Chapter 25
A Modified Teaching-Learning-Based Optimization Algorithm for Traveling Salesman Problem

Somnath Maji, Santanu Mondal, Samir Maity, Debasis Giri, and Manoranjan Maiti

Abstract In this investigation, we propose a modified teaching-learning-based optimization algorithm (mTLBO) for solving the traveling salesman problems. We design an mTLBO with Boltzmann selection, novel upgradation strategy in the teaching phase, and interactive group-based crossover for learners in the learning phase. In the teaching phase, we focus on different learning abilities of different subjects of individual learners and in the learning phase, learners are randomly divided to form different groups; it helps to maintain the diversity of the population and to avoid premature convergence. The proposed algorithm is tested against benchmark functions from TSPLIB. The results are compared with the proposed mTLBO, TLBO and standard Genetic Algorithm with Roulette wheel selection, cyclic crossover and random mutation. The effectiveness of the proposed algorithm is shown through statistical test ANOVA.

S. Maji (✉)
Department of Computer Science and Engineering, MAKAUT, Kolkata, India
e-mail: somnathmajivucs@gmail.com

S. Mondal
The University of Burdwan, Burdwan, India
e-mail: santanu.civil1994@gmail.com

S. Maity
Data Science, University of Kalyani, Nadia, India
e-mail: maitysamir13@gmail.com

D. Giri
Department of Information Technology, MAKAUT, Kolkata, India
e-mail: debasis_giri@hotmail.com

M. Maiti
Department of Mathematics, Vidyasagar University, West Bengal, India
e-mail: mmaiti2005@yahoo.co.in

S. Bhattacharyya et al. (eds.), *Human-Centric Smart Computing*, Smart Innovation, Systems and Technologies 316, https://doi.org/10.1007/978-981-19-5403-0_25

25.1 Introduction

Traveling salesman problem (TSP) is the most popular discrete optimization problem. The main aim of TSP is to obtain minimum distance/time/cost during the tour, starting from one city visiting all the city exactly once and returning to the starting city. As the TSP is NP-hard problem in nature, to solve this problem, several heuristic algorithms are used such as GA [1], ACO [2], FWA [3], PSO [4], etc. In recent years, mimicking the process of teaching-learning, Rao et al. [5] proposed a novel meta-heuristic teaching- learning-based optimization (TLBO) algorithm. There are two main phases in TLBO: the teaching and learning phases. In the teaching and learning phase, learners always try to improve themselves through the experience of the teacher and peer group. In practice, a learner cannot learn all the subjects equally and constantly compete with his peers. Inspired by these phenomena, we design TLBO in such a way that in the teaching phase, they upgrade themselves differently from different subjects. Also, an interactive group-based learning strategy is used to improve learners in the learning phase.

Mainly there are a lot of investigations in continuous optimization problems ([6, 7], etc.) through TLBO. Few researchers used TLBO in combinatorial optimization problems ([8–10], etc.). Li et al. [8] investigated flow-shop rescheduling problems using discrete TLBO. They explained the canonical form of TLBO. Then they updated the learner with the teaching phase-I, II and learning phase-I, II. Also, they consider for upgradation of learner IG-local search technique and repair newly generated learner. Wu et al. [10] solved TSP with crossover and reverse mutation techniques in the teaching phase and the same crossover operation in the learning phase. Saharan et al. [9] investigated discrete version of TLBO using Lehmer code (representation of permutation for a new mean of the class) and partially map crossover technique.

In the teaching phase, they all considered the mean of the class for upgradation of the learner, but in this investigation, we focus on individual differences between the learner and teacher and upgrade the learner differently for different subjects. In the learning phase, learners learn through interaction among them. They are always competitive, so we introduced novel interactive group-based crossover for upgradation the learner. As learners are selected randomly to form the groups, it helps to maintain the diversity of the population. The above two phases help to maintain a good balance between exploration and exploitation of the TLBO.

In this investigation, our proposed modified TLBO (mTLBO), combined with Boltzmann selection, novel upgradation strategy in the teaching phase and interactive group-based crossover for learners in the learning phase. The TSPLIB benchmark instances (cf. [11]) are solved using mTLBO. The ANOVA test is done for measuring the performance of the proposed mTLBO.

Novelties in this investigation are as follows:

- Modified TLBO is proposed
- Introducing new upgradation strategy in the teaching phase
- A Novel interactive group-based crossover in the learning phase
- Boltzmann selection is incorporated

- Statistical test, ANOVA is performed.

The paper is structured as follows: Sect. 25.1 presents a brief introduction. Sect. 25.2 explains details process of TLBO. Computational experiments are available in Sect. 25.3. Conclusion and future scope are presented in Sect. 25.4

25.2 Proposed Modified Teaching-Learning-Based Optimization Algorithm (mTLBO)

TLBO mimics the learning process of a teacher and a population of learners. In TLBO, teacher is represented by the best learner of the population. Here two phases are performed- teaching phase and learning phase (cf. Fig. 25.1).

Here modification is done in the teaching phase with a new upgradation strategy and learning phase with interactive group-based crossover.

25.2.1 Representation

A learner of TLBO can be expressed as $X_i = \{x_{i1}, x_{i2}, \ldots, x_{iN}\}, i = 1, 2 \ldots, M$, where, $x_{i1}, x_{i2}, \ldots, x_{iN}$ are different subjects/visiting nodes. An example of arbitrary learner/path (one-dimensional vector), with 5 different subjects/nodes maintaining TSP conditions is show in Fig. 25.2.

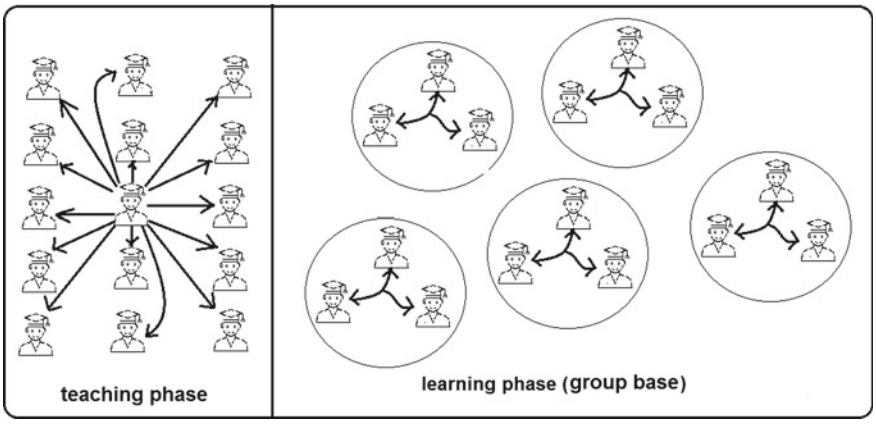


Fig. 25.1 Schematic diagram of TLBO

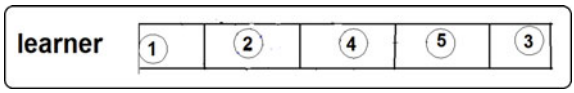


Fig. 25.2 Learner (path with 5 nodes)

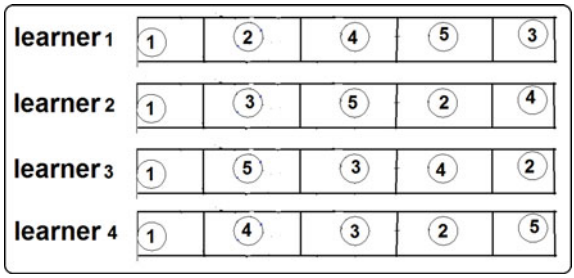


Fig. 25.3 Initialization with 4 learner (5 nodes each)

25.2.2 Initialization

For N cities, construct M routing paths, so total learner, i.e., population size = M. Here N represents the different subjects and M routing path represents different learners. So path/learner (total M paths) are randomly generated maintaining the TSP conditions. Assume, $f(X_i)$ is the fitness function of ith learner ($i = 1, 2, \ldots, M$). An example of 4 learners with 5 nodes is shown in Fig. 25.3.

25.2.3 Boltzmann Selection

We first calculate the Boltzmann-Probability [12] for each learner of the initial population using

$$p_{\mathrm{B}} = \mathrm{e}^{((g/G)*(f_{\min} - f(X_i))/KT)},$$

where, p_{B} = probability of each learner, $T = T_0(1-a)^k$, $k = (1 + C * \mathrm{rand}[0, 1])$, $C = \mathrm{rand}[1, 100]$, g = current generation number, $G =$ max-gen, $T_0 = \mathrm{rand}[50, 140]$, $a = \mathrm{rand}[0, 1]$, $f(X_i)$ is the objective function, $f_{\min} = \min f(X_i), i = 1, 2, \ldots, M$.

First, a predefined value, say probability of selection (p_s) is assigned. For each learner, $r \in [0, 1]$ is generated randomly. If $r < p_{\mathrm{s}}$ or $r < p_{\mathrm{B}}$, then the corresponding learner is selected for the teaching phase. Otherwise, learner corresponding to $f_{\min}$ is selected for the teaching phase.

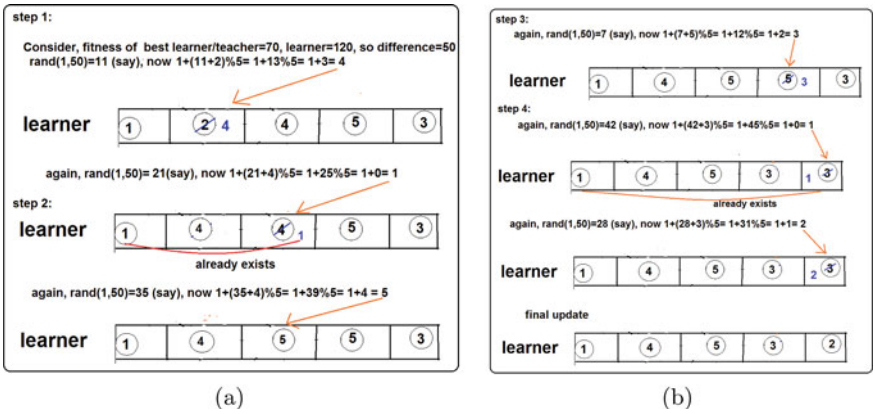


Fig. 25.4 Upgradation strategy of learner

25.2.4 Upgraded Teaching Phase

In mTLBO, we introduce an upgradation strategy in the teaching phase as shown in Fig. 25.4 and define through the Algorithm 1. Consider a TSP with 5 nodes, and all learners are started with node 1. First, evaluate the best learner, i.e., teacher, based on fitnesses from the population. Then choose each learner and try to upgrade using the best learner/teacher through the following steps.

step 1: Let the fitness of the best learner = 70. Choose an arbitrary learner (1-2-4-5-3-1), the fitness of this learner = 120 (say). Evaluate the difference between the learner and teacher (= 50). For 2nd node position (here for node 2), generate a random number (say 11) using rand function between (1,50). Now, $x_{ij} = 1 + (\text{rand}(1, \text{diff}(i)) + x_{ij})\ \%N = 1 + (11 + 2)\%5 = 1 + 13\%5 = 4$ So, learner updates his 2nd position with 4.

step 2: For 3rd node position (here for node 4), rand(1,50) = 21(say), now $x_{ij} = 1 + (21 + 4)\%5 = 1 + 25\%5 = 1$. But 1 is already exists in 1st position (as in TSP duplicate node is not allowed). So, again rand(1,50) = 35(say), now x_{ij}=1+(35+4)%5 = 1+39%5 =5. Hence, learner update his 3rd position with 5.

The remaining steps (step 3,4,5) are repetitively performed.

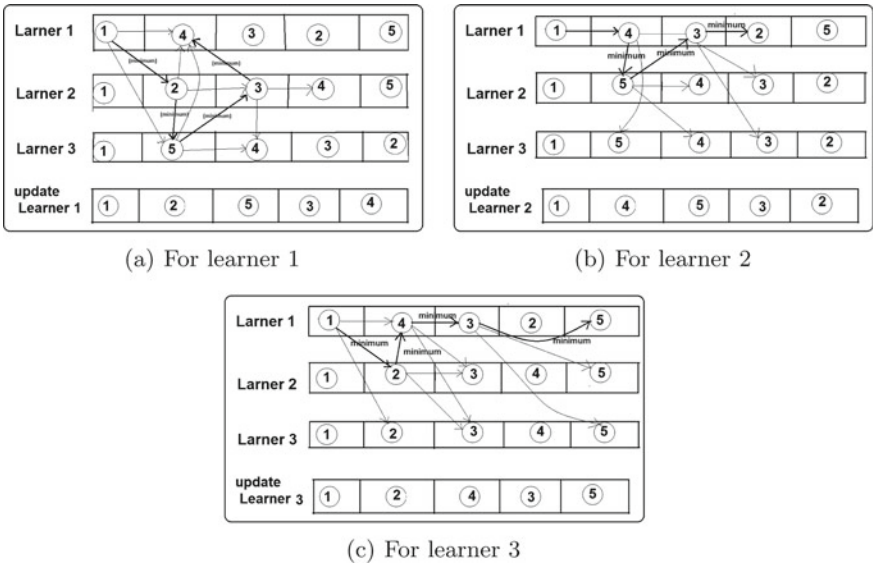
Finally, the updated learner becomes 1-4-5-3-2-1. Now, store/choose a better-fitted learner between previous and updated learners.

25.2.5 Interactive Group-Based Learning

The learning phase, inspired by real scenarios among learners, introduced a group-based learning strategy. This learning is performed through interactive crossover (Fig. 25.5). Naturally, the learner forms the group randomly and interact with each

Algorithm 1 Upgraded teaching phase

Step 1. start
Step 2. randomly generate M learner
Step 3. evaluate all fitness of the learner
Step 4. Teacher=best learner (based on fitness)
Step 5. for(i=1;i<M;i++)
Step 6. diff(i)=(T-f(X_i))
Step 7. **for**(j=1;j<N;j++)
Step 8. new x_{ij}=1+(rand(1,diff(i))+x_{ij})%N
Step 9. update x_{ij} maintaining TSP condition satisfied
Step 10. **end for**
Step 11. **if** f(new X_i) better than f(X_i)
Step 12. then x_{ij}=new x_{ij}
Step 13. **else** x_{ij}=x_{ij}
Step 14. **end if**
Step 15. end for
Step 16. end



(a) For learner 1 (b) For learner 2

(c) For learner 3

Fig. 25.5 Interactive group-based crossover among three learner

other to enrich themselves. Assume for this investigation, formation of a group with an equal number of learners. So they all try to upgrade themselves through competition/interaction (Algorithm 2).

Here, consider a group with equal number (K = 3, say) of learners (Fig. 25.1). To upgrade individual learners from the group, with an example for five node TSP (Fig. 25.5) is as follows.

Consider three distinct learners from an arbitrary group 1-4-3-2-5, 1-2-3-4-5, and 1-5-4-3-2, say. Now for learner 1, check the minimum distance among 1 to 4, 2, 5, say

1-2 is minimum. Then learner 1 updates with 1-2. Again check minimum distance among 2-4, 2-3, 2-5, say 2-5 is minimum. Then the learner 1 update with 1-2-5. By repeating this process, learner 1 is completed until cover all the nodes. As all nodes are covered, finally updated learner 1 is 1-2-5-3-4 (say).
For learner 2, a similar procedure as for learner 1 is followed. In this case, modification is done to avoid local optima. Learners 1 and 3 are the same as above, but learner 2 consider as 1-5-4-3-2 instead of 1-2-3-4-5. It also helps to maintain diversity in the population.
For learner 3, a similar process as learner 1 is followed, modification in learner 3 is 1-2-3-4-5 instead of 1-5-4-3-2.

Algorithm 2 Interactive group-based learning

Step 1: start
Step 2: population size= number of learners (M)
Step 3: total number of distinct groups (TG)=M/K //K=number of learner in each group
Step 4: **for**(i=1; i< TG; i++)
Step 5: choose K distinct learner from the group
Step 6: **for**(t=1; t< K; t++)
Step 7: **for**(j=1; j< N; j++) //N=total number of nodes
Step 8: first node of the each learners are same (say s_i)
Step 9: choose next node (say l_{new})= min(s_i, each next unvisited node)
Step 10: update learner with l_{new} node
Step 11: s_i=l_{new}
Step 12: **end for**
Step 13: **end for**
Step 14: **end for**
Step 15: end

Combination of the above steps leads to the proposed mTLBO presented in Algorithm 3.

Algorithm 3 Modified Teaching-learning-based optimization (mTLBO)

Step 1. start
Step 2. randomly generate initial learner/population (M)
Step 3. g=1
Step 4. **while**(g <= G)
Step 5. selecting M fittest solutions from using Boltzmann selection
Step 6. teaching phase using Algorithm 1
Step 7. learning phase using Algorithm 2
Step 8. g=g+1
Step 9. **end while**
Step 10. end

Table 25.1 Notation and description mTLBO, TLBO and GA

Notation	Description
mTLBO	Boltzmann selection, upgradation strategy, and interactive group-based crossover
TLBO [10]	Upgradation strategy using crossover and reverse mutation
GA	Roulette wheel selection, cyclic crossover, random mutation

25.3 Computational Experiment

We design the algorithm in C/C++ using the Codeblock compiler on a 5th Gen. Intel Core i3 CPU @ 3 GHz.

Parameter settings: We have tuned the parameters to achieve the best results from the range.
Common parameters: Max_Generation = 1000–2500, Number of learner/chromosome = 50–150, $p_s = 0.40$–0.70.
Parameters of mTLBO: K (number of learner in each group) = 3.
Parameters of GA: $p_c = 0.25$–0.50, $p_m = 0.10$–0.30.

25.3.1 Results Using mTLBO, TLBO, GA

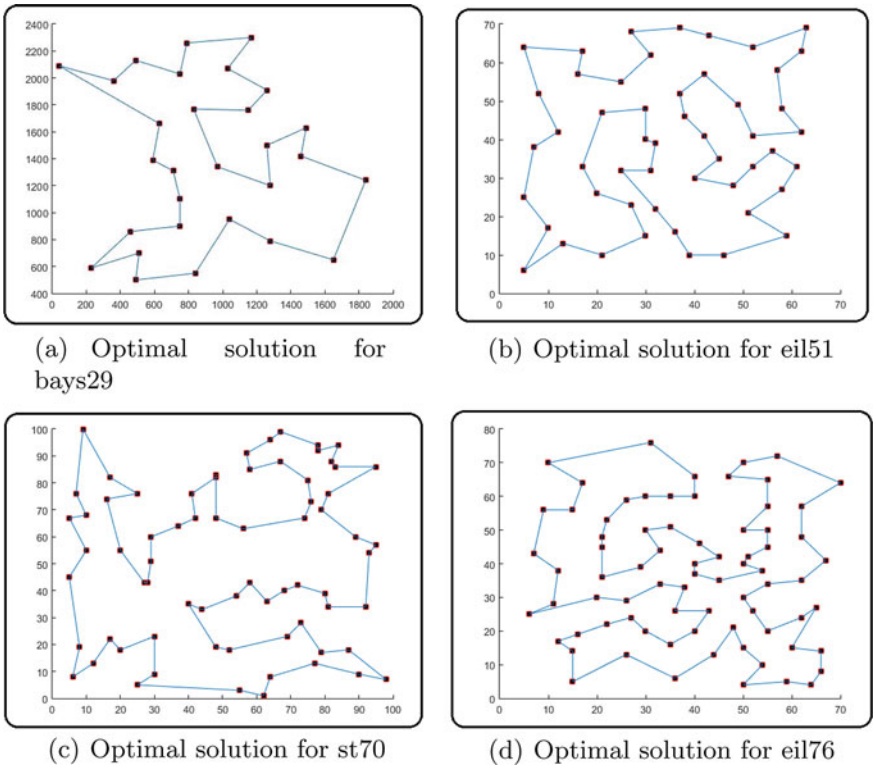
The performance of the intended mTLBO is established by solving 15 standard benchmark problems from TSPLIB [11] available in Table 25.2. Comparing all the problems regarding the total cost, iteration number, and CPU time (in second). All the results are obtained using 100 independent runs and taken using three different algorithms (c.f. Table 25.1)- mTLBO, TLBO [10] and GA. In the Table 25.2, "BKS" indicates the best-known solution, "BFS" indicates the best found solution. Some graphical results of benchmark instances are shown in Fig. 25.6. According to Table 25.2, mTLBO performs better compared to TLBO [10] and GA in terms of BFS, number of iteration and CPU time. The running time increased according to the size of the problems. By mTLBO, us16, gr17, gr21, bays29, dantzig42, eil51, st70, eil76, these problems are achieved the optimal solutions.

25.3.2 Supremacy of mTLBO Through ANOVA

In this investigation, assumptions of ANOVA (normal population distribution, distributions have the homogeneous variance, and data are independent) are satisfied. So to determine the effectiveness of the proposed mTLBO, we use one-way ANOVA test. The ANOVA test is applied to determine whether there are any statistically

Table 25.2 Results for standard TSP Problem (TSPLIB)

Instances	BKS	mTLBO			TLBO [10]			GA		
		BFS	Iteration	Time (s)	BFS	Iteration	Time (s)	BFS	Iteration	Time (s)
us16	6859	6859	58	0.06	6859	69	0.08	6859	75	0.08
gr17	2085	2085	71	0.07	2085	87	0.16	2085	83	0.17
gr21	2707	2707	163	0.10	2707	154	0.21	2710	199	0.19
bays29	2020	2020	137	0.20	2020	167	0.56	2024	156	0.58
dantzig42	699	699	247	0.37	702	281	0.89	714	297	0.93
eil51	426	426	304	0.67	430	301	1.42	449	377	1.88
berlin52	7542	7544	341	0.89	7556	414	1.62	7651	421	1.94
st70	675	675	456	1.39	675	493	2.53	699	521	2.78
eil76	538	538	508	1.96	541	556	2.93	578	582	2.99
rat99	1211	1214	763	2.54	1297	873	3.59	1258	893	3.67
kroA100	21282	21397	793	3.79	22367	1257	4.87	23654	1259	5.23
kroA150	26524	27456	1163	5.14	28457	1479	7.36	29841	1497	8.39
kroB200	29437	30659	1339	5.93	31274	1625	8.97	34187	1658	9.14
a280	2579	2687	1452	7.32	2698	1802	10.22	2941	1867	11.37
lin318	42029	44259	1952	8.53	45237	2001	11.49	49872	2086	13.67



(a) Optimal solution for bays29

(b) Optimal solution for eil51

(c) Optimal solution for st70

(d) Optimal solution for eil76

Fig. 25.6 Optimal graphical results for TSPLIB instances

Table 25.3 For different algorithms number of wins

Problem	us16	gr17	gr21	bays29	dantzig42	eil51	berlin52	st70	eil76	rat99	Mean
mTLBO	77	85	72	75	84	88	83	76	79	73	79.2
TLBO	69	78	68	67	71	72	73	74	68	72	71.2
GA	55	62	57	56	48	64	51	61	53	56	56.3

Table 25.4 ANOVA summary table

Source of variation	Sum of square	df	Mean of square	F
Between the groups	$SS_B = 2701.40$	$J - 1 = 2$	$MS_B = \frac{SS_B}{J-1} = 1350.70$	
Within the groups	$SS_W = 597.29$	$J(I - 1) = 27$	$MS_W = \frac{SS_W}{J(I-1)} = 22.12$	$\frac{MS_B}{MS_W} = 61.05$
Total	$SS_T = 3298.69$	$IJ - 1 = 29$		

significant differences between the means of two or more independent groups. Three algorithms (mTLBO, TLBO, and GA) with Maxgen = 2500 are used here. For the statistical test, 10 benchmark instances are considered, and the above three algorithms are tested against these instances. The number of wins (obtaining the optimal results) out of 100 independent runs of the above algorithms against the said test functions are recorded (cf. Table 25.3).

Here, the total number of the algorithm is (J) is 3, and the total number of instances (I) is 10. The means of mTLBO, TLBO and GA are 79.2, 71.2 and 56.3, respectively.

Here, calculated F (= 61.05 from Table 25.4) is higher than Critical F-values, $F_{0.05(2,27)} \approx 3.35$ (from the standard table). Thus between the groups, significant differences exist. Now, as F-ratio is significant with more than two groups, so multiple comparison test is performed to find different group means significant. Now, we perform Scheffe's multiple comparison F-test to determine whether the groups (mTLBO and TLBO) and/or (mTLBO and GA) are significant. Taking the first pair, we calculate F following $F = \frac{(\overline{X}_1 - \overline{X}_2)^2}{MS_W(\frac{1}{I} + \frac{1}{J})} = 15.43$. Similarly, $F = 95.21$ for the second group (mTLBO and GA). Here all the calculated values (61.05, 15.43 and 95.21) of F are higher than the tabulated value of F (3.35) and hence significant differences for both the groups exist. But, the mean of mTLBO is higher than the other means, TLBO and GA (cf. Table 25.4). Thus, it ultimately concluded that mTLBO is better than the other two algorithms-TLBO and GA.

25.4 Conclusion and Future Scope

In this paper, a modified TLBO (mTLBO) with (i) Boltzmann selection, (ii) upgradation strategy in the teaching phase, (iii) interactive group-based crossover for learners in the learning phase are proposed. Also, benchmark instances from TSPLIB are

solved and established the supremacy of mTLBO from other mentioned algorithms through ANOVA. The proposed algorithm can be used for solving real-life combinatorial optimization problems like Vehicle routing problem, Traveling Purchaser Problem, Facility location problems, etc. In this investigation, the mTLBO used for discrete problems can easily implemented for continuous optimization problems. In future, TLBO can be improved with different strategies in the teaching and learning phases.

References

1. Razali, N.M. and Geraghty, J., et al.: Genetic algorithm performance with different selection strategies in solving tsp. In: Proceedings of the World Congress on Engineering, vol. 2, pp. 1–6. International Association of Engineers Hong Kong (2011)
2. Chitty, D.M.: Applying aco to large scale TSP instances. In: UK Workshop on Computational Intelligence, pp. 104–118. Springer (2017)
3. Luo, H., Xu, W., Tan, Y.: A discrete fireworks algorithm for solving large-scale travel salesman problem. In: 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2018)
4. Zhang, J.-w., Si, W.-j.: Improved enhanced self-tentative PSO algorithm for TSP. In 2010 6th International Conference on Natural Computation, vol. 5, pp. 2638–2641. IEEE (2010)
5. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput.-Aided Des. **43**(3), 303–315 (2011)
6. Gorripotu, T.S., Samalla, H., Rao, J.M., Azar, A.T., Pelusi, D.: TLBO algorithm optimized fractional-order pid controller for agc of interconnected power system. In: Soft Computing in Data Analytics, pp. 847–855. Springer (2019)
7. Rao, R.: Review of applications of TLBO algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems. Dec. Sci. Lett. **5**(1), 1–30 (2016)
8. Li, J.Q., Pan, Q.K., Mao, K.: A discrete teaching-learning-based optimisation algorithm for realistic flowshop rescheduling problems. Eng. Appl. Artif. Intell. **37**, 279–292 (2015)
9. Saharan, S., Lather, J.S., Radhakrishnan, R.: Combinatorial problem optimization using TLBO. In: 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC), pp. 559–563. IEEE (2017)
10. Wu, L., Zoua, F., Chen, D.: Discrete teaching-learning-based optimization algorithm for traveling salesman problems. In MATEC Web of Conferences, vol. 128, p. 02022. EDP Sciences (2017)
11. Reinelt, G.: TSPLIB http://www.iwr.uni-heidelberg.de/groups/comopt/software. TSPLIB95 (1995)
12. Maity, S., Roy, A., Maiti, M.: A modified genetic algorithm for solving uncertain constrained solid travelling salesman problems. Comput. Indus. Eng. **83**, 273–296 (2015)