
MOPSO	9	6.8889	1.83333	.61111	5.4797	8.2981	4.00	10.00
Total	18	7.1111	2.11128	.49763	6.0612	8.1610	4.00	12.00
MID								
NSGAII	9	6.0690E3	1165.22723	3.88409E2	5173.3271	6964.6729	4595.00	7698.00
MOPSO	9	5.4867E3	1187.72208	3.95907E2	4573.7027	6399.6307	3948.00	6942.00
Total	18	5.7778E3	1180.06756	2.78145E2	5190.9995	6364.6671	3948.00	7698.00
DM								
NSGAII	9	1.0256E3	620.54253	2.06848E2	548.5643	1502.5468	262.00	2336.00
MOPSO	9	1.7407E3	917.52561	3.05842E2	1035.3940	2445.9393	812.00	2863.00
Total	18	1.3831E3	844.24132	1.98990E2	963.2798	1802.9424	262.00	2863.00
SNS								
NSGAII	9	9.7567E2	333.36129	1.11120E2	719.4225	1231.9108	426.00	1536.00
MOPSO	9	1.1139E3	294.65337	98.21779	887.3983	1340.3795	764.00	1692.00
Total	18	1.0448E3	313.38541	73.86565	888.9349	1200.6207	426.00	1692.00
Time								
NSGAII	9	1.9981E3	1101.03082	3.67010E2	1151.7839	2844.4383	849.00	3525.00
MOPSO	9	1.9333E3	1045.49821	3.48499E2	1129.6923	2736.9744	798.00	3279.00
Total	18	1.9657E3	1042.10098	2.45626E2	1447.4976	2483.9468	798.00	3525.00

Table 12: One-way ANOVA for the large size problems

Metric	Type	Sum of Squares	df	Mean Square	F	Sig.
NPS	Between Groups	.889	1	.889	.190	.669
	Within Groups	74.889	16	4.681		
	Total	75.778	17			
MID	Between Groups	1526004.500	1	1526004.500	1.102	.309
	Within Groups	2.215E7	16	1384219.125		
	Total	2.367E7	17			
DM	Between Groups	2301227.556	1	2301227.556	3.751	.071
	Within Groups	9815410.222	16	613463.139		
	Total	1.212E7	17			
SNS	Between Groups	85974.222	1	85974.222	.869	.365
	Within Groups	1583602.889	16	98975.181		
	Total	1669577.111	17			
Time(s)	Between Groups	18882.722	1	18882.722	.016	.900
	Within Groups	1.844E7	16	1152667.681		
	Total	1.846E7	17			

Table 13: Sensitivity analysis of the objective functions by changing a number of jobs

Stage	Number of jobs	Value of the 1^{st} objective function	Value of the 2^{nd} objective function	Computational time
1	3	269	0.0497	2
2	4	301	0.0613	4
3	5	494	0.0709	35
4	6	584	0.0791	119

Table 14: Sensitivity analysis of the objective functions by changing transportation times

Stage	Percentage of changes	Value of the 1^{st} objective function	Value of the 2^{nd} objective function
1	- 30%	512	0.0613
2	- 20%	604	0.0627
3	- 10%	945	0.0627
4	0 %	1285	0.0630
5	+ 10%	1648	0.0633
6	+ 20%	1983	0.0645
7	+ 30%	2524	0.0648

6. Conclusions

In this research, a bi-objective flexible flow shop problem with transportation times, transporter preventive maintenance planning, and limited intermediate buffers were investigated. The objective functions are to minimize the total completion time and the unavailability of the system at the same time. Also, a new mathematical model was proposed and two metaheuristic algorithms, including a non-dominated sorting genetic algorithm II (NSGA-II) and a multi-objective particle swarm optimization (MOPSO), was employed to solve the problem in a reasonable computational time. Numerical experiments were carried out to investigate the performance of the proposed NSGA-II and MOPSO. In order to evaluate the quality and diversification of the algorithms and compare the results of the proposed methods, the number of Pareto solutions (NPS), mean ideal distance (MID), diversification metric (DM), the spread of non-dominance solution (SNS) and computational time (Time) were considered as comparison metrics. In addition, one-way ANOVA tests for small and large size problems were performed. Computational results confirmed that MOPSO relatively outperforms NSGA-II in the most of test problems. Sensitivity analysis indicated that changes in the number of jobs have a direct impact on both objective functions and also, the effect of transportation time on the maximum completion time is much greater than the percentage increase in transportation time. For future research, considering other multi-objective metaheuristic algorithms such as multi-objective electromagnetism algorithm (MOEA), strength Pareto evolutionary algorithm2 (SPEA2), and multi-objective simulated annealing (MOSA). Another opportunity for the present research is investigating other objective functions like minimizing the weighted mean flow time or total penalties of tardiness or earliness is commended. In addition, the sequence-dependent setup times could be supposed in the model for another effort.

References

[1] S.S. Amelian, S.M. Sajadi, M. Navabakhsh and M. Esmaelian, *Multi-objective optimization for stochastic failure-prone job shop scheduling problem via a hybrid of NSGA-II and simulation method*, Expert Syst. (2019) e12455.

[2] H. Amin-Tahmasbi and R. Tavakkoli-Moghaddam, *Solving a bi-objective flow shop scheduling problem by a Multi-objective Immune System and comparing with $SPEA2+$ and SPGA*, Adv. Engin. Software 42(10) (2011) 772–779.
[3] S.F. Attar, M. Mohammadi, R. Tavakkoli-Moghaddam and S. Yaghoubi, *Solving a new multi-objective hybrid flexible flow shop problem with limited waiting times and machine-sequence-dependent set-up time constraints*, Int. J. Comput. Integ. Manufact. 27(5) (2014) 450–469.
[4] A. Berrichi, L. Amodeo, F. Yalaoui, E. Châtelet and M. Mezghiche, *Bi-objective optimization algorithms for joint production and maintenance scheduling: application to the parallel machine problem*, J. Intell. Manufact. 20(4) (2009) 389.
[5] A. Berrichi, F. Yalaoui, L. Amodeo and M. Mezghiche, *Bi-objective ant colony optimization approach to optimize production and maintenance schedules*, Comput. Oper. Res. 37(9) (2010) 1584–1596.
[6] S.H. Choi and K. Wang, *Flexible flow shop scheduling with stochastic processing times: A decomposition-based approach*, Comput. Indust. Engin. 63(2) (2012) 362–373.
[7] C.C. Coello and M.S. Lechuga, *MOPSO: A proposal for multiple objective particle swarm optimization*, Proc. 2002 Cong. Evolut. Comput. $CEC'02$ (Cat. No. 02TH8600) 2 (2002) 1051–1056.
[8] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, *A fast and elitist multi-objective genetic algorithm: $NSGA-\prod$*, IEEE Trans. Evolut. Comput. 6(2) (2002).
[9] R. Eberhart and J. Kennedy, *Particle swarm optimization*, Proc. IEEE Int. Conf. Neural Networks 4 (1995) 1942–1948.
[10] E. Gheisariha, M. Tavana, F. Jolai and M. Rabiee, *A simulation-optimization model for solving flexible flow shop scheduling problems with rework and transportation*, Math. Comput. Simul. 180 (2021) 152–177.
[11] A. Ghodratnama, F. Jolai and R. Tavakkoli-Moghaddam, *Solving a new multi-objective multi-route flexible flow line problem by multi-objective particle swarm optimization and NSGA-II*, J. Manufact. Syst. 36 (2015) 189–202.
[12] A. Hasani, S.M.H. Hosseini, *A bi-objective flexible flow shop scheduling problem with machine-dependent processing stages: Trade-off between production costs and energy consumption*, Appl. Math. Comput.386 (2020) 125533.
[13] M. Hekmatfar, S.F. Ghomi and B. Karimi, *Two-stage reentrant hybrid flow shop with setup times and the criterion of minimizing the makespan*, Appl. Soft Comput. 11(8) (2011) 4530–4539.
[14] F. Jolai, H. Asefi, M. Rabiee and P. Ramezani, *Bi-objective simulated annealing approaches for no-wait two-stage flexible flow shop scheduling problem*, Scientia Iranica 20(3) (2013) 861–872.
[15] M. Khalili, R. Tavakkoli-Moghaddam, *A multi-objective electromagnetism algorithm for a bi-objective flow shop scheduling problem*, J. Manufact. Syst. 31(2) (2012) 232–239.
[16] M.E. Kurz and R.G. Askin, *Scheduling flexible flow lines with sequence-dependent setup times*, European J. Oper. Res. 159(1) (2004) 66–82.
[17] K.S. Moghaddam, *Multi-objective preventive maintenance and replacement scheduling in a manufacturing system using goal programming*, Int. J. Prod. Econ. 146(2) (2013) 704–716.
[18] M. Mohammadi, J.Y. Dantan, A. Siadat and R. Tavakkoli-Moghaddam, *A bi-objective robust inspection planning model in a multi-stage serial production system*, Int. J. Prod. Econ. 56(4) (2018) 1432–1457.
[19] B. Naderi, R. Tavakkoli-Moghaddam and M. Khalili, *Electromagnetism-like mechanism and simulated annealing algorithms for flow shop scheduling problems minimizing the total weighted tardiness and makespan*, Knowledge-Based Syst. 23(2) (2010) 77–85.
[20] B. Naderi, M. Zandieh and V. Roshanaei, *Scheduling hybrid flow shops with sequence-dependent setup times to minimize makespan and maximum tardiness*, Int. J. Adv. Manufact. Tech. 41(11–12) (2009) 1186–1198.
[21] M. Naderi-Beni, E. Ghobadian, S. Ebrahimnejad and R. Tavakkoli-Moghaddam, *The fuzzy bi-objective formulation for a parallel machine scheduling problem with machine eligibility restrictions and sequence-dependent setup times*, Int. J. Prod. Res. 52(19) (2014) 5799–5822.
[22] S. Noori-Darvish and R. Tavakkoli-Moghaddam, *Minimizing the total tardiness and makespan in an open shop scheduling problem with sequence-dependent setup times*, J. Indust. Engin. Int. 8(1) (2012) 25.
[23] K.E. Parsopoulos and M.N. Vrahatis, *Particle swarm optimization method in multiobjective problems*, Proc. 2002 ACM Symp. Appl. Comput. (2002) 603–607.
[24] S. Raissi, R. Rooeinfar and V.R. Ghezavati, *Three hybrid metaheuristic algorithms for stochastic flexible flow shop scheduling problem with preventive maintenance and budget constraint*, J. Opt. Indust. Engin. (2019) 131–147.
[25] M. Reyes-Sierra and A. Carlos, *Multi-objective particle swarm optimizers: A survey of the state of the art*, Int. J. Comput. Intel. Res. 2(3) (2006) 287–308.
[26] R. Rooeinfar, R. Raissi and V.R. Ghezavati, *Stochastic flexible flow shop scheduling problem with limited buffers and fixed interval preventive maintenance: a hybrid approach of simulation and metaheuristic algorithms*, Simulation (2019) 509–528.
[27] B. Shahidi-Zadeh, R. Tavakkoli-Moghaddam, A. Taheri-Moghadam and I. Rastgar, *Solving a bi-objective unrelated*

parallel batch processing machines scheduling problem: A comparison study, Comput. Oper. Res. 88 (2017) 71–90.
[28] N. Sriniva and K. Deb, *Multi-objective optimization using non-dominated sorting in genetic algorithms*, J. Evolut. Comput. 2(3) (1994) 221–248.
[29] J.U. Sun, *A Taguchi approach to parameter setting in a genetic algorithm for a general job shop scheduling problem*, IEMS 6(2) (2007) 119–124.
[30] G. Taguchi, *Introduction to Quality Engineering: Designing Quality Into Products and Processes*, The Organization, the University of Michigan, 1986.
[31] R. Tavakkoli-Moghaddam, M. Azarkish and A. Sadeghnejad-Barkousaraie, *A new hybrid multi-objective Pareto archive PSO algorithm for a bi-objective job shop scheduling problem*, Expert Syst. Appl. 38(9) (2011) 10812–10821.
[32] R. Tavakkoli-Moghaddam, M. Azarkish and A. Sadeghnejad-Barkousaraie, *Solving a multi-objective job shop scheduling problem with sequence-dependent setup times by a Pareto archive PSO combined with genetic operators and VNS*, Int. J. Adv. Manufact. Tech. 53(5–8) (2011) 733–750.
[33] A. Villemeur, *Assessment, Hardware, Software and Human Factors*, Volume 2 of Reliability, availability, Maintainability and Safety Assessment, Wiley, 1992.
[34] S. Wang, *Bi-objective optimization for integrated scheduling of single machines with setup times and preventive maintenance planning*, Int. J. Prod. Res. 51(12) (2013) 3719–3733.
[35] S. Wang and M. Liu, *Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning*, J. Manufact. Syst. 37 (2015) 182–192.
[36] Y. Yusoff, M.S. Ngadiman and A.M. Zain, *Overview of NSGA-II for optimizing machining process parameters*, Procedia Engin. 15 (2011) 3978–3983.
[37] S.S. Zabihzadeh and J. Rezaeian, *Two meta-heuristic algorithms for flexible flow shop scheduling problem with robotic transportation and release time*, Appl. Soft Comput. 40 (2016) 319–330.
[38] M. Zandieh, S.F. Ghomi and S.M. Husseini, *An immune algorithm approach to hybrid flow shops scheduling with sequence-dependent setup times*, Appl. Math. Comput. 180(1) (2006) 111–127.
[39] M. Zandieh, S.M. Sajadi and R. Behnoud, *Integrated production scheduling and maintenance planning in a hybrid flow shop system: a multi-objective approach*, Int. J. Syst. Assur. Engin. Manag.8(2) (2017) 1630–1642.
[40] E. Zitzler and L. Thiele, *Multi-objective Evolutionary algorithms: A comparative case study and the strength Pareto approach*, IEEE Trans. Evolut. Comput. 3(4) (1999).