Journal of Ambient Intelligence and Humanized Computing (2023) 14:8375–8385
https://doi.org/10.1007/s12652-021-03602-1

ORIGINAL RESEARCH

Review and analysis for the Red Deer Algorithm

Raed Abu Zitar[1] · Laith Abualigah[2,3] · Nidal A. Al-Dmour[4]

Received: 10 August 2021 / Accepted: 3 November 2021 / Published online: 20 November 2021

Abstract

In this paper, the Red Deer algorithm (RDA), a recent population-based meta-heuristic algorithm, is thoroughly reviewed. The RD algorithm combines the survival of the fittest principle from the evolutionary algorithms and the productivity and richness of heuristic search techniques. Different variants and hybrids of this algorithm are presented and investigated. All the applications that were solved with this algorithm are presented. It is crucial to analyze the performance of this algorithm, therefore, the paper sheds light on the algorithm unique features and weaknesses covering the applications that are primarily suitable for it. The conclusions are presented, and further recommendations are suggested based on the review and analysis covered. The readers of this paper will have an understanding of the RD algorithm and its variants and, consequently, decide how suitable this algorithm is for their own business, research, or industrial applications.

Keywords Red Deer Algorithm · Meta-heuristics · Evolutionary · Productivity · Exploitation

1 Introduction

The search for optimal solutions is one of the most challenging topics in Artificial Intelligence (Abualigah et al. 2021). Industry, businesses, and all types of service providers are keen to find optimal solutions for their processes, maintaining the quality and the high performance of their standards. Machine learning, features extraction, regression, and classification operations are built on optimum parameters selection (Burke et al. 2013). Recently, hybrid techniques that are meta-heuristic-based are showing great success in different applications (Hussain et al. 2019). They are iterative techniques with several parameters that can be tuned to fuel their search engine and span the search spaces locally, globally, and collectively (Osman and Laporte 1996).

There are many types of metaheuristic search techniques (Blum and Roli 2003): nature/inspired vs. non/nature-inspired, single vs. various neighborhood structures, and memory usage vs. memory-less, single objective vs. multiple objective-based techniques, stochastic vs. deterministic, discrete vs. continuous, population-based vs. single point search techniques, dynamic vs. static objective functions, and so many. Recently, nature-inspired algorithms are classified into several categories such as evolutionary algorithms (EAs), swarm intelligence, physical-based, chemical-based, Local Search Algorithms (LSA), and human-based algorithms (Fausto et al. 2020; Meraihi et al. 2020), and many other application customized algorithms such as feature extraction-GWO (Alomari et al. 2021), COVID 19-optimization (Dalbah et al. 2021), JAYA-machine learning (Zitar et al. 2021), load dispatch-salp swarm (Alkoffash et al. 2021), and mathematical optimization (Al-Muhammed and Zitar 2018). Neural based techniques also are extensively used in optimization and machine learning (Ren et al. 2020). Other techniques that are norm based such as in Yan et al. (2020), where Linear Discriminant Analysis (LDA's) with edge classes that are based on norm maximization is investigated. It also proved to be an efficient algorithm for classification and optimizing distances between classes.

✉ Raed Abu Zitar
raed.zitar@sorbonne.ae

Laith Abualigah
aligah.2020@gmail.com

Nidal A. Al-Dmour
nidal@mutah.edu.jo

[1] Sorbonne Center of Artificial Intelligence, Sorbonne University-Abu Dhabi, Abu Dhabi, UAE

[2] Faculty of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan

[3] School of Computer Sciences, Universiti Sains Malaysia, 11800 Pulau, Pinang, Malaysia

[4] Department of Computer Engineering, Mutah University, Mutah, Jordan

The paper is arranged as follows: the inspiration and the description of the RDA is described in Sect. 2, including theoretical analysis and mathematical modeling. In Sect. 3, all the applications available in literature regarding the RDA were presented including all the variants and the hybridisations used. In Sect. 4, comparative studies are shown covering comprehensive testing suite of mathematical functions for the RDA and a group of the most popular metaheuristic algorithms. In Sect. 5, the outcomes of the paper are summarized with recommendations for future work and invitations for researchers to invest in this recently proposed algorithm.

2 Basic concepts of RD algorithm

2.1 Inspiration of RD algorithm

The inspiration of this algorithm started by observing the mating activities of the red deer species that live in Britain. A harem will be formed when a male succeeds in absorbing a group of hinds. The strongest male will be the commander of the harem. Commanders protect their territories and the harem. The roaring is implemented during the mating season, which is mostly Autumn. Only the strongest male after a fierce fight with the other males will be the commander of the harem. It is the only one allowed to mate with the females. The roaring could be followed by an approacher (another male) coming to the roaring male. A fight could be ignited, and the winner will take over the harem.

2.2 Procedural steps of RD algorithm

The RDA, like other meta-heuristics, begins with a random population, which is the RDs' counterpart. The best RDs in the population are chosen and dubbed the "male RD", while the remainder is dubbed the "hinds". First and foremost, the male RD must roar to be classified. They are classified into two groups based on the strength of a roaring phase (i.e., commanders and stags). After then, each harem's commander and stags fight it out to own their harem. Commanders subsequently form the harems. The quantity of hinds in harems is proportional to the leaders' roaring and fighting ability. As a result, commanders in harems mate with a large number of hinds. The other males (i.e., stags) mate with the nearest hind without regard for the harem's limitations. As a result, the RDA's stages, by mating with nearest hinds, are participating in the exploitation and exploration phases. All the previous steps are employed in a flowchart and mathematical model as shown next.

Figure 1 depicts the RDA's flowchart (Fathollahi-Fard et al. 2021) showing the flow of the RDA steps. The iterative behavior of the algorithm is presented in the flowchart. The initial and random population provides rich field for initial solutions. A random guided selection for the highest roaring males is done next. The winner between the commander and the stags get to form the harems. Mating with own harem and another harem provides a production phase for more innovative solutions. The stag is allowed to mate with nearest hind for more richness and unexpected good solutions.

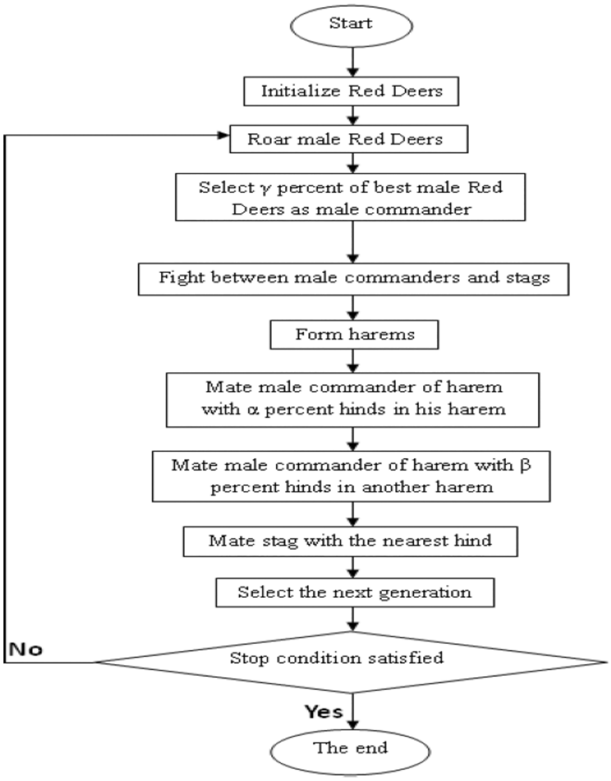
Start
Initialize Red Deers
Roar male Red Deers
Select γ percent of best male Red Deers as male commander
Fight between male commanders and stags
Form harems
Mate male commander of harem with α percent hinds in his harem
Mate male commander of harem with β percent hinds in another harem
Mate stag with the nearest hind
Select the next generation
No
Stop condition satisfied
Yes
The end

Fig. 1 The flowchart of the RD algorithm (Fathollahi-Fard et al. 2021)

The steps of the algorithm are mathematically described based on the models offered by Fathollahi-Fard et al. (2021). The steps are shown briefly since the goal here is to present some revisions.

To mathematically model the RDA we start by Eq. (1) in which an initial population for the RDs is created:

$$RD = X1, X2, X3, \dots, X_{Nvar} \tag{1}$$

Then the fitness of every member in the population is calculated according to Eq. (2):

$$Value = f(RD) = f(X1, X2, X3, \dots, X_{Nvar}) \tag{2}$$

Male RDs are attempting to boost their grace by roaring in this step. As a result, the roaring process may succeed or fail, just as it does in nature. Notably, in this algorithm, male RDs are the best options. In terms of the solution space, we look for the neighbors of the solution. If the objective functions of the neighbors are male RD, it is replaced by the preceding if it is better than the previous male RD. The following information is provided to keep males up to speed on their status. The following equation is proposed:

$$male_{new} = \begin{cases} male_{old} + a_1(UB - LB) * a_2 + LB), \text{if } a_3 \geq 0.5 \\ male_{old} - a_1((UB - LB) * a_2 + LB), \text{if } a_3 < 0.5 \text{ is less than } 0 \end{cases} \quad (3)$$

UB and *LB* limit the search space in order to develop a suitable male neighborhood solution. They are, respectively, the top and lower boundaries of search space. Note that the present position of male RD is $male_{old}$, and its next position is $male_{new}$. In terms of randomization; $a1$, $a2$, and $a3$ are the three stages of the roaring process in nature derived at random from a uniform distribution between zero and one.

The number of male commanders is computed as follows:

$$N_C = round(\gamma . N_{male}) \quad (4)$$

where N_C is the number of commanders that are naturally males, γ is a random number between 0 and 1, and N_{male} is the number of total males. It's worth noting that γ is the algorithm model's beginning value. It has a value range of zero to one. Finally, the number of stags is estimated using the following formula:

$$N_s = N_{male} - N_C \quad (5)$$

The fighting process are modeled by the following two mathematical formulas:

$$new_1 = (C + S)/2 + b_1((UB - LB) * b_2 + LB) \quad (6)$$

$$new_2 = (C + S)/2 - b_1((UB - LB) * b_2 + LB) \quad (7)$$

The two new solutions developed by the fighting process are new_1 and new_2. The symbols for commanders and stags, respectively, are *C* and *S*. *UB* and *LB* the set upper limits on the feasibility of the new solutions. The search space's upper and lower bounds b_1 and b_2 are formed as a result of the randomization of the fighting process between zero and one using a uniform distribution function. Considering four options: *C*, *S*, new_1 and new_2, only the best one in terms of the *OF* will be chosen (see below).

To form harems groups, we split hinds among commanders to construct harems, in proportion to:

$$V_n = v_n - \max v_i \quad (8)$$

where V_n is the normalized value of the n_{th} commander's power (i.e., its OF), and v_n is the power of the n_{th} commander (i.e., its OF). The following equation can be used to calculate commanders' normalized power.

$$P_n = \left| \frac{V_n}{\sum_{i=1}^{a_i} V_i} \right| \quad (9)$$

The number of hinds of a harem can be calculated as follows:

$$N.harem_n = round(P_n \cdot N_{hind}) \quad (10)$$

where N_{hind} is the total number of hinds.

This deer mating action is carried out by a commander with a percentage of hinds in his harem.

$$N.harem_n^{mate} = round(\alpha \cdot N.harem_n) \quad (11)$$

The number of hinds in the n_{th} harem that mate with their leader is $N.harem_n^{mate}$. In terms of the solution space, we choose $N.harem_n^{mate}$ of the$N.harem_k$ at random. In general, the mating process is described as follows:

$$\text{offs} = \frac{C + Hind}{2} + (UB - LB) \quad \text{x} \quad c \quad (12)$$

We pick a harem at random (call it *k*) and allow the male commander to mate with β percent of the harem's hinds. In fact, in order to expand his area, the commander can launch an attack on other harems. Assume β is the algorithm model's initial parameter value with range of values between zero and one. The number of hinds in the harem that mate with the commander is calculated using the following formula:

$$N.harem_k^{mate} = round(\beta \cdot N.harem_k) \quad (13)$$

where $N.harem_k^{mate}$is the number of hinds in the k_{th} harem that mate with the commander. It's worth noting that the mating procedure is carried out using Eq. (12). The distance between a stag and all hinds in $J - dimension$ space should be computed as follows:

$$d_i = \left\{ \sum_{j \epsilon J} (stag_j - hind_j^i)^2 \right\}^{1/2} \quad (14)$$

where d_i is the distance between the i_{th} hind and a stag. As a result, the hind picked is represented by the minimum value in this matrix. The mating process begins after a hind is chosen. In this calculation, a stag is considered instead of a commander.

3 Applications of RD algorithms

Authors in Fard and Hajiaghaei-Keshteli (2016) compared the RDA with 12 standard algorithms using a set of real-world engineering applications and multi-objective optimization problems such as SMSP, TSP, FCTP, and VRP. A multi-objective NRP evaluated four assessment metrics to analyze Pareto optimal sets. They concluded that the RDA was more accessible to tune than other algorithms, and the interactions between intensification and diversification phases can be easily employed. In Fazli et al. (2019) the RDA was used to solve the Coordinated Quay Crane Scheduling and Assignment Problem (CQCSAP). The verification was conducted through eight numerical instances in small and large sizes. The range of the parameters used in the tests was based on Lourenço et al. (2003). A comparison was made between RDA and Genetic Algorithm (GA). The results in Fazli et al. (2019) exhibited the high performance of RDA. In Dey et al. (2021), authors proposed Modified Red Deer Algorithm (MRDA) to find optimum thresholds of publicly available grayscale images. The performance of the MRDA was compared with RDA and classical Particle Swarm Optimization (PSO). Experiments were conducted using two publicly accessible real-world benchmark images. The results exhibited that MRDA outperformed both RDA and PSO. Moreover, in De et al. (2020), MRDA has been compared with RDA and Classical Genetic Algorithm (CGA). Experiments in De et al. (2020) were conducted on real-life gray-scale images. Both the t-test and Friedman Test were used to check which algorithm is the best. MRDA outperformed both RDA and GA and achieved competitive results.Authors in Brammya et al. (2019), introduced a meta-heuristic algorithm named DHOA, slightly different from RDA. It is based on the hunting behavior of humans toward deer. In DHOA, the hunters behave differently from DRA, where the two hunters move their best positions, termed as leader and successor. The experiments on DHOA show competitive results compared to other algorithms such as GWO, WOA, FF, PSO, etc. The hybrid model referred to Red Deer Adopted Wolf Algorithm (RDAWA) was introduced in Alotaibi (2021). The authors assess the potential movement of the financial exchange's stock value using the RDAWA. The RDAWA is a hybrid optimization model of two standard optimization models: RDA and GWO, respectively. The proposed stock market prediction model with three phases: feature extraction, optimal, feature selection, and prediction. The RDAWA model exhibited better results than other FF + Ensemble Classifiers and GWOC Ensemble Classifier approaches. The authors of Dogani et al. (2020) proposed a hybrid meta-heuristic algorithm RDA-SA, which is a combination of the Red Deer Algorithm) and Simulated Annealing. They analyzed the uncertain parameters in the water supply system remodeling and restoration. They found that the RDA-SA algorithm outperformed NSGA-II through different criteria and analyses. The red deer algorithm was also used to solve the real-world size case of truck scheduling problems in cross-docking with product time window (Zhou and Zong 2021). The lower bound of the problem is based on both the Lagrangian relaxation problem and the NP-hard nature of the truck scheduling problem. A modified RDA was proposed with a heuristic oscillating local search algorithm and adaptive memory programming. This modified RDA could overcome the original RDA's inferior capability concerning local search and run time. In another application, Sensor nodes in Wireless Sensor Networks (WSN) are responsible for collecting the data, and an efficient routing protocol is needed for data collection. An efficient routing protocol must be able to allocate resources with a low minimum rate of packet transmission loss. In Ambareesh and Madheswari (2021), a Hybrid Red Deer Salp Swarm (HRDSS) was proposed for routing in WSN. HRDSS is a hybrid of RDA and the salp swarm optimization algorithm. The experiments conducted in Ambareesh and Madheswari (2021) showed that DRDSS exhibited the best optimal routing path in comparison to other Routing Protocols. For WSN, RDA was also used in Nguyen et al. (2020) for energy-efficient and secure clustering-based data transmission in pervasive wireless networks. The clustering in WSN was implemented using block-chain enabled secure data transmission and the RDA (RDAC-BC). The experiments showed that RDAC-BC outperformed other algorithms in terms of energy, network lifetime, packet delivery ratio (PDR), and throughput.

The cross-docking system is for the purpose of product distribution from manufacturers to markets (Berghman et al. 2015). These products are transferred to the outbound dock in order to be loaded into shipping trucks (Fathollahi-Fard et al. 2020). Long-term storage is not permitted in cross-docking (Zuluaga et al. 2017). Additionally, cross-docking can remove products' retrieval costs. Therefore, an efficient cross-docking will lead to better distribution systems which result in the satisfaction of customers (Boysen and Fliedner 2010). Modified Red Deer Algorithm was proposed to Solve a Truck Scheduling Problem Considering Time Windows and Deadline for Trucks' Departure (Fathollahi-Fard et al. 2020). The main feature of MRDA is fewer controlling parameters needed, and therefore a better designation is achieved of search phases to find the global solution. The performance of MRDA was compared with four techniques: SA, GA, PSO, and ICA, and MRDA outperformed all of them. An Improved RDA (IRDA) was used as an Engineer design for Direct Current (DC) brushless motor (Fathollahi-Fard et al. 2019). The performance of IRDA was also compared through simulation with the other 12 algorithms.

IRDA was able to reduce time consumption and provide better efficiency compared with different algorithms.

4 Comparative results and discussions

This section presents comprehensive experiments and analysis for the RD algorithm and compares its results with other well-known optimization methods published in the literature.The details of the tested 10 classical benchmark functions are given in Table 1. This table presents the functions equations, specifications, dimension size, and the best fitness values of each problem of the 10 classical benchmark functions (Alsalibi et al. 2021; Shehab et al. 2020; Eid et al. 2021). The given test functions are separated into two main sections: (1) Unimodal functions with varying dimensions (F1–F7)and (2) Multimodal functions, which have varying dimensions (F8–F10). Many previous papers have employed these functions to test the effectiveness of different optimization methods.

The studied algorithm (i.e., Red Deer Algorithm Fathollahi-Fard et al. 2021) is compared with other recent and well-known methods, including:

- Whale Optimization Algorithm (WOA) is proposed by Mirjalili and Lewis (2016).
- Sine Cosine Algorithm (SCA) is proposed by Abualigah and Diabat (2021).
- Dragonfly Algorithm (DA) is proposed by Mirjalili (2016).
- Grey Wolf Optimizer (GWO) is proposed by Safaldin et al. (2021).
- Particle Swarm Optimization (PSO) is proposed by Eberhart and Kennedy (1995).
- Ant Lion Optimizer (ALO) is proposed by Mirjalili (2015).
- Marine Predators Algorithm (MPA) is proposed by Al-Qaness et al. (2020).
- Equilibrium Optimizer (EO) is proposed by Faramarzi et al. (2020).
- Aquila Optimizer (AO) is proposed by Abualigah et al. (2021).
- Arithmetic Optimization Algorithm (AOA) is proposed by Premkumar et al. (2021).

In the experiments listed here, Table 2 represents the results of the comparative methods in terms of the worst (Max), mean, best (Min), and standard deviation (STD) of fitness functions. Moreover, the Friedman ranking test (Mack and Skillings 1980) is implemented to rank the tested methods according to the obtained results in the overall the tested problems. For fairness, all applied optimization methods run using the same environment and with the same test conditions. Note that the parameters of the tested algorithms are taken from the original papers. All algorithms are tested on 30 runs and 500 iterations. It is exceptionally challenging for optimization algorithms to address numerical functions because of their original complex nature. Addressing such problems needs a precise equilibrium between its exploration and exploitation search processes. Table 2 shows the results of the tested algorithms to solve 10 benchmark functions when the dimension size is fixed to 10. Investigating the results in Table 2 explains that the RDA can give very competitive outcomes on small dimensional functions, mainly significantly improving in F7 and F10 and others. Accordingly, it can be decided that the RDA searches close to the optimum solution effectively and comparable to several other methods. Based on the Friedman ranking test, RDA got the fourth-ranked.According to the obtained results summarized in Table 2, RDA can also produce excellent results using the fixed-dimension multi-modal functions. These results confirm that the studied RDA is competitive

Table 1 Unimodal and multimodal benchmark functions

Function	Description	Dimensions	Range	f_{min}
F1	$f(x) = \sum_{i=1}^{n} x_i^2$	10, 100	[– 100, 100]	0
F2	$f(x) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	10, 100	[– 10, 10]	0
F3	$f(x) = \sum_{i=1}^{d} (\sum_{j=1}^{i} x_j)^2$	10, 100	[– 100, 100]	0
F4	$f(x) = max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	10, 100	[– 100, 100]	0
F5	$f(x) = \sum_{i=1}^{n-1} [100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2]$	10, 100	[– 30, 30]	0
F6	$f(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	10, 100	[– 100, 100]	0
F7	$f(x) = \sum_{i=0}^{n} i x_i^4 + \text{random}[0, 1)$	10, 100	[– 128, 128]	0
F8	$f(x) = \sum_{i=1}^{n} (-x_i sin(\sqrt{\lvert x_i \rvert}))$	10, 100	[– 500, 500]	– 418.9829× n
F9	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10cos(2\pi x_i) + 10]$	10, 100	[– 5.12, 5.12]	0
F10	$f(x) = -20exp(-0.2\sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}) -exp(\frac{1}{n} \sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$	10, 100	[– 32, 32]	0

Table 2 Results of the comparative methods using 10 benchmark functions (F1–F10) when the dimension size is 10

Fun	RDA	WOA	SCA	DA	GWO	PSO	ALO	MPA	EO	AO	AOA
F1											
Max	4.4665E−40	2.3167E−46	7.2511E−05	1.3751E+02	7.5101E−31	1.4732E−10	1.0587E−05	2.7291E−29	2.0817E−09	1.8038E−89	2.4868E−242
Mean	1.2191E−40	6.0948E−47	1.8138E−05	5.8711E+01	1.9205E−31	3.7321E−11	4.5206E−06	1.4899E−29	1.8221E−09	4.5095E−90	6.2169E−243
Min	8.0216E−44	7.1077E−56	2.9624E−11	1.2541E+01	1.4389E−33	1.3185E−15	6.6491E−07	1.8107E−32	1.3910E−09	1.2594E−104	0.0000E+00
Std	2.1691E−40	1.1396E−46	3.6249E−05	5.5707E+01	3.7265E−31	7.3336E−11	4.5575E−06	1.4038E−29	3.0043E−10	9.0190E−90	0.0000E+00
Ranking	4	3	10	11	5	7	9	6	8	2	1
F2											
Max	1.0506E−22	1.7012E−36	4.3347E−06	2.4696E+00	1.1915E−18	6.8402E−05	2.9451E+01	2.1140E−16	5.9555E+00	4.2744E−47	0.0000E+00
Mean	3.4374E−23	5.5542E−37	1.5693E−06	1.9512E+00	5.2089E−19	2.3262E−05	8.3979E+00	1.0684E−16	1.7971E+00	1.1357E−47	0.0000E+00
Min	3.1248E−24	3.2099E−40	1.3922E−09	1.3081E+00	2.3878E−19	9.8348E−08	3.2373E−01	1.0236E−17	6.3010E−02	1.9599E−57	0.0000E+00
Std	4.7942E−23	8.0213E−37	1.9693E−06	5.5303E−01	4.4952E−19	3.2121E−05	1.4093E+01	9.6004E−17	2.8010E+00	2.0963E−47	0.0000E+00
Ranking	4	3	7	10	5	8	11	6	9	2	1
F3											
Max	3.6076E−17	1.8594E+03	8.2819E−02	1.0110E+04	6.6339E−12	6.5661E−03	8.7320E+02	3.1183E−12	9.2498E+00	2.0536E−95	2.0803E−203
Mean	9.6288E−18	7.0169E+02	3.0307E−02	3.0864E+03	1.9281E−12	2.5817E−03	5.9432E+02	7.9978E−13	3.9518E+00	5.2358E−96	5.2030E−204
Min	1.2617E−22	3.0172E+01	9.1180E−05	2.1715E+02	2.2300E−14	7.6390E−04	2.6260E+02	2.2508E−16	3.6751E−02	7.9374E−126	1.2881E−226
Std	1.7641E−17	8.3892E+02	3.9133E−02	4.7334E+03	3.1669E−12	2.6990E−03	2.6492E+02	1.5459E−12	4.0214E+00	1.0202E−95	0.0000E+00
Ranking	3	10	7	11	5	6	9	4	8	2	1
F4											
Max	3.1877E+00	6.2719E+01	2.4831E−01	1.2600E+01	9.6839E−10	2.4099E−02	1.0916E+01	2.8518E−11	1.8005E−12	1.6117E−45	9.1751E−94
Mean	1.3425E+00	1.6422E+01	8.0817E−02	4.8558E+00	2.9582E−10	7.2128E−03	6.3043E+00	1.1245E−11	4.5856E−13	4.0325E−46	2.2938E−94
Min	3.2397E−03	4.0772E−01	2.0788E−03	0.0000E+00	1.9851E−12	8.3975E−04	1.4680E+00	1.9841E−12	1.4235E−15	1.4724E−57	4.6008E−146
Std	1.5134E+00	3.0868E+01	1.1285E−01	5.4090E+00	4.5238E−10	1.1270E−02	4.9197E+00	1.2377E−11	8.9467E−13	8.0565E−46	4.5875E−94
Ranking	8	11	7	9	5	6	10	4	3	2	1
F5											
Max	2.0684E+02	8.7313E+00	8.7356E+00	1.4292E+04	8.0676E+00	1.0292E+02	3.1696E+02	6.5392E+00	6.6470E+00	1.8186E-02	8.3000E+00
Mean	5.8016E+01	8.1127E+00	8.3831E+00	5.4573E+03	7.3965E+00	3.2743E+01	1.1168E+02	5.5906E+00	6.4133E+00	7.4648E-03	7.7767E+00
Min	7.2825E+00	7.5181E+00	7.8866E+00	2.0831E+02	6.2797E+00	7.1469E+00	4.0916E+00	4.6398E+00	6.2322E+00	8.2569E−04	7.5226E+00
Std	9.9219E+01	4.9652E−01	4.1700E−01	6.4294E+03	8.5289E−01	4.6880E+01	1.4667E+02	1.0923E+00	1.9987E-01	8.0002E−03	3.6331E−01
Ranking	9	6	7	11	4	8	10	2	3	1	5
F6											
Max	6.0786E−09	2.7778E−01	1.0395E+00	1.3543E+02	2.5142E−01	2.3736E−11	1.3537E−05	2.4102E−10	4.0810E−06	9.2063E−05	3.5369E−01
Mean	2.8409E−09	1.5339E−01	7.1853E−01	7.4233E+01	1.8744E−01	1.1350E−11	6.4479E−06	1.5832E−10	1.7758E−06	2.5651E−05	1.5807E−01
Min	6.6315E−10	3.5205E−02	4.9331E−01	1.5310E+01	1.8529E−05	1.8190E−13	3.0955E−07	5.0974E−11	1.3706E−07	1.5284E−06	4.8314E−02
Std	2.5151E−09	1.3551E−01	2.3990E−01	5.0094E+01	1.2496E−01	1.2805E−11	5.6425E−06	7.8880E−11	1.8502E−06	4.4311E−05	1.4029E−01
Ranking	3	7	10	11	9	1	5	2	4	6	8

Table 2 (continued)

Fun	RDA	WOA	SCA	DA	GWO	PSO	ALO	MPA	EO	AO	AOA
F7											
Max	1.2765E−03	1.3525E−02	8.7623E−03	3.0681E−02	4.0858E−03	2.2769E−02	1.7018E−01	1.4299E−03	3.3658E−03	5.8169E−02	4.0560E−04
Mean	5.1447E−04	8.1611E−03	3.9698E−03	2.5479E−02	3.6829E−03	1.7387E−02	1.2060E−01	1.1167E−03	1.9850E−03	2.4866E−02	1.4046E−04
Min	7.6545E−05	2.6240E−04	3.9324E−04	2.1248E−02	3.1882E−03	1.2322E−02	6.3527E−02	8.1528E−04	2.0217E−04	2.1658E−03	9.0946E−06
Std	5.5972E−04	5.7073E−03	3.8547E−03	4.6292E−03	4.1159E−04	4.6527E−03	4.9197E−02	2.5561E−04	1.3947E−03	2.4008E−02	1.7942E−04
Ranking	2	7	6	10	5	8	11	3	4	9	1
F8											
Max	− 2.1715E+03	− 2.5533E+03	− 1.7176E+03	− 2.1891E+03	− 2.0477E+03	− 1.2058E+03	− 1.8059E+03	− 3.3000E+03	− 3.0234E+03	− 2.1543E+03	− 2.0372E+03
Mean	− 2.5927E+03	− 2.9919E+03	− 1.8350E+03	− 2.9223E+03	− 2.5503E+03	− 1.5428E+03	− 2.2656E+03	− 3.4492E+03	− 3.2601E+03	− 3.2189E+03	− 2.3295E+03
Min	− 3.0221E+03	− 3.3436E+03	− 2.1322E+03	− 3.4536E+03	− 3.1028E+03	− 1.8752E+03	− 3.4200E+03	− 3.7753E+03	− 3.8345E+03	− 4.1883E+03	− 2.4998E+03
Std	3.5092E+02	3.3128E+02	1.9852E+02	5.2932E+02	4.3429E+02	2.8123E+02	7.7139E+02	2.2454E+02	3.8496E+02	1.1173E+03	2.1454E+02
Ranking	6	4	10	5	7	11	9	1	2	3	8
F9											
Max	4.3778E+01	1.3318E+01	1.4698E−05	3.1340E+01	7.7444E+00	1.3930E+01	3.2834E+01	2.4488E−02	0.0000E+00	1.0919E−04	0.0000E+00
Mean	3.1092E+01	3.3294E+00	5.5400E−06	2.2493E+01	3.8982E+00	1.1580E+01	2.9600E+01	6.1221E−03	0.0000E+00	3.8146E−05	0.0000E+00
Min	1.5919E+01	0.0000E+00	2.1295E−08	1.1716E+01	1.4211E−14	5.9698E+00	2.1889E+01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00
Std	1.1528E+01	6.6588E+00	6.9054E−06	8.4795E+00	3.9528E+00	3.7622E+00	5.2255E+00	1.2244E−02	0.0000E+00	5.1590E−05	0.0000E+00
Ranking	11	6	3	9	7	8	10	5	1	4	1
F10											
Max	4.4409E−15	2.4364E−05	6.4673E−07	5.0790E+00	2.9310E−14	2.5889E−07	1.3799E−03	7.9936E−15	7.9936E−15	8.8818E−16	8.8818E−16
Mean	7.9936E−15	1.1551E+00	1.1535E−01	1.9901E+01	3.2863E−14	3.8818E−05	2.5799E+00	7.9936E−15	7.9936E−15	2.6700E−11	8.8818E−16
Min	5.3291E−15	2.8880E−01	2.8839E−02	8.4875E+00	2.9310E−14	1.5777E−05	6.4567E−01	6.2172E−15	7.1054E−15	6.6764E−12	8.8818E−16
Std	4.4409E−15	1.6587E−05	3.8174E−07	3.8467E+00	2.5757E−14	2.5889E−07	6.7485E−04	4.4409E−15	4.4409E−15	8.8818E−16	8.8818E−16
Ranking	2	9	8	11	5	7	10	2	2	6	1
Mean rank	4.54	6.23	7.85	10.00	5.69	6.85	8.92	3.77	4.54	3.46	3.77
Friedman Ranking	4	7	9	11	6	8	10	2	4	1	2

regarding exploration search. RDA method obtained a set of good results for solving such problems, compared to several new and highly efficient algorithms. This confirms the ability of the studied RDA to solve complex problems and achieve good results as well.

Figure 2 presents the convergence rate of the RDA and the other well-known comparative methods over 500 iterations. The illustrated curves show that the RDA can converge quicker and smoother than most of the tested methods, as shown in F5 and F6 figures. Besides, most of the analyzed

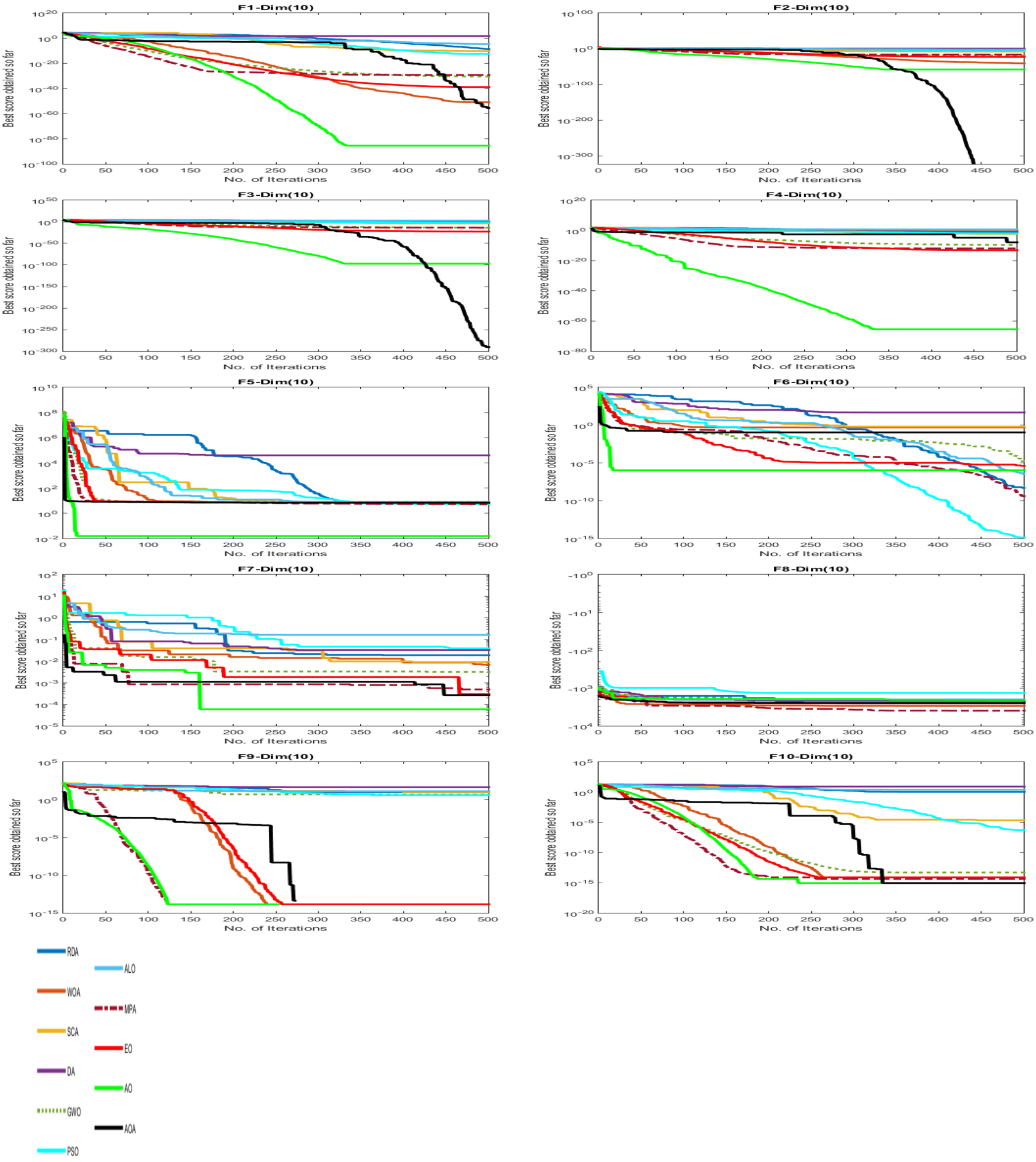

Fig. 2 The convergence curves of the tested optimization algorithms using 10 benchmark functions

optimization methods show stagnation for some of the tested functions. Thus, the RDA exceeds the most recently proposed optimization methods (i.e., WOA, SCA, PSO, and ALO), in terms of solutions and convergence speed. The curves also reveal that the studied RDA has significant changes in the first iterations and small changes in the final iterations where convergence and stability is expected.

Table 3 presents the results of the tested algorithms on four problems (F1–F4) through one second of simulations, where the used dimension is 10. We have taken 1 s, which is approximately equivalent to 500 repetitions. These observations are carried out to show the tested methods' time performance. It is evident from Table 3 that the RDA obtained more accurate and reliable results compared to other similar methods when the execution is implemented within the same time period. The same machine and operating system was used in all the simulations.

The limitations of this algorithm stem from its complexity. This is the case with all other meta heuristic techniques that are based on evolutionary methods. Moreover, the algorithms that are based on search populations are parallel in nature, and applying them on a sequential machine with a sequential programming language, as it is the case with most of our simulations, adds more limitations on the RDA capabilities to run faster and more efficiently. On the other hand, adding local operators to enhance exploitation capabilities, may induce more complexity and reduce execution speed. It is always the optimum trade off we try to seek between speed and accuracy. Optimum tuning the parameters for the algorithm is another challenge. Adaptive operators for tuning are beneficial but they add more burden on the complexity and speed.

5 Conclusion and future trends

This paper presented a study for a relatively new algorithm called the Red Deer Algorithm (RDA) (first introduced in 2016). It tackled aspects such as its procedure, applications, and optimization results. All the papers that have been published until this minute and all presented applications of this algorithm are reviewed in this work. These studies focused on various topics, showing the ability of the Red Deer Algorithm in solving various types of problems.

The main contribution of this work was to present all applications available in the literature until this minute for the RDA. Those applications include solutions for engineering, computer science, and business problems. The paper also covered all variants and hybrids presented so far in the literature for the RDA. Moreover, deep analyses of the algorithm based on optimizing standard benchmark mathematical functions of different dimensions and modes were conducted. It showed competitive performance for the RDA compared to other more famous and well-known optimization techniques of different types. The convergence behavior of the RDA was presented and its potential for further use in different applications was apparent. Section 3 summarized the applications and the different variants and hybrids of the RDA.

Table 3 The results of the tested algorithms on 4 problems (F1–F4) through 1 second

Function	Measure	**Algorithms**							
		AOA	PSO	AO	SCA	MPA	GWO	WOA	RDA
F1	Worst	1.88E−02	2.29E+00	2.71E−02	7.65E+01	6.53E−28	7.63E−06	1.32E+03	2.32E−182
	Average	9.85E−03	1.64E+00	1.34E−02	1.91E+01	1.63E−28	1.97E−06	8.45E+02	5.80E−183
	Best	1.67E−05	9.76E−01	2.07E−03	4.00E−147	2.84E−196	1.49E−63	3.76E−01	0.00E+00
	Rank	4	6	5	7	2	3	8	1
F2	Worst	2.51E−02	4.95E+00	2.50E−59	1.73E+00	2.63E−02	2.27E−04	1.00E+01	1.29E−10
	Average	1.73E−02	4.36E+00	6.26E−60	5.22E−01	1.52E−02	7.93E−05	6.40E+00	5.09E−11
	Best	1.28E−02	3.69E+00	8.26E−201	9.34E−18	2.75E−03	1.01E−07	4.13E+00	2.12E−42
	Rank	5	7	1	6	4	3	8	2
F3	Worst	3.98E−02	7.15E+00	9.07E−08	8.53E+03	2.70E−02	5.45E+03	3.26E+03	2.09E−02
	Average	2.24E−02	5.91E+00	2.27E−08	3.98E+03	6.74E−03	4.02E+03	1.97E+03	1.66E−02
	Best	1.24E−02	4.52E+00	2.13E−50	9.93E+02	1.65E−14	1.39E+03	6.78E+02	1.39E−02
	Rank	4	5	1	7	2	8	6	3
F4	Worst	9.01E−02	1.07E+00	1.10E−34	1.62E+01	8.97E−02	1.42E+01	2.29E+01	3.82E−11
	Average	4.18E−02	9.37E−01	2.74E−35	7.73E+00	6.22E−02	6.28E+00	1.74E+01	9.63E−12
	Best	6.61E−03	8.14E−01	4.22E−107	1.28E−02	3.78E−02	1.21E+00	1.19E+01	6.70E−33
	Rank	3	5	1	7	4	6	8	2
Final ranking		4	6	1	7	3	5	8	1

Based on the experimentation done in Sect. 4, the RDA proved to be competitive to comparative methods when applied to optimize different types of benchmark functions. Those functions had different modes, with multiple variables and many local minima. The standard *F* test showed that the RDA is ranked 4 among the comparative algorithms. Most of those algorithms are well-known and frequently used in different applications. Most of those algorithms, if not all, are metaheuristic-based algorithms that could be swarm intelligence-based, local search-based, physical-based, chemical-based, human-based, and others. The RDA proved its presence among them. The RDA, based on its procedure, behavior, and simulation results, showed exploration and exploitation capabilities.

Future work may include investigating other optimization techniques such as rank-constrained spectral clustering with flexible embedding framework (Li et al. 2018). In this technique, irrelevant information and noise are suppressed using a flexible embedding scheme used to unravel the intrinsic cluster structure. Other techniques (Li et al. 2019), exploit the semantic correlation between the concept and an event in case their was inadequacy of information for optimization or learning. In Li et al. (2018), the RDA can have interesting application in optimizing the affinity weights for solving the problem of selecting optimum features for computer vision problems.

The door is wide open for further applications for the algorithm. Starting from mathematical optimization, scheduling, all disciplines of engineering search and optimization applications; discrete or continuous, optimum control, trajectories tracking, all recognition problems, applications in business, management, operation research, inventory, including machine learning, training and learning, optimum features selection, computer science problems such as routing, optimum configurations, and the list could be extended more and more. The fact that the algorithm population members values could be mapped to any domain of continuous or discrete variables makes it highly applicable in many fields.

Finally, the RDA is still a a new and rich algorithm to explore and study. Many variants, adaptive, and hybridized versions of the algorithm yet to be discovered and explore. Future work will include improvising variants of this algorithm and applying it to engineering problems in optimization and machine learning. As with all other meta-heuristic techniques, the RDA needs more theoretical analysis and mathematical modeling using suitable stochastic processing techniques. It will become imperative to understand the behavior of these algorithms and be able to prove their convergence under certain conditions. The RDA is another artifact that came as a result of the capabilities of researchers to observe God-created behavior and to model it in a manner to serve science and human.

References

Abualigah L, Diabat A (2021) Advances in sine cosine algorithm: a comprehensive survey. Artif Intell Rev 20:1–42

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609

Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-qaness MA, Gandomi AH (2021) Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng 157:107250

Alkoffash MS, Awadallah MA, Alweshah M, Zitar RA, Assaleh K, Al-Betar MA (2021) A non-convex economic load dispatch using hybrid salp swarm algorithm. Arab J Sci Eng 20:1–20

Al-Muhammed MJ, Zitar RA (2018) Probability-directed random search algorithm for unconstrained optimization problem. Appl Soft Comput 71:165–182

Alomari OA, Makhadmeh SN, Al-Betar MA, Alyasseri ZAA, Doush IA, Abasi AK, Awadallah MA, Zitar RA (2021) Gene selection for microarray data classification based on gray wolf optimizer enhanced with triz-inspired operators. Knowl-Based Syst 223:107034

Alotaibi SS (2021) Ensemble technique with optimal feature selection for Saudi stock market prediction: a novel hybrid red deer-grey algorithm. IEEE Access 20:20

Al-Qaness MA, Ewees AA, Fan H, Abualigah L, Abd Elaziz M (2020) Marine predators algorithm for forecasting confirmed cases of covid-19 in Italy, USA, Iran and Korea. Int J Environ Res Public Health 17(10):3520

Alsalibi B, Abualigah L, Khader AT (2021) A novel bat algorithm with dynamic membrane structure for optimization problems. Appl Intell 51(4):1992–2017

Ambareesh S, Madheswari AN (2021) Hrdss-wmsn: a multi-objective function for optimal routing protocol in wireless multimedia sensor networks using hybrid red deer salp swarm algorithm. Wirel Personal Commun 20:1–30

Berghman L, Briand C, Leus R, Lopez P (2015) The truck scheduling problem at crossdocking terminals-exclusive versus mixed mode. In: 4th international conference on operations research and enterprise systems (ICORES 2015), p 247

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35(3):268–308

Boysen N, Fliedner M (2010) Cross dock scheduling: classification, literature review and research agenda. Omega 38(6):413–422

Brammya G, Praveena S, Ninu Preetha N, Ramya R, Rajakumar B, Binu D (2019) Deer hunting optimization algorithm: a new nature-inspired meta-heuristic paradigm. Comput J 20:20

Burke EK, Gendreau M, Hyde M, Kendall G, Ochoa G, Özcan E, Qu R (2013) Hyper-heuristics: a survey of the state of the art. J Oper Res Soc 64(12):1695–1724

Dalbah LM, Al-Betar MA, Awadallah MA, Zitar RA (2021) A modified coronavirus herd immunity optimizer for capacitated vehicle routing problem. J King Saud Univ Comput Inf Sci 20:20

De S, Dey S, Debnath S, Deb A (2020) A new modified red deer algorithm for multi-level image thresholding. In: 2020 fifth international conference on research in computational intelligence and communication networks (ICRCICN), IEEE, pp 105–111

Dey S, De S, Deb A, Debnath S (2021) Multilevel image segmentation using modified red deer algorithm. In: 2021 11th international conference on cloud computing, data science and engineering (confluence), IEEE, pp 362–368

Dogani A, Dourandish A, Ghorbani M, Shahbazbegian MR (2020) A hybrid meta-heuristic for a bi-objective stochastic optimization of urban water supply system. IEEE Access 8:135829–135843

Eberhart R, Kennedy J (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, vol 4, Citeseer, pp 1942–1948

Eid A, Kamel S, Abualigah L (2021) Marine predators algorithm for optimal allocation of active and reactive power resources in distribution networks. Neural Comput Appl 20:1–29

Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190

Fard AF, Hajiaghaei-Keshteli M (2016) Red deer algorithm (RDA); a new optimization algorithm inspired by red deers' mating. In: International conference on industrial engineering, IEEE (2016e), pp 33–34

Fathollahi-Fard AM, Niaz Azari M, Hajiaghaei-Keshteli M (2019) An improved red deer algorithm to address a direct current brushless motor design problem. Sci Iran 20:20

Fathollahi-Fard AM, Ahmadi A, Sajadieh MS (2020) An efficient modified red deer algorithm to solve a truck scheduling problem considering time windows and deadline for trucks' departure. Evol Comput Schedul 20:137–167

Fathollahi-Fard AM, Hajiaghaei-Keshteli M, Tavakkoli-Moghaddam R (2021) Red deer algorithm (RDA): a new nature-inspired metaheuristic. Soft Comput 20:1–29

Fausto F, Reyna-Orta A, Cuevas E, Andrade ÁG, Perez-Cisneros M (2020) From ants to whales: metaheuristics for all tastes. Artif Intell Rev 53(1):753–810

Fazli M, Fathollahi-Fard AM, Tian G (2019) Addressing a coordinated quay crane scheduling and assignment problem by red deer algorithm. Int J Eng 32(8):1186–1191

Hussain K, Salleh MNM, Cheng S, Shi Y (2019) Metaheuristic research: a comprehensive survey. Artif Intell Rev 52(4):2191–2233

Li Z, Nie F, Chang X, Nie L, Zhang H, Yang Y (2018) Rank-constrained spectral clustering with flexible embedding. IEEE Trans Neural Netw Learn Syst 29(12):6073–6082

Li Z, Nie F, Chang X, Yang Y, Zhang C, Sebe N (2018) Dynamic affinity graph construction for spectral clustering using multiple features. IEEE Trans Neural Netw Learn Syst 29(12):6323–6332

Li Z, Yao L, Chang X, Zhan K, Sun J, Zhang H (2019) Zero-shot event detection via event-adaptive concept relevance mining. Pattern Recogn 88:595–603

Lourenço HR, Martin OC, Stützle T (2003) Iterated local search. Handbook of metaheuristics. Springer, Berlin, pp 320–353

Mack GA, Skillings JH (1980) A friedman-type rank test for main effects in a two-factor anova. J Am Stat Assoc 75(372):947–951

Meraihi Y, Ramdane-Cherif A, Acheli D, Mahseur M (2020) Dragonfly algorithm: a comprehensive review and applications. Neural Comput Appl 20:20

Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98

Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67

Nguyen GN, Le Viet NH, Devaraj AFS, Gobi R, Shankar K (2020) Blockchain enabled energy efficient red deer algorithm based clustering protocol for pervasive wireless sensor networks. Sustain Comput Inform Syst 28:100464

Osman IH, Laporte G (1996) Metaheuristics: a bibliography

Premkumar M, Jangir P, Kumar BS, Sowmya R, Alhelou HH, Abualigah L, Yildiz AR, Mirjalili S (2021) A new arithmetic optimization algorithm for solving real-world multiobjective cec-2021 constrained optimization problems: diversity analysis and validations. IEEE Access 20:20

Ren P, Xiao Y, Chang X, Huang P-Y, Li Z, Chen X, Wang X (2020) A comprehensive survey of neural architecture search: Challenges and solutions. arXiv:2006.02903 (arXiv preprint)

Safaldin M, Otair M, Abualigah L (2021) Improved binary gray wolf optimizer and svm for intrusion detection system in wireless sensor networks. J Ambient Intell Humaniz Comput 12(2):1559–1576

Shehab M, Alshawabkah H, Abualigah L, Nagham A-M (2020) Enhanced a hybrid moth-flame optimization algorithm using new selection schemes. Eng Comput 20:1–26

Yan C, Chang X, Luo M, Zheng Q, Zhang X, Li Z, Nie F (2020) Self-weighted robust lda for multiclass classification with edge classes. ACM Trans Intell Syst Technol 12(1):1–19

Zhou B, Zong S (2021) Adaptive memory red deer algorithm for cross-dock truck scheduling with products time window. Eng Comput 20:20

Zitar RA, Al-Betar MA, Awadallah MA, Doush IA, Assaleh K (2021) An intensive and comprehensive overview of jaya algorithm, its versions and applications. Arch Comput Methods Eng 20:1–30

Zuluaga JPS, Thiell M, Perales RC (2017) Reverse cross-docking. Omega 66:48–57