Engineering Applications of Artificial Intelligence 72 (2018) 267–293

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

The Social Engineering Optimizer (SEO)

Amir Mohammad Fathollahi-Fard [a,*], Mostafa Hajiaghaei-Keshteli [a], Reza Tavakkoli-Moghaddam [b,c,*]

[a] Department of Industrial Engineering, University of Science and Technology of Mazandaran, Behshahr, Iran
[b] School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran
[c] LCFC, Arts et Métiers Paris Tech, Metz, France

A R T I C L E I N F O

Keywords:
Social Engineering Optimizer (SEO)
Meta-heuristics
Single-solution
Optimization techniques
Engineering applications

A B S T R A C T

Although several meta-heuristics have been developed in the last two decades, most of them are population-based, undergo many steps along with several parameters that make them hard to understand and code. In addition, there are same procedures in recent metaheuristics which make them similar. So, the researchers usually are confused to select a metaheuristic and cannot find any superiority or at least in any algorithms. Because of this, the researchers still use the old algorithms instead of the recent ones. Contrary to previous work, this paper aims to develop a simple, intelligent and new single-solution algorithm that has just four main steps and three simple parameters to tune. Social Engineering Optimizer (SEO) starts with two initial solutions divided into attacker and defender. The attacker obtains the rules of Social Engineering techniques to reach its desired goals. By these simple features, the algorithm does precisely both intensification and diversification phases. The basis of the algorithm depends on how an attacker attacks to a defender by four different associated techniques. Finally, the proposed SEO is applied to solve a set of benchmark functions, important engineering and multi-objective optimization problems. The result shows its superiority in comparison with other well-known and recent meta-heuristics.

1. Introduction

Optimization techniques play a key role in today's researches especially in engineering and operations management problems. These techniques are divided into two groups: mathematical programming and heuristic or meta-heuristic algorithms. The first classification in meta-heuristics divides them into two groups; single-solution instead of population-based algorithms.

Holland (1975) firstly introduced Genetic Algorithm (GA) as a population-based technique inspired by genetics for solving complex and huge real engineering problems. Later, Simulated Annealing (SA), based on an annealing process of metals, was introduced by Kirkpatrick et al. (1983) as a single-solution technique. These two algorithms have been the main ones amongst around 100 developed algorithms until now (Hussain et al., 2018).

Intensification and diversification are two main search phases in meta-heuristics (Golshahi-Roudbaneh et al., 2017). GA does these important phases by two operators: crossover and mutation which act blindly in search space (Hajiaghaei-Keshteli and Fathollahi Fard, 2018). In SA, the intensification phase is done by neighboring whereas the diversification phase is carried by changing the temperature and accepting rule (Fathollahi Fard and Hajiaghaei-Keshteli, 2018a). Additionally, in contrast to the traditional optimization algorithms, metaheuristic intelligent optimization algorithms e.g. Particle Swarm Optimization (PSO), Differential Evolution algorithm (DE), Artificial Bee Colony algorithm (ABC), Harmony Search (HS) and Biogeography-based Optimization (BBO) have been successful in tackling the real-optimization problems e.g. Vehicle Routing Problem (VRP), Single Machine Problem (SMP), Parallel Machine Scheduling (PMS), Fixed Charge Transportation Problem (FCTP) and Supply Chain Network Design (SCND) in recent years (Hussain et al., 2018).

In recent developed algorithms, the authors mostly have tried to find intelligent ways to search the potential areas by considering and balancing between these two phases. Imperialist Competitive Algorithm (ICA) inspired by imperialistic competition carries the diversification by forming different empires and does the exploitation in each empire (Atashpaz-Gargari and Lucas, 2007; Fathollahi Fard et al., 2017). In Keshtel Algorithm (KA) inspired by Keshtel's feeding, the Keshtel's search the potential places carefully by an intelligent way and do

* Corresponding authors.
E-mail addresses: amirfard@mazust.ac.ir (A.M. Fathollahi-Fard), tavakoli@ut.ac.ir (R. Tavakkoli-Moghaddam).

https://doi.org/10.1016/j.engappai.2018.04.009
Received 21 September 2017; Received in revised form 4 April 2018; Accepted 8 April 2018
Available online 8 May 2018
0952-1976/

Table 1
The list of main meta-heuristic algorithms between 1975 till 2017.

Meta-heuristic	Author	Inspired by	Classification
Genetic Algorithm (GA)	(Holland, 1975)	Genetic Evolutionary	Population-based
Scatter Search (SS)	(Glover, 1977)	Evolutionary Computation	Population-based
Simulated Annealing (SA)	(Kirkpatrick et al., 1983)	Annealing	Single-solution
Tabu Search (TS)	(Glover and McMillan, 1986)	Human's brain	Single-solution
Artificial Immune System (AIS)	(Farmer et al., 1986)	Immune system in human's body	Population-based
Variable Neighborhood Search (VNS)	(Mladenović and Hansen, 1997)	Distant neighborhoods	Single-solution
Memetic Algorithm (MA)	(Moscato, 1989)	Genetic Evolutionary	Population-based
Ant Colony Optimization (ACO)	(Dorigo et al., 1996)	Ant's feeding behavior	Population-based
Particle Swarm Optimization (PSO)	(Kennedy and Eberhart, 1995)	Swarm 's feeding behavior	Population-based
Differential Evolution (DE)	(Price and Storn, 1997)	Biological systems	Population-based
Cross Entropy Method (CMS)	(Rubinstein, 1997)	Generic approach	Population-based
Harmony Search (HS)	(Geem et al., 2001)	Composing a piece of music	Population-based
NSGA-II for multi-objective Optimization	(Deb et al., 2002)	Genetic and Pareto Optimization	Population-based
Bees Optimization (BO)	(Nakrani and Tovey, 2004)	Dancing of bees	Population-based
Glow-worm Swarm Optimization (GSO)	(Krishnand and Ghose, 2005)	Swarm behavior of glow-worm	Population-based
Artificial Bee Colony Algorithm (ABC)	(Karaboga, 2005)	Foraging behavior of a honeybee Swarm	Population-based
Honey Bee Mating Optimization (HMO)	(Hadad et al., 2006)	Bee's mating behavior	Population-based
Intelligent Water Drops (IWD)	(Shah-Hosseini, 2011)	Water rules	Population-based
Imperialist Competitive Algorithm (ICA)	(Atashpaz-Gargari and Lucas, 2007)	Imperialistic competition	Population-based
Firefly Algorithm (FA)	(Yang, 2008)	Flashing behavior of firefly	Population-based
Monkey Search (MS)	(Mucherino and Seref, 2007)	Monkey climbing trees looking for food	Population-based
League Championship Algorithm (LCA)	(Husseinzadeh Kashan, 2009)	Sport championship	Population-based
Gravitational Search Algorithm (GSA)	(Rashidi et al., 2009)	laws of gravity and mass interactions	Population-based
Cuckoo Search (CS)	(Yang and Deb, 2009)	Mating behavior of cuckoo	Population-based
Bat Algorithm (BA)	(Yang, 2010)	Echolocation behavior of micro bats	Population-based
Galaxy-based Search Algorithm (GbSA)	(Shah-Hosseini, 2011)	Rules of galaxy	Population-based
Spiral Optimization (SO)	(Tamura and Yasuda, 2011)	Analogy of spiral phenomena	Population-based
Teaching–Learning-Based Optimizer	(Rao et al., 2011)	Effect of influence of a teacher on learners	Population-based
Krill Herd	(Gandomi and Alavi, 2012)	Simulation of herding behavior of krill individual	Population-based
Ray Optimization	(Kaveh and Khayatazad, 2012)	Rays of light refracting their direction changes	Population-based
Differential Search Algorithm (DSA)	(Civicioglu, 2012)	Migration of super organism	Population-based
Keshtel Algorithm (KA)	(Hajiaghaei-Keshteli and Aminnayeri, 2013)	Keshtel's feeding behavior	Population-based
Dolphin echolocation	(Kaveh and Farhoudi, 2013)	Echolocation ability of dolphins	Population-based
Interior Search Algorithm (ISA)	(Gandomi, 2014)	Interior design and decoration	Population-based
Gray Wolf Optimizer (GWO)	(Mirjalili et al., 2014)	Leadership hierarchy and hunting mechanism of gray wolf	Population-based
Chicken Swarm Optimizer (CSO)	(Meng et al., 2014)	Chicken's feeding behavior	Population-based
Exchange Market Algorithm	(Ghorbani and Babaei, 2014)	Procedure of trading the shares on stock market	Population-based
Ant Lion Optimizer (ALO)	(Mirjalili, 2015c)	Hunting mechanism of Ant Lions in nature	Population-based
Optics Inspired Optimization (OIO)	(Husseinzadeh Kashan, 2015)	Optics rules	Population-based
Multi-Verse Optimizer (MVO)	(Mirjalili et al., 2015)	White and black hole and wormhole of multi-verse	Population-based
Elephant Herding Optimization (EHO)	(Wang et al., 2015)	Herding behavior of elephant group	Population-based
Dragonfly Algorithm (DA)	(Mirjalili, 2015c)	Static and dynamic swarming of dragonflies	Population-based
Moth-Flame Optimizer algorithm (MFO)	(Mirjalili, 2015c)	Flying moth in night by maintaining a fixed angle with respect to the moon	Population-based
Water Wave Optimizer (WWO)	(Zheng, 2015)	Phenomena of water	Population-based
Sine Cosine Algorithm (SCA)	(Mirjalili, 2015c)	Sine and cosine function	Population-based
Virus Colony Search (VCS)	(Li et al., 2016)	Diffusion and infection strategy for the host cells adopted by virus to survive	Population-based
Crow Search Algorithm (CSA)	(Askarzadeh, 2016)	Behavior of crows	Population-based
Whale Optimizer Algorithm (WOA)	(Mirjalili and Lewis, 2016)	Bubble-net behavior of social whales	Population-based
Sperm Whale Algorithm (SWA)	(Ebrahimi and Khamehchi, 2016)	Sperm Whale's lifestyle	Population-based
Red Deer Algorithm (RDA)	(Fathollahi Fard and Hajiaghaei-Keshteli, 2016)	Red Deer's mating	Population-based
Grasshopper Optimization Algorithm (GOA)	(Saremi et al., 2017)	Behavior of Swarm grasshopper	Population-based
Spotted Hyena Optimizer (SHO)	(Dhiman and Kumar, 2017)	Social relationship between spotted hyenas	Population-based
Salp Swarm Algorithm (SSA)	(Mirjalili et al., 2017)	Swarm behavior of salp's chain	Population-based

the exploration phase by moving and landing in the lake (Hajiaghaei-Keshteli and Aminnayeri, 2013). Furthermore, in Red Deer Algorithm (RDA) inspired by Red Deer's mating, the authors find a creative way to make a trade-off between the phases by three operators (Fathollahi Fard and Hajiaghaei-Keshteli, 2016; Golmohamadi et al., 2017; Samadi et al., 2018). Besides, in Whale Optimization Algorithm (WOA), the bubble-net hunting strategy of whales probes neighborhoods the best solutions and allows other search agents to explore the potential good areas (Mirjalili and Lewis, 2016; Sadeghi-Moghaddam et al., 2017). So, the knowledge of the search phases in meta-heuristics is useful for developing a new one (Kar, 2016). The list of main meta-heuristics with their characteristics is shown in Table 1. In the following sub-sections, the advantages and disadvantages about the population-based and single-solution meta-heuristics are stated. Consequently, our motivation to develop a new algorithm is clarified in the last sub-section.

1.1. Population-based vs. single-solution algorithms: advantages and disadvantages

As summarized in Table 1, most of the developed meta-heuristics are population-based. In this term, algorithms maintain and improve multiple candidate solutions via different operators. While in single-solution techniques, a solution is applied to find the optimal solution. When a population is used to search the space, the probability of reaching good solutions among candidate solutions is more than single. But, generally, population-based techniques need more time and also most of them have not good or greedy intensification phase in comparison to single-solution algorithms.

Overall, it is evident that the quantity of single-solution methods are less than the population based ones. In the last three decades, researchers have mainly focused on population-based approaches and a few single-solution methods were developed. Single-solution methods are simpler than the population ones. To the best of our knowledge, SA, VNS and TS as strong methods which their performances were probed in many contents, are the only existing single-solution metaheuristics in the literature. In other word, single-solution techniques have some benefits instead of population-ones. These methods are easy to code. Managing the search phases is simple *i.e.* reaching to a tradeoff between intensification and diversification phases are easier and usually better in comparison to population based ones. By this motivation, this paper considers a new plan to generate a single-solution algorithm with novel characteristics.

1.2. Motivation and differences of the proposed meta-heuristic

Generally speaking, all of recent developed algorithms are population-based meta-heuristics. As mentioned earlier, generally, a single-solution algorithm's coding is simpler than the population-based ones. By another point of view, it seems that although most of population-based meta-heuristics are good to make a balance between the phases, they have several steps and many parameters to tune (*e.g.* ICA, FA and WWO etc.) (Sörensen, 2015). Moreover, No Free Lunch theory denotes that none of the mentioned metaheuristics is able to solve all optimization problems (Wolpert and Macready, 1997). Regarding this theorem, there is always possibility that a new algorithm shows superior results in comparison of current metaheuristics based on the current or new optimization problems. Hence, these difficulties motivate our attempt to develop a new simple and efficient single-solution meta-heuristic.

In this paper, by utilizing Social Engineering (SE) phenomenon and its techniques, a new easy and an intelligent algorithm has been developed. This single-solution meta-heuristic starts with two initial random solutions. These two solutions are divided into two types: attacker and defender. The better solution is selected as the attacker. During the procedure of SE, the attacker wants to defeat the defender by SE attacks' techniques.

To cope with the proposed algorithm, the main contributions of this paper can be outlined as follows:

- A new single-solution meta-heuristic approach inspired by Social Engineering is developed.
- The proposed easy and intelligent method has four main steps and only three simple parameters to tune.
- The offered algorithm is applied to solve the twelve famous benchmark functions, common engineering and multi-objective problems.
- The introduced algorithm gives this opportunity to a user to choose the suitable technique among four ones.
- The results confirm the performance and effectiveness of offered method in practice.

The reminder of this paper is organized as follows. Section 2 introduces the Social Engineering and techniques. Section 3 investigates the Social Engineering Optimizer (SEO) and its steps in detail. In Section 4, the experimental studies are explored and performance and efficiency of the proposed method is analyzed by benchmark functions, engineering and also multi-objective problems. Finally, the results and future works are given in Section 5.

2. Social engineering

Social Engineering (SE) is defined as indirect attacks to obtain the people revealing their information by using certain techniques. Attackers want to reach their desired goals or objectives usually by advance technologies (Luo et al., 2013).

Sarah Granger explained the main cycles of Social Engineering in her works (Granger, 2001, 2002). Accordingly, she acknowledged that defense against the Social Engineering attacks has a common pattern such as a mass. In addition, she claimed that the Social Engineering attack procedure has a four step cycle.

In the first phase, the training and retraining from the attacker to a defender is considered. In this regard, the attacker wants to obtain some special information from the defender. This information can be put in different contents. Perhaps, the first questions are about famous special clip video, music, sport and public events that occur in the community and other dimension of personal family systems. Any forwards to start and conduct a SE attack needs a series of such data to identify the person's reactions to design an attack on him/her. The idea of SE experts is explained that all social media has expended its range to all ages for trying your aim and bringing the community in the form of their intended route, although they pretend that they are going to help them (Weinberg, 1966).

In the second phase, the attacker spots a SE attack. It is obvious that before carrying out an attack, on the point where the probability of success is more than other points, it must be identified. The attacker takes you (defender) to a position that he/she prefers the defender to be placed in the way of his/her goals. The most important thing to prevent an attack on his/her main demand is that defender can understand and think like an attacker. According to past learning to spot a Social Engineering attack the dependent techniques are different. Such as: pretext placement, obtaining, diversion theft, phishing and so on. In each technique, aggressive manner is so different and its profit is variable, too.

The next phase is how to respond this SE attack, reaction of the defender and how much information it wishes to make the striker. In SE attacks, the most important phase is how to respond to an attack. The previous experiences and his/her knowledge about this kind of attack may be helped him/her. At the least, in final step, the attacker try to steal each thing that is may be useful and strike to defender. Or if an attack is not achieved satisfactory results or otherwise offensive repeats or stopped by the person and someone else chooses.

3. Social Engineering Optimizer (SEO)

In this algorithm, each solution is a counterpart to a person and traits of each person such as his/her ability in contents of mathematical, sport, business are the counterparts of all variables of each solution in search space. To start the algorithm, the two random solutions are initialized and better solution is selected as attacker and the other is named as the defender. To simulate training and retraining of the attacker from the defender, some random experiments for each trait of the defender are designed. The attacker tries to evaluate the defender by his/her traits. The counterpart of training and retraining in search space is copying a trait from the attacker to the same trait in defender and computing the rate of retraining of the attacker from the defender, simultaneously. In the next step, spotting a SE attack from the attacker to the defender is the counterpart of changing the position of defender by an intelligent way in feasible space. In responding a SE attack, the fitness of new position of the defender is calculated and the old and current positions of the defender are compared and the best position is selected. If the ability of the defender is better than the attacker, positions of the attacker and the defender are changed. At the end, to strike to the defender, it is annihilated and replaced by a new random solution in search space. To ease the steps of SEO at the first glance, Fig. 1 shows the flowchart of proposed method.

Like other meta-heuristics, controlling the two main phases is considered for the proposed algorithm (Črepinšek et al., 2013). Local search or exploitation phase is done by training and retraining between the attacker and the defender. Also, spotting an attack and responding to it, does the intensification action. Indeed, the exploration phase is performed by annihilating the defender and creating a new one.

Due to the novelty of SE, there is no available mathematical formulation to employ in this study. Therefore, this work by considering the well-known and recent metaheuristics *e.g.* PSO, ICA and RDA proposes a set of formulations to consider the main properties of SE as follows.

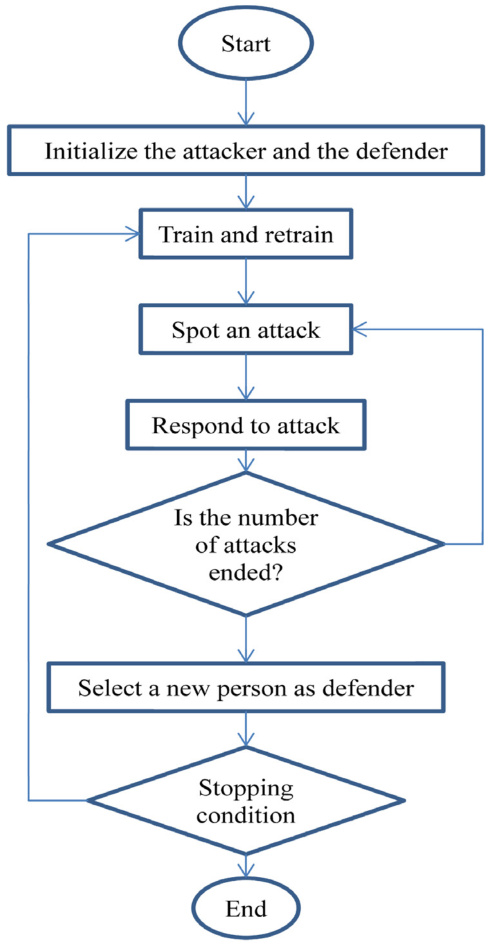


Fig. 1. The flowchart of proposed SEO.

3.1. Initialize the attacker and the defender

The goal of Optimizer is to find an optimal solution among all feasible solutions. In this way, an array of variable values to be optimized is designed. For instance in GA, the terminology of array is named "*chromosome*", but here in SEO, this term, "*person*" is used for this array. Therefore, the person is counterpart of the solution. In addition, in GA, "*gene*" is used for each variable of an array. But in SEO, "*trait*" is defined on the person for each variable. This is included in different contents of mathematical, sport, business and so on. In an N_{var}-dimensional Optimization problem, a person is a $1\times N_{\text{var}}$ array. This array is defined by:

$$person = [X_1, X_2, X_3, \ldots, X_{N_{var}}]. \tag{1}$$

Also the value of Objective Function (OF) is evaluated for two persons as follows:

$$Value = f(person) = f(X_1, X_2, X_3, \ldots, X_{N_{var}}). \tag{2}$$

The algorithm starts with two random solutions. The better solution is selected as the attacker and the other is named as the defender.

3.2. Train and retrain

This step aims to show the training and retraining the attacker from the defender. In this way, the attacker tries to test on each trait of the defender to recognize the most efficient trait. In this regard, $\propto$ percent of attacker traits are selected randomly and are replaced in the same traits of the defender as follows:

$$N_{Train} = round\{\propto .nVar\} \tag{3}$$

where $\propto$ is the percent of selected traits and $nVar$ is the number of all traits in a person. Accordingly, N_{Train} is the number of traits that will be tested on the some random traits of the defender.

To explain more, an example is displayed to clarify this operator in Fig. 2. Four traits are considered for each person. The red box is the symbol of the attacker and the green box is considered as the defender. In this instance, all of traits are between zero and one. The rate of $\propto$ is equal to 0.5 in this example. In this regard, the defender by retaining rate of 10 is replaced by the current defender.

3.3. Spot an attack

To spot an attack, we consider four different techniques involving: obtaining, phishing, diversion theft and pretext. These mentioned techniques are used in a random way to search the feasible space. In addition, they utilize just only one parameter named β as an input variable of search engine. The red circle is specified the attacker and the green for the defender. Also the blue circle is shown the new defender. In addition, the new position during the procedure of technique is shown by arrows. In all equations, def_{new} is shown the new position of the defender during attack and def_{old} and att are the current position of the defender and the attacker. The algorithm is developed in a way that the user can employ one operator among the four ones.

3.3.1. Obtaining

In this technique, the attacker abuses the defender directly as the guidance to obtain the desired goals. Fig. 3 is also demonstrated it as a visual simulation used in the fact. In this regard, one new solution is generated. To create the new position, this equation is proposed:

$$def_{new} = def_{old} \times (1 - \sin\beta \times U(0,1)) + \frac{(def_{old} + att)}{2} \times \sin\beta \times U(0,1) \tag{4}$$

Eq. (4) aims to illustrate the movement of defender by considering its current position and the average of distance between the attacker and the defender. This movement is based on random distribution *i.e. U(0, 1)* and the amount of β as the rate of spotting an attack.

3.3.2. Phishing

To design this technique, the attacker pretends to approach the defender and then the defender moves to a place that the attacker wants to be there. This proposed technique generates two new solutions as

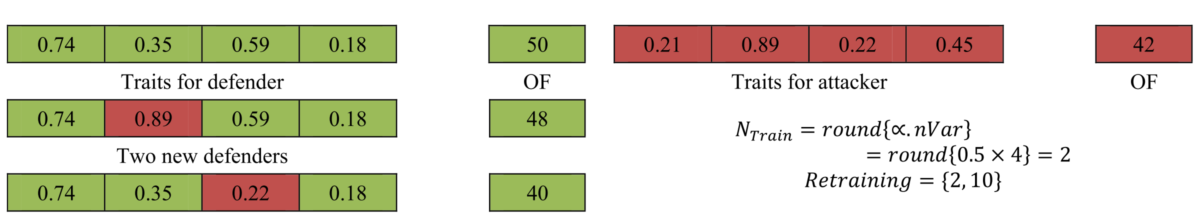


Fig. 2. A simple example of training and retraining process. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

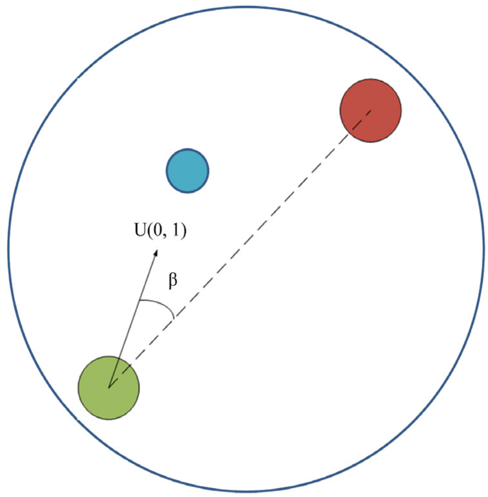


Fig. 3. The proposed obtaining technique (technique 1).

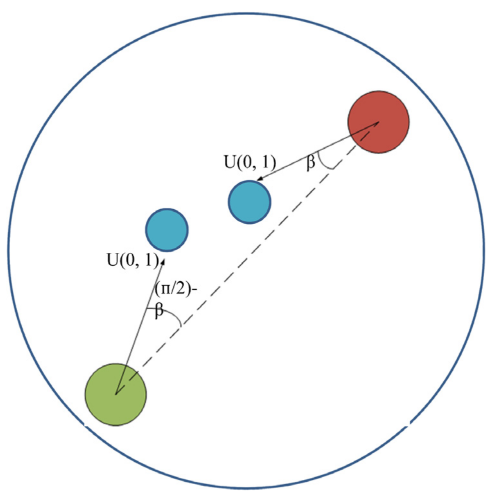


Fig. 4. The proposed phishing technique (technique 2).

shown in Fig. 4. The equations are displayed as follows:

$$def^{1}_{new} = att \times (1 - \sin\beta \times U(0,1)) + \frac{(def_{old} + att)}{2} \times \sin\beta \times U(0,1) \tag{5}$$

$$def^{2}_{new} = def_{old} \times \left(1 - \sin\left(\frac{\pi}{2} - \beta\right) \times U\,(0,1)\right) + \frac{(def_{old} + att)}{2} \times \sin\left(\frac{\pi}{2} - \beta\right) \times U(0,1) \tag{6}$$

Eqs. (5) and (6) illustrate two new defenders by a movement based on the attacker and the defender, respectively. In both of them, the average of distance between the attacker and the defender is the main sector of movement. Furthermore, the current position of attacker in Eq. (5) and also the current position of defender in Eq. (6) are the main drivers to move the defender in this attack. Like previous spots, the movement is based on a random distribution *i.e. U(0, 1)* and the amount of β to achieve this goal.

3.3.3. Diversion theft

In this technique, at first the attacker guides the defender to a position, that in fact, this is a deception for the defender. To depict this process, Fig. 5 shows this proposed technique. In this process, just one solution is created. The following equation is shown the formulation of

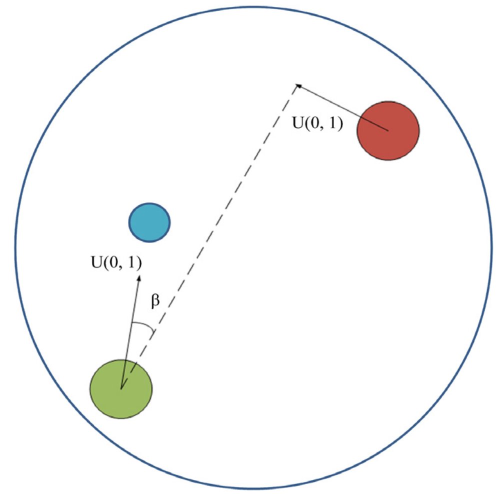


Fig. 5. The proposed diversion theft technique (technique 3).

this action:

$$def_{new} = def_{old} \times (1 - \sin\beta \times U(0,1)) + \frac{(def_{old} + att \times U(0,1) \times \sin(\frac{\pi}{2} - \beta))}{2} \times \sin\beta \times U(0,1) \tag{7}$$

Eq. (7) aims to present the movement of defender by considering its current position and the average of distance between the defender and a weighted amount of attacker to achieve the goal. The uniform distribution *U(0, 1)* and the amount of β play the main drivers of this movement of defender in this spotting attack.

3.3.4. Pretext

During this technique, the attacker baits some traits which are a favor of the defender and guides the defender. This technique is useful to defeat the defender. Fig. 6 displays this proposed technique. In this process one solution is generated and created during this process by following equation:

$$def_{new} = \left(def_{old} \times U(0,1) \times \sin\left(\frac{\pi}{2} - \beta\right)\right) \times (1 - \sin\beta \times U(0,1)) + \frac{((def_{old} \times U(0,1) \times \sin\left(\frac{\pi}{2} - \beta\right)) + att)}{2} \times \sin\beta \times U(0,1) \tag{8}$$

Eq. (8) includes two main terms. The first one is a weighted amount of current position of defender. The second term is the average of distance between the weighted defender and the attacker. Whole of this distance is also weighted by the rate of spotting the attack and the uniform distribution between zero and 1.

3.4. Respond to attack

New position of the defender is evaluated and compared with the old position of the defender. Then, the best position for the defender is selected and if the new position of the defender is better than attacker, the defender and attacker are exchanged as shown in Fig. 7.

3.5. Create a new person as defender

Here, the attacker annihilates the defender and creates a new person randomly to perform SE rules.

3.6. Stopping condition

Like other meta-heuristics, the stop condition may be maximum time of the simulation or the quality of the best solution ever found or other selected condition by user. Fig. 8 displays the pseudo-code of proposed algorithm.

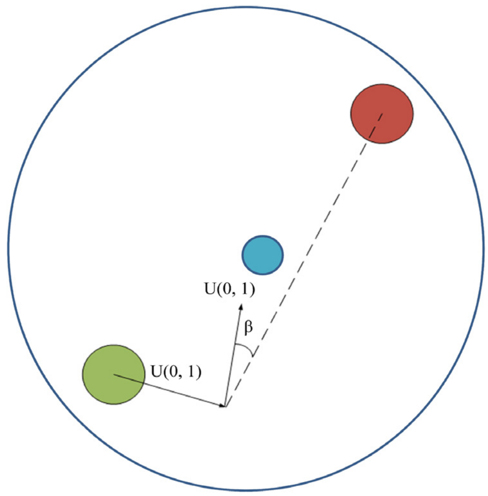


Fig. 6. The proposed pretext technique (technique 4).

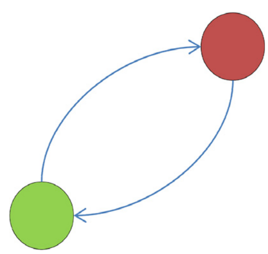


Fig. 7. The exchanging the defender and the attacker.

```
T1=clock;
Initialize attacker and defender
It=1;
while solving_time < Max_time
      Do training and retraining;
      Num_attack=1;
      while Num_attack < Max_attack
              Spot an attack;
              Check the boundary;
              Respond to attack;
              if the OF of defender is lower than attacker
                Exchange the defender and attacker position;
              endif
              Num_attack= Num_attack+1;
      endwhile
      Create a new solution as defender;
      It=It+1;
      T2=clock;
      Solving_time=T2- T1;
endwhile
Return attacker.
```

Fig. 8. The pseudo-code of proposed SEO.

4. Experimental studies

As mentioned earlier, the past decade has seen the rapid development of meta-heuristics with their applications in the engineering problems. For example, Hajiaghaei-Keshteli and Aminnayeri (2014) firstly used the Keshtel Algorithm (KA) in an integrated scheduling of production and rail transportation problem. Also, Nazeri-Shirkouhi et al. (2010) used the Imperialist Competitive Algorithm (ICA) to solve an integrated product mix-outsourcing problem for the first time, as well. On the other hand, it is formal to study the performance of recent algorithms with some benchmark functions. After studying the related and recent papers, we plan to use some well-known problems in various research areas to probe the behavior of SEO in different situations and to exhibit the performance and effectiveness of this new optimization algorithm.

First of all, the SEO is evaluated with some famous benchmark problems like other meta-heuristic papers. Consequently, in the second section, this study is focused on some interesting engineering problems. Then, these problems are solved with SEO and compared the results with other well-known meta-heuristic algorithms as well. In addition, the proposed algorithm is utilized to solve the multi-objective optimization problems. Accordingly, a new bi-objective model for a supply chain network is presented. The last two subsections are quite new in experimental studies in this research area for a meta-heuristic paper.

4.1. Benchmark functions

To evaluate a novel metaheuristic, it is always needed to use some benchmark functions. The literature reports that there are more than fifty mathematical functions to achieve this goal (Baykasoğlu and Akpinar, 2015a, b). This study considers twelve functions benchmarked from Ghorbani and Babaei (2014). Related researchers have studied on these functions to show the performance of novel meta-heuristics (Karaboga and Akay, 2009; He et al., 2009; Rao and Patel, 2013; Ghorbani and Babaei, 2014). In addition, all of them are adopted from CEC2005 (Liang and Suganthan, 2005) and CEC2010 (Zhao et al., 2010), too. These problems are numbered as P1 to P12 and all of them are minimization problems. In each standard function, the optimum value is equal to zero. The main reason behind of choosing these mathematical functions is that each function has some special properties to ease the assessment of metaheuristics' performance. In regards to explore the capabilities of the proposed algorithm, it is tried to evaluate different cases of the standard functions characteristics. As it is obvious, P2, P3, P4 and P5 are unimodal and non-separable where P8, P10, P11 and P12 functions have multimodal and non-separable characteristics. In addition, P5, P1 and P7 are unimodal and separable. Also, P9 is multimodal and separable. Since these functions are well-known and were used in several similar studies, the details about them are given in Supplementary Materials F1.

Notably, the proposed SEO is divided into four versions relay on four developed techniques in designing an attack. We also employ some well-known meta-heuristics such as: GA as well-known traditional method in the term of evolutionary algorithms; SA as the most relevant technique in the single-solution algorithms; PSO as one of the best technique in the last two decades; a set of successful recent developed metaheuristics i.e. ABC, ICA, FA and RDA (Golmohamadi et al., 2017; Samadi et al., 2018; Hajiaghaei-Keshteli and Fathollahi Fard, 2018) and also a state of art Optimizer called Linear-Success-History based Adaptive of Differential Evolution (L-SHADE) as an improved version of DE (Tanabe and Fukunaga, 2014); are utilized to compare with the four versions of SEO proposed.

Additionally, the comparisons during the benchmark functions are divided into three sections. Accordingly, this study not only compares the proposed algorithm in four versions with considering the maximum number of iterations (equal number of fitness evaluations) like several similar studies in this area but also proposes equal computational time for all metaheuristics to assume a fair environment for competition. The last but not the least is that several sensitivity analyses on the proposed algorithm have been performed to investigate the performance of algorithm. Finally, we evaluate the performance of proposed SEO with engineering applications.

4.1.1. The comparison based on equal number of iterations

In this comparison, first of all, the metaheuristics have been tuned. The values of SEO in four versions are set as follows:

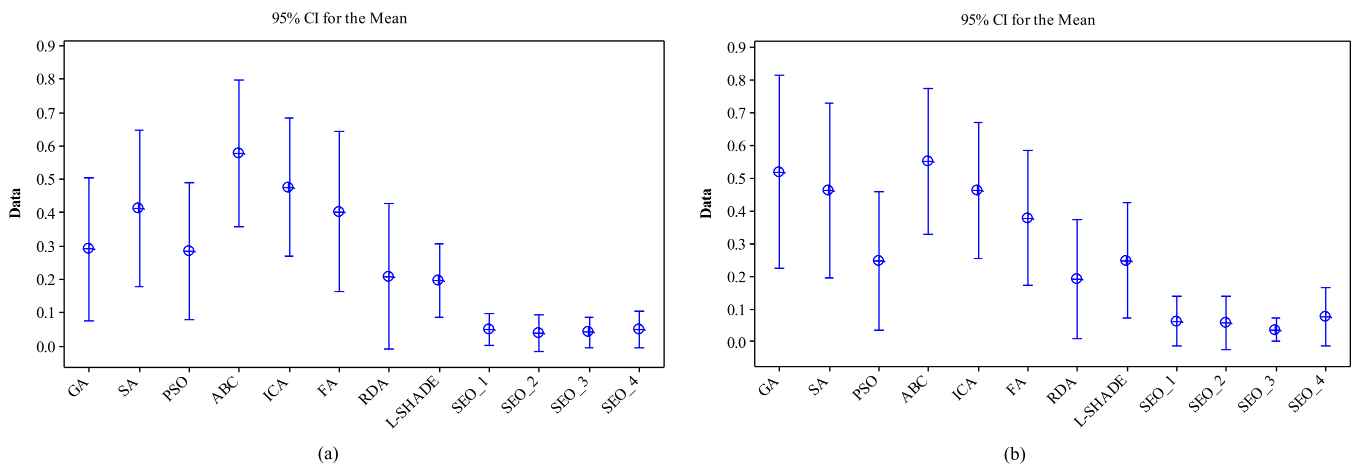


Fig. 9. The Means plot and LSD intervals for the algorithms in equal number of fitness evaluations for low (a) and high (b) dimensional.

(i) SEO_1: Rate of training = 0.2; Rate of spotting an attack = 0.25; Number of attacks = 50;
(ii) SEO_2: Rate of training = 0.2; Rate of spotting an attack = 0.50; Number of attacks = 50;
(iii) SEO_3: Rate of training = 0.2; Rate of spotting an attack = 0.05; Number of attacks = 50;
(iv) SEO_4: Rate of training = 0.2; Rate of spotting an attack = 0.05; Number of attacks = 50;

The tuned values of other algorithms' parameters have been reported in Supplementary Materials F2. Then, the algorithms have been run for 30 times and the best (B), the worst (W) and the average (M) as well as the standard deviation (SD) are noted and given in Table 2. For each benchmarked function, the low dimension ($D = 30$) and high dimension ($D = 100$) are considered. Note that this comparison is based on equal number of fitness evaluations. In this regard, the maximum number of iteration for all population-based algorithms (*i.e*. GA, PSO, ABC, FA, ICA, RDA and L-SHADE) is equal to 100. In addition, for the single solution algorithms (*i.e*. SEO and SA), the stopping condition is 1000 number of iterations. In regards to Table 2 for each test problem in the low and high dimensions, the ranks (R) of algorithms have been numbered. From the table, it is evident that the proposed versions of SEO have the best ranks. In most of test problem except P10 and P12, SEO shows the best values. As a result, the proposed SEO_2 is the best in total ranking by 2.32 in average of rank as shown in Table 2.

Furthermore, the statistical analyses have been done to highlight the performance of algorithm. Accordingly the means plots and Least Significant Difference (LSD) for the presented algorithms in low and high dimensional, separately, have been provided as seen in Fig. 9 The behavior of algorithms for both low (Fig. 9(a)) and high (Fig. 9(b)) are the same. All statistical results demonstrate that our proposed SEO_2 not only better than other similar versions but also has strong edge over the other algorithms.

4.1.2. The comparison based on equal solution time

In this comparison, the stopping condition is time interval and it is equal to 10 s in each problem. According to Mernik et al. (2015) and Draa (2015), the comparison based on equal number of iterations is not fair. Because the computational time of each algorithm in the same iteration is not the same, hence, the equal time is specified to probe the stop condition. According to the best of our knowledge (Mirjalili et al., 2017), there are not any results on the related papers based on equal time in a fair environment. So, this study firstly compares the proposed algorithm with other new and well-known algorithms based on equal time in a fair competition. So, first of all, all algorithms should be tuned. Due to the page limitation, the tuned values have been reported in Supplementary Materials F2. Similar to the previous sub-section the algorithms have been run for thirty times. The best worst and average of them and also standard deviation, are saved As can be envisaged, these outputs have been presented in Table 3.

Regarding the previous sub-section, the algorithms' comparison for both low and high dimensional functions are the same. Here, the analyses have been done for low dimensional functions ($D = 30$). From Table 3, the proposed four versions of SEO reach the best ranks. The average of rank is lower than four for the algorithms. The results demonstrate that SEO_1 is better than the other algorithms, slightly. Whereas in the previous comparison based on equal fitness evaluations SEO_2 shows the best performance. Furthermore, the convergence of algorithms has been analyzed by Fig. 10. As can be seen, the proposed four versions of SEO reveal a performance convergence in comparison of other algorithms. Moreover, to find out the best algorithm decisively, this study conducts a set of statistical comparisons among metaheuristics. The means plots and Least Significant Difference (LSD) for the presented algorithms have been provided as seen in Fig. 11. From this figure, there is a clear difference between the performances of algorithm. Similar to previous comparison, RDA has a neck and neck comparison with the employed versions of SEO. However, SEO_1 and SEO_3 show a better robustness of algorithms in comparison with other presented metaheuristics.

4.1.3. Sensitivity analyses

Notably, one of the advantages of the proposed method is the ability to choose the type of search engine. Hence, a user can select the most suitable technique for his/her problems. To investigate the convergence of the algorithm, a problem P9 is selected. The algorithm presented in this research needs only three parameters to tune. Besides, to set the techniques, just one important parameter is used. Furthermore, some experiments relied on the variable of β in the range of $(0, \frac{\pi}{2} \cong 1.57)$ are designed. The behavior of the algorithm in each technique is identified as depicted in Fig. 12. In this regard, twelve different cases are specified. These cases are numbered as C1 to C12. These cases undergo sensitivity analyses of the key-parameter in order to control the phases for each technique. These analyses show the importance of tuning for the key-parameter, like related studies (see: Liu et al., 2013). In addition, in all cases the rate of training is equal to 0.2 and the number of attacks is equal to 50, too. Besides, the time of simulation is considered on 10 s in all techniques. Also, each technique is run in all cases for thirty times. Moreover, to highlight the performance of four versions of SEO, the outputs of algorithms in very high dimensional between 200 and 1000 is evaluated. Table 4 shows the results. The used notations in the table are similar to previous comparisons during two last sub-sections. Generally, the results not only confirm the availability of all versions of SEO in very high dimensional but also reveal that SEO_2 is better than other

Table 2
The final outputs of algorithms in benchmark functions during thirty run times (B = best, W = worst, M = mean, SD = standard deviation, D = dimension, R = rank).

Function	D		GA	SA	PSO	ABC	ICA	FA	RDA	L-SHADE	SEO_1	SEO_2	SEO_3	SEO_4
P1	30	W	9.58E−02	2.78E−01	6.37E−03	8.37E−01	1.37E−02	4.93E−04	3.52E−04	1.92E−02	4.18E−07	6.48E−09	3.51E−8	**1.58E−09**
		M	8.16E−07	2.15E−02	5.78E−05	2.51E−02	5.36E−03	6.91E−05	1.73E−06	1.17E−07	2.59E−08	2.18E−13	4.26E−11	**5.37E−15**
		B	2.19E−08	4.28E−03	1.89E−06	3.16E−04	2.51E−04	3.78E−08	2.86E−11	5.48E−09	1.23E−12	**0**	7.15E−16	**0**
		SD	0.004783	0.050924	0.000972	0.46176	0.393635	0.000534	0.000136	1.20E−03	0.000047	0.000002	0.000007	**0.000001**
		R	7	12	10	11	9	8	5	6	4	2	3	**1**
	100	W	1.22	4.78	3.65E−02	2.42	5.84	2.76E−01	4.28E−03	2.39E+00	6.81E−06	**8.54E−07**	2.76E−05	5.38E−04
		M	4.67E−03	0.8514	8.35E−01	0.7534	4.32	5.47E−04	5.95E−04	1.22E−01	4.26E−07	**2.16E−09**	1.85E−06	1.85E−06
		B	6.18E−04	0.0251	4.28E−03	0.4627	2.17	2.81E−05	1.53E−05	8.37E−03	5.87E−09	**5.27E−10**	2.51E−07	7.29E−07
		SD	0.01854	1.0894	0.06854	1.0325	3.2817	0.05894	0.07634	2.18E−01	0.000007	**0.000005**	0.000056	0.000084
		R	7	10	9	11	12	6	5	8	2	**1**	3	4
P2	30	W	2.57E−03	5.16E−03	5.84E−06	1.25E−04	1.87E−04	2.35E−04	2.64E−04	6.45E−04	3.19E−05	**2.63E−06**	3.12E−04	5.84E−05
		M	6.82E−07	5.32E−05	5.73E−09	2.68E−05	2.17E−06	3.18E−07	3.85E−08	8.87E−06	2.87E−09	3.27E−10	6.27E−09	**1.58E−10**
		B	2.18E−09	3.11E−06	4.35E−10	3.19E−06	3.12E−07	2.57E−08	1.25E−11	4.44E−07	1.25E−11	**2.75E−14**	2.86E−12	1.07E−12
		SD	3.17E−02	4.28E−02	3.47E−04	2.64E−03	8.53E−03	4.81E−03	5.73E−04	1.43E−02	3.17E−04	5.28E−05	**2.18E−05**	3.84E−05
		R	7	11	6	12	10	8	5	9	4	**1**	3	2
	100	W	2.81E−01	5.18E−02	**5.38E−04**	3.18E−02	5.15E−02	1.75E−02	3.27E−01	1.73E−02	2.55E−02	3.18E−02	1.56E−03	2.55E−03
		M	2.57E−05	7.48E−03	**8.13E−07**	7.29E−03	3.59E−04	3.21E−04	4.38E−06	2.49E−03	5.48E−05	2.15E−06	3.28E−06	7.13E−05
		B	6.36E−07	1.94E−04	5.42E−08	5.42E−05	2.88E−05	2.72E−06	5.74E−08	2.77E−05	5.17E−07	**2.19E−08**	3.65E−07	8.12E−07
		SD	5.28E−02	6.53E−01	1.82E−03	4.18E−01	3.15E−02	3.22E−03	3.16E−03	1.31E−01	8.41E−03	1.52E−03	**2.59E−04**	2.64E−03
		R	6	12	2	11	9	8	3	10	5	**1**	4	7
P3	30	W	2.16E−02	4.17E−04	2.54E−06	3.17E−04	2.16E−03	3.15E−02	5.93E−03	7.93E−05	2.61E−05	**1.48E−07**	2.74E−05	2.18E−05
		M	3.63E−09	2.64E−10	3.72E−07	2.62E−05	3.71E−06	4.71E−06	8.15E−07	4.37E−06	4.61E−10	**2.12E−14**	3.65E−12	6.72E−12
		B	1.22E−11	3.14E−12	5.01E−10	4.83E−08	4.85E−08	1.85E−09	3.15E−12	9.66E−09	2.54E−25	**0**	1.93E−22	1.82E−21
		SD	7.52E−03	8.53E−03	2.17E−02	5.77E−03	3.15E−03	2.71E−02	8.52E−03	1.44E−03	4.27E−04	**6.81E−06**	1.77E−05	8.52E−04
		R	7	5	8	11	12	10	6	9	2	**1**	3	4
	100	W	2.45E−01	2.51E−03	7.18E−05	6.43E−02	4.36E−02	2.58E−01	4.26E−02	9.19E−03	4.63E−03	**2.18E−05**	2.67E−03	1.65E−04
		M	3.81E−08	2.78E−09	3.18E−06	3.17E−04	1.27E−04	2.83E−05	4.27E−06	1.06E−04	8.53E−08	3.19E−09	**2.17E−9**	5.14E−09
		B	2.51E−09	3.22E−10	5.17E−09	1.68E−06	3.19E−06	7.12E−07	1.29E−09	2.40E−07	7.28E−14	**2.17E−15**	2.65E−12	5.86E−13
		SD	4.87E−01	2.56E−02	2.15E−04	1.25E−01	1.78E−01	2.16E−01	2.81E−02	1.56E−02	2.18E−03	**1.28E−04**	3.81E−02	5.87E−03
		R	7	5	8	11	12	9	6	10	2	**1**	4	3
P4	30	W	78.93	8.18	22.91	9.54	6.19	7.52	5.01	**1.19**	3.28	2.81	5.92	1.42
		M	10.53	1.2794	5.73	0.8659	0.8134	0.5894	4.82E−04	1.08E−01	3.91E−03	5.71E−03	5.18E−05	**2.75E−06**
		B	3.8217	2.81E−02	3.91E−02	1.29E−02	5.28E−03	2.71E−05	2.77E−06	3.23E−03	5.18E−08	2.18E−04	**3.84E−09**	1.25E−08
		SD	9.4521	3.62	8.61	4.28	3.994	2.71E−00	3.82E−01	1.43E+00	0.7843	0.8754	**0.2785**	0.5645
		R	12	11	10	8	7	5	4	8	3	6	**1**	2
	100	W	95.62	10.64	34.17	15.85	12.95	11.37	7.83	3.19E+01	6.83	8.93	5.81	**4.97**
		M	11.84	4.26	7.95	3.1854	5.8728	0.8623	3.61E−02	5.92	3.81E−02	5.12E−01	4.73E−02	**1.86E−04**
		B	6.842	3.81E−01	2.18	1.6809	0.7854	3.72E−03	5.87E−05	1.14	1.25E−06	2.61E−03	3.46E−03	**8.25E−07**
		SD	24.81	3.72	2.0823	1.7905	2.6847	3.78	1.9923	8.27	0.8645	0.9387	**0.3623**	0.3783
		R	12	7	11	10	8	6	3	9	2	4	5	**1**
P5	30	W	98.47	95.27	99.15	114.46	119.76	102.54	87.54	**39.9**	75.18	68.52	82.17	84.94
		M	28.65	25.94	44.75	98.72	91.45	82.18	29.58	45.7	17.9328	**16.475**	19.58	18.6271
		B	19.75	18.75	32.15	30.99	31.58	30.85	24.61	15.8	15.29	**14.28**	15.4763	15.4763
		SD	44.87	30.98	486.74	1845.62	6254.91	1392.57	196.74	2.08E+03	**28.97**	48.73	38.761	40.632
		R	7	6	11	10	12	9	8	5	2	**1**	3	4
	100	W	123.25	105.89	114.95	156.75	142.68	116.75	95.47	**57.5**	89.25	91.27	88.43	91.27
		M	32.67	29.81	52.68	115.63	99.74	86.53	31.47	**17.6**	29.88	28.65	32.47	30.27
		B	20.64	19.41	36.27	32.16	34.82	33.81	26.91	**15.1**	24.86	25.651	27.91	25.48
		SD	56.14	32.8162	549.62	1974.25	754.38	1473.89	215.48	215.3	**30.81**	58.93	39.78	41.58
		R	3	2	12	9	11	10	8	**1**	4	6	7	5

(*continued on next page*)

Table 2 (*continued*)

Function	D		GA	SA	PSO	ABC	ICA	FA	RDA	L-SHADE	SEO_1	SEO_2	SEO_3	SEO_4
P6	30	W	3	3	1	3	7	0	0	0	0	0	0	0
		M	0.0008	1.89E−03	8.24E−18	2.35E−09	2.57E−01	0	0	0	0	0	0	0
		B	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
		SD	0	6.045955	7.348469	62.7283	18.54724	5.95219	1.06066	**0**	**0**	0.353553	0.353553	**0**
		R	7	9	10	12	11	8	6	3	**1**	4	5	2
	100	W	24	21	57	87	23	13	21	1.90E+01	**9**	13	11	**9**
		M	2.65E−02	1.75E−01	2.64E−04	2.19E−05	2.16E−00	1.75E−15	3.72E−10	8.80E−05	2.85E−16	3.72E−19	**5.13E−20**	5.82E−19
		B	1	1	1	1	3	**0**	1	**0**	**0**	**0**	**0**	**0**
		SD	1.65E−01	2.85E−02	1.87E−01	3.28E−02	1.57E−01	3.82E−02	3.16E−02	6.23E−02	5.28E−04	**3.19E−04**	5.14E−03	5.19E−04
		R	11	8	12	9	10	5	7	6	2	**1**	3	4
P7	30	W	13.87	14.56	13.87	13.69	12.54	9.76	9.81	11.75	8.76	9.15	**9.25**	9.54
		M	7.89	10.25	10.57	10.86	10.25	8.46	8.26	9.86	7.56	7.15	**6.82**	6.91
		B	6.54	7.81	7.89	6.47	8.92	7.15	6.38	7.65	6.84	5.16	**4.92**	5.84
		SD	1.584	2.71	1.47	1.5843	1.3672	1.91	2.685	1.36	0.98	**0.47**	0.86	0.75
		R	6	10	11	5	12	8	4	9	7	2	**1**	3
	100	W	18.95	17.24	15.49	15.86	14.74	12.85	10.86	12.85	11.54	**10.32**	10.58	10.39
		M	8.918	11.1288	11.92	11.63	10.54	9.86	9.24	11.76	9.54	8.99	**8.59**	9.64
		B	8.25	8.25	9.63	8.25	8.18	8.82	8.10	10.22	7.86	7.36	**7.12**	7.84
		SD	1.67	2.86	1.75	1.79	1.64	2.56	2.89	3.11	1.067	**0.58**	0.88	0.89
		R	7	8	11	9	6	10	5	12	4	2	**1**	3
P8	30	W	1.86E−04	1.47E−02	1.68E−03	5.42E−02	1.67E−02	2.72E−04	3.81E−03	4.90E−03	2.84E−05	**3.71E−06**	4.16E−05	2.85E−05
		M	3.72E−05	1.46E−04	5.37E−06	1.48E−04	5.78E−04	6.24E−07	5.87E−05	7.30E−05	2.67E−09	**1.36E−13**	1.72E−12	2.84E−11
		B	2.76E−07	5.81E−06	2.14E−08	3.27E−09	3.11E−08	6.82E−11	5.46E−09	2.91E−06	2.81E−16	**0**	3.82E−14	2.85E−12
		SD	2.81E−03	8.26E−02	3.81E−02	2.64E−02	4.18E−02	1.82E−03	7.53E−04	4.13E−02	5.86E−04	**7.86E−05**	2.45E−04	2.88E−04
		R	10	12	8	7	9	5	6	11	2	**1**	3	4
	100	W	1.53E−02	8.53E−01	2.84E−02	4.17E−01	5.83E−01	8.42E−03	5.17E−02	2.92E−01	3.76E−03	5.66E−04	**2.94E−04**	3.18E−04
		M	1.23E−04	3.32E−06	6.54E−06	1.54E−02	4.83E−02	1.28E−04	4.37E−07	2.42E−02	1.84E−07	**1.28E−10**	4.64E−08	3.95E−08
		B	5.92E−04	1.48e−08	4.79E−07	5.38E−05	5.92E−04	6.19E−05	8.29E−08	2.96E−04	8.29E−9	**8.29E−14**	8.29E−11	8.29E−10
		SD	0.000564	1.03E−05	0.000109	0.284115	0.088928	0.036815	0.000181	2.96E−02	8.7E−05	**4.32E−06**	3.04E−05	0.000129
		R	11	5	7	8	12	9	6	10	4	**1**	2	3
P9	30	W	43.82	42.81	39.81	38.29	43.81	35.92	32.18	**19.1**	24.81	22.67	25.74	24.81
		M	18.36	30.92	22.59	28.11	17.85	29.32	16.931	14.1	10.44	**3.96**	6.29	7.59
		B	17.4938	17.4938	12.4821	22.5918	16.3917	16.3917	8.5474	7.51	4.47	**1.53**	2.933	2.5933
		SD	9.461557	4.940765	12.68426	10.67775	0.182485	25.73	20.563	3.56	4.55	**1.55**	5.022	7.978
		R	11	10	7	12	8	9	6	5	4	**1**	2	3
	100	W	52.86	59.15	48.72	51.23	52.19	43.64	42.58	**29.6**	**31.86**	32.86	33.81	32.54
		M	28.71	34.82	27.91	30.82	24.71	32.81	19.76	17.4	18.65	**9.86**	14.56	12.84
		B	16.83	20.73	18.91	18.16	12.76	14.83	11.56	10.4	10.52	9.23	8.97	**8.75**
		SD	8.745	9.15	7.84	6.32	8.182	7.93	6.15	3.05	5.47	5.82	**4.03**	5.83
		R	9	12	11	10	7	8	6	4	5	3	2	**1**
P10	30	W	3.82E−02	0.8126	6.81E−01	2.71	1.28	3.88E−03	**8.53E−04**	9.03E−01	3.71E−02	6.22E−02	8.16E−02	4.72E−02
		M	6.32E−04	1.86E−02	4.82E−02	7.81E−02	4.82E−02	5.82E−05	**6.26E−07**	3.91E−02	8.52E−04	8.22E−02	1.73E−02	4.21E−02
		B	1.54E−06	3.66E−04	5.77E−04	2.97E−02	8.31E−02	5.88E−08	**1.43E−09**	9.90E−03	2.75E−06	1.86E−06	2.32E−06	5.96E−06
		SD	2.78E−02	1.66E−01	3.22E−02	4.37E−01	2.36E−01	6.82E−02	3.82E−03	1.46E−01	2.81E−03	**1.84E−03**	4.72E−03	8.21E−03
		R	2	8	9	12	11	6	**1**	10	5	3	4	7
	100	W	2.867	7.84	2.78	6.34	3.91	0.684	0.523	2.61E+00	0.9634	0.8734	**0.2716**	0.7532
		M	2.91E−02	3.82E−01	3.82E−02	4.71E−02	5.81E−02	1.57E−04	**5.81E−06**	1.91E−01	3.16−02	4.92E−03	1.58E−02	1.57E−03
		B	2.84E−04	1.94E−02	8.24E−03	3.66E−04	2.55E−05	9.51E−06	**8.94E−07**	9.70E−03	3.82E−04	1.65E−05	4.61E−04	3.92E−05
		SD	3.71E−01	4.14E−01	5.11E−02	3.81E−01	2.48E−02	1.753−03	**2.51E−04**	2.07E−01	3.81E−03	1.94E−03	3.72E−03	2.85E−03
		R	9	12	11	6	4	2	**1**	10	7	3	8	5

(*continued on next page*)

Table 2 (*continued*)

Function	D		GA	SA	PSO	ABC	ICA	FA	RDA	L-SHADE	SEO_1	SEO_2	SEO_3	SEO_4
P11	30	W	0.85	0.85	0.81	0.43	0.875	0.36	0.89	4.05E−01	0.52	**0.26**	0.75	0.87
		M	5.92E−03	2.81E−03	5.72E−03	6.81E−03	1.92E−04	6.42E−04	8.92E−03	2.86E−03	1.93E−05	**5.92E−05**	1.84E−04	8.52E−04
		B	7.83E−05	6.83E−05	5.82E−04	7.22E−05	7.61E−06	4.91E−06	7.61E−06	1.94E−04	6.58E−07	**5.81E−08**	8.94E−07	3.41E−06
		SD	0.0438	0.025072	0.001411	0.0619	0.08662	0.071474	0.027377	7.06E−04	0.00969	0.00405	0.009652	**0.002751**
		R	8	10	12	9	7	6	5	11	3	**1**	2	4
	100	W	4.81	2.75	7.84	4.92	5.37	2.81	3.97	2.69	2.55	**1.64**	2.17	1.98
		M	2.67E−01	8.43E−01	8.53E−01	5.76E−01	6.89E−01	4.76E−02	1.85E−02	3.45E−01	**5.47E−04**	6.13E−02	6.84E−03	5.94E−03
		B	4.82E−04	4.82E−03	4.11E−03	5.92E−03	3.88E−04	5.82E−04	6.11E−04	1.29E−04	5.92E−05	**3.11E−05**	6.73E−04	5.93E−05
		SD	0.83	0.57	0.0411	0.2119	0.366	0.274	0.377	1.83E−01	0.0969	0.0805	**0.0092**	0.20751
		R	8	10	11	9	12	7	6	4	2	**1**	5	3
P12	30	W	2.51	1.45	8.29E−01	1.25	1.45	1.25	5.82E−02	7.25E−01	5.28E−02	6.31E−02	8.11E−02	**2.93E−02**
		M	5.62E−07	3.78E−02	7.88E−05	3.82E−03	7.82E−05	5.83E−02	**6.84E−09**	3.91E−05	3.92E−05	8.15E−06	8.16E−07	8.10E−03
		B	8.99E−09	5.86E−05	7.81E−06	5.18E−04	4.29E−04	6.23E−04	**2.84E−11**	2.15E−04	7.83E−07	2.33E−08	8.42E−10	2.85E−07
		SD	**0.0048**	12.647	1.575753	52.6121	21.88676	26.9302	0.899861	7.30E+00	0.390465	15.82594	0.147952	2.50135
		R	3	8	7	10	12	11	**1**	9	5	4	2	6
	100	W	18.74	8.95	4.73	9.12	10.96	2.85	0.9372	2.98E+00	1.672	**0.38**	0.987	0.854
		M	1.62E−06	0.3447	2.38E−04	1.54E−02	5.66E−01	9.79E−01	**1.26E−07**	1.72E−01	1.45E−03	1.71E−04	9.01E−04	8.10E−03
		B	**6.39E−08**	0.0083	7.38E−05	6.38E−04	4.29E−04	6.19E−03	**6.39E−08**	2.77E−03	5.38E−04	6.31E−06	1.47E−07	2.85E−07
		SD	0.138	18.447	11.5753	59.21	45.86	34.02	0.9861	6.15E+00	**0.0465**	5.4	2.72	4.59
		R	2	12	8	9	10	7	**1**	11	6	5	4	3
Average of rank			7.47	9.13	9.304	9.56	9.608	7.39	4.69	7.86	3.69	**2.39**	3.34	3.47

Table 3
The final outputs of algorithms in benchmark functions during thirty run times in 30 dimensional (B = best, W = worst, M = mean and SD = standard deviation, R = rank).

		GA	SA	PSO	ABC	ICA	FA	RDA	L-SHADE	SEO_1	SEO_2	SEO_3	SEO_4
	W	1.79E−01	0.5039	6.37E−03	0.9846	9.1356	5.26E−02	**2.73E−05**	4.5678	4.18E−05	2.68E−04	3.51E−04	1.58E−03
	M	2.38E−05	0.0236	2.09E−04	0.4401	8.7839	6.91E−05	1.73E−05	4.39195	1.61E−06	**1.77E−09**	7.41E−05	2.71E−06
P1	B	2.89E−05	0.0195	2.71E−04	0.3981	1.9351	3.49E−06	2.95E−06	0.96755	2.18E−06	**7.39E−09**	9.42E−05	6.87E−07
	SD	0.001995	0.609244	0.007972	2.946176	6.293635	0.0042	0.041363	2.097878	0.000747	**0.0003122**	0.008837	0.0007671
	R	6	9	8	10	12	5	4	11	3	**1**	7	2
	W	5.24E−02	0.3589	**4.79E−05**	0.0183	1.9975	3.48E−03	1.84E−02	0.499375	1.64E−03	2.86E−02	0.0042	0.0467
	M	1.44E−07	0.0728	6.33E−08	0.0016	0.9975	1.25E−05	**5.28E−10**	0.3325	4.70E−03	1.61E−05	0.0008	0.0017
P2	B	2.85E−08	0.0016	7.62E−09	0.0009	0.0265	9.34E−05	**8.31E−10**	0.01325	2.97E−04	7.14E−05	4.86E−05	0.0006
	SD	6.95888E−05	0.772718	0.003611	0.897054	2.574623	1.299758	**6.91E−05**	0.858208	0.043709	0.063919	0.105645	0.059042
	R	3	10	2	9	12	6	**1**	11	7	5	4	8
	W	1.64E−01	1.37E−04	1.08E−04	2.06E−02	8.39E−02	6.37E−03	8.32E−03	0.04195	**3.45E−05**	1.28E−04	1.64E−03	5.98E−04
	M	5.47E−07	2.19E−10	2.53E−07	4.59E−03	2.55E−03	3.26E−04	3.28E−16	0.0085	2.38E−18	6.53E−10	5.28E−19	**2.19E−21**
P3	B	1.04E−08	8.54E−10	6.71E−08	1.87E−05	3.87E−04	5.81E−05	1.97E−18	0.000194	7.13E−20	2.43E−11	9.63E−19	**3.87E−23**
	SD	0.00020803	2.45E−08	6.74E−05	0.801829	0.010575	0.003137	6.11E−15	0.003525	9.6E−14	7.82E−09	1.3E−16	**5.02E−19**
	R	7	6	8	9	12	10	4	11	2	5	3	**1**
	W	86.1937	9.4382	27.8325	11.5473	6.5502	10.4588	4.89E−01	21.54843	5.233	9.1842	**4.59E−01**	4.371
	M	10.4588	0.164	1.34E−02	1.8633	0.4402	1.9382	**1.07E−04**	3.486267	0.0274	1.2347	0.0243	0.2293
P4	B	4.2813	0.0037	2.68E−02	0.0865	0.0573	0.0865	**2.34E−05**	2.14065	0.0043	0.0573	0.0107	0.0107
	SD	1.513564	0.675702	0.335663	2.76456	2.893141	6.378975	**0.001467**	0.756782	0.255964	1.658875	0.311946	0.235281
	R	12	8	2	9	7	10	**1**	11	3	6	4	5
	W	109.2195	**86.2819**	109.2195	148.1925	148.1925	109.2195	**86.2819**	87.0481	**86.2819**	**86.2819**	**86.2819**	**86.2819**
	M	27.7778	26.5151	47.1011	132.2873	9.63E+01	88.49522	26.2837	44.0957	**21.7232**	26.6891	28.1382	28.8505
P5	B	22.6841	22.6841	26.4192	28.1931	26.4192	28.1931	22.6841	**14.09655**	21.0266	26.4192	26.4192	26.4192
	SD	78.76174	32.8162	614.4787	1744.205	638.48	1384.006	177.0888	581.4017	**26.86603**	87.30601	43.96195	27.45818
	R	5	4	9	12	10	11	3	1	2	8	7	6
	W	3	3	1	3	7	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	M	0.0008	1.89E−03	8.54E−18	2.43E−09	2.57E−01	**0**	**0**	0.0082	**0**	**0**	**0**	**0**
P6	B	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	SD	2.4562	6.045955	7.348469	62.7283	18.54724	5.95219	1.06066	1.573	**0**	0.353553	0.353553	**0**
	R	6	9	10	12	11	7	5	8	**1**	3	4	2
	W	14.7633	16.3897	14.7633	14.7633	**10.8849**	**10.8849**	**10.8849**	13.6283	**10.8849**	**10.8849**	**10.8849**	**10.8849**
	M	8.4918	11.1288	10.1528	10.8443	9.0834	9.8375	8.3482	9.279167	8.4883	9.3847	9.2113	**8.3181**
P7	B	8.4511	8.4511	8.4511	8.4511	8.4511	8.4511	8.2278	**4.22555**	8.2278	8.4511	8.4511	**8.2278**
	SD	0.428202	0.27759	**0.251805**	0.422073	0.455663	0.561185	0.559116	0.187062	0.553255	0.535912	0.322646	0.661973
	R	10	8	7	9	11	12	4	**1**	3	6	5	2
	W	3.82E−02	3.82E−02	3.82E−02	2.32E−01	9.35E−01	8.57E−02	3.82E−02	0.23375	3.82E−02	3.82E−02	**4.59E−04**	**4.59E−04**
	M	1.23E−04	3.32E−06	6.54E−06	1.54E−02	4.83E−02	1.28E−04	4.37E−07	0.02415	1.84E−07	**1.28E−10**	4.64E−08	3.95E−08
P8	B	5.92E−04	1.48e−08	4.79E−07	5.38E−05	5.92E−04	6.19E−05	**8.29E−10**	0.000296	**8.29E−10**	**8.29E−10**	**8.29E−10**	**8.29E−10**
	SD	0.000564	1.03E−05	0.000109	0.284115	0.088928	0.036815	0.000181	0.029643	8.7E−05	**4.32E−08**	3.04E−05	0.000129
	R	11	6	7	8	12	9	5	10	4	**1**	2	3
	W	48.1923	44.5712	44.5712	48.1923	48.1923	37.2899	37.2899	24.09615	**26.4891**	27.4173	**26.4891**	**26.4891**
	M	19.8992	32.9452	23.879	28.9711	17.8435	29.5132	16.2931	9.657033	12.4404	**3.9806**	6.6029	7.5429
P9	B	17.4938	17.4938	12.4821	22.5918	16.3917	16.3917	8.5474	11.2959	4.3947	**2.5933**	**2.5933**	**2.5933**
	SD	9.461557	4.940765	12.68426	10.67775	0.182485	25.79303	20.5363	**3.55925**	5.558045	3.897055	5.810922	7.91078
	R	11	10	9	12	7	8	5	6	4	**1**	2	3
	W	5.21E−01	5.2831	0.8413	5.2810	2.5419	3.45E−02	**1.54E−03**	2.64155	0.8413	0.8244	0.8244	0.8413
	M	6.32E−04	0.3201	0.0023	0.0007	0.4816	1.63E−04	**1.54E−07**	0.16005	0.0173	0.0825	0.0016	0.0017
P10	B	1.47E−06	0.0796	3.28E−03	3.81E−05	0.0373	3.81E−05	**8.43E−09**	0.0398	3.28E−03	3.28E−03	3.28E−03	3.28E−03
	SD	0.001852	0.750565	0.410036	0.453589	0.264498	0.001307	**7.55E−05**	0.37528	0.061496	1.029812	0.062247	0.013683
	R	2	12	9	3	10	4	**1**	11	5	8	6	7

(*continued on next page*)

Table 3 (*continued*)

		GA	SA	PSO	ABC	ICA	FA	RDA	L-SHADE	SEO_1	SEO_2	SEO_3	SEO_4
	W	5.8744	1.8863	5.8744	5.8744	5.8744	1.2434	1.2434	1.958133	1.2434	1.2434	**0.8622**	**0.8622**
	M	0.0203	0.0097	0.0018	0.0061	0.0673	0.0034	1.23E−03	0.01015	**1.25E−05**	0.0091	0.0287	0.0593
P11	B	0.0004	4.21E−03	1.38E−03	5.39E−04	1.38E−03	5.39E−04	5.39E−04	0.0002	**4.71E−07**	5.39E−04	5.39E−04	3.28E−03
	SD	0.343835	0.025072	0.001411	0.362119	0.38662	0.271474	0.027377	0.114612	0.324969	**0.244205**	0.279652	0.259751
	R	12	11	9	4	7	8	5	10	**1**	2	3	6
	W	2.5811	1.4925	8.29E−01	1.4925	1.4925	1.4925	5.82E−02	0.74625	5.28E−01	6.31E−01	8.11E−01	**2.93E−02**
	M	1.62E−06	0.3447	2.38E−04	1.54E−02	5.66E−01	9.79E−01	**1.42E−07**	0.17235	1.45E−03	1.71E−04	9.01E−04	8.10E−03
P12	B	**6.39E−08**	0.0583	7.38E−05	6.38E−04	4.29E−04	6.19E−03	**6.39E−08**	0.02915	5.38E−04	6.31E−06	1.47E−07	2.85E−07
	SD	**0.001838**	12.65447	1.575753	52.6121	21.88676	26.9302	0.899861	6.327235	0.390465	15.82594	0.147952	2.50135
	R	**1**	12	7	8	9	10	2	11	6	5	3	4
Average of rank		7.16	8.75	7.25	8.75	10	8.33	**3.33**	8.5	3.416	4.25	4.16	4.083

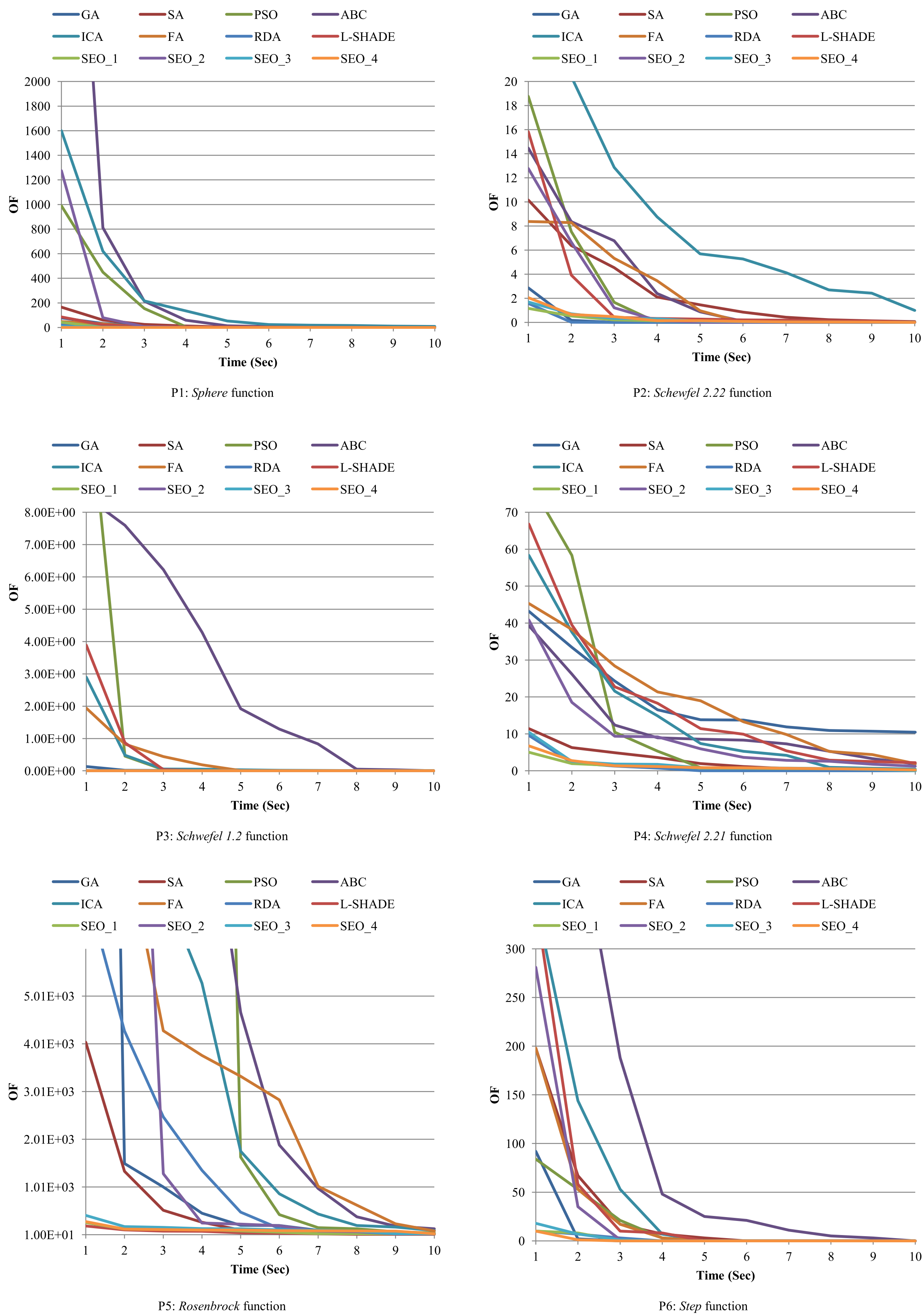
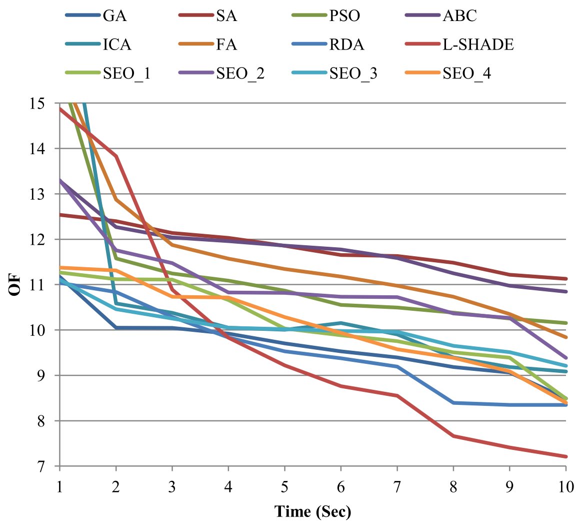
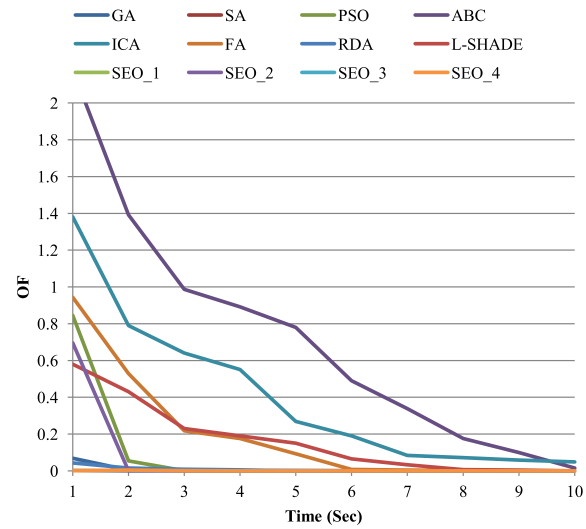


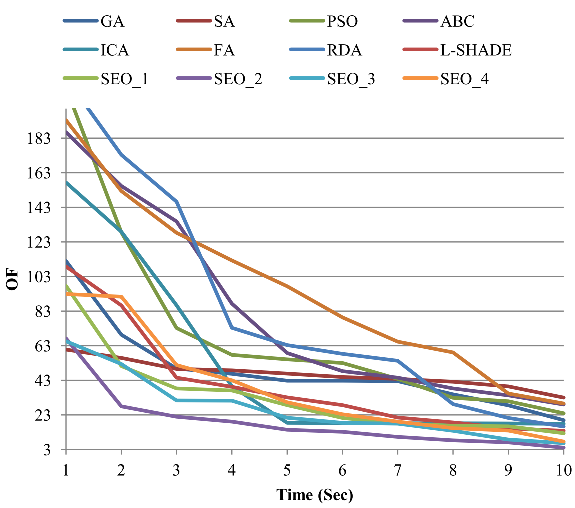
Fig. 10. The behavior of algorithms in benchmark functions.



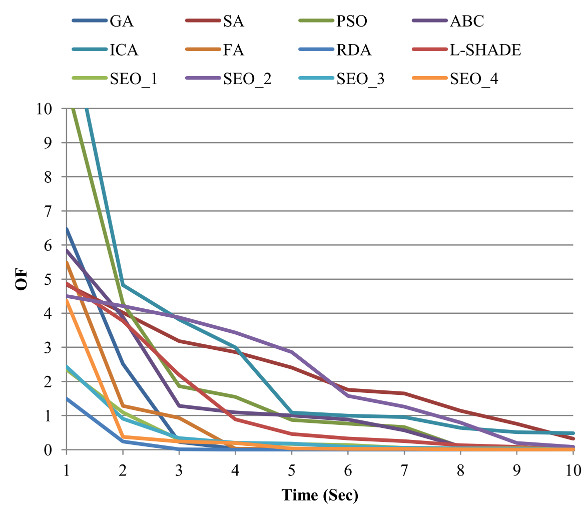
P7: *Quratic* function



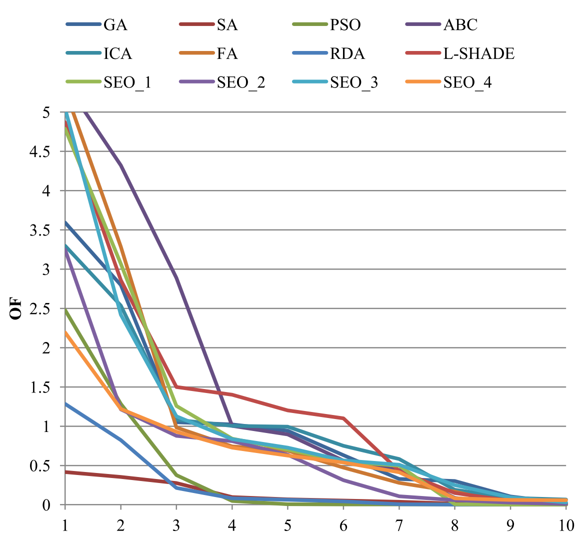
P8: *Greiwank* function



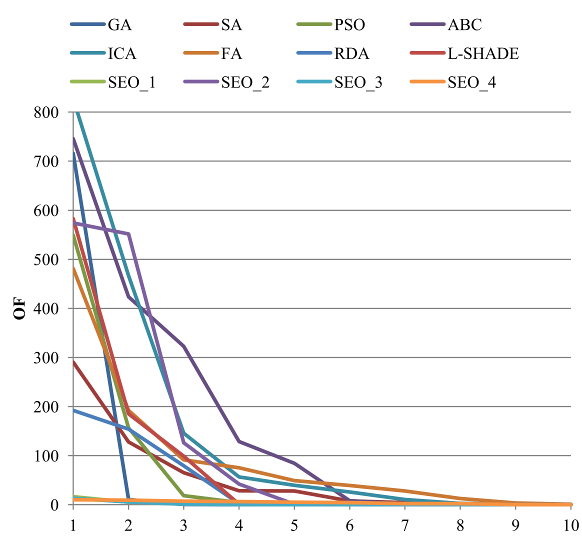
P9: *Rastrigin* function



P10: *Ackley* function



P11: *Penalised* function1



P12: *Penalised* function2

Fig. 10. (*continued*)

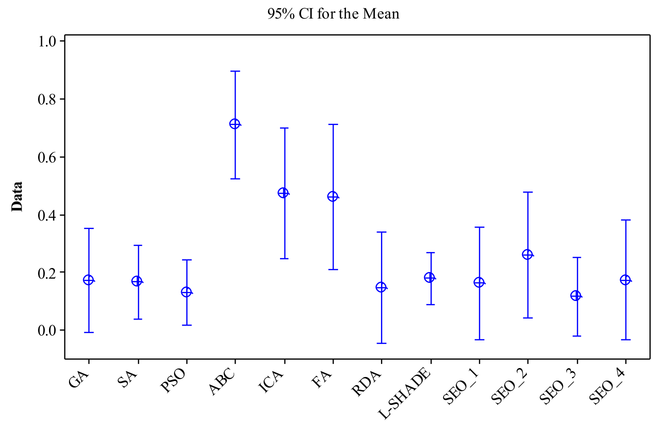


Fig. 11. The Means plot and LSD intervals of algorithms based equal computational time.

Table 4
Sensitivity analyses of algorithms on very high dimensional function for P9.

		SEO_1	SEO_2	SEO_3	SEO_4
$D = 200$	W	**57.89**	62.86	63.91	64.18
	M	**38.65**	39.52	45.67	44.18
	B	**19.64**	21.75	24.81	20.87
	SD	18.46	**8.765**	17.46	20.74
	R	**1**	3	4	2
$D = 400$	W	110.25	**104.85**	109.35	108.71
	M	52.81	**48.92**	50.84	56.83
	B	33.86	**32.78**	36.81	34.92
	SD	**25.81**	29.75	33.82	31.45
	R	2	**1**	4	3
$D = 600$	W	118.32	**112.67**	118.95	115.94
	M	**61.85**	63.92	75.31	72.67
	B	49.71	**47.93**	55.82	64.19
	SD	39.82	**36.81**	54.28	48.72
	R	2	**1**	3	4
$D = 800$	W	179.43	**164.93**	195.26	187.66
	M	93.72	**88.71**	109.75	104.68
	B	84.21	**72.19**	81.57	85.66
	SD	56.77	**52.18**	54.83	52.88
	R	3	**1**	2	4
$D = 1000$	W	198.24	**189.52**	207.45	217.23
	M	114.89	**102.75**	125.89	129.73
	B	95.27	**83.22**	103.82	100.57
	SD	68.73	**59.16**	65.49	71.88
	R	2	**1**	4	3
Average rank		2	**1.4**	3.4	3.2

similar versions in most of high dimensional except D = 200 in which SEO_1 shows a better rank.

4.2. Engineering applications

In this work, authors highlight the need of evaluating the proposed algorithm with a set of famous engineering problems and comparing the results with some other well-known meta-heuristics. The famous engineering applications can be divided into the scheduling and the transportation problems. Scheduling is utilized to assign plant and machinery resources, human resources plan (Saskena, 1970), production processes plan and purchase materials (Hajiaghaei-Keshteli et al., 2011). Hence, one of the important zones in engineering is Scheduling problem (SP). In this research area, the Single Machine Problem (SMP) and Parallel Machine Scheduling (PMS) problem are proposed as significant traditional problems. Besides, the Transportation Problem (TP) is one of the basic models in engineering sciences. Accordingly, the performance of the proposed algorithm is explored by a well-known problem in TP, namely, Vehicle Routing Problem (VRP). At the last but not the least, the Fixed Charge Transportation problem (FCTP) is utilized in this paper to show another engineering application of SEO proposed. Finally, a multi-objective case optimization problem is applied by a Supply Chain Network Design (SCND) model.

4.2.1. Scheduling problems

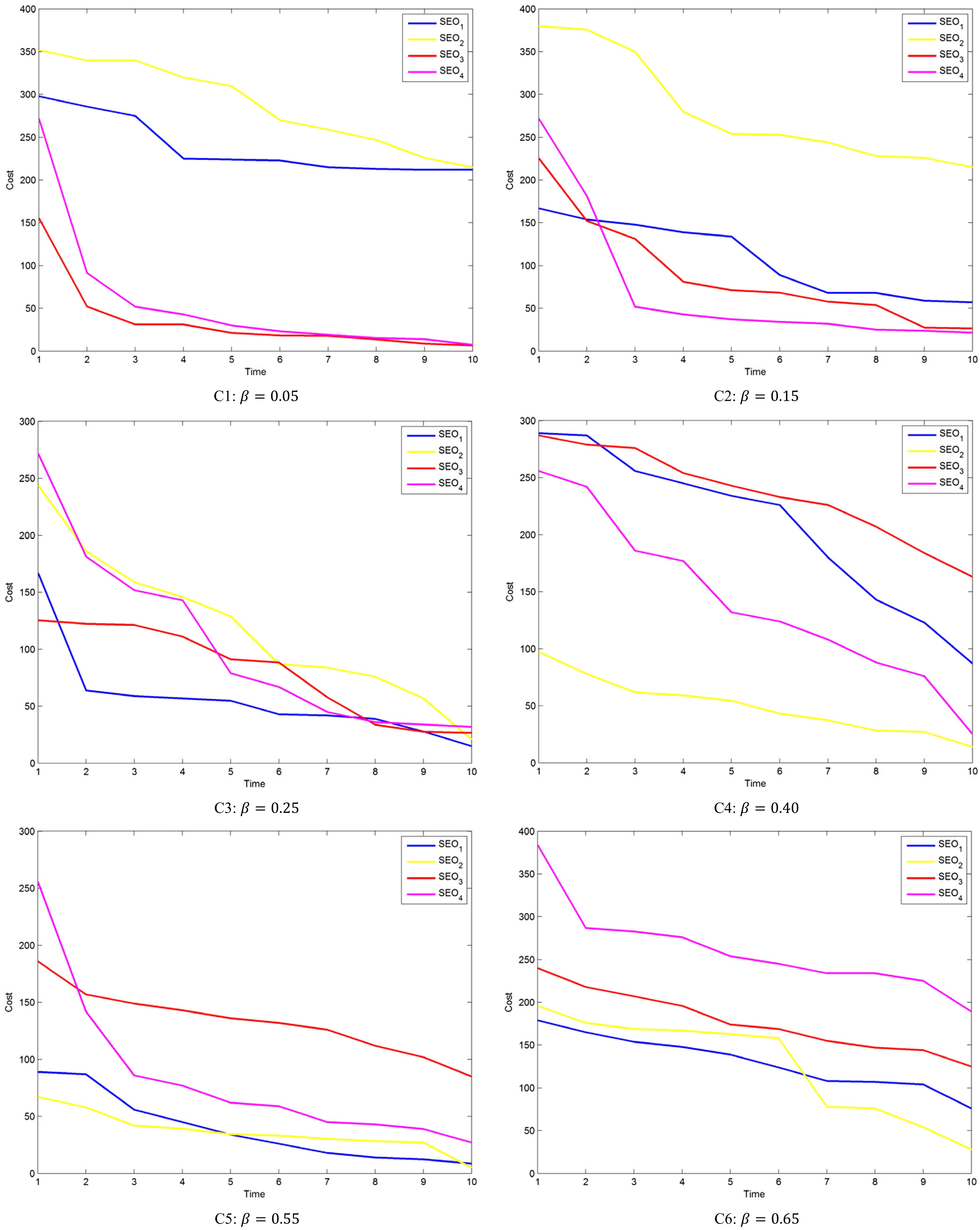
4.2.1.1. SMP. In this part, a single machine scheduling problem is considered. Single-machine scheduling or single-resource scheduling is the process of assigning a group of tasks to a single machine or resource. The tasks are arranged so that one or many performance measures may be optimized. In many practical scheduling problems, costs arising from both earliness and tardiness of individual jobs must be minimized. For example in production systems in which there are shipping dates for orders, inventory carrying cost are incurred if jobs are finished earlier than the shipping dates, and shortage costs (penalty costs and backorder costs) are incurred if jobs are finished later than the shipping dates. To minimize the total costs, tardiness and earliness of jobs should be minimized. The earliness and tardiness are defined as $\max_i(d_i - C_i, 0)$ and $\max_i(C_i - d_i)$, respectively, where d_i the due date and C_i is completion time of job i. In order to explore more, recent studies are suggested: Ben-Yehoshua and Mosheiov, 2016; Zhao et al., 2016.

This problem is known as an NP-*hard* problem (Feldman and Biskup, 2003). In the literature, there have been several studies reporting the application of meta-heuristics used in this type of problem (Hoogeveen and van de Velde, 1991; James, 1997; Lee and Kim, 1995; Feldman and Biskup, 2003; Hall et al., 1991).

In the encoding scheme, a two-stage technique called Random-Key (RK) is utilized to solve the proposed discrete problems. Researchers have studied the application of this technique repeatedly in the recent decade (Golshahi-Roudbaneh et al., 2017; Hajiaghaei-Keshteli et al., 2010). This technique helps user to solve the problem with different methods and operators. In RK, the two phases are existed. Firstly, a solution is made by random numbers. Secondly, this solution is converted to a feasible solution by using certain method. Hence, in the first phase, an array of variable between zero and one is initialized. Then, the variables of this array are sorted to specify the sequence of jobs to compute the objective function. Fig. 13 displays these two phases during main step of SEO for the proposed discrete problem. In this instance, the rate of β is equal to 0.5 in all techniques. In addition, the total number of traits is equal to 4, too.

To compare the proposed algorithms, three powerful methods are obtained: GA, ICA and RDA are used in this study. All of parameters of methods are tuned by Taguchi experimental design method (Taguchi, 2000). The tuned values of algorithms have been reported in Supplementary Materials F3. Furthermore, the problem tests are classified as well. The 6 different sizes of test problem are defined. In order to be fair, searching time should be identical in all algorithms, is equal to $\left(\frac{Number\ of\ jobs}{10}\right)$ seconds. The results are illustrated in Table 5. These results show, firstly, that the efficiency and performance of proposed SEO among the algorithms; secondly, that SEO_2 depicts the more superiority among others. It should be noted that the results shown in the table, is the best solution found among thirty runs. Also, the interaction of problem size and behavior of algorithms are depicted in Fig. 14, in terms of mean RPD in thirty runs. In order to know more about RPD, these papers are suggested: Golshahi-Roudbaneh et al. (2017) and also Hajiaghaei-Keshteli and Aminnayeri (2014).

4.2.1.2. PMS. To explore the scheduling problems, this section aims to consider a parallel machine scheduling problem (PMS) as an important engineering problem. The previous studies in this area show, firstly, that nowadays the importance of this type of problem in industry is increased; secondly, that it is known as an NP-*hard* problem; and lastly, several heuristics and meta-heuristics are developed to handle this problem (Cheng and Sin, 1990).



C1: $\beta = 0.05$

C2: $\beta = 0.15$

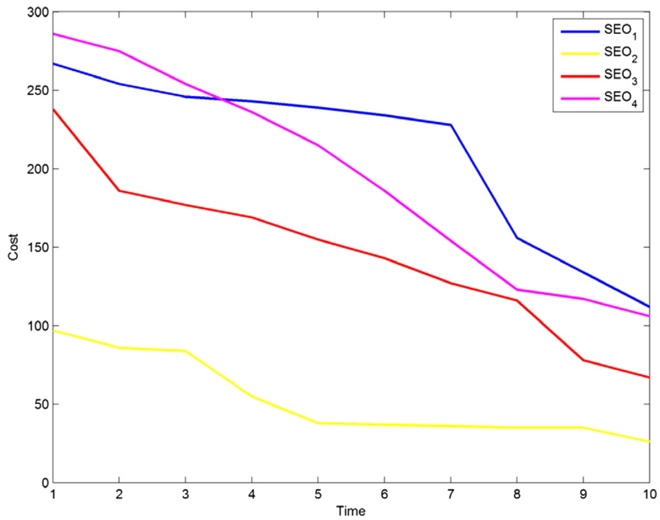
C3: $\beta = 0.25$

C4: $\beta = 0.40$

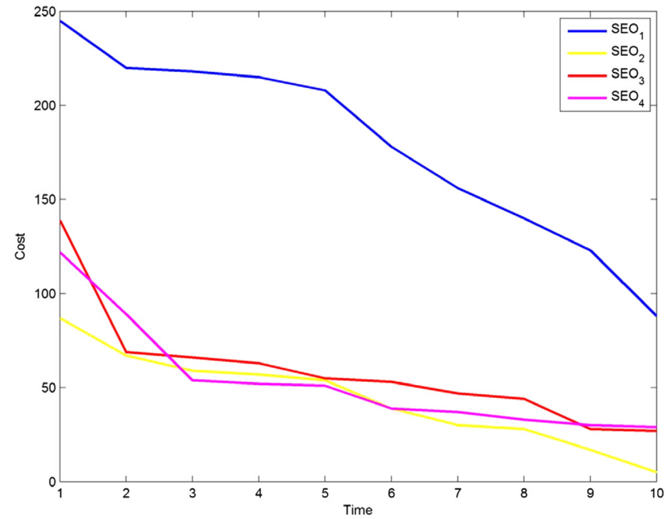
C5: $\beta = 0.55$

C6: $\beta = 0.65$

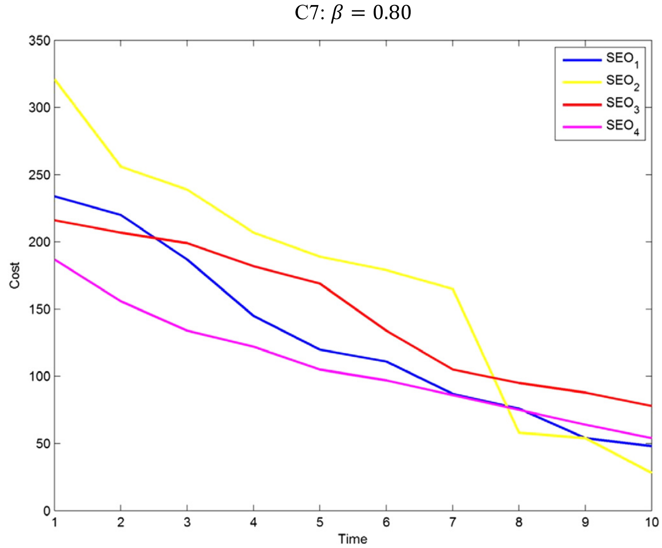
Fig. 12. The convergence analysis of proposed algorithm in different cases for each technique.



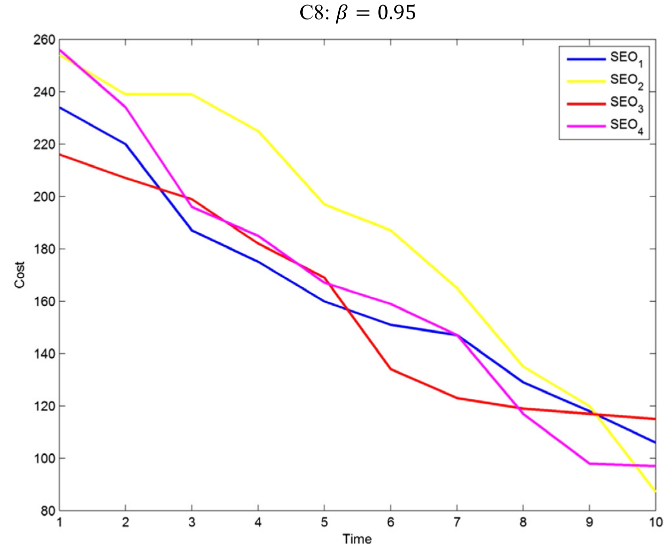
C7: $\beta = 0.80$



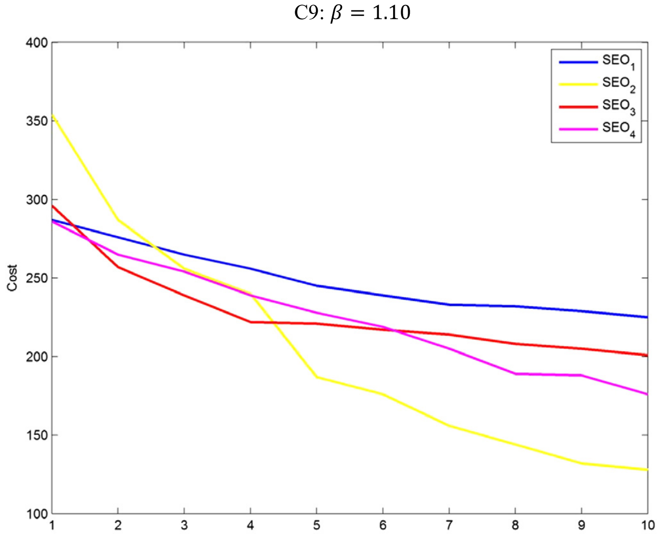
C8: $\beta = 0.95$



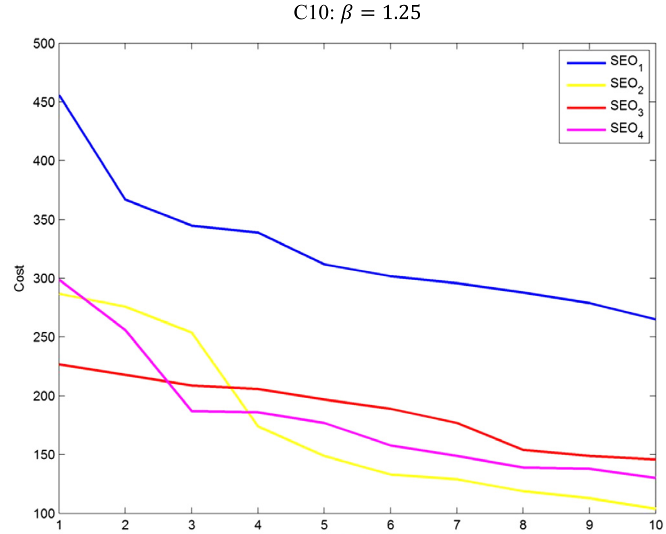
C9: $\beta = 1.10$



C10: $\beta = 1.25$



C11: $\beta = 1.40$



C12: $\beta = 1.55$

Fig. 12. (*continued*)

Defender:

0.67	0.34	0.89	0.26
3	2	4	1

Attacker:

0.08	0.76	0.12	0.73
1	4	2	3

Technique 1:

0.55	0.57	0.64	35
2	3	4	1

$$0.55 = 0.67 \times (1 - \sin 0.5 \times rand) + \left(\frac{0.67 + 0.08}{2}\right) \times \sin 0.5 \times rand$$

$$0.57 = 0.34 \times (1 - \sin 0.5 \times rand) + (\frac{0.34 + 0.76}{2}) \times \sin 0.5 \times rand$$

$$0.64 = 0.89 \times (1 - \sin 0.5 \times rand) + (\frac{0.89 + 0.12}{2}) \times \sin 0.5 \times rand$$

$$0.35 = 0.26 \times (1 - \sin 0.5 \times rand) + (\frac{0.26 + 0.73}{2}) \times \sin 0.5 \times rand$$

Technique 2:

0.22	0.95	0.18	0.48
2	4	1	3

0.72	0.56	0.39	0.47
4	3	1	2

$$0.22 = 0.08 \times (1 - \sin 0.5 \times rand) + \left(\frac{0.67 + 0.08}{2}\right) \times \sin 0.5 \times rand$$

$$0.95 = 0.76 \times (1 - \sin 0.5 \times rand) + (\frac{0.34 + 0.76}{2}) \times \sin 0.5 \times rand$$

$$0.18 = 0.12 \times (1 - \sin 0.5 \times rand) + (\frac{0.89 + 0.12}{2}) \times \sin 0.5 \times rand$$

$$0.48 = 0.73 \times (1 - \sin 0.5 \times rand) + (\frac{0.26 + 0.73}{2}) \times \sin 0.5 \times rand$$

$$0.72 = 0.67 \times (1 - \sin (\left(\frac{3.14}{2}\right) - 0.5) \times rand) + \left(\frac{0.67 + 0.08}{2}\right) \times \sin (\left(\frac{3.14}{2}\right) - 0.5) \times rand$$

$$0.56 = 0.34 \times (1 - \sin (\left(\frac{3.14}{2}\right) - 0.5) \times rand) + (\frac{0.34 + 0.76}{2}) \times \sin(\left(\frac{3.14}{2}\right) - 0.5) \times rand$$

$$0.39 = 0.89 \times (1 - \sin (\left(\frac{3.14}{2}\right) - 0.5) \times rand) + \left(\frac{0.89 + 0.12}{2}\right) \times \sin \left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand$$

$$0.47 = 0.26 \times (1 - \sin (\left(\frac{3.14}{2}\right) - 0.5) \times rand) + (\frac{0.26 + 0.73}{2}) \times \sin(\left(\frac{3.14}{2}\right) - 0.5) \times rand$$

Technique 3:

0.58	0.30	0.64	0.31
3	1	4	2

$$0.58 = 0.67 \times (1 - \sin 0.5 \times rand) + ((0.67 + 0.08 \times \sin(\left(\frac{3.14}{2}\right) - 0.5) \times rand)/2) \times \sin\ 0.5\ \times rand$$

$$0.30 = 0.34 \times (1 - \sin 0.5 \times rand) + ((0.34 + 0.76 \times \sin \left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand)/2) \times \sin\ 0.5\ \times rand$$

$$0.64 = 0.89 \times (1 - \sin 0.5 \times rand) + ((0.89 + 0.12 \times \sin \left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand)/2) \times \sin\ 0.5\ \times rand$$

$$0.31 = 0.26 \times (1 - \sin 0.5 \times rand) + ((0.26 + 0.73 \times \sin \left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand)/2) \times \sin\ 0.5\ \times rand$$

Technique 4:

0.35	0.29	0.18	0.23
4	3	1	2

$$0.35 = 0.67 \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand \times (1 - \sin 0.5 \times rand) + ((0.67\ \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand + 0.08)/2)\ \times \sin\ 0.5\ \times rand$$

$$0.29 = 0.34 \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand \times (1 - \sin 0.5 \times rand) + ((0.34\ \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand + 0.76)/2)\ \times \sin\ 0.5\ \times rand$$

$$0.18 = 0.89 \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand \times (1 - \sin 0.5 \times rand) + ((89\ \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand + 0.12)/2)\ \times \sin\ 0.5\ \times rand$$

$$0.23 = 0.26 \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand \times (1 - \sin 0.5 \times rand) + ((0.26\ \times \sin\left(\left(\frac{3.14}{2}\right) - 0.5\right) \times rand + 0.73)/2)\ \times \sin\ 0.5\ \times rand$$

Fig. 13. The encoding scheme of proposed SEO in four versions.

Table 5
The results of Single Machine Problem (SMP).

Number of jobs	Time (s)	GA	ICA	RDA	SEO_1	SEO_2	SEO_3	SEO_4
50	5	8019	8912	8358	8641	7934	8249	7831
100	10	111940	118537	114255	112269	109802	118357	110452
150	15	256639	254402	254517	248842	250374	248573	247721
200	20	962543	995432	998197	956733	946702	972881	958941
250	25	1636638	1622396	1609515	1529561	1495936	1671159	1502090
300	30	2294702	2327774	2308051	2204118	2380018	2239834	2358895

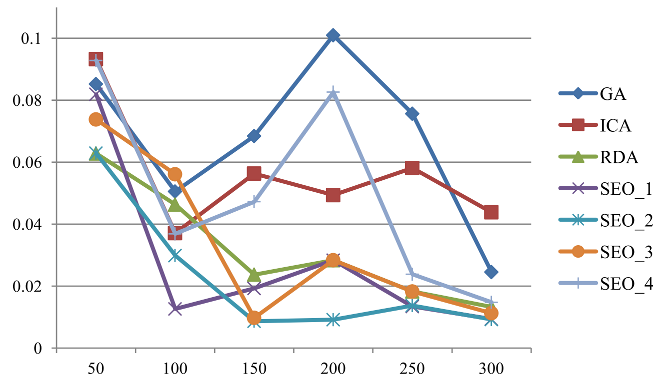


Fig. 14. The interaction between problem size of SMP and behavior of algorithms in terms of mean RPD in thirty time runs.

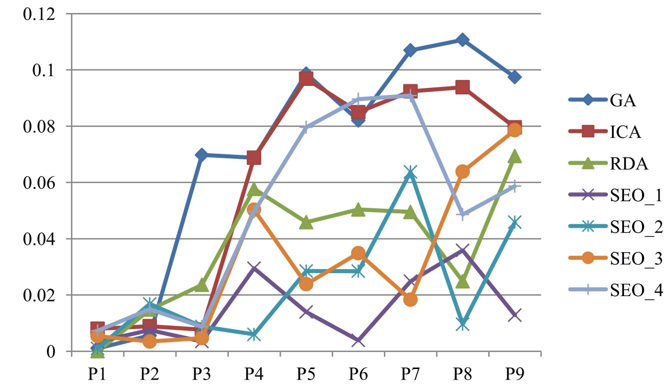


Fig. 15. The interaction between problem size of PMS and behavior of algorithms in terms of mean RPD in thirty time runs.

The PMS involves of scheduling a set of jobs N on a set of parallel machine M without interruption. Each job j (index $j = 1, \ldots, n$) becomes available for processing at time zero and has to be processed by exactly one out of the m (index $i = 1, \ldots, m$) parallel machines. Each machine can process only one job at a time and once a job is started at a certain machine, no preemption is permitted. In fact, various classes of PMS problems have been described to minimize the maximal completion time (C_i) of the machines. On the other word, ($C_{max} = \max_i C_i$) depends on how the processing time of jobs is predefined. Here, the processing time p_j of a job is independent of the machine. And also each machine i has a different speed s_i for the processing the jobs. Hence, the processing time of a job j on machine i is computed as follow: $p_{ij} = \frac{p_j}{s_i}$. When the processing time p_{ij} depends on the machine on which the job j is scheduled. For more information about the formulation of problem, see: Sels et al. 2015; Liaw, 2016.

The four versions of proposed algorithm are obtained to solve this problem. And also three powerful methods: GA, ICA and RDA are used to compare the result. All of parameters of meta-heuristic methods are tuned by Taguchi experimental design method. These tuned values of algorithms' parameters have been reported in Supplementary Materials F4. 9 different sizes of test problem are defined. The problems are numbered as P1 to P9. Table 6 is shown the final outputs of algorithm. Also in each size the searching time is specified as seen in the table. The results show the efficiency and performance of proposed algorithm to reach the best optimal solutions. Consequently, it seems the SEO_2 is better and more powerful than others. At the end, the interaction of problem size and behavior of algorithms is illustrated in Fig. 15, in terms of mean RPD in thirty runs.

4.2.2. Transportation problems

4.2.2.1. VRP. This section presents the application of the proposed novel meta-heuristic to the basic vehicle routing problem (VRP) as a kind of TP, in which customers of known demand are supplied from a single depot. Vehicles are subject to a weight limit and, in some cases, to a limit on the distance traveled. Only one vehicle is allowed to supply each customer. The VRP is happened when some customers are visited each other by some vehicles. In basic of VRP consists of a number of customers each requiring a specified weight of goods to be delivered. Each vehicle can carry a limited weight and may also be restricted in the total distance it can travel. In practice, this is often taken to be equivalent to minimizing the total distance traveled, or to minimizing the number of vehicles used and then minimizing total distance for this number of vehicles. For stating more information about the mathematical model and its description, see: Laporte, 1992; Lalla-Ruiz et al., 2016.

This problem is known as an NP-*hard* problem. The problem has been solved by using heuristics and meta-heuristics that include: Tabu search, simulated annealing, genetic algorithm, ant colony Optimization and variable neighborhood search and also bat algorithm are reported to solve this problem (Osman, 1993; Gendreau et al., 1994; Rego and Roucairol, 1996; Barbarosoglu and Ozgur, 1999; Hiquebran et al., 1994; Bullnheimer et al., 1999; Baker and Ayechew, 2003; Sarasola et al., 2016; Zhou et al., 2016).

Four meta-heuristics called GA, ICA, RDA and our proposed method, SEO in four versions, are presented in this section to handle the problem. The parameters are tuned with Taguchi experimental design method. These tuned values have been reported in Supplementary Materials F5. 10 different sizes are generated as test problems and numbered as P1 to P10. In order to be justly, like previous sections, searching time in each algorithm is based on the size of test problems as shown in Table 7. Each algorithm is performed for thirty times and noted the best output of method. To show the performance of proposed method, the results are given in Table 7. As seen in this table, the SEO_2 reached the better value and found the better solution in each test problems. Besides, the affection of the problem size on the proposed methods is analyzed as depicted in Fig. 16. It shows the interaction between the quality of the presented algorithms and the size of problems in terms of RPD in thirty runs. It is obviously that SEO exhibit robust performance, while the problem size increases, except in P5 the RDA has reached the better value.

4.2.2.2. FCTP. To investigate a famous kind of TP, the fixed charge transportation problem (FCTP) is illustrated in this part. The fixed charge problem is a problem of practical interest in both business and industry. The FCTP is a special case of the fixed cost linear programming problem, already proposed in the origins of the Operations Research

Table 6
The results of Parallel Machine Scheduling (PMS).

P_i	Number of jobs	Number of machines	Time (s)	GA	ICA	RDA	SEO_1	SEO_2	SEO_3	SEO_4
P1	10	3	100	87	87	87	87	84	87	87
P2	18	4	150	121	122	121	119	115	121	122
P3	24	6	200	132	127	125	126	125	127	127
P4	36	8	300	148	152	147	147	143	145	141
P5	48	12	350	138	136	135	139	135	134	138
P6	60	16	400	166	158	156	159	157	155	155
P7	96	20	450	223	232	216	216	210	217	215
P8	142	24	500	289	284	273	271	272	277	273
P9	184	28	600	354	367	334	342	332	335	341

Table 7
The results of Vehicle Routing Problem (VRP).

P_i	Cities	Vehicles	Time (s)	GA	ICA	RDA	SEO_1	SEO_2	SEO_3	SEO_4
P1	8	3	15	354.4722	354.4722	354.4722	354.4722	354.4722	354.4722	354.4722
P2	10	3	15	468.4243	506.4388	461.9244	467.3674	458.1982	473.2938	478.2938
P3	14	4	20	708.2543	534.0541	522.785	689.8417	518.8473	631.2984	539.2938
P4	20	4	20	667.3689	651.4673	594.0197	571.2938	578.2838	568.9382	592.7384
P5	25	5	40	1187.3328	815.8704	794.7334	829.1925	810.3844	926.3746	877.1928
P6	30	5	40	1026.0682	998.7305	975.1149	932.3847	956.3847	945.2761	981.4958
P7	40	6	50	1403.9785	1642.2008	1367.1802	1457.2938	1322.3945	1387.3847	1357.9182
P8	50	7	80	1608.58	1861.3722	1584.5904	1629.3948	1623.9583	1702.5837	1629.3948
P9	60	7	80	1859.7725	2319.0633	1748.1033	1748.1033	1774.8644	1802.8237	1733.0384
P10	70	8	90	2832.9527	3058.5962	2820.6076	2745.3766	2726.1827	2813.2983	2774.9157

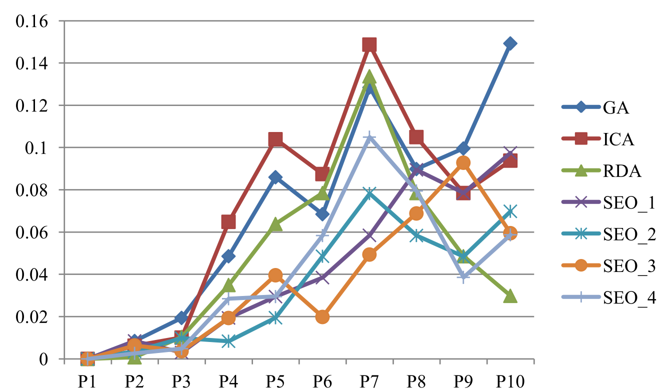


Fig. 16. The interaction between problem size of VRP and behavior of algorithms in terms of mean RPD in thirty time runs.

(Hirsch and Dantzig, 1968). The objective is to find the combination of routes that minimizes the total variable and fixed costs while satisfying the supply and demand requirements of each origin and destination. It has been shown that this fixed cost transportation problem is NP-*hard* problem (Hirsch and Dantzig, 1968). Klose (2008) shows a particular case of FCTP, the Single Source FCTP, is NP-*hard*, which also proves the NP-hardness of FCTP. Since the problem is NP-*hard*, the computational time to obtain exact solution increases in a polynomial fashion and very quickly becomes extremely difficult and long as the size of the problem increases. There have been several meta-heuristic methods which are used to handle this dilemma. The Problem is solved via spanning tree-based genetic algorithm by Hajiaghaei-Keshteli et al. (2010). They solved two examples of FCTP problem with various sizes via GA. Besides, El-Sherbiny and Alhamali (2013) solved the same problem by a hybrid particle Swarm algorithm with artificial immune learning in which a flexible particle is used instead of Prüfer number. Molla-Alizadeh-Zavardehi et al. (2011) used the Artificial Immune System (AIS) and Genetic Algorithm (GA) to solve this problem. Also, Molla-Alizadeh-Zavardehi et al. (2013) solved a fuzzy fixed charge with some famous meta-heuristics, such as: Simulated Annealing (SA) and Variable Neighborhood Search (VNS) as well as a hybrid method from both algorithms. In addition, Xie and Jia (2012) formulated FCTP using a mixed integer programming model based on steady-state genetic algorithm as framework, and minimum cost flow algorithm as decoder. They also proposed a hybrid genetic algorithm named NFCTP-HGA as a solution method of the model. Also Lotfi and Tavakkoli-Moghaddam (2013) used a priority based encoding for solving the FCTP via GA (pb-GA) for linear and nonlinear FCTP. And the recent paper, Altassan et al. (2014) designed a coding schema instead of Prüfer number and developed an algorithm for decoding the problem. They used Artificial Immune Algorithm to address the problem. They solved both balanced and unbalanced FCTP without introducing a dummy supplier or a dummy customer in detail. Golmohamadi et al. (2017) proposed a fuzzy fixed charge solid transportation problem considering batch transferring. They considered RDA and a hybrid version of this algorithm to address their problem. Recently, Sadeghi-Moghaddam et al. (2017) considered WOA to solve a FCTP in a fuzzy environment. To study more information about recent researches, see: Calvete et al. (2016) and Chen et al. (2017) and also Sadeghi-Moghaddam et al. (2017).

To explain the mathematical model and its description, see: Sadeghi-Moghaddam et al. (2017). In continuously, to evaluate the performance of proposed SEO, a plan is utilized to generate the test data. Table 8 shows the designed experiments. Also, these test problems are handled with 3 meta-heuristic algorithms else, include: GA, ICA and RDA. The parameters are tuned with Taguchi experimental design method. These tuned values have been reported in Supplementary Materials F6. To verify the statistical validity of the results, an analysis of variance (ANOVA) is performed to accurately analyze the results. The results demonstrate that there is a clear statistically significant difference between the performances of the algorithms. The means plot and LSD intervals (at the 95% confidence level) for all methods are shown in Fig. 17. Besides, In order to evaluate the robustness of the algorithms in different situations, the effects of the problem size on the performance of the algorithms are analyzed. Fig. 18 shows the interaction between the quality of the presented algorithms and the size of problems in terms of RPD in thirty runs.

4.2.3. Multi-objective optimization

In the real world, some problems which have conflict goals such as minimize the costs and maximize the profits are existed. The answer for such problems is a set of solutions called Pareto optimal solutions set (Fathollahi Fard et al., 2017). This set encompasses Pareto optimal

Table 8
Fixed-charge transportation test problems characteristics.

Problem size	Time (s)	Total supply	Problem type	Range of variable costs		Range of fixed costs	
				Lower limit	Upper limit	Lower limit	Upper limit
10 × 10	20	10,000	A	3	8	50	200
10 × 20	20	15,000	B	3	8	100	400
15 × 15	30	15,000	C	3	8	200	800
10 × 30	35	15,000	D	3	8	400	1,600
50 × 50	70	50,000					
30 × 100	90	30,000					
50 × 200	120	50,000					

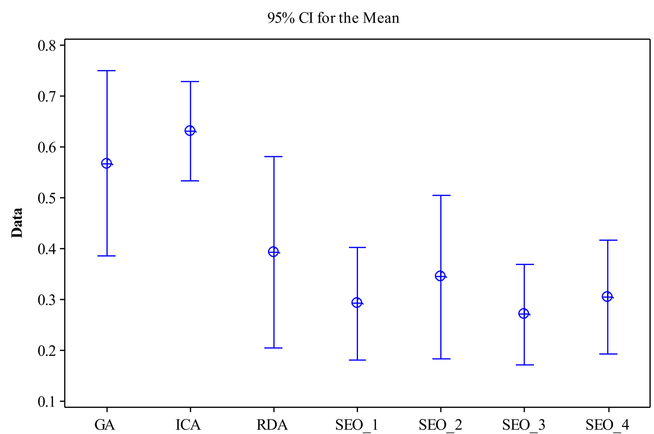


Fig. 17. Means plot and LSD intervals for the algorithms in FCTP.

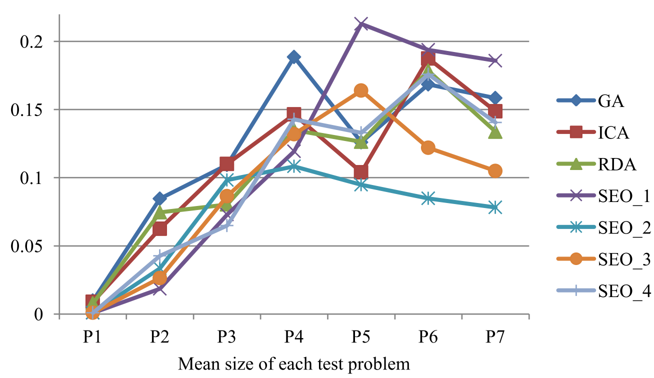


Fig. 18. The interaction between problem size of FCTP and behavior of algorithms in terms of mean RPD in thirty time runs.

solutions which explain the best trade-offs between the objectives (Mirjalili and Lewis, 2015). The model of a multi-objective problem in specific cases with two conflict goals is assumed that follows:

$$\min f_1$$
$$\max f_2$$
$$st.\ g_{(x)} \geq 0$$

when a solution dominated the other solution, if, it had better than in all objective functions. Furthermore, selection the best solution in first front of solutions is considered by calculating the crowding distance between solutions (Samadi et al., 2018). Note that for forming the Pareto optimal solutions set for proposed single-solution meta-heuristic, the best solution at the end of the iteration should be saved. Then, according to these solutions, the population of Pareto front is set (Hajiaghaei-Keshteli and Fathollahi Fard, 2018). Here, a simple multi-objective problem in the term of Supply Chain Network Design (SCND) is presented in the following section.

4.2.3.1. SCND. Supply Chain Management (SCM) can be defined as a management of all parties' concerns in supply chain such as replenishment, movement and storage of raw materials, inventory, and finished goods as well as transportation decisions (Hajiaghaei-Keshteli et al., 2014). In addition, Supply Chain Network Design (SCND) provides strategic decisions playing an important role in the performance of a supply chain and on competitive advantages, and also effects on other decisions in operational and tactical level of supply chain over the time (Fathollahi Fard et al., 2017). Choosing good facilities from the possible locations, determining numbers and capacities of network facilities as well as the material flow through the network are the main concerns and decisions in SCND (Fathollahi Fard and Hajiaghaei-Keshteli, 2018a). These decisions are the most important ones in SCM because any minor or major changes in such crucially important decisions in network influences both directly and indirectly on other decisions in lower levels, and consequently, causes many costs in whole chain (Devika et al., 2014).

There have been some important factors in this research zone: economical, environmental and social impact. These factors form the sustainable supply chain network (Hajiaghaei-Keshteli and Fathollahi Fard, 2018). In addition, several meta-heuristics have been developed to solve the SCND models. In order to study the literature of SCND and its classifications, we refer the reader to study the recent studies (Devika et al., 2014; Govindan et al., 2015b; Govindan and Soleimani, 2017; Samadi et al., 2018).

This section aims to develop a Mixed Integer Linear Programming (MILP) bi-objective closed-loop SCND model with supposition social factors. The developed deterministic model mainly focuses on the treatment process by taking into account four types of facilities called treatment centers in the reverse network: Recovering, Remanufacturing, Recycling, and Disposal. The graphical example of the proposed closed-loop SCND has been reported in Supplementary Materials F3. The considered model is based on the following common assumptions in the literature of this work (Hajiaghaei-Keshteli and Fathollahi Fard, 2018; Devika et al., 2014):

- The demand of each customer must be met.
- The number of facilities and their potential sites in each echelon is predefined.
- No flow exists between the facilities of the same echelon.
- As a special supposition in closed-loop logistics suggested by Van der Laan et al. (1999), it is assumed that the number of the EOL products returned to the collection/inspection centers is a fraction of the customers' demands. In addition, they are allocated to the treatment facilities based on their qualities.

In this problem, the amount of products which have to be manufactured at each plant, the assignment of customers to distribution centers and collection/inspection centers; and also the flow of materials should

Table 9
Model notation.

Indices	Description
i	Index of suppliers: $i \in \{1, 2, \dots, I\}$
j	Index of potential manufacturing centers: $j \in \{1, 2, \dots, J\}$
k	Index of potential distribution centers: $k \in \{1, 2, \dots, K\}$
l	Index of customer zones: $l \in \{1, 2, \dots, L\}$
m	Index of collection/inspection centers: $m \in \{1, 2, \dots, M\}$
n	Index of manufacturing centers: $n \in \{1, 2, \dots, N\}$
p	Index of potential remanufacturing centers: $p \in \{1, 2, \dots, P\}$
r	Index of potential recycling centers: $r \in \{1, 2, \dots, R\}$
s	Index of potential disposal centers: $s \in \{1, 2, \dots, S\}$
e, e'	Index of echelons: $e, e' \in \{i, j, k, l, m, n, p, r, s\}$
f_e, f'_e	Index of facilities in echelon e : $f_e, f'_e \in \{1, \dots, F_e\}$

be decided. In this regard, the notations of the proposed model are illustrated in Tables 9–11.

$$\begin{aligned}\text{Min } OBJ1 = &\sum_j fc_j Y_j + \sum_e \sum_{f_e} fc_{fe} Y_{fe} + \sum_i \sum_j pc_i X_{ij} \\ &+ \sum_j mc_j H_j + \sum_e \sum_{e'} \sum_{f_e} \sum_{f'_{e'}} tc_{f_e f'_{e'}} X_{f_e f'_{e'}} \\ &+ \sum_k \sum_l ac_{kl} d_l Z_{kl} + \sum_l \sum_m (cc_{lm} + vc_m) \\ &\propto_l d_l Z_{lm} - sc_d \left(\sum_n \sum_k X_{nk} \right) \\ &- sc_m \left(\sum_p \sum_j X_{pj} \right) - sc_r \left(\sum_r \sum_i X_{ri} \right) \\ &- sc_u \left(\sum_e \sum_{f_e} \sum_m \gamma_e X_{mf_e} \right)\end{aligned} \tag{9}$$

The first objective wants to minimize the total costs of the network. Generally, the total cost of CLSC is mines its profit accordingly. To establish the facilities, the first and second terms are computing the fixed costs of opening facilities. The third to seventh summations are affiliated with purchasing, manufacturing, handling, transportation and assignment along with collection costs. The last four terms stand for the savings resulted from reusing products at the manufacturing centers, from redistributing recovered or remanufactured products, and from selling products to reuse market. Considering the total cost and profits of a CLSC system is a common goal in different studies. Accordingly, the main novelty of proposed model is to consider the social impacts. In the following, the social objective of the model is computed as:

$$\begin{aligned}\text{Max } OBJ2 = &\varepsilon_{j0} \Bigg[\sum_j fj_j Y_j + \sum_e \sum_{f_e} fj_{f_e} Y_{f_e} + \sum_j vj_j H_j / p_j \\ &+ \sum_k \sum_j vj_k X_{jk} / p_k + \sum_m \sum_l vj_m Z_{lm} d_l \alpha_l / p_m \\ &+ \sum_e \sum_{f_e} \sum_m vj_{f_e} X_{mf_e} \beta_m^{f_e} / p_{f_e} \Bigg] - \varepsilon_{ld} [\sum_j fl_j Y_j \\ &+ \sum_e \sum_{f_e} fl_{f_e} Y_{f_e} + \sum_k \sum_j vl_k X_{jk} / p_k \\ &+ \sum_m \sum_l vl_m Z_{lm} d_l \alpha_l / p_m + \sum_e \sum_{f_e} \sum_m vl_{f_e} X_{mf_e} \beta_m^{f_e} / p_{f_e}]\end{aligned} \tag{10}$$

The second objective function is divided into two parts: job opportunities and workers' damages. Regarding the first issue, the fixed job opportunities are computed by the first and second terms of this formulation. This type of job considers the job positions like managers which are needed independent of the used capacity of the facility. Contrary to the fixed jobs, variable jobs are those that totally pertain on the used capacity of the facility. The third to sixth terms show these types of jobs. By another point of view, a facility requires more workers when it utilizes its whole capacity and, conversely, it uses fewer workers when less capacity is needed. The seventh to eleventh terms stand for the work's damages which are caused either during the establishment of facilities or during the manufacturing and handling of products. Moreover, a CLSC model usually has a set of constraint to limit the bounds of objective functions. Accordingly, the following assertions (11) to (16) ensure that the flow of products is maintained and the demands are convinced.

$$\sum_i X_{ij} = \sum_k X_{jk} \qquad \forall j \tag{11}$$

$$\sum_j X_{jk} = \sum_l d_l Z_{kl} \qquad \forall k \tag{12}$$

$$\sum_{f_e} X_{mf_e} = \beta_m^e \sum_l \alpha_l d_l Z_{lm} \qquad \forall m, e \in \{n, p, r, s\} \tag{13}$$

$$\sum_k X_{nk} = (1 - \gamma_n) \sum_m X_{mn} \qquad \forall n \tag{14}$$

$$\sum_j X_{pj} = (1 - \gamma_p) \sum_m X_{mp} \qquad \forall p \tag{15}$$

$$\sum_i X_{ri} = (1 - \gamma_r) \sum_m X_{mr} \qquad \forall r \tag{16}$$

The amount of products manufactured by each manufacturing center is computed by the following constraint:

$$H_j = \sum_i X_{ij} \qquad \forall j \tag{17}$$

One of the main limitations of CLSC proposed is that each customer zone should be assigned to only one distribution center for a forward system and one collection/inspection center for a reverse one:

$$\sum_k Z_{kl} = \sum_m Z_{lm} = 1 \qquad \forall l \tag{18}$$

Similar to other CLSC models, the variable capacity is considered by a set of constraint. Regarding the suppliers, the amount of products procured from them is restricted by its capacity:

$$\sum_j X_{ij} \le p_i \tag{19}$$

Similar to suppliers, a manufacturing center can manufacture only when it is opened and it has idle capacity:

$$H_j \le p_j Y_j \qquad \forall j \tag{20}$$

The flow of products through a facility is allowed only when the respective facility is operating and has enough capacity as well:

$$\sum_j X_{jk} \le p_k Y_k \qquad \forall k \tag{21}$$

$$\sum_l \propto_l d_l Z_{lm} \le p_m Y_m \qquad \forall m \tag{22}$$

$$\sum_m X_{mf_e} \le p_{f_e} Y_{f_e} \qquad \forall e \in \{n, p, r, s\} \tag{23}$$

To consider the location decisions of model, the number of facilities in each echelon is restricted by a pre-defined number.

$$\sum_j Y_j \le max_j \tag{24}$$

$$\sum_{f_e} Y_{f_e} \le max_e \qquad \forall e \in \{k, m, n, p, r, s\} \tag{25}$$

At the end, binary and positive variables are guaranteed.

$$Y_j, Y_{f_e}, Z_{kl}, Z_{lm} \in \{0, 1\} \tag{26}$$

$$X_{f_e f'_{e'}}, H_j \ge 0 \tag{27}$$

In regards to encoding scheme, like other presented engineering problems, *RK* technique is used in this sub-section. The encoding plan is illustrated as depicted in Fig. 19.

In order to solve the proposed problem, four metrics are presented. These metrics aim to evaluate the quality of Pareto optimal solutions (Diversification Metric (DM), Spread of Non-dominance Solution

Table 10
Model notation.

Parameters	Description
pc_i	Cost of purchasing raw material from supplier i
fc_{f_e}	Fixed opening cost of facility $f_e \mid e \in \{k, m, n, p, r, s\}$
fc_j	Fixed establishing cost of manufacturing center j.
mc_j	Manufacturing cost of each unit product at manufacturing center j.
vc_{f_e}	Per unit handling cost at facility $f_e \mid e \in \{k, m, n, p, r, s\}$
$tc_{f_e f'_{e'}}$	Per unit transportation cost from facility f_e to facility $f'_{e'}$
ac_{kl}	Per unit cost of assigning customer zone l to distribution center k.
cc_{lm}	Per unit cost collecting EOL products from customer zone l and shipping to collection/inspection center m.
fj_j	The number of fixed job opportunities (i.e. job opportunities which are independent of production capacity like managerial positions) created by establishing manufacturing center j
fj_{fe}	The number of fixed job opportunities created by establishing facility $f_e \mid e \in \{k, m, n, p, r, s\}$
vj_j	The number of variable job opportunities (i.e. job opportunities which vary by production capacity like manufacturing line workers) created through manufacturing at center j.
vj_{fe}	The number of variable job opportunities created through working of facility $f_e \mid e \in \{k, m, n, p, r, s\}$
fl_j	The lost days cost from work's damages at manufacturing center j.
fl_f	The lost days cost from work's damages during the establishment of facility $f_e \mid e \in \{k, m, n, p, r, s\}$
vl_j	The lost of days caused work's damages during the manufacturing at manufacturing center j.
vl_{fe}	The lost of days caused work's damages during the handling of products at facility $f_e \mid e \in \{k, m, n, p, r, s\}$
p_j	Capacity of manufacturing center j
p_{fe}	Capacity of facility $f_e \mid e \in \{k, m, n, p, r, s\}$
max_e	Maximum desired number of established sites in echelon $e \in \{k, m, n, p, r, s\}$
max_j	Maximum desired number of manufacturing center at site j
d_l	Demand of customer zone l.
b_j	The fraction of broken products manufactured at manufacturing center j.
$\propto_l$	The fraction of used products returned from customer zone l.
$\beta^n_m, \beta^p_m, \beta^r_m$ and β^s_m	The fraction of reusable, recoverable, recyclable and scrapped products in collection/inspection center m, respectively ($\beta^n_m + \beta^p_m + \beta^r_m + \beta^s_m = 1$).
γ_n, γ_p and γ_r	The fraction of products shipped from a recovering and remanufacturing and recycling center to used products market, respectively.
sc_d, sc_m and sc_r	Per unit monetary saving resulted from using recovered, remanufactured and recycled EOL products, respectively.
sc_u	Unit selling price of each EOL product in the reuse market.
ε_{jo} and ε_{ld}	The weights given to the elements of social impacts objective: (1) created job opportunities, and (2) worker's lost days, respectively.

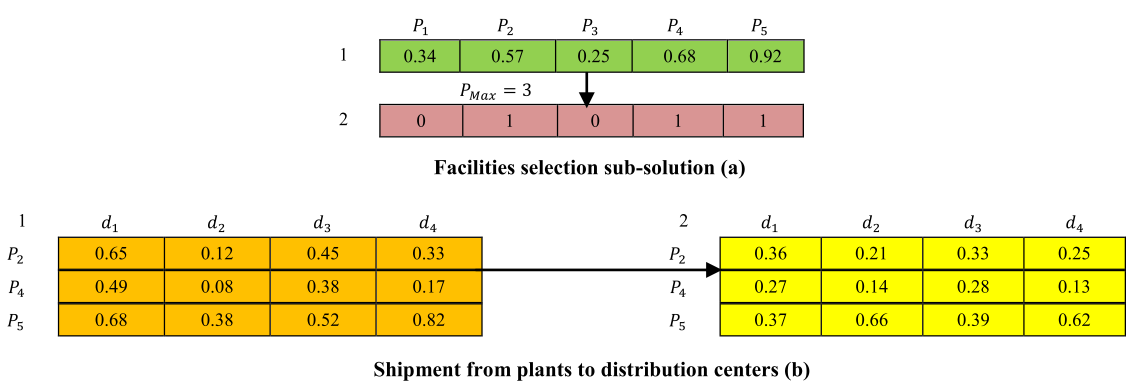


Fig. 19. The graphical illustration of structure solution with two parts, (a) Facilities selection sub-solution. (b) Flow of products sub-solution.

(SNS), Data Envelopment Analysis (DEA) and Percentage of Domination (POD)). The higher value of these metrics brings the better solution quality (Fathollahi Fard et al., 2017). These parameters are presented in recent researches. In this regard, to understand the evaluation metrics and their formulation, these papers are introduced: Behnamian and Fatemi Ghomi (2011); Fathollahi Fard and Hajiaghaei-Keshteli (2018b); Amin and Toloo (2007); Govindan et al. (2015a).

To compare the results based on four versions of SEO, three powerful methods like previous sections are employed: GA, ICA and RDA are utilized to handle this problem. The parameters of algorithms are tuned by Response Surface Method (*RSM*) as a famous technique which is widely applied in a variety of industrial settings and parameter of Optimizers, to select the best values for parameters. This technique is introduced by Box and Wilson. Furthermore, the tuned values of algorithms' parameters have been reported in Supplementary Materials F7. In order to study more about this technique, the interested readers are referred to Fathollahi Fard and Hajiaghaei-Keshteli (2018b).

Moreover, an experiment to analyze the performance of algorithms is designed. Considering the number of potential units in each echelon which directly indicates the complexity of problem, the problems are

Table 11
Model notation.

Variables	
Continues variables:	
$X_{f_e f_e}$	Flow of products from facility f_e to facility f_e
H_j	Amount of products manufactured at manufacturing center j
Binary variables:	
Y_{f_e}	1 if facility f_e|e {k, m, n, p, r, s }is to be established, 0 otherwise
Y_j	1 if manufacturing center j is to be established, 0 otherwise.
Z_{kl}	1 if customer zone l is assigned to distribution center k, 0 otherwise
Z_{lm}	1 if customer zone l is assigned to collection/inspection center m, 0 otherwise.

divided into three levels as seen in Table 12. In order to be fair, in each test problem, the searching time for all methods is based on its size.

Each algorithm is examined for 30 times in all test problems. And finally the best outputs are saved. This study compares the performance and efficiency of all of proposed methods with each other. In this way, first of all, we compute the evaluation metrics (*i.e.* DM, SNS, POD and DEA) based on Pareto optimal solutions for all algorithms. Due to the page limitation of journal, the results have been reported in Supplementary Materials F7.

Fig. 20 illustrates the non-dominated solutions for one run which are obtained by algorithms in P5 as an instance. As depicted in this figure, the number of solutions in best Pareto front is set to 5. As can be seen in Fig. 20, it is evident from the figure that the attainment surfaces for SEO_2 and RDA significantly overlap. Furthermore, to verify the statistical validity of the results, we have performed an analysis of variance (ANOVA) to accurately analyze the results (as seen in Supplementary Materials F7). The results demonstrate that there is a clear statistically significant difference between performances of the algorithms. The means plot and LSD intervals (at the 95% confidence level) for all methods are shown in Fig. 21. In this figure, the algorithms are explored for all of size of problems based on the evaluation metrics (the results in Supplementary Materials F7 are examined by RPD performance metric). The results show the superiority of SEO and the performance of this algorithm in this study. Besides, second version of the proposed method is more successful than other versions.

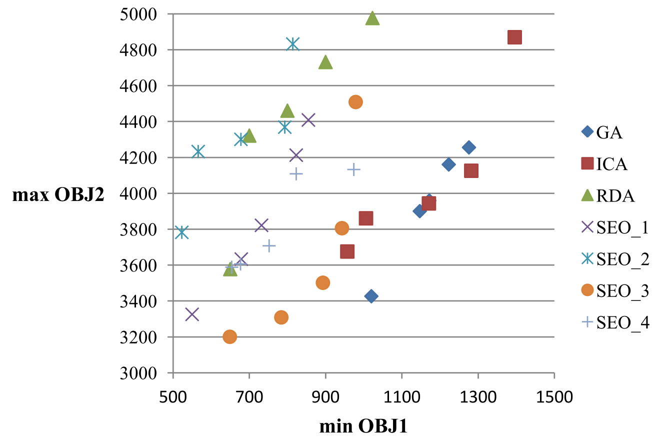


Fig. 20. Dispersion of non-dominated solutions obtained by different algorithms in problem 5.

5. Conclusion and future works

In this work, a new optimization algorithm inspired by Social Engineering is developed. This algorithm is so simple and intelligent. Besides, it makes a balance between the phases by creative ways. This method starts with two random solutions. The better solution is selected as the attacker. During the rules of SE, the attacker reaches the desired goals by obtaining defender and spots an attack. Regarding to design an attack, four techniques are proposed to consider a proper strategy.

To prove the efficiency of four versions of SEO, a series of analyses was conducted. First of all, proposed algorithm is investigated by the twelve benchmark functions to probe the convergence rate as well as some statistical analyses to evaluate the robustness of algorithms. In this item, it was observed and concluded all versions of algorithms are mostly better than considered old and recent algorithms except in a few test problems. The comparison is not only based on equal fitness evaluations but also considers the equal computational of algorithms to do the assessments in a fair environment. In addition, several sensitivity analyses for the critical parameters as well as the exploring the algorithm in high dimensional space. In regards to experiments, it was

Table 12
Level, number and size and also solving time of problems.

Problem levels	Problem numbers	Problem size (*I, J, K, L, M, N, P, R, S*)	Time (s)
Small	P1	(7, 5, 10, 9, 10, 2, 3, 2, 2)	10
	P2	(11, 7, 10, 12, 10, 3, 5, 5, 3)	10
	P3	(15, 9, 12, 15, 12, 4, 7, 6, 4)	10
	P4	(19, 11, 14, 18, 14, 5, 9, 8, 5)	12
	P5	(23, 13, 16, 21, 16, 6, 11, 10, 6)	12
	P6	(27, 15, 18, 24, 18, 7, 13, 10, 7)	12
	P7	(31, 17, 20, 27, 20, 8, 15, 12, 8)	15
	P8	(35, 19, 22, 30, 22, 9, 17, 12, 9)	15
Medium	P9	(55, 29, 31, 51, 31, 14, 19, 18, 16)	40
	P10	(59, 31, 33, 54, 33, 15, 21, 20, 17)	40
	P11	(63, 33, 35, 57, 35, 16, 23, 20, 18)	40
	P12	(67, 35, 37, 60, 37, 17, 25, 22, 19)	45
	P13	(71, 37, 39, 63, 39, 18, 27, 22, 20)	45
	P14	(75, 39, 41, 66, 41, 19, 29, 24, 21)	50
	P15	(79, 41, 43, 69, 43, 20, 31, 24, 22)	50
	P16	(83, 43, 45, 72, 45, 21, 33, 26, 23)	50
Large	P17	(103, 53, 57, 107, 57, 27, 35, 34, 28)	80
	P18	(107, 55, 59, 111, 59, 28, 36, 35, 29)	80
	P19	(111, 57, 61, 115, 61, 29, 37, 37, 30)	85
	P20	(115, 59, 63, 119, 63, 30, 38, 39, 31)	85
	P21	(119, 61, 65, 123, 65, 31, 39, 40, 32)	90
	P22	(123, 63, 67, 127, 67, 32, 40, 42, 33)	90
	P23	(127, 65, 69, 131, 69, 33, 41, 42, 34)	100
	P24	(131, 67, 71, 135, 71, 34, 42, 43, 35)	100

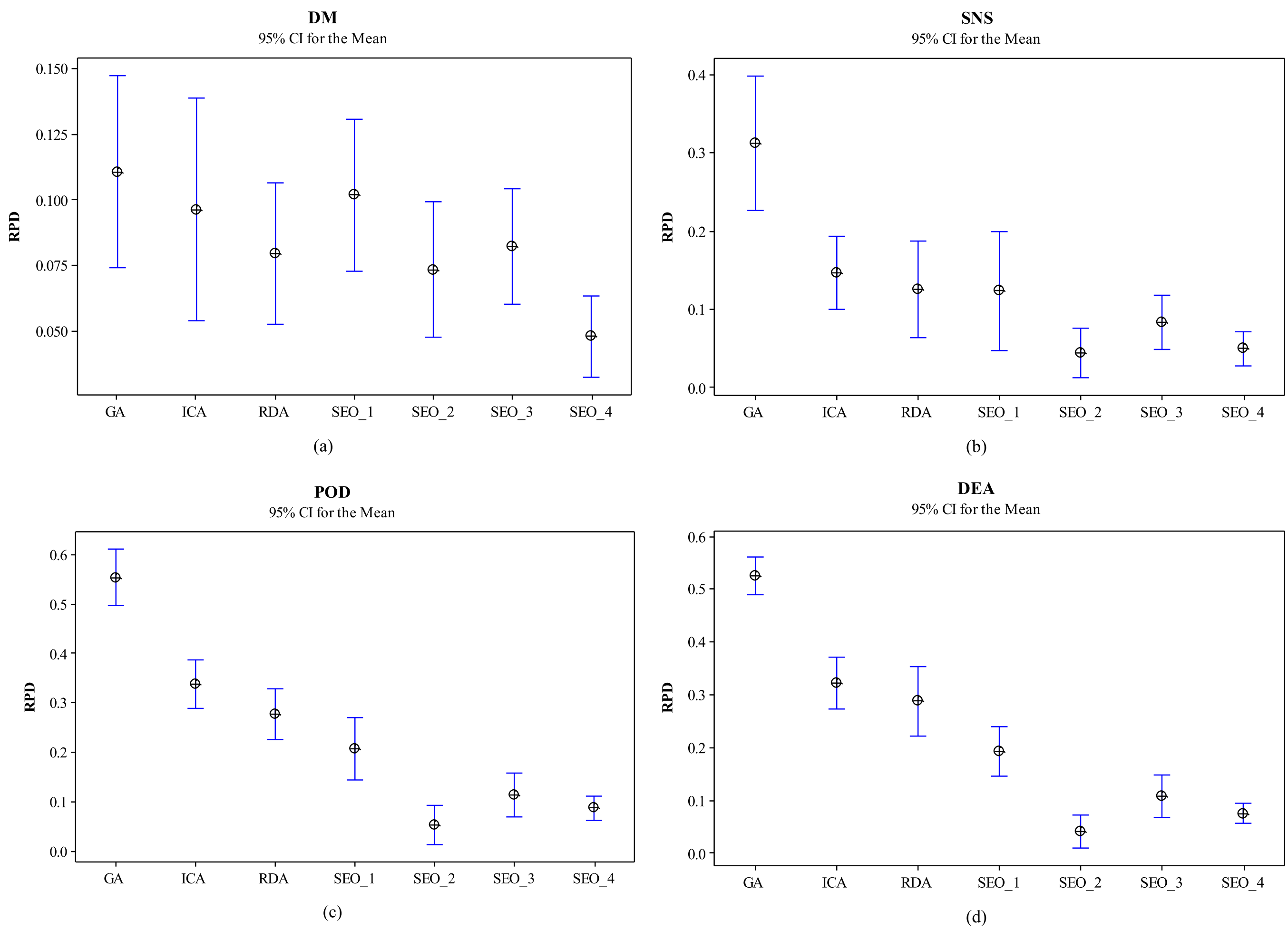


Fig. 21. Means plot and LSD intervals for algorithms based on the RPD for (a) DM, (b) SNS, (c) POD and (d) DEA.

observed and can be concluded that SEO is able to probe the promising search regions of search space and finds an optimal solution. Moreover, the applications of algorithm are applied to solve the multi-objective and well-known engineering problems. The result of experiments shows the performance and efficiency of proposed method in comparison of other well-known and recent metaheuristics.

Moreover, this study opens several research directions for future studies. First of all, more comprehensive analyses on the SEO may still required to be explored. Moreover, we can employ some other real techniques in SE attacks to develop the proposed SEO. In addition, some other real scale optimization problems can be utilized to evaluate the performance of the proposed algorithm.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.engappai.2018.04.009.

References

Altassan, K.M., El-Sherbiny, M.M., Abid, A.D., 2014. Artificial immune algorithm for solving fixed charge transportation problem. Appl. Math. Inf. Sci. 2, 751–759.

Amin, G.R., Toloo, M., 2007. Finding the most efficient DMUs in DEA: An improved integrated model. Comput. Ind. Eng. 52 (1), 71–77.

Askarzadeh, A., 2016. A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput. Struct. 169, 1–12.

Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: an algorithm for Optimizer inspired by imperialistic competition. In: IEEE Congress on Evolutionary Computation, Singapore, pp. 4661–4667.

Baker, B.M., Ayechew, M.A., 2003. A genetic algorithm for the vehicle routing problem. Comput. Oper. Res. 30, 787–800.

Barbarosoglu, G., Ozgur, D., 1999. A Tabu search algorithm for the vehicle routing problem. Comput. Oper. Res. 26 (25), 5–70.

Baykasoğlu, A., Akpinar, Ş, 2015a. Weighted Superposition Attraction (WSA): A Swarm intelligence algorithm for optimization problems–Part 1: Unconstrained optimization. Appl. Soft Comput.

Baykasoğlu, A., Akpinar, Ş., 2015b. Weighted superposition attraction (WSA): A Swarm intelligence algorithm for optimization problems—part 2: constrained optimization. Appl. Soft Comput. 37, 396–415.

Behnamian, J., Fatemi Ghomi, S., 2011. Hybrid flowshop scheduling with machine and resource-dependent processing times. Appl. Math. Model. 35 (3), 1107–1123.

Ben-Yehoshua, Y., Mosheiov, G., 2016. A single machine scheduling problem to minimize total early work. Comput. Oper. Res. 73, 115–118.

Bullnheimer, B., Hartl, R.F., Strauss, C., 1999. An improved ant system algorithm for the vehicle routing problem. Ann. Oper. Res. 89 (3), 19–28.

Calvete, H.I., Galé, C., Iranzo, J.A., 2016. An improved evolutionary algorithm for the two-stage transportation problem with fixed charge at depots. OR Spectrum 38 (1), 189–206.

Chen, L., Peng, J., Zhang, B., 2017. Uncertain goal programming models for bicriteria solid transportation problem. Appl. Soft Comput. 51, 49–59.

Cheng, T.C.E., Sin, C.C.S., 1990. A state-of-the-art review of parallel-machine scheduling research. European J. Oper. Res. 47 (3), 271–292.

Civicioglu, P., 2012. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. Comput. Geosci. 46, 229–247.

Črepinšek, M., Liu, S.H., Mernik, M., 2013. Exploration and exploitation in evolutionary algorithms: A survey. ACM Comput. Surv. 45 (3), 35.

Deb, K., Pratrap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6, 182–197.

Devika, K., Jafarian, A., Nourbakhsh, V., 2014. Designing a sustainable closed-loop supply chain network based on triple bottom line approach: A comparison of metaheuristics hybridization techniques. European J. Oper. Res. 235 (3), 594–615.

Dhiman, G., Kumar, V., 2017. Spotted hyena Optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Adv. Eng. Softw. 114, 48–70.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. The ant system: Optimizer by a colony of cooperating agents. IEEE Trans. Syst. Man. Cybern. B. 26 (1), 29–41.

Draa, A., 2015. On the performances of the flower pollination algorithm–Qualitative and quantitative analyses. Appl. Soft Comput. 34, 349–371.

Ebrahimi, A., Khamehchi, E., 2016. Sperm whale algorithm: an effective metaheuristic algorithm for production Optimizer problems. J. of Natur. Gas Sci. Eng. 29, 211–222.

El-Sherbiny, M.M., Alhamali, R.M., 2013. A hybrid particle Swarm algorithm with artificial immune learning for solving the fixed charge transportation problem. Comput. Ind. Eng. 64 (2), 610–620.

Farmer, J.D., Packard, N.H., Perelson, A.S., 1986. The immune system, adaptation and machine learning. Physica D 22 (1), 187–204.

Fathollahi Fard, A.M., Gholian-Jouybari, F., Paydar, M.M., Hajiaghaei-Keshteli, M., 2017. A bi-objective stochastic closed-loop supply chain network design problem considering downside risk. Ind. Eng. Manag. Sys. 16 (3), 342–362.

Fathollahi Fard, A.M., Hajiaghaei-Keshteli, M., 2016. Red Deer Algorithm (RDA); A New Optimizer Algorithm Inspired by Red Deers' Mating. In: Proceeding in 12th International Conference on Industrial Engineering, ICIE, Tehran, Iran, pp. 34–35.

Fathollahi Fard, A.M., Hajiaghaei-Keshteli, M., 2018a. A tri-level location–allocation model for forward/reverse supply chain. Appl. Soft Comput. 62, 328–346.

Fathollahi Fard, A.M., Hajiaghaei-Keshteli, M., 2018b. A bi-objective partial interdiction problem considering different defensive systems with capacity expansion of facilities under imminent attacks. Appl. Soft Comput. http://dx.doi.org/10.1016/j.asoc.2018.04.011.

Feldman, M., Biskup, D., 2003. Single machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches. Comput. Ind. Eng. 44, 307–323.

Gandomi, A.H., 2014. Interior Search Algorithm (ISA): A novel approach for global Optimizer. ISA Trans. 53, 1168–1183.

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

Geem, Z.W., Kim, J.H., Loganathan, G.V., 2001. A new heuristic Optimizer algorithm: harmony search. Simulation 76 (2), 60–68.

Gendreau, M., Hertz, A., Laporte, G., 1994. A Tabu search heuristic for the vehicle routing problem. Manage. Sci. 40 (12), 76–90.

Ghorbani, N., Babaei, E., 2014. Exchange market algorithm. Appl. Soft Comput. 19, 177–187.

Glover, F., 1977. Heuristics for integer programming using surrogate constraints. Decision Sci. 8 (1), 156–166.

Glover, F., McMillan, C., 1986. The general employee scheduling problem. An integration of MS and AI. Comput. Oper. Res. 13 (5), 563–573.

Golmohamadi, S., Tavakkoli-Moghaddam, R., Hajiaghaei-Keshteli, M., 2017. Solving a fuzzy fixed charge solid transportation problem using batch transferring by new approaches in meta-heuristic. Electron. Notes Discrete Math. 58, 143–150.

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.

Govindan, K., Jafarian, A., Nourbakhsh, V., 2015a. Bi-objective integrating sustainable order allocation and sustainable supply chain network strategic design with stochastic demand using a novel robust hybrid multi-objective metaheuristic. Comput. Oper. Res. 62, 112–130.

Govindan, K., Soleimani, H., 2017. A review of reverse logistics and closed-loop supply chains: a journal of cleaner production focus. J. Cleaner Prod. 142, 371–384.

Govindan, K., Soleimani, H., Kannan, D., 2015b. Reverse logistics and closed-loop supply chain: A comprehensive review to explore the future. European J. Oper. Res. 240 (3), 603–626.

Granger, S., 2001. Social Engineering Fundamentals, Part I: Hacker Tactics. In: Security Focus. December 18.

Granger, S., 2002. Social Engineering Fundamentals, Part II: Combat Strategies. In: Security Focus. January 9.

Hadad, O.B., Afshar, A., Marino, M.A., 2006. Honey-bees Optimizer (HBMO) algorithm: a new heuristic approach for water resources Optimizer. Water Resour. Manag. 20 (5), 661–680.

Hajiaghaei-Keshteli, M., Aminnayeri, M., 2013. Keshtel Algorithm (KA); a new Optimizer algorithm inspired by Keshtels' feeding. Proc. IEEE Conf. Ind. Eng. Manag. Syst. 2249–2253.

Hajiaghaei-Keshteli, M., Aminnayeri, M., 2014. Solving the integrated scheduling of production rail transportation problem by Keshtel algorithm. Appl. Soft Comput. 25, 184–203.

Hajiaghaei-Keshteli, M., Aminnayeri, M., Ghomi, S.F., 2014. Integrated scheduling of production and rail transportation. Comput. Ind. Eng. 74, 240–256.

Hajiaghaei-Keshteli, M., Fathollahi Fard, A.M., 2018. Sustainable closed-loop supply chain network design with discount supposition. Neural Comput. Appl. http://dx.doi.org/10.1007/s00521-018-3369-5.

Hajiaghaei-Keshteli, M., Molla-Alizadeh-Zavardehi, S., Tavakkoli-Moghaddam, R., 2010. Addressing a nonlinear fixed-charge transportation problem using a spanning tree-based genetic algorithm. Comput. Ind. Eng. 59, 259–271.

Hajiaghaei-Keshteli, M., Sajadifar, S.M., Haji, R., 2011. Determination of the economical policy of a three-echelon inventory system with (R, Q) ordering policy and information sharing. Internat. J. Adv. Manuf. Technol. 55 (5–8), 831–841.

Hall, N., Kubaik, G.W., Sethi, S.P., 1991. Earliness-tradiness scheduling problems, II: deviation of completion times about a restrictive common due date. Oper. Res. 39, 847–856.

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

Hiquebran, D.T., Alfa, A.S., Shapiro, D.H., Gittoes, 1994. A revised simulated annealing and cluster-2rst route-second algorithm applied to the vehicle routing problem. Eng. Optim. 22 (77), 107.

Hirsch, W.M., Dantzig, G.B., 1968. The fixed charge problem. Naval Res. Logist. Q. 15, 413–424.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. University of Michigan Press, Michigan, Ann Arbor.

Hoogeveen, J.A., van de Velde, S.L., 1991. Scheduling around small common due date. European J. Oper. Res. 55, 237–242.

Hussain, K., Salleh, M.N.M., Cheng, S., Shi, Y., 2018. Metaheuristic research: a comprehensive survey. Artif. Intell. Rev. 1–43.

Husseinzadeh Kashan, A., 2009. League championship algorithm: a new algorithm for numerical function Optimizer. In: Soft Computing and Pattern Recognition, 2009, SOCPARâĂŹ09. International Conference of (2009, December), IEEE, pp. 43–48.

Husseinzadeh Kashan, A., 2015. A new metaheuristic for Optimizer: Optics inspired Optimizer (OIO). Comput. Oper. Res. 55, 99–125.

James, R.J.W., 1997. Using Tabu search to solve the common due date early-tardy machine scheduling problem. Comput. Oper. Res. 24, 199–208.

Kar, A.K., 2016. Bio-inspired computing-A review of algorithms and scope of applications. Expert Syst. Appl. 59, 20–32.

Karaboga, D., 2005. An idea based on honey bee Swarm for numerical Optimizer. In: Technical Report-TR06. Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboga, D., Akay, B., 2009. A comparative study of artificial bee colony algorithm. Appl. Math. Comput. 214, 108–132.

Kaveh, A., Farhoudi, N., 2013. A new optimization method: dolphin echolocation. Adv. Eng. Softw. 59, 53–70.

Kaveh, A., Khayatazad, M., 2012. A new meta-heuristic method: ray optimization. Comput. Struct. 112, 283–294.

Kennedy, J., Eberhart, R., 1995. Particle Swarm Optimizer. In: Proc. IEEE Int. Conf. Neural Networks, pp.1942–1945.

Kirkpatrick, S., Gelatto, C.D., Vecchi, M.P., 1983. Optimizer by simulated annealing. Science 220, 671–680.

Klose, A., 2008. Algorithms for solving single-sink fixed-charge transportation problem. Comput. Oper. Res. 35, 2079–2092.

Krishnand, K.N., Ghose, D., 2005. Detection of multiple source locations using a glowworms metaphor with applications to collective robotics. In: Proceeding of the IEEE Swarm Intelligence Symposium, pp. 84–91.

Lalla-Ruiz, E., Expósito-Izquierdo, C., Taheripour, S., Voß, S., 2016. An improved formulation for the multi-depot open vehicle routing problem. OR Spectrum 38 (1), 175–187.

Laporte, G., 1992. The vehicle routing problem: an overview of exact and approximate algorithms. European J. Oper. Res. 59 (3), 45–58.

Lee, C.Y., Kim, S.J., 1995. Parallel genetic algorithms for the earliness–tardiness job scheduling problem with general penalty weights. Comput. Ind. Eng. 28, 231–243.

Li, M.D., Zhao, H., Weng, X.W., Han, T., 2016. A novel nature-inspired algorithm for Optimizer: Virus colony search. Adv. Eng. Softw. 92, 65–88.

Liang, J.J., Suganthan, P.N., 2005. Dynamic multi-swarm particle Swarm Optimizer with local search. In: The IEEE Congress on Evolutionary Computation, pp. 522–528.

Liaw, C.F., 2016. A branch-and-bound algorithm for identical parallel machine total tardiness scheduling problem with preemption. J. Ind. Prod. Eng. 1–9.

Liu, S.H., Mernik, M., Hrnčič, D., Črepinšek, M., 2013. A parameter control method of evolutionary algorithms using exploration and exploitation measures with a practical application for fitting Sovova's mass transfer model. Appl. Soft Comput. 13 (9), 3792–3805.

Lotfi, M.M., Tavakkoli-Moghaddam, R., 2013. A genetic algorithm using priority-based encoding with new operators for fixed charge transportation problems. Appl. Soft Comput. 13, 2711–2726.

Luo, X.R., Brody, R., Seazzu, A., Burd, S., 2013. Social engineering: The neglected human factor for managing information resources and technology. Emerging Appl. Theor. 151.

Meng, X., Liu, Y., Gao, X., Zhang, H., 2014. A new bio-inspired: Chicken Swarm Optimizer. Advances in Swarm Intelligence. Springer International Publishing, pp. 86–94.

Mernik, M., Liu, S.H., Karaboga, D., Črepinšek, M., 2015. On clarifying misconceptions when comparing variants of the artificial bee colony algorithm by offering a new implementation. Inform. Sci. 291, 115–127.

Mirjalili, S., 2015c. The ant lion Optimizer. Adv. Eng. Softw. 83, 80–98.

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.

Mirjalili, S., Lewis, A., 2015. Novel performance metrics for robust multi-objective Optimizer algorithms. Swarm Evolution Computing 21, 1–23.

Mirjalili, S., Lewis, A., 2016. The whale Optimizer algorithm. Adv. Eng. Softw. 95, 51–67.

Mirjalili, S., Mirjalili, S.M., Hatamlou, A., 2015. Multi-verse Optimizer: a nature-inspired algorithm for global Optimizer. Neural Comput. Appl. 1–19.

Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf Optimizer. Adv. Eng. Softw. 69, 46–61.
Mladenović, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24 (11), 1097–1100.
Molla-Alizadeh-Zavardehi, S., Hajiaghaei-Keshteli, M., Tavakkoli-Moghaddam, R., 2011. "Solving a capacitated fixed-charge transportation problem by artificial immune and genetic algorithms with a Prüfer number representation. Expert Syst. Appl. 38 (8), 10462–10474.
Molla-Alizadeh-Zavardehi, S., Sadi Nezhad, S., Tavakkoli-Moghaddam, R., Yazdani, M., 2013. Solving a fuzzy fixed charge solid transportation problem by metaheuristics. Math. Comput. Modelling 57, 1543–1558.
Moscato, P., 1989. On evolution search Optimizer genetic algorithms and martial arts: Towards memetic algorithms. In: Caltech Concurrent Computation Program, C3P Report 826.
Mucherino, A., Seref, O., 2007. Monkey search: a novel metaheuristic search for global Optimizer. Am. Inst. Phys. 953 (162).
Nakrani, S., Tovey, C., 2004. On honey bees and dynamic server allocation in internet hosting centers. Adapt. Behav. 12 (3–4), 223–240.
Nazeri-Shirkouhi, S., Eivazy, H., Ghodsi, R., Rezaie, K., Atashpaz-Gargari, E., 2010. Solving the integrated product mix outsourcing problem using the Imperialist Competitive Algorithm. Expert Syst. Appl. 37, 7615–7626.
Osman, I.H., 1993. Meta strategy simulated annealing and Tabu search algorithms for the vehicle routing problem. Oper. Res. 41 (4), 21–51.
Price, K.V., Storn, R., 1997. Differential evolution: a simple evolution strategy for fast Optimizer. Dr. Dobb's J. 22 (4), 18–24.
Rao, V., Patel, V., 2013. Comparative performance of an elitist teaching-learning-based Optimizer algorithm for solving unconstrained Optimizer problems. Inter. J. Ind. Eng. Comput. 4 (1), 29–50.
Rao, V., Savsani, V.J., Vakharia, D.P., 2011. Teaching-learning-based Optimizer: a novel method for constrained mechanical design Optimizer problems. Comput. Aided Des. 43 (3), 303–315.
Rashidi, E., Nezamabadi, H., Saryazdi, S., 2009. GSA: a gravitional search algorithm. Inf. Sci. 179 (13), 2232–2248.
Rego, C., Roucairol, C., 1996. Aparallel Tabu search algorithm using ejection chains for the vehicle routing problem. In: Osman, I., Kelly, J. (Eds.), Meta-Heuristics: Theory and Applications. Kluwer, Boston.
Rubinstein, R.Y., 1997. Optimizer of computer simulation models with rare events. European J. Oper. Res. 99 (1), 89–112.
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. 1–21.
Samadi, A., Mehranfar, N., Fathollahi Fard, A.M., Hajiaghaei-Keshteli, M., 2018. Heuristic-based metaheuristics to address a sustainable supply chain network design problem. J. Ind. Prod. Eng. 35 (2), 102–117.
Sarasola, B., Doerner, K.F., Schmid, V., Alba, E., 2016. Variable neighborhood search for the stochastic and dynamic vehicle routing problem. Ann. Oper. Res. 236 (2), 425–461.
Saremi, S., Mirjalili, S., Lewis, A., 2017. Grasshopper optimisation algorithm: Theory and application. Adv. Eng. Softw. 105, 30–47.
Saskena, J.P., 1970. Mathematical model of scheduling clients through welfare agencies. J. Canadian Oper. Res. Soc. 8, 185–200.
Shah-Hosseini, H., 2011. Principal components analysis by the galaxy-based search algorithm: a novel metaheurisitc for continuous Optimizer. Inter. J. Comput. Sci. Eng. 6 (1), 132–140.
Sörensen, K., 2015. Metaheuristics — the metaphor exposed. Int. Trans. Oper. Res. 22 (1), 3–18.
Taguchi, G., 2000. Introduction to quality engineering. In: White Plains, Asian Productivity Organization. UNIPUB.
Tamura, K., Yasuda, K., 2011. Spiral dynamic inspired Optimizer. J. Adv. Comput. Intell. Inf. 15 (8), 1116–1122.
Tanabe, R., Fukunaga, A.S., 2014. Improving the search performance of SHADE using linear population size reduction. In: Evolutionary Computation, CEC, 2014 IEEE Congress on. IEEE, pp. 1658–1665.
Van der Laan, E., Salomon, M., Dekker, R., 1999. An investigation of lead-time effects in manufacturing/remanufacturing systems under simple PUSH and PULL control strategies. European J. Oper. Res. 115 (1), 195–214.
Wang, G.G., Deb, S., Coelho, L.D.S., 2015. Elephant herding optimization. In: Computational and Business Intelligence. ISCBI, 2015 3rd International Symposium on. IEEE, pp. 1–5.
Weinberg, A.M., 1966. Can technology replace social engineering? Bull. At. Sci. 22 (10), 4–8.
Wolpert, D.H., Macready, W.G., 1997. No free lunch theorems for optimization. IEEE Trans. Evol. Comput. 1 (1), 67–82.
Xie, F., Jia, R., 2012. Nonlinear fixed charge transportation problem by minimum cost flow-based genetic algorithm. Comput. Ind. Eng. 63, 763–778.
Yang, X.S., 2008. Nature-Inspired Metaheurisitc Algorithm. Luniver Press.
Yang, X.S., 2010. A new metaheurisitc bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimizer. pp. 65–74.
Yang, X.S., Deb, S. 2009. Cuckoo search via Levy lights. In: World Congress on Nature & Biologically inspired Computing, NaBIC, pp. 210–214.
Zhao, C., Hsu, C.J., Wu, W.H., Cheng, S.R., Wu, C.C., 2016. Note on a unified approach to the single-machine scheduling problem with a deterioration effect and convex resource allocation. J. Manuf. Syst. 38, 134–140.
Zhao, S.Z., Suganthan, P.N., Das, S. 2010. Self-adaptive differential evolution with modified multi-trajectory search for CEC'2010 large scale Optimizer. In: Proceedings of International Conference On Swarm, Evolutionary and Memetic Computing, SEMCCO, pp. 1–10.
Zheng, Y.-J., 2015. Water wave Optimizer: A new nature-inspired metaheuristic. Comput. Oper. Res. 55, 1–11.
Zhou, Y., Lou, Q., Xie, J., Zheng, H., 2016. A hybrid bat algorithm with path reliking for the capacitated vehicle routing problem. In: Metaheuristics and Optimizer in Civil Engineering. Springer International Publishing, pp. 255–276.