Fuzzy Scheduling Problem on Unrelated Parallel Machine in JIT Production System

Javad Rezaeian[*], **Samir Mohammad-Hosseini, Sara Zabihzadeh, Keyvan Shokoufi**
Department of Industrial Engineering, Mazandaran University of Science and Technology, Iran
Email: j.rezaeian@ustmb.ac.ir

Abstract: This paper deals with unrelated parallel machines scheduling problem with sequence dependent setup times under fully fuzzy environment to minimize total weighted fuzzy earliness and tardiness penalties, which belongs to NP-hard class. Due to inherent uncertainty in Processing times, setup times and due dates of jobs, they are considered here with triangular and trapezoidal fuzzy numbers in order to take into account the unpredictability of parameters in practical settings. Although this study is not the first one to study on fuzzy parallel machines scheduling problem, it advances this area of research in three fields: (1) it selects a fuzzy environment to cover the whole area of the considered problem not just part of it, and also, it chooses an appropriate fuzzy method based on an in-depth investigation of the effect of spread of fuzziness on the variables; (2) It introduces a mathematical programming model for the addressed problem as an exact method; and (3) due to NP-hardness of the problem, it develops an existing algorithm in the literature for the considered problem through extensive simulated experiments and statistical tests on the same benchmark problem test by proposing a genetic algorithm (GA) and a modified simulated annealing (SA) methods to solve this hard combinatorial optimization problem. The result shows the superiority of our modified SA.

Keywords: fuzzy earliness and tardiness, unrelated parallel machine scheduling, sequence-dependent setup time, genetic algorithm, simulated annealing

1. Introduction

Unrelated parallel Machines Scheduling (UNPMS) problem is a class of parallel machine scheduling (PMS) problem in which machines are not identical and the processing time of jobs are related to both machine and job types. This framework for PMS problem is more practical than the other two parallel machine settings, i.e. identical and uniform parallel machines [1]. Moreover, considering setup time usually happens in industrial settings when various types of jobs are processed on machines [2]. Considering setup time as a distinct component of the processing time admits activities to be conducted simultaneously and consequently improves the performance of the manufacturing systems, significantly [3]. Finally, the Just-in-time (JIT) system is one of the most powerful systems that manufacturers are attempting to achieve it. Certainly, the systems in which earliness is also penalized are appropriate for models including storage costs and delivering spoilable commodities [4]. In real-world scheduling problems, the time related parameters of jobs are often encountered with imprecision. To cope with uncertainty issue in the context of scheduling, there are originally two methods to deal with this situation, (1) making use of the stochastic probabilistic theory and (2) fuzzy set theory. In the stochastic method, imprecise data are modelled by determining the probability distributions, for instance deduced from historical data [19]. Obviously, one of the most significant drawbacks of applying stochastic probabilistic theory is it's requiring a holistic knowledge about the statistical distributions of imprecise parameters [20]. The fuzzy method gives another way to model uncertainty, which appears to be more effective [21]. Fuzzy sets theory has been successfully utilized to treat imprecision in scheduling problems [22].

Here, we considered UNPMS problem with fuzzy processing times, fuzzy SD setup times and fuzzy due dates to minimize the total weighted fuzzy Earliness and Tardiness (E&T). In order to take into account the unpredictability of parameters in practical settings, Processing times, setup times and due dates of jobs, they are considered here with triangular and trapezoidal fuzzy numbers. The fuzzy set theory is applied to tackle the addressed problem, and a mathematical formulation is provided as an exact method. In view of the NP-hard nature of the problem, GA and SA methods are developed to solve large problems, evaluating performances of the proposed algorithms on a set of test problems. Moreover, another aim of introducing these algorithms is to develop and improve an existing algorithm in the literature.

In the literature, there are many surveys on UNPMS problem with setup time. Lee et al. [5] considered UNPMS with machine- and Sequence Dependent (SD) setup times to minimize the total tardiness. They introduced a Tabu Search (TS) algorithm that is constituted from various neighborhood generation methods. Haddad et al. [6] studied UNPMS *problem with setup time* in order to gain the maximum completion time, which is known as makespan. They proposed two algorithms that are based on Variable Neighborhood Descent and Iterated Local Search.

Lin and Ying [7] provided a hybrid artificial bee colony to minimize the makespan on UNPMS problem with job SD setup time. Caniyilmaz et al. [8] investigated UNPMS problem with setup time to minimize the sum of makespan and total tardiness. They developed a new neighborhood approach which is used by integrating into artificial bee colony and GA. Avalos-Rosales et al. [9] studied UNPMS problem with SD setup time to minimize makespan. They presented a new linearized makespan and several formulations.

Alvelos et al. [10] introduced a heuristic method based on column generation and a general purpose integer programming solver to address UNPMS problem with SD setup time. Their objective is to minimize the total weighted tardiness. Rezaeian et al. [11] considered UNPMS problem with SD setup times and due-date constraints to minimize the total cost of E&T. They provided a mathematical model and an integrated algorithm includes GA and SA to solve the addressed problem.

Also, Rezaeian et al. [12] studied UNPMS problem with SD setup time to minimize the sum of weighted E&T, and developed a mathematical formulation and a pareto based algorithm for the regarded problem. Gedik et al. [13] introduced a novel constraint programming for UNPMS problem with SD setup time. Shokoufi et al. [14] addressed uniform parallel machines scheduling problem with time-dependent learning effect, release date, allowable preemption and machine idle time to minimize the total weighted of E&T penalties. They provided a mathematical model, and two meta-heuristic algorithms. Soleimani et al. [44] addressed scheduling of unrelated parallel machines considering the simultaneous effects of start-time-related deterioration, position-related learning and sequence-related setup times, with the aim to achieve an optimized value for the mean weighted tardiness and power consumption minimization. They proposed three metaheuristic algorithms, namely, Genetic Algorithm (GA), Cat Swarm Optimization (CSO) and Interactive Artificial Bee Colony (IABC) for the problem.

The reader can refer to [15-18] for further investigation on UNPMS problem with setup time.

The above investigations include the PMS problems in a deterministic environment. In the following, we will deal with UPMS in fuzzy environment.

Yi and Wang [23] considered identical parallel machines with setup times to minimize the total E&T penalties. They developed a fuzzy logic embedded GA for the problem. Peng and Liu [24] developed three new types of fuzzy scheduling approach and designed a hybrid intelligent algorithm for PMS problem with fuzzy processing times. They propose a hybrid intelligent algorithm to solve the proposed models where fuzzy makespan, fuzzy lateness and fuzzy idleness are minimized. Anglani et al. [21] introduced a robust approach to minimize total setup cost in PMSP with sequence-dependent setups.. Then, a fuzzy mathematical programming model is formulated to provide optimal solution as a trade-off between total setup cost and the necessity degree. Petrovic and Duenas [22] provided a new fuzzy logic based on decision supporting system for PMS problem.

Raja et al. [25] utilized fuzzy logic to combine two goals in reducing E&T penalties. Gharehgozli et al. [26] formulated a goal programming model for PMS problem with release dates and SD setup times considering fuzzy goals and fuzzy processing time. The objectives are to minimize the total weighted tardiness and the total weighted flow time.

Balin [19] studied PMS problems considering fuzzy processing times. They improved a robust GA to minimize the makespan. Chyu and Chang [27] considered UNPMS problem with two fuzzy goals, namely makespan and average tardiness. They developed and a greedy randomized adaptive search and two SA to solve the problem. Alcan and Başlıgil [28] developed a GA with fuzzy processing times to minimize total completion time on UNPMS problem.

Torabi et al. [29] surveyed multi-objective UNPMS problem with release time and SD setup times considering fuzzy processing times and due dates. They presented a particle swarm optimization approach to obtain a near optimum solutions of Pareto frontier to minimize the total weighted flow time, total weighted tardiness, and total machine load variation. Yeh et al. [30] investigated PMS problem with learning effects and fuzzy processing times to minimize the makespan. They developed a simulated annealing algorithm and a GA for the problem. Rostami et al. [31] surveyed multi-objective UNPMS problem with learning effects and deterioration under fuzzy environment to minimize the total E&T and makespan. In order to gain a Pareto optimal front, they introduced a nonlinear mathematical formulation based on the fuzzy chance-constrained programming. Liao and Su [32] presented a hybrid ant colony optimization for fuzzy PMS problem. Manupati et al. [33] prepared a fuzzy mathematical formulation and a new based evolutionary artificial immune non-dominated sorting GA for multi-objective UNPMS problem with release dates, machine and SD setup times. Ahmet Arık Toksarı[45] investigated a multi-objective parallel machine scheduling problem under a fuzzy environment with fuzzy job deterioration effect, fuzzy learning effect and fuzzy processing times to minimize total tardiness penalty cost, to minimize earliness penalty cost and to minimize cost of setting due dates. They compared different approaches for modelling fuzzy mathematical programming models with a local search algorithm based on expected values of fuzzy parameters such as job deterioration effect, learning effect and processing times. Nailwal et el. [46] considered a bi-criteria scheduling on parallel machines in fuzzy environment which optimizes the weighted flow time and maximum tardiness. The processing time of jobs is represented by triangular fuzzy membership function. They provided a heuristic algorithm to find the optimal sequence of jobs processing on parallel machines. Naderi-Beni et al. [47] addressed a fuzzy bi-objective mixed-integer linear programming model for unrelated parallel machine scheduling with minimization workload imbalance and total tardiness, simultaneously. They considered sequence-dependent setup times, machine eligibility restrictions and release dates constraints. In their model, release dates, setup times and due dates are taken into account and modelled by fuzzy numbers. They applied a two-stage fuzzy approach to solve the model for small-scale problems. They presented two meta-heuristic algorithms, namely fuzzy multi-objective particle swarm optimization and fuzzy non-dominated sorting genetic algorithm (FNSGA-II) for solving large-scale instances.

To the best of our knowledge, this study is the first attempt dealing with fuzzy UNPMS problem with minimizing the sum of weighted E&T in which processing times, SD setup times, due dates, and objective function are affected by the fuzzy environment.

The reminder of our work is organized as follows. A review of the related fuzzy set theory is given in the Section 2. Section 3 represents a mathematical model for the regarded problem. In Section 4, two algorithms are introduced. In Section 5, the parameters of the algorithms are determined. Computational results are reported in Section 6. Finally, we concluded in Section 7.

2. Fuzzy set theory

2.1 Fuzzy numbers

This section illustrates the fuzzy numbers necessities which will be applied in next sections. Definitions and features are explained below.

A fuzzy number $\tilde{a}$ is a subset of real numbers determined by a function $\mu_{\tilde{a}} : R \rightarrow [0,1]$, named membership function of $\tilde{a}$. The α -level set of $\tilde{a}$, indicated by $\tilde{a}_{\alpha}$, is determined by $\tilde{a}_{\alpha} = \{x \in R : \mu_{\tilde{a}}(X) \geq \alpha\}$, for $\alpha \in [0,1]$. For a fuzzy number $\tilde{a}$ the α -level set of $\tilde{a}$ is a closed, bounded and convex subset of *R*, namely a closed interval in *R*. In this case, it is denoted by $\tilde{a}_{\alpha} = [\tilde{a}_{\alpha}^{L}, \tilde{a}_{\alpha}^{U}]$.

Proposition 2.1.1. Let $\tilde{a}$ and $\tilde{b}$ be two fuzzy numbers. Then, $\tilde{a} \oplus \tilde{b}$, $\tilde{a} \ominus \tilde{b}$ and $\tilde{a} \otimes \tilde{b}$ are also fuzzy numbers. Furthermore,

$$(\tilde{a} \oplus \tilde{b}) = [\tilde{a}_{\alpha}^{L} + \tilde{b}_{\alpha}^{L}, \tilde{a}_{\alpha}^{U} + \tilde{b}_{\alpha}^{u}],$$

$$(\tilde{a} \ominus \tilde{b})_{\alpha} = [\tilde{a}_{\alpha}^{L} - \tilde{b}_{\alpha}^{U}, \tilde{a}_{\alpha}^{U} - \tilde{b}_{\alpha}^{L}],$$

$$(\tilde{a} \otimes \tilde{b})_{\alpha} = [\min\{\tilde{a}_{\alpha}^{L}\tilde{b}_{\alpha}^{L}, \tilde{a}_{\alpha}^{L}\tilde{b}_{\alpha}^{U}, \tilde{a}_{\alpha}^{U}\tilde{b}_{\alpha}^{L}, \tilde{a}_{\alpha}^{U}\tilde{b}_{\alpha}^{U}\}, \max\{\tilde{a}_{\alpha}^{L}\tilde{b}_{\alpha}^{L}, \tilde{a}_{\alpha}^{L}\tilde{b}_{\alpha}^{U}, \tilde{a}_{\alpha}^{U}\tilde{b}_{\alpha}^{L}, \tilde{a}_{\alpha}^{U}\tilde{b}_{\alpha}^{U}\}] \quad .$$

In application of fuzzy theory, triangular and trapezoidal fuzzy numbers are utilized most frequently. A triangular fuzzy number $\tilde{a}$ is denoted by $\tilde{a} = (a^{L}, a, a^{U})$ and its membership function is defined by

$$\mu_{\tilde{a}}(x)=\begin{cases}\dfrac{x-a^L}{a-a^L}, \text{if } a^L \le x \le a,\\ \dfrac{a^U-x}{a^U-a}, \text{if } a \le x \le a^U,\\ 0, \quad \text{otherwise.}\end{cases} \tag{1}$$

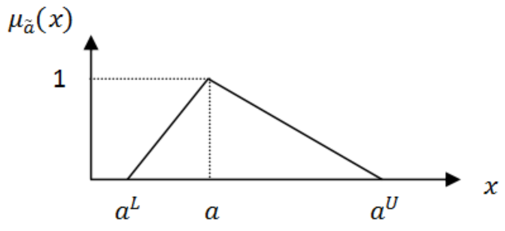

Figure 1. A triangular fuzzy number

A schematic view of this type of fuzzy number is depicted in Figure 1. The $\tilde{a}$ -level set (a closed interval) of $\tilde{a}$ is

$\tilde{a}_\alpha=[(1-\alpha)a^L+\alpha a,(1-\alpha)a^U+\alpha a]$,

where $\tilde{a}^L_\alpha=[(1-\alpha)a^L+\alpha a$ and $\tilde{a}^U_\alpha=(1-\alpha)a^U+\alpha a]$. (2)

It can be shown that $\tilde{a}\oplus\tilde{b}$, as shown below, is also a triangular fuzzy number:

$(\tilde{a}\oplus\tilde{b})_\alpha=(a^L,a,a^U)+(b^L,b,b^U)=(a^L+b^L,a+b,a^U+b^U)$. (3)

For a trapezoidal fuzzy number, denoted by $\tilde{a}=(a^L,a_1,a_2,a^U)$, the membership function is defined to be (see Figure 2)

$$\mu_{\tilde{a}}(x)=\begin{cases}\dfrac{x-a^L}{a-a^L}, \text{if } a^L \le x \le a_1,\\ 1, \quad \text{if } a_1 \le x \le a_2,\\ \dfrac{a^U-x}{a^U-a}, \text{if } a_2 \le x \le a^U,\\ 0, \quad \text{otherwise.}\end{cases} \tag{4}$$

It can be seen that

$\tilde{a}^L_\alpha=(1-\alpha)a^L+\alpha a_1$ and $\tilde{a}^U_\alpha=(1-\alpha)a^U+\alpha a_2$, (5)

where $\tilde{a}_\alpha=[\tilde{a}^L_\alpha, \tilde{a}^U_\alpha]$.

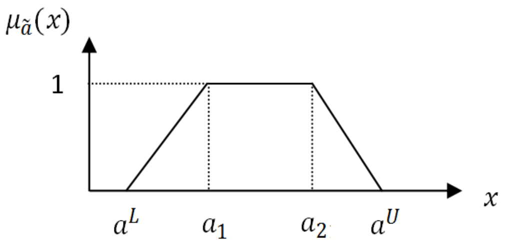

Figure 2. A trapezoidal fuzzy number

In this study, job processing times and setup times are regarded as triangular fuzzy numbers, and due dates are presumed to be trapezoidal fuzzy numbers. In a fuzzy scheduling problem, some data like setup times and processing times cannot be collected precisely in some situations. These values are generally modeled as triangular fuzzy numbers. The idea of fuzzy due date implies some pliability of job due date. The membership value of fuzzy due date represents the satisfaction degree of job completion. Trapezoidal fuzzy numbers are used to describe this notion of deadline values [34].

2.2 Fuzzy earliness and tardiness

In the considered fuzzy scheduling problem, processing times p_{jk}, setup times s_{ij} and due dates d_j are assumed to be fuzzy numbers, where *i* and *j* are indices of job and *k* is the index of machine. Therefore, from Proposition 2.1.1, the earliness $\tilde{E}_j$ and tardiness $\tilde{T}_j$ corresponding to job *j* are also fuzzy numbers and are defined to be

$$\tilde{E}_j = \tilde{\max}\{0, \tilde{d}^L_{j\alpha} \ominus \tilde{C}_j\} \text{and} \tilde{T}_j = \tilde{\max}\{0, \tilde{C}_j \ominus \tilde{d}_j\}.$$

Where C_j denotes the completion time of job *j*. Therefore, the α -level closed intervals of $\tilde{E}_j$ and $\tilde{T}_j$ can be obtained by

$$\tilde{E}^L_{j\alpha} = \max\{0, \tilde{d}^L_{j\alpha} \ominus \tilde{C}^U_{j\alpha}\} \text{and } \tilde{E}^L_{j\alpha} = \max\{0, \tilde{d}^U_{j\alpha} \ominus \tilde{C}^L_{j\alpha}\}, \tag{6}$$

$$\tilde{T}^L_{j\alpha} = \max\{0, \tilde{C}^L_{j\alpha} \ominus \tilde{d}^U_{j\alpha}\} \text{and } \tilde{T}^L_{j\alpha} = \max\{0, \tilde{C}^U_{j\alpha} \ominus \tilde{d}^L_{j\alpha}\}. \tag{7}$$

In this study, the goal is the sum of weighted fuzzy E&T penalties, given by

$$\tilde{f}(\pi) = \bigoplus_{j=1}^{n} (e_j \tilde{E}_j \oplus t_j \tilde{T}_j). \tag{8}$$

Where e_j and t_j are the E&T penalties per unit of time, respectively. A ranking method introduced by Fortemps and Roubens [34] is applied to discover an optimum schedule π^* to minimize the fuzzy objective function. In this method, for any two fuzzy numbers $\tilde{a}$ and $\tilde{b}$, $\tilde{a} \leq \tilde{b}$ if and only if $\eta(\tilde{a}) \leq \eta(\tilde{b})$, where $\eta(\tilde{a})$ is calculated by

$$\eta(\tilde{a}) = \tfrac{1}{2}\int_0^1 (\tilde{a}^L_\alpha + \tilde{a}^U_\alpha) d\alpha, \tag{9}$$

where

$$\tilde{f}_\alpha(\pi) = [\tilde{f}^L_\alpha(\pi), \tilde{f}^U_\alpha(\pi)] = [\Sigma^n_{j=1}(e_j \tilde{E}^L_{j\alpha} + t_j \tilde{T}^L_{j\alpha}), \Sigma^n_{j=1}(e_j \tilde{E}^U_{j\alpha} + t_j \tilde{T}^U_{j\alpha})]. \tag{10}$$

Substituting Eq. (10) in Eq. (9) we get

$$\eta\left(\tilde{f}(\pi)\right) = \tfrac{1}{2}\Sigma^n_{j=1}(e_j h_j + t_j u_j), \tag{11}$$

where

$$h_j = \int_0^1 \max\left\{0, \tilde{d}^L_{j\alpha} \ominus \tilde{C}^U_{j\alpha}\right\} d\alpha + \int_0^1 \max\left\{0, \tilde{d}^U_{j\alpha} \ominus \tilde{C}^L_{j\alpha}\right\} d\alpha, \tag{12}$$

$$u_j = \int_0^1 \max\left\{0, \tilde{C}^L_{j\alpha} \ominus \tilde{d}^U_{j\alpha}\right\} d\alpha + \int_0^1 \max\left\{0, \tilde{C}^U_{j\alpha} \ominus \tilde{d}^L_{j\alpha}\right\} d\alpha. \tag{13}$$

We compute $\eta(\tilde{f}(\pi))$ by considering the fuzzy processing times $\tilde{p}^L_{jk} = (p^L_{jk}, p_{jk}, p^U_{jk})$ and the fuzzy setup times $\tilde{s}_{ij} = (s^L_{ij}, s_{ij} s^U_{ij})$ to be triangular fuzzy numbers and the fuzzy due dates and $\tilde{d}_j = (d^L_j, d_{j1}, d_{j2} d^U_j)$ to be trapezoidal fuzzy numbers, for *j* = 1,2,...,*n* and *k* = 1,2,...,*m*. Therefore, it is seen that

$$\tilde{d}^L_{j\alpha} = (1-\alpha)d^L_j + \alpha d_{j1} \text{ and } \tilde{d}^U_{j\alpha} = (1-\alpha)d^U_j + \alpha d_{j2}. \tag{14}$$

The fuzzy completion time $\tilde{C}_j$ of each job *j* will be a triangular fuzzy number $\tilde{C}_j = (C^L_j, C_j, C^U_j)$ because of triangular shape of the processing times and the setup times. Therefore,

$$\tilde{C}^L_{j_\alpha} = (1-\alpha)C^L_j + \alpha C_j \text{ and } \tilde{C}^U_{j_\alpha} = (1-\alpha)C^U_j + \alpha C_j . \quad (15)$$

Now, regarding the graph of $\tilde{C}_j$ as a triangle and the graph of $\tilde{d}_j$ as a trapezoid, there are five cases explaining their positional relations, as given by (14). In other words, the following five cases are to be discussed in order to calculate the $e_j h_j$ and the $t_j u_j$ in (12) and (13).

Case (I): If $C^U_j \le d^L_j$, then the graph of $\tilde{C}_j$ is completely on the left side of the graph of $\tilde{d}_j$. In this case,

$$e_j h_j + t_j u_j = \tfrac{1}{2} e_j (d^L_j + d_{j1} + d_{j2} + d^U_j - C^L_j - 2C_j - C^U_j) . \quad (16)$$

Case (П): If $C_j \le d_{j1}$ and $C^U_j \ge d^L_j$, then there is an intersection of the left side of $\tilde{C}_j$ and the right side of $\tilde{d}_j$. The intersection α^* is a point such that

$$\tilde{C}^U_{j_{\alpha^*}} = \tilde{d}^L_{j_{\alpha^*}} \text{ or } (1-\alpha^*)C^U_j + \alpha^* C_j = (1-\alpha)d^L_j + \alpha d_{j1} ,$$

and thus,

$$\alpha^* = \frac{C^U_j - d^L_j}{C^U_j - C_j + d_{j1} - d^L_j} , \quad (17)$$

which implies

$$e_j h_j + t_j u_j = \frac{1}{2} e_j (d^L_j + d_{j1} + d_{j2} + d^U_j - C^L_j - 2C_j - C^U_j) + \frac{1}{2}(e_j + t_j) \frac{(C^U_j - d^L_j)^2}{C^U_j - C_j + d_{j1} - d^L_j} . \quad (18)$$

Case (III): If $d_{j1} \le C_j \le d_{j2}$ then, regardless of the intersection between $\tilde{C}_j$ and $\tilde{d}_j$, we have

$$e_j h_j + t_j u_j = \frac{1}{2} e_j (d^U_j + d_{j2} - C^L_j - C_j) + \frac{1}{2} t_j (C^U_j + C_j - d^L_j - d_{j1}) . \quad (19)$$

Case (IV): If $C_j \ge d_{j2}$ and $C^L_j \le d^U_j$ then there is an intersection of the right side of $\tilde{C}_j$ and the left side of $\tilde{d}_j$. The intersection α^* is a point such that

$$\tilde{C}^L_{j_{\alpha^*}} = \tilde{d}^U_{j_{\alpha^*}} \text{or, } (1\text{-}\alpha^*)C^L_j + \alpha^* C_j = (1\text{-}\alpha)d^U_j + \alpha d_{j2} ,$$

and thus,

$$\alpha^* = \frac{d^U_j - C^L_j}{C_j - C^L_j + d^U_j - d_{j2}} . \quad (20)$$

In this case, we obtain

$$e_j h_j + t_j u_j = \frac{1}{2} t_j (C^L_j + 2C_j + C^U_j - d^L_j - d_{j1} - d_{j2} - d^U_j) + \frac{1}{2}(e_j + t_j) \frac{(d^U_j - C^L_j)^2}{C_j - C^L_j + d^U_j - d_{j2}} . \quad (21)$$

Case (V): If $d^U_j \le C^L_j$ then the graph of $\tilde{d}_j$ is totally on the left side of the graph of $\tilde{C}_j$, which gives

$$e_j h_j + t_j u_j = \frac{1}{2} t_j (C^L_j + 2C_j + C^U_j - d^L_j - d_{j1} - d_{j2} - d^U_j) . \quad (22)$$

For each job j, one of the above five cases, i.e. (I)–(V), will be occurred, and the value $e_j h_j + t_j u_j$ can be obtained. Therefore, for any given schedule π, the defuzzified objective value is computed by Eq. (11).

3. Mathematical model

3.1 Problem description

The considered problem can be stated as follows: There are n jobs that are ready to be processed at time zero on m unrelated parallel machines, and only one machine is required to complete each job without interruption. Setup times are job SD. Job processing time, setup times and due dates are given. It is presumed that the setup times and processing times are triangular fuzzy numbers and due dates are trapezoidal ones. For each job in a given sequence, the fuzzy completion time, fuzzy E&T are computed by related fuzzy operations. Here, the objective is minimizing the sum of weighted fuzzy E&T penalties of all scheduled jobs.

3.2 Notations

Indices		Parameters	
i, j	indices for jobs ($i, j = 1,\ldots,n$)	$\tilde{p}_{ik}$	fuzzy processing time of job i on machine k
K	index for machines ($k = 1,\ldots,m$)	$\tilde{s}_{ij}$	fuzzy setup time of job j when it follows job i
		$\tilde{d}_i$	fuzzy due date of job i
		e_i	earliness weight of job i
		t_i	tardiness weight of job i
		$\tilde{R}$	a large positive triangular fuzzy number

3.3 Decision variables

X_{ik}	1 if job i is processed on machine k; 0, otherwise
Y_{ijk}	1 if job i precedes job j on machine k; 0, otherwise
$\tilde{C}_i$	Fuzzy completion time of job i
$\tilde{E}_i$	Fuzzy earliness of job i
$\tilde{T}_i$	Fuzzy tardiness of job i

With these definitions and notations, the proposed model can be formulated as follows:

$$\min \Sigma_{j=1}^{n} (e_j \tilde{E}_j \oplus t_j \tilde{T}_j) \tag{23}$$

s.t.

$$\Sigma_{k=1}^{m} X_{ik} = 1, \qquad i = 1,\ldots,n \tag{24}$$

$$\Sigma_{j=1}^{n} Y_{ijk} \le X_{ik} \qquad j \ne i \quad , \quad i = 1,\ldots,n \quad , \quad k = 1,\ldots,m \tag{25}$$

$$\Sigma_{i=0}^{n} Y_{ijk} = X_{jk} \qquad i \ne j \quad , \quad i = 1,\ldots,n \quad , \quad k = 1,\ldots,m \tag{26}$$

$$\tilde{C}_j \ominus \tilde{C}_i \oplus \tilde{R}(1\text{-}Y_{ijk}) \ge \tilde{p}_{jk} \oplus \tilde{s}_{ij} \quad j \ne i \,, i = 0,\ldots,n \,, j = 1,\ldots,n \,, k = 1,\ldots,m \tag{27}$$

$$\tilde{E}_j = \tilde{\max}(0, \tilde{d}_j \Theta \tilde{C}_j) \qquad j = 1,\ldots,n \tag{28}$$

$$\tilde{T}_j = \tilde{\max}(0, \tilde{C}_j \Theta \tilde{d}_j) \qquad j = 1,\ldots,n \tag{29}$$

$$X_{ik} \in \{0,1\}, \quad i = 1,\ldots,n, \quad k = 1,\ldots,m \tag{30}$$

$$Y_{ijk} \in \{0,1\}, \quad i = 0,...,n, \quad j = 1,...,n, \quad k = 1,...,m \tag{31}$$

The objective function in (23) is to minimize the total fuzzy E&T penalties of all jobs. Constraint (24) assures that each job is processed on precisely one machine. Constraints (25) and (26) guarantee that each job comes immediately before, and immediately after, another job. Constraint (27) satisfies that the fuzzy completion time of job *i* in a typical order on a machine must be equal or greater than the sum of the fuzzy completion time of the previous job, the fuzzy setup time and the fuzzy processing time of the current job. For each job, constraints (28) and (29) give the fuzzy E&T values, respectively. Finally, constraints (30) and (31) indicate that the variables X_{ik} and Y_{ijk} are binary. A typical test problem and optimal solution obtained by the proposed model are demonstrated in Section 6.

4. Proposed algorithms

Since the addressed problem is NP-Hard, exact methods cannot solve large and medium sizes in short times [35]. So, heuristic algorithms should be implemented.

4.1 Genetic algorithm

The GA, an optimization technique based heuristic procedures for scheduling problems were successful in many applications [36]. The whole process of this approach can be explained shortly as follows:

Chromosome representation: Each chromosome is a matrix with dimensions (2, n) which n denotes the number of jobs. A job permutation from 1 to n is placed in the first row and the second row is filled by the machine number corresponding to the job in the first row. If machine number of two jobs are identical, the job which appearing earlier is scheduled before the other job. This procedure ensures that each job is assigned to one machine and the sequencing of jobs is determined.

Generation of initial population: S chromosomes of solutions are considered as the initial population which S equivalent to population size.

Fitness function: The fitness value of each chromosome is calculated by

$$F(k) = \sum_{j=1}^{n} (e_j \tilde{E}_j + t_j \tilde{T}_j), \quad k = 1,...,S \tag{32}$$

where *F*(*k*) is the fitness value of the *k th* chromosome in the population and $\sum_{j=1}^{n} (e_j \tilde{E}_j + t_j \tilde{T}_j)$ is the objective function value of the problem.

Selection strategy: In selection strategies, it is important to prevent the algorithm to quickly converge to local minimum. Roulette-wheel approach applies a probabilistic function to pick chromosomes. The probability of selection a chromosome is relative to its fitness.

uses a probability distribution for selection in which the selection probability of a given string is proportional to its fitness. It is calculated by Eq. 33, where *Z*(*k*) shows the selection probability of *k th* chromosome.

$$Z(k) = \frac{\frac{1}{F_k}}{\sum_{j=1}^{S} \frac{1}{F_k}}, \quad k = 1,...,S \tag{33}$$

Local search: This procedure is widespread employed in GAs to make progress on the quality of solution [1]. Reallocating job on a tardier machine to a machine with fewer tardy may suitable to minimize total tardiness. Although, it is occasionally useful to reallocate job on a tardier machine to an intermediate machine and reallocate job on the intermediate machine to a machine with fewer tardy in an UNPMS environment [37]. Here, Chen's method [37] is modified as a local search procedure. In this procedure, one job on a machine with more weighted E&T is reallocated to a machine with fewer tardy. This job is chosen from the tardy jobs or from the jobs scheduled before any tardy job randomly. Afterward, a job with less tardy machine is selected and reassigned to a machine with more weighted E&T. A job is selected from less tardy machine randomly. Figure 3 presents a realization of the local search. The third machine has more weighted E&T and the tardy jobs on this machine are jobs 1 and 3. The second machine is a machine with fewer tardy. Therefore, one job should be chosen from the third machine and reallocated to the second machine. As shown in Figuure 3, job 2 is selected from the third machine randomly and reallocated to the second machine. During the local search, new solutions are produced and solutions with better qualities are accepted as new members of the current population. The probability of applying the local search is determined by a given local search rate P_s and chromosomes are selected from the population randomly.

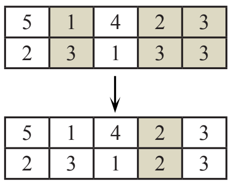

Figure 3. Job reallocation in a local search procedure

Crossover: Crossover operation is utilized to generate a new offspring from randomly selected pairs of parents by uniting together. The one-point crossover operator adjusted to the parallel machines is one of the most applied crossover operators. A point on both parents' chromosomes is selected randomly, and designated a 'crossover point'. Then, the tails of its two parents are swapped to get new off-springs. Utilizing this operator in initial sequences may make infeasible chromosomes. To have feasible chromosomes, each absent job is chosen randomly and relocated with a duplicated job and the required machine corresponding to each duplicated job is also chosen randomly.

Mutation: At first, this operator chooses randomly a job from the first row of the matrix and exchanges its equivalent machine number with another one randomly.

Reproduction: This process accepts a copy of chromosome to the next generation based on a probability related to its fitness. The tournament selection method is applied to make selection of parents.

Stopping criterion: The search process is terminated after generating a pre-specified number (G_{max}) of populations.

The genetic operators are depicted in Figure 4.

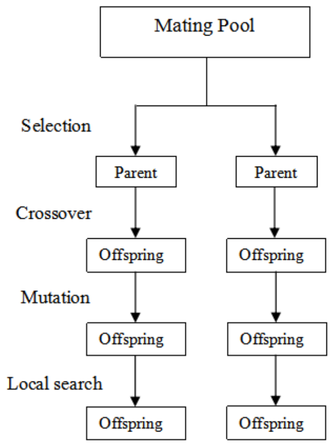

Figure 4. Genetic operators

4.2 Simulated annealing

SA is an optimization algorithm with random search strategies, which is a branch of stochastic optimization. The gradual cooling of metals in metallurgy inspires the algorithm. It begins with high temperature. The temperature is decreased by setting a scheduled annealing. At first, an initial solution is created by chance. Then, neighborhood solutions are produced in each temperature. Each solution with worse objective than the currently held solution is also accepted by probabilistic function in order to acquire the better result by escaping from the local optima. The search space for finding neighborhood solutions becomes small and small as the temperature decreases. The main components of the introduced SA for implementation are as follows.

Step 1:

Initial solution: a first solution is produced from the final population of GA randomly.

Step 2:

Neighborhood generation: A neighborhood solution is generated based on a heuristic perturbation of the current solutions. Here, a job swapping method is implemented. In this method, a job on a machine that has more weighted E&T value is exchange with a job on a machine with less one. This method is performed to produce neighborhood solutions at every iteration of SA. It should be noted that machines are categorized by their associated weighted E&T value. The total weighted E&T of each job is divided by the number of machines and then, machines having associated weighted E&T

values more than this fraction are grouped with a machine with a high value of associated weighted E&T.

Step 3:

Cost function: Calculate the objective values for the solutions; i.e., the total weighted fuzzy E&T.

Step 4:

Approval Odds: This process determines accepting or rejecting a generated solution based on its objective in comparison to the present solution. This process accepts the generated solution if its objective is smaller than the present solution; if not, there is still a chance to accept it because of evading of the local optimum. If its objective is equal or greater than the current solution, it is accepted based on $p = exp^{(\frac{-\Delta E}{T})}$, Where ΔE is measure to which neighbor solution becomes worse from the current solution, and T is a temperature parameter in the current iteration. A number from 0 to 1 is produced through a uniform distribution randomly. If p > rand (0, 1), then the answer is deteriorated. Otherwise, another neighborhood will be chosen.

Step 5:

Cooling rate: The temperature lessens slowly within the search operation by the following formula:

$$Temp_r = cr * Temp_{r-1}, \quad 0 \le cr < 1, r = 1, 2, ..., \tag{34}$$

where $Temp_r$ is the present temperature after the *r*th repetition and cr is a continuous number between 0 and 1 as the cooling rate.

Step 6:

The length of Markov chains: The SA algorithm includes of two loops: the inner loop and the outer loop. The inner loop finds different moves from the current solution and the outer loop implements the cooling schedule.

The inner loop or the number of moves generated for a fixed value at each temperature is denoted by *G*.

$$G = \gamma \times n \times m\,. \tag{36}$$

Where n indicate the number of jobs, m denote the number of machines, and γ is a certain coefficient.

The approved moves numbers at each temperature is a definite parameter [38], denoted by F and is determined by:

$$F = \beta \times n \times m\,. \tag{35}$$

Where β is a certain coefficient.

If generated moves at each temperature reaches its maximum (*G*), the inner loop is stop and SA carries on with the next outer loop step after declining its temperature.

Step 7:

Stopping criterion: The SA stops when the best fitness value does not change after a number of consecutive iterations (*Iter_max*).

5. Parameter setting

Algorithm calibration is an important process in the design of a meta-heuristic algorithm. Different values of parameters can affect computational time and solution quality. In order to determine the parameters values, here the Taguchi method is applied. Taguchi method explores the whole parameter space by utilizing a small number of trials through a particular design of levels. The Taguchi orthogonal arrays, response table, and the signal-to-noise (S/N) ratio are employed to determine suitable levels of parameters and to examine the effect of parameters on the goals. The goal is to maximize the S/N ratio. Objective functions are arranged into three groups based on the Taguchi approach: smaller-the-better, larger-the-better, and nominal-the-best. Here, we use the larger better type with its corresponding S/N ratio [39] which as follows:

$$S/N\ ratio = -10 \times \log_{10}(1/Objective\ fuction)^2. \tag{38}$$

The parameters of the proposed GA and modified SA (SA_Mod) and their considered levels are shown in tables 1 and 2, respectively.

Table 1. Considered levels for GA parameters

GA parameters	Considered levels
Pop_size	40-50-60
G_{Max}	70-80-90
P_C	0.5-0.6-0.7
P_M	0.08-0.10-0.12
P_S	0.05-0.10-0.15

Table 2. Considered levels for SA_Mod parameters

SA_Mod parameters	Considered levels
$Temp_0$	400-450-500
cr	0.95-0.97-0.99
β	0.05-0.10-0.15
γ	0.10-0.15-0.20
Iter_max	20-30-40

As shown in tables 1 and 2 problems have four parameters and three levels. In order to conduct the experiment, the introduced GA and SA_Mod are implemented five times for each selected problem and the reverse average of the objective function is used to compare their performances. The preferred level of GA and SA_Mod parameters are reported in tables 3 and 4, respectively.

Table 3. Selected level for GA parameters

GA parameters	Selected level
Pop_size	50
G_{Max}	70
P_C	0.6
P_M	0.12
P_S	0.05

Table 4. Selected level for SA_Mod parameters

SA_Mod parameters	Selected level
$Temp_0$	450
cr	0.99
β	0.05
γ	0.15
Iter_max	20

6. Computational results

The introduced GA and SA_Mod are assessed on a benchmark problem data set existing in the previous literature[37]. Chen [37] generated 960 instances based on the six factors which is indicated in Table 5. Table 5 shows six parameters which have 3, 4, 2, 2, 2, and 2 levels. Furthermore, five instances are created for each combination of the six parameters. To sum up, the instances bed equal to $3\times4\times2\times2\times2\times2 = 192$ and a total of $192\times5 = 960$ tests. The files of instances and the results of the suggested SA_HEU by Chen are achievable from Chen's web site [37].

Table 5. Settings of parameter based on Chen's problem data set

Machines (m)	4, 6, 8
Jobs (n)	30, 50, 70, 90
Job due date (τ)	0.4, 0.8
Range of job due date (R)	0.4, 1
Number of job families (F)	$[n/7]+1$, $[n/8]+1$
Job proportion from primary customers (P)	0.2, 0.3

6.1 Comparative performance of algorithms

The Proposed mathematical model is coded in Lingo software environment and a test problem with five jobs and three machines is solved. Tables 6 and 7 show the data for the given problem and the computational results are exhibited in table 8. Moreover, table 9 indicates the sequence of jobs on different machines for the given problem. The results contain the fuzzy completion times, the optimal case of overlapping between fuzzy completion times and due dates (see subsection 3.2), defuzzified value of the sum of weighted fuzzy E&T for each job, and finally, the optimal job sequence.

Table 6. Problem data (processing time, due date, E&T weights)

Job	$\tilde{P}_{j1}$	$\tilde{P}_{j2}$	$\tilde{P}_{j3}$	$\tilde{d}_j$	e_j	t_j
1	(90.4,96.2,98.0)	(100.8,97.4,91.6)	(85.1,82.0,80.3)	(53.4, 56.0,60.9,66.0)	0.53	0.17
2	(74.3,69.0,67.8)	(98.8,94.1,89.7)	(81.2,76.9,73.9)	(61.7,66.0,68.3,70.6)	0.69	0.75
3	(36.4,34.9,33.7)	(86.7,84.1,79.6)	(98.7,95.5,94.3)	(62.2,64.4,65.8,67.2)	0.91	0.15
4	(30.0,26.8,23.4)	(72.9,68.2,63.6)	(39.1,34.8,30.4)	(62.4,68.3,72.0,75.7)	0.08	0.44
5	(100.3,96.4,93.7)	(32.4,30.1,25.4)	(61.0,55.5,51.0)	(53.2,54.3,60.1,65.9)	0.77	0.82

Table 7. Problem data (SD setup times)

Job	1	2	3	4	5
1	(0,0,0)	(40.3,42.6,46.7)	(23.3,28.9,31.4)	(39.0,40.3,44.0)	(40.1,41.2,43.9)
2	(26.7,31.9,34.2)	(0,0,0)	(18.5,24.1,26.5)	(38.9,40.1,43.8)	(46.3,47.4.50.0)
3	(10.3,15.5,17.8)	(44.9,47.2,51.3)	(0,0,0)	(23.9,25.2,28.9)	(14.1,15.2,17.9)
4	(10.8,16.0,18.2)	(21.7,24.0,28.1)	(27.8,33.4,35.8)	(0,0,0)	(31.7,32.8,35.4)
5	(15.1,20.3,22.6)	(15.6,17.9,21.9)	(26.4,32.0,34.4)	(11.8,13.0,16.7)	(0,0,0)

Table 8. Obtained results from the given problem

Job	$\tilde{C}_j$	$\eta(e_j\tilde{E}_j + t_j\tilde{T}_j)$
1	(146.5,157.9,168.7)	32.7
2	(67.8,69.0,74.3)	6.6
3	(120.0,128.0,137.2)	19.3
4	(63.6,68.2,72.9)	6.3
5	(51.0,55.5,61.0)	11.2
total		76.1

Table 9. Optimal schedule of jobs

Machine	Scheduled jobs
1	2, 3
2	4
3	5, 1

The Proposed algorithms are implemented in MATLAB 7.1 software environment and the numerical experiments are executed by a computer with Intel Core TM i3 @ 2.10 GHz processor and 2 GB of RAM memory. A set of large problems are solved by the two presented algorithms. To generate the problem data, the followings are used: the processing times, setup times and due-dates are presumed to be triangular and Trapezoidal fuzzy numbers as follows:

$$\tilde{p}_{ik} = (p_{ik} - w_{ik}, p_{ik}, p_{ik} + w_{ik}), \tilde{s}_{ij} = (s_{ij} - w_{ij}, s_{ij}, s_{ij} + w_{ij}) ,$$

$$\tilde{d}_j = (d_{j2} - w_j - w'_j, d_{j2} - w_j, d_{j2} + w_j) ,$$

where p_{ik} and s_{ij} have the uniform distributions of U [10 100] and U [10 40], respectively, and the w_j, w'_j, w_{ik} and w_{ij} values are chosen from $U^{[1,6]}$ randomly. The d_{j2} values are generated by

$$U[SUMP(1\text{-}\tau\text{-}\tfrac{RD}{2}), SUMP(1\text{-}\tau\text{+}\tfrac{RD}{2})] ,$$

where τ is the due date priority factor (say, 0.6), RD is the due date range factor (say, 0.1) and

$SUMP = \sum_{i=1}^{n}\sum_{k=1}^{m}(p_{ik} + (\sum_{j=1}^{n} s_{ij})/n)/m^2$.

Table 10 shows the obtained computational results with different combinations of jobs and machine numbers: n = {25, 50, 75, 100}, m = {5, 10, 15}. The results prove that the introduced SA_Mod method outperforms the proposed GA in terms of solution quality and computational efficiency.

Table 10. Comparison of solution precisions and computational times of GA and SA_Mod

Problem number	Problem size (job*machine)	GA		SA_Mod		Solutions of GA improved by SA_Mod (%)	Solutions of SA_mod improved by GA (%)
		Solution	CPU time(s)	Solution	CPU time(s)		
P01	25*5	591.2	21.1	665.5	0.7	0	11.16
P02	25*10	510.6	19.3	558.7	1.2	0	8.6
P03	25*15	339.2	19.8	344.9	1.7	0	1.65
P04	50*5	2268.1	28.9	1825.4	3.9	24.25	0
P05	50*10	1291.0	27.1	875.1	4.6	47.53	0
P06	50*15	1187.0	28.2	813.7	7.7	45.88	0
P07	75*5	7445.8	39.1	4747.0	14.5	56.85	0
P08	75*10	2580.7	34.8	1624.9	9.9	58.82	0
P09	75*15	2206.8	35.0	1432.1	14.2	54.1	0
P10	100*5	8269.4	51.3	4770.8	20.9	73.333	0
P11	100*10	5574.9	41.1	2842.7	22.4	96.11	0
P12	100*15	4335.3	40.1	2140.1	30.9	102.57	0

Now, knowing that the performance of SA_Mod is better than GA, the SA_Mod is compared with other algorithms to demonstrate the effectiveness of the algorithm in solving the proposed problem. To this end, the performance of SA_Mod is compared to SA_HEU given by Chen [37]. The considered problem in our work is a special case of the problem addressed in [37]. However, Chen did not consider earliness and weight for tardiness. Therefore, if the earliness and weight for tardiness are set to zero our problem, and the parameters are not fuzzy, the considered problem turns to be the same as Chen's in [37]. The results which are gained by the two algorithms are presented in table 11.

Table 11. Comparison of average solution and computational times of the SA_Mod and SA_HEU

		SA_HEU		SA_Mod		Solutions of SA_HEU improved by SA_Mod (%)
		Solution	CPU time	Solution	CPU time	
n	30	1412.54	5.71	1346.78	12.50	4.88
	50	2986.67	12.26	2687.84	32.71	11.12
	70	6106.60	26.06	5306.94	54.27	15.07
	90	10106.76	47.36	8473.51	84.51	19.27
m	4	7941.49	35.63	6443.26	41.61	23.25
	6	4795.58	19.54	4312.7	46.77	11.2
	8	2759.10	13.38	2627.82	49.62	4.99
F	[N/7]+1	5166.52	22.64	4418.36	45.24	16.93
	[N/8]+1	5164.27	23.06	4512.07	46.76	14.45
τ	0.4	1.41	0.97	0.71	14.76	98.59
	0.8	10329.38	44.73	8917.98	77.24	15.83
R	0.4	9047.71	27.07	7955.37	50.38	13.73
	1	1283.08	18.63	972.88	41.61	31.89
Overall average		5161.62	22.85	4459.71	46.00	21.63

Moreover, the performance of SA_Mod, is compared to SA_HEU, the proposed algorithm by Chen [37], through the parameters like Min solution, Average solution, and Max solution for an extensive investigation of the effectiveness of the SA_Mod algorithm. The results are demonstrated in table 12.

Table 12. Comparison of average, maximum, minimum solutions of the SA_Mod and SA_HEU

		SA_HEU			SA_Mod		
		Min.	Avg.	Max.	Min.	Avg.	Max.
n	30	1404.25	1412.54	1419.19	1340.80	1346.78	1357.31
	50	2961.08	2986.67	3004.90	2666.86	2687.84	2726.27
	70	6040.07	6106.60	6140.02	5256.13	5306.94	5480.14
	90	10020.51	10106.76	10254.30	8403.93	8473.51	8620.62
m	4	7861.48	7941.49	7995.00	6382.78	6443.26	6591.58
	6	4737.57	4795.58	4835.69	4271.81	4312.7	4395.81
	8	2720.38	2759.10	2783.12	2596.19	2627.82	2713.77
F	[*n*/7]+1	5104.22	5166.52	5206.68	4369.15	4418.36	4571.43
	[*n*/8]+1	5108.74	5164.27	5202.53	4464.7	4512.07	4633.17
τ	0.4	1.30	1.41	1.52	0.66	0.71	0.87
	0.8	10211.66	10329.38	10407.69	8833.19	8917.98	9143.69
R	0.4	8972.78	9047.71	6100.69	7891.12	7955.37	8168.25
	1	1240.18	1283.08	1308.52	942.73	972.88	1057.66
Overall average		5106.48	5161.62	4973.83	4416.93	4459.71	4573.9

As shown in table 12, the SA_Mod has produced more proper values of Min., Avg. and Max. solution than the ones obtained by SA_HEU.

6.2 Performance evaluation

In order to measure the performance of the introduced algorithms, relative percentage deviation (RPD) is calculated for the problems and the results obtained by our SA_Mod and SA_HEU are compared. The RPD value is computed as follows:

$$RPD = \left(\frac{Average_{Sol} - Best_{Sol}}{Best_{Sol}}\right)\times 100, \tag{38}$$

where $Average_{Sol}$ is the average solutions achieved by the algorithm which runs 10 times and $Best_{Sol}$ is the best known solution for each problem. The RPD results are listed in table 13.

Table 13. RPD results for algorithms

		SA_HEU	SA_Mod
		RPD	RPD
n	30	0.59	0.45
	50	0.86	0.79
	70	1.10	0.97
	90	0.86	0.83
m	4	1.02	0.95
	6	1.22	0.96
	8	1.42	1.21
F	[n/7]+1	1.22	1.12
	[n/8]+1	1.09	1.06
τ	0.4	8.46	7.58
	0.8	1.15	0.96
R	0.4	0.83	0.81
	1	3.46	3.20
Overall average		1.79	1.61

After computing the RPDs, in order to make precise comparisons between SA_Mod and SA_HEU, RPD means plot with least significant difference (LSD) intervals at a 95% confidence level from the analysis of variance (ANOVA) for total tardiness are examined and depicted in Fig. 5. It is clear that the SA_Mod shows a better performance than SA_HEU.

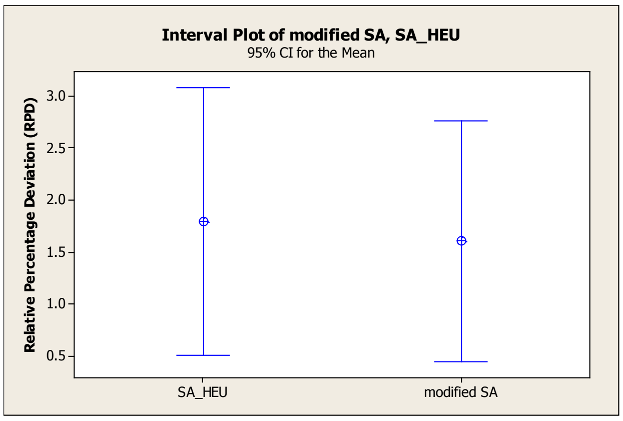

Figure 5. RPD means plot with LSD intervals at a 95% confidence level for total tardiness

7. Conclusions

We presented unrelated parallel machine scheduling (UNPMS) problem to minimize the total weighted fuzzy Earliness and Tardiness (E&T) considering fuzzy processing times, setup times and due dates. A mathematical model is proposed for the problem. Two algorithms, a genetic algorithm (GA) and a modified simulated annealing (SA) are also provided to solve large problems. The Taguchi method is used to set the values of the parameters of the two algorithms. The introduced GA and SA_Mod are assessed on a benchmark problem data set existing in the literature [37]. In 9 cases out 12, the SA_Mod obtained far better solutions with less computational times than the GA. Since the SA_Mod outperforms the GA in both qualities of solution and computational time, the performance of SA_Mod is compared to the proposed method by Chen [37] under 960 instances based on the six factors which is generated by him. The SA_Mod improved the obtained solution by SA_HEU ranging 4.9% to 98.6%. In other words, SA_Mod achieve better solutions with average of 21.6% rather than the SA_HEU. Furthermore, the overall average of RPD proves the solutions' quality of the SA_Mod against the SA_HEU. So, Various comparative results showed a better performance of the proposed SA_Mod algorithm.

As a future research, other meta-heuristic and hybrid algorithms can be considered for solving the problem. Also, other restrictions can be considered for the problem that is more consistent with the manufacturing environment.

References

[1] E. Vallada, R. Ruiz. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research.* 2011; 211: 612-622.

[2] K.C. Ying, S.W. Lin. Unrelated parallel machine scheduling with sequence and machine-dependent setup times and due date constraints. *International Journal of Innovative Computing, Information and Control.* 2012; 8: 3279-3297.

[3] A. Allahverdi, J.N.D. Gupta, T. Aldowaisan. A review of scheduling research involving setup considerations, OMEGA. *The International Journal of Management Science.* 1999; 27: 219-239.

[4] J. Rezaeian, K. Shokoufi, S. Haghayegh, *et al*. Designing an integrated production/distribution and inventory planning model of fixed-life perishable products. *Journal of Optimization in Industrial Engineering*. 2016; 9(19): 47-59.

[5] J.H. Lee, J.M. Yu, D.H. Lee. A tabu search algorithm for unrelated parallel machine scheduling with sequence and machine-dependent setups: minimizing total tardiness. *The International Journal of Advanced Manufacturing Technology.* 2013; 69: 2081-2089.

[6] M.N. Haddad, L.P. Cota, M.J.F. Souza, *et al*. Solving the unrelated parallel machine scheduling problem with setup times by efficient algorithms based on iterated local search. *International Conference on Enterprise Information Systems.* 2014; 227: 131-148.

[7] S.W. Lin, K-Ch.Ying. ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times. *Computers & Operations Research*. 2014; 51: 172-181.

[8] E, Caniyilmaz, B, Benli, M-S. Ilkay. An artificial bee colony algorithm approach for unrelated parallel machine scheduling with processing set restrictions, job sequence-dependent setup times, and due date. *The International Journal of Advanced Manufacturing Technology.* 2015; 77: 2105-2115.

[9] O. Avalos-Rosales, F. Angel-Bello, A. Alvarez. Efficient metaheuristic algorithm and re-formulations for the unrelated parallel machine scheduling problem with sequence and machine-dependent setup times. *The International Journal of*

Advanced Manufacturing Technology. 2015; 76: 1705-1718.

[10] F. Alvelos, M. Lopes, H. Lopes. A matheuristic based on column generation for parallel machine scheduling with sequence dependent setup times. In: Fonseca R., Weber GW., Telhada J. (eds.) *Computational Management Science*. 2016.

[11] J. Rezaeian, S. Mohammad Hosseini. Scheduling unrelated parallel machines with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology.* 2015; 81: 1487-1496.

[12] J. Rezaeian, M. Zarei, K. Shokoufi. Pareto-based multi-criteria evolutionary algorithm for a parallel machines scheduling problem with sequence-dependent setup Times. 2017; 30: 1863-1869.

[13] R. Gedik, D. Kalathia, G. Egilmez, *et al*. A constraint programming approach for solving unrelated parallel machine scheduling problem. *Computers & Industrial Engineering.* 2018; 121: 139-149.

[14] K. Shokoufi, J. Rezaeian, B. Shirazi, *et al*. Preemptive just-in-time scheduling problem on uniform parallel machines with time-dependent learning effect and release dates. *International Journal of Operational Research.* 2019; 3: 339-368.

[15] H-J. Kim. Bounds for parallel machine scheduling with predefined parts of jobs and setup time. *Annals of Operations Research.* 2018; 261: 401-412.

[16] C.M. Joo, B.S. Kim. Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability. *Computers & Industrial Engineering*. 2015; 85: 102-109.

[17] S. Baez, F. Angel-Bello, A. Alvarez. Time-dependent formulations for minimizing total completion time in a parallel machine scheduling problem with dependent setup times. *IFAC-PapersOnLine*. 2016; 49: 857-862.

[18] R, Gedik, D. Kalathia, G. Egilmez, *et al*. A constraint programming approach for solving unrelated parallel machine scheduling problem. *Computers & Industrial Engineering*. 2018; 121: 139-149.

[19] S. Balin. Parallel machine scheduling with fuzzy processing times using a robust genetic algorithm and simulation. *Information Sciences.* 2011; 181: 3551-3569.

[20] J. Balasubramanian, I.E. Grossmann. Scheduling optimization under uncertainty an alternative approach. *Computers and Chemical Engineering.* 2003; 27: 469-490.

[21] A. Anglani, A. Grieco, E. Guerriero, *et al*. Robust scheduling of parallel machines with sequence dependent setup costs. *European Journal of Operational Research*. 2005; 161: 704-720.

[22] D. Petrovic, D. Alejandra. A fuzzy logic based production scheduling/rescheduling in the presence of uncertain disruptions. *Fuzzy Sets Systems.* 2006; 157: 2273-2285.

[23] Y. Yi, D.W. Wang. Soft computing for scheduling with batch setup times and earliness-tardiness penalties on parallel machines. *Journal of Intelligent Manufacturing*. 2003; 14: 311-322.

[24] J. Peng, B. Liu. Parallel machine scheduling models with fuzzy processing times. *Information Sciences.* 2004; 166: 49-66.

[25] K. Raja, C. Arumugam, V. Selladurai. Non-identical parallel-machine scheduling using genetic algorithm and fuzzy logic approach. *International Journal of Services and Operations Management.* 2008; 1: 333-346.

[26] 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.* 2009; 25: 853-859.

[27] C.C. Chyu, W.S. Chang. Optimizing fuzzy makespan and tardiness for unrelated parallel machine scheduling with archived metaheuristics. *The International Journal of Advanced Manufacturing Technology.* 2011; 57: 763-776.

[28] P. Alcan, H. Başlıgil. A genetic algorithm application using fuzzy processing times in non-identical parallel machine scheduling problem. *Advances in Engineering Software.* 2012; 45: 272-280.

[29] S.A.Torabi, N.Sahebjamnia, S.A. Mansouri, *et al*. A particle swarm optimization for a fuzzy multi-objective unrelated parallel machines scheduling problem. *Applied Soft Computing.* 2013; 13: 4750-4762.

[30] W.C. Yeh, P.J. Lai, W.C. Lee, *et al*. Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects. *Information Sciences.* 2014; 269: 142-158.

[31] M. Rostami, A. Ebrahimzadeh Pilerood, M. Mahdavi Mazdeh. Multi-objective parallel machine scheduling problem with job deterioration and learning effect under fuzzy environment. *Computers & Industrial Engineering.* 2015; 85: 206-215.

[32] T.W. Liao, P. Su. Parallel machine scheduling in fuzzy environment with hybrid ant colony optimization including a comparison of fuzzy number ranking methods in consideration of spread of fuzziness. *Applied Soft Computing.* 2017; 56: 65-81.

[33] V.K. Manupati, G. Rajyalakshmi, F.T.S. Chan, *et al*. A hybrid multi-objective evolutionary algorithm approach for handling sequence and machine-dependent set-up times in unrelated parallel machine scheduling problem. *Sādhanā*. 2017; 42: 391-403.

[34] P. Fortemps, M. Roubens. Ranking and defuzzification methods based on area compensation. *Fuzzy Sets and Systems*.

1996; 82: 319-330.

[35] J. Rezaeian, K. Shokoufi, S. Poursafary. Developing a permutation method using tabu search algorithm: A case study of ranking some countries of west asia and north africa based on important development criteria. *Journal of Optimization in Industrial Engineering*. 2015; 8 (17): 21-30.

[36] M. Mitchell. *An Introduction to Genetic Algorithms*. London: MIT Publishing; 1999.

[37] J.F. Chen. Scheduling on unrelated parallel machines with sequence and machine-dependent setup times and due-date constraints. *International Journal of Advance Manufacturing.* 2009; 44: 1204-1212.

[38] E. Figielska. A genetic algorithm and a simulated annealing algorithm combined with column generation technique for solving the problem of scheduling in the hybrid flow-shop with additional resources. *Computers & Industrial Engineering.* 2009; 56: 142-151.

[39] M.S. Phadke. *Quality Engineering Using Robust Design*. Prentice-Hall; 1989.

[40] M. S. Gharajeh. Behavior-based decision making: A tutorial. *International Journal of Dynamics and Control.* 2018; 6: 1816-1840.

[41] M. S. Gharajeh. Applications of virtualization technology in grid systems and cloud servers. In: P. K. Das, G. C. Deka. (eds.) Hershey, PA: IGI Global; 2017: 1-28.

[42] Abualigah, L. M. Q. Feature selection and enhanced krill herd algorithm for text document clustering. *Studies in Computational Intelligence*. 2019.

[43] Abualigah, L. M. Q., Hanandeh, E. S. Applying genetic algorithms to information retrieval using vector space model. *International Journal of Computer Science, Engineering and Applications*. 2015; 5: 19.

[44] Soleimani, H., Ghaderi, H., Zarbakhshnia, N., *et al*. Scheduling of unrelated parallel machines considering sequence-related setup time, start time-dependent deterioration, position-dependent learning and power consumption minimization. *Journal of Cleaner Production*. 2020; 249.

[45] Ahmet Arık, O., Toksarı M. D. Multi objective fuzzy parallel machine scheduling problems under fuzzy job deterioration and learning effects. *International Journal of Production Research*. 2018; 56: 2488-2505.

[46] Nailwal, K. K., Gupta, D., *et al*. Fuzzy bi-criteria scheduling on parallel machines involving weighted flow time and maximum tardiness. *Cogent Mathematics*. 2015; 2.

[47] Naderi-Beni, M., Ghobadian, E., Ebrahimnejad, S., *et al*. Fuzzy bi-objective formulation for a parallel machine scheduling problem with machine eligibility restrictions and sequence-dependent setup times. *International Journal of Production Research*. 2014; 52: 5799-5822.