Shuffled teaching learning-based algorithm for solving robot path planning problem

Geetanjali Singh*, Nirmala Sharma and Harish Sharma

RTU Kota,
Rajasthan, India
Email: info.geet2412@gmail.com
Email: nsharma@rtu.ac.in
Email: hsharma@rtu.ac.in
*Corresponding author

Abstract: To evade the big and destructive obstacles in the real world scenario, such as bomb blast, nuclear activities, and fire breakdowns, robots are necessary. Robot path planning (RPP) problem is one of the interesting NP-hard problems in the world of robotics. The RPP problem can be dealt with, using swarm intelligence (SI) based optimisation algorithms. Teaching learning based optimisation (TLBO) algorithm is a very efficient and reliable swarm intelligence based algorithm in the history of optimisation. This paper proposed a hybridised version of TLBO with shuffled frog leaping algorithm (SFLA) to improve the efficiency in terms of exploitation and to overcome the slow convergence rate. The proposed variant is named as shuffled teaching learning-based optimisation (STLBO) algorithm. For checking the efficiency and accuracy of the proposed STLBO, it is applied to 12 continuous benchmark functions and compared with different nature inspired algorithms (NIA). To check the robustness of the propounded STLBO, it is implemented to solve the problem of RPP. Through simulation results and statistical analyses, the effectiveness of the proposed STLBO is proved.

Keywords: teaching learning-based optimisation; TLBO; shuffled frog leaping algorithm; SFLA; robot path planning; RPP; swarm intelligence-based algorithm; optimisation.

Reference to this paper should be made as follows: Singh, G., Sharma, N. and Sharma, H. (2020) 'Shuffled teaching learning-based algorithm for solving robot path planning problem', *Int. J. Metaheuristics*, Vol. 7, No. 3, pp.265–283.

Biographical notes: Geetanjali Singh is an Assistant Professor at RBS Engineering & Technical Campus, Agra, UP, India. He holds B-Tech and M-Tech. His research interests include soft computing, swarm intelligence based algorithms, artificial intelligence and optimisation techniques.

Nirmala Sharma is an Assistant Professor at RTU, Kota, Rajasthan, India. He holds BE and M-Tech. His research interests include data structure & algorithms, distributed algorithms, soft computing, operating system and real time system. He has published five journal articles and around four conference papers.

Harish Sharma is an Associate Professor at RTU, Kota, Rajasthan, India. He holds M-Tech and PhD. His research interests include swarm intelligence based algorithms, evolutionary algorithms, security, histopathological image analysis, pattern recognition, and medical imaging. He has published 20 journal articles and around 20 conference papers.

1 Introduction

The robot path planning (RPP) problem has been actively researched from 1970s till date, which aims to find out the optimised path from a given start location to the end location in a search space (Liu et al., 2005). Different sort of algorithms, like potential field method (Barraquand et al., 1992), roadmap cell decomposition and mathematical programs (Masehian and Sedighizadeh, 2007), including optimisation algorithms etc have been used to solve the RPP problem. Swarm intelligence (SI)-based algorithms are also used to solve the optimal path finding problems. The emergence of SI, in the field of artificial intelligence (AI), has proved to be a boon. Complex problems get easily solved with the advancement of SI-based algorithms which gives optimal results using previous intelligence, social learning with randomness in the potential solutions. Many heuristic algorithms like genetic algorithms (GAs), particle swarm optimisation (PSO), ant colony optimisation (ACO), simulated annealing (SA), and Tabu search (TS) (Nesmachnow, 2014) have already been applied to solve the RPP problem efficiently. Some other recent variants have also used their algorithms to solve the RPP problem (Gao et al., 2016; Sharma et al., 2019; Barraquand et al., 1992). In this paper, a hybridised variant of the two SI-based algorithms, namely TLBO and SFLA is proposed to solve the RPP problem accurately and efficiently. The population of the propounded algorithm always work in a mutual group and exchange information with each other to update the worst solution with the best one. The propounded hybridised variant is named as shuffled Teaching learning-based optimisation (STLBO) algorithm.

TLBO (Rao et al., 2011) is a SI-based algorithm which depicts the teaching behaviour of a teacher in a class and the learner's behaviour of grasping that knowledge. It comprises of two phases, i.e., teacher phase and learner phase. Teacher phase initially sort the population according to the fitness value, which is used to calculate the performance of the algorithm and finds the fittest learner and call it as a teacher which update rest of the learners, whereas the learner phase compares the knowledge of learners and improve the knowledge of the weaker student among them. SFLA (Eusuff et al., 2006) is a recent meta-heuristic which is based on the 'memetic' evolution concept. The population comprises of frogs, partitioned in memeplexes. Each memeplex is considered as an individual search area, where local search is performed and frogs communicate with each other to identify the optima within the search region. After a specified number of iterations, this information is communicated to other memeplexes by the process of shuffling and thereafter global search is performed. This process repeats until the stopping criteria are met. The memeplex partitioning phenomenon helps to ameliorate the exploitation ability of the SFLA algorithm within the memeplex while improves the exploration through sub-partitioning the swarm in the search area. Therefore, in this strategy, the concept of memeplexes, i.e., the partitioning the swarm of frogs is incorporated in the proposed variant of TLBO, namely STLBO to remove the drawbacks of slow convergence rate of TLBO. Further, the proposed TLBO is applied to solve the optimal RPP problem.

The remaining paper is organised as follows: in Section 2, the RPP problem is reviewed in detail. In Section 3, the problem formulation of RPP is presented. The basic TLBO algorithm is discussed in Section 4. In Section 5 the variant of TLBO, namely STLBO is proposed. Simulation results and statistical analyses are presented in Section 6. Section 7 presents the implementation and simulation results of the RPP problem. Finally, Section 8 includes the conclusion made to the work.

2 Literature review

With the advancement of technology, robots came into a light to handle the complex disastrous situations automatically, a human is unable to. They are programmed, to see the obstacles in their path and hence avoids the collision with them. RPP problem is a very complex problem of path selection and considered as a non-deterministic polynomial (NP) problem. Many types of research have been done on this problem and still, researchers are working to optimise the RPP problem. Bhattacharjee et al. (2011) applied artificial bee colony algorithm to solve the multi-RPP problem. She introduced a local trajectory planning mechanism to obtain the next positions of the robots from their current positions, to reduce the path and the spaces between the obstacles. Qin et al. (2004) introduced PSO with a mutation operator to resolve the path planning problem for mobile robots. Moreover, in that paper, the minimum path length from the initial position to the final position is also measured through the Dijkstra and the proposed modified PSO algorithms. For uncertain environments, a multi-objective PSO is proposed by Zhang et al. (2013) for the RPP problem. The degree of risk of a path is calculated by using a membership function, and another metrics, i.e., the distance of the path is also calculated. Further, through several test problems, the high quality optimal paths are demonstrated. Chen and Li (2006) proposed another strategy to evolve a smooth path for calculating the distance of the path covered by the robots. In that research, they used a stochastic PSO algorithm to simulate the path planning problem. Subsequently, Wang and Chirikjian (2000) proposed a new artificial Potential field method for path planning of non-spherical robots. This method is inspired by the steady heat transfer mechanism with small thermal conductivity. Another SI-based algorithm, namely ACO is applied by Brand et al. (2010) to solve the RPP problem in a dynamic environment. The proposed work includes two different pheromone re-initialisation schemes. In Tuncer and Yildirim (2012) proposed an improved GA to solve the dynamic path planning problem of a robot by finding a feasible route from a starting position to the target position. An improved mutation operator strategy is incorporated to avoid the premature convergence in the path.

3 RPP problem formulation

To formalise the RPP problem, a set of principles are generated based on some assumptions. The objective is to find an optimised path free from a collision, from the initial state to the target state as listed below.

1 The problem consists of a robot and a two-dimensional space, which includes the dangerous obstacles in the path.

2 The initial and final positions are determined.

3 There are several obstacles in the space whose radius and coordinates are defined by r, x-axis, and y-axis.

4 A robot aligns itself towards the goal by following a path.

5 The path contains several handle points or segments defined by n, at which the robot can change its direction to left or right.

6 The points make a complete path from source s to target t represented as (s, n_i, t), where $(i \in 1, 2, ..., n)$

7 If in case the movement of robot results in clashing with the objects, it has the ability to turn left or right by an angle of rotation.

8 If the robot reaches the target without collision, the final path is generated to the goal position. Let (x, y) be the current location of the robot at time t then at time $t + 1$, the next location (x', y') is calculated as follows:

$$x' = x + v \times \cos\theta\delta t$$
$$y^{'} = y + v \times \sin\theta\delta t$$

where v is the robots' velocity, θ is angle of rotation and δt represents a change in time instance.

9 The distance d travelled by the robot with velocity v is defined by equation $d = \sqrt{(x' - x)^2 + (y' - y)^2}$.

10 The objective of the RPP problem is to minimise the total distance covered, i.e., $\sum d$.

4 Teaching learning-based optimisation

Teaching learning-based optimisation (TLBO) algorithm depicts the behaviour of a class, consisting of teachers and learners. The teacher provides his/her knowledge regarding various subjects and the learners try to grasp the knowledge taught by the teacher. The TLBO algorithm comprises of two phases: teacher phase and the learner phase. In this process, the whole population is accounted as a collection of learners and different subjects are taught to learners, which are equivalent to the design variables of the population and the output of the learner phase is equal to the fitness value. Firstly, the best learner is picked and is considered a teacher, refers to the best solution having the highest fitness value. The detailed working of the two phases (Rao et al., 2011), i.e., teacher phase and learner phase is described below.

4.1 Teacher phase

The teacher phase simulates the behaviour of learners who collect knowledge from the teacher on various subjects and the teacher is responsible for bringing all the learners close to itself according to its capabilities, knowledge, and experience. The teacher tries to promulgate his knowledge to improve the overall mean of the class. There are N number of learners (population size, $k \in 1, ..., N$) and M number of subjects (subjects, $j \in 1, ..., M$), which are the design variables. The learners acquire knowledge by calculating the difference of the mean of the teacher and the current learner. Consider that the i^{th} learner is X_i, the best learner or teacher of the class and rest of the learners bring their mean close to the teacher by using the given equation:

$$Difference - Mean_{ji} = rand_i \times (M_T - TF \times M_{ji}) \tag{1}$$

where TF – teaching factor whose value is either 1 or 2 randomly, $rand_i$ is a random number between [0, 1], M_T is the position of the teacher and M_{ji} is the position of learners at i^{th} iteration and for j^{th} subject. The value of TF is defined by an equal probability

$$TF = round[1 + rand(0,1)] \tag{2}$$

The position update equation of the old solution in teacher phase is expressed by the equation:

$$X_{new} = X_{old} + Difference - Mean_{ji} \tag{3}$$

where X_{old} is the old position of the learner and X_{new} is the updated value of X_{old}. X_{new} value is accepted only when its result is better then the previous value otherwise it is neglected. Accepted values in the teacher phase are further provided as input to the next phase, i.e., learner phase.

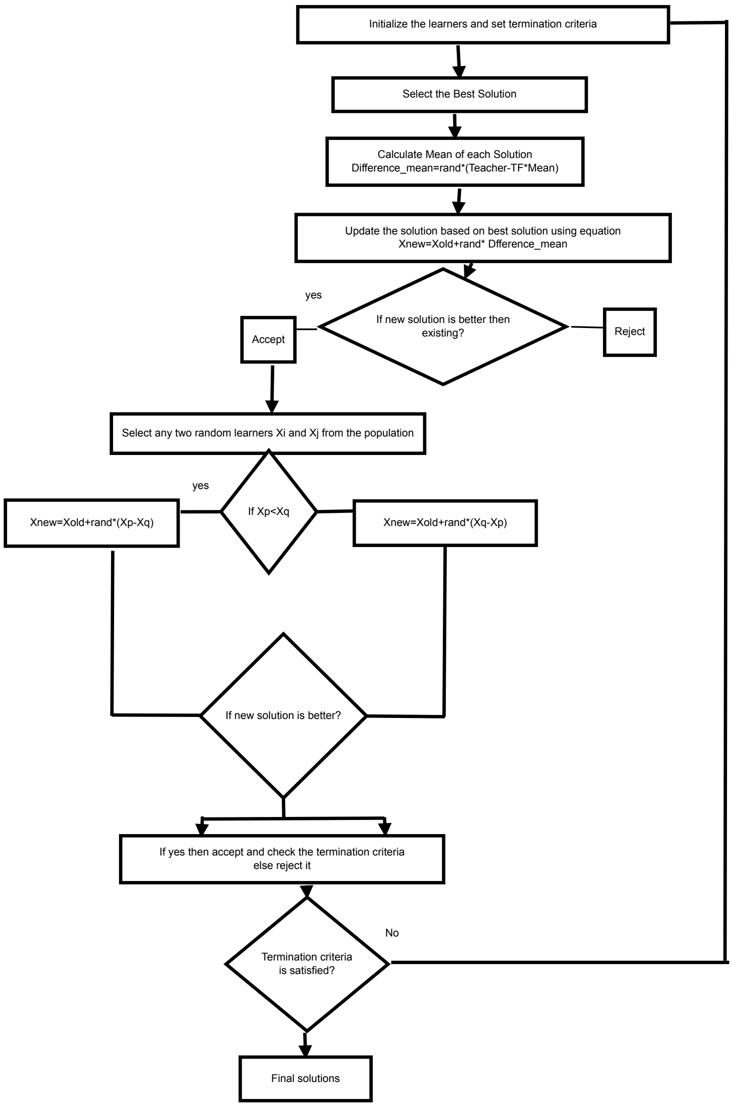
4.2 Learner phase

This is the next phase of the optimisation algorithm in which learners interact with each other and propagate their ideas and knowledge to each other. This phase selects any two learners randomly, provides a comparison mechanism and output the best value out of two. The learning scenario of this phase is explained as:

Select any two learners X_p and X_q randomly from the population N, such that $p \neq q$. The updated parameter X_{new} is described by equation (4). Equations (1), (2), (3) and (4) are taken from Rao et al. (2012).

$$\begin{aligned}
&\text{if } f(X_p) < f(X_q) \\
&X_{new} = X_i + rand_i \times (X_p - X_q) \\
&\text{else} \\
&X_{new} = X_i + rand_i \times (X_q - X_p)
\end{aligned} \tag{4}$$

Figure 1 TLBO algorithm



The TLBO algorithm is depicted through a flowchart given in Figure 1, starting with the initialisation of population the flow proceeds towards the teacher phase and the output of this phase is further provided as input to the second phase which results the best value of the objective function.

5 Shuffled teaching learning-based optimisation

TLBO lack in converging the solutions to global optima fastly (Rao and Patel, 2012). So to boost the effectiveness of the algorithm and to remove the drawbacks of stucking the solutions into the local optimum, a hybrid version of TLBO and SFLA algorithm is proposed. SFLA is one of the efficient memetic algorithm, where the word 'memetic' refers to the contagious information spread by replicating itself and thus affects the living organisms which result in different behaviour of the infected population (Eusuff et al., 2006). Inspired by this behaviour, the proposed algorithm is named as STLBO in which the swarm of learners are divided into different groups called sections, each of which is allowed to participates independently. After a predefined number of iterations, sections are forced to mix and form a new section through the shuffling concept.

To ensure competitive behaviour, learners are sorted according to the fitness value and then they are distributed among different sections on the basis of their fitness respectively for example, if there are two sections A and B then the fittest learner will go to section A, second fit learner goes to section B, third fit learner again goes to section A and so on. This division is repeated for all sections. This ensures the information exchange in an efficient way. Within each section, learners are influenced by the other learners of the section and update the positions accordingly. Further, in each section, a best learner is picked and replaces the worst learner of that section. In the position update process, the learners are selected using a triangular probability distribution to ensure a competitive behaviour as good learners are selected over the bad ones (Eusuff et al., 2006). Further, inspired from the exponent decreasing inertia weight (Bansal et al., 2011), a new TF is proposed in the STLBO which helps in improving the exploitation capability of the proposed algorithm.

The detailed working of the STLBO algorithm is explained as follows:

Step 1 *Initialisation of population:* The population of learners is defined by $P = m \times n$. Here m is the number of sections of a class and n is the number of learners in a particular section.

Step 2 *Teacher phase:* Initially, the best learner is chosen and nominated as a teacher in the same manner as it is chosen in the basic TLBO algorithm. The position of all the learners is updated using the equation (3). The modified TF defined in the equation (3) is calculated as shown in equation (5)

$$TF = (w_{\max} - w_{\min} - d) \times exp(1/(1 + d \times It/MaxIt)) \times rand. \tag{5}$$

Here $w_{\min}$ = 0.4, $w_{\max}$ = 0.9 and d = 0.4 which is taken from Bansal et al. (2011). $MaxIt$ is representing the maximum number iteration and It represents the current iteration.

Step 3 *Fitness calculation and rank the learners:* In this step, objective value is calculated for each learner. Then the learners are sorted in order of their decreasing objective value in the array X_i, such that $i = 1$ represent the best learner. Rank is assigned to each learner and its position is recorded.

Step 4 *Partition learners into sections (memeplex):* Partition the sorted learners (stored in array X) into sections such that learner with rank 1 go to Section 1, learner with rank 2 go to Section 2, learner with rank m go to section m, and $m + 1$ rank learner again goes to section 1, and this process continues repeatedly.

Step 5 *Construction of submemeplex:* Calculate weights by using the following probability distribution formula given in equation 6

$$Prob = 2 \times (P + 1 - R)/(P \times (P + 1)) \tag{6}$$

where P is the population size (total number of learners) and R is representing the rank of the learner. By using this probability distribution method the local best solution and worst solution of a particular section is identified.

Step 6 *Local learning method:* Evolve each section for a predetermined number of iterations. The best and the worst learner of the section have been identified in step 4. The position of the worst learner is updated by grasping knowledge from the local best solution of the section. The position of the worst learner is locally updated as follows: If rand (0, 1) is greater than 0.5.

$$new_{posj} = LB_{posj} + rand(0,1) \times (LB_{posj} - W_{posj}) \tag{7}$$

where W_{posj} refers to j^{th} dimension of the worst learner of that section, LB_{posj} is the local best learner's position in j^{th} direction and new_{posj} is the updated value of the worst learner. rand (0, 1) is uniformly distributed number in the range (0, 1).

Step 7 *Global learning phase:* If the worst learner of a section is unable to update its position in a predefined limit then the worst learner is updated globally by using the following equation (8).

$$new_{posj} = GB_{posj} + rand(0,1) \times (GB_{posj} - W_{posj}) \tag{8}$$

where GB_{posj} is the position of the global best learner in the j^{th} direction. new_{posj} is the updated value of the worst learner. W_{posj} refers to j^{th} dimension of the worst learner.

Step 8 *Shuffle learners:* In this phase, all the learners of the population are shuffled and are again arranged in the decreasing order of their fitness value.

The flow chart of the proposed STLBO algorithm is shown in Figure 2. Based upon the above discussion, the pseudocode of STLBO algorithm is shown in Algorithm 1.

Algorithm 1 STLBO

1 Initialise the population $P = m \times n$.
2 Until <termination condition>.
3 Teacher phase.
4 Shuffled phase:
  a Objective value is computed for each learner.
  b Rank the learners as per their calculated objective value.
  c Partition the learners into sections.
  d Performance evolution of each section locally as shown equation (7).
  e Update the learners globally and shuffle sections as shown in equation (8).
5 Print the best solution as the global optimum solution.

6 Simulation results and statistical analyses

To demonstrate the accuracy and efficiency of STLBO algorithm, 12 different global optimisation benchmark functions (f_1 to f_{12}) are chosen from CEC2005 (Hansen, 2006), CEC2013 (Li et al., 2013) as shown in Table 1. These problems possess a different level of complexity.

Figure 2 STLBO

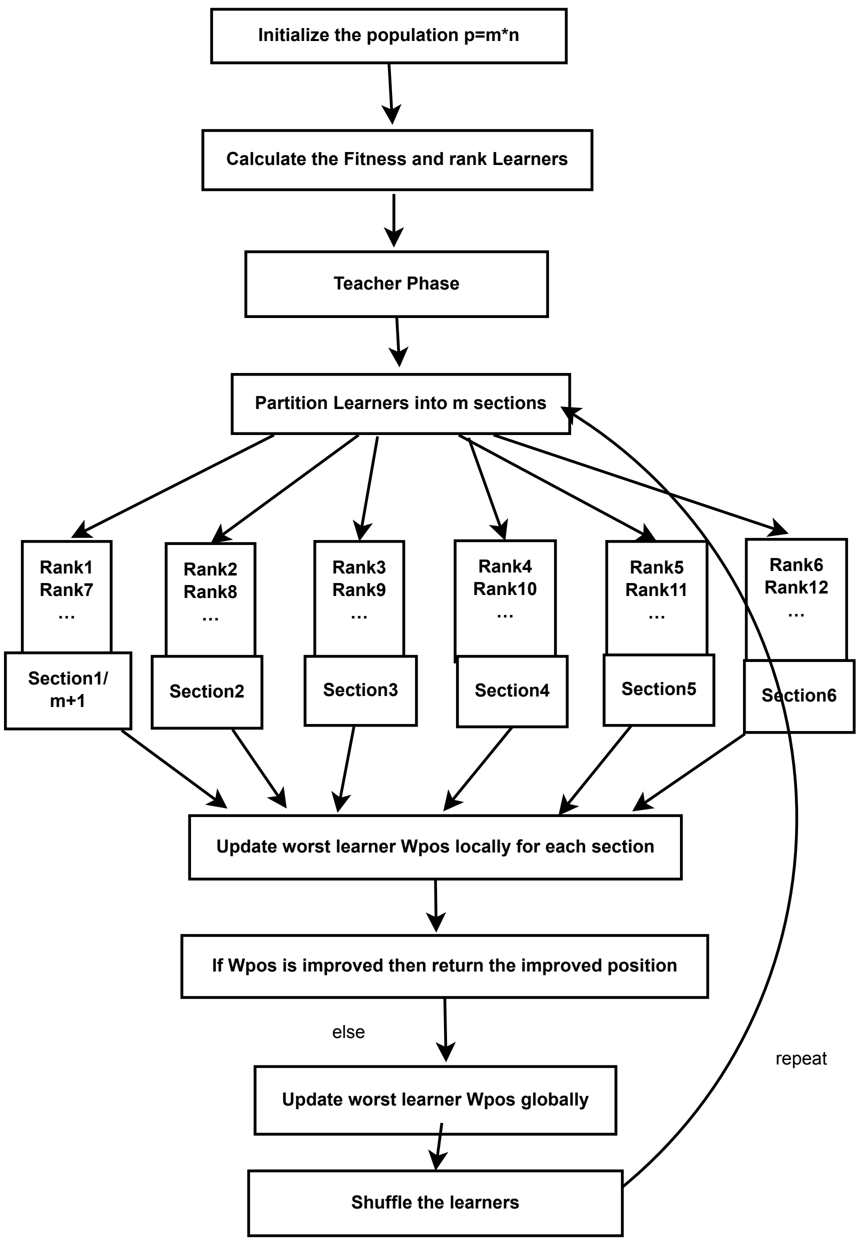


Table 1 Test problems

Test problem	*Objective function*	*Search range*	*D*	*AE*
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	[–5.12, 5.12]	30	1.0E-05
Rastrigin	$f_2(x) = 10D + \sum_{i=1}^{D}[x_i^2 - 10\cos(2\pi x_i)]$	[–5.12, 5.12]	30	1.0E-05
Griewank	$f_3(x) = 1 + \frac{1}{4000}\sum_{i=1}^{D} X_i^2 - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})$	[–600, 600]	30	1.0E-05
Ackley	$f_4(x) = -20 + e + exp(-\frac{0.2}{D}\sqrt{\sum_{i=1}^{D} {x_i}^3})$	[–30, 30]	30	1.0E-05
Zakharov	$f_5(x) = \sum_{i=1}^{D} {x_i}^2 + (\sum_{i=1}^{D}\frac{ix_i}{2})^2 + (\sum_{i=1}^{D}\frac{ix_1}{2})^4$	[–5.12, 5.12]	30	1.0E-02
Schewel	$f_6(x) = \sum_{i=1}^{D}\lvert x_i\rvert + \prod_{i=1}^{D}\lvert VE_i\rvert$	[–10, 10]	30	1.0E-05
Levy Montalvo 2	$f_7(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2$ $\times(1 + \sin^2(3\pi x_{i+1}))$ $+(x_D - 1)^2(1 + \sin^2(2\pi x_D))$	[–5, 5]	30	1.0E-05
Brannin's function	$f_8(x) = a(x_2 - bx_1^2 + cx_1 - d)^2$ $+e(1-f)\cos x_1 + e$	[–5, 0], [10, 15]	2	1.0E-04
Shifted sphere	$f_9(x) = \sum_{i=1}^{D} z_i^2 + f_{bias},\ z = x - o,$ $x = [x_1, x_2, \ldots .x_D],\ o = [o_1, o_2, \ldots o_D]$	[–100, 100]	100	1.0E-01
Shifted Griewank	$f_{10}(x) = \sum_{i=1}^{D}\frac{z_i^2}{4{,}000} - \prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}}) + 1$ $+f_{bias},\ z = (x - o),\ x = [x_1, x_2, \ldots .x_D],$ $o = [o_1, o_2, \ldots o_D]$	[–600, 600]	100	1.0E-01
Michalewicz	$f_{11}(x) = -\sum_{i=1}^{D}\sin x_i(\sin(\frac{i.{x_i}^2}{\pi})^{20})$	[0, π]	10	1.0E-05
Inverted cosine wave	$f_{12}(x) =$ $-\sum_{i=1}^{D-1}\left(\exp\left(\frac{-(x_i^2 + x_{i+1}^2 + 0.5x_ix_{i+1})}{8}\right)\times \mathrm{I}\right)$	[–5, 5]	10	1.0E-05

Notes: D: dimension, AE: acceptable error.

Table 2 Comparison of the results of selected test problems

Test problem	*Algorithm*	*SD*	*ME*	*AFE*	*SR*
f_1	STLBO	4.9638	8.8551E-06	113.6333	30
	TLBO	9.45801E-05	2.16571E-05	150	30
	SFLA	7.06726E-07	9.02173E-06	10,562.2333	30
	PSO	1.88424E-06	7.70209E-06	7,360	30
	GSA	9.41716E-07	8.76556E-06	95,093.3333	30
	CMA-ES	7.98343E-07	8.70431E-06	31,604.8	30
	BBO	3.47221E-11	3.034E-11	3,855	30
f_2	STLBO	0.0874	0.0550	191,035.7333	4
	TLBO	5.4218	9.8860	192,413.3333	2
	SFLA	9.1671	37.9078	200,000	0
	PSO	14.8652	52.8322	200,000	0
	GSA	3.5867	13.8630	200,000	0
	CMA-ES	9.0504	154.1173	200,000	0
	BBO	10.0599	37.8415	200,000	0
f_3	STLBO	0.1036	0.0261	115,500.0333	16
	TLBO	0.2453	0.4551	200,000	0
	SFLA	0.2942	0.5543	200,000	0
	PSO	0.9423	1.6927	200,000	0
	GSA	0.2345	0.4750	197,296.6667	1
	CMA-ES	45.5752	176.0700	200,000	0
	BBO	0.26741	0.5652	200,000	0

Table 2 Comparison of the results of selected test problems (continued)

Test problem	*Algorithm*	*SD*	*ME*	*AFE*	*SR*
f_4	STLBO	0.0599	0.0417	60,883.2333	28
	TLBO	0.5551	1.5148	197,570	1
	SFLA	0.5811	1.9450	200,000	0
	PSO	0.6859	2.3694	200,000	0
	GSA	0.2563	2.4825	200,000	0
	CMA-ES	0.0008	0.0034	200,000	0
	BBO	0.6421	2.6392	200,000	0
f_5	STLBO	28.6898	12.8317	82,943.7666	19
	TLBO	0.1597	0.4953	200,000	0
	SFLA	0.6437	0.5534	149,901.0333	8
	PSO	0.1851	0.1351	116,980	13
	GSA	8.82348E-07	8.80625E-06	90,630	30
	CMA-ES	1.2784E-06	9.33177E-06	33,607	1
	BBO	1.1431	1.7747	200,000	0
f_6	STLBO	0.5206	0.3031	185,748	30
	TLBO	0.1772	0.0589	78,500	22
	SFLA	8.89428E-07	8.78484E-06	19,064.4670	30
	PSO	0.0248	0.0066	20,616.6666	28
	GSA	6.12656E-07	8.99947E-06	95,498.3333	30
	CMA-ES	8.07488E-07	9.03694E-06	38,128.4	30
	BBO	2.03666E-06	8.66847E-06	17,946.6666	30
f_7	STLBO	1.36373E-06	9.26894E-06	6,667.61	30
	TLBO	3.05292E-05	3.91136E-05	18,266.6700	30
	SFLA	Nan	Nan	200,000	0
	PSO	3.18219E-05	3.43023E-05	1,146.6666	30
	GSA	3.29262E-05	4.92519E-05	37,113.3333	30
	CMA-ES	2.99E-05	3.72E-05	980.2667	30
	BBO	1.5479E-05	8.23829E-05	52,245	30
f_8	STLBO	1.30077E-06	9.19766E-06	71,641.5000	22
	TLBO	4.5159	9.6576	200,050	0
	SFLA	Nan	Nan	200,000	0
	PSO	7.0657	13.8962	200,000	0
	GSA	1.5644	5.1406	200,000	0
	CMA-ES	6.5937	5.9502	156,954.9000	7
	BBO	3.5664	7.8270	200,000	0
f_9	STLBO	9.31587E-06	1.3439E-05	1,235.9333	30
	TLBO	1.18373E-05	1.33911E-05	1,576.6666	30
	SFLA	Nan	Nan	200,000	0
	PSO	9.8253E-06	1.43773E-05	1,263.3333	30
	GSA	1.15984E-05	1.16961E-05	49,801.6666	30
	CMA-ES	9.83E-06	1.56E-05	1,441.8333	30
	BBO	0.3451	0.1904	68,246.6666	23

Table 2 Comparison of the results of selected test problems (continued)

Test problem	*Algorithm*	*SD*	*ME*	*AFE*	*SR*
f_{10}	STLBO	3.35196E-06	4.25819E-06	899.6000	30
	TLBO	3.17314E-06	4.40877E-06	1,133.3333	30
	SFLA	2.86E-06	3.44E-06	435.966	30
	PSO	8.12541E+39	1.04505E+40	3,000	30
	GSA	344.9912	1,780.4655	200,000	0
	CMA-ES	2.90E-06	3.49E-06	577.6667	30
	BBO	2.86279E-06	5.89745E-06	2,495	30
f_{11}	STLBO	1.36373E-06	9.26894E-06	71,641.5000	30
	TLBO	793.2988	305.4806	200,000	0
	SFLA	11,981.715	246,233.2979	200,000	0
	PSO	228.6428	102.7164	200,000	0
	GSA	25,714.5176	494,538.8858	200,000	0
	CMA-ES	5.27E-07	9.40E-06	135,618	22
	BBO	3.71463E-05	3.5000	200,000	0
f_{12}	STLBO	1.30077E-06	9.19766E-06	57,468.5333	30
	TLBO	3.6176	1.5767	200,000	0
	SFLA	88.023	2,151.6069	200,000	0
	PSO	0.5445	0.3885	200,000	0
	GSA	327.4422	4,333.4013	200,000	0
	CMA-ES	5.79E-07	9.36E-06	107,273	0
	BBO	2.0434	2.0304	200,000	0

To appraise the performance of the propounded STLBO algorithm, a comparison is made among STLBO, TLBO (Rao et al., 2012), SFLA (Eusuff et al., 2006), PSO (Kennedy and Eberhart, 1995), GSA (Rashedi et al., 2009), covariance matrix adaptation evolution strategy (CMAES) (Iruthayarajan and Baskar, 2010), and biogeography-based optimisation (BBO) (Simon, 2008). To test them over the given problems, some experiments are taken into consideration which is denoted as: The functions are evaluated on 30 runs with 5,000 iterations per run. The population size P = 50. The remaining parameter setting of the considered problems is kept same as mentioned in their research articles.

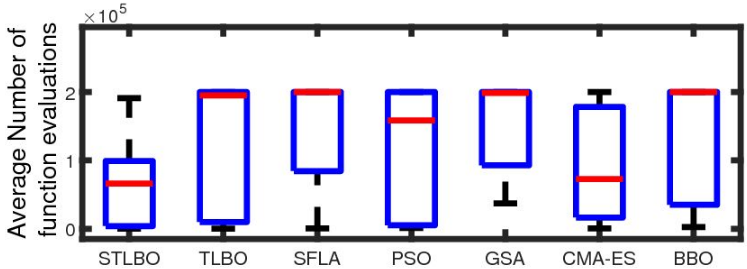
Table 2 provides a numerical report of the comparisons among the considered algorithms. The numerical results are presented in the form of standard deviation (SD), mean error (ME), average number of function evaluations (AFE) and success rate (SR). Here SR denotes the number of times the algorithm achieved the optima with the acceptable error in 30 runs.

While observing the results mentioned in Table 2, it is clear that the proposed algorithm, i.e., STLBO performs better than the TLBO, SFLA, PSO, GSA, CMA-ES, and BBO in terms of accuracy (i.e., ME), reliability (i.e., SR), and efficiency (i.e., AFE).

6.1 Statistical analysis

In addition, some more statistical tests are done like boxplots, acceleration rate (AR) (Sharma et al., 2018), and Mann-Whitney U rank sum test (Sharma et al., 2016). To compare the overall comparison in terms of AFEs, boxplots are generated for the average number of function evaluations. The boxplots for STLBO, TLBO, SFLA, PSO, GSA, CMA-ES, and BBO are presented in Figure 3. It is clear from this figure that the STLBO is very cost effective in terms of AFE's because median and interquartile range is very low for STLBO as compared to the considered algorithms.

Figure 3 Boxplots graph for average number of function evaluation (see online version for colours)



Besides, a comparison is made among the stated algorithms on the basis of the AR (Sharma et al., 2018) which is calculated as shown in equation (9).

$$AR = \frac{AFE_{ALGO}}{AFE_{STLBO}} \tag{9}$$

Here $ALGO \in \{TLBO, SFLA, PSO, GSA, CMA-ES, BBO\}$.

The value of $AR > 1$ shows the superior value of the proposed STLBO to the comparative algorithm. The results are depicted in Table 3, which concludes that the convergence speed of STLBO is proved to be better than the other stated algorithms.

Table 3 AR of $STLBO$ compare to the $TLBO$, $SFLA$, PSO, GSA, $CMA-ES$ and BBO

Test problems	*TLBO*	*SFLA*	*PSO*	*GSA*	*CMA-ES*	*BBO*
f_1	1.3200	92.9501	64.7697	836.8436	278.1296	33.9249
f_2	1.0072	1.0469	1.0469	1.0469	1.0469	1.0469
f_3	1.7316	1.7316	1.7316	1.7081	1.7316	1.7316
f_4	3.2450	3.2849	3.2849	3.2849	3.2849	3.2849
f_5	2.4112	1.8072	1.4103	1.0926	0.4051	2.4112
f_6	0.4226	0.1026	0.1109	0.5141	0.2052	0.0966
f_7	2.7396	29.9957	0.1719	5.5662	0.1470	7.8356
f_8	2.7923	2.7916	2.7916	2.7916	2.1908	2.7916
f_9	1.2756	161.8210	1.0221	40.2947	1.1665	55.2187
f_{10}	1.2598	0.4846	3.3348	222.3210	0.6421	2.7734
f_{11}	2.7916	2.7916	2.7916	2.7916	1.8930	2.7916
f_{12}	3.4801	3.4801	3.4801	3.4801	1.8666	3.4801

Mann-Whitney U rank sum (Sharma et al., 2018) test based on MFE's at α = 0.05 significance level is also performed and the obtained results are shown in Table 4. The obtained outcomes prove the validity of the proposed approach.

Table 4 Comparison based on AFE and the Mann-Whitney U rank sum test at

F-No.	*STLBO vs. TLBO*	*STLBO vs. SFLA*	*STLBO vs. PSO*	*STLBO vs. GSA*	*STLBO vs. CMA-ES*	*STLBO vs. BBO*
f_1	+	+	+	+	+	+
f_2	+	+	+	+	+	+
f_3	+	+	+	+	+	+
f_4	+	+	+	+	+	+
f_5	+	+	+	+	-	+
f_6	-	-	-	-	-	-
f_7	+	+	-	+	-	+
f_8	+	+	+	+	+	+
f_9	+	+	+	+	+	+
f_{10}	+	-	+	+	-	+
f_{11}	+	+	+	+	+	+
f_{12}	+	+	+	+	+	+

Notes: '+' indicates STLBO is better, '-' indicates STLBO is worse and '=' indicates that there is no noticeable difference.

7 Solving RPP problem using STLBO

The implementation of the proposed STLBO to search for an optimal path is described in steps below:

Step 1 Model the 2D workspace of the robot's movement based on the starting and the finishing positions, number of obstacles, and number of handles.

Step 2 Set the parameters needed, size of the population, maximum iteration, number of runs.

Step 3 Implement the propounded STLBO algorithm to search the optimal path of the given space.

Step 4 Calculate the fitness values and detect the collision of the obstacles if any.

Step 5 While the maximum number of iterations has not reached.

- a Find the next fit position.
- b Update the position of the robot locally as described by equation (6).
- c Make the next position as the current and moves in the forward direction to the next position, until it reaches the target.
- d Update the solution's position globally.
- e Store the feasible results and calculate the optimise path length.
- f Increment the iteration counter $t = t + 1$.

Step 6 Output the optimal path and pilot the robot to reach the target position.

The simulation results of the STLBO, TLBO, and PSO are recorded under the following environment:

- Operating system: Windows 10
- Processor: Intel core-i5
- Language: MATLAB 12.1
- Maximum iterations: 5,000.

The experiments have been carried out for three cases:

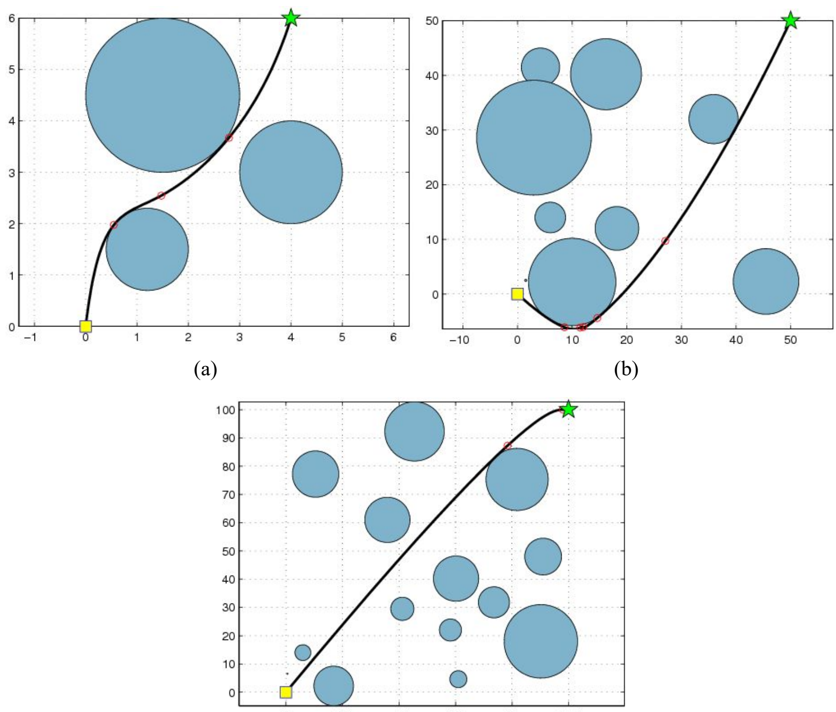
Case 1 3 obstacles, 3 handle points, start point (0, 0) and target point (4, 6).

Case 2 9 obstacles, 5 handle points, start point (0, 0) and target point (50, 50).

Case 3 15 obstacles, 8 handle points, start point (0, 0) and target point (100,100).

Figure 4 shows the simulation of STLBO for the three cases.

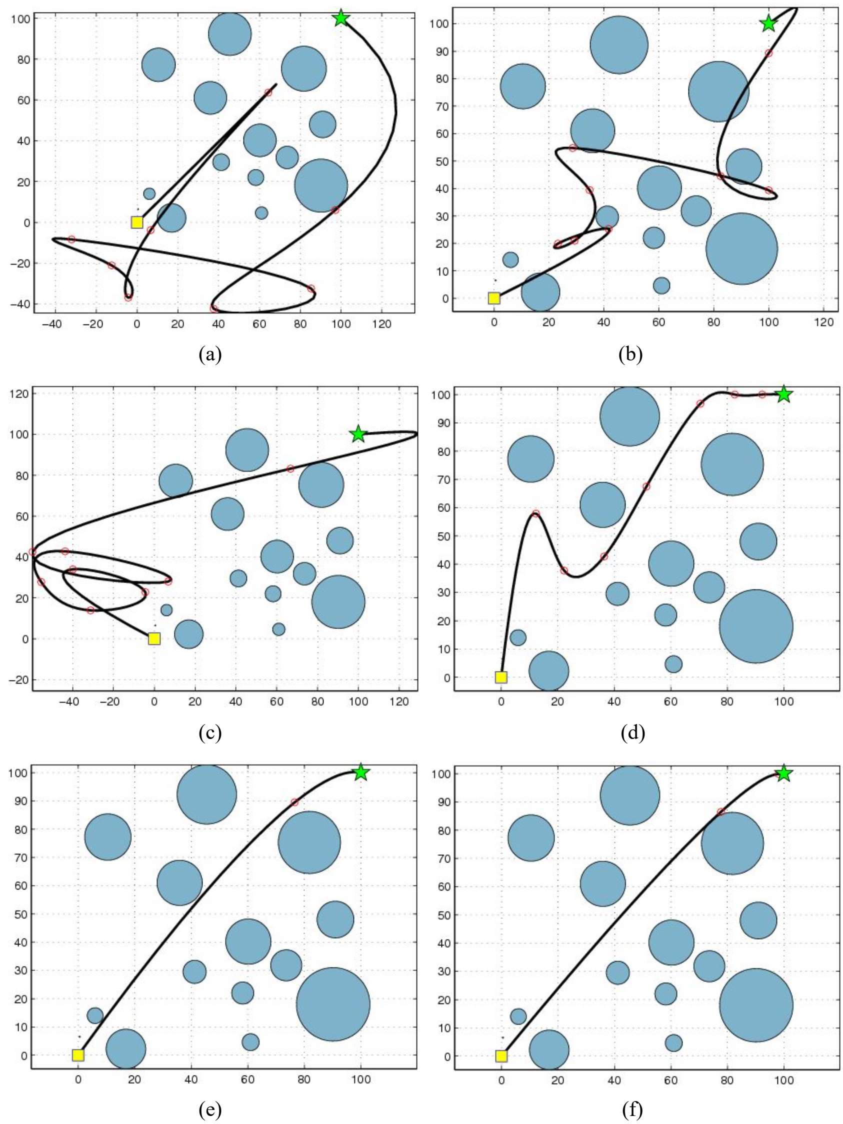
Figure 4 Different cases considered for RPP problem, (a) case 1 (b) case 2 (c) case 3 (see online version for colours)



(a) (b) (c)

Table 5 shows the simulation results of the propounded STLBO, TLBO, and PSO in terms of optimal distance for all the three cases.

Figure 5 Path traversed at different instances (see online version for colours)



It is clear from Table 5 that the optimal distance measured by the STLBO is less, compared to the TLBO and PSO algorithms.

Further, to demonstrate the complete working of the STLBO algorithm to find an optimal path, the algorithm is simulated for 15 obstacles and 8 handles with upper and lower dimensions are kept 100 and –100. The target position is set as (100, 100). The results are shown in Figure 5. These cases are recorded at different instances of time.

Table 5 Compared results of the optimal path

NO	*NH*	*Algorithms*	*OD*
3	3	PSO	7.6109
		TLBO	7.5984
		STLBO	7.5491
9	5	PSO	82.0904
		TLBO	96.8758
		STLBO	81.9512
15	8	PSO	144.7534
		TLBO	143.3134
		STLBO	142.2676

Notes: NO: number of obstacles, NH: number of handles and OD: optimal distance.

8 Conclusions

Simulation of natural phenomena for solving the different complex optimisation problems has been an inspirational and interesting field for various researchers. This paper presents a new variant of TLBO algorithm which is a hybridisation of the TLBO and shuffled frog leaping algorithm (SFLA). The proposed hybridised version is named as STLBO algorithm. The proposed STLBO algorithm is simple in structure and easy to implement with very few parameters required to adjust the values. To evaluate the performance, the presented STLBO algorithm is applied on the 12 benchmark functions and the obtained results are compared with the stated algorithms namely, TLBO, SFLA, PSO, GSA, CMA-ES, and BBO algorithm. Through statistical analyses, the competitiveness of the proposed STLBO is proved. Further, the propounded STLBO is applied to solve the RPP problem. The simulation results have been compared with the basic TLBO and PSO which proved that STLBO is an efficient meta-heuristic to resolve the RPP problem.

References

Bansal, J.C., Singh, P.K., Saraswat, M., Verma, A., Jadon, S.S. and Abraham, A. (2011) 'Inertia weight strategies in particle swarm optimization', in *2011 Third World Congress on Nature and Biologically Inspired Computing (NaBIC)*, IEEE, pp.633–640.

Barraquand, J., Langlois, B. and Latombe, J-C. (1992) 'Numerical potential field techniques for robot path planning', *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 22, No. 2, pp.224–241.

Bhattacharjee, P., Rakshit, P., Goswami, I., Konar, A. and Nagar, A.K. (2011) 'Multi-robot path-planning using artificial bee colony optimization algorithm', in *2011 Third World Congress on Nature and Biologically Inspired Computing (NaBIC)*, IEEE, pp.219–224.

Brand, M., Masuda, M., Wehner, N. and Yu, X-H. (2010) 'Ant colony optimization algorithm for robot path planning', in *2010 International Conference on Computer Design and Applications (ICCDA)*, IEEE, Vol. 3, pp.V3–436.

Chen, X. and Li, Y. (2006) 'Smooth path planning of a mobile robot using stochastic particle swarm optimization', in *Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation*, IEEE, pp.1722–1727.

Eusuff, M., Lansey, K. and Pasha, F. (2006) 'Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization', *Engineering Optimization*, Vol. 38, No. 2, pp.129–154.

Gao, X., Mu, Y. and Gao, Y. (2016) 'Optimal trajectory planning for robotic manipulators using improved teaching-learning-based optimization algorithm', *Industrial Robot: An International Journal*, Vol. 43, No. 3, pp.308–316.

Hansen, N. (2006) *Compilation of Results on the 2005 CEC Benchmark Function Set*, May.

Iruthayarajan, M.W. and Baskar, S. (2010) 'Covariance matrix adaptation evolution strategy based design of centralized PID controller', *Expert systems with Applications*, Vol. 37, No. 8, pp.5775–5781.

Kennedy, J. and Eberhart, R. (1995) 'Particle swarm optimization (PSO)', in *Proc. IEEE International Conference on Neural Networks*, Perth, Australia, pp.1942–1948.

Li, X., Tang, K., Omidvar, M.N., Yang, Z., Qin, K. and China, H. (2013) 'Benchmark functions for the CEC – 2013 special session and competition on large-scale global optimization', *Gene*, Vol. 7, No. 33, p.8.

Liu, G., Li, T., Peng, Y. and Hou, X. (2005) 'The ant algorithm for solving robot path planning problem', in *Null*, IEEE, pp.25–27.

Masehian, E. and Sedighizadeh, D. (2007) 'Classic and heuristic approaches in robot motion planning – a chronological review', *World Academy of Science, Engineering and Technology*, Vol. 29, No. 1, pp.101–106.

Nesmachnow, S. (2014) 'An overview of metaheuristics: accurate and efficient methods for optimisation', *International Journal of Metaheuristics*, Vol. 3, No. 4, pp.320–347.

Qin, Y-Q., Sun, D-B., Li, N. and Cen, Y-G. (2004) 'Path planning for mobile robot using the particle swarm optimization with mutation operator', in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, IEEE, Vol. 4, pp.2473–2478.

Rao, R. and Patel, V. (2012) 'An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems', *International Journal of Industrial Engineering Computations*, Vol. 3, No. 4, pp.535–560.

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.

Rao, R.V., Savsani, V.J. and Vakharia, D.P. (2012) 'Teaching – learning-based optimization: an optimization method for continuous non-linear large scale problems', *Information Sciences*, Vol. 183, No. 1, pp.1–15.

Rashedi, E., Nezamabadi-Pour, H. and Saryazdi, S. (2009) 'GSA: a gravitational search algorithm', *Information Sciences*, Vol. 179, No. 13, pp.2232–2248.

Sharma, A., Sharma, H., Bhargava, A., Sharma, N. and Bansal, J.C. (2016) 'Optimal power flow analysis using Lévy flight spider monkey optimisation algorithm', *International Journal of Artificial Intelligence and Soft Computing*, Vol. 5, No. 4, pp.320–352.

Sharma, N., Sharma, H. and Sharma, A. (2018) 'Beer froth artificial bee colony algorithm for job-shop scheduling problem', *Applied Soft Computing*, Vol. 68, pp.507–524, DOI: https://doi.org/10.1016/j.asoc.2018.04.001.

Sharma, P., Sharma, H., Kumar, S. and Sharma, K. (2019) 'Black-hole gbest differential evolution algorithm for solving robot path planning problem', in *Harmony Search and Nature Inspired Optimization Algorithms*, Vol. 741, pp.1009–1022, Springer, Singapore.

Simon, D. (2008) 'Biogeography-based optimization', *IEEE Transactions on Evolutionary Computation*, Vol. 12, No. 6, pp.702–713.

Tuncer, A. and Yildirim, M. (2012) 'Dynamic path planning of mobile robots with improved genetic algorithm', *Computers & Electrical Engineering*, Vol. 38, No. 6, pp.1564–1572.

Wang, Y. and Chirikjian, G.S. (2000) 'A new potential field method for robot path planning', in *IEEE International Conference on Robotics and Automation, Proceedings, ICRA'00*, IEEE, Vol. 2, pp.977–982.

Zhang, Y., Gong, D-w. and Zhang, J-h. (2013) 'Robot path planning in uncertain environment using multi-objective particle swarm optimization', *Neurocomputing*, 1 March, Vol. 103, pp.172–185, DOI: https://doi.org/10.1016/j.neucom.2012.09.019.