Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A multi-objective improved teaching-learning based optimization algorithm for unconstrained and constrained optimization problems

R. Venkata Rao[a*] and Vivek Patel[b]

[a]*S.V. National Institute of Technology, Ichchanath, Surat, Gujarat – 395 007, India*
[b]*L.E. College, Morbi, Gujarat – 395 007, India*

C H R O N I C L E

Article history:
Received July 2 2013
Received in revised format September 7 2013
Accepted September 15 2013
Available online September 23 2013

Keywords:
Multi-objective optimization
Teaching-learning based optimization
Inverted generational distance

A B S T R A C T

The present work proposes a multi-objective improved teaching-learning based optimization (MO-ITLBO) algorithm for unconstrained and constrained multi-objective function optimization. The MO-ITLBO algorithm is the improved version of basic teaching-learning based optimization (TLBO) algorithm adapted for multi-objective problems. The basic TLBO algorithm is improved to enhance its exploration and exploitation capacities by introducing the concept of number of teachers, adaptive teaching factor, tutorial training and self-motivated learning. The MO-ITLBO algorithm uses a grid-based approach to adaptively assess the non-dominated solutions (i.e. Pareto front) maintained in an external archive. The performance of the MO-ITLBO algorithm is assessed by implementing it on unconstrained and constrained test problems proposed for the Congress on Evolutionary Computation 2009 (CEC 2009) competition. The performance assessment is done by using the inverted generational distance (IGD) measure. The IGD measures obtained by using the MO-ITLBO algorithm are compared with the IGD measures of the other state-of-the-art algorithms available in the literature. Finally, Lexicographic ordering is used to assess the overall performance of competitive algorithms. Results have shown that the proposed MO-ITLBO algorithm has obtained the 1st rank in the optimization of unconstrained test functions and the 3rd rank in the optimization of constrained test functions.

1. Introduction

Finding the global optimum value(s) of a problem involving more than one objective with conflicting nature arises in many scientific applications. The problem of optimization involving more than one objective function with conflicting nature is known as multi-objective optimization (MOO) problem. Multi-objective optimization has been defined as finding a vector of decision variables while optimizing several objectives simultaneously with a given set of constraints. Unlike the single objective optimization, MOO solutions are in such a way that the performance of each objective cannot be

* Corresponding author. Tel: 91-261-2201661, Fax: 91-261-2201571
E-mail: ravipudirao@gmail.com (R. Venkata Rao)

doi: 10.5267/j.ijiec.2013.09.007

improved without sacrificing the performance of another one. Hence, the solution of MOO problem is always a trade-off between the objectives involved in the problem. Moreover, the obtained result in multi-objective optimization is a set of solutions because the objective functions are conflicting in nature (Akbari & Ziarati, 2012; Zhou et al., 2011).

The multi-objective optimization techniques can be classified into three main groups: Priori techniques, Progressive techniques and Posteriori techniques (Veldhuizen, 1999). Priori techniques employ decision making before the optimization algorithm starts searching the search space. These techniques are divided into three sub-groups: Lexicographic techniques, linear fitness combination techniques and nonlinear fitness combination techniques. In Progressive techniques, there is a direct interaction between the decision making and the search process of the optimization algorithm. Posteriori techniques provide a set of solutions with the search process of MOO problem for the decision making (Coello et al., 2007). These techniques are divided into many sub-groups like independent sampling, aggregation selection, criterion selection, Pareto sampling, Pareto-based selection, Pareto rank and niche-based selection, Pareto elitist-based selection, and hybrid selection. Among all these techniques, most of the research is focused on Pareto based techniques.

The computational effort required to solve the MOO problems is quite considerable. Moreover, many of these problems cannot be solved analytically and consequently they have to be addressed by numerical algorithms. Recently several authors have proposed different evolutionary and swarm intelligence based MOO algorithms to solve these types of problems. Some of the evolutionary MOO algorithms that aimed to obtain a true Pareto front for multi-objective problems include the following:

- Multiple Trajectory Search (MTS) (Tseng & Chen, 2009)
- Dynamical Multi-Objective Evolutionary Algorithm (DMOEADD) (Liu et al., 2009)
- LiuLi Algorithm (Liu and Li, 2009)
- Generalized Differential Evolution 3 (GDE3) (Kukkonen & Lampinen, 2009)
- Multi-Objective Evolutionary Algorithm based on Decomposition (MOEAD) (Zhang et al., 2009)
- Enhancing MOEA/D with Guided Mutation and Priority Update (MOEADGM) (Chen et al., 2009)
- Local Search Based Evolutionary Multi-Objective Optimization Algorithm (NSGAIILS) (Sindhya et al., 2009)
- Multi-Objective Self-adaptive Differential Evolution Algorithm with Objective-wise Learning Strategies (OWMOSaDE) (Huang et al., 2009)
- Clustering Multi-Objective Evolutionary Algorithm (Clustering MOEA) (Wang et al., 2009)
- Archive-based Micro Genetic Algorithm (AMGA) (Tiwari et al., 2009)
- Multi-Objective Evolutionary Programming (MOEP) (Qu & Suganthan, 2009)
- Differential Evolution with Self-adaptation and Local Search Algorithm (DECMOSA-SQP) (Zamuda et al., 2009)
- An Orthogonal Multi-objective Evolutionary Algorithm with Lower-dimensional Crossover (OMOEAII) (Gao et al., 2009)
- NSGA-II (Deb et al., 2002)
- Evaluating the epsilon-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions (Deb et al., 2005)

Similarly, different types of swarm intelligence based algorithm have been presented in the literature to solve the MOO problems. Some of the swarm intelligence algorithms which efficiently solved the multi-objective problems include the following:

- Multi-objective Particle Swarm optimization (MOPSO) (Coello et al., 2004)

- PSO-based multi-objective optimization with dynamic population size and adaptive local archives (Leong & Yen, 2008)
- Covering Pareto-optimal fronts by sub swarms in multi-objective particle swarm optimization (Mostaghim & Teich, 2004)
- Particle swarm inspired evolutionary algorithm (PS-EA) for multi-objective optimization problem (Srinivasan & Seow, 2003)
- Interactive Particle Swarm Optimization (IPSO) (Agrawal et al., 2008)
- Dynamic Multiple Swarms in Multi-Objective Particle Swarm Optimization (DSMOPSO) (Yen & Leong, 2009)
- Autonomous bee colony optimization for multi-objective function (Zeng et al., 2010)
- A multi-objective artificial bee colony for optimizing multi-objective problems (Hedayatzadeh et al., 2010)
- A novel multi-objective optimization algorithm based on artificial bee colony (Zou et al., 2011)
- Multi-objective bee swarm optimization (Akbari & Ziarati, 2012)
- Multi-objective artificial bee colony algorithm (Akbari & Ziarati, 2012)

The evolutionary and swarm intelligence based algorithms are probabilistic algorithms and required common controlling parameters like population size and number of generations. Besides the common control parameters, different algorithms require their own algorithm-specific control parameters. For example, GA uses mutation rate and crossover rate. Similarly, PSO uses inertia weight, social and cognitive parameters. The proper tuning of the algorithm-specific parameters is a very important factor for the efficient working of the evolutionary and swarm intelligence based algorithms. The improper tuning of the algorithm-specific parameters either increases the computational effort or yields the local optimal solution. Considering this fact, recently Rao et al. (2011, 2012a; 2012b), Rao and Patel (2012, 2013a; 2013b, 2013c) introduced the Teaching-learning based optimization (TLBO) algorithm which does not require any algorithm-specific parameters. TLBO requires only common control parameters like population size and number of generations for its working. Thus, TLBO can be said as an algorithm-specific parameter-less algorithm.

In the present work, a multi-objective improved teaching-learning based optimization (MO-ITLBO) algorithm is proposed for multi-objective unconstrained and constrained optimization problems. The improved TLBO (ITLBO) algorithm incorporates some modifications in the basic TLBO algorithm to enhance its exploration and exploitation capacities. The MO-ITLBO algorithm uses a fixed size archive to maintain the good solutions obtained in every iteration. The ε - dominance method is used to maintain the archive (Deb et al., 2005). In ε - dominance method the size of the final external archive depends on the ε value, which is usually a user-defined parameter. The solutions kept in the external archive are used by the learners to update their knowledge. The proposed algorithm uses a grid to control the diversity over the external archive.

The remainder of this paper is organized as follows. Section 2 briefly describes the basic TLBO algorithm. Section 3 explains the modifications in the basic TLBO algorithm and the proposed MO-ITLBO algorithm. Section 4 presents experimentation on unconstrained and constrained test functions. Finally, the conclusion of the present work is presented in section 5.

2. Teaching-learning-based optimization (TLBO) algorithm

Teaching-learning is an important process where every individual tries to learn something from other individuals to improve himself/herself. Rao et al. (2011, 2012a; 2012b), Rao and Patel (2012, 2013a; 2013b, 2013c) proposed an algorithm known as teaching-learning based optimization (TLBO) which simulates the traditional teaching-learning phenomenon of the classroom. The algorithm simulates two fundamental modes of learning: (i) through teacher (known as teacher phase) and (ii) interacting with the other learners (known as the learner phase). TLBO is a population based algorithm where a group of students (i.e. learners) is considered as population and the different subjects offered to the learners is analogous with the different design variables of the optimization problem. The grades of a learner in each subject represent a possible solution to the optimization problem (value of design variables) and the mean result of a learner considering all subjects corresponds to the quality of the associated solution (fitness value).The best solution in the entire population is considered as the teacher.

At the first step, the TLBO generates a randomly distributed initial population $p_{initial}$ of n solutions, where n denotes the size of population. Each solution X^k (k = 1, 2, ..., n) is a m-dimensional vector where m is the number of optimization parameters (design variables). After initialization, the population of the solutions is subjected to repeated cycles, i = 1, 2, ..., g, of the teacher phase and learner phase. Working of the TLBO algorithm is explained below with the teacher phase and learner phase.

2.1. Teacher phase

This phase of the algorithm simulates the learning of the students (i.e. learners) through teacher. During this phase a teacher conveys knowledge among the learners and puts efforts to increase the mean result of the class. Suppose there are 'm' number of subjects (i.e. design variables) offered to 'n' number of learners (i.e. population size, k=1,2,…,n). At any sequential teaching-learning cycle i, let us denote as $M_{j,i}$ the mean result of the learners in a particular subject 'j' (j=1,2,…,m). Since a teacher is the most experienced person on a subject, the best learner in the entire population is considered as a teacher in the algorithm. Let $X^b_{j,i}$, ($b \in k$) be the grades of the best learner and $f(X^b)$ the result of the best learner considering all the subjects, who is identified as a teacher for that cycle. Teacher will put maximum effort to increase the knowledge level of the whole class, but learners will gain knowledge according to the quality of teaching delivered by a teacher and the quality of learners present in the class. Considering this fact the difference between the grade of the teacher and mean grade of the learners in each subject is expressed as,

$$Difference\_Mean_{j,i} = r_i\,(X^b_{j,i} - T_F M_{j,i}), \tag{1}$$

where $X^b_{j,i}$ is the grade of the teacher (i.e. best learner) in subject j. T_F is the teaching factor which decides the value of mean to be changed, and r_i is a random number in the range [0, 1]. The value of T_F can be either 1 or 2 and decided randomly as,

$$T_F = round\,[1 + r_i], \tag{2}$$

where r_i is a random number in the range [0, 1]. The value of T_F is not given as an input to the algorithm and its value is randomly decided by the algorithm using Eq. (2).

Based on the $Difference\_Mean_{j,i}$, the existing solution 'k' is updated in the teacher phase according to the following expression.

$$X'^k_{j,i} = X^k_{j,i} + Difference\_Mean_{j,i}, \tag{3}$$

where $X'^k_{j,i}$ is the updated value of $X^k_{j,i}$. The algorithm accepts $X'^k_{j,i}$ if it gives a better function value otherwise keeps the previous solution. All the accepted grades (i.e design variables) at the end of the teacher phase are maintained and these values become the input to the learner phase.

2.2. Learner phase

This phase of the algorithm simulates the learning of the students (i.e. learners) through interaction among themselves. The students can also gain knowledge by discussing and interacting with the other students. A learner will learn new information if the other learners have more knowledge than him or her. The learning phenomenon of this phase is expressed below. The algorithm randomly selects two learners p and q such that $f(X^p) \neq f(X^q)$ (where $f(X^p)$ and $f(X^q)$ are the updated result of the learners p and q considering grades of all the subjects at the end of teacher phase and $p, q \in k$)

$$X''^{p}_{j,i} = X'^{p}_{j,i} + r_i\,(X'^{p}_{j,i} - X'^{q}_{j,i}), \text{ If } f(X^p) < f(X^q), \tag{4a}$$
$$X''^{p}_{j,i} = X'^{p}_{j,i} + r_i\,(X'^{q}_{j,i} - X'^{p}_{j,i}), \text{ If } f(X^q) < f(X^p), \tag{4b}$$

(Above equations are for minimization problem, reverse is true for maximization problem)

where $X''^{p}_{j,i}$ is the updated value of $X'^{p}_{j,i}$. The algorithm then accepts $X''^{p}_{j,i}$ if it gives a better function value. More details about the TLBO algorithm and its codes can be found at https://sites.google.com/site/tlborao/.

3. Multi-objective Improved TLBO (MO-ITLBO) algorithm

The proposed MO-ITLBO algorithm is the improved version of the basic TLBO algorithm. In the basic TLBO algorithm, the result of the learners is improved either by a teacher (through the classroom teaching) or by interacting with other learners. However, in the traditional teaching-learning environment the students also learn during the tutorial hours by discussing with their fellow classmates or even by discussing with the teacher. Sometimes the students are self-motivated and try to learn the things by self-learning. Furthermore, the teaching factor in the basic TLBO algorithm is either 2 or 1 which reflects two extreme circumstances where the learner learns either everything or nothing from the teacher. During the course of optimization, this situation results in a slower convergence rate of optimization algorithm. So considering this fact, to enhance the exploration and exploitation capacity, some modifications have been introduced in the basic TLBO algorithm.

The basic TLBO algorithm has been already modified by Rao and Patel (20132013b, 2013c) to improve its performance and applied it to the optimization of thermal systems. In the present work the previous modifications are further enhanced and new modifications are introduced to improve the performance of the algorithm.

3.1. Number of teachers

Population sorting is an important concept used in evolutionary algorithms to avoid the premature convergence. In the basic TLBO algorithm the population sorting mechanism is provided by introducing the multi teacher concept.

In the teacher phase of the TLBO algorithm, the teacher who is a highly learned person will impart the knowledge to students and tries to improve the mean result of the class. In the classical teaching-learning environment, the class contains diverse students (i.e. intelligent, average, below average) that learn from the teacher. Since the teacher is a highly learned person so it is difficult for below average students to cope up with him/her. So, in this situation the teacher has to put more effort to increase the mean result of the learner and even with this effort it might happen that apparent improvements in the results will not be observed.

Below average students can easily cope up with the average students than a highly learned person. So, if the below average students first learn from the average students or intelligent students and then they learn from the highly learned person then their results will improve more effectively, as well as the mean result of the class. Considering this fact, in the basic TLBO algorithm the students are divided into groups based on their results. The best learner of each group acts as a teacher for that group and tries to increase the mean result of his/her group. If the level (i.e. result) of the individual in the group reaches up to the level of the teacher of that group then this individual is assigned to the next group (i.e. next better teacher). The Pseudo code of this modification is given in Fig.1.

```
Initialize the population randomly and evaluate the same.
For RN = 1: Number of runs.
  Rank the evaluated solutions (In ascending order for the minimization problem and in descending order for the
      maximization problem)
  Select the best solution f(X^b). This solution acts as the chief teacher (T_1) of the class. Mathematically, T_1 = f(X^b)
  Select the other teachers (T_s) based on the best solution (i.e. f(X^b))
      T_s = f(X^b) ± r_i × f(X^b) s = 2, 3, ….,N
  (Where, r_i is the random number. If the value of the right side of the above equation is not equal to any of the
  values of the initially evaluated population then the value closer to that is selected from the initial population).
  Once, the teachers are identified, distribute the learners to the teachers based on their fitness value (i.e. result) as,
          For k =1 to Population
             If  T_1 ≤ f(X^k) < T_2
                Assign the learner f(X^k) to teacher 1 (i.e T_1)
             Else If  T_2 ≤ f(X^k) < T_3
                Assign the learner f(X^k)to teacher 2 (i.e T_2)
                .
                .
                .
             Else If  T_{N-1} ≤ f(X^k) < T_N
                Assign the learner f(X^k) to teacher N-1 (i.e T_{N-1})
             Else
                Assign the learner f(X^k) to teacher T_N
             End If
          End For
          Teacher phase
          Learner phase
End For
```

Fig. 1. Pseudo code for selection of teacher and distribution of students

3.2. Adaptive teaching factor

Another modification is related to the teaching factor (T_F) of the basic TLBO algorithm. The teaching factor decides the value of mean to be changed. In the basic TLBO, the decision of the teaching factor is a heuristic step and it can be either 1 or 2. This practice corresponds to the situation where learners learn nothing from the teacher or learn all the things from the teacher respectively. But in actual teaching-learning phenomenon this fraction is not always at its end state for learners but varies in-between also. The learners may learn in any proportion from the teacher. In the optimization algorithm a lower value of T_F allows the finer search in small steps but causes slow convergence. A larger value of T_F speeds up the search but it reduces the exploration capability. Considering this fact the teaching factor is modified as,

$$\left(T_F\right)_{s,i} = \left(\frac{f\left(X^k\right)}{T_s}\right)_i \qquad \text{If } T_s \neq 0 \tag{5a}$$

$$\left(T_F\right)_i = 1 \qquad \text{If } T_s = 0 \tag{5b}$$

where $f(X^k)$ is the result of any learner k associated with group 's' considering all the subjects at iteration i and T_s is the result of the teacher of the same group at the same iteration i. Thus, teaching factor in ITLBO algorithm is the ratio of the result of the learner to the result of the teacher during an iteration. The teaching factor varies automatically during the search depending upon the result of the learner and the teacher. Thus, automatic tuning of T_F improves the performance of the algorithm.

3.3. Learning through tutorial

This modification is based on the fact that students can also learn by discussing with their fellow classmates or even with the teacher during the tutorial hours while solving the assigned tasks. Since the students can increase their knowledge by discussing with the other students or teacher, we incorporate this search mechanism in the teacher phase. So, in the ITLBO algorithm, the learner improved his/her result in the teacher phase through the classroom teaching provided by the teacher along with the discussion with the fellow classmates or teacher during tutorial hours. Mathematically this modification can be modeled as:

$$X'^{k}_{j,i} = (X^{k}_{j,i} + Difference\_Mean_{j,i}) + r_i\,(X^{h}_{j,i} - X^{k}_{j,i}) \quad \text{If } f(X^h) < f(X^k),\ h \neq k, \tag{6a}$$
$$X'^{k}_{j,i} = (X^{k}_{j,i} + Difference\_Mean_{j,i}) + r_i\,(X^{k}_{j,i} - X^{h}_{j,i}) \quad \text{If } f(X^k) < f(X^h),\ h \neq k, \tag{6b}$$

where the first term on the right side indicates the classroom learning and the second term indicates learning through the tutorial.

3.4. Self-motivated learning

In the basic TLBO algorithm, the results of the students are improved either by learning from the teacher or by interacting with the other students. However, it is also possible that students are self-motivated and improve their knowledge by self-learning. Thus, the self-learning aspect to improve the knowledge is considered in the ITLBO algorithm. Since the students learn without the aid of the teacher, we incorporate this search mechanism in the learner phase. Mathematically this modification can be modeled as:

$$X''^{p}_{j,i} = [X'^{p}_{j,i} + r_i\,(X'^{p}_{j,i} - X'^{q}_{j,i})] + [r_i\,(X^{s}_{j,i} - E_F\,X'^{p}_{j,i})], \quad \text{If } f(X'^{p}) < f(X'^{q}) \tag{7a}$$
$$X''^{p}_{j,i} = [X'^{p}_{j,i} + r_i\,(X'^{q}_{j,i} - X'^{p}_{j,i})] + [r_i\,(X^{s}_{j,i} - E_F\,X'^{p}_{j,i})], \quad \text{If } f(X'^{q}) < f(X'^{p}) \tag{7b}$$

($p \neq q$ and $p, q, s \in k$, X^{s}_{j} is the grade of the teacher associated with group 's' in 'j' subject)

where r_i is a random number in the range [0, 1]. E_F is the exploration factor and its value is decided randomly as:

$$E_F = \text{round}\,(1+r_i) \tag{8}$$

The first term on the right side of Eq. (7a) and (7b) indicates the learning by interacting with the other learners and the second term indicates the self-motivated learning.

3.5. External Archive

The main objective of the external archive is to keep a historical record of the non-dominated vectors found along the search process. This algorithm uses a fixed size external archive to keep the best non-dominated solutions that it has found so far. In the proposed algorithm an ε-dominance method is used to maintain the archive. This method has been used widely in multi-objective optimization algorithms to manage the archive. The archive is a space with dimension equal to the number of problem's objectives. The archive is empty at the beginning of the search. In ε-dominance method each dimension of the objective space is divided into segments whose width is ε, so that the objective space is divided into squares, cubes or hyper-cubes for two, three and more than three objectives respectively. If a box that holds the solution(s) can dominate other boxes then those boxes (along with the solution(s) in them) will be removed. Then each box is examined to check if only one non-dominated solution is present, while the dominated ones are eliminated. Finally, if a box still has more than one solution then the solution with the minimum distance from the lower left corner of the box (for minimization problem) and upper right corner (for maximization problem) will stay and the others will be removed. It is observed from the literature that the use of ε-dominance guarantees that the retained solutions are non-dominated with respect to all solutions generated during the execution of the algorithm. The proposed MO-ITLBO algorithm uses the grid based approach for the archiving process which was previously used by MOABC algorithm (Akbari & Ziarati, 2012).
The schematic diagram of the proposed algorithm is shown in Fig. 2.

Initialization

Set Population size, Function evaluation, No. of teachers
Define Optimization problem as, Minimize or Maximize $f(X)$
Initialize Population (i.e. learners, $k=1,2,..n$),
Design variables (i.e. number of subjects offered to the learners, $j=1,2,..m$)
External archive

Selection of teachers

2

Rank the evaluated population i.e. solutions (in ascending order for the minimization problem and in descending order for the maximization problem)

Select the best solution (i.e. the solution obtained the first rank) $f(X^b)$. This solution acts as the chief teacher (T_1) of the class (i.e. $T_1 = f(X^b)$).

Select the other teachers (T_s) based on the best solution (i.e. $f(X^b)$)

$$T_s = f(X^b) \pm r_i * f(X^b) \quad s = 2, 3, \ldots, N$$

(If the equality is not met, select the T_s closer to the value calculated above)

Assign learners to teachers

For *$k=1$ to Population*
If $T_1 \leq f(X^k) < T_2$
Assign the learner $f(X^k)$ to teacher *1* (i.e T_1).
Else If $T_2 \leq f(X^k) < T_3$
Assign the learner $f(X^k)$to teacher *2* (i.e T_2).
.
.
Else If $T_{N-1} \leq f(X^k) < T_N$
Assign the learner $f(X^k)$ to teacher *N-1* (i.e T_{N-1})
Else
Assign the learner $f(X^k)$ to teacher T_N
End If
End For

Teacher phase

Calculate the mean result of each group of learners in each subject (i.e. ($M_{s,j}$)
For *s = 1 to No. of group (i.e No. of teacher)*
For *j = 1 to No. of Design variables*
Calculate the difference between the current mean and the corresponding result of the teacher of that group by utilizing the adaptive teaching factor
$Difference\_Mean_{s,j} = r_i\,(X^s_{j,} - T_F M_{s,j})$
(where X^s_j is the grade of the teacher associated with group 's' in 'j' subject and $M_{s,j}$ is the mean grade of the learner of group 's' in 'j' subject)
End For
End For
Update the learners' knowledge with the help of teacher's knowledge along with the knowledge acquired by the learners' during the tutorial hours.
For *j = 1 to No. of Design variables*
$X'^k_j = (X^k_j + Difference\_Mean_{s,j}) + r_i\,(X^h_j - X^k_j)$ If $f(X^h) < f(X^k)$, $h \neq k$
$X'^k_j = (X^k_j + Difference\_Mean_{s,j}) + r_i\,(X^k_j - X^h_j)$ If $f(X^k) < f(X^h)$, $h \neq k$
End For
If the result has improved
Keep the improved result
Else
Keep the previous result
End If

1

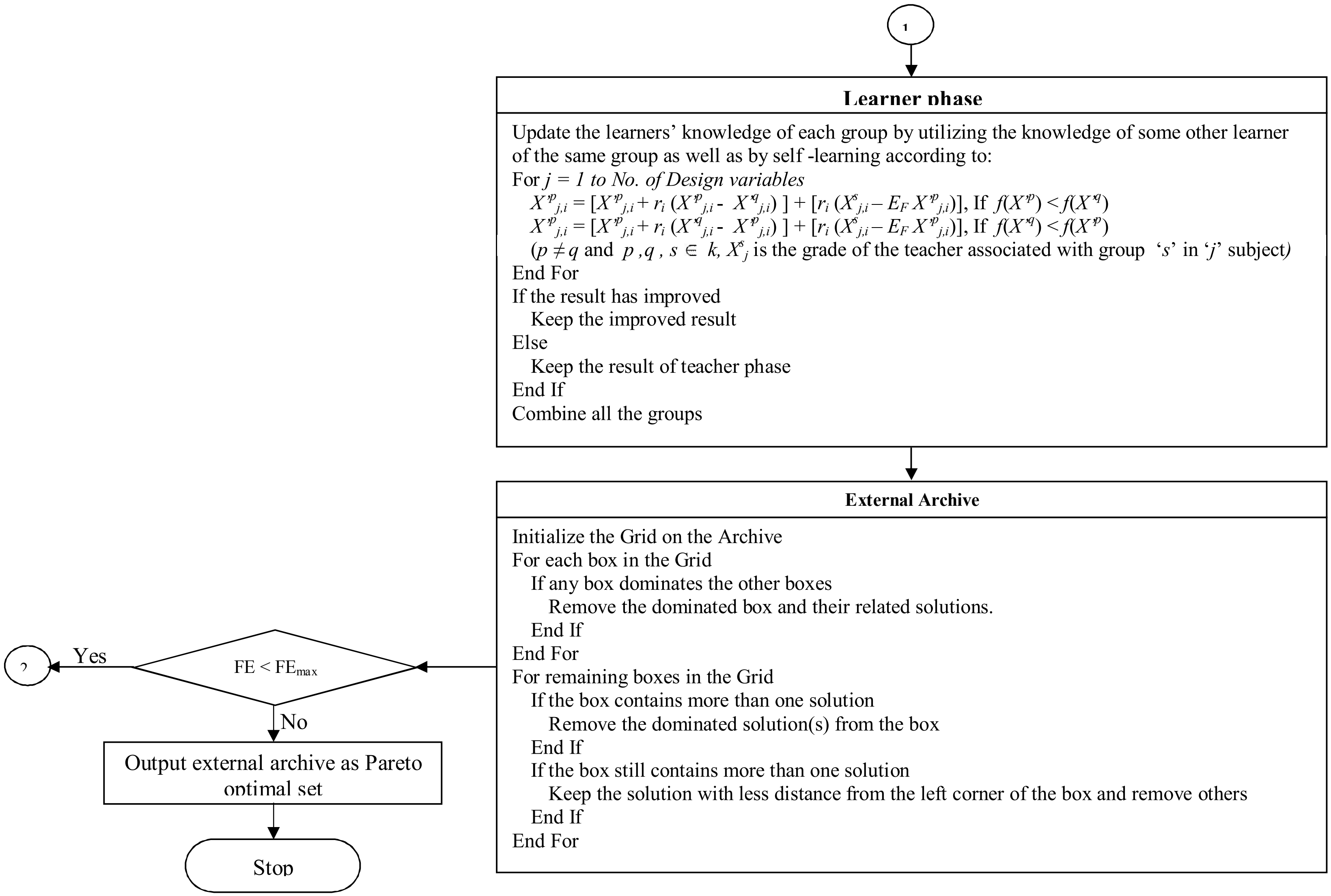

Fig. 2. Schematic diagram of the MO-ITLBO algorithm

Both the teacher phase and learner phase iterate cycle by cycle as shown in Fig. 2 till the termination criterion is satisfied. In the present work, total number of function evaluations is set as termination criterion for the proposed algorithm. At the termination of the algorithm, the external archive found by the algorithm is returned as the output.

The proposed MO-ITLBO algorithm is implemented on both unconstrained and constrained problems. For the constrained optimization problems it is necessary to incorporate any constraint handling techniques within the MO-ITLBO algorithm. In this work, superiority of the feasible solution method (SF) (Qu & Suganthan, 2011) is used to handle the constraints with the proposed algorithm.

At this point it is important to clarify that in the MO-ITLBO algorithm, the solution is updated in the teacher phase as well as in the learner phase. Also, if duplicate solutions are present then they are randomly modified. So the total number of function evaluations in the proposed algorithm is = {(2 × population size × number of generations) + (function evaluations required for the duplicate elimination)}. In the entire experimental work of this paper, the above formula is used to count the number of function evaluations while conducting experiments with proposed algorithm. To demonstrate the effect of the modifications introduced to improve the performance of the TLBO algorithm, a step-by-step comparison of the performance of basic TLBO and the ITLBO algorithms for Rastrigin function is given in Appendix-A. It may be observed that the modifications have improved the performance of the TLBO algorithm.

The next section deals with the experimentation of MO-ITLBO algorithm on various multi-objective unconstrained and constrained functions.

4. Experimental investigation

In this section, the ability of the MO-ITLBO algorithm is assessed by implementing it for the parameter optimization of 20 well defined benchmark functions of CEC 2009 (Zhang et al., 2009). Out of 20 functions, 10 functions are unconstrained (UF1-UF10) and the remaining 10 are constrained functions (CF1-CF10). The UF1-UF7 and CF1-CF7 are two objective benchmark functions while UF8-UF10 and CF8-CF10 are three objective benchmark functions. The detailed mathematical formulations of the considered test functions are given in Zhang et al. (2009). The Pareto front of these functions has many characteristics e.g. some of them are convex while others are concave or some of them are continuous and some others are discontinuous. A common platform is required in the field of optimization to compare the performance of different algorithms for different benchmark functions. For the present work this common platform is provided by CEC 2009. As suggested in this common platform, in the present work the total number of function evaluations is set as 300000 for each test problem in the present work. The MO-ITLBO algorithm is experimented on each function with population size 50 and the number of teachers 4. The proposed algorithm is executed 30 times for each test function and the average results obtained using the proposed algorithm are compared with the results of the other algorithms available in the literature.

4.1. Performance Metric

The inverted generational distance (IGD) measure is used for quantitative assessment of the performance of the proposed algorithm. The IGD measure is defined as: Let P^* be a set of uniformly distributed points along the Pareto front in the objective space. Let A be an approximate set to the Pareto front, the average distance from P^* to A is defined as follows,

$$IGD\left(A,P^*\right)=\frac{\sum_{\tau\in P^*} d\left(\tau,A\right)}{\left|P^*\right|}, \tag{6}$$

where $d(\tau, A)$ is the minimum Euclidian distance between υ and the other points in A. Both diversity and convergence of the approximated set A could be measured using IGD (A, P^*). If P^* has a large number of members to represent the Pareto front precisely. Moreover, to maintain the common platform for comparison, the archive size is adjusted to 100 and 150 for two objective functions and three objective functions respectively.

4.2. Performance analysis of unconstrained benchmark functions

In the first experiment the proposed algorithm is implemented on 10 unconstrained benchmark functions taken from CEC 2009. For each test function the MO-ITLBO algorithm has been executed for 30 times. The result of each benchmark function is presented in Table 1 in the form of best solution, worst solution, mean solution and standard deviation obtained through 30 independent runs of the MO-ITLBO algorithm. The graphical representation of the produced Pareto front with the MO-ITLBO algorithm for UF1-UF10 is shown in Figs. 3(a)-3(j).

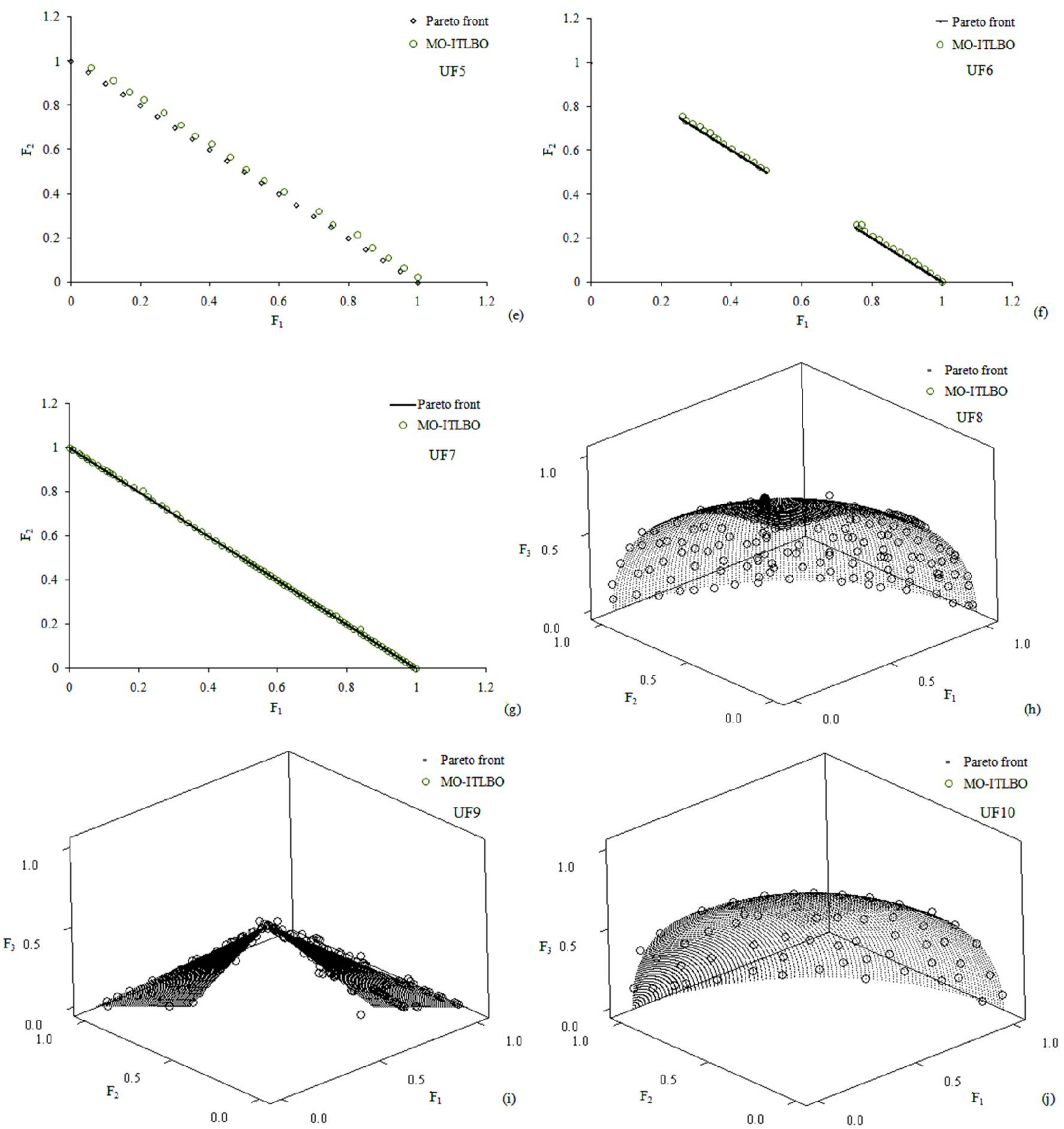

Fig. 3(a)-(j). The Pareto front obtained by the MO-ITLBO algorithm for unconstrained test functions UF1-UF-10

Table 1

IGD values obtained with MO-ITLBO for different unconstrained test functions (UF1-UF10) in 30 independent runs.

Test function	Best	Worst	Mean	SD
UF 1	0.00391	0.00487	0.00421	8.04E-04
UF 2	0.00462	0.00593	0.00519	1.73E-03
UF 3	0.03486	0.06174	0.04681	6.48E-03
UF 4	0.03743	0.04609	0.04378	1.07E-02
UF 5	0.04015	0.09923	0.07482	8.62E-03
UF 6	0.00868	0.03247	0.01144	1.01E-02
UF 7	0.01106	0.08481	0.04127	2.38E-02
UF 8	0.05107	0.05832	0.06126	1.65E-03
UF 9	0.06836	0.20036	0.12379	8.97E-02
UF 10	0.1187	0.18962	0.14714	1.29E-02

Table 2

Comparison of mean IGD values and the standard deviation (SD) obtained with different algorithms for different unconstrained test functions (UF1-UF10) in 30 independent runs

Algorithm	Measure	UF 1	UF 2	UF 3	UF 4	UF 5	UF 6	UF 7	UF 8	UF 9	UF 10
MO-ITLBO	Mean IGD	0.00421	0.00519	0.04681	0.04378	0.07482	0.01144	0.04127	0.06126	0.12379	0.14714
	SD	8.04E-04	1.73E-03	6.48E-03	1.07E-02	8.62E-03	1.01E-02	2.38E-02	1.65E-03	8.97E-02	1.29E-02
MOABC	Mean IGD	0.00618	0.00484	0.0512	0.05801	0.077758	0.06537	0.05573	0.06726	0.0615	0.19499
	SD	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
MTS	Mean IGD	0.00646	0.00615	0.0531	0.02356	0.01489	0.05917	0.04079	0.11251	0.11442	0.15306
	SD	3.49E-04	5.08E-04	1.17E-02	6.64E-04	3.28E-03	1.06E-02	1.44E-02	1.29E-02	2.55E-02	1.58E-02
DMOEADD	Mean IGD	0.01038	0.00679	0.03337	0.04268	0.31454	0.06673	0.01032	0.06841	0.04896	0.32211
	SD	2.37E-03	2.02E-03	5.68E-03	1.39E-03	4.66E-02	1.03E-02	9.46E-03	9.12E-03	2.23E-02	2.86E-01
LiuLi	Mean IGD	0.00785	0.0123	0.01497	0.0435	0.16186	0.17555	0.0073	0.08235	0.09391	0.44691
Algorithm	SD	2.09E-03	3.32E-03	2.4E-02	6.5E-04	2.82E-02	8.29E-02	8.9E-04	7.33E-03	4.71E-02	1.3E-01
GDE3	Mean IGD	0.00534	0.01195	0.10639	0.0265	0.03928	0.25091	0.02522	0.24855	0.08248	0.43326
	SD	3.42E-04	1.54E-03	1.29E-02	3.72E-04	3.95E-03	1.96E-02	8.89E-03	3.55E-02	2.25E-02	1.23E-02
MOEAD	Mean IGD	0.00435	0.00679	0.00742	0.06385	0.18071	0.00587	0.00444	0.0584	0.07896	0.47415
	SD	2.90E-04	1.82E-03	5.89E-03	5.34E-03	6.81E-02	1.71E-03	1.17E-03	3.21E-03	5.32E-02	7.36E-02
MOEADGM	Mean IGD	0.0062	0.0064	0.0429	0.0476	1.7919	0.5563	0.0076	0.2446	0.1878	0.5646
	SD	1.13E-03	4.3E-04	3.41E-02	2.22E-03	5.12E-01	1.47E-01	9.4E-04	8.54E-02	2.87E-02	1.02E-01
NSGAIILS	Mean IGD	0.01153	0.01237	0.10603	0.0584	0.5657	0.31032	0.02132	0.0863	0.0719	0.84468
	SD	7.3E-03	9.11E-03	6.86E-02	5.12E-03	1.83E-01	1.91E-01	1.95E-02	1.24E-02	4.5E-02	1.63E-01
OW	Mean IGD	0.0122	0.0081	0.103	0.0513	0.4303	0.1918	0.0585	0.0945	0.0983	0.743
MOSaDE	SD	1.2E-03	2.3E-03	1.9E-02	1.9E-03	1.74E-02	2.9E-02	2.91E-02	1.19E-02	2.44E-02	8.85E-02
Clustering	Mean IGD	0.0299	0.0228	0.0549	0.0585	0.2473	0.0871	0.0223	0.2383	0.2934	0.4111
MOEA	SD	3.3E-03	2.3E-03	1.47E-02	2.7E-03	3.84E-02	5.7E-03	2.00E-03	2.3E-02	7.81E-02	5.01E-02
AMGA	Mean IGD	0.03588	0.01623	0.06998	0.04062	0.09405	0.12942	0.05707	0.17125	0.18861	0.32418
	SD	1.03E-02	3.17E-03	1.4E-02	1.75E-03	1.21E-02	5.66E-02	6.53E-02	1.72E-02	4.21E-02	9.57E-02
MOEP	Mean IGD	0.0596	0.0189	0.099	0.0427	0.2245	0.1031	0.0197	0.423	0.342	0.3621
	SD	1.28E-02	3.8E-03	1.32E-02	8.35E-04	3.44E-02	3.45E-02	7.51E-04	5.65E-02	1.58E-01	4.44E-02
DECMOSA-	Mean IGD	0.07702	0.02834	0.0935	0.03392	0.16713	0.12604	0.02416	0.21583	0.14111	0.36985
SQP	SD	3.94E-02	3.13E-02	1.98E-01	5.37E-03	8.95E-02	5.62E-01	2.23E-02	1.21E-01	3.45E-01	6.53E-01
OMOEAII	Mean IGD	0.08564	0.03057	0.27141	0.04624	0.1692	0.07338	0.03354	0.192	0.23179	0.62754
	SD	4.07E-03	1.61E-03	3.76E-02	9.67E-04	3.9E-03	2.45E-03	1.74E-03	1.23E-02	6.48E-02	1.46E-01

NA – Not available

In this experiment, the performance of MO-ITLBO algorithm is compared with other well-known optimization algorithms such as MOABC, MTS, DMOEADD, LiuLi Algorithm, GDE3, MOEAD, MOEADGM, NSGAIILS, OWMOSaDE, Clustering MOEA, AMGA, MOEP, DECMOSA-SQP and OMOEAII. Table 2 shows the comparative results of the considered algorithms in the form of mean solution (i.e. mean IGD value) obtained through 30 independent runs.

It is observed from the results that the MO-ITLBO algorithm outperforms the other algorithms for UF1 function. It can also be seen from Fig. 3(a) that the proposed algorithm produced an archive whose members are uniformly distributed over the Pareto front in the two dimension objective space.

The MO-ITLBO algorithm gives a competitive result on UF2 function and obtained the 2nd rank among 15 algorithms. The MOABC algorithm obtained the 1st rank for this function. Fig 3(b) shows the Pareto front of this function obtained by the proposed algorithm.

The MO-ITLBO algorithm obtained the 5th rank on UF3 test function compared to the other 14 algorithms. The best result is obtained by the MOEAD algorithm for UF3 function. For UF4 test function, the MO-ITLBO algorithm obtained the 8th rank among all algorithms. For this function the MTS algorithm produced the best result. Figs. 3(c) and 3(d) show the Pareto front obtained by the proposed algorithm for UF3 and UF4, respectively.

The UF5 and UF6 functions have a discontinuous Pareto front and are relatively hard to solve. The MO-ITLBO algorithm obtained the 3rd and 2nd rank for UF5 and UF6 test problems respectively. The MTS algorithm and the GDE3 algorithm obtained better results than the proposed algorithm for UF5 function while the MOEAD algorithm produced better results than the MO-ITLBO algorithm for UF6 test function. Figs. 3(e) and 3(f) show the graphical representation of the results obtained by the proposed algorithm for UF5 and UF6 test functions respectively. The MO-ITLBO algorithm is placed at the 12th rank on UF7 test function. The Pareto front produced by the proposed algorithm is shown in Fig. 3(g). It is observed from the Fig 3(g) that an entire portion of the Pareto front is not covered by the MO-ITLBO algorithm and as a result this function increases the IDG measure.

The UF8-UF10 are the three objective test functions experimented with the proposed algorithm. The MO-ITLBO, MOABC and MOEAD algorithms show almost comparable performance on UF8 test function. The MOEAD algorithm produces the best result. The proposed algorithm obtained the 2nd rank on UF8 test function. Fig. 3(h) shows the Pareto front produced by the proposed algorithm. It is observed from the Fig. 3(h) that the solution points obtained by the MO-ITLBO algorithm cover considerable part of the objective space.

The MO-ITLBO algorithm obtained the 9th rank among 15 algorithms on UF9 test function. The DMOEADD algorithm produced the best result among all the algorithms on this test function. As shown in Fig. 3(i) that the solution points obtained by the proposed algorithm do not cover the entire objective space, which in turn increases the IGD measure of this test function. On UF10 test function, the MO-ITLBO and the MTS algorithms produced competitive results. The proposed algorithm obtained the 1st rank for optimization of this test function. The quality of the Pareto front produced by the proposed algorithm is shown in Fig. 3(j). It is observed from the Fig. 3(j) that the obtained Pareto front by the proposed algorithm covers larger part of the objective space.

4.2. Performance analysis of constrained benchmark functions

In this experiment the MO-ITLBO algorithm is implemented on 10 constrained benchmark functions of CEC 2009. In this work the superiority of the feasible solution method is used as a constrained handling technique within the MO-ITLBO algorithm. The results of each benchmark function are presented in Table 3 in the form of best solution, worst solution, mean solution and standard deviation obtained through 30 independent runs of the MO-ITLBO algorithm. Table 4 shows the comparative results between the considered algorithms in the form of mean solution (i.e. mean IGD value) obtained through 30 independent runs.

Figs. 4(a)-4(j) show the approximated Pareto front produced by the proposed algorithm for CF1-CF10 functions. The CF1 and CF2 are discontinuous test functions. The MO-ITLBO algorithm obtained the 4th rank for both the test problems. The LiuLi algorithm and the DMOEADD algorithm produced the best results for CF1 and CF2 respectively. The approximated Pareto front produced by the proposed algorithm is shown in Figs. 4(a) and 4(b) for CF1 and CF2 respectively.

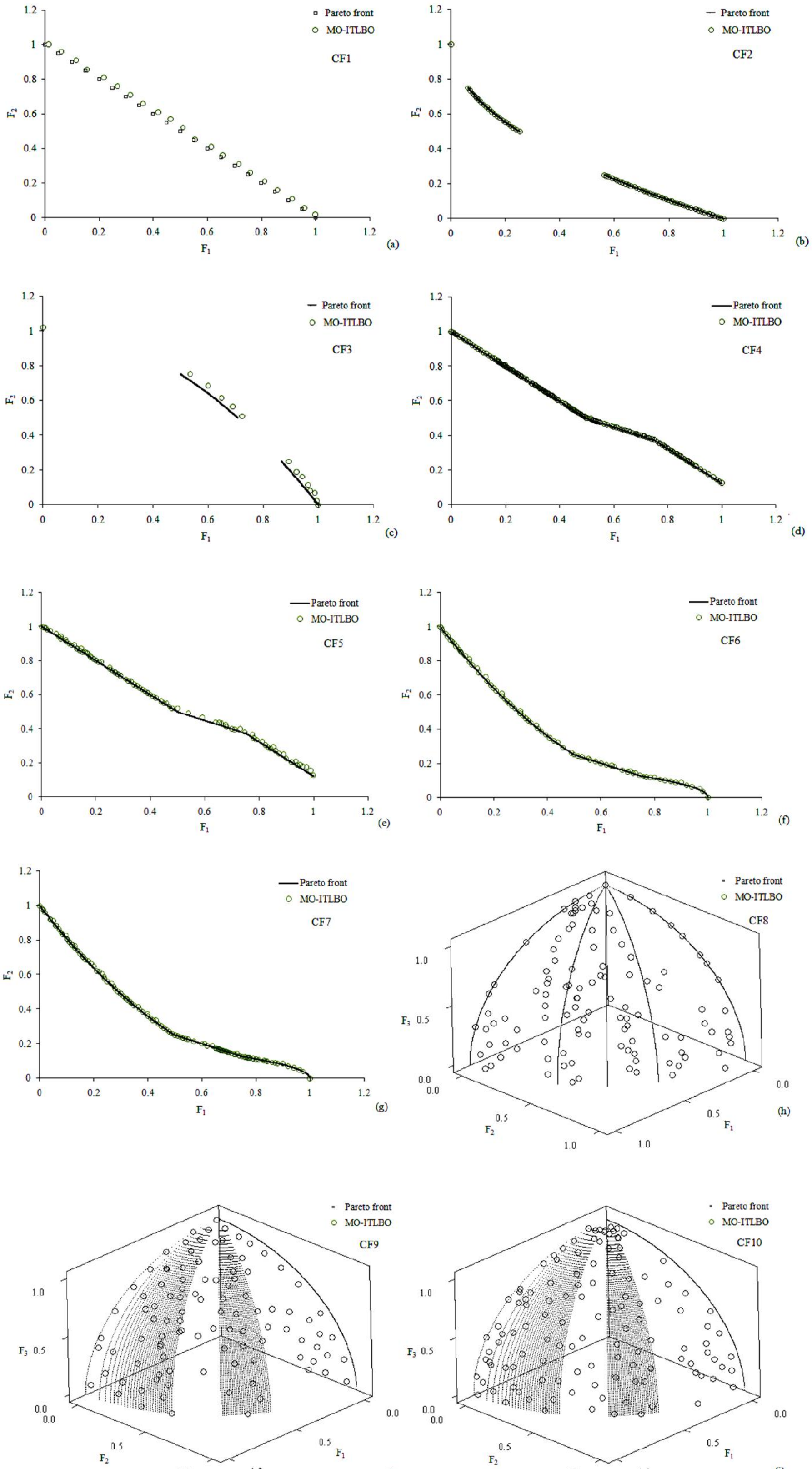

Fig. 4(a)-(j). The Pareto front obtained by the MO-ITLBO algorithm for constrained test functions CF1-CF-10

The test function CF3 has a discontinuous Pareto front and relatively hard to solve. The DMOEADD, MOABC and MO-ITLBO algorithms produced competitive results on CF3 test function. The proposed algorithm obtained the 2nd rank in optimization of this test function. Fig. 4(c) shows that the proposed algorithm has not covered the entire objective space.

Table 3
IGD value obtained with MO-ITLBO for different constrained test functions (CF1-CF10) in 30 independent runs.

	Best	Worst	Mean	SD
CF 1	0.00628	0.01624	0.01007	2.11E-03
CF 2	0.00283	0.01421	0.00924	3.56E-03
CF 3	0.05216	0.09731	0.08242	1.00E-02
CF 4	0.00124	0.00831	0.00518	1.98E-03
CF 5	0.00712	0.09871	0.06789	8.96E-02
CF 6	0.00642	0.01132	0.00916	6.48E-03
CF 7	0.01012	0.03052	0.01916	3.69E-03
CF 8	0.04809	0.15922	0.10482	4.37E-02
CF 9	0.04536	0.05849	0.05018	2.97E-03
CF 10	0.09213	0.36382	0.18341	3.12E-02

Table 4
Comparison of mean IGD values and standard deviation (SD) obtained with different algorithms for different constrained test functions (CF1-CF10) in 30 independent runs

Algorithm		CF1	CF2	CF3	CF4	CF5	CF6	CF7	CF8	CF9	CF10
MO-ITLBO	Mean IGD	0.01007	0.0092	0.08242	0.00518	0.06789	0.00916	0.01916	0.10482	**0.05018**	0.1834
	SD	2.11E-03	3.56E-03	1.00E-02	1.98E-03	8.96E-02	6.48E-03	3.69E-03	4.37E-02	**2.97E-03**	3.12E-02
MOABC	Mean IGD	0.00992	0.01027	0.08621	**0.00452**	0.06781	**0.00483**	**0.01692**	----	----	----
	SD	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
MTS	Mean IGD	0.01918	0.02677	0.10446	0.01109	0.02077	0.01616	0.02469	1.0854	0.08513	**0.1376**
	SD	2.57E-03	1.47E-02	1.56E-02	1.37E-03	2.42E-03	5.99E-03	4.65E-03	2.19E-01	8.19E-03	**9.22E-03**
DMOEADD	Mean IGD	0.01131	**0.0021**	**0.0563**	0.00699	**0.01577**	0.01502	0.01905	**0.0475**	0.1434	0.1621
	SD	2.76E-03	**4.53E-04**	**7.57E-03**	1.46E-03	**6.66E-03**	6.46E-03	6.12E-03	**6.39E-03**	2.14E-02	3.16E-02
LiuLi Algorithm	Mean IGD	**0.00085**	0.0042	0.1829	0.01423	0.10973	0.01394	0.10446	0.06074	0.05054	0.1974
	SD	**1.10E-04**	2.64E-03	4.21E-02	3.29E-03	3.07E-02	2.59E-03	3.51E-02	1.30E-02	3.36E-03	7.60E-02
GDE3	Mean IGD	0.0294	0.01597	0.1275	0.00799	0.06799	0.06199	0.04169	0.1387	0.1145	0.4923
	SD	2.29E-03	7.56E-03	2.39E-02	1.23E-03	1.35E-02	2.69E-02	1.08E-02	5.86E-02	2.21E-02	1.68E-03
MOEADGM	Mean IGD	0.0108	0.008	0.5134	0.0707	0.5446	0.2071	0.5356	0.4056	0.1519	0.3139
	SD	2.50E-03	9.99E-03	7.14E-02	1.01E-01	1.72E-01	1.00E-04	1.00E-01	1.28E-01	4.13E-02	1.04E-01
NSGAIILS	Mean IGD	0.00692	0.01183	0.23994	0.01576	0.1842	0.02013	0.23345	0.11093	0.1056	0.3592
	SD	2.51E-03	1.30E-02	8.58E-02	4.53E-03	6.08E-02	1.74E-02	8.69E-02	3.68E-02	2.93E-02	7.50E-02
DECMOSA-SQP	Mean IGD	0.10773	0.0946	1000000	0.15265	0.41275	0.14782	0.26049	0.17634	0.12713	0.50705
	SD	1.96E-01	2.94E-01	0.00E+00	4.67E-01	5.91E-01	1.25E-01	2.60E-01	6.26E-01	1.46E-01	1.20E+00

NA – Not available

The MO-ITLBO algorithm obtained the 2nd rank among 9 algorithms on CF4 test function. Only the MOABC algorithm produced better results than the proposed algorithm on this test function. Fig. 4(d) shows that the MO-ITLBO algorithm successfully converges to Pareto front with uniform distribution of solution points over the Pareto front.

The MO-ITLBO algorithm obtained the 4th rank on CF5 test function. The DMOEADD algorithm obtained the best result on this test function. The approximated Pareto front produced by the proposed algorithm is shown in Fig. 4(e). It is observed from Fig. 4(e) that despite the good convergence the proposed algorithm has not fully covered the entire objective space.
The MOABC shows the best result and obtained the 1st rank on the CF6 test problem. The MO-ITLBO algorithm produced the competitive results and obtained the 2nd rank on this test problem. The graphical representation of the produced solutions is given in Fig. 4(f).

The MOABC, DMOEADD and MO-ITLBO algorithms have competitive performance on the CF7 test problem. The proposed algorithm achieves the 3rd rank while the MOABC algorithm is placed at the 1st

position. It is observed from Fig. 4(g) that the proposed algorithm shows good convergence with small discontinuity in the produced solutions.

The CF8 is the first three objective constrained test function experimented in this work. The proposed algorithm achieves the 3rd rank among 8 algorithms on this test function. The DMOEADD algorithm obtained the 1st rank on CF8 function. Fig. 4(h) shows an approximated Pareto front produced by the proposed algorithm in three dimension objective space.

The MO-ITLBO, MTS and LiuLi algorithms produced competitive results on the CF9 test problem. The proposed algorithm surpasses the other algorithms and obtained the 1st rank on this test problem. The quality of the solution points produced by the proposed algorithm is shown in Fig. 4(i). It is observed from Fig. 4(i) that the MO-ITLBO algorithm produced an appropriate distribution of the solution points in three dimension objective space.

The MTS algorithm outperforms other algorithms in solving CF10 test function. The proposed algorithm has obtained the 3rd rank among all other algorithms. Fig. 4(j) shows a graphical representation of the produced solutions.

In order to access the overall performance of the MO-ITLBO algorithm among the 15 algorithms in optimizing unconstrained test functions and 10 algorithms in optimizing constrained test functions, the Lexicographic ordering is used. The Lexicographic ordering determines the overall rank of the considered algorithms. For any test function, the algorithm, which gives the best mean IGD value as compared to rest of algorithms obtains the first rank for that test function and the next better performing algorithm occupies the second rank, and so on. In the present work the ranks are given to the considered algorithms for each un-constrained and constrained functions. After that, the average ranking is obtained for each considered algorithm for the unconstrained as well as the constrained functions. The algorithm with minimum average rank is identified as the best algorithm and the first rank is assigned in lexicographic ordering. In the similar way, the next better average rank is identified and the algorithm associated with that rank is placed at the second place in the lexicographic ordering and so on. Table 5 shows the ranking of the considered algorithms in optimizing the unconstrained and constrained test problems separately. It is observed from Table 5 that the MO-ITLBO algorithm obtained the 1st rank among the 15 algorithms in optimization of unconstrained test functions. Similarly, The MO-ITLBO algorithm is the 3rd best algorithm in optimization of constrained test functions.

Table 5
The Lexicographic ordering of the unconstrained and constrained test problems

	Unconstrained functions		Constrained functions
Rank	Algorithm	Rank	Algorithm
1	MO-ITLBO	1	MOABC
2	MTS	2	DMOEADD
2	MOEAD	3	MO-ITLBO
3	MOABC	4	LiuLi Algorithm
3	DMOEADD	5	MTS
4	LiuLi Algorithm	6	NSGAIILS
5	GDE3	6	GDE3
6	AMGA	7	MOEADGM
7	MOEADGM	8	DECMOSA-SQP
8	DECMOSA-SQP		
9	MOEP		
10	NSGAIILS		
11	Clustering MOEA		
12	OW MOSaDE		
13	OMOEAII		

To investigate the results obtained using different algorithms in-depth, a statistical test, known as t-test, is performed in the present work. The t-test is performed on the pairs of the algorithms to identify the differences of significance between the results of different algorithms. In the present work the Modified Bonferroni Correction is adopted while performing the t-test (Karaboga & Akay, 2009). For t-test, the *p*-value for each function is calculated and then the *p*-values are ranked in the ascending order. The inverse ranks are then obtained and the significance ratio is obtained by dividing the significance level (α) by the inverse rank. In the present work the t-test is performed at a significance level of 0.025. The results of each benchmark function obtained through 30 independent runs are used to perform the t-tests. For any function if the difference in the *p*-values of the proposed algorithm and the other algorithm is less than the significance ratio then it indicates the statistically better performance of the proposed algorithm. Table 6 shows the results of t-test where the pairwise comparisons between the proposed algorithm and the other algorithms are given separately for the unconstrained and constrained functions. For any function, '1' indicates that the performance of the MO-ITLBO algorithm is better than its counterpart algorithm and '0' indicates vice versa. Symbol '-' is used where there is no significant performance difference between the MO-ITLBO algorithm and its counterpart algorithm.

Table 6
Pairwise comparison of the MO-ITLBO algorithm with the other algorithms for unconstrained and constrained test functions

Unconstrained Functions

	UF 1	UF 2	UF 3	UF 4	UF 5	UF 6	UF 7	UF 8	UF 9	UF 10
MTS	1	-	-	0	0	1	-	1	-	-
MOEAD	-	1	0	1	1	0	0	0	-	1
MOABC	1	0	-	1	-	1	1	-	0	-
DMOEADD	1	1	0	-	1	1	0	1	0	1
LiuLi Algorithm	1	1	0	-	1	1	0	1	-	1
GDE3	1	1	1	0	0	1	0	1	0	1
AMGA	1	1	1	-	1	1	-	1	1	1
MOEADGM	1	1	-	-	1	1	0	1	1	1
DECMOSA-SQP	1	1	1	0	1	1	0	1	-	1
MOEP	1	1	1	-	1	1	0	1	1	1
NSGAIILS	1	1	1	1	1	1	0	1	1	1
Clustering MOEA	1	1	1	1	1	1	0	1	1	1
OW MOSaDE	1	1	1	1	1	1	1	1	0	1
OMOEAII	1	1	1	-	1	1	0	1	1	1

Constrained Functions

	CF 1	CF 2	CF 3	CF 4	CF 5	CF 6	CF 7	CF 8	CF 9	CF 10
MOABC	-	-	1	0	-	0	-	NA	NA	NA
DMOEADD	-	0	0	1	0	1	-	0	1	-
LiuLi Algorithm	0	0	1	1	-	1	1	0	-	-
MTS	1	1	1	1	0	1	1	1	1	0
NSGAIILS	0	-	1	1	1	1	1	-	1	1
GDE3	1	1	1	1	-	1	1	-	1	1
MOEADGM	-	-	1	1	1	1	1	1	1	1
DECMOSA-SQP	1	1	1	1	1	1	1	-	1	1

'1' indicates that the performance of the MO-ITLBO algorithm is better than its counterpart algorithm
'0' indicates that the performance of the counterpart algorithm is better than the MO-ITLBO algorithm
'-' indicates that there is no significance difference in performance between the MO-ITLBO algorithm and its counterpart algorithm
NA indicates that the results are not available

In order to identify the convergence of the MO-ITLBO algorithm to the optimal Pareto front, an unconstrained function (UF1), and constrained function (CF4) are considered for the experimentation. Figs. 5 and 6 show the convergence of the unconstrained and constrained test function respectively at intervals of 50000 function evaluations. It is observed from the Figs. 5 and 6 that with the increase in the function evaluations the approximated solution points in the two dimensional objective space are also increased. Moreover, the distribution of the solution points becomes uniform as the number of function evaluations proceeds. Both these points indicate that the performance of the MO-ITLBO algorithm is continuously improved throughout the function evaluations.

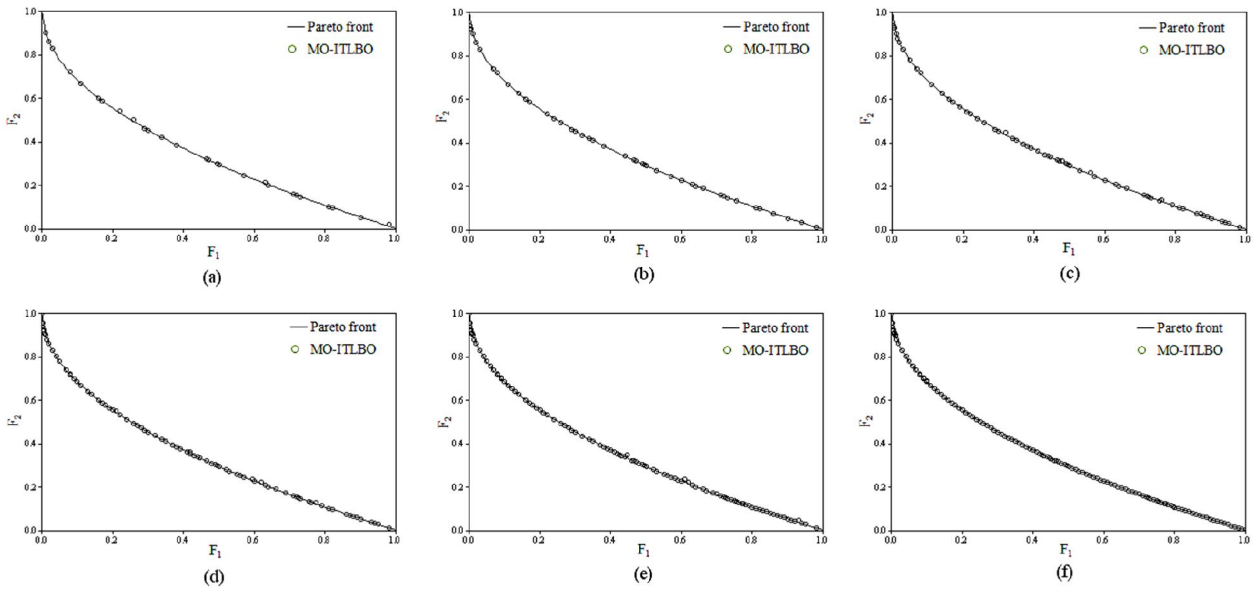

Fig. 5. Convergence of the MO-ITLBO algorithm for the unconstrained test function UF1 (a) 50000 FE (b) 100000 FE (c) 150000 FE (d) 200000 FE (e) 250000 FE and (f) 300000 FE

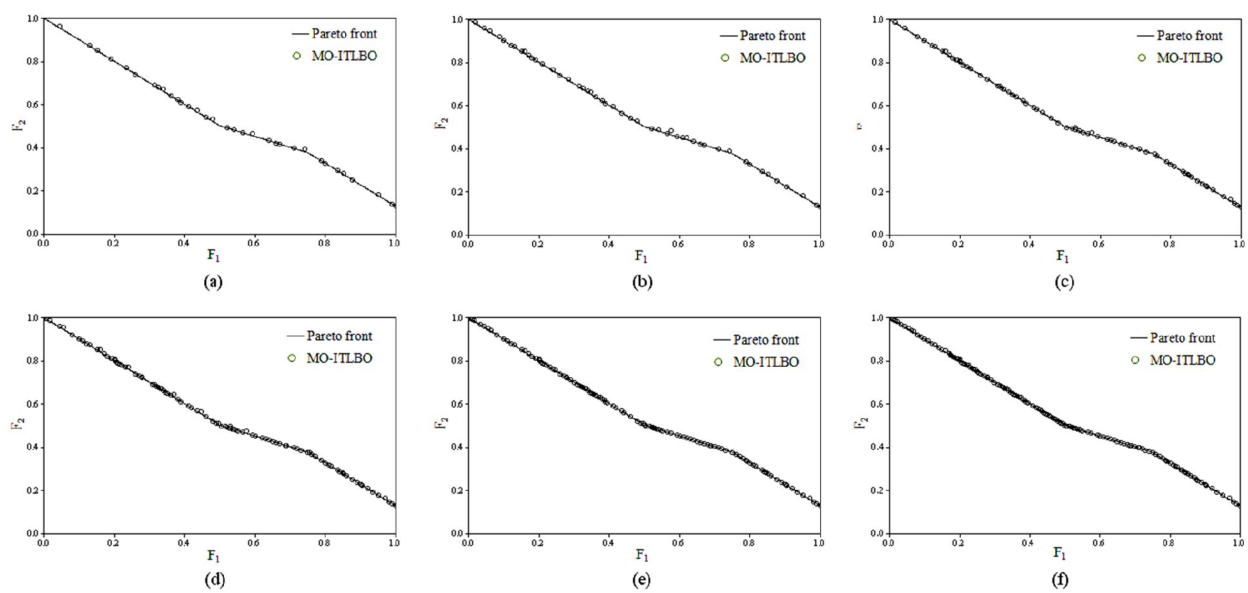

Fig. 6. Convergence of MO-ITLBO algorithm for the constrained test function CF4 (a) 50000 FE (b) 100000 FE (c) 150000 FE (d) 200000 FE (e) 250000 FE and (f) 300000 FE

5. Conclusions

In this work an improved TLBO algorithm has been adapted to handle the MOO problems. Two new search mechanisms are introduced in the TLBO algorithm in the form of tutorial training and self-motivated learning. Moreover, the teaching factor of the basic TLBO algorithm is modified and an adaptive teaching factor is introduced. Furthermore, more than one teacher is introduced for the learners in the proposed algorithm. The MO-ITLBO algorithm used a fixed size archive to maintain the good solutions obtained during every iteration and a grid-based approach to control the diversity over the external archive. The performance of the MO-ITLBO algorithm is evaluated by conducting experiments on a range of multi-objective unconstrained and constrained test problems and the results obtained using the MO-ITLBO algorithm are compared with that of the other state-of-the-art algorithms available in the literature. The experimental results have shown satisfactory performance of the MO-ITLBO algorithm for the MOO problems. The proposed algorithm can be easily customized to suit the optimization of any problem involving multiple objectives. Hence, the proposed optimization algorithm may be tried by the researchers of the industrial engineering field.

References

Agrawal, S., Dashora, Y., Tiwari, M. K., & Son, Y. J. (2008). Interactive particle swarm: A pareto-adaptive metaheuristic to multiobjective optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *38*(2), 258-277.

Akbari, R., & Ziarati, K. (2012). Multi-Objective bee swarm optimization.*International Journal of Innovative Computing Information and Control*, *8*(1B), 715-726.

Chen, C.M., Chen, Y. & Zhang, Q. (2009). Enhancing MOEA/D with guided mutation & priority update for multi-objective optimization. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 209–216.

Coello Coello, C.A., Lamont, G.B. & Van Veldhuizen, D.A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer-Verlag.

Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation,*, *8*(3), 256-279.

Deb, K., Mohan, M. & Mishra, S. (2005). Evaluating the epsilon-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. *Evolutionary Computations,* 13(4), 501–525.

Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. (2002). A fast & elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transaction on Evolutionary Computations.*, 6(2), 182-197.

Huang, V.L., Zhao, S.Z., Mallipeddi, R. & Suganthan, P.N. (2009). Multi-objective optimization using self-adaptive differential evolution algorithm. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 190–194.

Liu, H. & Li, X. (2009). The multi-objective evolutionary algorithm based on determined weight & sub-regional search. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 1928–1934.

Liu, M., Zou, X., Chen, Y. & Wu, Z. (2009). Performance assessment of DMOEA-DD with CEC 2009 MOEA competition test instances. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 2913-2918.

Gao, S., Zeng, S., Xiao, B., Zhang, L., Shi, Y., Tian, X., Yang, Y., Long, H., Yang, X., Yu, D. & Yan, Z. (2009). An orthogonal multi-objective evolutionary algorithm with lower-dimensional crossover. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 1959–1964.

Hedayatzadeh, R,, Hasanizadeh, B., Akbari, R. & Ziarati, K. (2010). A multi-objective artificial bee colony for optimizing multi-objective problems. *In: 3rd International Conference on Advanced Computer Theory & Engineering (ICACTE)*, 5, 271–281.

Karaboga, D., Akay, B. (2009). A comparative study of Artificial Bee Colony algorithm, *Applied Mathematics and Computations,* 214, 108–132.

Kukkonen, S. & Lampinen, J. (2009). Performance assessment of generalized differential evolution with a given set of constrained multi-objective test problems. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 1943–1950.

Leong, W.F. & Yen, G.G. (2008). PSO-based multi-objective optimization with dynamic population size & adaptive local archives. *IEEE Transaction on Systems and Man Cybernetics*, 38(5), 1270–1293.

Mostaghim, S. & Teich, J. (2004). Covering Pareto-optimal fronts by sub swarms in multi-objective particle swarm optimization. *In: 2004 IEEE Congress on Evolutionary Computation*, 19-23 June, Portl&, USA, 1404–1411.

Qu, B.Y. & Suganthan, P.N. (2011). Constrained multi-objective optimization algorithm with ensemble of constraint handling methods. *Engineering Optimization,* 43(4), 403-434.

Qu, B.Y. & Suganthan, P.N. (2009). Multi-objective evolutionary programming without non-domination sorting is up to twenty times faster. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 2934–2939.

Rao, R.V., Savsani, V.J. & Vakharia, D.P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer Aided Design*, 43(3), 303-315.

Rao, R.V., Savsani, V.J. & Vakharia, D.P. (2012). Teaching-learning-based optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences*, 183(1), 1-15.

Rao, R.V. & Patel, V. (2012). An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. *International Journal of Industrial Engineering Computations*, 3(4), 535-560.

Rao, R., & Patel, V. (2013). Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *4*(1), 29-50.

Rao, R.V. & Patel, V. (2013b). Multi-objective optimization of two stage thermoelectric cooler using a modified teaching–learning-based optimization algorithm. *Engineering Applications of Artificial Intelligence,* 26(1), 430-445.

Rao, R.V. & Patel, V. (2013c). Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm. *Applied Mathematical Modeling,* doi.org/10.1016/j.apm.2012.03.043.

Srinivasan, D. & Seow, T.H. (2003). Particle swarm inspired evolutionary algorithm (ps-ea) for multi-objective optimization problem. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 2292–2297.

Sindhya, K., Sinha, A., Deb, K. & Miettinen, K. (2009). Local search based evolutionary multi-objective optimization algorithm for constrained & unconstrained problems. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 2919–2926.

Tiwari, S., Fadel, G., Koch, P. & Deb, K. (2009). Performance assessment of the hybrid archive-based micro genetic algorithm on the CEC09 test problems. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 1935-1942.

Tseng, L.Y. & Chen, C. (2009). Multiple trajectory search for unconstrained/constrained multi-objective optimization. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 1951–1958.

Van Veldhuizen, D.A. (1999). Multi-objective evolutionary algorithms: classifications, analyses & new Innovations. *Evolutionary Computations*, 8(2), 125-147.

Wang, Y., Dang, C., Li, H., Han, L. & Wei, J. (2009). A clustering multi-objective evolutionary algorithm based on orthogonal & uniform design. *In: 2009 IEEE Congress on Evolutionary Computation*, 18-21 May, Trondheim, Norway, 2927–2933.

Yen, G. G., & Leong, W. F. (2009). Dynamic multiple swarms in multiobjective particle swarm optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *39*(4), 890-911.

Zamuda, A., Brest, J., Boskovic, B. & Zumer, V. (2009). Differential evolution with self adaptation & local search for constrained multi-objective optimization. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 192–202.

Zhang, Q., Liu, W. & Li, H. (2009). The performance of a new version of MOEA/D on CEC09 unconstrained mop test instances. *In: 2009 IEEE Congress on Evolutionary Computation,* 18-21 May, Trondheim, Norway, 203–208.

Zhang, Q., Zhou, A., Zhao, S., Suganthan, P.N., Liu, W. & Tiwari, S. (2009). Multi-objective optimization test instances for the congress on evolutionary computation (CEC 2009) special session & competition. *Working Report CES-887*. University of Essex, UK.

Zeng, F., Decraene, J., Low, M.Y.H., Hingston, P., Wentong, C., Suiping, Z. & Ch&ramohan, M. (2010). Autonomous bee colony optimization for multi-objective function. *In: 2010 IEEE Congress on Evolutionary Computation*, 18-23 July, Barcelona, Spain, 1–8.

Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N. & Zhang Q. (2011). Multi-objective evolutionary algorithms: a survey of the state-of-the-art. *Swarm & Evolutionary Computation*, 1(1), 32–49.

Zou, W., Zhu, Y., Chen, H. & Shen, H. (2011). A novel multi-objective optimization algorithm based on artificial bee colony. *In: Genetic & Evolutionary Computation Conference (GECCO'11),* 12-16 July, Dublin, Ireland, 103–104.

APPENDIX - A:

Stepwise comparison of the TLBO and the ITLBO algorithms for Rastrigin function for demonstration.

Step 1: Define the optimization problem: Minimize $f(X)$ = Minimize $f(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$

Basic TLBO

Step 2: initialize the optimization parameters

- Population size = 10
- Number of design variables = 2
- Limits of design variables = $-5.12 \le x_{,1,2} \le 5.12$

Step 3: Initialize the population by random generation and evaluate them

x_1	x_2	$f(X)$
-1.0483	0.6283	18.8781
-1.9397	-2.742	22.6028
-1.6623	0.8464	23.0891
0.8974	2.2713	19.2487
0.6677	1.0151	16.4803
1.7232	1.0223	15.8380
-1.1075	2.7974	18.3765
-1.1479	-1.1465	10.5354
-1.9776	-3.776	26.7694
0.6497	2.7974	31.3082

Step 4: Teacher Phase

Calculate the mean of the population column-wise which will give the mean for the particular subject as,

$x_1 \quad x_2$

M_j = [-0.49453, 0.37137]

The best solution will act as the teacher for that iteration

$X_{teacher} = X_{f(X)minimum}$
= [-1.1479 -1.1465]

The teacher will try to shift the mean from M_j towards $X_{,teacher}$, and the difference between the two means is expressed as

$Difference\_Mean_j = r_i\,(X^b_j - T_F M_j)$

Value of T_F is randomly selected as 1 or 2. The obtained difference is added to the current solution to update its values. For example, taking T_F=2 leads to the following.

$X'^k_j = X^k_j + Difference\_Mean_j$,

Improved TLBO

Step 2: initialize the optimization parameters

- Population size = 10
- Number of design variables = 2
- Number of teacher = 2
- Limits of design variables = $-5.12 \le x_{,1,2} \le 5.12$

Step 3: Initialize the population by random generation and evaluate them

x_1	x_2	$f(X)$
-1.0483	0.6283	18.8781
-1.9397	-2.742	22.6028
-1.6623	0.8464	23.0891
0.8974	2.2713	19.2487
0.6677	1.0151	16.4803
1.7232	1.0223	15.8380
-1.1075	2.7974	18.3765
-1.1479	-1.1465	10.5354
-1.9776	-3.776	26.7694
0.6497	2.7974	31.3082

Step 4: Selection of teachers and distribution of learners to teachers

Rank the evaluated solutions in ascending order and select the best solution as chief teacher (T_1). In the present case, [-1.1479 -1.1465] with objective function value [10.5354] acts as the chief teacher.
Based on the chief teacher, the other teacher is selected as,
T_2= 10.5354+ (rand*10.5354) which gives T_2= 21.57. So, select the solution whose value is nearer to the calculated value of T_2.
Hence T_2=22.6028.
Now the distribution of the learners to the teachers gives the following two groups:

Group 1

x_1	x_2	$f(X)$
-1.1479	-1.1465	10.5354
0.8974	2.2713	19.2487
1.7232	1.0223	15.8381
-1.1075	2.7974	18.3766
0.6677	1.0151	16.4804
-1.0483	0.6283	18.8781

Group 2

x_1	x_2	$f(X)$
-1.9397	-2.742	22.6029
-1.6623	0.8464	23.0892
-1.9776	-3.776	26.7694
0.6497	2.7974	31.3083

Step 5: Teacher Phase with tutorial training

For each group, calculate the mean of the population column-wise, which will give the mean for the particular subject as,

Group 1: $x_1 \quad x_2$ — M_j = [-0.0025, 1.0979]

Group 2: $x_1 \quad x_2$ — M_j = [1.2324, -0.7185]

The best solution will act as a teacher for the respective group.

x_1	x_2	$f(X)$
-1.0653	0.3893	19.7719
-1.9567	-2.981	13.1827
-1.6793	0.6074	35.355
0.8804	2.0323	7.81
0.6507	0.7761	25.2765
1.7062	0.7833	24.2417
-1.1245	2.5584	30.0546
-1.1649	-1.3855	25.6415
-1.9946	-4.015	20.1399
0.6327	2.5584	43.0446

X_{new} is accepted if it gives the better function value.

x_1	x_2	$f(X)$
-1.0483	0.6283	18.8781
-1.9567	-2.981	13.1826
-1.6623	0.8464	23.0891
0.8804	2.0322	7.8092
0.6677	1.0151	16.4803
1.7232	1.0223	15.838
-1.1075	2.7974	18.3765
-1.1479	-1.1465	10.5354
-1.9946	-4.015	20.1404
0.6497	2.7974	31.3082

Group 1 $T_1 = [\,-1.1479\ -1.1465]$ Group 2 $T_2 = [\,-1.9397\ -2.742]$

In each group, the teacher will try to shift the mean from M_j towards T_1 *or* T_2 and the difference between the two means is expressed as,

$$Difference\_Mean_j = r_i\,(X^b_j - T_F M_j)$$

Where T_F is the adaptive teaching factor and is calculated by using equation (5). The obtained difference is added to the current solution along with the tutorial training to update its values as,

$$X'^k_j = (X^k_j + Difference\_Mean_j) + rand * (X^h_j - X^k_j).$$

Group 1

x_1	x_2	$f(X)$
-2.9197	-4.7826	40.8
0.0554	2.1032	7.0189
1.2254	-0.9316	11.7126
-2.9606	2.591	34.2591
0.1357	0.1001	5.3634
-1.9492	-2.1025	10.7082

Group 2

x_1	x_2	$f(X)$
-2.542	-3.1222	38.6202
-1.9554	-0.9287	6.0936
-2.0882	-4.0048	21.8626
1.4506	1.8994	27.1887

X_{new} is accepted if it gives the better function value.

Group 1

x_1	x_2	$f(X)$
-1.1479	-1.1465	10.5354
0.0554	2.1032	7.0189
1.2254	-0.9316	11.7126
-1.1075	2.7974	18.3766
0.1357	0.1001	5.3634
-1.9492	-2.1025	10.7082

Group 2

x_1	x_2	$f(X)$
-1.9397	-2.742	22.6029
-1.9554	-0.9287	6.0936
-2.0882	-4.0048	21.8626
1.4506	1.8994	27.1887

Step 5: Learner Phase

In this phase, the learners increase their knowledge with the help of their mutual interaction. The related mathematical expression is explained under sub-section 2.2 . Obtain X_{new} after the learner phase.

x_1	x_2	$f(X)$
-1.09416	-2.8838	13.804
-1.9567	-2.9810	13.1826
-1.16771	0.785	14.8464
0.8804	2.0322	7.8092
2.0797	1.0886	8.2012
2.1942	1.0390	12.6876
-1.1075	2.7974	18.3765
-1.1479	-1.1465	10.5354
-1.9946	-4.0150	20.1404
0.6497	2.7974	31.3082

This completes one iteration and the population goes to the next iteration.

Step 6: Learner Phase with self-motivated learning

In this phase, learners increase their knowledge with the help of their mutual interaction along with the self motivation. The related mathematical expression is explained under sub-section 3.4. Obtain X_{new} after this phase.

Group 1

x_1	x_2	$f(X)$
-1.1479	-1.1465	10.5354
0.0554	2.1032	7.0189
1.2254	-0.9316	11.7126
-0.7779	2.9637	17.9465
0.1357	0.1001	5.3634
-0.119	1.9764	6.7049

Group 2

x_1	x_2	$f(X)$
-1.7732	1.0051	12.7620
-1.9554	-0.9287	6.0936
-1.8735	0.2386	15.8830
-1.8647	-0.1567	11.4123

This completes one iteration. Now both the groups are to be merged and this population goes to the next iteration.