Engineering Applications of Artificial Intelligence 72 (2018) 393–414

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Tree Growth Algorithm (TGA): A novel approach for solving optimization problems

Armin Cheraghalipour [a], Mostafa Hajiaghaei-Keshteli [a,*], Mohammad Mahdi Paydar [b]

[a] Department of Industrial Engineering, University of Science and Technology of Mazandaran, University Street, P.O. Box: 4851878413, Behshahr, Iran
[b] Department of Industrial Engineering, Babol Noshirvani University of Technology, Shariati Street, P.O. Box 484, Babol, Iran

ARTICLE INFO

Keywords:
Trees growth algorithm
Combinatorial optimization
Metaheuristic algorithm
Approximate methods

ABSTRACT

Nowadays, most of real world problems are complex and hence they cannot be solved by exact methods. So generally, we have to utilize approximate methods such as metaheuristics. So far, a significant amount of metaheuristic algorithms are proposed which are different with together in algorithm motivation and steps. Similarly, this paper presents the Tree Growth Algorithm (TGA) as a novel method with different approach to address optimization tasks. The proposed algorithm is inspired by trees competition for acquiring light and foods. Diversification and intensification phases and their tradeoff are detailed in the paper. Besides, the proposed algorithm is verified by using both mathematical and engineering benchmarks commonly used in this research area. This new approach in metaheuristic is compared and studied with well-known optimization algorithms and the comparison of TGA with standard versions of these employed algorithms showed the superiority of TGA in these problems. Also, convergence analysis and significance tests via some nonparametric technique are employed to confirm efficiency and robustness of the TGA. According to the results of conducted tests, the TGA can be considered as a successful metaheuristic and suitable for optimization problems. Therefore, the main purpose of providing this algorithm is achieving to better results, especially in continuous problems, due to the natural behavior inspired by trees.

1. Introduction and literature review

In today's real world problems are often complex and huge. Hence, they cannot be solved by the exact methods in a proper time and cost (Cheraghalipour et al., 2017). There are different methods for solving an optimization problem. These methods are growing very fast in quantity and also in solution approaches. Some of these methods are inspired from natural processes called metaheuristics. Compared to conventional methods based on formal logics or mathematical programming, these meta-heuristic algorithms are generally more powerful (Gandomi et al., 2013a, b; Golshahi-Roudbaneh et al., 2017). There have been developed more than seventy metaheuristics in the last four decades. Most of them are evolutionary based algorithms which conduct the solution population in some iterations into achieve global optimum. Natural organisms, Social behaviors, feeding and mating behaviors especially in birds and fishes, are the main origin of metaheuristics ideas (Sadeghi-Moghaddam et al., 2017).

These types of algorithms are employed extensively in complex optimization problems. Actually, exploration and exploitation are the main phases of the metaheuristics (Yang, 2010b). The exploration phase ensures that the algorithm explores the search space carefully while the exploitation phase seeks out around the best solutions and chooses the best candidates or places. Most of new metaheuristics have been developed based on a new solution approaches to be faster is searching feasible areas, especially in large problems.

Generally, Optimization techniques can be divided in two groups, mathematical programming and metaheuristic algorithms and also, the existing metaheuristic algorithms may be divided into two main categories; Evolutionary algorithms and Swarm algorithms.

Evolutionary algorithms such as Evolutionary strategy (ES) (Fogel et al., 1966), genetic algorithm (GA) (Holland, 1975), Harmony search (HS) (Geem et al., 2001), and differential evolution (DE) (Storn and Price, 1997) are mainly inspired by biological evolution and mechanisms such as crossover and mutation.

The most well-known example in swarm intelligence is particle swarm optimization (PSO) (Eberhart and Kennedy, 1995). The algorithm is based on particle simulation collective behavior (or animals) and also is inspired by the social behavior of bird flocking or fish schooling. Some other well-known swarm algorithms are ant colony

* Corresponding author.
E-mail addresses: a.cheraghalipour.ie@mazust.ac.ir (A. Cheraghalipour), mostafahaji@mazust.ac.ir (M. Hajiaghaei-Keshteli), paydar@nit.ac.ir (M.M. Paydar).

https://doi.org/10.1016/j.engappai.2018.04.021
Received 6 January 2017; Received in revised form 28 February 2018; Accepted 16 April 2018
Available online 8 May 2018

optimization (ACO) (Dorigo et al., 1996), inspired by the collective foraging behavior of ants, bacterial foraging behavior optimization (BFO) algorithm (Liu and Passino, 2002), and bee colony optimization algorithm (BCO), which is based on the simulation of the food foraging behavior of honey bees (Teodorović, 2009). Also, some new versions of these algorithms such as PSO and ACO are developed (Atashnezhad et al., 2014; Yaralidarani and Shahverdi, 2016).

Recently, novel and strong algorithms have been developed in the literature. Yang (2010b) proposed one of strong optimization algorithm, called firefly algorithm (FA), which is inspired by the firefly's biochemical and social aspects. Yang and Deb (2010) developed a meta-heuristic algorithm via Levy Flights, called cuckoo search (CS). In addition, Hajiaghaei-Keshteli and Aminnayeri formulated a new swarm intelligence algorithm, namely Keshtel (KA). This algorithm provides new approaches in balancing between diversification and intensification phases. KA is inspired by feeding the greedy birds, namely Keshtel (Hajiaghaei-Keshteli and Aminnayeri, 2014, 2012). Besides, there exist some of recently developed metaheuristics in literature such as, Sine Cosine Algorithm (SCA), Interior Search Algorithm (ISA) and Sperm Whale Algorithm (SWA) (Ebrahimi and Khamehchi, 2016; Gandomi, 2014; Mirjalili, 2016a). Also, some multi-objective version of Simulated Annealing (SA) is developed (Alok et al., 2015; Bandyopadhyay et al., 2008). Furthermore, some comprehensive researches is existed that can be used for reader (Azamathulla, 2013, 2012; Zahiri et al., 2015, 2014).

To best our knowledge, about of vegetation only an algorithm under the title Planet Growth Simulation Algorithm (PGSA), published which inspired the growth of plants (Li and Wang, 2008). So, we find that TGA is a pioneer in the field of trees (Cheraghalipour and Hajiaghaei-keshteli, 2017). So far, a lot of metaheuristic algorithms have been developed which some of them mentioned in the literature review and a more comprehensive list of these algorithms is given in the Table 1.

Here, a new metaheuristic algorithm, called Trees Growth algorithm (TGA), is developed for global optimization. The TGA has two main phases; intensification and diversification. In intensification phase, we let the best trees, which is satisfied with light absorbing, compete on the food source. The new developed method in this phase ensures that each best tree moves toward a better food source, i.e. we move just toward the better food source or equivalently, toward the local optimum (or may be the global optimum). In the latter phase, we let some of the other trees compete for light absorbing and move to reach new or virginal places (solutions). Like other metaheuristics, the balance between intensification and diversification can be obtained by well tuning of the parameters. The performance and efficiency of the TGA was tested on some benchmark and engineering problem. Also, convergence analysis and significance tests via some nonparametric technique are employed to confirm efficiency and robustness of the TGA. The results confirm the applicability of TGA for solving optimization tasks.

The paper is organized as follows: Section 1 provides a brief review of the meta-heuristic algorithms. Section 2 presents the competition for light, logging of old trees, reproduction of trees and the characteristics of the proposed TGA, including the formulation of the algorithm. Numerical examples are presented in Section 3 to verify the efficiency of the TGA using some well-known standard benchmarks and engineering design problems which have been previously employed to validate different algorithms. Finally, some concluding remarks and suggestions for future research are provided in Section 4.

2. Common behaviors of trees

For centuries the trees have been considered as passive creations which have no free will. It is believed that their growth has been predetermined and the only obstruction to their growth would be temporary tension. Since the trees' movement cannot be noticed, it seems that they lack intelligent behavior. Nevertheless, the trees have been scattered on various lands and have occupied 99 percent of the Earth's accumulations. Therefore, there is inconsistency between the vastness of the areas that the plants have occupied and the common belief toward them. Because light has always been easily accessible, since the beginning of time, the sea plants with photosynthesis capability avoided movement and remained stagnant. But, this stagnation led to the increase in competition among life forms since the food was not equally dispersed in various parts of the sea. This led to an evolved generation of the plants which had the ability to move and explore.

2.1. Tree's competition in a forest

The trees in the jungle compete over light and nutrients (Vilà and Sardans, 1999), while the trees are still young, they mostly compete over light and as they grow older, they focus more on the nutrients (Coomes and Allen, 2007). In this investigation, it has been endeavored to study the greedy behavior of the trees in order to attract light which leads to them raising indirectly toward the Sun. Also the competition among the healthy trees which have received satisfactory amount of light will be discussed.

If we define intelligence as being in harmony with the environment, then the intelligence can be easily observed in the flexible behavior that the plants show in order to get enough light. The growing branches of the trees can sense the nearest competitors and through the usage of ultraviolet light predict the results of their activities, consequently, they have the chance to prevent these results from happening. These processes are mediated through "Phytochrome" molecules (Li et al., 2010). Phytochromes are light sensors and receivers of the plants. Each phytochrome is consisted of a receiving section and a transformation section of light. The receiving section has Tetrapyrrole structure. This section is connected to the transformation section (which is a type of protein) through "cysteine amino acid". This is how the phytochrome is shaped. Phytochromes shift from active to de-active with response to different wavelengths. The de-active phase is changed to active with response to red photons and attract ultraviolet photons better. Through this ability, the phytochromes can estimate the changes in wavelength among other plants (Smith, 2000).

The Sun light has equal amount of red and ultraviolet light. But this proportion reduces in hotbed, since the photosynthesis factors (e.g. chlorophyll) absorbs red light. One of the most important criterion for evaluating the neighboring plants is the degree of change in red and far red light proportion. In high populated areas, the far red light emitting from the leaf, is a clear message which implied that the competitors are near. After the reduced proportion of red to far red light is inferred, the sun favoring plant (Christie and Murphy, 2013) initiates the growth in length and if this strategy did not work it would stretch its leaves to the areas in which the changes in the light is less likely to happen. If the growth in length works out, the other responsive aspects of "shade avoidance" would accelerate the flowering process and premature production of the seeds, this phenomenon increases the chance of survival.

Phytochromes were first created in the prokaryote plants ancestors and it seems that they work as light sensors of the simplest from. Although the ability of the phytochromes in transforming the active form to de-active form in prokaryotes with response to sun quality was not of great importance, this ability was evolved and reformed through the evolution of dry plants and turned into a very important and complex sensor which is equal in worth with the visual ability in animals. In other words, the phytochromes can be considered as the eyes of the plants.

The tendency toward light is called phototropism (light tendency) (Briggs, 2014). Charles Darwin, the father of science of evolution, was the first who described this phenomenon scientifically. Through his famous voyage with Biggle (the name of the ship), he kept the birds he had collected on the way. He fed these birds with the buds from a special type of herbal plant named "Yulaf". Darwin nurtured these buds in small containers in a dark room. One day, this keen naturalist, observed that these plants have leaned toward the only window through which

Table 1
List of metaheuristics.

Reference	Algorithm name	Initial population
Robbins and Monro (1951)	Stochastic Approximation Method	Population based
Hooke and Jeeves (1961)	Pattern Search (PS)	Population based
Rastrigin (1963)	Random Search (RS)	Population based
Matyas (1965)	Random Optimization	Population based
Fogel et al. (1966)	Evolutionary strategy (ES)	Population based
Hastings (1970)	Metropolis–Hastings algorithm	Population based
Kernighan and Lin (1970)	Graph Partitioning Method	Population based
Holland (1975)	Genetic Algorithm (GA)	Population based
Glover (1977)	Scatter Search	Population based
Kirkpatrick et al. (1983)	Simulated Annealing (SA)	Solution based
Glover (1986)	Tabu Search (TS)	Solution based
Farmer et al. (1986)	Artificial Immune System (AIS)	Population based
Moscato (1989)	Memetic Algorithm	Population based
Koza (1992)	Genetic Programming (GP)	Population based
Dorigo et al. (1996)	Ant Colony Optimization (ACO)	Population based
Fonseca and Fleming (1993)	Multi-Objective GA (MOGA)	Population based
Battiti and Brunato (2010)	Reactive Search Optimization (RSO)	Population based
Srinivas and Deb (1994)	NSGA for Multi-Objective Optimization	Population based
Eberhart and Kennedy (1995)	Particle Swarm Optimization (PSO)	Population based
Hansen and Ostermeier (2001)	CMA-ES	Population based
Storn and Price (1997)	Differential Evolution (DE)	Population based
Rubinstein (1997)	Cross Entropy Method (CEM)	Population based
Hansen et al. (1997)	Variable Neighborhood search(VNS)	Population based
Taillard and Voss (1999)	Partial Optimization Metaheuristic Under Special Intensification Conditions (POPMUSIC)	Population based
Geem et al. (2001)	Harmony Search (HS)	Population based
Hanseth and Aanestad (2001)	Bootstrap Algorithm (BA)	Population based
Larrañaga and Lozano (2002)	Estimation of Distribution Algorithms (EDA)	Population based
Deb et al. (2002)	NSGA-II for Multi-Objective Optimization	Population based
Liu and Passino (2002)	Bacterial Foraging behavior Optimization (BFO)	Population based
Nakrani and Tovey (2004)	Bees Optimization	Population based
Krishnanand and Ghose (2005, 2006)	Glowworm Swarm Optimization(GSO)	Population based
Basturk and Karaboga (2006)	Artificial Bee Colony Algorithm (ABC)	Population based
Pham et al. (2005)	Bees Algorithms (BA)	Population based
Haddad et al. (2006)	Honey-bee Mating Optimization (HMO)	Population based
Shah-Hosseini (2009)	Intelligent Water Drops (IWD)	Population based
Atashpaz-Gargari and Lucas (2007)	Imperialist Competitive Algorithm (ICA)	Population based
Mucherino and Seref (2007)	Monkey Search (MS)	Population based
Wierstra et al. (2008)	Natural Evolution Strategies	Population based
Yang (2009)	Firefly Algorithm (FA)	Population based
Teodorović (2009)	Bee Colony Optimization (BCO)	Population based
He et al. (2009)	Group Search Optimizer (GSO)	Population based
Yang and Deb (2009)	Cuckoo Search (CS)	Population based
Rashedi et al. (2009)	Gravitational Search Algorithm (GSA)	Population based
Husseinzadeh Kashan (2014, 2009)	League Championship Algorithm (LCA)	Population based
Kadioglu and Sellmann (2009)	Dialectic Search	Population based
Yang (2010a)	Bat Algorithm (BA)	Population based
Lourenço et al. (2010)	Iterated Local Search(ILS)	Population based
Shah-Hosseini (2011)	Galaxy-based Search Algorithm (GbSA)	Population based
Tamura and Yasuda (2011b, a)	Spiral Optimization (SO)	Population based
Rajabioun (2011)	Cuckoo Optimization Algorithm (COA)	Population based
Rao et al. (2011)	Teaching-Learning-Based Optimization (TLBO)	Population based
Alsheddy (2011)	Guided Local Search (GLS)	Population based
Gandomi and Alavi (2012)	Krill Herd (KH) Algorithm	Population based
Hajiaghaei-Keshteli and Aminnayeri (2014, 2012)	Keshtel Algorithm (KA)	Population based
Civicioglu (2012)	Differential Search Algorithm (DSA)	Population based
Alatas (2012)	Artificial Chemical Reaction Optimization Algorithm (ACROA)	Population based
Husseinzadeh Kashan (2013)	Optics Inspired Optimization(OIO)	Population based
Husseinzadeh Kashan et al. (2015)	Grouping Evolution Strategies (GES)	Population based
Sadollah et al. (2013)	Mine Blast Algorithm (MBA)	Population based
Hatamlou (2013)	Black Hole (BH)	Population based
Civicioglu (2013a)	Artificial Cooperative Search Algorithm(ACS)	Population based
Kaveh and Mahdavi (2015)	Colliding Bodies Optimization (CBO)	Population based
Gandomi (2014)	Interior Search Algorithm (ISA)	Population based
Salimi (2014)	Stochastic Fractal Search (SFS)	Population based
Cheng and Prayogo (2014)	Symbiotic Organisms Search (SOS)	Population based
Zheng (2015)	Water Wave Optimization (WWO)	Population based
Dogan and Ölmez (2015)	Vortex Search Algorithm (VSA)	Population based
Wang et al. (2015)	Elephant Herding Optimization (EHO)	Population based
Baykasoğlu and Akpinar (2017)	Weighted Superposition Attraction (WSA)	Population based
Mirjalili (2016b)	Dragonfly algorithm	Population based
Liang et al. (2016)	Virus Optimization Algorithm	Population based
Mirjalili (2016a)	Sine Cosine Algorithm (SCA)	Population based
Ebrahimi and Khamehchi (2016)	Sperm Whale Algorithm (SWA)	Population based
Mirjalili et al. (2017)	Salp Swarm Algorithm	Population based

a gleam of light was coming into the room. He conducted experiments based on his observations which he later on published as an article under the title of "The Power of Movement in Plants" (Darwin and Darwin, 1880). Darwin had proven that the stem of Yulaf bud raised erectly in the absence of light, but the stems of the buds which were exposed to sunlight leaned toward the light. Nevertheless, he mentioned that if the tip of the stems was cut, the bud would not lean toward the light. Further experiments which were conducted based on Darwin's investigation led to discovering a new hormone in plants called "auxin" (Friml et al., 2002; Haga and Sakai, 2012).

Based on what the founders of auxin has mentioned, this molecule is made on the tip of the stem and causes the asymmetrical growth of the stem on both sides, which means the side which receives less light stores more auxin and grows faster, consequently the stem will lean toward the light. Before all this to happen, the tree must recognize from which side the Sun is shining. The plants, similar to how animals see, can sense the light and the direction from which it is shown, they use specific sensors to analyze different attributes that various types of light have and also evaluate these traits. In order to analyze the light, the tree has to compare two sides (the one that receives the light and the side that does not) in order to recognize the gradient of the light on their body. In order for the trees to analyze the light, they make use of information such as gradient of the light, the duration of the sunlight, the portion of various wavelength in the light, etc. the receiver which give the tree the ability to recognize the one-sided light is called "phototropin". Phototropins are the molecules which are connected to other phosphate molecule and change their activities. To be exact, Flavin which creates the receiving part of this molecule receives the photon message and transform it to chemical reactions. Such reactions lead to the change in genes and cause the tree to lean toward the light.

Then based on the results of the studies conducted so far, it can be concluded that the trees compete over light and if they are cooped up under bigger trees, they change the direction in which their branches grow and get to the light available in free spaces between other trees as shown in Fig. 1.

2.2. Trees' reproduction

Similar to many other types of plants, the trees reproduce in three ways, namely, pollination, impregnation, and scattering. Since the trees are stable, through years, many tools for reproduction and survival have improved within them.

2.2.1. Pollination

The trees are either gymnosperm or angiosperm. In gymnosperm trees, the male and female flowers are placed on the same tree. The seeds of these trees are scattered with wind. The male flower has small tubal structure similar to the leaf and flag which creates cloud pollen in early spring. The female flower is the tubal miniature copy of the adult flower which is formed eventually. Angiosperms have created varieties of different flowers some of which are scattered with wind and some others have aromatic calycle or nectar for bugs and other creatures. In some species, the trees are specifically male or female. The flowers of such trees have either compound or separate male and female parts (Thien et al., 2000).

2.2.2. Pollination by insects

Insects have essential role in plants' reproduction. Some insects such as bees land on the flowers and the pollens stick to their feet and finally moved to other flowers.

2.2.3. Scattering seed

The seed of the gymnosperms are usually scattered through wind. The tubes are opened up with response to heat and moisture and release the winged seeds in a specific period of time in order to make sure that some have the chance to survive in appropriate conditions (Ruxton and Schaefer, 2012). Some of pine trees have seeds which remain inside the shell until they are moved by other animals.

2.2.4. Self-reproducing

Some trees have the ability to be shredded, which means while the stem is inside the ground, the root remains in the soil. After destroying the progenitor tree, the reproduction cycle remains. The shallow root creates a new root in adjacency of the progenitor and a new tree will rise.

2.3. Cutting down old and weak trees

Due to some reasons such as road construction, wood smuggling, paddocking, etc. many trees are cut. But we focus our criterion on a utopia in which only the weak trees are cut. Wood-eating beetles and mushroom pests (Gough et al., 2014), along with micro-pollens, cause the spread of disease in the jungle and, therefore, the foresters have to cut down the sick trees in order to prevent the disease from spreading.

2.4. The proposed tree growth algorithm

As mentioned earlier, the proposed algorithm is inspired by the trees competition for acquiring light and foods. In this algorithm, the main phases are divided into four groups.

In one of these groups, called the best trees group, some better trees, due to favorable conditions for growth, will grow further and because of amount of received light be satisfied, their competition focuses for food. Since the growth of trees done slowly, it makes the good trees, basically tall and smooth and most importantly are older than the others, That due to the increasing age of the tree, its growth rate lower than before (young trees), and their main competition focuses on food in roots.

In the other group, called the competition for light group, some of the trees to reach the light, move to distance between the close best trees under different angles. In the other group, called the remove and replace group, some weak trees, which do not have little growth or for reasons stated in Section 2 are cut by foresters and replace it with new trees are planted. And finally in the last group, called the reproduction group, the best Trees, because growth has been favorable, they begin to multiply and create new plants. Since that arise near the mother tree inherit some of the factors that location. The detailed algorithm is described below:

1. Randomly generate initial population of trees on the upper and lower bounds and calculate their fitness values.
2. Find the best tree. If there is a minimum optimization problem, the best tree as the minimum objective function and vice versa. This element at jth iteration, T_{GB}^{j} is the global best of it.
3. Allow N_1 better solutions have local search using formula (1). (For any solution check several local search. If the new solutions value is better than initial response, replace it.)

$$T_i^{j+1} = \frac{T_i^j}{\theta} + rT_i^j. \tag{1}$$

That θ is trees reduction rate of power, due to aging, high growth and reduce food around. And r is $U(0,1)$, which due to trees satisfaction of the light, its roots are instructed to move to absorb food which is caused growth at a rate of rT_i^j units. For clarification of this process, it is given in Fig. 2.

4. Move N_2 solutions to distance between the close best solutions under different α angles. To do this, first, find the distance between the selected trees and other with formula (2).

$$d_i = \left(\sum_{i=1}^{N1+N2} \left(T^j{}_{N2} - T_i^j \right)^2 \right)^{\frac{1}{2}} \quad \& \quad d_i = \begin{cases} d_i & \text{if } T^j{}_{N2} \neq T_i^j \\ \infty & \text{if } T^j{}_{N2} = T_i^j. \end{cases} \tag{2}$$

Then choose two solutions x_1 and x_2 with minimal d_i and to get a linear combination between the trees, as shown in Fig. 3, the formula (3) used. ($\lambda = U(0,1)$)

$$y = \lambda x_1 + (1-\lambda)x_2. \tag{3}$$

Fig. 1. The competition over light.

(a) Initial chromosome:
$T_i^{\,j} = uniform \sim [-10,10]$

(b) Step one $\frac{T_i^{\,j}}{\theta}$ where $\theta = 0.8$

(c) Step two $r \times T_i^{\,j}$ where $r = 0.4$

New chromosome after using formula (1)

(a)	-4.5	2	3.75	-3.42	1.3
(b)	-5.62	2.5	4.69	-4.27	1.62
(c)	-1.8	0.8	1.5	-1.37	0.52
(b+c)	-7.43	3.3	6.19	-5.64	2.15

Fig. 2. An example to show how we can use the formula (1).

Finally, to move this tree between two adjacent trees with an $\alpha_i = U(0,1)$ angles, as Fig. 4, obtained with formula (4). To transparency of formula (2)–(4) their process is illustrated in Fig. 5.

$$T^j{}_{N_2} = T^j{}_{N_2} + \alpha_i y. \tag{4}$$

5. Remove N_3 worse solution, and instead of them randomly generated solutions.
6. Create new population (new population $N = N_1\ N_2 + N_3$).
7. N_4 new solution is generated and each new solution are changed by mask operator respect to best solution (of the population N_1) randomly, then added to the new population (new population = new population + N_4). A sample of this step is shown in Fig. 6.
8. After sorting the new population, the number of initial population N of this new population we consider as initial population for the next iteration (according to the roulette wheel or tournament or choose the best solutions).

If any of the stop criteria is not satisfied, repeat from step 2. The pseudo code of the proposed TGA is presented in Fig. 7 and TGA flow chart present in Fig. 8.

2.5. Motivation, advantages, and differences of TGA with similar algorithms

Voraciously behavior of trees in competition for light, motivates us to develop this algorithm. As previously mentioned, like all metaheuristic, TGA is composed of two intensification and diversification phases. In intensification phase, with changing in the chromosome, objective function is improved. In this phase, we let the best trees, which are satisfied with light absorbing and lengthened sufficiently, compete on food sources, and improve some of their chromosomes' gens which related to water, food, etc.

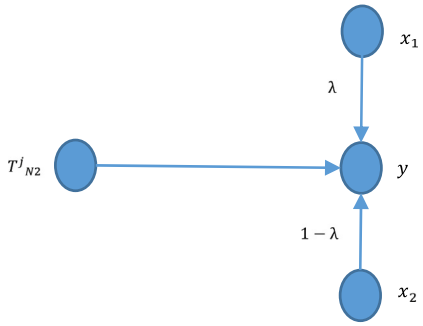

Fig. 3. Linear combination.

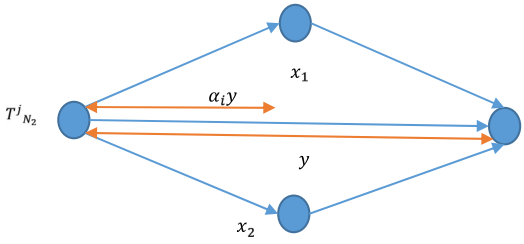

Fig. 4. Moving between two adjacent trees with angle.

In diversification phases, three subsections are proposed to reach new or virginal places (solutions).

As is clear in Fig. 8, the two phases are represented by a different color scheme. By well tuning the parameters, the balance between

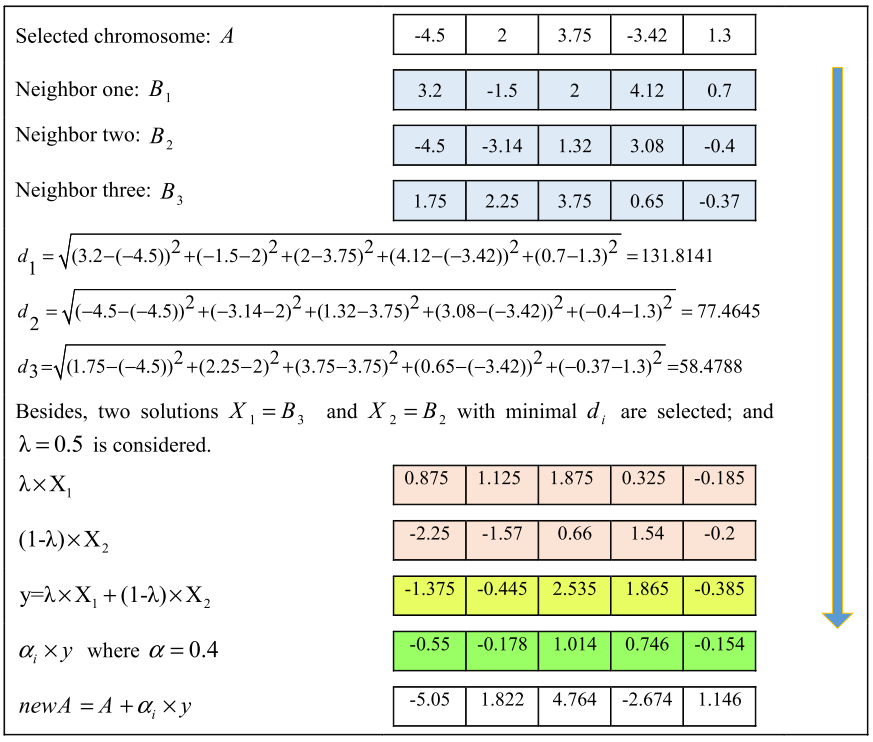

Fig. 5. An example to show how we can use the formula (2)–(4).

New generated solution:	0.35	-1.75	2.43	-0.325	-3.92
Mask operator:	0	1	1	0	1
A best solution:	0.47	1.23	-0.23	-2.7	4.37
New solution after changing by mask operator:	0.35	1.23	-0.23	-0.325	4.37

Fig. 6. An example to show how we can use mask operator.

intensification and diversification can be obtained. In order to determine advantages of TGA versus other metaheuristics, a comparison among TGA, GA, as a primary and robust algorithm, and PSO, as a swarm intelligence algorithm in particle simulation collective behavior is performed.

- TGA like GA and PSO is population based. Intensification and diversification phases of PSO are done with one action, movement among particles with considering global best and personal best. In GA, intensification phase is done via mutation, as a blind local search. But in TGA, competition on achieving food continues so much until a better answer is found, unless around foods are vanished. So the intensification phase of TGA is more greedily than them.
- In GA diversification phases is performed by the crossover as well as moving among the particles in PSO. While in TGA, diversification phase is done in three sections; Elimination of the worst solution and generate randomly instead of them, moving between neighbor trees to absorb the light, and reproduction of trees and producing seedlings that will be added to the current population.
- Last but not least, note that the searching approaches of these algorithms are different. PSO inspires by the collective behavior of particles, and GA simulates natural evolution which occurs on chromosomes. But TGA is inspired by the voraciously behavior of the trees.

3. Experimental results and discussion

To study the performance and behavior of the TGA in different situations, it is validated by considering three perspectives.

Case 1: It solved using 30 well-known bench-mark problems in both low and high dimension problems.
Case 2: Its performance in solving 3 engineering optimization.
Case 3: Its performance in solving 2 industrial engineering optimization.

Also, in this section TGA is tuned and the convergence and significance tests is performed. Due to the random nature of the TGA and other meta-heuristic algorithms, their performance cannot be judged by the result of a single run. More than one trial with independent population initializations should be made to evaluate the performance of the approach. Therefore, in this study the results are obtained in 30 trials. The population size of 100 was used for tuning algorithm and solving engineering problems while for solving case 1 the population size equal to 20 is considered.

3.1. Tuning

All metaheuristics have some factors which should be tuned before the simulation. So, to use the algorithm in its best condition, we tune its parameters. The first step is to identify the main factor of algorithms, which are given in the Table 2.

Initialization
While any stop criteria is not satisfied
Find the T_{GB}^{j}

For $i = 1{:}N1$
For $l = 1{:}L$
$T_i^j = \frac{T_i{}^{j-1}}{\theta} + rT_i{}^{j-1}$
if $f(T_i^j) \le f(T_i^{j-1})$
$T_i^j = T_i^j$
else $T_i^j = T_i^{j-1}$
end if
end for
end for

For $i = (N1+1){:}(N1+N2)$
$$d_i = \left(\sum_{i=1}^{N1+N2} \left(T^j{}_{N2} - T_i^j\right)^2\right)^{\frac{1}{2}}$$
$$d_i = \begin{cases} d_i & if\ \ T^j{}_{N2} \neq T_i^j \\ \infty & if\ \ T^j{}_{N2} = T_i^j \end{cases}$$
$x_1 = \mathrm{T}(d_1)\ \&\ x_2 = Td_2\ \ after\ sorting\ the\ d_i$
$y = \lambda x_1 + (1-\lambda)x_2$
$T^j{}_{N2} = T^j{}_{N2} + \alpha_i y$
end for

For $i = (N1+N2+1){:}N$
$T_i^j = creat\ random\ T_i^j$
end for

For $k = 1{:}N1$
$S_k^j = creat\ random\ S_k^j$
$f = rand\ sample\ from\ N1$
$S_k^j = do\ mask\ operation\ \ S_k^j\ with\ \ T_f^j$
end for

end while

Fig. 7. Pseudo cod of TGA.

Table 2
The main factors of the algorithm with levels.

Factor	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7
N_1	45	40	30	20			
N_2	40	35	30	20			
N_4	50	40	30				
λ	1	0.5	0.1				
θ	1.4	1.3	1.2	1.1	1	0.9	0.8

There are other parameters that we consider them as known parameters and as the past algorithms set them. These parameters are as follows:

- **Max iteration:** that we have taken it 250 for large-scale problems and 50 for the small sizes.
- **nTree:** The trees population size of each iteration that it consider $N = 100$.
- And since that $N = N_1 + N_2 + N_3$, we select two more key parameters, N_1 and N_2, for Tuning and we have placed $N_3 = 100 - (N_1 + N_2)$.

According to Table 2, we understand that the number of trials for tuning is 7ˆ1 × 3ˆ2 × 4ˆ2 = 1008. Which their experiments is very difficult and time consuming, so instead of full factorial experiments we use the Taguchi table for tuning and the proposed Taguchi table is shown in the Table 3.

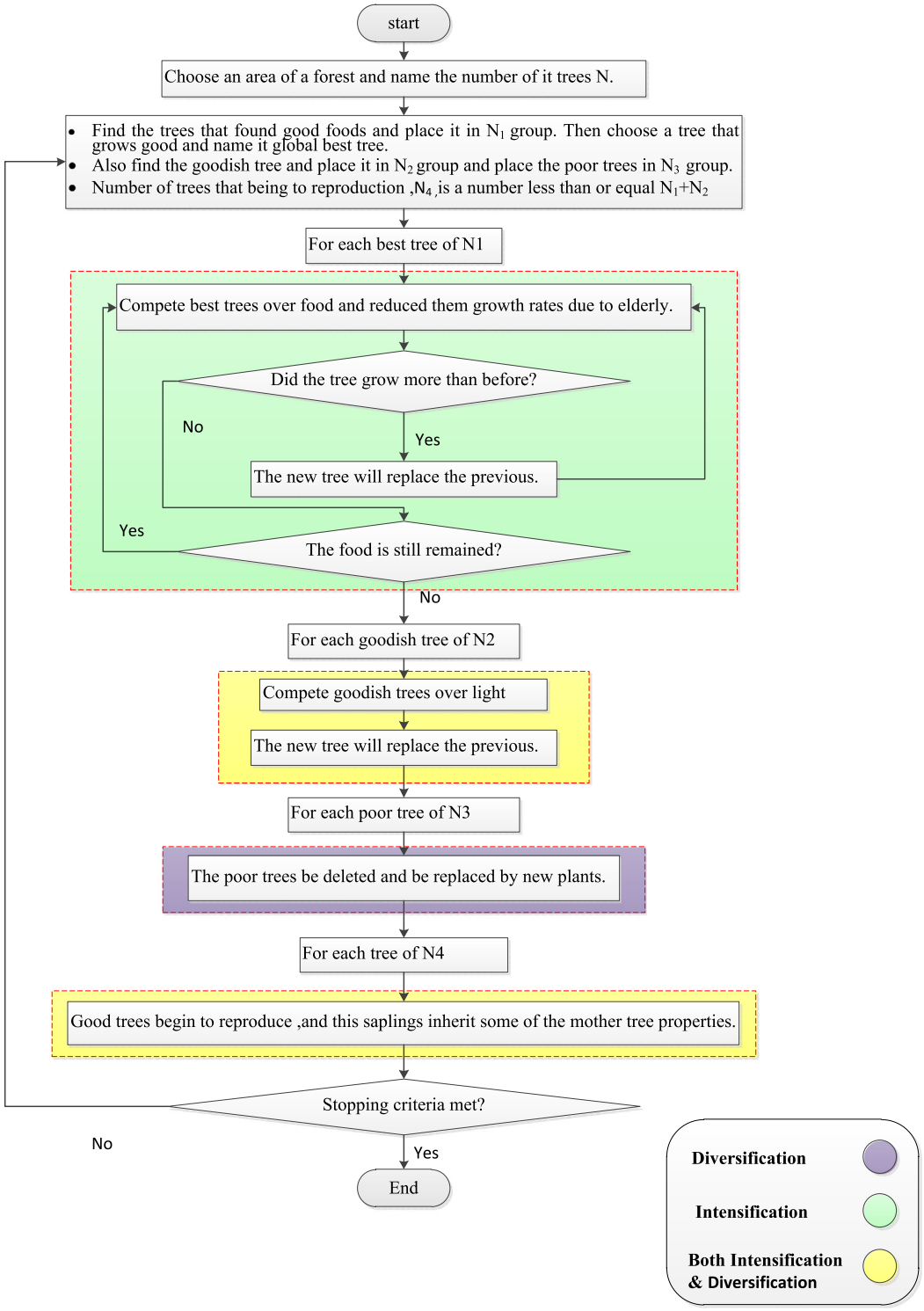

Fig. 8. TGA's flowchart.

We considered several benchmarks to tune and set average of 30 run for each benchmarks. We see the average of four benchmarks value, (Rastrigin Problem (RG), Rosenbrock Problem (RB), Easom Problem (EP), and Goldstein and Price Problem (GP)), per 30 run due to the value of parameters in Taguchi as shown in Table 4.

Then, for each factor in any benchmark, we average from benchmark values in the same levels and draw its figures as Figs. 9–12.

Since all benchmark function is minimization, so the best level of each factor is the lowest level of them. Since only for four benchmark tuned operation was performed, for another benchmark and engineering problems, we also use the obtained parameters.

3.2. Case 1: thirty benchmark function

Here thirty benchmark functions based on Baykasoğlu and Akpinar (2017) and Civicioglu (2013b) is selected which the descriptions of them are presented in Table 5. Also, the TGA performance was compared with seven optimization algorithms such as TS, CMAES, SA, WSA, BPA,

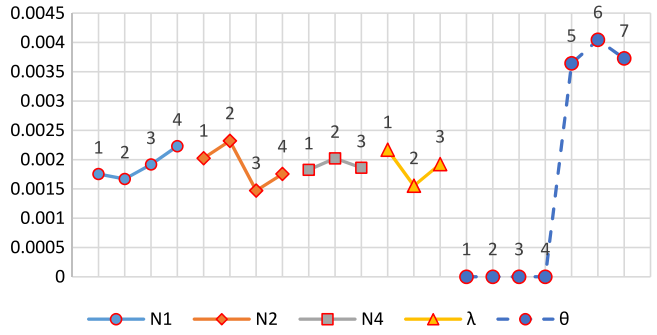

Fig. 9. Mean value plot for each level of the factors in Rastrigin function.

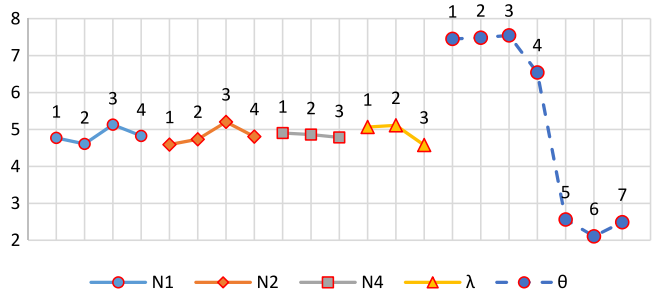

Fig. 10. Mean value plot for each level of the factors in Rosenbrock function.

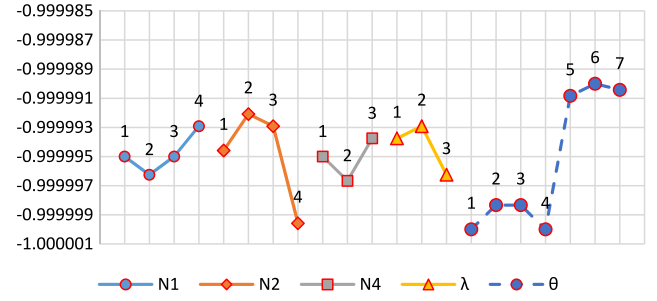

Fig. 11. Mean value plot for each level of the factors in Easom function.

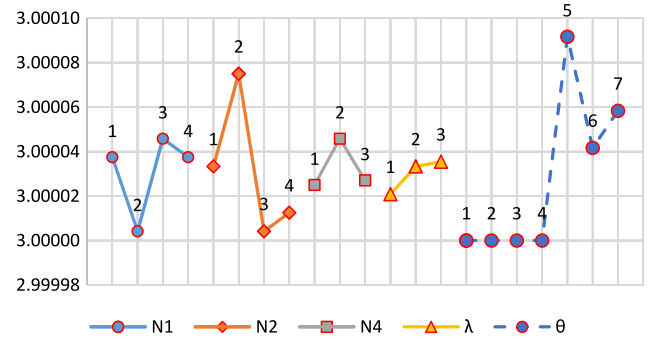

Fig. 12. Mean value plot for each level of the factors in Goldstein & Price function.

IBPA, and LADA which all of the problem are solved by Baykasoğlu and Akpinar (2017) and Civicioglu (2013b). In this paper, such as these two mentioned work, the values of *maxiter* and *ntree* are considered with 5000 and 20 respectively and each problem solved per 30 trial and any value less than 10^{-12} reported as zero.

Table 3
Original Taguchi table for this algorithm.

Experiment	Factor				
	N_1	N_2	N_4	λ	θ
1	1	1	1	1	1
2	1	2	3	3	2
3	2	3	2	1	3
4	2	4	3	3	4
5	3	1	3	2	5
6	3	2	1	3	6
7	4	3	3	2	7
8	4	4	2	3	7
9	1	3	3	3	4
10	1	4	1	2	3
11	2	1	3	3	2
12	2	2	2	2	1
13	3	3	1	3	7
14	3	4	3	1	7
15	4	1	2	3	6
16	4	2	3	1	5
17	1	3	2	3	5
18	1	4	3	1	6
19	2	1	1	3	7
20	2	2	3	1	7
21	3	3	3	3	1
22	3	4	2	2	2
23	4	1	3	3	3
24	4	2	1	2	4
25	1	1	3	2	7
26	1	2	2	3	7
27	2	3	3	2	6
28	2	4	1	3	5
29	3	1	2	1	4
30	3	2	3	3	3
31	4	3	1	1	2
32	4	4	3	3	1

The statistical results of the TGA and other algorithms on thirty benchmark problems are presented in Table 6. As it is shown, the proposed algorithm is far better than others in most of cases. Here, the bold values in the Table 6 refers to the best solution which TGA have overall good performance to satisfying these tests. Also, CMAES and WSA as the second and third best algorithms after TGA are shown.

The stated data in Table 6 is also visually compared in Fig. 13, which is designed by simply summing of all the thirty benchmarks under both the "Best" and "Mean" columns in Table 6 for each algorithm. The obtained results strongly confirm that the performance of the proposed algorithms in comparison with other 7 algorithms. Furthermore, these results are compared in term of total "AvgTime" as Fig. 14 which simply sums the "AvgTimes" values for each algorithm over the thirty benchmarks. According to this figure, it is clearly seen that TGA has less total execution times than other 7 algorithms. On the other hand, the reader should be aware that the algorithms were not compared by using the same machine. So, Fig. 14 should be commented within this direction.

3.2.1. Nonparametric statistical tests

It should be noted that the standard deviation and mean indicators are not enough lonely for comparing these algorithm. To this end, three well-known and strong nonparametric statistical tests include Wilcoxon test, Kruskal Wallis Test, and Friedman test are selected to more analyze the performance of these eight mentioned algorithms.

Wilcoxon signed-rank test is a nonparametric statistical test used to evaluate the similarity of two dependent degree-scale samples. This test takes into account the difference between rankings, so variables can have different or spaced solutions. This test, corresponds to the two dependent sample t test, and if not the conditions to perform the t test, Wilcoxon as a successor can be used. Also, the Friedman test is used for two-way analysis of variance (for nonparametric data) through

Table 4
Average of five benchmarks value per ten run due to Taguchi.

Exp.	Factor					Problem			
	N_1	N_2	λ	N_4	θ	Rastrigin	Rosenbrock	Easom	Goldstein & Price
1	45	40	1	50	1.4	0	7.349933	−1	3
2	45	35	0.1	30	1.3	0	7.388433	−1	3
3	40	30	1	40	1.2	0	7.480633	−0.99999	3
4	40	20	0.1	30	1.1	0	5.1708	−1	3
5	30	40	0.5	30	1	0.0037524	2.3251	−1	3.0002
6	30	35	0.1	50	0.9	0.0042161	1.511	−0.99998	3.000166
7	20	30	0.5	30	0.8	0.0027334	3.6235	−0.99998	3.000033
8	20	20	0.1	40	0.8	0.0040190	1.730133	−1	3.0001
9	45	30	0.1	30	1.1	0	5.6055	−1	3
10	45	20	0.5	50	1.2	0	7.521633	−1	3
11	40	40	0.1	30	1.3	0	7.470433	−1	3
12	40	35	0.5	40	1.4	0	7.6598	−1	3
13	30	30	0.1	50	0.8	0.003993	3.684233	−0.99998	3
14	30	20	1	30	0.8	0.0033724	3.4351	−1	3
15	20	40	0.1	40	0.9	0.0054032	1.21092	−0.99999	3
16	20	35	1	30	1	0.005677	1.666853	−0.99997	3.000166
17	45	30	0.1	40	1	0.0024381	3.653933	−0.99999	3
18	45	20	1	30	0.9	0.0039508	2.968566	−1	3
19	40	40	0.1	50	0.8	0.0036971	1.317726	−1	3.000033
20	40	35	1	30	0.8	0.0043569	2.431633	−0.99998	3
21	30	30	0.1	30	1.4	0	7.316233	−1	3
22	30	20	0.5	40	1.3	0	7.551433	−0.99999	3
23	20	40	0.1	30	1.2	0	7.634133	−0.99999	3
24	20	35	0.5	50	1.1	0	7.723866	−1	3
25	45	40	0.5	30	0.8	0.0033358	1.751633	−0.99997	3.000033
26	45	35	0.1	40	0.8	0.0043114	1.940933	−1	3.000266
27	40	30	0.5	30	0.9	0.0026064	2.7401	−0.99999	3
28	40	20	0.1	50	1	0.0027010	2.599156	−1	3
29	30	40	1	40	1.1	0	7.668166	−1	3
30	30	35	0.1	30	1.2	0	7.5524	−1	3
31	20	30	1	50	1.3	0	7.530766	−0.99999	3
32	20	20	0.1	30	1.4	0	7.4833	−1	3

Table 5
Test problems for the first part of the performance evaluation of TGA.

ID	Name of the problem	Dimension	Type	Range
F1	Ackley's Problem (ACK)	10	Multimodal	(−30,30)
F2	Aluffi–Pentini's Problem (AP)	2	Multimodal	(−10,10)
F3	Becker and Lago Problem (BL)	2	Multimodal	(−10,10)
F4	Bohachevsky 1 Problem (B1)	2	Multimodal	(−50,50)
F5	Bohachevsky 2 Problem (B2)	2	Multimodal	(−50,50)
F6	Branin Problem (BP)	2	Multimodal	$-5 \le x1 \le 10$ $0 \le x2 \le 15$
F7	Camel Back-3 Three Hump Problem (CB3)	2	Multimodal	(−5,5)
F8	Camel Back-6 Six Hump Problem (CB6)	2	Multimodal	(−5,5)
F9	Cosine Mixture Problem (CM)	2, 4	Unimodal	(−1,1)
F10	Dekkers and Aarts Problem (DA)	2	Multimodal	(−20,20)
F11	Easom Problem (EP)	2	Unimodal	(−10,10)
F12	Epistatic Michalewicz 5 Problem (EM5)	5	Multimodal	$(0, \pi)$
F13	Exponential Problem (EXP)	10	Unimodal	(−1,1)
F14	Goldstein and Price Problem (GP)	2	Multimodal	(−2,2)
F15	Gulf Research Problem (GRP)	3	Unimodal	(0,100)
F16	Hosaki Problem (HSK)	2	Multimodal	(0,6)
F17	Kowalik Problem (KL)	4	Multimodal	(0,0.42)
F18	Levy and Montalvo 1 Problem (LM1)	3	Multimodal	(−10,10)
F19	Levy and Montalvo 2 Problem (LM2)	5, 10	Multimodal	(−5,5)
F20	Meyer and Roth Problem (MR)	3	Multimodal	(−10,10)
F21	Miele and Cantrell Problem (MCP)	4	Multimodal	(−1,1)
F22	Modified Rosenbrock Problem (MRP)	2	Unimodal	(−5,5)
F23	Multi-Gaussian Problem (MGP)	2	Multimodal	(−2,2)
F24	Powell's Quadratic Problem (PWQ)	4	Multimodal	(−10,10)
F25	Rastrigin Problem (RG)	10	Multimodal	(−5.12,5.12)
F26	Rosenbrock Problem (RB)	10	Unimodal	(−30,30)
F27	Salomon Problem (SAL)	5, 10	Multimodal	(−100,100)
F28	Schaffer 1 Problem (SF1)	2	Multimodal	(−100,100)
F29	Schaffer 2 Problem (SF2)	2	Multimodal	(−100,100)
F30	Schwefel Problem (SWF)	10	Unimodal	(−500,500)

ranking. The nonparametric Friedman test's power is the same as the strongest parametric F test. To achieve more detail about these three nonparametric statistical tests, refers to the Derrac et al. (2011).

Based on Derrac et al. (2011), to implement these tests, the normalized score of each algorithm per each problem as a numbers between [0,1] is applied. To this end, first the best values of Table 6 are normalized and then using the SPSS software the results of analyzing

Table 6
Computational results for the benchmark set.

Function	$f(x^*)$	Results	BPA	IBPA	LADA	TS	SA	WSA	CMAES	TGA
F1	0	Best	0.00411	0.00815	0.0008822	0.14185	0.01261	0.888E−15	0.0000000012	**0.00**
		Mean	0.02843	0.02260	0.00473	0.38528	0.05488	0.888E−15	0.0000000018	**0.00**
		StdDev	0.01338	0.01021	0.00157	0.07488	0.05456	1.0029E−31	0.0000000004	0.00
		AvgTime	112	102	68	108	115	33	16	**12**
F2	≈−0.3523	Best	−0.35238	−0.35238	−0.35238	−0.35238	−0.35238	−0.35238	−0.35238	**−0.35239**
		Mean	−0.35238	−0.35238	−0.35238	−0.35228	−0.35213	−0.35236	−0.35238	**−0.35239**
		StdDev	1.449E−6	1.067E−6	5.576E−7	2.183E−5	6.654E−5	8.761E−6	0.00	6.09E−07
		AvgTime	71	65	36	66	75	20	12	**9**
F3	0	Best	1.017E−8	3.217E−9	1.259E−9	3.955E−7	1.374E−6	5.589E−8	**0.00**	1.07E−08
		Mean	2.697E−7	2.826E−7	2.486E−7	7.637E−6	2.772E−5	1.267E−7	**0.00**	3.70E−07
		StdDev	2.323E−7	2.838E−7	2.704E−7	6.302E−6	2.970E−5	3.877E−8	0.00	5.84E−07
		AvgTime	55	47	22	48	57	17	12	**9**
F4	0	Best	3.656E−5	2.362E−7	6.628E−6	1.123E−4	2.592E−5	0.00	0.00	**0.00**
		Mean	0.00179	6.419E−4	5.250E−4	0.02104	0.00276	0.00	0.00	**0.00**
		StdDev	0.00159	7.611E−4	0.00122	0.01801	0.00306	0.00	0.00	0.00
		AvgTime	62	55	29	57	70	24	13	**10**
F5	0	Best	2.927E−6	3.397E−6	4.911E−5	6.223E−4	1.589E−4	0.00	0.00	**0.00**
		Mean	0.00103	0.00786	0.00324	0.02354	0.00396	0.00	0.00	**0.00**
		StdDev	0.00144	0.03976	0.00643	0.02656	0.00667	0.00	0.00	0.00
		AvgTime	64	57	29	58	71	24	12	**9**
F6	5/4pi	Best	0.39788	0.39788	0.39788	0.39788	0.39788	0.39788	0.39788	**0.39789**
		Mean	0.39788	0.39788	0.39791	0.39792	0.39796	0.39790	0.39788	**0.39789**
		StdDev	2.361E−6	2.156E−6	9.373E−5	3.543E−5	8.443E−5	9.475E−6	0.00	0.00
		AvgTime	60	53	24	53	63	18	12	**9**
F7	0	Best	6.035E−10	5.405E−9	7.522E−9	1.385E−8	1.174E−6	0.00	0.00	**0.00**
		Mean	4.285E−7	3.199E−7	8.626E−7	1.503E−5	3.471E−5	0.00	0.00	**0.00**
		StdDev	6.255E−7	3.123E−7	1.752E−6	1.437E−5	3.458E−5	0.00	0.00	0.00
		AvgTime	94	86	58	86	96	25	14	**10**
F8	≈−1.0316	Best	−1.03162	−1.03162	−1.03162	−1.03162	−1.03162	−1.03161	**−1.03163**	−1.03163
		Mean	−1.03162	−1.03162	−1.03162	−1.03160	−1.03155	−1.03161	**−1.03163**	−1.03113
		StdDev	2.255E−6	2.221E−6	1.747E−6	2.060E−5	5.615E−5	4.516E−16	0.00	1.45E−03
		AvgTime	118	111	82	112	122	19	12	**9**
F9	0.2	Best	0.19999	0.19999	0.19999	0.19999	0.19999	0.2	0.2	**0.2**
		Mean	0.19999	0.19999	0.19999	0.19998	0.19984	0.2	0.2	**0.2**
		StdDev	2.647E−7	1.646E−7	6.431E−8	4.271E−6	3.309E−5	8.469E−17	0.00	0.00
		AvgTime	71	65	33	59	78	25	13	**11**
	0.4	Best	0.39999	0.39999	0.39987	0.39994	0.39991	0.4	0.4	**0.4**
		Mean	0.39999	0.39999	0.39873	0.39937	0.39855	0.4	0.4	**0.4**
		StdDev	4.354E−6	4.951E−6	6.909E−4	3.604E−4	7.050E−4	1.693E−16	0.00	0.00
		AvgTime	89	82	59	78	97	26	13	**11**
F10	−24,777	Best	−24,776.51	−24,776.51	−24,776.51	−24,776.51	−24,776.51	−24,776.52	**−24776.52**	−24775.72
		Mean	−24,776.47	−24,776.47	−24,776.39	−24,776.24	−24,776.46	−24,776.49	**−24776.52**	−24775.66
		StdDev	0.05370	0.05579	0.25194	0.27919	0.09251	0.01570	0.00	0.0944586327
		AvgTime	74	67	39	67	83	26	12	**9**
F11	−1	Best	−0.99999	−0.99999	−0.99999	−0.99999	−0.99999	−0.99999	**−1.00**	−0.999999
		Mean	−0.99999	−0.83334	−0.99999	−0.46667	−0.99991	−0.99957	**−1.00**	−0.999999
		StdDev	2.003E−6	0.3790100000	2.885E−6	0.5073300000	1.011E−4	2.025E−4	0.00	1.59E−06
		AvgTime	73	66	37	66	73	27	12	**9**
F12	≈−9.660152	Best	**−9.55986**	−9.52608	−9.27356	−8.97041	−9.52701	−7.20847	−8.61049	−7.83315
		Mean	−9.19363	**−9.28089**	−8.59255	−8.72569	−9.14595	−6.74067	−5.98581	−7.34378
		StdDev	0.21804	0.16721	0.26812	0.13066	0.22567	7.65571	1.06168	0.33437
		AvgTime	394	379	336	390	399	54	22	**17**
F13	1	Best	0.99998	0.99998	0.99986	0.99994	0.99207	1.00	1.00	**1.00**
		Mean	0.99994	0.99995	0.99968	0.99752	0.98487	1.00	1.00	**1.00**
		StdDev	2.469E−5	1.634E−5	1.123E−4	7.975E−4	0.0044600000	0.00	0.00	0.00
		AvgTime	97	89	39	87	103	30	14	**10**
F14	3	Best	3.00000	3.00000	3.00000	3.00000	3.00000	**3.00000**	3.00000	3.00207
		Mean	**3.00002**	5.70001	10.00710	3.00053	3.00049	3.00032	3.00020	3.11253
		StdDev	2.586E−5	8.23847	16.46670	5.751E−4	0.00114	1.622E−4	2.96E−16	1.34E−01
		AvgTime	51	45	16	45	58	24	12	**9**
F15	0	Best	**1.208E−6**	5.399E−6	8.124E−5	3.120E−5	2.352E−5	32.83	3.72892	1.52E−05
		Mean	0.00129	0.00157	5.362E−4	**2.047E−4**	2.868E−4	32.83	3.72892	7.05E−02
		StdDev	0.00130	0.00162	3.456E−4	1.382E−4	2.222E−4	1.445E−15	0.00	2.60E−01
		AvgTime	5234	5211	5205	5207	5269	121	78	**51**
F16	≈−2.3458	Best	−2.34581	−2.34581	−2.34581	−2.34581	−2.34581	−2.34581	−2.34581	**−2.34581**
		Mean	−2.34581	−2.34581	−2.34581	−2.34579	−2.34573	−2.34558	−2.34576	**−2.34581**
		StdDev	8.203E−8	1.356E−7	5.327E−8	3.169E−6	1.994E−5	1.631E−4	3.273E−6	1.36E−07
		AvgTime	57	50	22	50	58	27	15	**9**
F17	≈3.0748E−4	Best	3.075E−4	3.075E−4	3.105E−4	**3.075E−4**	3.087E−4	3.079E−4	3.086E−4	3.084E−04
		Mean	3.105E−4	3.118E−4	3.684E−4	**3.083E−4**	3.204E−4	3.111E−4	3.204E−4	3.144E−04
		StdDev	3.488E−6	4.096E−6	6.400E−5	6.638E−7	6.842E−6	1.706E−6	6.648E−7	7.402E−06
		AvgTime	89	84	54	83	91	57	22	**10**
F18	0	Best	1.005E−7	1.095E−7	8.240E−8	3.124E−6	3.515E−5	3.315E−7	**0.00**	3.47E−04

(*continued on next page*)

Table 6 (*continued*)

Function	$f(x^*)$	Results	BPA	IBPA	LADA	TS	SA	WSA	CMAES	TGA
		Mean	7.457E−6	4.979E−6	2.322E−6	1.436E−4	3.392E−4	6.705E−6	**0.00**	2.87E−02
		StdDev	6.041E−6	6.716E−6	2.867E−6	1.186E−4	2.257E−4	3.392E−6	0.00	2.83E−02
		AvgTime	84	76	48	76	86	37	23	**19**
F19	0	Best	0.00138	0.01882	0.18022	0.17907	0.00706	0.00103	0.00	**0.00**
		Mean	0.30253	0.55175	2.62975	0.57022	0.27072	0.02390	0.00	**0.00**
		StdDev	0.27903	0.48275	0.67992	0.32601	0.25005	0.01237	0.00	0.00
		AvgTime	64	58	36	58	72	29	27	**18**
	0	Best	0.00387	0.00962	0.06966	0.05080	0.01122	0.00257	0.02836	**0.00173**
		Mean	0.35532	0.47384	0.59860	0.45002	0.35846	0.08629	0.36690	**0.34527**
		StdDev	0.40486	0.41149	0.41381	0.24229	0.28039	0.05220	0.31849	0.40532
		AvgTime	65	58	36	58	72	33	31	**20**
F20	≈0.4E−4	Best	4.356E−5	**4.358E−5**	4.426E−5	4.357E−5	4.574E−5	4.409E−5	4.773E−5	4.058E−05
		Mean	6.049E−5	6.084E−5	6.496E−5	4.832E−5	5.903E−5	**4.773E−5**	9.99E−05	4.992E−05
		StdDev	3.054E−5	2.270E−5	2.264E−5	7.299E−6	1.027E−5	2.271E−6	5.182E−05	4.972E−05
		AvgTime	68	61	34	61	70	25	32	**23**
F21	0	Best	6.684E−12	5.606E−12	5.260E−8	3.376E−11	1.098E−9	8.934E−13	0.00	**0.00**
		Mean	9.185E−9	1.120E−8	2.627E−6	1.353E−9	4.094E−7	2.414E−10	0.00	**0.00**
		StdDev	8.576E−9	1.235E−8	3.129E−6	2.167E−9	5.113E−7	1.753E−10	0.00	0.00
		AvgTime	147	140	127	140	152	79	12	**10**
F22	0	Best	1.960E−6	5.865E−7	3.210E−8	1.623E−5	1.732E−5	4.291E−9	0.00	**0.00**
		Mean	8.632E−4	0.00211	0.00147	7.907E−4	0.00105	**6.812E−7**	2.97E−03	2.23E−03
		StdDev	0.00237	0.00403	0.00218	7.948E−4	1.037E−4	3.871E−7	3.83E−03	3.58E−03
		AvgTime	59	52	23	52	64	17	12	**9**
F23	≈1.29695	Best	−1.29695	−1.29695	−1.29695	−1.29694	−1.29694	−1.29695	−1.29556	**−1.29695**
		Mean	**−1.29695**	−1.27557	−1.29160	−1.29681	−1.29665	−1.29666	−1.26188	−1.29540
		StdDev	4.888E−6	0.03605	0.02033	1.286E−4	3.470E−4	1.778E−4	0.03632	0.00489
		AvgTime	121	114	86	114	121	35	13	**10**
F24	0	Best	1.651E−5	2.294E−7	3.530E−4	0.00132	1.113E−4	0.00	0.0000001	**0.00**
		Mean	0.00136	0.00102	0.03571	0.01224	0.00356	0.00	0.0000005	**0.00**
		StdDev	0.00133	0.00136	0.03693	0.00897	0.00330	0.00	0.0000003	0.00
		AvgTime	97	91	69	92	106	29	12	**10**
F25	0	Best	0.16961	0.08790	0.00606	4.58753	0.03932	0.00	4.97480	**0.00**
		Mean	0.43289	0.29275	0.01584	6.35541	0.15916	0.00	14.61362	**0.00**
		StdDev	0.16469	0.12481	0.00554	0.89405	0.09522	0.00	9.55280	0.00
		AvgTime	133	125	61	134	141	32	22	**14**
F26	0	Best	3.3005	1.6578	13.1161	24.7395	0.9187	8.9167	**0.0201**	0.5653
		Mean	13.5681	12.1420	26.4740	66.1024	6.4109	8.9449	**0.0603**	0.8231
		StdDev	17.9707	14.9202	14.9521	19.1763	1.8180	0.0160	0.0439	0.3419
		AvgTime	89	83	34	83	98	25	14	**9**
F27	0	Best	0.09987	0.09987	0.09987	0.09996	0.09987	**0.00**	0.09987	0.09987
		Mean	0.20329	0.18655	0.14497	0.30807	0.20668	**0.00**	0.09988	0.13987
		StdDev	0.09609	0.07301	0.04966	0.08101	0.09040	0.00	0.00001	0.05164
		AvgTime	67	60	26	60	69	20	17	**13**
	0	Best	0.29988	0.29987	0.30989	1.03152	0.30006	**0.00**	0.10408	0.09987
		Mean	0.45003	0.47596	0.57499	1.39242	0.59059	**0.00**	0.17422	0.11987
		StdDev	0.08947	0.12510	0.08872	0.16112	0.15625	0.00	0.04158	0.04216
		AvgTime	91	84	40	86	97	24	14	**11**
F28	0	Best	1.2051E−4	1.979E−5	1.535E−6	1.409E−4	6.671E−4	0.00	0.00777	**0.00**
		Mean	0.00749	0.00536	0.00557	0.00621	0.00840	0.00	0.00984	**0.00**
		StdDev	0.00362	0.00474	0.00444	0.00343	0.00277	0.00	0.00123	0.00
		AvgTime	65	62	28	56	63	25	16	**13**
F29	0	Best	0.06332	0.08329	0.21116	0.58920	0.06096	8.937E−77	0.127875	**0.00**
		Mean	0.16367	3.21774	3.24847	3.63356	0.30990	1.201E−76	0.233297	**0.00**
		StdDev	0.05498	1.40596	1.33647	0.77425	0.35340	1.343E−77	0.076864	0.00
		AvgTime	102	103	67	95	104	23	15	**14**
F30	≈−418.9829D	Best	−4188.83	−4188.80	**−4189.79**	−4146.86	−4189.64	−4080.13	−8340.0386	−4189.702
		Mean	−4186.57	−4187.48	**−4189.74**	−4099.80	−4189.39	−3505.52	−6835.1837	−4189.292
		StdDev	1.41713	0.90048	0.03670	23.24570	0.23796	328.20	750.73	0.23796
		AvgTime	137	129	71	134	145	27	25	**18**

Note: $f(x^*)$ refers optimum objective value of the functions. The bold values are the best solutions found by the algorithms

Table 7
Wilcoxon signed ranks test results.

Comparison	$R+$	$R-$	p-value	Results
TGA versus CMAES	110	215	0.158	–
TGA versus WSA	166	134	>0.2	–
TGA versus SA	435	126	0.006	TGA shows an improvement over SA, with a level of significance $\alpha = 0.01$
TGA versus TS	435	126	0.006	TGA shows an improvement over TS, with a level of significance $\alpha = 0.01$
TGA versus LADA	421	140	0.012	TGA shows an improvement over LADA, with a level of significance $\alpha = 0.01$
TGA versus IBPA	392	162	0.046	TGA shows an improvement over IBPA, with a level of significance $\alpha = 0.05$
TGA versus BPA	398	163	0.036	TGA shows an improvement over BPA, with a level of significance $\alpha = 0.05$

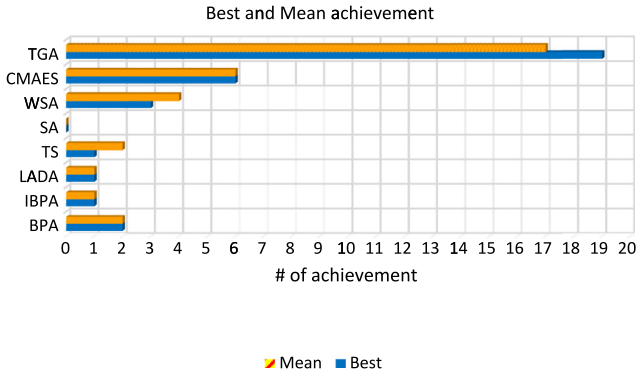

Fig. 13. Visually comparison of the 8 algorithms.

Fig. 14. Total execution times for algorithm over 33 functions.

Table 8
Ranks achieved by the Friedman and Kruskal Wallis tests.

Algorithms	Mean rank		Final priority
	Friedman	Kruskal Wallis	
BPA	4.258	123.38	4
IBPA	4.167	121.08	3
LADA	4.894	146.82	6
TS	5.424	180.98	8
SA	5.045	144.70	7
WSA	4.136	111.41	2
CMAES	4.515	128.85	5
TGA	3.561	102.79	1
Test statistics	Friedman	Kruskal Wallis	
N	33	33	
Chi-Square	15.849	24.129	
df	7	7	
p-value	0.0265	0.00108	

methods are computed. The obtained results of Wilcoxon signed-rank test are provided in Table 7. As is clear, TGA shows an improvement versus SA, TS, and LADA with a level of significance $\alpha = 0.01$, and versus IBPA and BPA with a level of significance $\alpha = 0.05$. It should be noted that R^+ and R^- represents the sum of positive ranks and some of negative ranks, respectively.

Also, the Friedman ranks test and Kruskal Wallis test for 33 benchmarks are performed which the obtained results are presented in Table 8. Based on these results, the first rank is related to the TGA and it show the best performance of the proposed algorithm. Also, the values of Chi-Square, *df*, and *p*-value are provided in this table. Finally, based on the results of these three nonparametric tests, the strong performance of TGA versus others 7 employed algorithms is proved.

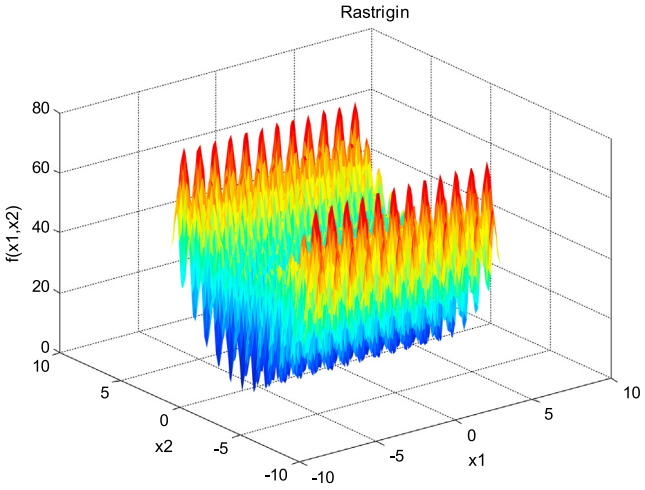

Fig. 15. The 3D plot of two-dimensional Rastrigin problem RG (2).

3.2.2. Convergence analysis of *TGA*

Here the convergence behavior of the proposed TGA algorithm on two benchmark functions is analyzed. To this end, Rastrigin Problem (RG) and Schaffer1 Problem (SF1) are selected as illustration. TGA have been generated with 100 initial tree (population) and performed per 150 iterations for both functions. Then, the initial population and the population after every 30 iterations are scattered to visualize the convergences of the population during the implementation of TGA. Two dimensional RG (2) is a minimization problem and has a minima located at the origin with the function value of 0, and 3D plot of this function is shown in Fig. 15. At first, the contour plot of the RG (2) is figured and then the initial population are scattered on first contour plot. After that, per each 30 iterations, the changed population are scattered on a different contour plot which these results are illustrated in Fig. 16. These results indicate the convergence behavior of the TGA toward the global optima and TGA has a satisfactory level of convergence speed. As randomly generated initial population shows the satisfactory level of diversification, and convergence speed to find the global optima shows the satisfactory level of intensification. Also, Fig. 17 is illustrated to better presentation of the TGA convergence speed and obtained fitness value. Based on these results, it can be report that TGA have high speed and strong convergence behavior on Rastrigin Problem, and it find the global optima (0) after less than 15 iteration.

Another part of the convergence analysis is done on Schaffer1 Function (SF1). Similarly, this problem is a minimization problem and the global optima value of SF1 is equal to 0. The 3D plot of this function is shown in Fig. 18. Likewise previous problem, at first, the contour plot of the SF1 is figured and then the initial population are scattered on first contour plot. After that, per each 30 iterations, the changed population are scattered on a different contour plot which these results are illustrated in Fig. 19. These results indicate the convergence behavior of the TGA toward the global optima and TGA has a satisfactory level of convergence speed. As randomly generated initial population shows the satisfactory level of diversification, and convergence speed to find the global optima shows the satisfactory level of intensification. Also, Fig. 20 is provided to better illustration of the TGA convergence speed and obtained fitness value. Due to these results, it can be find that TGA have high speed and strong convergence behavior on Schaffer1 Function, and it provide the global optima (0) after less than 25 iteration.

3.3. Case 2: Three engineering optimization

Since, the engineering optimization problems presented in real world are usually non-linear and involve with complex geometrical

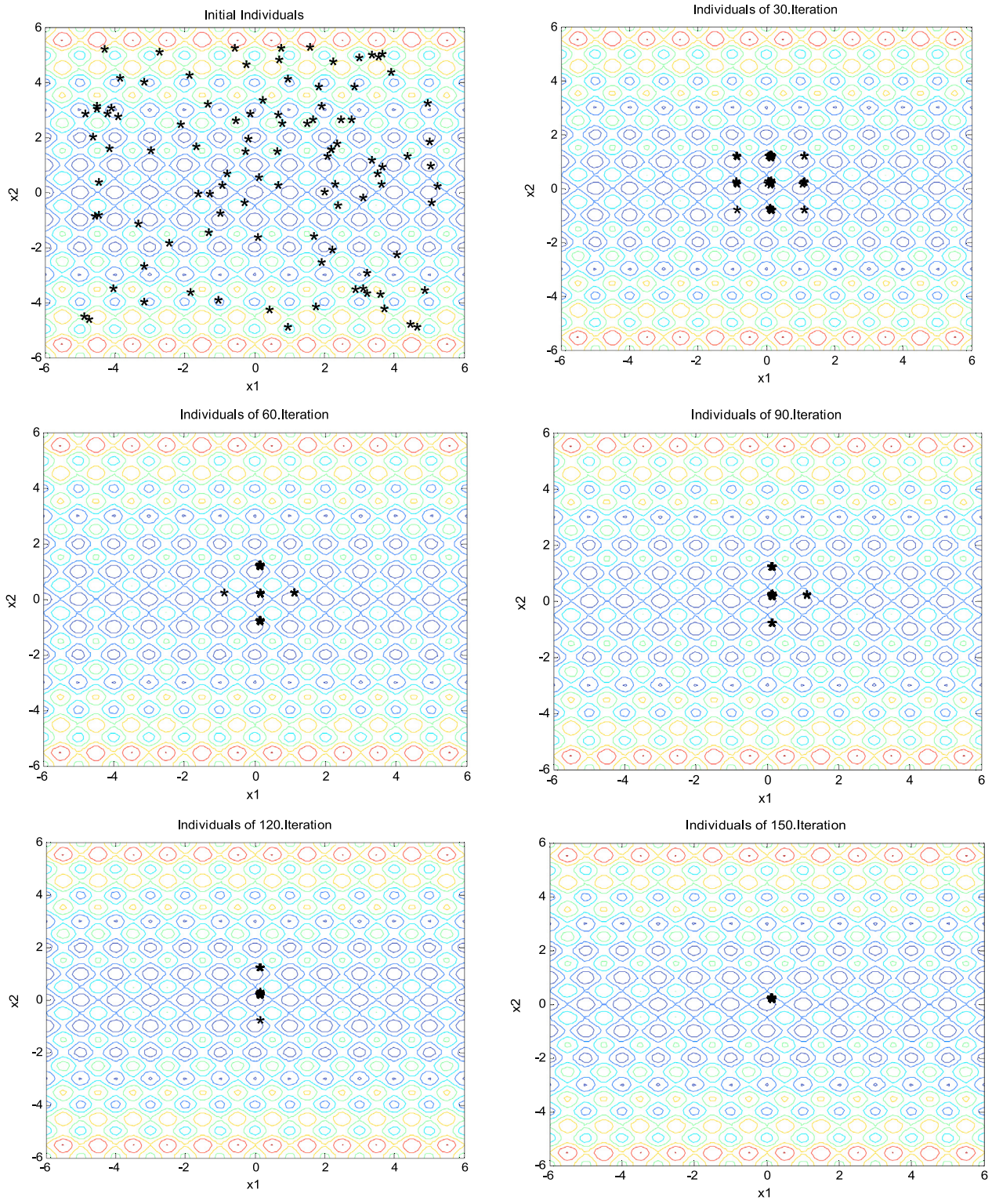

Fig. 16. Convergence behavior of TGA for Rastrigin problem RG (2).

and mechanical constraints, thus to measure performance of TGA, three constrained engineering design problems are applied.

3.3.1. Welded beam design

The introduced TGA was utilized to the design of a well-known welded beam problem for minimum overall cost of construction (see Fig. 21). This engineering problem is well-known in the literature, and in this paper the model formulation and results of other algorithms by adapt on the Gandomi (2014) are used. After implementation of TGA through this problem, the obtained results of Table 9 confirm the efficiency of TGA. The global optimum obtained by this algorithm with value of **2.2555** as the best value among the applied algorithms is selected. That is while the maximum number of evaluation equal to 30 000 is considered and relevant feasible solution is obtained as $x = [0.19579801, 7.59365961, 9.04767033, 0.20574969]$.

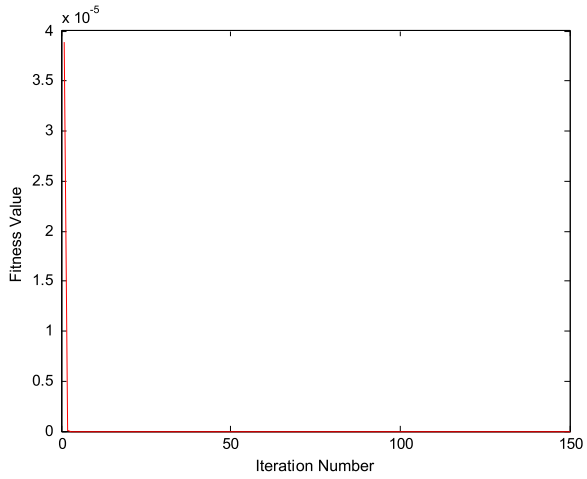

Fig. 17. Fitness value of the best tree versus iteration number for Rastrigin problem RG (2).

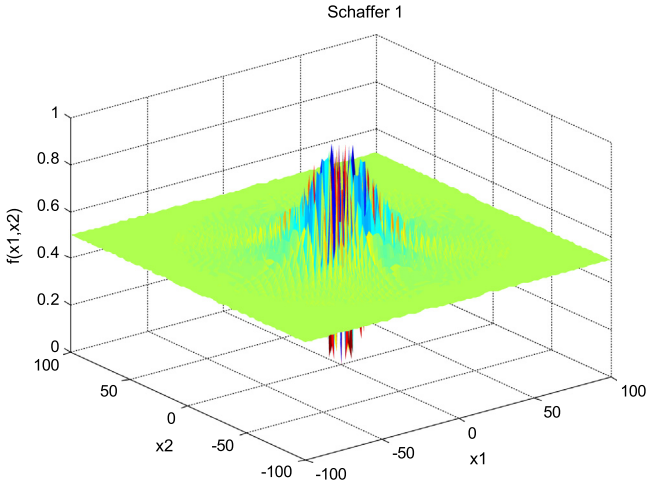

Fig. 18. The 3D plot of two-dimensional Schaffer1 Function (SF1).

Table 9
Best results of the welded beam design example using different methods.

Method	Best	Mean	Worst	Std. dev.	No. eval.
SBS	2.4426	N/A	N/A	N/A	20,000
TCA	2.3811	2.7104	2.4398	9.31E−2	320,000
GA–AP1	2.3814	5.9480	3.4956	9.09E−1	320,000
GA–AIS1	2.3812	2.3899	2.4139	N/A	320,000
GA–AIS2	2.3834	4.0560	2.9930	2.02E−1	320,000
GA	2.4331	N/A	N/A	N/A	40,080
GA–PLP	2.3812	N/A	2.6458	N/A	320,080
GA–AP2	2.3816	2.4172	2.9553	N/A	320,000
MOEA	2.3829	2.4209	N/A	2.56E−2	80 000
BFO	2.3868	2.4040	N/A	1.60E−2	48 000
SCA	2.3854	3.2550	6.3997	9.60E−1	33 095
GA–SR	2.5961	10.1833	4.3326	1.29E+0	320 000
AATM	2.3823	2.3870	2.3916	2.20E−3	30 000
ISA	2.3812	2.4973	2.6700	1.02E−1	30 000
TGA	**2.2555**	**2.7170**	**2.8323**	**0.74036**	30 000

3.3.2. Pressure vessel design

Pressure vessel design as another engineering problem is selected to compare performance of TGA versus other algorithms. This problem is shown in Fig. 22 and similar with previous problem is adapted on Gandomi (2014) and results of this study is used to compare with TGA.

Table 10
Statistical results of the pressure vessel design example by different models.

Method	Best	Mean	Worst	Std. dev.	No. eval.
SBS	6171.000	6335.050	6453.650	N/A	20,000
TCA	6390.554	7694.067	6737.065	357	80,000
GA–AP1	6065.821	8248.003	6632.376	515	80,000
GA–AP2	6060.188	6311.766	6838.939	N/A	80,000
GA–AIS1	6060.138	6845.496	6385.942	N/A	80,000
GA–AIS2	6059.854	7388.160	6545.126	124	80,000
GA-CP	6288.745	6293.840	6308.150	7.4133	900,000
GA-DP	6127.414	6616.933	7572.659	358.85	2,500,000
GA–AIS3	6061.120	6734.090	7368.060	458	150,000
GA–SR	6832.584	8012.615	7187.314	267	80,000
GA	6059.946	6177.250	6469.320	130.93	80,000
PSO1	6061.078	6147.130	6363.800	86.455	200,000
PSO2	6544.270	9032.550	11,638.200	995.57	100,000
CPSO	6363.804	6147.133	6061.078	86.450	240,000
MOEA	6059.926	6172.527	N/A	124	80,000
BFO	6060.460	6074.625	N/A	15.6	48,000
BCO	6059.768	6060.210	N/A	0.0069	240,000
DE	6059.734	6085.230	6371.050	43.013	240,000
AATM	6059.726	6061.988	6090.802	4.70	30,000
ISA	6059.714	6410.087	7332.846	384.6	5,000
TGA	**5981.536**	**6554.637**	**7523.677**	**391.601**	**5,000**

Table 11
Statistical results of the spring design example using different methods.

Method	Best	Mean	Worst	Std. dev.	No. eval.
TCA	**0.012665**	**0.012732**	**0.013309**	**9.40E−5**	**36,000**
GA–AP	0.012679	0.014022	0.017794	1.47E−3	36,000
GA-AIS	0.012666	0.013880	0.012974	N/A	36,000
GA-AIS	0.012666	0.013131	0.015318	6.28E−4	36,000
GA–SR	0.012680	0.013993	0.017796	1.27E−3	36,000
GA–CP	0.012705	0.012769	0.012822	3.94E−5	900,000
GA	0.012681	0.012742	0.012973	5.90E−5	30,000
PSO	0.012675	0.012730	0.012924	5.20E−5	200,000
PSO	0.013120	0.022948	0.050365	7.21E−3	100,000
CPSO	0.012924	0.012730	0.012674	5.20E−4	240,000
SA-DS	**0.012665**	**0.012665**	**0.012665**	**N/A**	**49,531**
HS	0.012666	N/A	N/A	N/A	200,000
HS	0.012671	N/A	N/A	N/A	50,000
MOEA	0.012680	0.012960	N/A	3.63E0−4	80 000
BFO	0.012671	0.012759	N/A	1.36E0−4	48 000
BCO	0.012667	0.012700	N/A	2.4E−7	240 000
SCA	0.012669	0.012923	0.016717	5.92E−4	25 167
AATM	0.012668	0.012708	0.012861	4.50E−5	25 000
ISA	**0.012665**	**0.013165**	**0.012799**	**1.59E−2**	**8 000**
TGA	**0.012665**	**0.021408**	**0.022712**	**0.018501**	**5 000**

After implementation of TGA through this problem, the achieved results of Table 10 approve the efficiency of TGA. The global optimum obtained by this algorithm with value of **5981.536** as the best value among the applied algorithms is selected. That is while the maximum number of evaluation equal to 5000 is considered and relevant feasible solution is obtained as $x = [0.79338939, 0.38658012, 40.46600627, 197.97648814]$.

3.3.3. Tension/compression spring design

Third engineering problem used in this paper is Tension/compression spring design as illustrated in Fig. 23. Likewise previous problems, this problem is adapted on Gandomi (2014) and results of this study is used to compare with TGA. After implementation of TGA through this problem, the achieved results of Table 11 approve the efficiency of TGA. The global optimum obtained by this algorithm with value of **0.012665** as the best value among the applied algorithms as well as ISA, SA-DS, and TCA is selected. That is while the maximum number of evaluation equal to 5000 is considered.

3.4. Case 3: Two industrial engineering optimization

In this section we consider two industrial engineering problems to prove performance of TGA. Because of the most popular problems in the operation research area we choice single machine and transportation

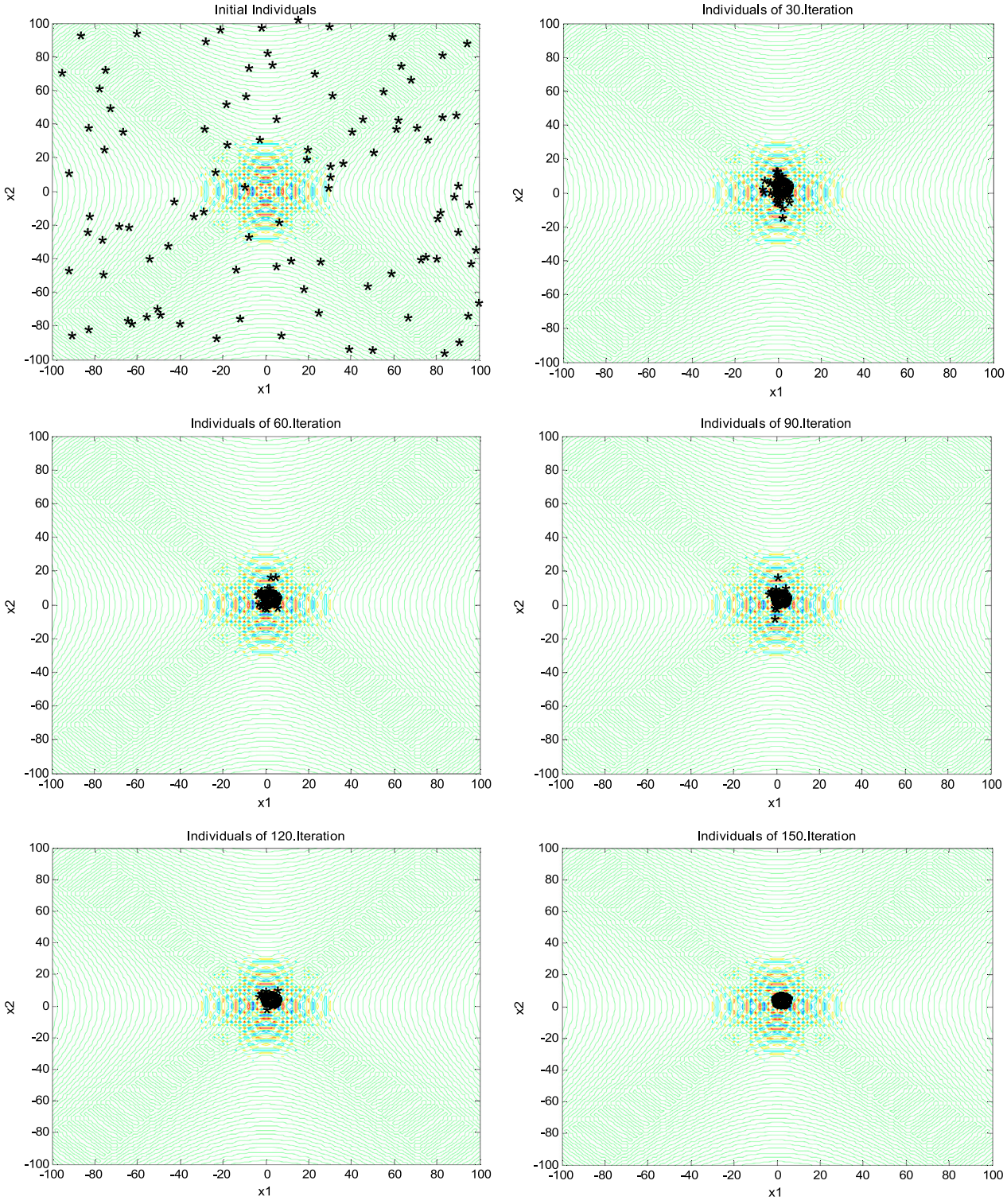

Fig. 19. Convergence behavior of TGA for Schaffer1 Function (SF1).

problems to evaluate the performance of this algorithm and compare it with other algorithms mentioned in follow. The SA, GA and PSO are selected for comparison. For SA initial temperature as T0 = 99, decreasing parameter $\alpha = 0.99$, and maximum iteration equal to 20 000 is considered. To implement the GA, we consider the mutation probability as MR = 0.1 and crossover probability as CR = 0.8, as considered in the previous papers for the problems. In PSO the acceleration constants equal to (C_1, C_2 = 2) and inertia weight (ω) begins with 0.9, and also linearly decreases to 0.1 during the iterations to do both exploration and exploitation (Gandomi, 2014; Mafarja and Mirjalili, 2017). Also, maximum iteration number for PSO and GA equal to 200 is assumed.

3.4.1. *Single machine scheduling*

Single machine scheduling is one of industrial engineer problems that we present it in this section for prove the efficiency of TGA. Single machine scheduling environment is actually used in various industries (Wagner et al., 2002). Writing articles on the subject of timing and scheduling model earliness/tardiness, due to the importance

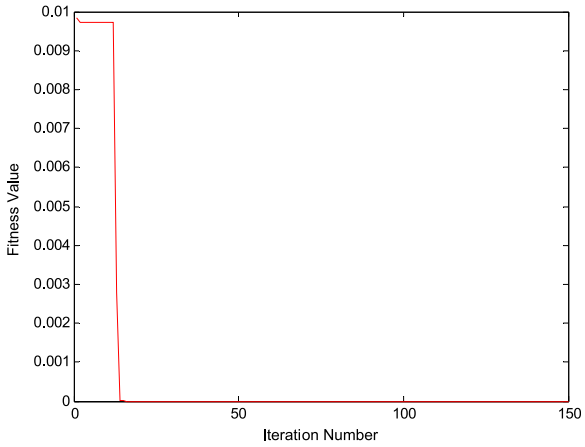

Fig. 20. Fitness value of the best tree versus iteration number for Schaffer1 Function (SF1).

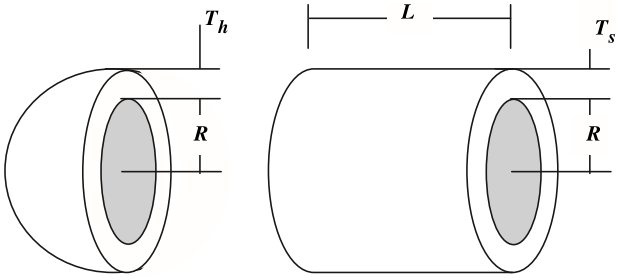

Fig. 21. Schematic of the welded beam design problem (Gandomi, 2014).

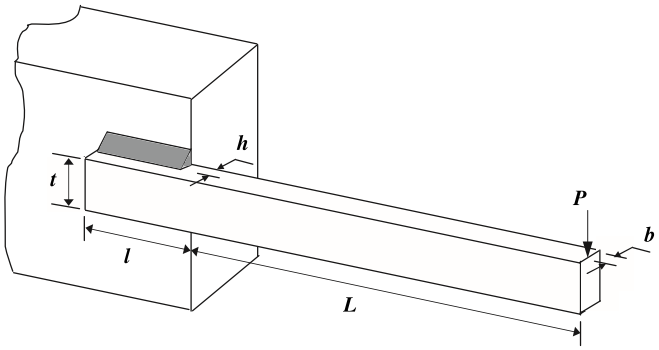

Fig. 22. Schematic of the pressure vessel design problem (Gandomi, 2014).

and relevance of their use is increasing significantly (Hendel et al., 2009). Production Planning Manager, in these organizations considers undesirable earliness and tardiness. In fact, scheduling problems with earliness and tardiness crimes are compatible with concepts such as Just In Time (JIT) and supply chain management. JIT production stresses that earliness costs, such as tardiness can be reduced. This is because earliness duties it is possible to warehouse costs, including lost opportunity of investment in stock, insurance costs and leads waste of warehouse. Instead, work that to be prepared later than delivery time, is possible lead to customer dissatisfaction, contractual penalties, loss of sales and credits (Liao and Cheng, 2007).

Single machine scheduling problems with earliness and tardiness costs without allowing standstill as NP-*hard* problem has been studied extensively by Lenstra and Section (1977). Tardiness penalty in power two in various industries, proper use and used to work with delays. In fact, delay as an important characteristic in service quality and customer

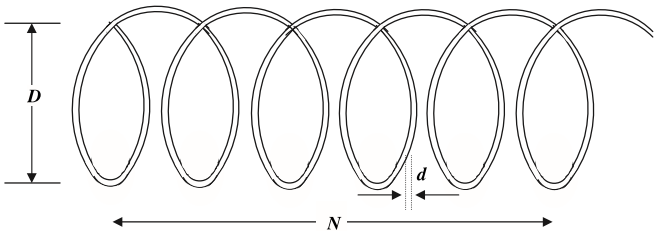

Fig. 23. Schematic of the tension/compression spring design problem (Gandomi, 2014).

dissatisfaction tends to increase tardiness as the cost function (Taguchi, 1986). Also, Tardiness penalty, in some situations can be compared to the maximum delay functions or linear delays are normally preferred (Sun et al., 1999). This issue by considering the work breakdown structure in single machine scheduling problem with allowed idle time $\sum_{i=1}^{n} \alpha_i E_i + \beta_i T_i$, is also investigated by Hendel et al. (2009).

Indices:

i: Jobs indices
j: Period time (sequence) indices

Parameters:

d_i: Delivery time for ith job
p_i: Production time for ith job
α_i: Earliness penalties for ith job
β_i: Tardiness penalties for ith job

Variables:

E_i: Earliness time for ith job
T_i: Tardiness time for ith job
C_i: Completion time for ith job

$$X_{ij} = \begin{cases} 1 & \text{if the } i\text{th job happen in } j\text{th period time} \\ 0 & \text{o.w.} \end{cases}$$

Objective function:

$$\min Z = \sum_{i}^{n} \alpha_i \times E_i + \beta_i \times T_i \tag{5}$$

$$s.t.:$$

$$\sum_{i} X_{ij} = 1 \tag{6}$$

$$\sum_{j} X_{ij} = 1 \tag{7}$$

$$C_i = \sum_{i} P_i \tag{8}$$

$$E_i = \max(0, C_i - d_i) \tag{9}$$

$$T_i = \max(0, d_i - C_i) \tag{10}$$

$$X_{ij} = \{0, 1\}. \tag{11}$$

The goal (5) is to find a schedule that minimizes the total cost of earliness and tardiness and former limitation ensures that any job can happen only once at any solution and second limitation ensures that in any time can happen just a job. In limitation of (8) guarantees Completion time for any jobs and, in limitation of (9) and (10) calculated Earliness and tardiness time for any job.

According to the above description, we know that the chromosome of this problem is sequence of doing jobs. In this paper we consider 20 jobs, 150 iteration and parameters of problem was presented in Table 12.

To compare the performance of this algorithm we solve this problem with some well-known algorithms and the results were presented in Table 13.

Table 12
Parameters of single machine scheduling.

Job	P_i	d_i	α_i	β_i
1	4	3	2	4
2	3	5	2	4
3	5	7	2	4
4	2	6	2	4
5	3	8	2	4
6	5	9	2	4
7	7	7	2	4
8	6	8	2	4
9	3	12	2	4
10	2	13	2	4
11	9	16	2	4
12	10	19	2	4
13	8	20	2	4
14	2	12	2	4
15	1	9	2	4
16	12	10	2	4
17	10	20	2	4
18	6	14	2	4
19	9	12	2	4
20	1	11	2	4

Table 13
Result of single machine scheduling.

Problem	TGA	PSO	SA	GA
Scheduling	2264	2276	2314	2356
Gap	0	0.0053	0.0220	0.0406

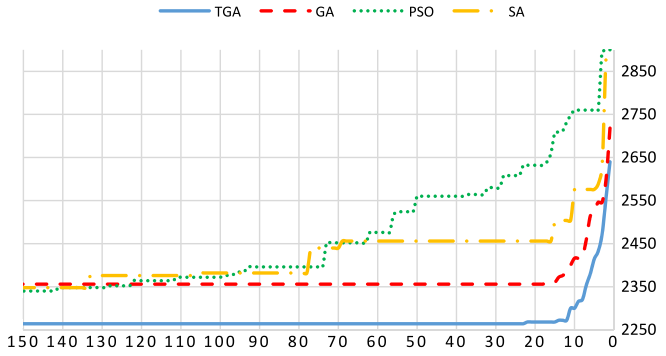

Fig. 24. The algorithms' behavior in single machine scheduling.

After this experiment we see that the proposed algorithm, TGA, shows better performance among Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Simulation Annulling (SA) in this problem as shown in Fig. 24.

This performance by the gap's Equation ($Gap = \frac{alg-best}{best}$) assessment that we see the TGA contains the smallest gap.

3.4.2. Transportation

Another engineering problems that we consider in this paper for prove the efficiency of TGA, is TSP. Traveling salesman problem (TSP) is one of the most important problems in optimizing compounds which is used in many engineering sciences and has attracted the attention of scientists and researchers. In this problem the seller from the desired point in the name of the warehouse began to move and after meeting the n customer returns to the starting location.

Each customer is visited only once. The purpose of this problem determine the route with the minimum cost for the seller. In this problem the number of possible solutions for the n points regardless of the warehouse, is equal with number of permutation which is composed of numbers from 1 to n. The purpose of this problem is find a permutation with least cost of moving. Since the number of possible solutions of this problem is $(n!)$, by increasing problem nodes, number of feasible

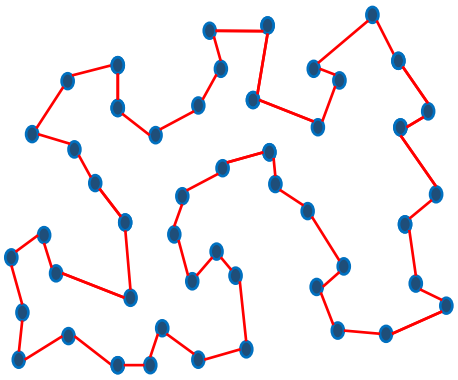

Fig. 25. General plan of TSP function.

solution grows strongly and no longer can easily compare all solutions in a reasonable time and achieve to optimal solution.

In addition, TSP using in many practical problems such as timing of vehicle (Park, 2001), timing crews plane (Saleh and Chelouah, 2004), a combination Chinese postman problem (Chan and Mercier, 1989), ordering Workshop (Lawler et al., 1985), mission design for Independent movable robots (Brumitt and Stentz, 1996) and connect the computer (Zhang and Korf, 1995). The applications of this problem is not limited to those listed and to find complete information can be used this Ref. Bektas (2006).

Indices:

i: Beginning nodes indices

j: Destination nodes indices

Parameters:

C_{ij}: Cost of any movement

Variables:

$$X_{ij} = \begin{cases} 1 & \text{if Seller move from } i \text{ to } j \text{ nodes} \\ 0 & \text{o.w.} \end{cases}$$

Objective function:

$$\min Z = \sum_{i=0}^{n} \sum_{j=0}^{n} C_{ij} \times X_{ij} \tag{12}$$

$$s.t.:$$

$$\sum_{i=0}^{n} X_{ij} = 1 \quad \forall i = 1, 2, \ldots, n \tag{13}$$

$$\sum_{j=0}^{n} X_{ij} = 1 \quad \forall j = 1, 2, \ldots, n \tag{14}$$

$$X_{ij} = \{0, 1\}. \tag{15}$$

In this model, objective (12) is to reduce the cost of transportation and first limitation category shows that, in each node only enters a bow. The second limitation category refers to the fact that in each node only take out a bow and the last limitation show that X_{ij} is binary. Fig. 25 is the general plan of the TSP problem.

According to the above description, we know that the chromosome of this problem is sequence of moving seller. In this paper, at one dimension we consider 15 cities, 200 iteration and parameters of problem was presented in Table 14.

To compare the performance of this algorithm we solve this problem with some algorithms were studied and the results were presented in Table 15.

After this experiment we see that purposed algorithm, TGA, shows better performance among Genetic Algorithm (GA), Particle Swarm

Table 14
Cost of each movement C_{ij} (randomly).

i	j														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.00	30.59	18.87	15.62	20.22	30.53	50.54	51.09	53.23	15.62	29.15	33.62	25.46	36.40	23.35
2	30.59	0.00	45.65	24.08	10.44	60.17	65.80	71.12	42.58	26.91	55.73	63.32	53.67	66.76	53.94
3	18.87	45.65	0.00	22.80	35.51	16.00	62.37	59.62	72.01	20.40	34.44	19.03	8.25	24.19	17.69
4	15.62	24.08	22.80	0.00	15.26	38.47	65.19	66.53	59.03	2.83	44.65	41.59	31.05	46.10	35.51
5	20.22	10.44	35.51	15.26	0.00	49.77	60.11	64.07	45.10	18.03	46.57	52.92	43.42	56.32	43.57
6	30.53	60.17	16.00	38.47	49.77	0.00	61.07	55.44	80.60	36.22	31.02	3.16	8.25	8.54	12.37
7	50.54	65.80	62.37	65.19	60.11	61.07	0.00	11.66	46.10	65.86	30.07	62.29	63.20	59.03	49.00
8	51.09	71.12	59.62	66.53	64.07	55.44	11.66	0.00	56.72	66.71	25.30	56.14	58.94	52.01	44.15
9	53.23	42.58	72.01	59.03	45.10	80.60	46.10	56.72	0.00	61.39	58.25	83.24	77.70	83.49	69.43
10	15.62	26.91	20.40	2.83	18.03	36.22	65.86	66.71	61.39	0.00	44.20	39.32	28.64	44.01	33.96
11	29.15	55.73	34.44	44.65	46.57	31.02	30.07	25.30	58.25	44.20	0.00	32.25	33.73	29.41	19.10
12	33.62	63.32	19.03	41.59	52.92	3.16	62.29	56.14	83.24	39.32	32.25	0.00	11.05	6.40	14.32
13	25.46	53.67	8.25	31.05	43.42	8.25	63.20	58.94	77.70	28.64	33.73	11.05	0.00	16.76	14.87
14	36.40	66.76	24.19	46.10	56.32	8.54	59.03	52.01	83.49	44.01	29.41	6.40	16.76	0.00	14.21
15	23.35	53.94	17.69	35.51	43.57	12.37	49.00	44.15	69.43	33.96	19.10	14.32	14.87	14.21	0.00

Table 15
Result of algorithms for TSP.

Problem	TGA	PSO	SA	GA
TSP	248.03	332.93	317.13	349.55
Gap	0	0.3422	0.2786	0.4093

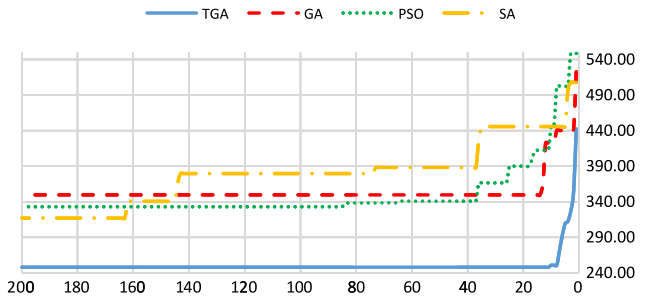

Fig. 26. The algorithms' behavior in TSP function.

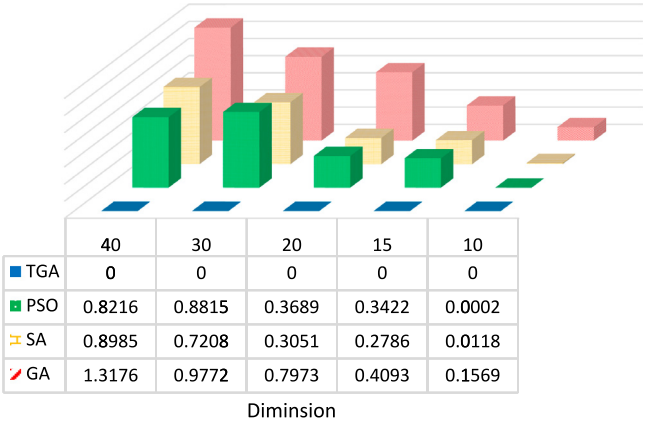

Fig. 27. Amount of algorithms' gap with TGA in various dimension.

Table 16
Result of TSP in various dimension.

Dimension	Problem	TGA	PSO	SA	GA
10	TSP	**234.80**	234.85	237.56	271.65
	Gap	**0**	0.0002	0.0118	0.1569
	Time	8.12	6.52	**1.86**	5.23
15	TSP	248.03	332.93	317.13	349.55
	Gap	0	0.3422	0.2786	0.4093
	Time	9.93	6.72	**2.1**	6.71
20	TSP	**271.57**	371.76	354.42	488.09
	Gap	**0**	0.3689	0.3051	0.7973
	Time	10.41	6.71	**3.14**	6.82
30	TSP	**311.75**	586.57	536.45	616.4
	Gap	**0**	0.8815	0.7208	0.9772
	Time	10.47	7.02	**3.1**	7.2
40	TSP	**374.38**	681.96	710.75	867.67
	Gap	**0**	0.8216	0.8985	1.3176
	Time	11.43	7.89	**3.42**	7.35

Optimization (PSO) and Simulation Annulling (SA) in this problem as shown in Fig. 26.

This performance is depicted by the gap's Equation ($Gap = \frac{alg-best}{best}$) assessment that we see the TGA contains the smallest gap.

For further comparison, we solved this problem for different dimensions. After performing the experiment, the comparing results in Table 16 shows the superiority of TGA in all dimensions.

As illustrated in the Fig. 27, as the dimensions of the problem increases, the gap between the results of the TGA in comparison with other algorithms will be larger.

4. Conclusion and future research

In this paper a novel population-based optimization algorithm was proposed as an alternative for solving optimization problems among the current techniques in the literature. Special characteristics of Trees growing behavior had been the basic motivation for development of this new optimization algorithm. This algorithm is different from the previous ones in some new concepts. At first, the TGA is very simple to code and use in different problem types. Also solutions focus in the trees fierce competition for food. It searches cleverly to find the better solution in the neighborhood of possible solutions, in the intensification phase. Besides, by changing or tuning the N_1, N_2, N_3, N_4, θ and λ, the tradeoff between intensification and diversification can be obtained or tuned for a problem by its properties and dimensions.

The introduced algorithm was tested on thirty benchmark cost functions and several engineering functions. The comparison of TGA with standard versions of CMAES, TS, SA, WSA, BPA, IBPA, and LADA showed the superiority of TGA in these problems in the algorithms' behavior and finding the global optima. According to the results of conducted tests, the TGA can be considered as a successful metaheuristic and suitable for optimization problems. Also, to more analyze the results, some well-known nonparametric statistical tests are utilized. The results demonstrate that there is a clear statistically significant difference between performances of TGA and other algorithms. After this analysis, we found that TGA obtained answers has a better quality than other.

This paper can be expand and improve with several research instructions for future studies. As a direction to develop the current algorithm, we propose binary and multi objective version of the algorithm that can

be employed in binary and multiple objectives respectively. Swirling, levy flight, crossover, and other relevant operators can be utilized to this algorithm in order to improve its performance. Also, hybridization of TGA algorithm with other algorithms is proposed. And last but not least, the performance of TGA can be tested on the other real world engineering optimization problems.

References

Alatas, B., 2012. A novel chemistry based metaheuristic optimization method for mining of classification rules. Expert Syst. Appl. 39, 11080–11088.

Alok, A.K., Saha, S., Ekbal, A., 2015. A new semi-supervised clustering technique using multi-objective optimization. Appl. Intell. 43, 633–661. http://dx.doi.org/10.1007/s10489-015-0656-z.

Alsheddy, A., 2011. Empowerment scheduling: a multi-objective optimization approach using guided local search. Sch. Comput. Sci. Electron. Eng. University.

Atashnezhad, A., Wood, D.A., Fereidounpour, A., Khosravanian, R., 2014. Designing and optimizing deviated wellbore trajectories using novel particle swarm algorithms. J. Nat. Gas Sci. Eng. 21, 1184–1204. http://dx.doi.org/10.1016/j.jngse.2014.05.029.

Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In: 2007 IEEE Congress on Evolutionary Computation. IEEE, pp. 4661–4667. http://dx.doi.org/10.1109/CEC.2007.4425083.

Azamathulla, H.M., 2012. Linear Programming for Irrigation Scheduling - A Case Study. Nova Science Publishers.

Azamathulla, H.M., 2013. A review on application of soft computing methods in water resources engineering. In: Metaheuristics in Water, Geotechnical and Transport Engineering. Elsevier, pp. 27–41. http://dx.doi.org/10.1016/B978-0-12-398296-4.00002-7.

Bandyopadhyay, S., Saha, S., Maulik, U., Deb, K., 2008. A simulated annealing-based multiobjective optimization algorithm: AMOSA. IEEE Trans. Evol. Comput. 12, 269–283. http://dx.doi.org/10.1109/TEVC.2007.900837.

Basturk, B., Karaboga, D., 2006. An artificial bee colony (ABC) algorithm for numeric function optimization. In: IEEE swarm Intell. Symp.

Battiti, R., Brunato, M., 2010. Reactive search optimization: Learning while optimizing. In: Handb. Metaheuristics, Vol. 146. pp. 543–571. http://dx.doi.org/10.1007/978-1-4419-1665-5_18.

Baykasoğlu, A., Akpinar, Ş., 2017. Weighted Superposition Attraction (WSA): A swarm intelligence algorithm for optimization problems–Part 1: Unconstrained optimization. Appl. Soft Comput. 56, 520–540. http://dx.doi.org/10.1016/j.asoc.2015.10.036.

Bektas, T., 2006. The multiple traveling salesman problem: An overview of formulations and solution procedures. Omega 34, 209–219. http://dx.doi.org/10.1016/j.omega.2004.10.004.

Briggs, W.R., 2014. Phototropism: Some history, some puzzles, and a look ahead. J. Plant Physiol. 164, 13–23. http://dx.doi.org/10.1104/113.230573.

Brumitt, B.L., Stentz, A., 1996. Dynamic Mission Planning for Multiple Mobile Robots, pp. 2396–2401.

Chan, D., Mercier, D., 1989. IC insertion: an application of the travelling salesman problem. Int. J. Prod. Res. 27, 1837. http://dx.doi.org/10.1080/00207548908942657.

Cheng, M.-Y., Prayogo, D., 2014. Symbiotic Organisms Search: A new metaheuristic optimization algorithm. Comput. Struct. 139, 98–112.

Cheraghalipour, A., Hajiaghaei-keshteli, M., 2017. Tree growth algorithm (TGA): An effective metaheuristic algorithm inspired by trees behavior. In: 13th International Conference on Industrial Engineering. Scientific Information Databases, Babolsar, pp. 1–8.

Cheraghalipour, A., Paydar, M.M., Hajiaghaei-keshteli, M., 2017. An integrated approach for collection center selection in reverse logistic. Int. J. Eng. Trans. A Basics 30, 1005–1016. http://dx.doi.org/10.5829/ije.2017.30.07a.10.

Christie, J.M., Murphy, A.S., 2013. Shoot phototropism in higher plants: New light through old concepts. Am. J. Bot. 100, 35–46. http://dx.doi.org/10.3732/ajb.1200340.

Civicioglu, P., 2012. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. Comput. Geosci. 46, 229–247. http://dx.doi.org/10.1016/j.cageo.2011.12.011.

Civicioglu, P., 2013a. Artificial cooperative search algorithm for numerical optimization problems. Inf. Sci. (Ny). 229, 58–76.

Civicioglu, P., 2013b. Backtracking Search Optimization Algorithm for numerical optimization problems. Appl. Math. Comput. 219, 8121–8144. http://dx.doi.org/10.1016/j.amc.2013.02.017.

Coomes, D.A., Allen, R.B., 2007. Effects of size, competition and altitude on tree growth. J. Ecol. 95, 1084–1097. http://dx.doi.org/10.1111/j.1365-2745.2007.01280.x.

Darwin, C., Darwin, F., 1880. Power of Movement in Plants. John Murray, New york.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6, 182–197. http://dx.doi.org/10.1109/4235.996017.

Derrac, J., García, S., Molina, D., Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. 1, 3–18. http://dx.doi.org/10.1016/j.swevo.2011.02.002.

Dogan, B., Ölmez, T., 2015. A new metaheuristic for numerical function optimization: Vortex Search algorithm. Inf. Sci. (Ny). 293, 125–145.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. Ant system : Optimization by a colony of cooperating agents. IEEE Trans. Syst. Man. Cybern. 26, 29–41.

Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory. In: MHS'95. Proc. Sixth Int. Symp. Micro Mach. Hum. Sci. pp. 39–43. https://doi.org/10.1109/MHS.1995.494215.

Ebrahimi, A., Khamehchi, E., 2016. Sperm whale algorithm: an effective metaheuristic algorithm for production optimization problems. J. Nat. Gas Sci. Eng. http://dx.doi.org/10.1016/j.jngse.2016.01.001.

Farmer, J.D., Packard, N., Perelson, A., 1986. The immune system, adaptation and machine learning. Physica D 2, 187–204.

Fogel, L.J., Owens, A.J., Walsh, M.J., 1966. Artificial Intelligence Through Simulated Evolution. John Wiley.

Fonseca, C.M., Fleming, P.J., 1993. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. Icga 93, 416–423. https://doi.org/citeulike-article-id:2361311.

Friml, J., Wiśniewska, J., Benková, E., Mendgen, K., Palme, K., 2002. Lateral relocation of auxin efflux regulator PIN3 mediates tropism in Arabidopsis. Nature 415, 806–809. http://dx.doi.org/10.1038/415806a.

Gandomi, A.H., 2014. Interior search algorithm (ISA): A novel approach for global optimization. ISA Trans. 53, 1168–1183. http://dx.doi.org/10.1016/j.isatra.2014.03.018.

Gandomi, A.H., Alavi, A.H., 2012. Krill herd: a new bio-inspired optimization algorithm. Commun. Nonlinear Sci. Numer. Simul. 17 (12), 4831–4845.

Gandomi, A.H., Toropov, V.V., Sahab, M.G., 2013a. 2 A review on traditional and modern structural optimization: Problems and techniques. In: Metaheuristic Applications in Structures and Infrastructures, first ed. Elsevier Inc. http://dx.doi.org/10.1016/B978-0-12-398364-0.00002-4.

Gandomi, A.H., Yang, X., Talatahari, S., Alavi, A.H., 2013b. 1 metaheuristic algorithms in modeling and optimization. In: Metaheuristic Applications in Structures and Infrastructures, first ed. Elsevier Inc. http://dx.doi.org/10.1016/B978-0-12-398364-0.00001-2.

Geem, Z.W., Kim, J.-H., Loganathan, G.V., 2001. A new heuristic optimization algorithm: Harmony search. Simulation 76, 60–68. http://dx.doi.org/10.1177/003754970107600201.

Glover, F., 1977. Heuristics for integer programming using surrogate constraints. Decis. Sci. 8, 156–166. http://dx.doi.org/10.1111/j.1540-5915.1977.tb01074.x.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. 13 (5), 533–549.

Golshahi-Roudbaneh, A., Hajiaghaei-Keshteli, M., Paydar, M.M., 2017. Developing a lower bound and strong heuristics for a truck scheduling problem in a cross-docking center. Knowl.-Based Syst. 129, 17–38. http://dx.doi.org/10.1016/j.knosys.2017.05.006.

Gough, L.A., Birkemoe, T., Sverdrup-Thygeson, A., 2014. Reactive forest management can also be proactive for wood-living beetles in hollow oak trees. Biol. Cons. 180, 75–83. http://dx.doi.org/10.1016/j.biocon.2014.09.034.

Haddad, O.B., Afshar, A., Mariño, M.A., 2006. Honey-bees mating optimization (HBMO) algorithm: A new heuristic approach for water resources optimization. Water Resour. Manage. 20, 661–680. http://dx.doi.org/10.1007/s11269-005-9001-3.

Haga, K., Sakai, T., 2012. PIN auxin efflux carriers are necessary for pulse-induced but not continuous light-induced phototropism in arabidopsis. J. Plant Physiol. 160, 763–776. http://dx.doi.org/10.1104/112.202432.

Hajiaghaei-Keshteli, M., Aminnayeri, M., 2012. Keshtel algorithm (KA); A new optimization algorithm inspired by keshtels ' feeding. In: Proceeding IEEE Conf. Ind. Eng. Manag. Syst. Vol. 1, pp. 2249–2253.

Hajiaghaei-Keshteli, M., Aminnayeri, M., 2014. Solving the integrated scheduling of production and rail transportation problem by Keshtel algorithm. Appl. Soft Comput. 25, 184–203. http://dx.doi.org/10.1016/j.asoc.2014.09.034.

Hansen, P., Mladenovic, N., Perez, J.A.M., 1997. Variable neighbourhood search: methods and applications. Ann. Oper. Res. 201 (175), 367–407.

Hansen, N., Ostermeier, A., 2001. Completely derandomized self-adaptation in evolution strategies. Evol. Comput. 9, 159–195. http://dx.doi.org/10.1162/106365601750190398.

Hanseth, O., Aanestad, M., 2001. Bootstrapping networks, communities and infrastructures. On the evolution of ICT solutions in heath care. In: First Int. Conf. Inf. Technol. Heal. Care (ITHC, Sept. 6–7, 2001). Erasmus Univ. Rotterdam, Netherlands.

Hastings, W., 1970. Monte Carlo sampling methods using Markov Chains and their applications. Biometrika 57, 97–109.

Hatamlou, A., 2013. Black hole: A new heuristic optimization approach for data clustering. Inf. Sci. (Ny). 222, 175–184. http://dx.doi.org/10.1016/j.ins.2012.08.023.

He, S., Wu, Q.H., Saunders, J., 2009. Group search optimizer: an optimization algorithm inspired by animal searching behavior. Evol. Comput. IEEE Trans. 13 (5), 973–990.

Hendel, Y., Runge, N., Sourd, F., 2009. Discrete Optimization The one-machine just-in-time scheduling problem with preemption 6, pp. 10–22. https://doi.org/10.1016/j.disopt.2008.08.001.

Holland, J.H., 1975. Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. Q. Rev. Biol. 1, 211. http://dx.doi.org/10.1086/418447.

Hooke, R., Jeeves, T.A., 1961. Direct search: solution of numerical and statistical problems. J. Assoc. Comput. 8 (2), 212–229.

Husseinzadeh Kashan, A., 2009. League Championship Algorithm: a new algorithm for numerical function optimization. In: Proc. IEEE Int. Conf. Soft Comput. Pattern Recognit. SoCPaR 20, pp. 43–48.
Husseinzadeh Kashan, A., 2013. A new metaheuristic for optimization: Optics inspired optimization (OIO). Comput. Oper. Res. 55, 99–125.
Husseinzadeh Kashan, A., 2014. League Championship Algorithm (LCA): An algorithm for global optimization inspired by sport championships. Appl. Soft Comput. 16, 171–200.
Husseinzadeh Kashan, A., Akbari, A.A., Ostadi, B., 2015. Grouping evolution strategies: An effective approach for grouping problems. Appl. Math. Model. 39, 2703–2720. http://dx.doi.org/10.1016/j.apm.2014.11.001.
Kadioglu, S., Sellmann, M., 2009. Dialectic search. In: Proc. Int. Conf. Princ. Pract. Constraint Program. (CP), Springer, pp. 486–500.
Kaveh, A., Mahdavi, V.R., 2015. Colliding Bodies Optimization, Colliding Bodies Optimization: Extensions and Applications. Springer International Publishing, Cham, . http://dx.doi.org/10.1007/978-3-319-19659-6.
Kernighan, B.W., Lin, S., 1970. An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. 49, 291–307. http://dx.doi.org/10.1002/j.1538-7305.1970.tb01770.x.
Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220, 671–680. http://dx.doi.org/10.1126/science.220.4598.671. (80-.).
Koza, J., 1992. Genetic Programming: On the Programming of Computers by Natural Selection. MIT Press, Cambridge, MA.
Krishnanand, K., Ghose, D., 2005. Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In: Proc. IEEE Swarm Intell. Symp. pp. 84–91.
Krishnanand, K., Ghose, D., 2006. Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. Multiagent Grid Syst. 2, 209–222.
Larrañaga, P., Lozano, J.A., 2002. Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Bost. Kluwer Acad. Publ.
Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R., Shmoys, D.B., 1985. The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. In: Wiley Series in Discrete Mathematics & Optimization, Wiley, p. 476.
Lenstra, J.K., Section, I., 1977. Complexity of machine scheduling problems 1, pp. 343–362.
Li, T., Wang, Z., 2008. Application of plant growth simulation algorithm on solving facility location problem. Syst. Eng. - Theory Pract. 28, 107–115. http://dx.doi.org/10.1016/S1874-8651(10)60025-7.
Li, H., Zhang, J., Vierstra, R.D., Li, H., 2010. Quaternary organization of a phytochrome dimer as revealed by cryoelectron microscopy. Proc. Natl. Acad. Sci. USA 107, 10872–10877. http://dx.doi.org/10.1073/pnas.1001908107.
Liang, Y., Cuevas Juarez, J.R., Wang, H., Lai, Y., 2016. A novel metaheuristic for continuous optimization problems: Virus optimization algorithm. Eng. Optim. 48, 73–93. http://dx.doi.org/10.1080/0305215X.2014.994868.
Liao, C., Cheng, C., 2007. A variable neighborhood search for minimizing single machine weighted earliness and tardiness with common due date q 52, pp. 404–413. https://doi.org/10.1016/j.cie.2007.01.004.
Liu, Y., Passino, K.M., 2002. Biomimicry of social foraging bacteria for distributed optimization: Models, principles, and emergent behaviors. J. Optim. Theory Appl. 115, 603–628. http://dx.doi.org/10.1023/A:1021207331209.
Lourenço, H.R., Martin, O., Stützle, T., 2010. Iterated local search: Framework and applications. In: Handboo. ed. In: International Series in Operations Research & Management Science.
Mafarja, M.M., Mirjalili, S., 2017. Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. Neurocomputing http://dx.doi.org/10.1016/j.neucom.2017.04.053.
Matyas, J., 1965. Random optimization. Autom. Remote Control 26 (2), 246–253.
Mirjalili, S., 2016a. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput. Appl. 27, 1053–1073. http://dx.doi.org/10.1007/s00521-015-1920-1.
Mirjalili, S., 2016b. SCA: A sine cosine algorithm for solving optimization problems. Knowl.-Based Syst. http://dx.doi.org/10.1016/j.knosys.2015.12.022.
Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M., 2017. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Adv. Eng. Softw. 114, 163–191. http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.
Moscato, P., 1989. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Caltech Concurr. Comput. Progr. (report 82).
Mucherino, A., Seref, O., 2007. Monkey search: a novel metaheuristic search for global optimization. In: AIP Conference Proceedings. AIP, pp. 162–173. http://dx.doi.org/10.1063/1.2817338.
Nakrani, S., Tovey, C., 2004. On honey bees and dynamic server allocation in internet hosting centers. Adapt. Behav. 12, 223–240. http://dx.doi.org/10.1177/105971230401200308.
Park, Y.-B., 2001. A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines. Int. J. Prod. Econ. 73, 175–188. http://dx.doi.org/10.1016/S0925-5273(00)00174-2.
Pham, D.T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S., Zaidi, M., 2005. The Bees Algorithm - Technical Report. Cardiff Manuf. Eng. Centre, Cardiff Univ.
Rajabioun, R., 2011. Cuckoo optimization algorithm. Appl. Soft Comput. 11, 5508–5518. http://dx.doi.org/10.1016/j.asoc.2011.05.008.
Rao, R.V., Savsani, V.J., Vakharia, D.P., 2011. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Comput. Des. 43, 303–315. http://dx.doi.org/10.1016/j.cad.2010.12.015.
Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., 2009. GSA: A gravitational search algorithm. Inf. Sci. (Ny). 179, 2232–2248. http://dx.doi.org/10.1016/j.ins.2009.03.004.
Rastrigin, L.A., 1963. The convergence of the random search method in the extremal control of a many parameter system. Autom. Remote Control 24 (10), 1337–1342.
Robbins, H., Monro, S., 1951. A stochastic approximation method. Ann. Math. Stat. 22, 400–407. http://dx.doi.org/10.1214/aoms/1177729586.
Rubinstein, R.Y., 1997. Optimization of computer simulation models with rare events. European J. Oper. Res. 99, 89–112.
Ruxton, G.D., Schaefer, H.M., 2012. The conservation physiology of seed dispersal. Philos. Trans. R. Soc. Lond. B. Biol. Sci. 367, 1708–1718. http://dx.doi.org/10.1098/rstb.2012.0001.
Sadeghi-Moghaddam, S., Hajiaghaei-Keshteli, M., Mahmoodjanloo, M., 2017. New approaches in metaheuristics to solve the fixed charge transportation problem in a fuzzy environment. Neural Comput. Appl. http://dx.doi.org/10.1007/s00521-017-3027-3.
Sadollah, A., Bahreininejad, A., Eskandar, H., Hamdi, M., 2013. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. Appl. Soft Comput. J. 13, 2592–2612. http://dx.doi.org/10.1016/j.asoc.2012.11.026.
Saleh, H.A., Chelouah, R., 2004. The design of the global navigation satellite system surveying networks using genetic algorithms. Eng. Appl. Artif. Intell. 17, 111–122. http://dx.doi.org/10.1016/j.engappai.2003.11.001.
Salimi, H., 2014. Stochastic fractal search: A powerful metaheuristic algorithm. Knowl.-Based Syst. 75, 1–18.
Shah-Hosseini, H., 2009. The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. Int. J. Bio-Inspired Comput. 1 (1/2), 71–79.
Shah-Hosseini, H., 2011. Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation. Int. J. Comput. Sci. Eng. 6, 132. http://dx.doi.org/10.1504/IJCSE.2011.041221.
Smith, H., 2000. Phytochromes and light signal perception by plants–an emerging synthesis. Nature 407, 585–591. http://dx.doi.org/10.1038/35036500.
Srinivas, N., Deb, K., 1994. Multiobjective function optimization using nondominated sorting genetic algorithms. Evol. Comput. 2, 221–248.
Storn, R., Price, K., 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 34, 1–359. http://dx.doi.org/10.1023/A:1008202821328.
Sun, X., Noble, J.S., Klein, C.M., 1999. Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness.
Taguchi, G., 1986. Introduction to quality engineering: designing quality into products and processes, illustrate. ed. The Organization, White Plains.
Taillard, ric D., Voss, S., 1999. POPMUSIC: Partial Optimization Metaheuristic Under Special Intensification Conditions, Technical Report (Institute for Computer Sciences).
Tamura, K., Yasuda, K., 2011a. Primary study of spiral dynamics inspired optimization. IEE J. Trans. Electr. Electron. Eng. 6, S98–S100. http://dx.doi.org/10.1002/tee.20628.
Tamura, K., Yasuda, K., 2011b. Spiral dynamics inspired optimization. J. Adv. Comput. Intell. Intell. Inform. 15 (8), 1116–1122.
Teodorović, D., 2009. Bee colony optimization (BCO). Innov. Swarm Intell. 39–60.
Thien, L.B., Azuma, H., Kawano, Shoichi, 2000. New perspectives on the pollination biology of basal angiosperms. Int. J. Plant 161.
Vilà, M., Sardans, J., 1999. Plant competition in Mediterranean-type vegetation. J. Veg. Sci. 28, 1–294. http://dx.doi.org/10.2307/3237150.
Wagner, B.J., Davis, D.J., Kher, H.V., 2002. The Production of Several Items in a Single Facility with Linearly Changing Demand Rates 33.
Wang, G.-G., Deb, S., Coelho, L., dos, S., 2015. Elephant herding optimization. In: 2015 3rd International Symposium on Computational and Business Intelligence. (ISCBI), IEEE, pp. 1–5. http://dx.doi.org/10.1109/ISCBI.2015.8.
Wierstra, D., Schaul, T., Peters, J., Schmidhuber, J., 2008. Natural Evolution Strategies. 2008 IEEE Congr. Evol. Comput. IEEE World Congr. Comput. Intell. pp. 3381–3387. https://doi.org/10.1109/CEC.2008.4631255.
Yang, X.-S., 2009. Firefly algorithms for multimodal optimisation. In: 5th Symp. Stoch. Algorithms, Found. Appl.
Yang, X.-S., 2010a. A New Metaheuristic Bat-Inspired Algorithm. Nat. Inspired Coop. Strateg. Optim. 2010. In: Stud. Comput. Intell., Springer, Berlin, pp. 65–74.
Yang, X., 2010b. Nature-Inspired Metaheuristic Algorithms, second ed.
Yang, X.-S., Deb, S., 2009. Cuckoo search via Levy flights. In: World Congr. Nat. Biol. Inspired Comput. pp. 210–214. https://doi.org/10.1109/NABIC.2009.5393690.
Yang, X.-S., Deb, S., 2010. Cuckoo Search via Levy Flights.
Yaralidarani, M., Shahverdi, H., 2016. An improved Ant Colony Optimization (ACO) technique for estimation of flow functions (kr and Pc) from core-flood experiments. J. Nat. Gas Sci. Eng. 33, 624–633. http://dx.doi.org/10.1016/j.jngse.2016.05.067.
Zahiri, A., Azamathulla, H.M., Ghorbani, K., 2014. Prediction of local scour depth downstream of bed sills using soft computing models. In: Computational Intelligence Techniques in Earth and Environmental Sciences. Springer Netherlands, Dordrecht, pp. 197–208. http://dx.doi.org/10.1007/978-94-017-8642-3_11.

Zahiri, A., Dehghani, A.A., Azamathulla, H.M., 2015. Application of gene-expression programming in hydraulic engineering. In: Handbook of Genetic Programming Applications. Springer International Publishing, Cham, pp. 71–97. http://dx.doi.org/10.1007/978-3-319-20883-1_4.

Zhang, W., Korf, R.E., 1995. Artificial Intelligence Performance of linear-space search algorithms * 3702.

Zheng, Y.J., 2015. Water wave optimization: A new nature-inspired metaheuristic. Comput. Oper. Res. 55, 1–11. http://dx.doi.org/10.1016/j.cor.2014.10.008.