Iranian Journal of Fuzzy Systems

Volume 16, Number 4, (2019), pp. 21-40

University of Sistan and Baluchestan

A bi-objective model for a scheduling problem of unrelated parallel batch processing machines with fuzzy parameters by two fuzzy multi-objective meta-heuristics

A. Sadati[1], R. Tavakkoli-Moghaddam[2], B. Naderi[3] and M. Mohammadi[4]

[1] *Department of Industrial Engineering, Science and Research Branch, Islamic Azad University,Tehran, Iran*
[2] *School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran and Arts et Mtiers ParisTech, LCFC, Metz, France*
[3,4] *Department of Industrial Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran*

sadati_azam@iaukhomein.ac.ir, tavakoli@ut.ac.ir, bahman.naderi@aut.ac.ir, mohammadi@khu.ac.ir

Abstract

This paper considers a bi-objective model for a scheduling problem of unrelated parallel batch processing machines to minimize the makespan and maximum tardiness, simultaneously. Each job has a specific size and the data corresponding to its ready time, due date and processing time-dependent machine are uncertain and determined by trapezoidal fuzzy numbers. Each machine has a specific capacity, in which the number of jobs assigned to each batch on the machine does not violate the machine capacity. The batch processing time, the batch ready time and the batch due date are presented by the longest processing time, the longest ready time and the shortest due date of the jobs that belong to the batch, respectively. To determine the longest and shortest time, the method suggested by Jimnez et al.[18] is used for ranking the fuzzy numbers. A bi-objective fuzzy mixed-integer linear programming model is proposed and solved by two exact methods (i.e., two-phase fuzzy and ϵ-constraint) for small-sized problems to obtain a set of Pareto solutions. Because the problem belongs to the class NP-hard, two meta-heuristics, namely fuzzy non-dominated sorting genetic algorithm (FNSGA-II) and fuzzy multi-objective discrete teachinglearning-based optimization (FMODTLBO), are proposed. Then, the comparison of results is illustrated to show their performances. Furthermore, a new representation of the solutions is a matrix with two rows and N columns (i.e., jobs) used to assign the jobs to the batches that processed on the machines.

Keywords: Batch processing, unrelated parallel machines scheduling, fuzzy parameters, fuzzy multi-objective meta-heuristics.

1 Introduction

In the recent decades, a scheduling problem of the batch processing machines (BPMs) in many modern manufacturing industries (e.g., chemical, metalworking, printing industry, food and mineral processing, wafer fabrication process and ship scheduling at navigation) has received a considerable attention. These are applied in a burn-in operation for final testing in a semiconductor manufacturing system [33, 39]. Because of the great application potential of the BPMs, this research is motivated by a parallel batch processing machines (PBPMs), in which different machines perform the same function while considering different processing velocity and capacity (i.e., unrelated PBPMs (UPBPMs)). In BPM, each machine can process a number of jobs as a batch without any exceeding machine capacity. In this study, each job has a different size, machine-dependent processing time, ready time and due date. The batch processing time (BPT), the batch ready time (BRT) and the batch due date (BDD) are given by the longest processing time, longest ready time and shortest due date of jobs that belong to the batch, respectively.

Corresponding Author: R. Tavakkoli-Moghaddam
Received: October 2017; Revised: June 2018; Accepted: December 2018.

In order to improve the performance of production systems, we consider both manufacturer concerns (e.g., waiting time and work-in-process inventory) and customer concerns (e.g., assuring on-time receipt). For this purposes, a bi-objective model is considered to be able to minimize the makespan (i.e., maximum complete time, known as Cmax) and maximum tardiness (i.e., Tmax) simultaneously. To make our study more realistic, the fuzzy processing times, fuzzy ready times and fuzzy due dates are also considered. The main motivations that these times are considered as a fuzzy number are as follows [27]:

- The information required for these times may be vague or not precisely measurement.
- The uncertainty inherent in real scheduling environments can be modeled by the fuzzy scheduling algorithms. By modeling these times with a fuzzy number, the system designer can build the exibility into the scheduling algorithm and reach a better solution.
- The quality and quantity of available information may be dampened because imprecision and vagueness as a result of personal bias and subjective opinion.

In this paper, a trapezoid membership function is employed. This choice has been made because this function is easily implemented and felexible in applying to various real-world problems.

According to the above issues, a fuzzy bi-objective UPBPMs model is considered in such a way that the makespan and maximum tardiness are to be minimized.

Wang and Chou [37] addressed a PBPMs scheduling problem to minimize the makespan and formulated the problem in the form of a MIP model. Then, proposed simulated annealing (SA) and genetic algorithm (GA). In order to assign jobs to batches, a multi-stage dynamic programming algorithm is applied. Mehdizadeh et al. [26] proposed a vibration damping optimization (VDO) algorithm to minimize the total weighted completion for a parallel machines (PMs) scheduling problem.

Damodaran et al. [10] considered an identical PBPMs (IPBPMs) scheduling problem with preemption in order to minimize the makespan. They used a greedy randomized and adaptive search procedure (GRASP) method to solve the problem. Damodaran and Velez-Gallego [9] showed the IPBPMs scheduling problem to minimize the makespan and calculated a lower bound for the problem. They proposed SA and two other heuristics for the comparison of a numerical instance and also used a constructive heuristic method [8] for their problem. Cheng et al. [7] considered the IPBPMs scheduling problem to minimize the makespan and suggested an improved ant colony optimization (ACO) method. Ma et al. [25] considered an online IPBPMs scheduling problem to minimize the total weighted completion time by using a $4(1+\epsilon)$-competitive online algorithm. Attar et al. [3] addressed a biogeography-based optimization (BBO) algorithm to solve a UPMs problem in a flexible flow shop environment. Dousthaghi et al. [12] presented a mixed-integer nonlinear programming (MINLP) model and proposed a hybrid PSO for a UPMs scheduling problem in a flexible job shop environment with shelf life. Zhou et al. [39] considered a uniform PBPMs scheduling problem to minimize the makespan and proposed a discrete differential evolution (DDE) algorithm. Li et al. [21] studied a UPBPMs scheduling problem to minimize the makespan and proposed two groups of heuristics based on best fit LPT. Joo and Kim [19] studied a UPBPMs problem with heterogeneous delivery trucks (i.e., heterogeneous trucks with different capacities and travel time) to minimize the makespan. They proposed rule-based meta-heuristics using a single-stage GA framework.

Hulett et al. [15] considered a non-identical PBPMs scheduling to minimize the total weighted tardiness and proposed a particle swarm optimization (PSO) algorithm. Arroyo and Leung [1] studied a UPBPMs scheduling problem with arbitrary jobs size and unequal ready time to minimize the makespan. They proposed several heuristics based on first-fit and best-fit and then presented an MIP and a lower bound to study the quality of their heuristics, and also proposed a meta-heuristic algorithm based on iterated greedy [2] for their problem. Jiang et al. [17] considered a uniform PBPMs problem scheduling with batch transportation and proposed a hybrid discrete PSO-GA algorithm. Tavakkoli-Moghaddam et al. [36] showed a UPMs scheduling problem with precedence relations between jobs and used a GA to minimize the number of tardy jobs and the total completion time. Kashan et al. [20] considered a single BPM scheduling problem to minimize the makespan and maximum tardiness. They represented two different designs of chromosomes in the proposed two GAs. Cheng et al. [6] presented an MIP model for a PBPMs scheduling problem to minimize the makespan and total completion time, in which jobs do not have identical sizes. Xu et al. [38] considered an identical PBPMs scheduling problem to minimize the makespan and maximum tardiness and proposed a multi-objective ACO (MOACO) algorithm to solve their problem. Shahidi-Zadeh et al. [33] studied a UPBPMs scheduling with considering release time and ready time for jobs to minimize the makespan and tardiness/earliness penalties as well as the purchasing cost of machines as a novel objective, simultaneously. They proposed a multi-objective harmony search (MOHS). Shahvari and Logendran [34] addressed a UPBPMs with sequence and machine-dependent batch scheduling to minimize the total weighted completion time and tardiness. They proposed a multi-level tabu search, also they

considered a UPBPMs problem with dual-resources (i.e., machines and operations) to minimize the cost of tardy and early jobs and makespan, simultaneously. Four bi-objective PSO-based search algorithms are proposed [35]. Jia et al. [16] considered IPBPMs with jobs arriving dynamically to minimize makespan and the total electric cost and proposed a Pareto-based ACO algorithm.

Cheng et al. [5] introduced a fuzzy model on a single BPM. Processing time of the batches and adjust the time of the machines between the batches are considered triangular fuzzy numbers. An improved ACO algorithm is proposed to minimize the makespan. Gharehgozli et al. [13] considered a PMs scheduling problem as a mixed-integer goal programming model with fuzzy processing times to minimize two fuzzy objectives. Li et al. [22] studied a single BPM scheduling problem, in which due date and precedence relations are fuzzy based on the satisfaction level about completion times of jobs and precedence between two jobs, respectively. Furthermore, Molla-Alizadeh-Zavardehi et al. [28] considered a fuzzy single BPM scheduling with the fuzzy due date to maximize the total degree of satisfaction and proposed the imperialist competitive algorithm (ICA). Given the above background, the main contributions of this paper are as follows:

- Developing a bi-objective mixed-integer linear programming model with fuzzy parameters.
- Considering the tardiness of the batches.
- Considering the machines with different processing velocity (i.e., different processing time) and capacity, simultaneously.
- Applying two exact methods, namely two-phase fuzzy and ϵ-constraint method, to solve small-sized problems.
- Applying two meta-heuristic algorithms, namely fuzzy non-dominated sorting genetic algorithm (FNSGA-II) and fuzzy multi-objective discrete teaching-learning-based optimization (FMODTLBO) with the capability of solving fuzzy multi-objective problems without requiring them to be defuzzied.

The rest of this paper is organized as follows. The model is formulated with a fuzzy mixed-integer linear programming model in Section 2. Exact methods are described for small-sized problems in Section 3. Section 4 presents details of the solution methodology (i.e., defining the proposed meta-heuristics in details). The numerical and evaluating results are provided in Section 5. The conclusion and future research are drawn in Section 6.

2 Mathematical Model

Our problem is formulated as a bi-objective FMILP model. This model is modified from the models presented by Wang and Chou [37], Arroyo and Leung [1] and Shahidi-Zadeh et al. [33]. It is assumed that N compatible jobs are grouped in several batches on M UPBPMs. The number of batches on the machine k, B_k ($\sum_{k=1}^{M} B_k \leq N$), is not determined until all jobs have been assigned to batches and machines. Each job is defined by its ready time r_j, size s_j, due date d_j and processing time-dependent machine p_{jk}. The data corresponding to ready times, due dates, and processing times are uncertain and determined by trapezoidal fuzzy numbers. Machine k has a capacity L_k and the number of jobs assigned to each batch on machine k should not exceed L_k. Preemption is not allowed (i.e., once processing of a batch on a machine starts, cessation cannot occur and no job is neither added nor removed from the batch until processing of the batch is completed). The processing time of batch l on machine k (i.e., P_{kl}), the ready time of batch l on the machine k (i.e., r_{kl}) and the due date of batch l on the machine k (i.e., d_{kl}) are presented by the longest processing time ($P_{kl} = max_{j \in l} p_{jk}$) longest ready time ($r_{kl} = max_{j \in l} r_j$) and shortest due date ($d_{kl} = min_{j \in l} d_j$) of the jobs that belong to the batch l, respectively. To determine the longest and shortest times, we use the method suggested by Jimnez et al. [18] for ranking fuzzy numbers. Completion time $c_{kl} = P_{kl} + r_{kl}$ and tardiness $t_{kl} = max(0, c_{kl} - min\{d_j \mid j \in l\})$ [38]. Objective functions are the makespan (i.e., last batch that completed, $C_{max} = max_{\forall k,l} c_{kl}$) and maximum tardiness (i.e., the longest tardiness of all the batches, $T_{max} = max_{\forall k,l} t_{kl}$). The model notations are as follows:

Indices:

k : Machines indices ($k = 1, \ldots, M$)

j : Jobs indices ($j = 1, \ldots, N$)

l : Batch indices ($l = 1, \ldots, N$)

Parameters:

N : Number of jobs

M : Number of machines

L_k : Capacity of machine k

$\tilde{r_j}$: Fuzzy ready time of job j

$\tilde{d_j}$: Fuzzy due date of job j

s_j : Size of job j

$\tilde{p_{jk}}$: Fuzzy processing time of job j on machine k

U : A very large positive number

DecisionVariables:

x_{kjl} : Equals 1, if batch l being processed on machine k includes job j; and 0, otherwise

c_{kl} : Completion time of batch l on machine k

P_{kl} : Processing time of batch l on machine k

r_{kl} : Ready time of batch l on machine k

t_{kl} : Tardiness batch l on machine k

T_{max} : Maximum tardiness

C_{max} : Makespan

According to the above-mentioned notations, the proposed problem, which can be denoted $R_m \mid p_{jk}^{\sim}, \tilde{d_j}, \tilde{r_j}, s_j, p-batch|C_{max}, T_{max}$, is formulated as follows:

$$Minf_1(x) = C_{max} \tag{1}$$

$$Minf_2(x) = T_{max} \tag{2}$$

$$s.t. \quad \sum_{k=1}^{M}\sum_{l=1}^{N} x_{kjl} = 1; \quad \forall j \tag{3}$$

$$\sum_{j=1}^{N} s_j.x_{kjl} \leq L_k; \quad \forall k, l \tag{4}$$

$$\sum_{j=1}^{N} x_{kjl} - \sum_{j=1}^{N} x_{kjl-1} \leq 0; \quad \forall l \geq 2, k \tag{5}$$

$$r_{kl} \geq \tilde{r_j}.x_{kjl}; \quad \forall j, k, l \tag{6}$$

$$P_{kl} \geq \tilde{p}_{jk}.x_{kjl}; \quad \forall j, k, l \tag{7}$$

$$r_{kl} \geq c_{kl-1}; \quad \forall k, l \geq 2 \tag{8}$$

$$c_{kl} \geq P_{kl} + r_{kl}; \quad \forall k, l \tag{9}$$

$$t_{kl} \geq c_{kl} - \tilde{d_j} - U(1 - x_{kjl}); \quad \forall j, k, l \tag{10}$$

$$C_{max} \geq c_{kl}; \quad \forall k, l$$

(11)

$$T_{max} \geq t_{kl}; \quad \forall k, l \tag{12}$$

$$x_{kjl} = 0 or 1; \quad \forall j, k, l \tag{13}$$

$$r_{kl} \geq 0, P_{kl} \geq 0, c_{kl} \geq 0, t_{kl} \geq 0, T_{max} \geq 0, C_{max} \geq 0; \quad \forall k, l \tag{14}$$

Objective functions (1) and (2) minimize the makespan and maximum tardiness, respectively. Constraint (3) ensures that each job belongs to only one batch and is processed only on one machine. Constraint (4) guarantees that the total jobs assigned to each batch do not violate the machine capacity. Constraint (5) ensures that until one batch on a machine is empty, jobs are not assigned to subsequent batches. Constraints (6) and (7) define the ready time and the processing time of a batch. Constraints (8) and (9) state that the ready time of a batch on each machine must be greater than or equal to the completion time of the preceding batch and completion time of a batch is equal to the sum of the ready time of the batch and its processing time. Constraint (10) determines tardiness of a batch. Constraints (11) and (12) are the definition of the makespan and maximum tardiness, respectively. Constraints (13) and (14) specify a domain of variables.

3 Exact methods

In this paper, we employ two exact methods (namely, a two-phase fuzzy method and a ϵ-constraint method) for solving small-sized problems. Before explaining the methods, the fuzzy mathematical model provided in Section 2 must be converted to its crisp equivalent mathematical model according to the Jimenezs method [18]. Some definitions are presented. A fuzzy linear programming (FLP) is as follows:

$$\begin{aligned} &Min \;\; \tilde{c}^t x \\ &s.t. \;\; x \in \{x \in R_n \mid \tilde{A} \geq \tilde{B}, x \geq 0\} \end{aligned} \tag{15}$$

Definition 3.1. *The expected interval (EI) and expected value (EV) of a trapezoidal fuzzy number (e.g.,* $\tilde{A} = (a_1, a_2, a_3, a_4)$ *) are definded by [18]:*

$$EI(\tilde{A}) = [E_1^A, E_2^A] = [\frac{1}{2}(a_1 + a_2), \frac{1}{2}(a_3 + a_4)] \tag{16}$$

$$EV(\tilde{A}) = \frac{E_1^A + E_2^A}{2} = \frac{a_1 + a_2 + a_3 + a_4}{4} \tag{17}$$

Definition 3.2. *Suppose* $\tilde{A}$ *and* $\tilde{B}$ *be two fuzzy numbers,* $\tilde{A}$ *is bigger than or equal to* $\tilde{B}$ *at least at the degree of satisfaction* α*, it is presented by* $\tilde{A} \geq_\alpha \tilde{B}$*, when [18]:*

$$(1-\alpha)E_2^A + \alpha E_1^A \geq \alpha E_2^B + (1-\alpha)E_1^B \;\; or \;\; \frac{E_2^A - E_1^B}{E_2^A - E_1^A + E_2^B - E_1^B} \geq \alpha \tag{18}$$

Definition 3.3. *Any obtained optimal solution by the model* (19) *is an* α*-satisfied optimal solution (the satisfaction degree* α *is the degree that the decision-maker (DM) is willing to accept a solution) of the model* (15)[18].

$$Min \;\; EV(\tilde{c})x \;\; s.t. \;\; x \in \{x \in R^n \mid \tilde{A}x \geq_\alpha \tilde{B}, x \geq 0\} \tag{19}$$

Parameters	Generation way
p_{jk}	Discrete Uniform [1, 100]
r_j	Discrete Uniform [0, 100]
d_j	Uniform $[P*(1-t-\frac{r}{2}),(1-t+\frac{r}{2})$, $t=0.8$, $r=0.2$, $P=\frac{\sum_{j=1}^{N}\sum_{k=1}^{M}p_{jk}}{2*M}$
x_1, x_2, x_3, x_4	2* Uniform [0, 1]
$p_{\tilde{jk}}$	$[p_{jk}x_1, p_{jk}x_2, p_{jk}x_3, p_{jk}x_4]$
$\tilde{d_j}$	$[d_jx_1, d_jx_2, d_jx_3, d_jx_4]$
$\tilde{r_j}$	$[r_jx_1, r_jx_2, r_jx_3, r_jx_4]$
L_k	Discrete Uniform [10, 20]
s_j	Discrete Uniform [1, 5]
α	0.3,0.5,0.9

Table 1: Parameters of the Numerical Example

According to the above definitions, the fuzzy mathematical model is converted to its crisp equivalent mathematical as follows:

Objective functions (1) and (2)

$$s.t.\ \ r_{kl} \geq [\alpha E_2^{r_j} + (1-\alpha)E_1^{r_j}].x_{kjl};\ \ \forall j,k,l \tag{20}$$

$$P_{kl} \geq [\alpha E_2^{p_{jk}} + (1-\alpha)E_1^{p_{jk}}].x_{kjl};\ \ \forall j,k,l \tag{21}$$

$$t_{kl} \geq c_{kl} - [\alpha E_2^{d_j} + (1-\alpha)E_1^{d_j}] - U(1-x_{kjl});\ \ \forall j,k,l \tag{22}$$

The rest of constraints remains unchanged.

We assume the processing times, ready times and due dates are trapezoidal fuzzy numbers as follows:
$\tilde{r_j} = (r_j^1, r_j^2, r_j^3, r_j^4)$ $\tilde{d_j} = (d_j^1, d_j^2, d_j^3, d_j^4)$ $p_{\tilde{jk}} = (p_{jk}^1, p_{jk}^2, p_{jk}^3, p_{jk}^4)$

Thereby the final crisp equivalent mathematical is as follows:

Objective functions (1) and (2)

$$s.t.\ \ r_{kl} \geq [\alpha\frac{r_j^3 + r_j^4}{2} + (1-\alpha)\frac{r_j^1 + r_j^2}{2}].x_{kjl};\ \ j,k,l \tag{23}$$

$$P_{kl} \geq [\alpha\frac{p_{jk}^3 + p_{jk}^4}{2} + (1-\alpha)\frac{p_{jk}^1 + p_{jk}^2}{2}].x_{kjl};\ \ \forall j,k,l \tag{24}$$

$$t_{kl} \geq c_{kl} - [\alpha\frac{d_j^3 + d_j^4}{2} + (1-\alpha)\frac{d_j^1 + d_j^2}{2}] - U(1-x_{kjl});\ \ \forall j,k,l \tag{25}$$

The rest of constraints remains unchanged.

The final crisp mathematical model is solved by ε-constraint and two-phase fuzzy method using Lingo 8.0 software. Fuzzy processing times, fuzzy ready times, fuzzy due dates, machine capacities, job sizes and other parameters are generated according to Table 1.

3.1 ε-constraint method

In the form of this method, one of the objective functions is placed as an objective function to be optimized and other objective functions are transferred into constraints as follows:

$$Min\ \ f_j(x)\ \ s.t.\ \ f_h(x) \leq \varepsilon_h;\ \ h=1,\ldots,m,\ \ h \neq j,\ \ f_h^{min} \leq \varepsilon_h \leq f_h^{max}\ \ x \in S \tag{26}$$

Where$j \in \{1, \ldots, m\}$ and ε_h is upper bound for the objective $h(h \neq j)$. Various Pareto solutions can be found by changing the value of ε_h. We can let $f_h^{min} = f_h^*$ (where f_h^* is ideal value of f_h that can be computed by minimizing f_h individually) and $f_h^{max} = f_h^{nadir}$ (f_h^{nadir} is constructed from the worst value of f_h that can be estimated by using a payoff table [4]). In this study, the makespan is placed as an objective function that should be minimized, the maximum tardiness is transferred into constraints as follows:

$$Min \;\; C_{max} \;\; s.t. \;\; T_{max} \leq \varepsilon_T; \;\; T_{min}^* \leq \varepsilon_T \leq T_{max}^{nadir} \;\; Constraints \;\; 3-5, 8, 9, 11, 12, 13, 14, 23-25 \tag{27}$$

The model (27) is solved by Lingo 8.0 software. The computational results, for a sample problem with nine jobs and three machines, are shown in Table 2.

3.2 Two-phase fuzzy method

The two-phase fuzzy method is used for solving multiple objective linear programming (MOLP) with a fuzzy compromise approach [24]. In this paper, we use a two-phase fuzzy method for solving the following model:

$$Min \;\; f_1(x) = C_{max} \;\; Min \;\; f_2(x) = T_{max} \;\; s.t. \;\; Constraints \;\; 3-5, 8, 9, 11, 12, 13, 14, 23-25 \tag{28}$$

Definition 3.4. *Membership function of each objective function $f_k(k = 1, \ldots, m)$ can be defined as the following* [24]*:*

$$\mu_k(x) = \begin{cases} 1 & \text{if } f_k(x) \leq f_k^{min} \\ \frac{f_k^{max} - f_k(x)}{f_k^{max} - f_k^{min}} & \text{if } f_k^{min} \leq f_k(x) \leq f_k^{max} \\ 0 & \text{if } f_k(x) > f_k^{max} \end{cases} \tag{29}$$

Where f_k^{min} is the ideal solution and f_k^{max} is the negative ideal solution. f_k^{min} and f_k^{max} are considered similar to as mentioned in $\varepsilon - constraint$ method i.e. $f_k^{min} = f_k^*$ and $f_k^{max} = f_k^{nadir}$.

In the first step, Zimmermann [40] suggested max-min operator approach for solving the model (28) as follows:

$$Max \;\; \lambda \;\; s.t. \;\; \lambda \leq \mu_k(x); \;\; k = 1, \ldots, m \;\; \lambda \in [0, 1]; \;\; x \in S; \tag{30}$$

We have:

$$Max \;\; \lambda \;\; s.t. \;\; \lambda \leq \frac{C_{max}^{nadir} - C_{max}}{C_{max}^{nadir} - C_{max}^*}; \;\; \lambda \leq \frac{T_{max}^{nadir} - T_{max}}{T_{max}^{nadir} - T_{max}^*}; \;\; \lambda \in [0, 1]; \; Constraints \;\; 3-5, 8, 9, 11, 12, 13, 14, 23-25 \tag{31}$$

Where λ is the overall satisfaction degree of the objective functions. For the mentioned sample problem in Subsection 3.1, the model (31) solved by Lingo 8.0 software and the optimal values of the variables λ, C_{max} and T_{max} are shown in Table 3 as λ^*, C_{max}^1 and T_{max}^1 .

In the second step, Li et al [24] proposed the following linear programming model:

$$Max \;\; \omega = \sum_{k=1}^{m} v_k \lambda_k \; s.t. \;\; \lambda_k^l \leq \lambda_k \leq \mu_k(x); \;\; k = 1, \ldots, m \; \lambda_k \in [0, 1]; \;\; k = 1, \ldots, m \;\; \sum_{k=1}^{m} v_k = 1; \; x \in S; \tag{32}$$

Where λ_k^l is the minimum satisfaction degree of the kth objective function and is equal to membership functions of C_{max}^1 and T_{max}^1 [24]. We have:

$$\begin{aligned} Max \;\; \omega = 0.5\lambda_1 + 0.5\lambda_2 \; s.t. \;\; \lambda_1 \leq \frac{C_{max}^{nadir} - C_{max}}{C_{max}^{nadir} - C_{max}^*}; \; \lambda_1 \geq \frac{C_{max}^{nadir} - C_{max}^1}{C_{max}^{nadir} - C_{max}^*}; \; \lambda_2 \leq \frac{T_{max}^{nadir} - T_{max}}{T_{max}^{nadir} - T_{max}^*}; \\ \lambda_2 \geq \frac{T_{max}^{nadir} - T_{max}^1}{T_{max}^{nadir} - T_{max}^*}; \; \lambda_1, \lambda_2 \in [0, 1]; \; Constraints \;\; 3-5, 8, 9, 11, 12, 13, 14, 23-25 \end{aligned} \tag{33}$$

The model (33) solved by Lingo 8.0 software and the optimal values of the variables ω, C_{max} and T_{max} are shown in Table 3 as ω^*, C_{max}^{final} and T_{max}^{final}.

The Pareto solutions obtained from the exact methods are shown in Tables 2 and 3.

			Pareto Solutions	
α	T^{*}_{max}	T^{nadir}_{max}	T_{max}	C_{max}
0.3	103.4145	105.1837	103.4145	124.3091
			105.0543	117.0035
0.5	124.2714	128.8552	124.2714	146.7826
			127.5069	143.9846
0.9	171.7004	177.5852	171.7004	197.6462

Table 2: Results of the ε-constraint Method

α	C^{*}_{max}	C^{nadir}_{max}	T^{*}_{max}	T^{nadir}_{max}
0.3	117.0035	124.3091	103.4145	105.1837
0.5	143.9846	155.5164	124.2714	128.8552
0.9	197.6462	197.6462	171.7004	177.5852

	Results of the first step			Results of the second step			Pareto Solutions	
α	λ^{*}	C^{1}_{max}	T^{1}_{max}	ω^{*}	C^{final}_{max}	T^{final}_{max}	T_{max}	C_{max}
0.3	0.73	119.4062	105.0543	0.87	117.003	105.0543	105.0543	117.0035
0.5	0.76	146.7826	125.3836	0.88	146.782	124.2714	124.2714	146.7826
0.9	0.76	197.6462	171.7004	-	-	-	171.7004	197.6462

Table 3: Results of the Two-phase Fuzzy Approach

4 Solution methodology

Arroyo and Leung [1] showed that the $R_m \mid s_j, r_j, p-batch \mid C_{max}$ problem is NP-hard, so $R_m \mid p\tilde{_{jk}}, \tilde{d}_j, \tilde{r_j}, s_j, p-batch \mid C_{max}, T_{max}$ problem that is more complicated because it is both multi-objective and uncertain is NP-hard as well. However, exact methods can not solve the large-sized problems in a polynomial time. Therefore, we proposed two meta-heuristics algorithms, namely fuzzy non-dominated sorting genetic algorithm (FNSGA-II) and fuzzy multi-objective discrete teaching-learning-based optimization (FMODTLBO) in order to solve the medium and large-sized problems.

In this section, two meta-heuristic algorithms (i.e. FMODTLBO and FNSGA-II) with the capability of solving the problems without requiring them to be defuzzied are explained. Thereby, the objective functions are fuzzy numbers.

Definition 4.1. *A multi-objective problem is handled as follows* [4]*:*

$$Minimize(f_1(x), f_2(x), \ldots, f_m(x)) \quad s.t. \quad x \in S; \tag{34}$$

$x_i \in S$ dominates $x_j \in S$ if

$$f_k(x_i) \leq f_k(x_j) \text{ for all } k \in \{1, 2, \ldots, m\} \; f_k(x_i) < f_k(x_j) \quad \text{for at least one } k \in \{1, 2, \ldots, m\} \tag{35}$$

Since the objective functions are fuzzy numbers, we cannot compare them with Expression (35). Therefore, it is done using the fuzzy ranking method proposed by Jimenez et al [18] (Eq. 18).

Considering the above-mentioned description in this section, two fuzzy meta-heuristic algorithms are expressed in this paper.

4.1 FMODTLBO algorithm

Teachinglearning-based optimization (TLBO) proposed by Roa et al. [32] is based on learning a group of learners of a teacher in a class and is considered as a population-based meta-heuristic algorithm. The best learner in each population is considered as the teacher. The learning process includes a teacher stage, in which the learners learn through the teacher, and a learner stage, where the learners increase their knowledge through interaction with each other. The original TLBO was considered for continuous problems; however, we use a discrete representation for our solutions. Therefore, we use a discrete version of TLBO [23]. Let N_{pop} be the population size and $Max - It$ be the maximum number of times to repeat the algorithm until the termination condition is met.

This section includes solution representation, initial population, Feasible solutions and proposed FMODTLBO algorithm.

4.1.1 Solution representation

We use a new representation of the solutions in the proposed algorithms. Our representation is a matrix $2 \times N$ that its first and second rows show the batches and the machines numbers, respectively. For example, a representation for the mentioned sample problem in Sub-section 3.1 is shown in Figure 1.

4.1.2 Initial population

The main steps to create an initial population are as follows:

Step 1) Generate $M + N - 1$ random numbers between 0 and 1, then according to Figure 1, find the jobs that assigned to each machine ($M = 3$, $N = 9$). Numbers that are smaller than and equal to N represent the jobs, and numbers that are larger than N represent the machines. For example, number $N+1$ is $Machine_1$, $N+2$ is $Machine_2$, and similarity $N + M - 1$ is $Machine_{M-1}$.

Step 2) assign the jobs on each machine to the batches and create a solution according to proposed representation in Sub-section 4.1.1. This work is done by using the batch first-fit (BFF) heuristic method. On each machine, consider the first job that is not assigned to the certain batch and find the first batch with enough space to insert it. If no batch is found with this condition, create a new batch. Repeat Step 2 until no job remains unassigned to the batches. It is shown that if the jobs on each machine are sorted based on their processing times in a descending order and then are assigned to the batch based on the BFF algorithm (called the batch first-fit longest processing times (BFFLPT) method), it is considered a better comparison to the BFF for minimizing the makespan. If the jobs on each machine are sorted based on their due dates in an ascending order and then are assigned to batch based on the BFF (called the batch fist-fit earliest due date (BFFEDD) method), it is considered a better comparison to the BFF for minimizing the maximum tardiness.

Step 3) Repeat Steps 1 and 2 until an initial population is created. Since we consider two objectives (i.e., minimizing the makespan and minimizing the maximum tardiness), one half of the initial population is created by BFFLPT procedure and another half of the initial population is created by BFFEDD procedure.

4.1.3 Feasible solutions

The solutions obtained from the proposed algorithms may be not feasible because for the following reasons:

1. Lack of some batch numbers in the first row of the solution representation for one or more machine. To solve this problem, we change the batch numbers so that the batch numbers are sequential.

2. Total job sizes assigned to a batch on a machine exceed the machine capacity. In this case, we use two heuristics, HF1 and HF2 [20] considering the makespan and the maximum tardiness, respectively. We consider these two objectives simultaneously, and then we choose HF1 or HF2 randomly with an equal chance. In following, the details of the heuristics HF1 and HF2 are provided.

HF1: On each machine

Step 1) Consider the batches that exceed the machine capacity and select the batch with the longest BPT. Then, choose the job with the largest processing time in the selected batch.

Step 2) Insert the selected job in Step 1 in the BFF (with the minimum residual capacity) having the BPT longer than the processing time of the selected job. If there is no batch with this condition, insert the job in BFF with the longest BPT. If the batch with the residual capacity greater than or equal to the size of the selected job does not exist, insert the selected job in a new batch.

Step 3) Repeat Steps 1 and 2 until there is no batch that exceeds the machine capacity.

HF2: On each machine

Step 1) Consider the batches that exceed the machine capacity and select the batch with the shortest BDD. Then, choose the job with the shortest due date in the selected batch.

Step 2) Insert the selected job in Step 1 in the BFF (with the minimum residual capacity) having the shortest BDD. If the batch with the residual capacity greater than or equal to the size of the selected job does not exist, insert the selected job in a new batch.

Step 3) Repeat Steps 1 and 2 until there is no batch that exceeds the machine capacity.

4.1.4 Proposed FMODTLBO algorithm

The proposed FMODTLBO algorithm proposed in this research is described as follows:

	Job_1	Job_2	Job_3	Job_4	Job_5	Job_6	Job_7	Job_8	Job_9
Batch No.	1	1	3	2	2	1	1	3	2
Machine No.	2	3	2	2	3	1	1	2	2

$Machine_3$	$Batch_1$	$Batch_2$
	Job_2	Job_5

$Machine_2$	$Batch_1$	$Batch_2$	$Batch_3$
	Job_1	Job_4, Job_9	Job_3, Job_8

$Machine_1$	$Batch_1$
	Job_6, Job_7

Solution Representation, Encoding and Decoding Procedures

Producing 11 (3+9-1) real random in (0,1) and putting them into the boxes:

1	2	3	4	5	6	7	8	9	10	11
0.905	0.127	0.913	0.964	0.097	0.278	0.546	0.957	0.970	0.157	0.632

Sorting real numbers in descending order:

9	4	8	3	1	11	7	6	10	2	5
0.970	0.964	0.957	0.913	0.905	0.632	0.546	0.278	0.157	0.127	0.097

Decoding procedure:

$Machine_1$	7,6
$Machine_2$	9,4,8,3,1
$Machine_3$	2,5

Initial Population, Encoding and Decoding Procedures.

Figure 1: Solution Representation and Initial Population

Step 1) Create an initial population with N_{pop} learners or solutions by using Sub-section 4.1.2. Then, calculate the value of the objective functions C_{max} and T_{max} (Eqs. 11 and 12).

Step 2) Use a fast non-dominated sorting approach proposed by Deb et al. [11] to compute the ranks and the crowding distances of the learners. Since the objective functions are fuzzy numbers, a fuzzy ranking method [18] (Eq. 18) is used to compare objective functions to compute the ranks and the excepted values of objective functions (Eq. 17) are used in order to compute the crowding distances.

Step 3) Teacher stage: In this stage, the learner's level of knowledge in iteration t ($x_{i,t}$) is transferred by using DM_t (i.e., a difference between the teacher ($x_{T,t}$) and mean result of learners (M_t)). Updated learner ($x'_{i,t}$) is considered by:

$$x'_{i,t} = x_{i,t} + DM_t \tag{36}$$

$$DM_t = r_t(x_{T,t} - T_F M_t) \tag{37}$$

Where T_F is the teaching factor and its value can be either 1 or 2 and r_t is a random number in the range [0, 1]. The detailed implementation of our discrete problem is given below:

Step 3.1) Select teacher: In iteration t, the teacher ($x_{T,t}$) is considered as the best learner, who is a learner (i.e., solution) with the lowest rank. if the ranks are equal, we select the solution with the greatest crowding distance.

Step 3.2) Mean results of learners: For obtaining the mean result of learners at iteration t (M_t), a heuristic is proposed as follows:

Step 3.2.1) Compute round($\frac{max_{i=1,\ldots,N_{pop}} rank_{x_{i,t}} + min_{i=1,\ldots,N_{pop}} rank_{x_{i,t}}}{2}$).

Step 3.2.2) Consider the learner (i.e., one of the learners), whose rank is equal to the computed value in Step 3.2.1 as the mean results of the learners, M_t.

Example 4.2. *Consider the example mentioned in Sub-section 4.1.1 and create a population with three learners as follows:*

$x_{1,t} = \begin{Bmatrix} 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 3 & 2 & 3 & 1 & 2 & 1 & 2 & 2 & 1 \end{Bmatrix}$,$x_{2,t} = \begin{Bmatrix} 1 & 1 & 1 & 2 & 3 & 2 & 1 & 1 & 1 \\ 1 & 3 & 3 & 1 & 1 & 1 & 2 & 2 & 1 \end{Bmatrix}$ *and* $x_{3,t} = \begin{Bmatrix} 1 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 \\ 3 & 1 & 2 & 2 & 2 & 1 & 1 & 2 & 1 \end{Bmatrix}$. *The computed ranks of the learners are 2, 1 and 1, respectively. Therefore, we have:*

$x_{T,t} = \begin{Bmatrix} 1 & 1 & 1 & 2 & 3 & 2 & 1 & 1 & 1 \\ 1 & 3 & 3 & 1 & 1 & 1 & 2 & 2 & 1 \end{Bmatrix}$ *and* $M_t = \begin{Bmatrix} 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 3 & 2 & 3 & 1 & 2 & 1 & 2 & 2 & 1 \end{Bmatrix}$.

Step 3.3) DM_t: The difference between the teacher ($x_{T,t}$) and the mean result of the learners (M_t) at iteration t is computed by using Eq. 37. In Example 4.2, suppose that $T_F = 1$ and $r_t = 0.8$:

$DM_t = \begin{Bmatrix} 0 & 0 & 0 & 0.8 & 1.6 & 0 & 0 & 0 & 0 \\ 1.6 & 0.8 & 0 & 0 & -0.8 & 0 & 0 & 0 & 0 \end{Bmatrix}$

Step 3.4) Update the learners: Each learner in the population is updated by using Eq. 36. In the first row, consider the integer part of each number. In the second row, if the updated number is negatively converted to a positive number, then consider the integer part of each number. if the integer part is equal to zero or is greater than M, put a random integer number in the interval $[1, M]$ instead of it. It should be noted that an updated learner may be an unfeasible solution for the reasons mentioned in Sub-section 4.1.3. Therefore, we should use methods described in this section to make it feasible. In Example 4.2, the updated form of $x_{1,t}$ is $x'_{1,t} = \begin{Bmatrix} 1 & 1 & 1 & 1.8 & 2.6 & 2 & 1 & 1 & 1 \\ 1.4 & 2.8 & 3 & 1 & 1.2 & 1 & 2 & 2 & 1 \end{Bmatrix}$. With considering the integer part of obtained numbers as $\begin{Bmatrix} 1 & 1 & 1 & 1 & 2 & 2 & 1 & 1 & 1 \\ 1 & 2 & 3 & 1 & 1 & 1 & 2 & 2 & 1 \end{Bmatrix}$ and using HF1, feasible $x'_{1,t}$ is $\begin{Bmatrix} 3 & 1 & 1 & 1 & 2 & 2 & 1 & 1 & 1 \\ 1 & 2 & 3 & 1 & 1 & 1 & 2 & 2 & 1 \end{Bmatrix}$.

Step 3.5) Acceptance updated learner: If feasible $x'_{i,t}$ dominates $x_{i,t}$, then replace $x_{i,t}$ by $x'_{i,t}$. This work is done by the fuzzy ranking method (Eq. 18) and Expression (35).

Step 4) Learner stage: In this stage, we first compute the ranks and the crowding distances of the learners obtained from teacher stage (similar to what was mentioned in Step 2). Then, we do the following steps in iteration t:

Step 4.1) For learner $x_{i,t}$, randomly select another learner $x_{j,t}(i \neq j)$.

Step 4.2) Update learner $x_{i,t}$ by using Eq. 38 or 39.

- If $rank_{x_{i,t}} < rank_{x_{j,t}}$(if $rank_{x_{i,t}} = rank_{x_{j,t}}$ then if $crowding\ \ distance_{x_{i,t}} > crowding\ \ distance_{x_{j,t}}$)

$$x'_{i,t} = x_{i,t} + r_t(x_{i,t} - x_{j,t}) \quad r_t \in (0, 1) \tag{38}$$

- If $rank_{x_{j,t}} < rank_{x_{i,t}}$(if $rank_{x_{j,t}} = rank_{x_{i,t}}$ then if $crowding\ \ distance_{x_{j,t}} > crowding\ \ distance_{x_{i,t}}$)

$$x'_{i,t} = x_{i,t} + r_t(x_{j,t} - x_{i,t}) \quad r_t \in (0, 1) \tag{39}$$

Step 4.3) Make $x'_{i,t}$ feasible as mentioned in Sub-section 4.1.3.

Step 4.5) If feasible $x'_{i,t}$ dominates $x_{i,t}$, then replace $x_{i,t}$ by $x'_{i,t}$ similar to what was mentioned in Step 3.5.

Example 4.3. *Consider the example mentioned in Step 3.1 and suppose that* $x_{1,t}$ *and* $x_{3,t}$ *are the two learners, which are selected.* $rank_{x_{3,t}}$ *is less than* $rank_{x_{1,t}}$*. Therefore, by using Eq. 38 and* $r_t = 0.4$ *, we have* $x'_{3,t} = \left\{ \begin{array}{ccccccccc} 1 & 1 & 1 & 2.4 & 1 & 2 & 1 & 1 & 1 \\ 3 & 0.6 & 1,6 & 2.4 & 2 & 1 & 0.6 & 2 & 1 \end{array} \right\}$*. According the expressed comments in Steps 3 and 4, the final solution is* $x'_{3,t} = \left\{ \begin{array}{ccccccccc} 1 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 \\ 3 & 3 & 1 & 2 & 2 & 1 & 2 & 2 & 1 \end{array} \right\}$.

Step 5) Repeat Steps 2 to 4 until $Max - It$ occurs.

4.2 Fuzzy non-dominated sorting genetic algorithm (FNSGA-II)

The proposed FNSGA-II algorithm proposed in this research is described as follows [27]:

4.2.1 Parameters

Pop : Population

N_{pop} : Size of the population

P_c : Percentage of the offspring population that is completed by the crossover operation

P_m : Percentage of the offspring population that is completed by the mutation operation

N_{P_c} : Numbers of offspring that is created by the crossover operation

N_{P_m} : Numbers of offspring that is created by the mutation operation

Tournament-size: Number of individuals that are selected for the tournament

Max-It : Maximum number of times to repeat the algorithm (i.e., termination condition)

4.2.2 Steps

Step 1) Create an initial population with N_{pop} chromosomes or individuals by using Sub-section 4.1.2. Then, calculate the value of the objective functions C_{max} and T_{max} (Eqs. 11 and 12).

Step 2) Compute the ranks and the crowding distances of the individuals(similar to what was mentioned in Step 2 of Sub-section 4.1.4).

Step 3) Create the offspring population with N_{pop} numbers that includes N_{P_c} $(round(N_{pop} \times P_c))$ offspring obtained from the crossover operation (Figure 2), N_{P_m} $(round(N_{pop} \times P_m))$ offspring obtained from the mutation operation (Figure 2) and the residue of the offspring population selected from the parent population (all of the selections for the crossover operation, mutation operation and the residue of offspring population doing in $Tournament\ \ size$, using the computed ranks and crowding distances in Step 2).

Step 4) Combine the parent population and the offspring population and create a population with $2 \times N_{pop}$ individuals then use the fast non-dominated sorting approach proposed by Deb et al. [11] to compute the ranks and the crowding distances (similar to what was mentioned in Step 2 of Sub-section 4.1.4) of the individuals of the current population and create a new population with N_{pop} individuals, using the method offered by Deb et al. [11].

Step 5) Repeat Steps 2 to 4 until $Max - It$ occurs.

4.2.3 Crossover

We use the parameterized uniform crossover operator, which is a general form of a uniform crossover, in which the swapping probability of each gene locus is controllable [31]. In this paper, the probability of a gene to be selected by parents will be determined by the result of a biased coin (Figure 2).

4.2.4 Mutation

We use a swapping mutation operator. For each selected parent, two genes are selected randomly and their locations are interchanged (Figure 2).

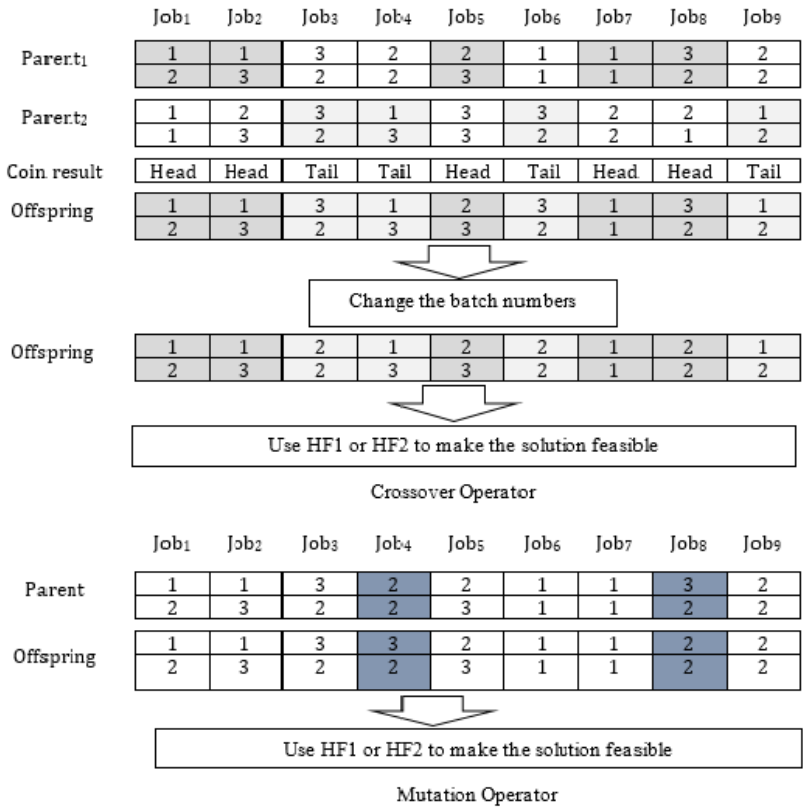

Figure 2: Crossover and Mutation Operators in FNSGA-II Algorithm

5 Computational experiments

In order to test the effectiveness of the FNSGA-II and FMODTLBO, several test problems are solved and then their performances are compared with a number of evolution metrics. The proposed meta-heuristics are coded in MATLAB R2016a software.

5.1 Test problem instance

Computational experiments are conducted in medium and large-sized problems according to Table 4.

5.2 Evolution metric

Quality of non-dominated solutions obtained from proposed meta-heuristic algorithms is used to compare these algorithms. In this paper, three metrics [14] are employed as follows:

5.2.1 Number of non-dominated solutions ($N-metric$)

The number of the total Pareto solutions obtained from each algorithm, T, is not considered as $N-metric$ because a Pareto solution of an algorithm may dominate one or more Pareto solutions of another algorithm or vice versa, therefore, this metric considers the number of final the non-dominated solutions obtained by algorithms. The larger $N-metric$, the better the algorithm is.

Medium- sized problems						Large- sized problems					
Problem	M	N	Problem	M	N	Problem	M	N	Problem	M	N
1	3	10	9	5	20	1	7	30	9	9	40
2	3	20	10	5	40	2	7	60	10	9	80
3	3	30	11	5	60	3	7	90	11	9	120
4	3	40	12	5	80	4	7	120	12	9	160
5	4	15	13	6	25	5	8	35	13	10	45
6	4	30	14	6	50	6	8	70	14	10	90
7	4	45	15	6	75	7	8	105	15	10	135
8	4	60	16	6	90	8	8	140	16	10	180

Table 4: Test Problem Instances

5.2.2 Ratio of non-dominated solutions ($R-metric$)

The value of the $R-metric$ is equal to the ratio of the $N-metric$ to T. The obtained $R-metric$ belongs to [0, 1] interval. $R-metric$ of a proposed algorithm refers to the ratio of a number of the total Pareto solutions that are not dominated by another proposed algorithm. The larger $R-metric$, the better the algorithm is.

5.2.3 Spread of Pareto solutions ($S-metric$)

$S-metric$ used for estimating the spread of the total Pareto solutions is calculated by Eq. (40).

$$S-metric=\sqrt{\frac{1}{T-1}\sum_{t=1}^{T}(\bar{d}-d_t)^2} \tag{40}$$

$$\bar{d}=\frac{\sum_{t=1}^{T}d_t}{T} \tag{41}$$

Where d_t is the Euclidean distance between each of Pareto solutions and their closest neighbor. The lower $S-metric$, the better the algorithm is. For computing this metric, we consider the excepted value of the objective functions (Eq. 17).

In this paper, we also consider the central processing unit time (CPUT) metric as the running time of algorithms.

5.3 Parameter setting

The quality of the solutions obtained from the proposed FNSGA-II and FMODTLBO algorithms is affected by the values of their parameters. To set the parameters of these algorithms in this papers, the response surface methodology (RSM) [29] using Design Expert 10.0.3.0 software is applied. The RSM explores the relation between several explanatory (i.e., input) variables and one or more response (i.e., output) variable. The main idea of the RMS is to use a sequence of designed experiments to obtain an optimal response variable by analyzing a regression equation. Since the proposed algorithms in this paper generate fuzzy Pareto solutions, the number of Pareto solutions created is used as responses [30]. First, the parameters (input variables) of each algorithm are discovered. Then, for each of them two levels, low and high, are considered (Table 5). By analyzing the results of Design Expert software, the best combination of parameters for medium and large-sized problems are created (Table 6).

5.4 Numerical examples and computational results

The details about the parameters of the numerical examples explained in Table 1. Each proposed algorithm runs 30 times on each test problem (test problems are shown in Tables 4) and the average results are recorded in Tables 7 and 8. The statistical results in order to compare the performance of the proposed algorithms are recorded in Tables 9 - 12.

Table 9 shows that the ρ-values in the results of the CPUT using the t-test method is less than 0.05 for medium-sized problems and is greater than 0.05 for large-sized problems. Therefore, there is a significant difference between the

Proposed algorithm	Parameters	Low	High
FMODTLBO	$Max - It$	5	15
	N_{pop}	25	35
	T_F	1	2
FNSGA-II	$Max - It$	10	90
	N_{pop}	10	50
	P_c	0.1	0.9
	P_m	0.02	0.1

Table 5: Parameters of the Proposed Algorithms and Their Levels

Proposed algorithm	Parameters	Medium-sized problems	Large-sized problems
FMODTLBO	$Max - It$	5	10
	N_{pop}	35	30
	T_F	1	1
FNSGA-II	$Max - It$	40	50
	N_{pop}	35	50
	P_c	0.6	0.5
	P_m	0.07	0.06

Table 6: Best Combination of the Parameter Values for Proposed Algorithms ($\alpha = 0.3$)

	CPUT		$S - metric$		$N - metric$		$R - metric$	
M/N	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II
3/10	10.36	29.06	0	9.55	2.67	4.67	1	0.93
3/20	16.84	33.72	23.97	22.82	3.33	7	1	1
3/30	27.13	40.51	20.31	56.34	3	5.67	1	1
3/40	36.86	47.95	0	28.69	3.67	6.67	1	1
4/15	11.69	31.42	4.89	12.55	2	3.67	0.89	1
4/30	22.26	44.62	35.09	11.84	2.67	5.67	0.98	1
4/45	37.89	46.45	21.76	20.25	4	3.67	1	1
4/60	55.54	61.53	23.32	21.32	4	5	1	1
5/20	13.86	32.37	15.36	8.49	2.67	5.33	0.88	1
5/40	29.71	43.31	3.17	40.78	2.33	5.33	0.89	0.93
5/60	46.89	59.93	7.56	18.59	2.67	7	0.83	0.95
5/80	66.79	73.96	30.24	43.44	3.33	6	0.83	1
6/25	16.58	35.05	16.46	14.95	2.33	4.33	0.82	1
6/50	32.86	46.81	14.91	14.63	4.33	6.33	1	1
6/75	52.37	66.09	24.67	20.86	5.67	8	0.81	1
6/90	70.24	80.91	30.42	50.54	4	6.67	1	0.95
Average	34.24	48.36	17.00	24.46	3.29	5.68	0.93	0.98

Table 7: Outputs of the FNSGA-II and FMODTLBO Algorithms for the Medium-sized Problems ($\alpha = 0.3$)

	CPUT		$S-metric$		$N-metric$		$R-metric$	
M/N	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II	FMOD TLBO	FNSGA-II
7/30	42.62	95.21	13.1	7.37	3	6.67	0.88	0.95
7/60	86.07	108.45	0	23.34	0.67	0.67	0.5	0.67
7/90	150.09	155.81	34.96	34.65	3.67	6	1	1
7/120	236.41	202.91	45.02	64.22	3	5.33	0.92	1
8/35	49.86	92.21	6.86	12.08	5	6.33	1	0.89
8/70	107.53	128.05	13.25	15.69	5.33	7.33	0.88	1
8/105	178.85	184.09	32.39	70.01	2.67	7.67	1	1
8/140	275.59	229.01	232.85	94.35	0.67	1	0.5	0.6
9/40	50.64	98.53	12.34	57.08	4.33	5	0.98	1
9/80	133.36	130.38	5.66	20.41	5.67	9	1	1
9/120	186.36	184.32	38.11	25.59	3.33	4.67	0.98	1
9/160	305.98	284.25	9.58	48.58	5.33	9	1	1
10/45	56.4	98.7	5.68	13.72	4	8	1	1
10/90	134.57	139.18	13.76	20.55	5.33	8.33	1	1
10/135	208.01	178.71	57.88	32.67	3.33	5	1	1
10/180	364.66	302.2	27.83	49.11	4.67	5.33	1	1
Average	160.44	161.00	34.33	35.01	3.75	5.95	0.92	0.94

Table 8: Outputs of the FNSGA-II and FMODTLBO Algorithms for the Large-sized Problems ($\alpha = 0.3$)

				Confidence interval of the difference 95 percentage	
Problem	DF	t	$\rho-$value	Lower	Upper
Meduim	30	-2.256	0.032	-26.89075	-1.33675
Large	30	-0.019	0.985	-59.73570	58.60945

Table 9: t -test for the CPUT

two meta-heuristic algorithms in terms of the CPUT for medium-sized problems. The confidence interval shows that FMODTLBO is better than FNSGA-II and there is no significant difference between the two meta-heuristic algorithms in terms of the CPUT for large-sized problems. Table 10 shows that the ρ-values in the results of the $S-metric$ using the t-test method are greater than 0.05 for both medium and large-sized problems. Therefore, there is no significant difference between the two meta-heuristic algorithms in terms of the $S-metric$ for both medium and large-sized problems. Table 11 shows that the ρ-values in the results of the $N-metric$ using the t-test method are less than 0.05 for both medium and large-sized problems. Therefore, there is a significant difference between the two meta-heuristic algorithms in terms of the $N-metric$ for both medium and large-sized problems. The confidence intervals show that FNSGA-II is better than FMODTLBO for both medium and large-sized problems. Table 12 shows that the ρ-values in the results of the $R-metric$ using the t-test method are less than 0.05 for medium-sized problems and greater than 0.05 for large-sized problems. Therefore, there is a significant difference between the two meta-heuristic algorithms in terms of the $R-metric$ for medium-sized problems. The confidence interval shows that FNSGA-II is better than the FMODTLBO and there is no significant difference between the two meta-heuristic algorithms in terms of the $R-metric$ for large-sized problems.

Table 13 show trapezoidal Pareto solutions obtained by FMODTLBO and FNSGA-II in the problem with nine jobs and three machines. The results show that the proposed algorithms produce similar Pareto solutions. Table 14 show trapezoidal Pareto solutions obtained by FMODTLBO and FNSGA-II in the problem with 180 jobs and 10 machines. This problem has the largest size in our research.

				Confidence interval of the difference 95 percentage	
Problem	DF	t	$\rho-$value	Lower	Upper
Meduim	30	-1.649	0.11	-17.27945	1.84070
Large	30	-0.165	0.87	-33.51424	28.49549

Table 10: t -test for the $S-metric$

				Confidence interval of the difference 95 percentage	
Problem	DF	t	$\rho-$value	Lower	Upper
Meduim	30	-6.143	0.000	-3.19287	-1.59963
Large	30	-3.045	0.005	-3.68906	-0.72719

Table 11: t -test for the $N-metric$

				Confidence interval of the difference 95 percentage	
Problem	DF	t	$\rho-$value	Lower	Upper
Meduim	30	-2.481	0.019	-0.09458	-0.00917
Large	30	-0.563	0.578	-0.13602	0.07727

Table 12: t -test for the $R-metric$

6 Conclusions

This paper studied a bi-objective model for a scheduling problem of unrelated parallel batch processing machines (UPBPMs) that minimizes the makespan and maximum tardiness considering a real-world application through fuzzy machine-dependent processing times, fuzzy ready times and fuzzy due dates. A bi-objective fuzzy mixed-integer linear programming (FMILP) model was proposed. A batch with these parameters (i.e., BPT, BRT and BDD) was presented by the longest processing time, longest ready time and shortest due date of the jobs belonging to the batch, respectively. The presented model was solved for small-sized problems by two methods (i.e., two-phase fuzzy and ϵ-constraint methods) in order to obtain a set of Pareto solutions. Additionally, two meta-heuristics (i.e., FNSGA-II and FMODTLBO) were proposed. In these algorithms, a matrix with two rows and N columns was used to show the jobs that were assigned to the batches that were processed on the machines. The results showed that the FNSGA-II was relatively better than FMODTLBO. In this study, we attempted to consider real conditions in an industrial environment. Of course, there are other conditions that help researchers to improve our research, such as pre-emption, precedence constraints, machine failures, machine-dependent/independent setup time and describing uncertainty in the scheduling problem using stochastic methods. Additionally, one can use other meta-heuristic algorithms and compare the related results with our FNSGA-II and FMODTLBO results.

References

[1] J. E. Arroyo, J. Leung, *Scheduling unrelated parallel batch processing machines with non-identical job sizes and unequal ready times*, Computers and Operations Research, **78** (2017), 117–128.

[2] J. E. Arroyo, J. Leung, *An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times*, Computer and Industrial Engineering, **105** (2017), 84–100.

[3] S. F. Attar, M. Mohammadi, R. Tavakkoli-Moghaddam, *Hybrid flexible flowshop scheduling problem with unrelated parallel machines and limited waiting times*, International Journal Advanced Manufacturing Technology, **68** (2013), 1583–1599.

[4] J. Brank, K. Deb, K. Miettinen, R. Slowinski, *Multiobjective optimization- interactive and evolutionary approach*, Springer, 2008.

[5] B. Cheng, K. Li, B. Chen, *Scheduling a single batch-processing machine with non-identical job size in fuzzy environment using an improved ant colony optimization*, Journal of Manufacturing Systems, **29**(1) (2010), 29–34.

[6] B. Cheng, S. Yang, X. Hu, B. Chen, *Minimizing makespan and total completion time for parallel batch processing machines with non-identical job sizes*, Applied Mathematical Modelling, **36**(7) (2012), 3161–3167.

[7] B. Cheng, Q. Wang, S. Yang, X. Hu, *An improved ant colony optimization for scheduling identical parallel batching machines with arbitrary job sizes*, Applied Soft Computing, **13**(2) (2013), 765–772.

[8] P. Damodaran, M. C. Velez-Gallego, *Heuristics for makespan minimization on parallel batch processing machines with unequal job ready times*, International Journal Advanced Manufacturing Technology, **49**(9) (2010), 1119–1128.

		Trapezoidal objective function values		Excepted value	
Algorithm	α	C_{max}	T_{max}	C_{max}	T_{max}
FMODTLBO	0.3	[22.63 61.87 122.87 228.33]	[-6.84 42.74 105.93 218.34]	108.92	90.04
	0.5	[16.34 144.29 233.91 336.4]	[-8.58 123.87 222.59 335.36]	182.74	168.31
		[30.37 123.85 214.86 378.10]	[0.903 104.73 197.93 368.11]	186.79	167.92
		[25.28 133.76 216.44 375.65]	[-20.07 88.72 195.03 362.47]	187.78	156.54
	0.9	[7.75 61.98 91.99 149.76]	[-22.03 40.86 80.38 138.62]	77.87	59.46
		[19.86 51.22 73.07 190.53]	[-25.49 6.17 51.66 177.35]	83.67	52.42
FNSGA-II	0.3	[22.63 61.87 122.87 228.33]	[-6.84 42.74 105.93 218.34]	108.92	90.04
	0.5	[16.34 144.29 233.91 336.4]	[-5.76 123.75 219.58 334.95]	182.74	168.13
		[30.37 123.85 214.86 378.10]	[0.903 104.73 197.93 368.11]	186.79	167.92
		[25.28 133.76 216.44 375.65]	[-20.07 88.72 195.03 362.47]	187.78	156.54
	0.9	[7.75 61.98 91.99 149.76]	[-22.03 40.86 80.38 138.62]	77.87	59.46
		[19.86 51.22 73.07 190.53]	[-25.49 6.17 51.66 177.35]	83.67	52.42

Table 13: Trapezoidal Pareto Solutions for the Problem with 3 Machines and 9 Jobs Solved by Proposed Algorithms

	Trapezoidal objective function values ($\alpha = 0.3$)		Excepted value	
Algorithm	C_{max}	T_{max}	C_{max}	T_{max}
FMODTLBO	[311.9 873 1356 1782]	[-210.54 78.62 228.87 341.56]	1080.7	109.6
	[263.5 712.6 1265.1 1598.5]	[-860.7 -296 1091.1 1674.5]	959.9	402.2
	[338 733.6 1192.3 1645.9]	[-680.65 48.41 619.75 982.14]	977.5	242.4
	[367.5 807 1256.9 1760.2]	[-514 25.2 275.4 1164.6]	1047.9	237.8
FNSGA-II	[486 797 1267.9 1543.9]	[-787 304.85 545.94 695.79]	1023.7	189.9
	[321.4 621.3 1190 1493.5]	[-246.51 -9.72 401.91 624.33]	906.55	192.5
	[299.2 688.1 1093.8 1418.6]	[-563.7 14.2 746.6 1570.6]	874.9	441.9
	[581.3 1002.2 1324.3 1668]	[-470.34 -2.42 465.42 676.79]	1144	167.4

Table 14: Trapezoidal Pareto Solutions for the Problem with 10 Machines and 180 Jobs Solved by Proposed Algorithms

[9] P. Damodaran, M. C. Velez-Gallego, *A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times*, Expert Systems with Applications, **39**(1) (2012), 1451–1458.

[10] P. Damodaran, M. C. Velez-Gallego, J. Maya, *A GRASP approach for makespan minimization on parallel batch processing machines*, **22**(5) (2011), 767–777.

[11] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation, **6**(2) (2002), 182–197.

[12] S. Dousthaghi, R. Tavakoli-Moghadam, A. Makui, *Solving the economic lot and delivery scheduling problem in a flexible job shop with unrelated parallel machines and a shelf life by a proposed hybrid PSO*, International Journal Advanced Manufacturing Technology, **68** (2013), 1401–1416.

[13] A. H. Gharehgozli, R. Tavakkoli-Moghaddam, N. Zaerpour, *A fuzzy-mixed-integer goal programming model for a parallel-machine scheduling problem' with sequence-dependent setup times and release dates*, Robotics and Computer-Integrated Manufacturing, **25** (2009), 853–859.

[14] Y. Y. Han, X. W. Gong, X. Y. Sun, Q. K. Pan, *An improved NSGA-II algorithm for multi- objective lot-streaming flow shop scheduling problem*, International Journal of Production Research, **25**(8) (2013), 2211–2231.

[15] M. Huletta, P. Damodaran, M. Amouie, *Scheduling non-identical parallel batch processing machines to minimize total weighted tardiness using particle swarm optimization*, Computers and Industrial Engineering, **113** (2017), 425–436.

[16] Z. Jia, Y. Zhang, J. Leung, K. Li, *Bi-criteria ant colony optimization algorithm for minimizing makespan and energy consumption on parallel batch machines*, Applied Soft Computing, **55** (2017), 226–237.

[17] L. Jiang, J. Pei, X. Liu, P. M. Pardalos, Y. Yang, X. Qian, *Uniform parallel batch machines scheduling considering transportation using a hybrid DPSO-GA algorithm*, The International Journal of Advanced Manufacturing Technology, **89** (2017), 1887–1900.

[18] M. Jimnez, M. Arenas, A. Bilbao, M. V. Rodrguez, *Linear programming with fuzzy parameters: an interactive method resolution*, European Journal of Operational Research, **177** (2007), 1599–1609.

[19] C. M. Joo, B. S. Kim, *Rule-based meta-heuristics for integrated scheduling of unrelated parallel machines, batches, and heterogeneous delivery trucks*, Applied Soft Computing, **53** (2017), 457–476.

[20] A. H. Kashan, B. Karimi, F. Jolai, *An effective hybrid multi-objective genetic algorithm for bi-criteria scheduling on a single batch processing machine with non-identical job sizes*, Engineering Applications of Artificial Intelligence, **23**(6) (2010), 911–922.

[21] X. Li, Y. Huang, Q. Tan, H.P. Chen, *Scheduling unrelated parallel batch processing machines with non-identical job sizes*, Computers and Operations Research, **40**(12) (2013), 2983–2990.

[22] X. Li, H. Ishii, M. Chen, *Single machine parallel-batching scheduling problem with fuzzy due-date and fuzzy precedence relation*, International Journal of Production Research, **53**(9) (2015), 2707–2717.

[23] J. Q. Li, Q. K. Pan, K. Mao, *A discrete teaching learning based optimization algorithm for realistic flowshop rescheduling problem*, Engineering Application of Artificial Intelligence, **37** (2015), 279–292.

[24] X. Q. Li, B. Zhang, H. Li, *Computing efficient solutions to fuzzy multiple objective linear programming problems*, Fuzzy Sets and Systems, **157** (2006), 1328–1332.

[25] R. Ma, L. Wan, L. Wei, J. Yuan, *Online bounded-batch scheduling to minimize total weighted completion time on parallel machines*, International Journal of Production Economics, **156** (2014), 31–38.

[26] E. Mehdizadeh, R. Tavakkoli-Moghaddam, M. Yazdani, *A vibration damping optimization algorithm for a parallel machines scheduling problem with sequence-independent family setup times*, Applied Mathematical Modelling, **39** (2015), 6845–6859.

[27] A. Mehrabian, R. Tavakoli-Moghaddam, K. Khalili-Damaghani, *Multi-objective routing and scheduling in flexible manufacturing system under uncertainty*, Iranian Journal of Fuzzy Systems, **14** (2017), 45–77.

[28] S. Molla-Alizadeh-Zavardehi, R. Tavakkoli-Moghaddam, F. Hosseinzadeh Lotfi, *A modified imperialist competitive algorithm for scheduling single batch-processing machine with fuzzy due date*, The International Journal of Advanced Manufacturing Technology, **85**(12) (2016), 2439–2458.

[29] D. C. Montgomery, *Design and analysis of experiments*, 8th Ed., Mississauga, John Wiley and Sons, 2012.

[30] M. Naderi-Beni, E. Ghobadian, S. Ebrahimnejad, R. TavakkoliMoghaddam, *Fuzzy bi-objective formulation for a parallel machine scheduling problem with machine eligibility restrictions and sequence-dependent setup times*, International Journal of Production Research, **52**(19) (2014), 5799–5822.

[31] F. Nadi, A. Tajudin Khader, *A study on the utility of parametric uniform crossover for adaptation of crossover operator*, Artificial Intelligence: Methodology, Systems, and Applications, **7557** (2012), 178–183.

[32] R. V. Rao, V. J. Savsani, D. P. Vakharia, *Teachinglearning-based optimization: A novel method for constrained mechanical design optimization problems*, Computer-Aided Design, **43** (2011), 303–315.

[33] B. Shahidi-Zadeh, R. Tavakkoli-Moghaddam, A. Taheri-Moghadamb, I. Rastgar, *Solving a bi-objective unrelated parallel batch processing machines scheduling problem: A comparison study*, Computers and Operations Research, **88** (2017), 71–90.

[34] O. Shahvari, R. Logendran, *An Enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes*, Computers and Operations Research, **77** (2017), 154–176.

[35] O. Shahvari, R. Logendran, *A bi-objective batch processing problem with dual-resources on unrelated-parallel machines*, Applied Soft Computing, **61** (2017), 174–192.

[36] R. Tavakkoli-Moghaddam, F. Taheri, M. Bazzazi, M. Izadi, F. Sassani, *Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints*, Computers and Operations Research, **36** (2009), 3224–3230.

[37] H. M. Wang, F. D. Chou, *Solving the parallel batch-processing machines with different release times, job sizes, and capacity limits by meta-heuristics*, Expert Systems with Applications, **37** (2010), 1510–1521.

[38] R. Xu, H. Chen, X. Li, *A bi-objective scheduling problem on batch machines via a pareto-based ant colony system*, International Journal of Production Economics, **145**(1) (2013), 371–386.

[39] S. Zhou, M. Liu, H. Chen, X. Li, *An effective discrete differential evolution algorithm for scheduling uniform parallel batch processing machines with non-identical capacities and arbitrary job sizes*, International Journal of Production Economics, **179** (2016), 1–11.

[40] H. J. Zimmermann, *Fuzzy programming and linear programming with several objective functions*, Fuzzy Sets and Systems, **1** (1978), 45–55.