Research Article GU J Sci 32(3): 871-882 (2019) DOI: 10.35378/gujs.484643

Gazi University

Journal of Science

http://dergipark.gov.tr/gujs

# DGO: Dice Game Optimizer

Mohammad DEHGHANI[1,*], Zeinab MONTAZERI[1], Om Parkash MALIK[2]

[1]*Shiraz University of Technology, Dept. of Electrical and Electronics Engineering , Shiraz, Iran*

[2]*University of Calgary, Dept. of Electrical Engineering, Calgary Alberta Canada*

## Highlights

- The paper focuses on proposing a new optimization algorithm called Dice Game Optimizer (DGO).
- DGO can be used to solve optimization constrained and unconstrained problems in different sciences.
- The proposed DGO algorithm is tested on 23 standard benchmark test functions.
- The performance of DGO is also examined on one engineering design problem (Pressure vessel design).
- The results show the merits of the DGO algorithm compared to the existing algorithms.

**Article Info**

*Received: 18/11/2018*
*Accepted: 02/04/2019*

**Keywords**

*Metaheuristic algorithms*
*Optimization*
*Game*
*Dice game*
*Dice Game Optimizer*

**Abstract**

In recent years, optimization algorithms have been used in many applications. Most of these algorithms are inspired by physical processes or living beings' behaviors. This article suggests a new optimization method called "Dice Gaming Optimizer" (DGO), which simulates dice gaming laws. This algorithm is inspired by an old game and the searchers are a set of players. Each player moves in the playground based on at least one and maximum six different players called guide's players. The number of guide's players for each player is determined by the number of dice. DGO is tested on 23 standard benchmark test functions and also compared with eight other algorithms such as: Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Cuckoo Search (CS), Ant-Lion Optimizer (ALO), Grey Wolf Optimizer (GWO), Grasshopper Optimization Algorithm and Emperor Penguin Optimizer (EPO). Moreover, a real-life engineering design problem is solved by DGO. The results indicate that DGO have better performance as compared to the other well-known optimization algorithms.

## 1. INTRODUCTION

In recent years, meta-heuristic algorithms have been used instead of a comprehensive search of the problem space [1-6]. Metaheuristic algorithms have proven high functionality in many fields, such as protection [7], power engineering [8], electronics [9, 10], etc. Metaheuristic search algorithms are inspired by physical, biological, and nature processes, and most of them act in a population-based environment. Unlike classical methods, metaheuristic search algorithms are random-based and perform space searches in parallel. Another difference is the lack of use of gradient information. These types of methods use only the fitness function to conduct search, but because of the swarm intelligence, they are able to discover the answer. Swarm intelligence appears in cases where a population of non-expert agents exists. Each of these agents, under certain conditions, exhibits simple behavior and also interact with each other. Membership interactions cause unexpected external effects, and ultimately, the entire set can detect a solution without a central controller [11, 12].

Heuristic algorithms are categorized into four categories: Physics-based, Swarm-based, Evolutionary-based and Game-based algorithms.

*Corresponding author, e-mail: adanbax@gmail.com

### 1.1. Physics-Based Algorithms

The basis for designing these algorithms is the laws of physics [13]. For example Simulated Annealing (SA) [2] is based on the process of cooling metals during metalworking. Gravitational Search Algorithm (GSA) [14] is a physics-based legalization method. The process of this algorithm is based on the gravitational force between objects. Water Cycle algorithm (WCA) [15] proposes an optimization model by simulating the natural event of the water cycle. Spring Search Algorithm (SSA) [6] Inspired by Hooke's law. In SSA search agents are group of weights which connected together with springs[16]. Some of the other this category algorithms are: Galaxy-based Search Algorithm (GbSA) [17], Curved Space Optimization (CSO) [18], Artificial Chemical Reaction Optimization Algorithm (ACROA) [19], Small World Optimization Algorithm (SWOA) [20] and Central Force Optimization (CFO) [21].

### 1.2. Swarm-Based Algorithms

One of the popular swarm-based algorithms is Particle Swarm Optimization (PSO) [22] that derived from the social behavior of the bird group during migration. Grey Wolf Optimizer (GWO) [23] is based on the collective life of gray wolfs. Grasshopper Optimization Algorithm (GOA) [24] simulates the behavior of grasshopper as an optimization process. Emperor Penguin Optimizer (EPO) [25] is based on the imitation of the emperor's penguins. Ant Colony (AC) [26] simulates the behavior of ants when searching for food. Some of the other this category algorithms are: Artificial Bee Colony (ABC) [27], Cuckoo Search (CS) [28], Ant Lion Optimizer (ALO) [29], Bat inspired Algorithm (BA) [30], Whale Optimization Algorithm (WOA) [31] and Spotted Hyena Optimizer (SHO) [32].

### 1.3. Evolutionary-Based Algorithms

Evolutionary-based algorithms combines aspects of natural selection and continuity of coordination. Genetic Algorithm (GA) [33] derived from the genetic law and reincarnation is based on the theory of Darwinian evolution. Some of the other this category algorithms are: Differential Evolution (DE) [34], Evolution Strategy (ES) [35], Genetic Programming (GP) [36], and Biogeography-based Optimizer (BBO) [37].

### 1.4. Game-Based Algorithms

These algorithms is developed based on the rules of various games. Orientation Search Algorithm (OSA) [38] is inspired by the rules of the orientation game. In this game, players moves in the orientation of the referee’s hand.

In this paper, the use of a game called dice game is being studied for the design of the optimization algorithm. An optimizer called the dice game optimizer (DGO) is designed using the rules governing the game and the impact of the players in the game from each other.

The dice game is introduced in Section 2 and the dice game optimizer is described in Section 3. The results are presented in Section 4. Finally, the conclusions are given in Section 5.

## 2. DICE GAME

The dice game, an old game, may be described as follows: Players are in the playing field. The game space consists of squares, each of which has a certain score. Each player is placed in one of these squares. Maybe even a few players together in a square. Each player has two attributes in this game: location and score. Players do not have any information about the scores of each other. At each stage of the game, each player may be connected to at least one player and at most six other players, called guide’s players. The number of guide’s players is determined for each player by throwing a dice. After this step, each player will be informed of the guide’s player's score. Then, each player moves by comparing its score and the score of the guide's players. At each stage, the player with the highest score is encouraged. The game continues until players compete with each other. In the end, after several repetitions, the winner of the game is determined.

## 3. DICE GAME OPTIMIZATION (DGO)

With the simulation of the dice game above, an optimization algorithm called dice game optimizer is designed in this section. DGO is described in two steps:

- Formation of the system, laws and parameters setting,
- The passage of time and update the parameters.

### 3.1. Formation of the System, Laws and Parameters Setting

First, the system's space is determined. The search agents are a set of players. Each player has a position and a score.

Now imagine the system as a set of *m* players. The initial position of the players is created randomly on the playing field (problem definition space).

In equation (1), the position '$d$' of player '$i$' is shown as $x_i^d$.

$$X_i = (x_i^1, \dots, x_i^d, \dots, x_i^n) \tag{1}$$

After the formation of the system, the rules are specified. Players compete with each other to determine the winner according to the rules of the game.

- Calculation of each player's score

In order to simulate the score of each player, a fitness function is used. It assigns higher score to the player with a better location. This parameter is computed according to equation (2)

$$Score_i = \frac{fit_i - fit\,(player_{worst})}{\sum_{j=1}^{N} fit_j - fit\,(player_{worst})}. \tag{2}$$

Here, $Score_i$, is the score of player *I*, $fit_i$, is the value of the fitness function, *N* is the number of players, $player_{best}$ is the position of the best player and $player_{worst}$ is the position of the worst player. These positions are indicated in equations (3) and (4)

$$player_{best} = location\ of\ \min(fit_j)\ \ \&\ \ j \in \{1{:}N\}, \tag{3}$$

$$player_{worst} = location\ of\ \max(fit_j)\ \ \&\ \ j \in \{1{:}N\}, \tag{4}$$

- Tossing dice for each player

At this stage of the game, each player tosses a dice once. A dice number is a discrete number between 1 and 6 that represents the number of player's guide of each player. The number of dice for each player is specified in equation (5)

$$Dice_i = K\ \&\ \ K \in [1\ 2\ 3\ 4\ 5\ 6] \tag{5}$$

$Dice_i$, is the dice number for *i*-th player. This number is specified by K.

- Selection of the Guide's players for each player

For each player, based on the number of dice (K), guide players are selected randomly among the players. These players are specified in equation (6)

$$X_{Guide_i}^{k} = X_1 : X_K. \tag{6}$$

Here $X_{Guide_i}^{k}$, is the position guide player number *k* of player '$i$'.

- Update of the position of each player

Now, $X^{i,d}$ is calculated by equation (7)

$$X^{i,d} = X_0^{i,d} + \sum_{k=1}^{Dice_i} \left( r_k \left( X^{i,d} - X_{Guide_i}^{k,d} \right) sign\left( Score_i - Score_{Guide_k} \right) \right), \tag{7}$$

where $r_k$, is the random number with normal distribution in the interval $[0-1]$ and $Score_{Guide_k}$, is the score of guide player number *k*.

### 3.2. The passage of Time and Update the Parameters

At first, each player stays randomly at a point of the game space, which is the answer to the problem. At each moment of time, players' positions are evaluated. Then the position of each player is updated after the calculation of Equations 1 to 7. The flowchart of DGO shown in Figure 1.

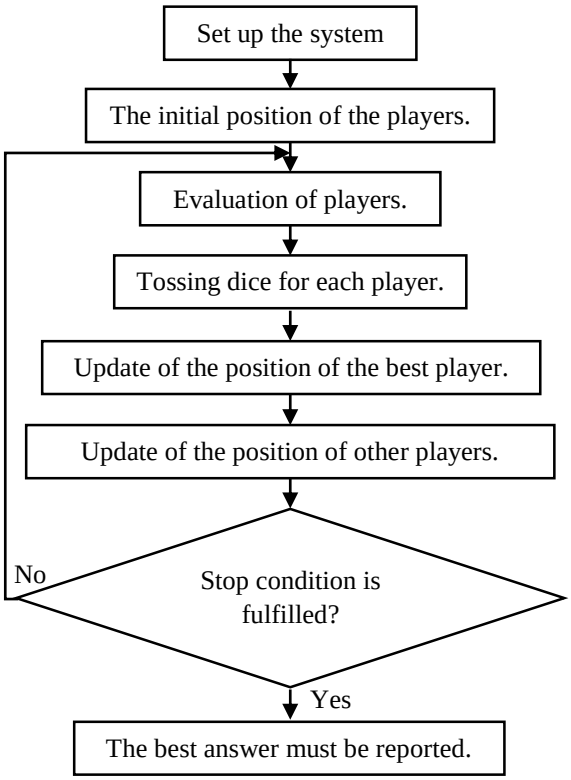


***Figure 1.*** *Flowchart of DGO*

## 4. RESULT

### 4.1. Standard Benchmark Test Functions

Performance of the proposed algorithm is evaluated by applying the standard benchmark test functions shown in Tables 1-3 [39].

***Table 1.*** *Unimodal test functions*

| | |
|---|---|
| $F_1(x) = \sum_{i=1}^{m} x_i^2$ | $[-100,100]^m$ |
| $F_2(x) = \sum_{i=1}^{m} \lvert x_i \rvert + \prod_{i=1}^{m} \lvert x_i \rvert$ | $[-10,10]^m$ |
| $F_3(x) = \sum_{i=1}^{m} \left( \sum_{j=1}^{i} x_i \right)^2$ | $[-100,100]^m$ |
| $F_4(x) = \max\{ \lvert x_i \rvert , \quad 1 \le i \le m \}$ | $[-100,100]^m$ |
| $F_5(x) = \sum_{i=1}^{m-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)]$ | $[-30,30]^m$ |
| $F_6(x) = \sum_{i=1}^{m} ([x_i + 0.5])^2$ | $[-100,100]^m$ |
| $F_7(x) = \sum_{i=1}^{m} i x_i^4 + random(0,1)$ | $[-1.28,1.28]^m$ |

***Table 2.*** *Multimodal test functions*

| | |
|---|---|
| $F_8(x) = \sum_{i=1}^{m} -x_i \sin(\sqrt{\lvert x_i \rvert})$ | $[-500,500]^m$ |
| $F_9(x) = \sum_{i=1}^{m} [\, x_i^2 - 10\cos(2\pi x_i) + 10]$ | $[-5.12,5.12]^m$ |
| $F_{10}(x) = -20\exp\left( -0.2\sqrt{\frac{1}{m}\sum_{i=1}^{m} x_i^2} \right) - \exp\left( \frac{1}{m}\sum_{i=1}^{m} cos(2\pi x_i) \right) + 20 + e$ | $[-32,32]^m$ |
| $F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{m} x_i^2 - \prod_{i=1}^{m} cos\left( \frac{x_i}{\sqrt{i}} \right) + 1$ | $[-600,600]^m$ |
| $F_{12}(x) = \frac{\pi}{m}\left\{ 10\sin(\pi y_1) + \sum_{i=1}^{m} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} + \sum_{i=1}^{m} u(x_i, 10,100,4)$ <br> $u(x_i, a, i, n) = \begin{cases} k(x_i - a)^n & x_i > -a \\ 0 & -a < x_i < a \\ k(-x_i - a)^n & x_i < -a \end{cases}$ | $[-50,50]^m$ |
| $F_{13}(x) = 0.1\left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{m} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_m)] \right\} + \sum_{i=1}^{m} u(x_i, 5,100,4)$ | $[-50,50]^m$ |

***Table 3.*** *Multimodal test functions with fixed dimension*

| | |
|---|---|
| $F_{14}(x) = \left( \frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$ | $[-65.53,65.53]^2$ |
| $F_{15}(x) = \sum_{i=1}^{11} \left[ a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$ | $[-5,5]^4$ |
| $F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$ | $[-5,5]^2$ |
| $F_{17}(x) = \left( x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6 \right)^2 + 10\left( 1 - \frac{1}{8\pi} \right) cos x_1 + 10$ | [-5,10] × [0,15] |
| $F_{18}(x) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$ | $[-5,5]^2$ |
| $F_{19}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - P_{ij})^2)$ | $[0,1]^3$ |
| $F_{20}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - P_{ij})^2)$ | $[0,1]^6$ |
| $F_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + 6c_i]^{-1}$ | $[0,10]^4$ |
| $F_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + 6c_i]^{-1}$ | $[0,10]^4$ |
| $F_{23}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + 6c_i]^{-1}$ | $[0,10]^4$ |

### 4.2. Algorithms Used for Comparison

In order to evaluate the performance of DGO, its performance has been compared with eight optimization algorithms: Genetic Algorithm (GA) [40], Particle Swarm Optimization (PSO) [41], Artificial Bee Colony (ABC) [27], Cuckoo Search (CS) [28], Ant Lion Optimizer (ALO) [29], Grey Wolf Optimizer (GWO) [23], Grasshopper Optimization Algorithm (GOA) [24], Emperor Penguin Optimizer (EPO) [25].

### 4.3. Performance Comparison

#### 4.3.1. Unimodal Test Functions with High Dimension

Functions F1 to F7 are Unimodal test functions. The average results obtained during 20 times the independent implementation of the algorithms are presented in Table 4. These results indicate that DGO performs better than other algorithms in all F1 to F7 functions.

#### 4.3.2. Multimodal Test Functions with High Dimension

In multimodal test functions, the number of local responses increases exponentially with increasing function dimensions. Therefore, it is hardly possible to achieve the minimum answer in this type of functions. In this type of function, reaching the nearest answer indicates the high power of the algorithm in passing the wrong local answers. The results of evaluating functions F8 to F13 for 20 independent runtimes are shown in Table 5. In all of these functions, DGO has a better performance.

#### 4.3.3. Multimodal Test Functions with Low Dimension

Functions F14 to F23 have a low number of dimensions and also low local answers. The results of the 20-time implementation of DGO and other algorithms are presented in Table 6. These results show that DGO performs effectively for these types of functions also.

***Table 4.*** *Results for DGO and other algorithms in Unimodal test functions*

| | | GA | PSO | ABC | CS | ALO | GWO | GOA | EPO | DGO |
|---|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | Ave | 9.75E-13 | 2.49E-09 | 3.93E-10 | 1.58E-01 | 3.94E-10 | 4.61E-23 | 7.86E-10 | 5.71E-28 | 6.74E-35 |
| | std | 1.01E-11 | 7.01E-09 | 4.06E-09 | 1.09E-01 | 4.07E-09 | 7.37E-23 | 8.11E-09 | 8.31E-29 | 9.17E-36 |
| $F_2$ | Ave | 3.27E-18 | 3.65E-04 | 3.00E-20 | 1.98E-01 | 3.29E-18 | 1.20E-34 | 5.99E-20 | 6.20E-40 | 7.78E-45 |
| | std | 2.55E-17 | 9.20E-04 | 5.55E-18 | 7.05E-02 | 3.11E-17 | 1.30E-34 | 1.11E-17 | 3.32E-40 | 3.48E-45 |
| $F_3$ | Ave | 3.85E-10 | 7.00E+00 | 4.60E-05 | 2.48E+03 | 4.60E-05 | 1.00E-14 | 9.19E-05 | 2.05E-19 | 2.63E-25 |
| | std | 3.68E-09 | 3.57E+00 | 3.08E-04 | 1.95E+03 | 3.08E-04 | 4.10E-14 | 6.16E-04 | 9.17E-20 | 9.83E-27 |
| $F_4$ | Ave | 4.59E+01 | 4.62E+01 | 4.37E-01 | 9.79E+00 | 4.63E+01 | 2.02E-14 | 8.73E-01 | 4.32E-18 | 4.65E-26 |
| | std | 2.84E+01 | 2.84E+01 | 5.95E-02 | 4.23E+00 | 2.84E+01 | 2.43E-14 | 1.19E-01 | 3.98E-19 | 4.68E-29 |
| $F_5$ | Ave | 2.92E+02 | 3.03E+02 | 4.59E+02 | 4.28E+02 | 7.24E+02 | 2.79E+01 | 8.91E+02 | 5.07E+00 | 5.41E-01 |
| | std | 2.17E+01 | 4.03E+01 | 1.49E+02 | 1.06E+03 | 1.69E+02 | 1.84E+00 | 2.97E+02 | 4.90E-01 | 5.05E-02 |
| $F_6$ | Ave | 4.87E-01 | 1.58E-01 | 3.29E-01 | 2.60E+00 | 1.58E-01 | 6.58E-01 | 8.18E-17 | 7.01E-19 | 8.03E-24 |
| | std | 2.19E-01 | 4.99E-02 | 1.69E-01 | 5.37E-01 | 4.99E-02 | 3.38E-01 | 1.70E-18 | 4.39E-20 | 5.22E-26 |
| $F_7$ | Ave | 7.30E-04 | 3.49E-02 | 2.69E-01 | 2.95E-02 | 2.69E-01 | 7.80E-04 | 5.37E-01 | 2.71E-05 | 3.33E-08 |
| | std | 1.84E-03 | 1.60E-02 | 9.47E-02 | 3.27E-02 | 9.61E-02 | 3.85E-04 | 1.89E-01 | 9.26E-06 | 1.18E-06 |

***Table 5.*** *Results for DGO and other algorithms in Multimodal test functions*

| | | GA | PSO | ABC | CS | ALO | GWO | GOA | EPO | DGO |
|---|---|---|---|---|---|---|---|---|---|---|
| $F_8$ | Ave | -5.06E+02 | -5.06E+02 | -3.28E+02 | -3.28E+02 | -3.30E+02 | -6.14E+02 | -4.69E+01 | -8.76E+02 | -1.2E+04 |
| | std | 4.33E+01 | 4.33E+01 | 4.28E+01 | 4.28E+01 | 6.63E+01 | 9.32E+01 | 3.94E+01 | 5.92E+01 | 9.14E-12 |
| $F_9$ | Ave | 1.22E-01 | 1.22E-01 | 2.79E+01 | 2.79E+01 | 2.41E-01 | 4.34E-01 | 4.85E-02 | 6.90E-01 | 8.76E-04 |
| | std | 4.06E+01 | 4.06E+01 | 2.22E+01 | 2.22E+01 | 2.04E+01 | 1.66E+00 | 3.91E+01 | 4.81E-01 | 4.85E-02 |
| $F_{10}$ | Ave | 5.26E-11 | 5.26E-11 | 7.75E+00 | 7.75E+00 | 1.42E-08 | 1.63E-14 | 2.83E-08 | 8.03E-16 | 8.04E-20 |
| | std | 1.10E-10 | 1.10E-10 | 4.06E+00 | 4.06E+00 | 2.17E-07 | 3.14E-15 | 4.34E-07 | 2.74E-14 | 3.34E-18 |
| $F_{11}$ | Ave | 3.28E-06 | 3.28E-06 | 4.25E+00 | 4.25E+00 | 1.16E-03 | 2.29E-03 | 2.49E-05 | 4.20E-05 | 4.23E-10 |
| | std | 4.17E-05 | 4.17E-05 | 1.99E+00 | 1.99E+00 | 2.69E-03 | 5.24E-03 | 1.34E-04 | 4.73E-04 | 5.11E-07 |
| $F_{12}$ | Ave | 9.05E-08 | 9.05E-08 | 2.62E+01 | 2.62E+01 | 1.97E-02 | 3.93E-02 | 1.34E-05 | 5.09E-03 | 6.33E-05 |
| | std | 4.83E-07 | 4.83E-07 | 1.24E+02 | 1.24E+02 | 1.24E-02 | 2.42E-02 | 6.23E-04 | 3.75E-03 | 4.71E-04 |
| $F_{13}$ | Ave | 6.33E-02 | 6.33E-02 | 1.41E+02 | 1.41E+02 | 2.38E-01 | 4.75E-01 | 9.94E-08 | 1.25E-08 | 0.00E+00 |
| | std | 4.44E-02 | 4.44E-02 | 4.32E+02 | 4.32E+02 | 1.19E-01 | 2.38E-01 | 2.61E-07 | 2.61E-07 | 0.00E+00 |

***Table 6.*** *Results for DGO and other algorithms in Multimodal test functions with low dimension*

| | | GA | PSO | ABC | CS | ALO | GWO | GOA | EPO | DGO |
|---|---|---|---|---|---|---|---|---|---|---|
| $F_{14}$ | Ave | 3.44E+00 | 3.58E+00 | 3.53E+00 | 5.20E+00 | 2.49E+00 | 3.71E+00 | 1.26E+00 | 1.08E+00 | 9.98E-01 |
| | std | 1.16E+00 | 1.18E+00 | 2.06E+00 | 2.04E+00 | 2.27E+00 | 3.86E+00 | 6.86E-01 | 4.11E-02 | 7.64E-12 |
| $F_{15}$ | Ave | 4.85E-02 | 4.13E-02 | 5.85E-02 | 6.00E-02 | 2.34E-02 | 3.66E-02 | 1.01E-02 | 8.21E-03 | 3.3E-04 |
| | std | 2.11E-02 | 2.39E-03 | 4.74E-02 | 3.86E-02 | 3.99E-02 | 7.60E-02 | 3.75E-03 | 4.09E-03 | 1.25E-05 |
| $F_{16}$ | Ave | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.02E+00 | -1.03E+00 |
| | std | 8.31E-06 | 2.10E-07 | 4.16E-06 | 1.82E-08 | 1.62E-05 | 7.02E-09 | 3.23E-05 | 9.80E-07 | 5.12E-10 |
| $F_{17}$ | Ave | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 | 3.98E-01 |
| | std | 1.91E-04 | 4.70E-16 | 9.55E-05 | 4.78E-15 | 3.81E-04 | 7.00E-07 | 7.61E-04 | 5.39E-05 | 4.56E-21 |
| $F_{18}$ | Ave | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 | 3.00E+00 |
| | std | 7.72E-06 | 3.33E-05 | 1.62E-02 | 1.62E-02 | 1.48E-05 | 7.16E-06 | 2.25E-05 | 1.15E-08 | 1.15E-18 |
| $F_{19}$ | Ave | -3.81E+00 | -3.81E+00 | -3.84E+00 | -3.80E+00 | -3.80E+00 | -3.84E+00 | -3.75E+00 | -3.86E+00 | -3.86E+00 |
| | std | 1.03E-03 | 2.19E-10 | 2.08E-01 | 2.08E-01 | 2.06E-03 | 1.57E-03 | 2.55E-03 | 6.50E-07 | 5.61E-10 |
| $F_{20}$ | Ave | -2.73E+00 | -2.86E+00 | -2.10E+00 | -1.82E+00 | -3.06E+00 | -3.27E+00 | -2.84E+00 | -2.81E+00 | -3.31E+00 |
| | std | 3.30E-01 | 3.52E-01 | 4.31E-01 | 3.48E-01 | 2.22E-01 | 7.27E-02 | 3.71E-01 | 7.11E-01 | 4.29E-05 |
| $F_{21}$ | Ave | -5.58E+00 | -6.37E+00 | -5.08E+00 | -5.95E+00 | -5.97E+00 | -9.65E+00 | -2.28E+00 | -8.07E+00 | -10.15E+00 |
| | std | 2.01E+00 | 2.56E+00 | 1.66E+00 | 1.30E+00 | 1.67E+00 | 1.54E+00 | 1.80E+00 | 2.29E+00 | 1.25E-02 |
| $F_{22}$ | Ave | -2.75E+00 | -5.76E+00 | -4.67E+00 | -3.79E+00 | -2.52E+00 | -1.04E+00 | -3.99E+00 | -10.01E+00 | -10.40E+00 |
| | std | 5.04E-01 | 1.55E+00 | 1.57E+00 | 1.32E+00 | 9.95E-01 | 2.73E-04 | 1.99E+00 | 3.97E-02 | 3.65E-07 |
| $F_{23}$ | Ave | -5.30E+00 | -6.15E+00 | -7.34E+00 | -5.92E+00 | -7.50E+00 | -1.05E+01 | -4.49E+00 | -3.41E+00 | -10.53E+00 |
| | std | 1.68E+00 | 2.45E+00 | 2.22E+00 | 1.97E+00 | 9.80E-01 | 1.81E-04 | 1.96E+00 | 1.11E-02 | 5.26E-06 |

Convergence curves of DGO and other optimization algorithms shown in Figure 2. DGO is very competitive over other optimization algorithms. Convergence curves of three models of functions is drawn up. In unimodal functions such as $F_5$, multimodal test functions with high dimension such as $F_{12}$ and multimodal test functions with low dimension such as $F_{15}$ DGO converges with more precision and quickly in the search space due to its adaptive mechanism.

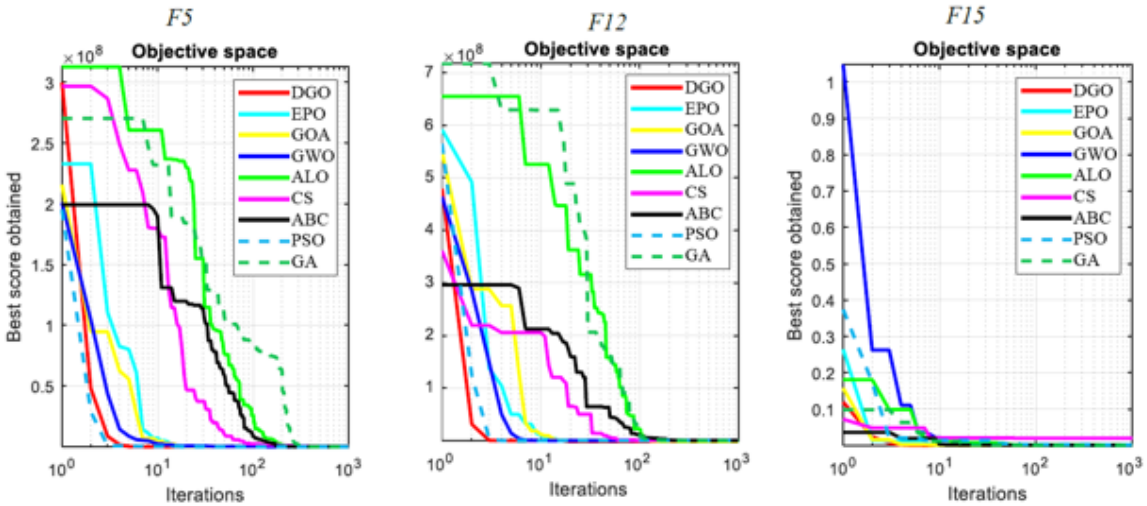


***Figure 2.*** *Convergence curves of DGO and other optimization algorithms on benchmark test problems*

### 4.3.4. Pressure Vessel Design

In this section, DGO has been applied on an engineering design problem. Mathematical model of this problem is adapted from [42]. Table 7 and Table 8 shows the performance of SSA and other algorithms. SSA provides optimal solution at (0.778099, 0.383241, 40.315121, 200.00000) with corresponding fitness value equal: 5880.0700.

***Table 7.*** *Comparison results for pressure vessel design problem*

| Algorithms | Optimum variables | | | | Optimum cost |
|---|---|---|---|---|---|
| | $T_s$ | $T_h$ | $R$ | $L$ | |
| DGO | 0.778099 | 0.383241 | 40.315121 | 200.00000 | **5880.0700** |
| EPO | 0.778210 | 0.384889 | 40.315040 | 200.00000 | 5885.5773 |
| GOA | 0.779035 | 0.384660 | 40.327793 | 199.65029 | 5889.3689 |
| GWO | 0.778961 | 0.384683 | 40.320913 | 200.00000 | 5891.3879 |
| ALO | 0.845719 | 0.418564 | 43.816270 | 156.38164 | 6011.5148 |
| CS | 0.817577 | 0.417932 | 41.74939 | 183.57270 | 6137.3724 |
| ABC | 1.085800 | 0.949614 | 49.345231 | 169.48741 | 11550.2976 |
| PSO | 0.752362 | 0.399540 | 40.452514 | 198.00268 | 5890.3279 |
| GA | 1.099523 | 0.906579 | 44.456397 | 179.65887 | 6550.0230 |

***Table 8.*** *Statistical results for pressure vessel design problem*

| Algorithms | Best | Mean | Worst | Std. Dev. | Median |
|---|---|---|---|---|---|
| DGO | **5880.0700** | **5884.1401** | **5891.3099** | 024.341 | **5883.5153** |
| EPO | 5885.5773 | 5887.4441 | 5892.3207 | **002.893** | 5886.2282 |
| GOA | 5889.3689 | 5891.5247 | 5894.6238 | 013.910 | 5890.6497 |
| GWO | 5891.3879 | 6531.5032 | 7394.5879 | 534.119 | 6416.1138 |
| ALO | 6011.5148 | 6477.3050 | 7250.9170 | 327.007 | 6397.4805 |
| CS | 6137.3724 | 6326.7606 | 6512.3541 | 126.609 | 6318.3179 |
| ABC | 11550.2976 | 23342.2909 | **33226.2526** | 5790.625 | 24010.0415 |
| PSO | 5890.3279 | 6264.0053 | 7005.7500 | 496.128 | 6112.6899 |
| GA | 6550.0230 | 6643.9870 | 8005.4397 | 657.523 | 7586.0085 |

### 4.3.5. Wilcoxon Signed Rank Test

Wilcoxon signed rank test [43] is used to compare the data in two groups dependent on each other. Based on the fitness function, the Wilcoxon test was performed at 95% confidence level (the zero hypothesis in this test indicates a lack of difference and the opposite hypothesis indicates the difference), and the result is that all eight algorithms are in terms of accuracy, has low accuracy than DGO. Wilcoxon signed rank test results shown in Tables 9 to 12. In these tables, -1 means worse, 0 means equal and 1 means better.

***Table 9.*** *Wilcoxon signed rank tests for unimodal functions $F_1$-$F_7$*

| | **EPO** | **GOA** | **GWO** | **ALO** | **CS** | **ABC** | **PSO** | **GA** |
|---|---|---|---|---|---|---|---|---|
| **$F_1$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_2$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_3$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_4$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_5$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_6$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_7$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

***Table 10.*** *Wilcoxon signed rank tests for multimodal functions $F_8$-$F_{13}$*

| | **EPO** | **GOA** | **GWO** | **ALO** | **CS** | **ABC** | **PSO** | **GA** |
|---|---|---|---|---|---|---|---|---|
| **$F_8$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_9$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_{10}$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_{11}$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_{12}$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| **$F_{13}$** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

***Table 11.*** *Wilcoxon signed rank tests for multimodal function $F_{14}$-$F_{23}$*

| | EPO | GOA | GWO | ALO | CS | ABC | PSO | GA |
|---|---|---|---|---|---|---|---|---|
| $F_{14}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{15}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{16}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{17}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{18}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{19}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{20}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{21}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{22}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |
| $F_{23}$ | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

***Table 12.*** Wilcoxon signed rank tests for pressure vessel design problem

| | EPO | GOA | GWO | ALO | CS | ABC | PSO | GA |
|---|---|---|---|---|---|---|---|---|
| **Pressure vessel design** | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 |

## 5. CONCLOSION

In this paper, a novel optimization method called Dice Game Optimizer is introduced. DGO is based on the laws of dice game. In this game, players try to collect more score by competing together.

DGO has been tested on 23 benchmark test functions. The results demonstrate that DGO have good performance as compared with Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Cuckoo Search (CS), Ant-Lion Optimizer (ALO), Grey Wolf Optimizer (GWO), Grasshopper Optimization Algorithm (GOA) and Emperor Penguin Optimizer (EPO). The results on the unimodal and multimodal test functions show the superior exploitation and exploration capability of DGO. Furthermore, DGO's performance was evaluated on an engineering problem (pressure vessel design). According to results, DGO has a high efficiency in solving this type of problem.

Accordingly, DGO can be used in different fields of science for optimization. In addition, the presentation of the binary version of DGO can be part of future study suggestions.

## REFERENCES

[1] Tang, K.S., Man, K.F., Kwong, S., and He, Q., "Genetic algorithms and their applications", IEEE signal processing magazine, 13(6): 22-37, (1996).

[2] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., "Optimization by simulated annealing", science, 220: 671-680, (1983).

[3] Dehghani, M., Mardaneh, M., and Malik, O.P., "FOA: 'Following' Optimization Algorithm for solving Power engineering optimization problems", Journal of Operation and Automation in Power Engineering, 8: 118-130, (2019).

[4] Drigo, M., "The ant system: optimization by a colony of cooperating agents", IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26: 1-13, (1996).

[5] Montazeri, Z., and Niknam, T., "Optimal utilization of electrical energy from power plants based on final energy consumption using gravitational search algorithm", Електротехніка і Електромеханіка, 4: 70-73, (2018).

[6] Dehghani, M., Montazeri, Z., Dehghani, A., and Seifi, A., "Spring search algorithm: A new meta-heuristic optimization algorithm inspired by Hooke's law", in 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 0210-0214, (2017).

[7] Ehsanifar, A., Dehghani, M., and Allahbakhshi, M., "Calculating the leakage inductance for transformer inter-turn fault detection using finite element method", in 2017 Iranian Conference on Electrical Engineering (ICEE), Tehran, 1372-1377, (2017).

[8] Dehghani, M., Mardaneh, M., Montazeri, Z., Ehsanifar, A., Ebadi, M.J., and Grechko, O.M., "Spring search algorithm for simultaneous placement of distributed generation and capacitors", Електротехніка і Електромеханіка, 6: 68-73, (2018).

[9] Dehghani, M., Montazeri, Z., Ehsanifar, A., Seifi, A., Ebadi, M.J., and Grechko, O., "Planning of energy carriers based on final energy consumption using dynamic programming and particle swarm optimization", Електротехніка і Електромеханіка, 5: 62-71, (2018).

[10] Montazeri, Z., and Niknam, T., "Energy carriers management based on energy consumption", in 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 0539-0543, (2017).

[11] Dehbozorgi, S., Ehsanifar, A., Montazeri, Z., Dehghani, M., and Seifi, A., "Line loss reduction and voltage profile improvement in radial distribution networks using battery energy storage system", in 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 0215-0219, (2017).

[12] Tarasewich, T., and McMullen, P.R., "Swarm intelligence: power in numbers", Communications of the ACM, 45: 62-67, (2002).

[13] Barry, J., and Thron, C., "A Computational Physics-Based Algorithm for Target Coverage Problems", in Advances in Nature-Inspired Computing and Applications, ed: Springer, 269-290, (2019).

[14] Rashedi, E., H. Nezamabadi-Pour, H., and Saryazdi, S., "GSA: a gravitational search algorithm", Information Sciences, 179: 2232-2248, (2009).

[15] Eskandar, H., Sadollah, A., Bahreininejad, A., and Hamdi, M., "Water cycle algorithm–A novel metaheuristic optimization method for solving constrained engineering optimization problems", Computers & Structures, 110: 151-166, (2012).

[16] Dehghani, M., Montazeri, Z., Dehghani, A., Nouri, N., and Seifi, A., "BSSA: Binary spring search algorithm", in 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 0220-0224, (2017).

[17] Shah-Hosseini, H., "Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation", International Journal of Computational Science and Engineering, 6: 132-140, (2011).

[18] Moghaddam, F.F., Moghaddam, R.F., and Cheriet, M., "Curved space optimization: A random search based on general relativity theory", arXiv preprint arXiv, 1208.2214, (2012).

[19] Alatas, B., "ACROA: artificial chemical reaction optimization algorithm for global optimization", Expert Systems with Applications, 38: 13170-13180, (2011).

[20] Du, H., Wu, X., and Zhuang, J., "Small-world optimization algorithm for function optimization", in International Conference on Natural Computation, 264-273, (2006).

[21] Formato, R.A., "Central force optimization: a new nature inspired computational framework for multidimensional search and optimization", in Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), ed: Springer, 221-238, (2008).

[22] Bansal, J.C., "Particle Swarm Optimization", in Evolutionary and Swarm Intelligence Algorithms, ed: Springer, 11-23, (2019).

[23] Mirjalili, S., Mirjalili, M., and Lewis, A., "Grey wolf optimizer", Advances in Engineering Software, 69: 46-61, (2014).

[24] Saremi, S., Mirjalili, S., and Lewis, A., "Grasshopper optimisation algorithm: theory and application", Advances in Engineering Software, 105: 30-47, (2017).

[25] Dhiman, G., and Kumar, V., "Emperor penguin optimizer: A bio-inspired algorithm for engineering problems", Knowledge-Based Systems, 159: 20-50, (2018).

[26] Dorigo, M., and Stützle, T., "Ant colony optimization: overview and recent advances", in Handbook of metaheuristics, ed: Springer, 311-351, (2019).

[27] Karaboga, D., and Basturk, B., "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems", in International fuzzy systems association world congress, 789-798, (2007).

[28] Gandomi, A.H., Yang, X.S., and Alavi, A.H., "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems", Engineering with computers, 29: 17-35, (2013).

[29] Mirjalili, S., "The ant lion optimizer," Advances in Engineering Software, 83: 80-98, (2015).

[30] Yang, X.S., "A new metaheuristic bat-inspired algorithm", in Nature inspired cooperative strategies for optimization (NICSO 2010), ed: Springer, 65-74, (2010).

[31] Mirjalili, S., and Lewis, A., "The whale optimization algorithm", Advances in Engineering Software, 95: 51-67, (2016).

[32] Dhiman, G., and Kumar, V., "Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications", Advances in Engineering Software, 114: 48-70, (2017).

[33] Castillo, O., and Aguilar, L.T., "Genetic Algorithms", in Type-2 Fuzzy Logic in Control of Nonsmooth Systems, ed: Springer, 23-39, (2019).

[34] Storn, R., and Price, K., "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization, 11: 341-359, (1997).

[35] Beyer, H.G., and Schwefel, H.P., "Evolution strategies–A comprehensive introduction", Natural Computing 1: 3-52, (2002).

[36] Koza, J.R., "Genetically breeding populations of computer programs to solve problems in artificial intelligence", in [1990] Proceedings of the 2nd International IEEE Conference on Tools for Artificial Intelligence, 819-827, (1990).

[37] Mirjalili, S., "Biogeography-Based Optimisation", in Evolutionary Algorithms and Neural Networks, ed: Springer, 57-72, (2019).

[38] Dehghani, M., Montazeri, Z., Malik, O.P., Ehsanifar, A., and Dehghani, A., "OSA: Orientation Search Algorithm", International Journal of Industrial Electronics, Control and Optimization, 2: 99-112, (2019).

[39] Yao, X., Liu, Y., and Lin, G., "Evolutionary programming made faster", IEEE Transactions on Evolutionary Computation, 3: 82-102, (1999).

[40] Mirjalili, S., "Genetic Algorithm", in Evolutionary Algorithms and Neural Networks, ed: Springer, 43-55, (2019).

[41] Mirjalili, S., "Particle Swarm Optimisation", in Evolutionary Algorithms and Neural Networks, ed: Springer, 15-31, (2019).

[42] Kannan B., and Kramer, S.N., "An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design", Journal of Mechanical Design, 116: 405-411, (1994).

[43] Woolson, R., "Wilcoxon signed-rank test", Wiley Encyclopedia of Clinical Trials, 1-3, (2007).