Solving multi-objective flexible flow-shop scheduling problem using teaching-learning-based optimization embedded with maximum deviation theory

Raviteja Buddala*
School of Mechanical Engineering
Vellore Institute of Technology, Vellore 632 014 India
Email: raviteja317@gmail.com

Siba Sankar Mahapatra
Department of Mechanical Engineering
National Institute of Technology
Rourkela 769008 India
Email: mahapatrass2003@gmail.com

Manas Ranjan Singh
Silicon Institute of Technology Bhubaneswar 751024 India
Email: manasranjan.singh@gmail.com

*Corresponding author

Abstract

Flexible flow-shop scheduling problem (FFSP) is an extended special case of basic flow-shop scheduling problem (FSP). FFSP is treated as complex NP-hard scheduling problem. A good scheduling practice enables the manufacturer to compete effectively in the market place. An efficient schedule should address multiple conflicting objectives so that customer satisfaction can be improved. In this work, a novel approach based on teaching-learning-based optimization (TLBO) technique incorporated with maximum deviation theory (MDT) is applied to generate schedules that simultaneously optimize conflicting objective measures like makespan and flowtime. Results indicate that the proposed multi-objective TLBO (MOTLBO) outperforms non-dominated sorting genetic algorithm II (NSGA-II) and multi-objective particle swarm optimization (MOPSO) in majority of the problem instances.

Keywords: Flexible flow-shop scheduling; Flowtime; Makespan; Maximum deviation theory; Non-dominated solutions; Multi-objective optimization; Teaching-learning-based optimization

1 Introduction

Flexible flow-shop scheduling problem (FFSP) possesses an additional complexity of assignment of jobs to available parallel machines as compared to basic flow-shop scheduling problem (FSP). Duplication of machines at each stage in a FFSP helps in eliminating bottleneck conditions, reducing the makespan, increasing the capacity of shop floor and so on. In fact, FFSP is an extension of basic FSP (Singh and Mahapatra, 2012). A large body of research has gone into proposing various methodologies for solving FFSP with an objective to minimize makespan (Buddala and Mahapatra, 2018; Mishra and Shrivastava, 2018). Apart from makespan, many performance measures of scheduling need to be satisfied not only to generate a practical schedule but also improve the customer perception. This helps the manufacturing industries to survive in the market competition and also maintain goodwill with the customers. From managerial point of view, throughput (defined as the amount of work done per unit time) is an important performance measure of scheduling. However, makespan minimization causes increase of throughput in a shop floor. It also ensures maximum utilization of the machines. Flowtime is an important performance measure from both operator as well as business point of view. It is defined as the amount of time that a job spends in a shop floor. Mean flowtime is defined as the average time that a job spends in a shop floor (Baker and Trietsch, 2013). Minimizing makespan or increasing throughput causes increase in the waiting time of jobs in a shop floor resulting in

mean flowtime to increase. Increase in flowtime also leads to increase the risk of tardiness (delivering the jobs beyond the due dates). Hence, flowtime can be viewed as a contradictory objective with respect to makespan. Therefore, in the present work, multi-objective optimization of contradictory objectives such as makespan and mean flowtime are chosen for the study. For a multi-objective scheduling problem, the problem solving strategy may be a weighted sum approach aggregating both the objectives into an equivalent single objective. However, determination of appropriate weights for individual objectives at priori is a challenging task and mostly depends on subjective or intuitive approach of the decision maker. In the past, a good number of studies have adopted weighted sum approach to solve different multi-objective scheduling problems (Xia and Wu ,2005; Tay and Ho,2008; Li et al., 2010). In order to overcome the drawback of weighted sum approach, the present study focuses on pareto-based method for solving multi-objective FFSP motivated from the works of various researchers (Kacem et al., 2002; Lei ,2008; Singh et al., 2016). In the present work, a novel multi-objective teaching-learning-based optimization (MOTLBO) is proposed to obtain a set of non-dominated solutions. It helps the decision maker to select any one solution among the set of possible solutions. However,, maximum deviation theory proposed by Wang (Yingming, 1997) is used in order to rank the non-dominated solutions so that the decision making process becomes easy. Reddy et al. (2018) have recently used MOTLBO to solve flexible job-shop scheduling problem (FJSP).

The current paper presents a novel approach based on teaching-learning-based optimization (TLBO) technique for solving multi-objective flexible flow shop scheduling problem with the goal of finding approximations of the optimal Pareto front. The Pareto-optimal solutions obtained through multi-objective TLBO (MOTLBO) have been ranked by the composite scores obtained through maximum deviation theory (MDT) to avoid subjectiveness and impreciseness in the decision-making. The instances from literature are solved and results are compared with non-dominated sorting genetic algorithm-II (NSGA-II) and particle swarm optimization (MOPSO) in terms of performance metrics.

2 Literature review

Extensive survey of scheduling literature reveals that makespan is the main objective, mostly gained huge attention in the past, in solving scheduling problems. In the pursuit of solving FFSP using traditional as well as meta-heuristic approaches, mostly a single objective performance

measure like makespan is widely adopted (Dios et al., 2018). For instance, Carlier and Neron (2000) have proposed a solution methodology based on branch and bound algorithm. Solutions based on exact methods breaks down when the problem size increases. Therefore, a large number of meta-heuristics have been adopted to solve FFSP in order to achieve quality solutions for large scale problem in finite computational time. Some of the important meta-heuristic approaches used to solve FFSP are artificial immune system (Engin and Doyen, 2004), genetic algorithm (Kahraman, Engin et al., 2008), quantum immune algorithm (Niu et al., 2009), particle swarm optimization (Liao et al. 2012), improved cuckoo search algorithm (Marichelvam et al., 2014), improved version of discrete artificial bee colony (Cui and Gu ,2015), harmony search algorithm (Marichelvam and Geetha, 2016) and JAYA and teaching-learning-based optimization (Buddala and Mahapatra, 2018). However, sufficient attention is not paid to consider other performance measures like flowtime, tardiness and sequence dependent setup times while solving FFSP. Guinet and Solomon (1996) have applied heuristic techniques to minimize makespan as well as tardiness. Botta-Genoulaz (2000) has also adopted a heuristic approach to minimize tardiness. In order to minimize total tardiness, Lee and Kim (2004) have applied branch and bound algorithm to solve FFSP. Nishi et al. (2013) have used Lagranginan with cut generation technique to minimize the total weighted tardiness. Behnamian and Zandieh (2011) have applied a discrete colonial competitive algorithm to minimize the penalty costs that occur due to tardiness. Recently, few studies have focused to tackle the sequence dependent setup times of FFSP (Kia et al., 2010; Maleki-Darounkolaei et al., 2012; Kia, et al. 2017). Azizoglu et al. (2001) have applied branch and bound algorithm to minimize total flowtime. Marichelvam and Prabaharan (2012) have used bat algorithm to minimize both makespan and mean flowtime. Pan and Dong (2014) have proposed an improved version of migrating birds optimization technique to minimize total flowtime.

Nevertheless, critical analysis of scheduling literature suggests that limited attempt has been made to solve multi-objective optimization in FFSP. Cho et al (2017) have proposed a two level method for scheduling bi-objective FFSP. Marichelvam et al. (2014) have solved the multi-objective optimization problem of makespan and mean flowtime with weighted sum method using discrete firefly algorithm. Wang and Liu (2014) have suggested a two stage solution procedure for multi-objective FFSP problem. Huang et al. (2015) have proposed a modified particle swarm optimization (PSO) algorithm called subgroup PSO technique to optimize

makespan and tardiness as the multi-objective performance measures. Considering same performance measures, Tran and Ng (2013) have proposed a hybrid water flow algorithm. Shahvari and Logendran (2016) have proposed a tabu search based algorithm to solve the multi-objective optimization of total tardiness and makespan using weighted sum method. Recently, Li et al. (2018) have proposed an algorithm called energy-aware multi-objective optimization algorithm (EA-MOA) to optimize makespan and energy consumption using pareto based method. It is clear from the literature that less attempt has been made to solve multi-objective optimization problem of FFSP using pareto-based methods. Research related to solving multi-objective FFSP using pareto-based optimization method considering makespan and mean flowtime as objectives is extremely low. The present study attempts to address the literature gap by focussing on multi-objective FFSP using pareto based method. For more information, a study on critical review of multi-objective optimization methods used for FFSP with relative merits and demerits can be referred (Sun et al., 2011).

On the other hand, teaching-learning-based optimization (TLBO), proposed by Rao et al. (2011), has been applied to several constrained and unconstrained optimization problems. The major advantage of using the TLBO algorithm is that it does not possess any tuning parameter. Therefore, setting the tuning parameter for a given problem, a time consuming process, can be minimized. Recently, TLBO has been applied to solve some of the scheduling problems. Xie et al. (2014) have applied TLBO to permutation flow shop scheduling problems. Keesari and Rao (2014) have applied TLBO to job-shop scheduling problems. Xu et al. (2015) have adopted TLBO to solve flexible job-shop scheduling problems. Buddala and Mahapatra (2016, 2018) have used TLBO to solve flexible flow-shop scheduling problems. With an inspiration from the recent success of TLBO in solving scheduling problems, the present work proposes TLBO to solve multi-objective FFSP.

3 Flexible flow-shop scheduling

The FFSP is considered as a special case of flow-shop scheduling problem (FSP). FFSP has an additional complexity called assignment of jobs to available parallel machines as compared to that of basic flow-shop scheduling problem FSP. FFSP is considered as NP-hard problem (Gupta 1988). A FFSP consists of J jobs that are to be processed at g number of operation centres called stages. Each job j (j=1, 2, …., J) must be processed at each stage t (t= 1, 2, …., g). Each stage may have more than one parallel machine. All parallel machines in a stage require equal amount

of time to execute an operation of a job. At any given instant, a job can be processed only on one machine and vice versa. The processing time of a job *j* at a given stage *t* denoted as *p (j, t)* is known in advance and deterministic. All jobs are different from each other. An operation cannot be interrupted till it is completed once it starts on a machine (pre-emption condition). The objective is to find the best permutation of jobs at each stage so that the objectives of scheduling are optimized.

3.1 Problem representation

In the present work, a real number encoding system proposed by Niu et al. (2009) is adopted to solve multi-objective FFSP. In order to allot each job to a machine, the integer part is used and the fractional part is used to sequence the jobs allotted to each machine. Let us consider a FFSP example problem with four jobs and three stages. Also, let us consider that stage one has one machine, stage two has two machines and stage three has three machines (m_1=1, m_2=2 and m_3=3). We generate twelve (4×3=12) number of random real numbers using uniform distribution between [*1, 1+m(k)*] where *k* is number of parallel machines in a particular stage. This is represented in the second row of table 1. The processing times are

$$p(j,t) = \begin{pmatrix} 2 & 1 & 3 \\ 1 & 2 & 1 \\ 3 & 3 & 4 \\ 2 & 1 & 2 \end{pmatrix}$$

In row one, for example, "*g1 j2*" means job 2 at stage 1. The assignment of machines at each stage is given at the third row of table 1 (according to the integer value of real number). Sequencing of jobs assigned to same machine is done according to the ascending order of fractional values. For example, all jobs are assigned to machine 1, in first stage as there is only one machine. Now, the sequencing of jobs assigned to machine 1 is done according to the increasing order of fractional values. The obtained sequence is j3>j2>j1>j4. This is because, ascending order of fractional values for the jobs at stage one is 0.26 (j3) <0.37 (j2) <0.53 (j1) <0.84 (j4).

Table1 Problem mapping representation

g1 j1	*g1 j2*	*g1 j3*	*g1 j4*	*g2 j1*	*g2 j2*	*g2 j3*	*g2 j4*	*g3 j1*	*g3 j2*	*g3 j3*	*g3 j4*
1.53	1.37	1.26	1.84	2.86	1.24	2.47	1.68	2.19	3.23	2.58	1.75
1	1	1	1	2	1	2	1	2	3	2	1

4 Teaching-learning-based optimization (TLBO)

In the last few decades, many complex problems have been tackled successfully using the nature inspired meta-heuristic techniques. Among them, a recent meta-heuristic technique called teaching-learning-based optimization algorithm is frequently used due to its simplicity and "tuning parameter free" feature apart from its efficiency. Based on the general learning process of a student, Rao et al. (2011) have proposed the teaching-learning-based optimization (TLBO) algorithm. In general, the learning process of a student occurs in two phases. They are known as 'teacher phase' and 'student phase' as explained below:

4.1 Teacher phase

A teacher gives a lecture and some instructions to his/her students. The students follow their teacher's instructions and gain some knowledge. Thus, a teacher enhances the knowledge of a class of students. In TLBO, 'best student' of the class is considered as the 'teacher' and the 'class of students' as the population of the algorithm. The gained knowledge of a student is given by the equation 1:

$$S_n = S_o + r \times (S_t - (T_f \times S_m)) \quad (1)$$

where S_n is the new knowledge of a student, S_o is the old knowledge of a student, r is a random number (0,1), S_m is the mean knowledge of all the students of the class, T_f is called teaching factor. T_f is assigned a value one or two randomly and it does not require any tuning. Sn is accepted if it provides a better solution.

4.2 Student phase

Not all students of the class can learn equally. The knowledge gained by a student depends on the capacity to learn and understand. This varies from student to student. Therefore, after a class is taught, the weak students try to approach their good learning fellow students to discuss and clarify their doubts. During this discussion process again students gain some knowledge. This is explained as below in equations 2 and 3

$$S_{na} = S_{oa} + r \times (S_a - S_b) \text{ if } S_a \text{ is better than } S_b \quad (2)$$

$$S_{na} = S_{oa} + r \times (S_b - S_a) \; otherwise \quad (3)$$

S_a and S_b are two students of the class (a ≠ b), S_{na} and S_{oa} are the new and old knowledge of student S_a, r is a random number (0,1). S_{na} is accepted if it provides a better solution. The basic TLBO is extended to be applied to multi-objective optimization problems.

5 Multi-objective TLBO (MOTLBO)

Multi-objective optimization (MOO) of conflicting and contradistinctive natured objectives has drawn the attention of researchers in the last few years. Such kind of optimization problems with two or more objectives arise in many real world applications. With the recent success of TLBO, it has been extended to solve the multi-objective optimization problems (Zou et al., 2013; Rao and Patel, 2014; Yu et al, 2015; Patel and Savsani, 2016) which is normally called as multi-objective teaching-learning-based optimization (MOTLBO). The important variation between the basic TLBO (single objective) and MOTLBO is the distribution of teacher. In MOTLBO, teacher should be redefined to get a set of pareto optimal solutions (non-dominated solutions). There is only one teacher for a single objective problem. As more than one contradistinctive conflicting objective is optimized in a MOO problem, there will be multiple non-dominated solutions. Therefore, any of these non-dominated solutions can be used as teacher in MOTLBO.

In a MOO problem of conflicting and contradistinctive natured objectives, optimal solutions mean that these solutions are non-dominating solutions in the search region and none of the other solutions are exceptionally good in all the objectives of multi-objective optimization. These non-dominated solutions are termed as pareto optimal solutions. Each and every solution in the non-dominated set of solutions dominates every other solution in the set in at least one objective of the MOO. In order to extract the non-dominated solutions from total solutions, each single solution has to be compared with each and every other solution in the total population. Following are the rules for comparing the solutions to find the non-dominated solutions.

$$Obj1(p) \leq Obj1\ (q)\ and\ Obj2\ (p) < Obj2\ (q) \quad (4)$$

$$Obj1(p) < Obj1\ (q)\ and\ Obj2\ (p) \leq Obj2\ (q) \quad (5)$$

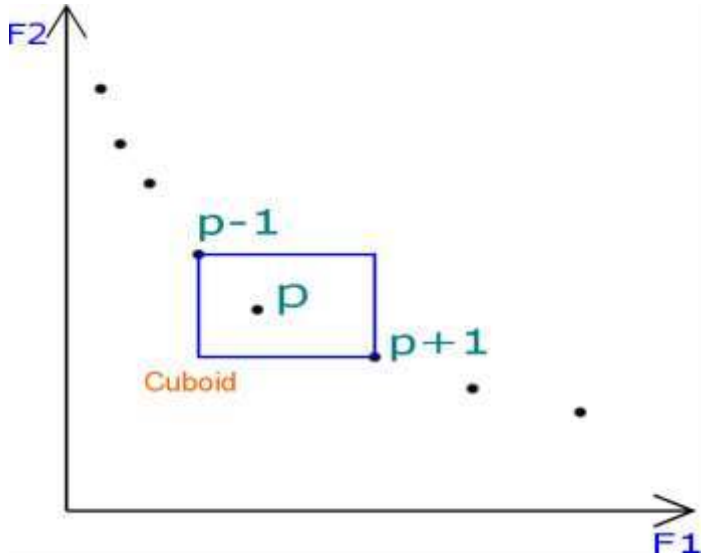
where p and q indicate two different solution members of the population. Obj1 and Obj2 are values of the two objective functions in the MOO problem. A MOO aims to meet two criteria such as (i) Pareto optimal solution set and (ii) Distribution and diversity in the solutions.

The ‘teacher’ who teaches a class of students is fixed in case of solving a single objective optimization problem whereas in case of multi-objective optimization problem, each student may have more than one teacher. Out of which one teacher is chosen to teach the student. Such group of teachers are usually stored in a different place from the class called external archive ‘*EA*’. The non-dominated solutions found so far are stored in this external repository ‘*EA*’. This external archive ‘*EA*’, maintained by MOTLBO, is updated after each iteration. In the beginning, the ‘*EA*’

is empty and can store a maximum of user specified number of pareto-optimal solutions. Of course, there is a chance that the total number of solutions in the '*EA*' may exceed the maximum size limit. In such case, excess solutions are omitted using size controlling methods. There are many methods to control the external archive '*EA*'. They are size control based on epsilon dominance (Mostaghim and Teich, 2003), size control based on maximum fitness (Li, 2004), size control based on crowding distance (Raquel and Naval Jr, 2005). It may so happen that the non-dominated solutions may grow quickly; hence, it is critical to control the archive size. Studies of Alvarez-Benitez (2005) elaborately discusses on control on archive size in MOO.

In order to find well distributed non-dominating solutions for multi-objective optimization problems using meta-heuristic techniques, crowding distance (CD) is the most widely used size control technique (Singh et al., 2016). The crowding distance technique is first used in MOTLBO by Zou et al. (2013) for the selection of best 'teacher'. With an inspiration from previous works, crowding distance technique is applied in the present work to generate a well distributed non-dominating solution. The crowding distance technique possesses the capability to converge towards the pareto front. In order to find at what extent the non-dominated solutions are crowded, the crowding distance factor is used. It actually provides a good estimate of density of solutions that surround a particular solution (Deb et al., 2002). The calculation of crowding distance is shown in the figure 1. The dots in the figure show the pareto optimal solutions of MOO.

Figure 1 Crowding distance



Crowding distance is the average distance, measured along each objective in a MOO, of two solutions that are on either side of a particular solution (point p). It is the largest cuboid that encloses a particular solution as shown in figure 1. The solutions with highest (f_{max}) and lowest (f_{min}) objective functional values are called boundary solutions. The CD values for the boundary solutions are taken as infinity. For the remaining non-dominated solutions, the CD values are calculated using the equation 6 as follows:

$$CD = \frac{f_{p+1} - f_{p-1}}{f_{max} - f_{min}} \tag{6}$$

Finally, the summation of individual crowding distance (CD) factors measured along each of the objective gives the total crowding distance value for a particular solution. Using the CD values, non-dominated solutions are sorted in the decreasing order of CD values. Only top ten percent of the non-dominated solutions are used as teacher.

The teaching learning based algorithm may exhibit a tendency to converge rapidly and get struck at the local optimum due to loss of diversity among the population. To alleviate such a drawback and maintain the diversity, mutation technique (often used in genetic algorithm) is incorporated to the MOTLBO algorithm. Mutation is only applied when the external archive '*EA*' does not show any change in the obtained non-dominated solutions for some fixed number of iterations. The 'teacher' who teaches a class of students is fixed in case of solving a single objective optimization problem whereas each student may have more than one 'teacher' in case of multi-objective optimization problem. Out of which one teacher is chosen to teach the student.

Therefore, in MOTLBO algorithm, the 'teacher' is to be redefined to get the set of pareto optimal solutions. All the solutions present near the pareto front has equal chance to be a 'teacher'. As the iterations proceed, every student of the class will be at his best knowledge level. The knowledge of the students is updated using the equations 9, 10 and 11.

5.1 Multi-objective optimization (MOO)

In a multi-objective optimization problem, the solution is a vector of decision variables that simultaneously satisfy all the constraints of the given problem and optimizes the function vector with each element as each objective of the given multi-objective optimization problem. Generally, a MOO is formulized as

Minimize (or Maximize) $F(\boldsymbol{X}) = [f_1(x), f_2(x), \ldots\ldots, f_z(x)]$

Subject to $h(x) \leq 0$ and $g(x)=0$

In a MOO, solutions optimize (minimize or maximize) the elements of the function vector $F(\boldsymbol{X})$ where $\boldsymbol{X}$ is n dimensional decision variable vector $\boldsymbol{X}= (x_1, x_2, x_3, \ldots\ldots, x_n)$. Constraints $h(x) \leq 0$ and $g(x) = 0$ decide the feasible solutions in optimizing the function vector $F(\boldsymbol{X})$ with 'z' number of objective functions. In the present work, contradictory objectives makespan and mean flowtime of FFSP are minimized. They are explained as follows:

Makespan (F_1): The first objective is makespan (C_{max}) minimization. Makespan is defined (Pinedo 2018) as the completion time of last lob that leaves a manufacturing system. Minimization of makespan is nothing but increasing the throughput of a shop floor. This ensures maximum utilization of the machines in a shop floor.

Mean flowtime (F_2): The second objective is mean flowtime (MF) minimization. Flowtime (fj) of a job j is defined as the amount of time a job j spends in the manufacturing system.

$$f_j = Cj - rj \tag{7}$$

Mean flowtime (*MF*) is defined as the average flowtime of a job in the manufacturing system.

$$MF = \frac{1}{J}\sum_{j=1}^{J} f_j \tag{8}$$

where C_j is the completion time of a job j and r_j is the release date of job j into the shop floor.

A brief explanation of the proposed MOTLBO algorithm is provided in the following sections 5.2 and 5.3.

5.2 MOTLBO algorithm (pseudo code)

1. For i=1 to TS (TS means total students in the class)
    a. Initialize the knowledge of all students of the class randomly.

b. Evaluate the knowledge of each student.
c. Compare the new knowledge of the student with its previous knowledge. If the new knowledge is better than the old knowledge, update the old knowledge with the new knowledge.
d. Find out the best student ('teacher') of the class.

2. End For
3. Initialize iteration counter ic=0
4. Obtain the non-dominated solutions and store them in the external archive '*EA*'.
5. Repeat
   a) Evaluate the CD values to each of the non-dominated solutions in '*EA*'.
   b) Non-dominated solutions are sorted in decreasing order using the crowding distance values.
   c) For i=1 to TS
      i. Teacher is selected randomly from the top ten percent solutions from the '*EA*'.
      ii. Evaluate the new knowledge of the students from teacher phase.
      iii. $S_n = S_o + r \times (S_t - (T_f \times S_m))$ (9)
      iv. Evaluate the new knowledge of the students from student phase.
      v. $S_{na} = S_{oa} + r \times (S_a - S_b)$ if S_a is better than S_b (10)
      $S_{na} = S_{oa} + r \times (S_b - S_a)$ *otherwise* (11)
      vi. Perform mutation if there is no change in '*EA'*.
   d) End For
   e) Insert the new non-dominated solutions into the external archive '*EA'*. Now compare each solution with the other solutions and eliminate the dominated solutions if found any. If '*EA*' reaches its maximum limit, new non-dominated solutions are inserted based on the following criteria:
      i. Find CD values to each of the solution present in '*EA*'.
      ii. Sort the solutions in the decreasing order of CD values.
      iii. Replace the new solution with the bottom 10 percent solutions. End For
   f) Update the new knowledge of each student.
6. Until the termination criteria is met.

5.3 Solution ranking using maximum deviation theory (MDT)

As the MOTLBO generates many non-dominated solutions, selecting one best solution among them merely depends on the decision maker's choice. To take a decision, generally multi-attribute decision making (MADM) techniques are applied. Scores are obtained for each solution and the solution with the maximum score is chosen as the best solution. However, in MADM, the scores are obtained using weighted sum method to convert the multi-objective problem into an equivalent single objective problem. The experts in the field suggest the weight vales. The pre-assigned weights affect the scores which in turn influence the rank of non-dominated solutions. In order to alleviate the drawback of pre-assigned weights, Yingming (1997) has proposed maximum deviation theory (MDT) to extract right information from the available data. The idea of MDT is simple. Attributes with similar values should be assigned smaller weights when compared to attributes with larger deviations.

In order to compare two or more different attributes (objectives) of different scales and units, normalization is to be carried out to bring all the attributes to a common scale of measurement. Here, the makespan and mean flowtime belong to different scales. They should be normalized and brought to a common scale. Normalization depends on 'lower the better' or 'higher the better' type. Equations 12 and 13 are used to evaluate the normalized values of the attributes. The decision matrix consists of attribute values (objective functional vales) of 'n' number of non-dominated solutions obtained for the 'm' number of attributes (objectives) using MOTLBO. Each element of the decision matrix indicates the z^{th} attribute value (z=1, 2, …..., m) of y^{th} alternative (non-dominated solution) (y=1, 2, …..., n).

$$x^*_{yz} = \frac{\max(x_{yz}) - x_{yz}}{\max(x_{yz}) - \min(x_{yz})} \text{ for lower the better type.} \quad (12)$$

$$x^*_{yz} = \frac{x_{yz} - \max(x_{yz})}{\max(x_{yz}) - \min(x_{yz})} \text{ for higher the better type.} \quad (13)$$

The performance value difference for each non-dominated solution (alternative) is computed. The following equation gives the deviation value of the non-dominated solution (alternative) (A_y | y=1, 2, …..., n) from all other non-dominated solutions (alternatives) for the objective (attribute) (A_z| z=1, 2, …..., m).

$$D_{yz}(w_z) = \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) w_z \quad (14)$$

where w_z is the attribute weight to be calculated and $D_{yz}(w_z)$ is the deviation of the non-dominated solutions (alternatives).

For the objective (attribute) ($A_z \mid z=1, 2, \ldots..., m$), the total deviation value of all alternatives with respect to other alternatives can be calculated using the equation 15 as follows:

$$D_z(w_z) = \sum_{y=1}^{n} D_{yz}(w_z) = \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) w_z \quad (15)$$

where $D_z w_z$ is the total deviation of all the non-dominated solutions (alternatives) for a single objective (attribute).

Now the total deviation value '$D(w_z)$' of all objectives (attributes) for all alternatives with respect to other alternatives can be calculated using the equation 16 as follows:

$$D(w_z) = \sum_{z=1}^{m} D_z(w_z) = \sum_{z=1}^{m} \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) w_z \quad (16)$$

In case of MADM problems where the weights of the attributes are completely unknown, Yingming (1997) has proposed a linear programming model to find the weight vector where the deviation values for all the attributes are maximized.

$$\text{Max } D(w_z) = \sum_{z=1}^{m} \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) w_z \;\; s.t \; \sum_{z=1}^{m} w_z^2 = 1, w_z \geq 0 \quad (17)$$

To solve the above model a Lagrange function is constructed as follows:

$$L(w_z, \gamma) = \sum_{z=1}^{m} \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) w_z + \gamma \left(\sum_{z=1}^{m} w_z^2 - 1\right) \quad (18)$$

where $\boldsymbol{\gamma}$ is the Lagrange multiplier. The partial derivative of equation 18 with respect to $\boldsymbol{\gamma}$ and w_z and equating them to zero, we get following equations 19 and 20

$$\frac{\partial L}{\partial w_z} = \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz}) + 2\gamma w_z = 0 \quad (19)$$

$$\frac{\partial L}{\partial \gamma} = \sum_{z=1}^{m} w_z^2 - 1 = 0 \quad (20)$$

Solving the equations 18, 19 and 20 we get expression for normalized attribute weights (w_z) as follows

$$w_z = \frac{\sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz})}{\sum_{z=1}^{m} \sum_{y=1}^{n} \sum_{k=1}^{n} d(\hat{r}_{yz}, \hat{r}_{kz})} \quad (21)$$

By the summation of weighted performance of all objectives (attributes), the composite score of each non-dominated solution (alternative) is estimated and thus the non-dominated solutions (alternatives) obtained through MOTLBO are ranked.

6. Results and discussion

Multi-objective teaching-learning-based optimization (MOTLBO) has been developed in the present work to solve flexible flow-shop scheduling problem (FFSP) with an aim to minimize two contradictory objectives called makespan and mean flowtime. As the objectives are contradictory in nature, it is difficult to find the existence of only one optimized solution. Due to

such a reason, a set of non-dominated pareto optimal solutions are to be found using the pareto based method. By using the problem representation given in the section 3, the proposed MOTLBO is applied to solve the multi-objective FFSP problem. The problem is solved using MATLAB software on Windows 7 platform. The specifications of the desktop computer are 4GB RAM, 500 GB ROM, intlel i7processor running at 3.40 GHz.

Experiments have been conducted on 77 standard benchmark problems to demonstrate the efficiency of proposed MOTLBO (Carlier and Neron, 2000). The obtained results of MOTLBO are compared with the results of MOPSO and NSGA-II taken from Singh et al. (2014). In table 2, for example, the notation of the instance j10c5b3 means that the problem contains 10 jobs and 5 stages. The letter *j* indicates jobs, *c* indicates stages, *b* indicates machine distribution structure and *3* indicates the index of the problem. Problem sizes in the present work vary from 10 jobs× 5 stages to 15 jobs× 10 stages.

On the basis of pareto dominance relation, the comparison of non-dominated solutions obtained using the pareto approach is made. If a solution P is better than a solution Q in all the objectives or solution P is not worse than solution Q for all the objectives, then it means that solution P dominates solution Q. If solution P is not dominated by any other solution then the solution P is said to be a pareto optimal solution. Through the distribution of obtained solutions, pareto optimal solutions produced by the pareto approach represent the trade-off between the objectives. Out of the several non-dominated solutions generated by the pareto optimal approach, a user can select any one of the solution of their choice.

6.1 Performance measures

In case of single objective optimization problems like makespan or mean flowtime or tardiness etc., results obtained by different meta-heuristics for any of these single objectives can be directly compared with each other to test the performance of algorithms. In case of multi-objective optimization problems using the pareto based approach, set of non-dominated solutions is produced. In order to compare the efficiency of the set of non-dominated solutions obtained by different algorithms, the following performance measures are used (Ahmadi et al.,2016). (i) Mean ideal distance (MID), (ii) Rate of achievement of multi-objectives simultaneously (RAS), (iii) Spread of non-dominant solutions (SNS) and (iv) Diversity (D).

(i) Mean ideal distance (MID)

Mean ideal distance (MID) is used to find out the convergence power and efficiency of an algorithm (Karimi et al.,2010). The measurement of MID gives the proximity between pareto optimal solutions and the ideal point. In case of bi-objective optimization problems, (0,0) is generally taken as the ideal point. In the present work, as the goal is to minimize the objectives, lower values of MID indicate the better performance of the algorithm. The formula to find MID is given in the equation 22 as follows:

$$MID = \frac{\sum_{i=1}^{n} \sqrt{f_{1i}^2 + f_{2i}^2}}{n} \tag{22}$$

where f_{1i} and f_{2i} are the functional values of the objectives for a pareto optimal solution I, n is the number of non-dominated solutions

(ii) Rate of achievement of multi-objectives simultaneously (RAS)

It is the rate at which all the objectives in a multi-objective optimization problem are approaching towards the best possible solution. Lower the values of RAS imply better the quality of solutions obtained. Thus, it implies the better performance of an algorithm. The formula to find RAS is given in the equation 23 as follows:

$$RAS = \frac{\sum_{i=1}^{n} |f_{1i} - f_1^{best}| + |f_{2i} - f_2^{best}|}{n} \tag{23}$$

where f_1^{best} and f_2^{best} are the best functional values of the objectives makespan and mean flowtime.

(iii)Spread of non-dominant solutions (SNS)

Spread of non-dominant solutions is another criterion to evaluate the quality of obtained non-dominated solutions. This performance measure indicates the spacing or consistency of distance between the obtained non-dominated solutions. Smaller value of SNS indicates a good consistency of spread between the solutions. In other words, good quality solutions are obtained (Ahmadi et al., 2016). The formula to find SNS is given in the equation 24 as follows:

$$SNS = \sqrt{\frac{\sum_{i=1}^{n} \left(MID - \sqrt{f_{1i}^2 + f_{2i}^2} \right)^2}{n-1}} \tag{24}$$

(iv) Diversity (D)

This performance measure provides the measure of diversity in obtained non-dominated solutions. Greater the diversity, better the quality of solutions obtained (Zitzler,1999). The formula to find diversity (D) is given in the equation 25 as follows:

$$D = \sqrt{(maxf_1 - minf_1)^2 + (maxf_2 - minf_2)^2} \quad (25)$$

where $maxf_1$ and $minf_1$ are the maximum and minimum functional values of the first objective function (makespan) and $maxf_2$ and $minf_2$ are the maximum and minimum functional values of the second objective function (mean flowtime).

To determine the effectiveness of proposed MOTLBO, experiments have been conducted on 77 benchmark problems taken from Carlier and Neron (2000) and the results are compared with MOPSO and NSGA-II taken from Singh et al. (2014). The results are shown in table 2. In table 2, first column shows the name of the problem. Next there are four major columns - one for each performance measure (MID, RAS, SNS and D). Each performance measure has again three sub columns corresponding to the results of each algorithm (MOTLBO, NSGA-II and MOPSO). Numbers in bold letters in table 2 indicate the best values for each instance corresponding to each performance measure. For performance measure like mean ideal distance (MID), MOTLBO produces superior solutions in case of 62 problems. For the performance measure like rate of achievement of multi-objectives simultaneously (RAS), MOTLBO results in superior solution for 55 problems. For the performance measure like spread of non-dominant solutions (SNS), MOTLBO generates superior solution for 58 problems. For the performance measure like diversity (D), MOTLBO gives superior solutions for 16 problems. From the above results, it is indicated that MOTLBO possess the capability to outperform MOPSO and NSGA-II in first three performance measures (MID, RAS, SNS). In the fourth performance like measure diversity (D), MOPSO (gives best results to 46 problems) has outperformed MOTLBO and NSGA-II. From the observation of overall performance analysis, it can be concluded that MOTLBO is the one of the competing algorithms that can be applied to solve multi-objective optimization of FFSP.

In order to select the one best solution out of the many non-dominated solutions generated using the MOTLBO, MDT technique has been applied. In this method composite score for each non-dominated solution is evaluated and the solution with maximum composite score is selected as the best solution. For better understanding, how a solution ranking is generated using MDT,

solution ranking is provided for an instance j10c10c3 in table 3 and figure 2 is provided to describe the nature of pareto front obtained for the conflicting natured objectives of makespan and mean flowtime. From figure 2, it is obvious that an increase in one objective results in the decrement of other objective. From this result, it can be inferred that (i) Focus on single objective optimization may lead to inferior performance in other objectives and (ii) It is not always easy to find the trade-off relationship between the conflicting natured objectives.

To further demonstrate the effectiveness of proposed MOTLBO over the MOPSO and NSGA II, pareto front obtained by each of these algorithms are compared for four random benchmark problems (j10c5a2, j10c10a2, j15c5a1 and j15c10a1) in figure 3, figure 4, figure 5 and figure 6 respectively. From these figures, it is indicated that pareto front obtained by MOTLBO is superior than that obtained with MOPSO and NSGA II. Therefore, it is believed that the proposed MOTLBO performs better than the other algorithms considered in this work.

Table 2 Performance results of pareto front obtained by makespan and mean flowtime objectives

	MID			RAS			SNS			D		
Problem	MOTLBO	NSGA-II	MOPSO	MOTLBO	NSGA-II	MOPSO	MOTLBO	NSGA-II	MOPSO	MOTLBO	NSGA-II	MOPSO
j10c5a2	**104.3198**	116.272	114.647	**5.506**	5.866	8.157	1.796	**1.493**	2.908	7.95	9.941	**15.453**
j10c5a3	**135.8**	141.165	140.991	4.725	3.828	**3.6**	**1.2784**	1.396	1.508	**6.965**	6.462	6.356
j10c5a4	**140.397**	153.429	152.812	**5.12**	9.6	8.691	**1.376**	4.584	3.798	8.342	**18.821**	17.664
j10c5a5	**140.934**	153.26	153.59	4.549	**3.48**	4.91	1.376	0.99	**0.832**	7.085	5.936	**8.345**
j10c5a6	**127.863**	142.015	140.019	**4.043**	5.443	6.273	1.537	**0.808**	1.899	6.428	8.345	**10.825**
j10c5b1	**151.706**	153.29	154.428	5.912	**2.833**	5.25	1.83	1.108	**0.511**	**7.775**	4.472	5.656
j10c5b2	125.4822	123.346	**122.853**	5.186	4.925	**3.075**	1.193	**0.452**	0.703	**7.616**	5.08	7.295
j10c5b3	**128.061**	131.888	131.302	5.263	3.699	**3.365**	1.044	0.872	**0.34**	**9.161**	7.584	7.192
j10c5b4	144.931	137.831	**137.453**	**4.252**	5.614	5.825	**1.42**	1.879	2.159	6.514	5.656	**7.746**
j10c5b5	176.32	**113.214**	115.875	**6.051**	7.67	6.791	**1.952**	3.74	4.33	**9.049**	7.277	8.109
j10c5b6	**135.85**	148.912	146.24	**6.615**	16.923	16.602	**3.326**	8.459	7.537	10.625	10.206	**11.37**
j10c5c1	**84.25**	141.039	139.922	6.957	4.88	**2.944**	2.602	1.079	**0.872**	**10.93**	4.77	7.56
j10c5c2	**89.4**	152.719	151.899	**4.255**	5.966	6.05	**1.51**	3.32	3.409	6.5	12.182	**13.201**
j10c5c3	**85.891**	134.272	135.733	**4.704**	6.069	7.585	**1.473**	1.609	2.058	6.982	**11.095**	10.284
j10c5c4	**81.598**	112.185	110.519	**3.538**	7.677	7.477	**1.64**	2.823	1.901	6.118	**14.038**	12.389
j10c5c5	**95.76**	113.18	110.641	**5.012**	9.687	5.623	**1.378**	2.672	1.735	6.773	**14.045**	5.063
j10c5c6	**82.795**	110.302	111.747	3.848	4.92	**2.4**	1.539	**0.428**	0.52	**6.302**	6.253	3.551
j10c5d1	**79.955**	108.134	107.962	5.676	10.471	**5.04**	**1.247**	2.811	3.657	7.401	11.403	**18.236**
j10c5d2	**88.09**	127.49	125.16	**3.514**	5.21	4.775	1.723	**0.274**	0.381	6.072	7.172	**7.632**
j10c5d3	**78.039**	99.637	97.951	**3.79**	7.05	5.801	**1.514**	2.532	3.27	6.264	10.25	**11.162**
j10c5d4	**83.63**	113.808	109.844	**3.923**	6.12	6.375	**1.578**	3.773	3.913	6.245	10.084	**13.388**
j10c5d5	**78.971**	115.26	114.932	**3.59**	10.133	9.075	**1.687**	3.996	4.737	6.146	**18.629**	17.535
j10c5d6	**75.055**	100.285	98.803	**338**	3.475	4.98	1.764	**1.311**	1.797	6.053	5.546	**7.102**
j10c10a1	**148**	196.419	194.815	**3.982**	10.67	9.93	**1.863**	2.012	2.747	6.395	13.313	**17.915**
j10c10a2	**167.624**	225.517	222.837	**4.208**	15.483	16.1	**1.828**	2.498	4.25	6.527	24.022	**28.801**
j10c10a3	**159.221**	204.318	204.042	**7.407**	13.77	14.418	3.045	**1.165**	3.058	11.925	22.177	**27.62**
j10c10a4	**158.29**	205.065	204.076	3.714	5.16	**3.25**	1.88	**1.027**	1.063	**6.371**	5.635	6.348
j10c10a5	**157.45**	203.83	204.64	**3.954**	4.641	4.903	1.805	1.382	**1.169**	**6.708**	6.356	6.025

j10c10a6	**154.128**	198.768	197.18	**3.596**	9.071	8.556	**1.98**	4.359	**6.186**	6.159	**19.201**	18.56
j10c10b1	**172.452**	191.78	189.903	**4.135**	7.025	5.849	1.786	**0.876**	1.837	6.789	6.832	**8.534**
j10c10b2	**166.74**	234.201	232.462	**5.104**	6.281	5.29	**1.665**	6.277	3.414	**7.113**	6.862	6.043
j10c10b3	**179.003**	200.572	196.029	**5.015**	7.514	6.328	**1.712**	1.843	2.898	7.332	12.762	**13.277**
j10c10b4	**168.945**	230.546	227.104	**3.846**	10.27	11.114	**1.848**	2.119	3.712	6.324	15.367	**18.2**
j10c10b5	**175.124**	221.463	223.239	4.065	**2.982**	5.74	1.801	**0.761**	1.054	**6.645**	2.492	5.281
j10c10b6	**174.798**	211.607	209.365	**4.146**	5.84	4.21	**1.821**	6.462	8.137	6.472	14.852	**17.198**
j10c10c1	**124.758**	220.81	218.419	**3.753**	6.294	5.166	**1.821**	4.752	6.096	**6.429**	2.641	3.201
j10c10c2	**126.0779**	207.456	206.117	**5.102**	9.237	8.667	**1.659**	4.958	5.507	**7.22**	3.483	4.279
j10c10c3	**125.364**	216.192	214.976	**3.588**	5.25	4.416	2.108	0.281	**0.497**	7.175	**7.971**	7.211
j10c10c4	**123.781**	226.74	227.167	**3.31**	10.229	10.827	**1.977**	6.662	7.463	6.039	10.346	**11.379**
j10c10c5	**134.326**	216.96	214.21	**3.586**	8.823	8.075	**1.903**	10.81	12.37	6.141	**18.559**	17.236
j10c10c6	**114.183**	209.127	210.571	**3.492**	7.156	6.441	**1.901**	9.722	10.568	6.113	13.046	**14.279**
j15c5a1	223.151	210.434	**210.196**	**6.483**	7.055	7.324	**0.279**	1.509	1.853	9.278	11.734	**12.132**
j15c5a2	202.79	175.532	**174.618**	**4.326**	10.232	9.645	**1.066**	1.375	1.427	6.926	13.213	**15.652**
j15c5a3	**160.49**	161.157	160.604	**3.661**	5.991	5.566	**0.59**	2.976	2.323	5.289	**10.499**	10.44
j15c5a4	191.576	**153.233**	154.639	**4.705**	5.107	4.911	**0.83**	3.719	2.895	6.451	12.751	**13.892**
j15c5a5	201.635	198.156	**196.087**	**4.892**	6.854	8.506	**0.451**	3.194	4.979	6.584	13.238	**14.391**
j15c5a6	222.45	176.028	**174.378**	**5.604**	7.198	6.327	**0.902**	4.675	3.492	7.341	15.187	**15.494**
j15c5b1	212.83	153.727	**152.846**	6.912	6.183	**5.412**	**0.867**	4.591	4.821	**12.26**	8.734	10.471
j15c5b2	191.968	149.773	**148.245**	5.23	5.014	**4.819**	**0.843**	6.247	7.419	7.341	**13.619**	12.408
j15c5b3	198.585	**166.406**	167.924	**3.769**	8.173	8.56	**1.383**	9.281	10.371	6.3	13.443	**14.209**
j15c5b4	186.96	174.765	**173.707**	4.813	3.412	**2.5**	**0.951**	1.972	1.146	6.965	12.272	**13.905**
j15c5b5	212.249	181.075	**180.204**	**4.08**	7.863	6.173	**1.204**	5.019	6.702	6.514	10.713	**11.492**
j15c5b6	222.261	155.686	**152.704**	4.549	5.7865	**4.346**	**1.104**	2.988	3.379	6.708	7.78	**8.483**
j15c5c1	**115.249**	180.626	179.215	3.561	3.2951	**2.625**	**1.391**	3.397	2.519	**6.107**	5.637	4.242
j15c5c2	**122.211**	199.406	197.906	**3.867**	6.279	5.811	**1.317**	1.745	2.494	6.167	9.287	**10.394**
j15c5c3	**118.058**	171.763	169.941	3.765	2.881	**2.763**	**1.332**	5.523	6.942	6.184	12.492	**14.225**
j15c5c4	**118.752**	203.59	202.873	**3.918**	9.831	10.317	**1.106**	1.186	1.701	6.676	14.372	**16.505**
j15c5c5	**103.679**	134.456	134.127	**3.5**	5.25	4.066	**1.401**	1.822	1.661	6.107	**9.738**	8.814
j15c5c6	**123.064**	155.178	157.763	3.979	**1.6512**	1.916	**1.193**	6.746	7.932	6.347	**12.398**	11.165

j15c5d1	207.099	158.803	**157.102**	**4.261**	9.318	8.229	**1.094**	2.97	3.025	6.863	13.753	**14.402**
j15c5d2	**114.08**	169.323	167.545	3.571	4.421	**3.485**	1.269	**1.109**	2.439	6.324	7.529	**8.464**
j15c5d3	**112.032**	166.41	166.342	**3.58**	11.071	10.354	**1.306**	5.776	6.682	6.146	16.687	**17.125**
j15c5d4	**111.22**	167.046	165.223	4.8	4.561	**3.412**	**0.879**	1.187	1.476	**7.0666**	5.98	6.198
j15c5d5	**106.265**	168.252	167.156	**4.266**	10.752	11.053	**1.23**	3.169	3.924	6.462	14.289	**17.684**
j15c5d6	**108.916**	164.273	163.56	**3.552**	6.52	8.452	**1.517**	3.431	2.667	6.062	11.712	**13.513**
j15c10a1	**256.154**	307.508	307.126	**3.831**	8.141	8.782	**1.778**	3.001	3.363	6.291	12.887	**14.866**
j15c10a2	**220.023**	285.981	285.453	**3.673**	6.46	5.683	1.8	1.698	**1.337**	6.1288	**11.1606**	9.881
j15c10a3	**217.99**	295.652	294.108	**3.801**	8.315	7.4794	**1.814**	2.521	3.298	6.151	12.968	**13.014**
j15c10a4	**244.8**	298.763	300.731	**5**	6.934	6.456	**1.607**	3.874	4.074	6.964	13.275	**15.403**
j15c10a5	**201.11**	275.043	273.554	**3.688**	3.864	4.591	**1.778**	6.402	7.528	6.151	12.151	**13.85**
j15c10a6	**218.852**	309.584	308.875	4.688	4.578	**3.941**	**1.623**	9.052	8.154	6.91	13.67	**14.788**
j15c10b1	**238.525**	328.57	327.432	**4.178**	7.801	6.773	**1.735**	2.991	3.436	6.446	10.083	**11.609**
j15c10b2	**203.189**	320.931	321.608	**3.647**	11.728	10.195	**1.826**	6.991	7.46	6.202	**17.321**	16.585
j15c10b3	**239.196**	350.458	348.036	4.239	4.367	**3.034**	**2.778**	5.075	6.297	6.486	10.211	**12.51**
j15c10b4	**237.624**	334.895	334.412	**3.617**	6.289	7.545	**1.848**	4.528	5.478	6.211	9.148	**10.188**
j15c10b5	**215.537**	353.425	354.318	**4.494**	6.904	6.267	**2.719**	10.032	9.258	6.527	**17.574**	16.123
j15c10b6	**235.379**	320.679	318.618	4.632	2.982	**2.123**	**1.601**	7.626	6.477	6.789	9.678	**10.083**

Numbers in bold indicate the best values.

Table 3 Solution ranking for the problem j10c10c3 obtained through MDT

Run	Makespan (C_{max})	Mean Flowtime (MF)	Normalized C_{max}	Normalized MF	Weighted C_{max}	Weighted MF	Composite Score	Rank
1	116	40.4069	1	0	0.503832	0	0.503832	5
2	117	39.75862	0.857143	0.41048	0.431856	0.203667	0.635523	3
3	**118**	**39.38621**	**0.714286**	**0.646288**	**0.35988**	**0.320668**	**0.680548**	**1**
4	119	39.1869	0.571429	0.772489	0.287904	0.383284	0.671188	2
5	121	38.93103	0.285714	0.934498	0.143952	0.463668	0.60762	4
6	123	38.82759	0	1	0	0.496168	0.496168	6

Numbers in bold indicate the best result obtained through MDT.

Figure 2 Pareto front obtained for the problem j10c10a3 using MOTLBO

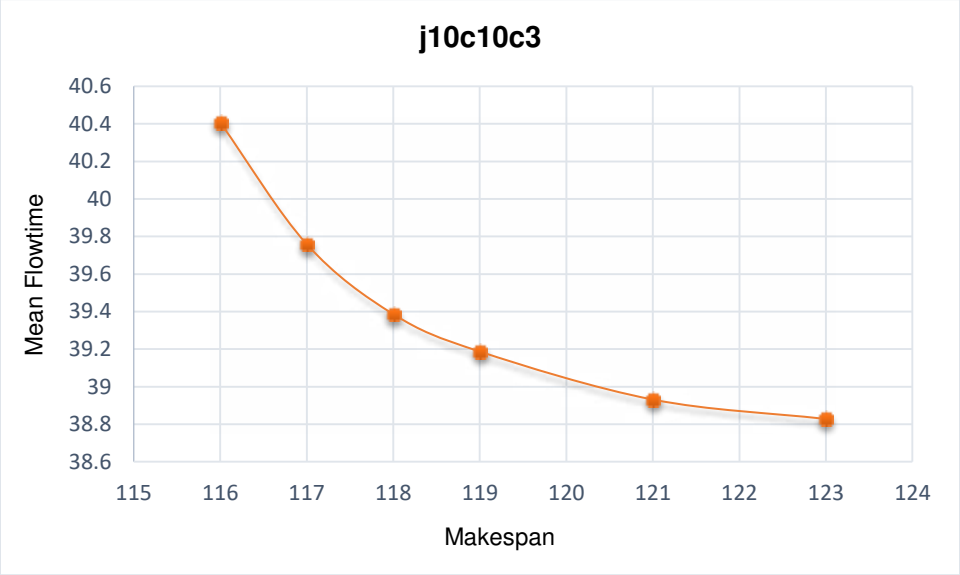


Figure 3 Pareto front obtained for the problem j10c5a2

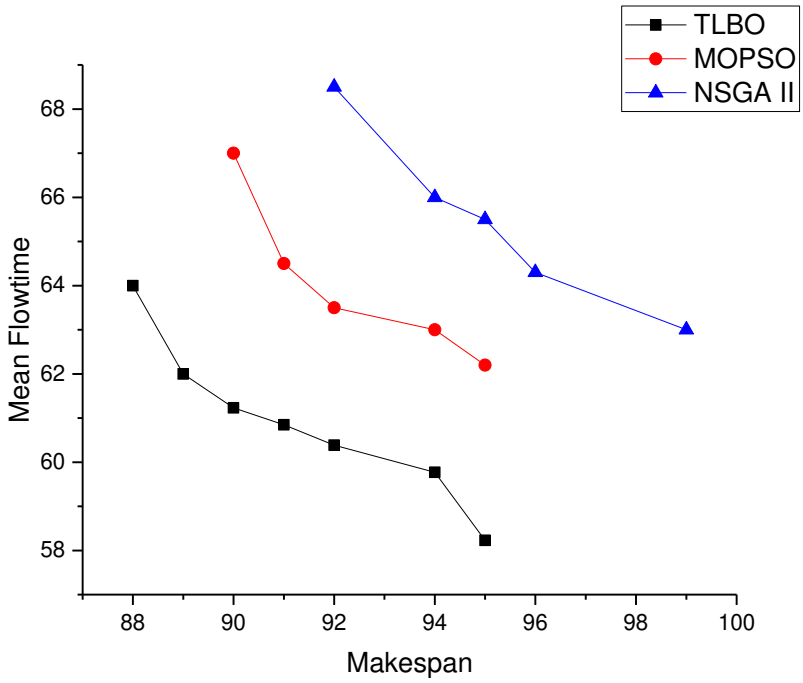


Figure 4 Pareto front obtained for the- problem j10c10a2

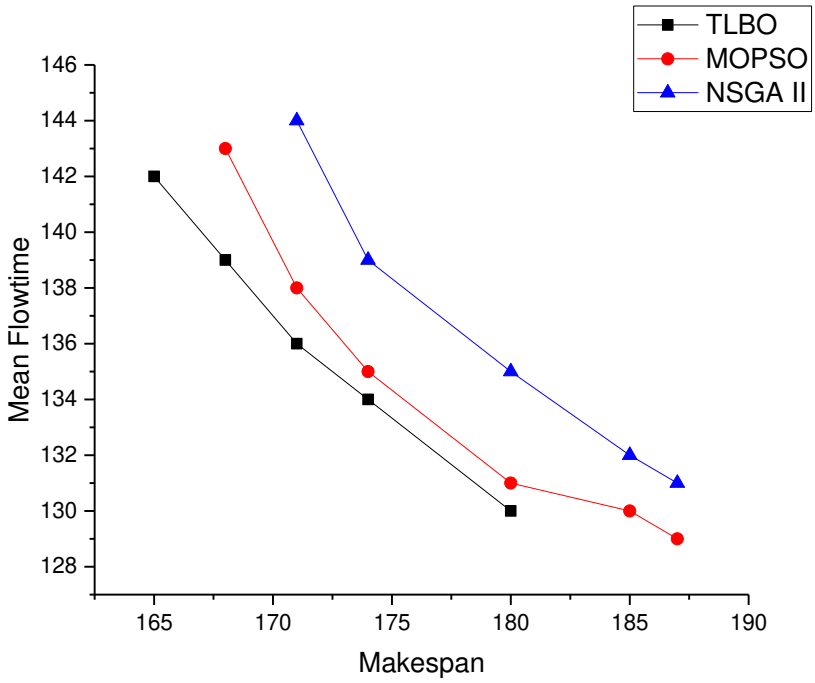


Figure 5 Pareto front obtained for the problem j15c5a1

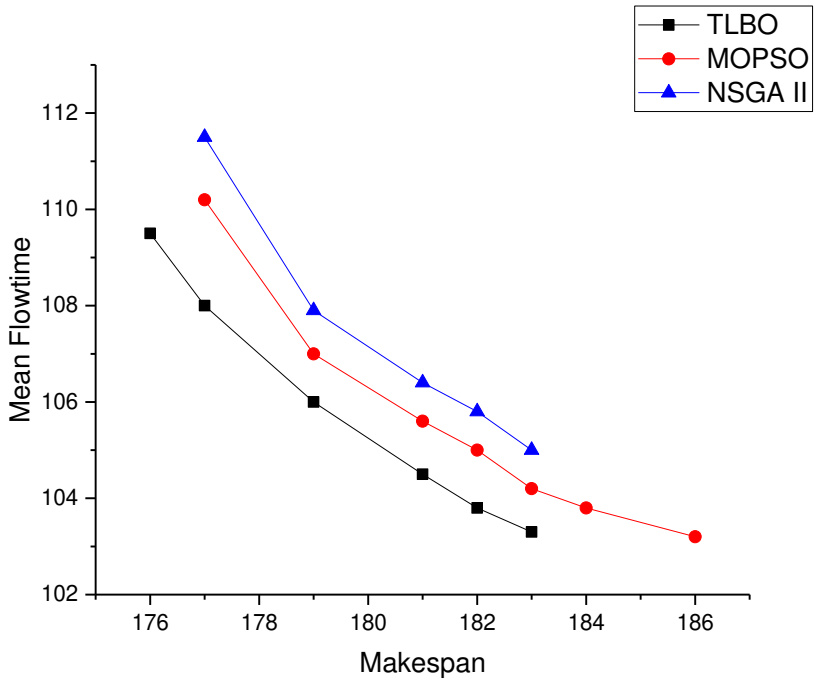
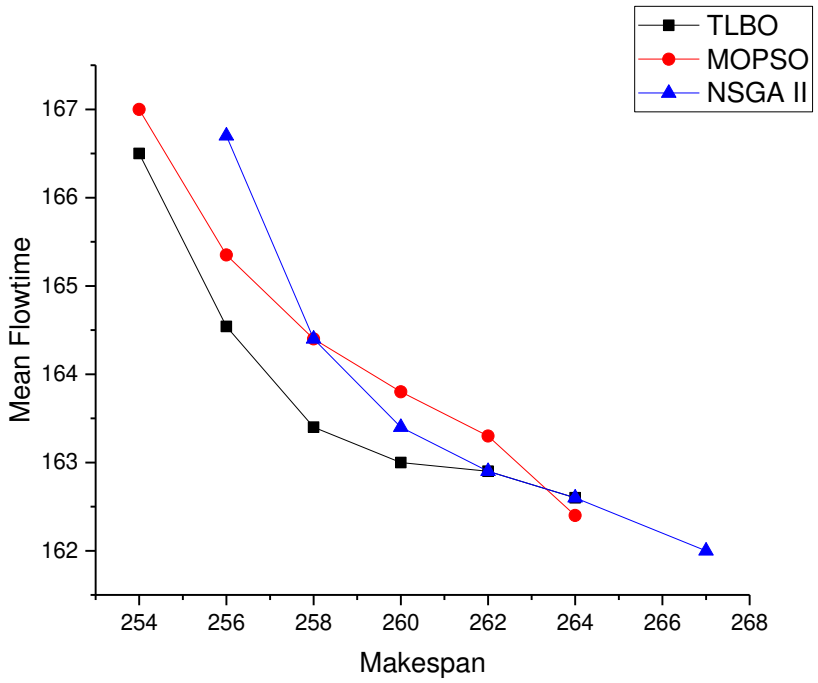


Figure 6 Pareto front obtained for the problem j15c10a1



7. Conclusions

The present work provides an efficient methodology for finding effective solution for multi-objective optimization commonly encountered in scheduling in a flexible flow-shop manufacturing setup. It is demonstrated that MOTLBO can produce near optimal schedule to FFSP by simultaneous consideration of contradistinctive and conflicting natured objectives such as makespan and mean flowtime. A pareto based method is adopted to find a set of non-dominated solutions. In order to generate diversified solutions, crowding distance approach in embedded in the proposed methodology. Mutation strategy, commonly used in genetic algorithm, is adopted in the proposed methodology in order to prevent the premature convergence of MOTLBO. The best solution among the non-dominated set of solutions is chosen using maximum deviation theory (MDT). In a multi-objective optimization problem, maximum deviation theory determines the unknown weight values of the objectives (attributes). Using these attribute weights, a composite score (which is the sum of attribute weights) is calculated for each non-dominated solution (alternative). Ranking of solutions is made in descending order of composite score so that the best unique solution can be selected. MDT helps the decision makers not to rely on imprecise and subjective decision making for selection of the best solution. However, it is observed that MOTLBO generates less diversified solutions as compared to MOPSO. It can be concluded that the proposed MOTLBO is one of the competing algorithm in solving the multi-objective FFSP because it outperforms MOPSO and NSGA-II in majority of instances evaluated under various performance measures.

Flexible flow shop scheduling finds widespread applications in industries particularly confectionery, printing, sugar industries etc. in which multiple processors are available at each stage. Further, effective solution addressing multiple conflicting objectives encountered in the field is of great interest. Therefore, managers look for simple but robust solution of practical problems with less computational efforts. Extensive computational experience on the proposed algorithm suggests that the algorithm can provide reasonably good solution with less computational effort while solving multi-objective FFSP.

In future, the work can be extended to study the performance of other meta-heuristic techniques to solve multi-objective FFSP. The study can be also extended for the multi-objective optimization of FFSP considering other objectives like tardiness, robustness and stability used in the context of scheduling.

Acknowledgement
The authors extend their heart felt gratitude to the Editor-in-chief and anonymous reviewer(s) of International Journal of Industrial and Systems Engineering for providing constructive suggestions that helped to improve the literal and technical content of the paper.

References

Ahmadi, E., Zandieh, M., Farrokh, M., and Emami, S. M. (2016) 'A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms', Computers and Operations Research, Vol. 73, pp.56-66.

Alvarez-Benitez, J. E., et al. (2005) 'A MOPSO algorithm based exclusively on pareto dominance concepts', International Conference on Evolutionary Multi-Criterion Optimization, Springer.

Azizoglu, M., Cakmak, E., and Kondakci, S. (2001) 'A flexible flowshop problem with total flow time minimization', European Journal of Operational Research, Vol.132, No.3, pp. 528-538.

Baker, K. R. and D. Trietsch (2013) 'Principles of sequencing and scheduling', John Wiley and Sons.

Behnamian, J. and Zandieh, M. (2011) 'A discrete colonial competitive algorithm for hybrid flowshop scheduling to minimize earliness and quadratic tardiness penalties', Expert Systems with Applications, Vol.38, No.12, pp.14490-14498.

Botta-Genoulaz, V. (2000) 'Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness', International Journal of Production Economics, Vol. 64, No.1-3, pp.101-111.

Buddala, R. and Mahapatra, S. S. (2016)' An effective teaching learning based optimization for flexible job shop scheduling', International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), IEEE.

Buddala, R. and Mahapatra, S. S. (2018) 'Improved teaching–learning-based and JAYA optimization algorithms for solving flexible flow shop scheduling problems',Journal of Industrial Engineering International, Vol.14, no.3,pp. 555-570.

Buddala, R. and Mahapatra, S. S. (2018) 'An integrated approach for scheduling flexible job-shop using teaching–learning-based optimization method', Journal of Industrial Engineering International, Vol.15,No.1,pp, 181-192.

Buddala, R. and Mahapatra, S. S. (2018) 'Two-stage teaching-learning-based optimization method for flexible job-shop scheduling under machine breakdown', The International Journal of Advanced Manufacturing Technology',Vol.100, No.5-8,pp.1419-1432.

Carlier, J. and Neron, E. (2000) 'An exact method for solving the multi-processor flow-shop', RAIRO-Operations Research, Vol.34, No.1, pp. 1-25.

Cho, H.-M. and Jeong, I.-J. (2017) 'A two-level method of production planning and scheduling for bi-objective reentrant hybrid flow shops', Computers and Industrial Engineering , Nol.106, No.1,pp.174-181.

Cui, Z. and Gu, X. (2015) 'An improved discrete artificial bee colony algorithm to minimize the makespan on hybrid flow shop problems', Neurocomputing, Vol.148, pp.248-259.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T.A.M.T., 2002 'A fast and elitist multiobjective genetic algorithm: NSGA-II', IEEE transactions on evolutionary computation, Vol. 6, No.2, pp.182-197.

Dios, M., Fernandez-Viagas, V., and Framinan, J. M. (2018) 'Efficient heuristics for the hybrid flow shop scheduling problem with missing operations', Computers and Industrial Engineering, Vol.115, pp.88-99.

Engin, O. and Doyen, A. (2004)'A new approach to solve hybrid flow shop scheduling problems by artificial immune system',Future generation Computer Systems, Vol.20, No.6,pp.1083-1095.

Guinet, A. and Solomon, M. (1996) 'Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time', International Journal of Production Research, Vol.34,No.6,pp.1643-1654.

Gupta, J. N. (1988) 'Two-stage, hybrid flowshop scheduling problem', Journal of the Operational Research Society,Vol.39,No.4,pp.359-364.

Huang, R. H., Yang, C. L., and Hsu, C. T. (2015) 'Multi-objective two-stage multiprocessor flow shop scheduling–a subgroup particle swarm optimisation approach', International Journal of Systems Science Vol.46,No.16, pp.3010-3018.

Kacem, I., Hammadi, S., and Borne, P. (2002) 'Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic, Mathematics and computers in simulation,Vol.60, No.3-5, pp. 245-276.

Kahraman, C., Engin, O., Kaya, I., and Kerim Yilmaz, M. (2008) 'An application of effective genetic algorithms for solving hybrid flow shop scheduling problems', International Journal of Computational Intelligence Systems,Vol.1, No.2, pp.134-147.

Karimi, N., Zandieh, M., and Karamooz, H. R. (2010) 'Bi-objective group scheduling in hybrid flexible flowshop: a multi-phase approach', Expert Systems with Applications, Vol.37,No.6,pp.4024-4032.

Keesari, H. and Rao, R. (2014) 'Optimization of job shop scheduling problems using teaching-learning-based optimization algorithm', Opsearch,Vol. 51,No.4,pp.545-561.

Kia, H. R., Davoudpour, H., and Zandieh, M. (2010) 'Scheduling a dynamic flexible flow line with sequence-dependent setup times: a simulation analysis', International Journal of Production Research,Vol.48,No.14,pp.4019-4042.

Kia, H., Ghodsypour, S. H., and Davoudpour, H. (2017) 'New scheduling rules for a dynamic flexible flow line problem with sequence-dependent setup times', Journal of Industrial Engineering International,Vol.13, No.3,pp.297-306.

Lee, G. C., and Kim, Y. D. (2004)'A branch-and-bound algorithm for a two-stage hybrid flowshop scheduling problem minimizing total tardiness', International Journal of Production Research, Vol.42,No.22,pp.4731-4743.

Lei, D. (2008) 'Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems',The International Journal of Advanced Manufacturing Technology,Vol.37,No.1-2,pp.157-165.

Li, J. Q., Pan, Q. K., and Liang, Y. C. (2010) 'An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems', Computers and Industrial Engineering,Vol.59,No.4,pp.647-662.

Li, J. Q., Sang, H. Y., Han, Y. Y., Wang, C. G., abd Gao, K. Z. (2018) 'Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions', Journal of Cleaner Production,Vol. 181,pp.584-598.

Li, X. (2004) 'Better spread and convergence: Particle swarm multiobjective optimization using the maximin fitness function', Genetic and Evolutionary Computation Conference, Springer.

Liao, C. J., Tjandradjaja, E., & Chung, T. P. (2012) 'An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem', Applied Soft Computing,Vol.12,No.6, pp.1755-1764.

Maleki-Darounkolaei, A., Modiri, M., Tavakkoli-Moghaddam, R., & Seyyedi, I. (2012) 'A three-stage assembly flow shop scheduling problem with blocking and sequence-dependent set up times', Journal of Industrial Engineering International,Vol.8,No.1,pp.26-30.

Marichelvam, M. and Geetha, M. (2016) 'Application of novel harmony search algorithm for solving hybrid flow shop scheduling problems to minimise makespan', International Journal of Industrial and Systems Engineering,Vol. 23, No.4,pp.467-481.

Marichelvam, M. and T. Prabaharam (2012) 'A bat algorithm for realistic hybrid flowshop scheduling problems to minimize makespan and mean flow time', ICTACT Journal on soft Computing,Vol.3,No.1,pp.428-433.

Marichelvam, M. K., Prabaharan, T., and Yang, X. S. (2014) 'Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan', Applied Soft Computing,Vol.19,pp. 93-101.

Marichelvam, M. K., Prabaharan, T., and Yang, X. S. (2013) 'A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems', IEEE transactions on evolutionary computation,Vol.18,No.2,pp.301-305.

Mirabi, M., Ghomi, S. F., & Jolai, F. (2014) 'A novel hybrid genetic algorithm to solve the make-to-order sequence-dependent flow-shop scheduling problem', Journal of Industrial Engineering International,Vol.10, No.2,pp.57.

Mishra, A. and Shrivastava, D. (2018) 'A TLBO and a Jaya heuristics for permutation flow shop scheduling to minimize the sum of inventory holding and batch delay costs', Computers and Industrial Engineering,Vol. 124,pp.509-522.

Mostaghim, S. and Teich, J. (2003) 'Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO)', Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE, IEEE.

Nishi, T. and Hiranaka, Y. (2013)'Lagrangian relaxation and cut generation for sequence-dependent setup time flowshop scheduling problems to minimise the total weighted tardiness', International Journal of Production Research,Vol.51,No.16,pp.4778-4796.

Niu, Q., Zhou, T., & Ma, S. (2009) 'A quantum-inspired immune algorithm for hybrid flow shop with makespan criterion', Journal of Universal Computer Science,Vol.15,No.4,pp.765-785

Pan, Q.-K. and Dong, Y. (2014) 'An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation', Information Sciences,Vol.277,pp.643-655.

Patel, V. K. and Savsani, V. J. (2016) 'A multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO)', Information Sciences,Vol. 357,pp.182-200.

Pinedo, M. L. (2018)'Scheduling: theory, algorithms, and systems', Springer.

Rao, R. and Patel, V. (2014) 'A multi-objective improved teaching-learning based optimization algorithm for unconstrained and constrained optimization problems', International Journal of Industrial Engineering Computations.Vol.5. No.1,pp.1-22.

Rao, R. V., Savsani, V. J., and Vakharia, D. P. (2011) 'Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems',Computer-Aided Design,Vol.43,No.3,pp.303-315.

Raquel, C. R., and Naval Jr, P. C. (2005) 'An effective use of crowding distance in multiobjective particle swarm optimization, Proceedings of the 7th annual conference on Genetic and evolutionary computation, ACM.

Reddy, M. S., Ratnam, C., Rajyalakshmi, G., and Manupati, V. K. (2018) 'An effective hybrid multi objective evolutionary algorithm for solving real time event in flexible job shop scheduling problem', Measurement, Vol.114, pp.78-90.

Shahvari, O. and Logendran, R. (2016) 'Hybrid flow shop batching and scheduling with a bi-criteria objective', International Journal of Production Economics,Vol.179,pp.239-258.

Singh, M. R. (2014) 'A study on flexible flow shop and job shop scheduling using meta-heuristic approaches' Ph.D. Dissertation. Department of Mechanical Engineering. Rourkela, India, National Institute of Technology, Rourkela. Ph.D.

Singh, M. R. and Mahapatra S. (2012) 'A swarm optimization approach for flexible flow shop scheduling with multiprocessor tasks', The International Journal of Advanced Manufacturing Technology,Vol.62,No.1-4,pp. 267-277.

Singh, M. R., Singh, M., Mahapatra, S. S., & Jagadev, N. (2016) 'Particle swarm optimization algorithm embedded with maximum deviation theory for solving multi-objective flexible job shop scheduling problem', The International Journal of Advanced Manufacturing Technology,Vol.85, No.9-12, pp. 2353-2366.

Sun, Y., Zhang, C., Gao, L., & Wang, X. (2011) 'Multi-objective optimization algorithms for flow shop scheduling problem: a review and prospects', The International Journal of Advanced Manufacturing Technology,Vol.55, No.5-8, pp.723-739.

Tay, J. C. and Ho, N. B. (2008) 'Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems', Computers and Industrial Engineering,Vol.54,No.3,pp.453-473.

Tran, T. H. and Ng, K. M. (2013) 'A hybrid water flow algorithm for multi-objective flexible flow shop scheduling problems', Engineering Optimization,Vol.45,No.4,pp.483-502.

Wang, S. and M. Liu (2014). "Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method." International Journal of Production Research 52(5): 1495-1508.

Xia, W. and Wu Z. (2005) 'An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems', Computers and Industrial Engineering,Vol. 48, No.2, pp.409-425.

Xie, Z., Zhang, C., Shao, X., Lin, W., and Zhu, H. (2014) 'An effective hybrid teaching–learning-based optimization algorithm for permutation flow shop scheduling problem', Advances in Engineering Software, Vol.77,pp.35-47.

Xu, Y., Wang, L., Wang, S. Y., & Liu, M. (2015) 'An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time', Neurocomputing,Vol.148,pp:260-268.

Yingming, W. (1997) 'Using the method of maximizing deviation to make decision for multiindices', Journal of Systems Engineering and Electronics,Vol. 8,No.3,pp.21-26.

Yu, K., Wang, X., & Wang, Z. (2015) 'Self-adaptive multi-objective teaching-learning-based optimization and its application in ethylene cracking furnace operation optimization', Chemometrics and Intelligent Laboratory Systems,Vol.146,pp.198-210.

Zitzler, E. (1999) 'Evolutionary algorithms for multiobjective optimization: Methods and applications', Ph.D. Dissertation. Zurich, Switzerland, Swiss Federal Institute of Technology (ETH). Ph.D.

Zou, F., Wang, L., Hei, X., Chen, D., & Wang, B. (2013) 'Multi-objective optimization using teaching-learning-based optimization algorithm', Engineering Applications of Artificial Intelligence,Vol.26, No.4,pp. 1291-1300.