Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2019.Doi Number

Nuclear Reaction Optimization: A novel and powerful physics-based algorithm for global optimization

Zhenglei Wei, Changqiang Huang, Xiaofei Wang, Tong Han, Yintong Li
Institute of Aeronautics Engineering, Air Force Engineering University, Xi'an, Shaanxi 710038, China

Corresponding author: Zhenglei Wei (e-mail: zhenglei_wei@ 126.com).

This work was supported by the National Natural Science Foundation of China (51579209 & 61601505), the Aeronautical Science Foundation of China (20175196019) and the Natural Science Foundation of Shanxi Province (2017JM6078).

ABSTRACT Meta-heuristic algorithms have gained substantial popularity in recent decades and have focused on applications in a wide spectrum of fields. In this paper, a new and powerful physics-based algorithm named nuclear reaction optimization (NRO) is presented. Meanwhile, NRO imitates the nuclear reaction process and consists of two phases, namely, a nuclear fission (NFi) phase and a nuclear fusion (NFu) phase. The Gaussian walk and differential evolution operators between nucleus and neutron are employed for exploitation and appropriate exploration in the (NFi) phase, respectively. Meanwhile, the variants of differential evolution operator are utilized for exploration in the NFu phase, which consists of the ionization and fusion stages. Additionally, variants of Levy flight are used for random searching to escape from the local optima in each stage of NFu phase. The exploration and exploitation abilities of NRO can be balanced due to a combination of the two phases. Both constrained and unconstrained benchmark functions are employed for testing the performance of NRO. To make comparisons between NRO and the state-of-the-art algorithms, twenty-three classic benchmark functions and twenty-night modern benchmark functions are performed. Moreover, three engineering design optimization problems are solved as constrained benchmark functions by using NRO and the compared algorithms. The results illustrate that the proposed nuclear reaction optimization algorithm is a potential and powerful approach for global optimization.

INDEX TERMS Differential evolution operator, engineering design optimization, global optimization, Levy flight strategy, nuclear reaction optimization (NRO), physics-based algorithm

I. INTRODUCTION

As the complexity of optimization problems is increasing sharply with the need for optimization algorithms' computational accuracy and convergence rate is gradually rising, optimization algorithms are more important in artificial intelligence fields [1]. Optimization algorithms include stochastic and deterministic algorithms [2]. However, when addressing high-dimensional optimization problems with many local optimum values, deterministic methods (called gradient-based algorithms) that use objective function derivations do not search the solution space effectively or are not applicable [2].

During the last decade, many stochastic algorithms inspired by natural systems have been presented to solve a variety of optimization problems [3]. In stochastic algorithms, some possible solutions are initially selected, then, during an iterative process, this population searches for the solution domain without substantial gradient information until a satisfactory solution is found or the maximum number of iterations is reached [2].

During the past few decades, researchers have proposed several meta-heuristic approaches for stochastic algorithms. Meanwhile, recent literatures present the broad classification of nature-inspired algorithms [4][5], which can be categorized as evolutionary algorithms (EAs) [6], swarm intelligence (SI) algorithms [7][8], human behavior-based algorithms and physics-chemical systems-based algorithms [9][10]. The evolutionary algorithms inspired by natural evolution have attracted many researchers' attention at the initial stage of optimization algorithm development. The typical representatives of EAs are the genetic algorithm (GA), which is inspired by Darwin's theory of biological evolution [11], and the differential evolution (DE) algorithm, which is similar to GA and utilizes the mutation, cross and selection operators to find the optimum value [12]. Besides, various additional EAs have been proposed, such as genetic programming (GP) [13], evolutionary programming (EP) [14], evolution strategy (ES) [15] and the bird mating optimizer (BMO) [16]. Recently, more state-of-the-art EAs emerged, such as backtracking search optimization algorithm (BSA) [17], forest optimization algorithm (FOA) [18], interior search algorithm (ISA) [19], Jaya algorithm [20], rain forest algorithm (RFA) [21] and competitive optimization algorithm (COOA) [22].

The SI algorithms are enlightened by the processes of decentralized and self-organized systems, which can adjust the balance between exploration and exploitation [23][24]. The most representative algorithms inspired from natural creatures' colonies include particle swarm optimization (PSO) [25], which imitates the intelligent social behavior of bird flock; ant colony algorithm (ACO) [26], which finds the optimum path of ant colony by pheromones; artificial bee colony (ABC) algorithm [27] inspired by bees' foraging behavior; the bat algorithm (BA) [28] inspired by the echolocation behavior of bats; the cuckoo search (CS) [29] inspired by creatures' parasitic interactions and grey wolf optimizer (GWO) [30] inspired by grey wolf pack encircling, hunting and attacking prey. Especially in recent decades, many novel SI algorithms have been proposed and applied to engineering optimization problems, such as the grasshopper optimization algorithm (GOA) [31], the slap swarm algorithm (SSA) [32], the butterfly-inspired algorithm [33], the whale optimization algorithm (WOA) [34], the squirrel search algorithm [35], the symbiotic organisms search (SOS) [36] and the spider monkey optimization (SMO) [37].

The third main branch of meta-heuristics is human behavior-based algorithms inspired by the social behavior of human beings. The teaching-learning based optimization (TLBO) algorithm [38], which simulates teaching and learning behaviors, is one of the most popular algorithms [39]. The newest human behavior-based algorithm is the socio evolution and learning optimization (SELO) algorithm, which is inspired by the social learning behavior of society and families [40]. Recent literatures have proposed several popular human behavior-based algorithms, which include the cognitive behavior optimization algorithm (COA) [41], human mental search (HMS) [42], the ideology algorithm [43], social learning optimization (SLO) [44] and social group optimization (SGO) [45].

The physics-chemical systems-based algorithms are inspired from physical phenomena or chemical reactions. The gravitational search algorithm (GSA) [46] is a typical representative physics-based algorithm, which is inspired by the laws of gravity and mass interactions and attracts many researchers to improve its performance [47][48]. Chemical reaction optimization (CRO) [10] is an alternative popular approach, which mimics the process of a chemical reaction. Other novel physics-chemical systems-based algorithms are proposed in recent literature, such as the multi-verse optimizer (MVO) [49], the mine blast algorithm (MBA) [50], the lightning search algorithm (LSA) [51], the water evaporation optimization (WEO) algorithm [52], the charged system search (CSS) algorithm [53], the ray optimization (RO) algorithm [54], colliding bodies optimization (CBO) [55], simulated annealing (SA) [11], the big bang-big crunch (BB-BC) algorithm [56], thermal exchange optimization (TEO) [57], the vibrating particles system (VPS) algorithm [58], atom search optimization (ASO) [59], electro-search algorithm (ESA) [60] and vortex search algorithm (VSA) [61], which is a new individual-based algorithm based on vortex patterns. In recent years, some novel algorithms which have adopted ideas based on the laws of physics and chemistry have been proposed. Artificial chemical reaction optimization algorithm (ACROA) [62] utilizes a variety of chemical reactions to simulate the optimization process. Galactic swarm optimization (GSO) [63] is proposed based on the motion of stars inside galaxies and superclusters of galaxies in the cosmos. Electromagnetic field optimization (EFO) [64] is inspired by the behavior of electromagnets with different polarities and the law of golden ratio. Supernova optimizer (SO) [65] mimics the supernova phenomena. Sonar inspired optimization (SIO) [66] uses the underwater acoustics phenomena to perform wider range of search. In this paper, a novel physics-based algorithm is proposed and detailed in the following sections.

Based on the search direction, the use of memory, the type of neighborhood exploration used or the number of current solutions during iteration process, stochastic algorithms are divided into two types: population-based and individual-based [67]. In addition to SA and VSA, tabu search (TS), iterated local search (ILS), guided local search (GLS), pattern search (PS) and variable neighborhood search (VNS) algorithms are individual-based algorithms [61]. Although individual-based methods possess better exploitation, only a few solutions do not share information and allow algorithms to easily fall into the local optima. For example, VSA has poor memory requirements and does not use other solutions of a vortex circle, so the algorithm falls into a local optimum. However, population-based algorithms can generate diverse solutions randomly and improve the search capacity through various operators during the optimization process. Compared with individual-based algorithms, diverse solutions can enhance the diversity of a population, widen the scope of the searching space, and share various individuals' information reciprocally, which assist population-based algorithms to avoiding getting caught in a local optimum. Meanwhile, because of increasing population size, population-based algorithms must increase computational complexity and time [2]. Therefore, the main strategic point of a novel meta-heuristic algorithm must concern effective, particular and diplomatic techniques, which can produce algorithms with a balance between exploration and exploitation [67].

For many population-based algorithms, adjusting the parameters for balancing between exploration and exploitation is critical. While exploration can search the solution space globally to guarantee solution diversity and escape the local optima, exploitation can find better solutions in the neighboring region of current agents to improve the speed of an algorithm [2]. Numerous algorithms can adjust the critical parameters of strategic methods to control the functions of exploration and exploitation during iterations. For example, the GWO algorithm searches the optima by tuning the control coefficients, which is initially a large value for exploration and then gradually decreases for better exploitation [30]. In addition to the tuning parameter method for creating a balance between exploration and exploitation, distributing the tasks of exploration and exploitation can determine the quality of the solutions. For example, in the hummingbirds optimization algorithm (HOA), a better solution can be considered for searching locally and a worse solution that possesses a lower evaluation value can perform a global search [2]. Therefore, weighing the balance between exploitation and exploration should be considered in the process of designing a novel algorithm.

The differential evolution (DE) algorithm is hailed as a very effective global optimizer, which has received warm praise from researchers in various fields. Because the DE algorithm possesses simple and efficient operators, which include mutation, cross and selection, there are a variety of derivative algorithms that are improved and compared with DE and other state-of-the-art algorithms, such as the strategy adaption self-adaptive differential evolution (SADE) algorithm [68] and the success-history-based parameter adaptation differential evolution algorithm using linear population size reduction (L-SHADE) [69]. In recent years, besides classical variants of DE, many researchers attempt to employ different strategies to improve the performance of DE. Many newly proposed DE algorithms are very powerful and efficient, such as adaptive success-based tribal DE algorithm with dynamic population reduction (*s*TDE-*d*R) [70], MODE with reference axis vicinity mechanism (AMODE-RAVM) [71], abstract convex underestimation-assisted multistage DE (UMDE) [72] and DE with underestimation-based multi-mutation strategy (UMS) [73]. Meanwhile, although a few meta-heuristic algorithms are inspired from natural systems, there are DE core operators applied to these algorithms. Stochastic fractal search (SFS) [74] is a powerful meta-heuristic algorithm for standard benchmark functions. Meanwhile, it is partially inspired by the DE algorithm. The SFS algorithm's updating process adopts the improved mutation and cross. In the water cycle algorithm (WCA) [75], a mutation operator of DE is used in the process of rivers flowing to the sea. During the iterations of the three phases (mutualism, commensalism and parasitism) of the SOS algorithm [36], the variants of differential evolution operators are applied for generating various populations. As a result, we propose a novel algorithm that adopts variants of differential evolution operator.

Despite the trend that various novel algorithms appear similar to mushrooms after the rain, the no free lunch (NFL) theorem [76] has proven that no single algorithm is best for all optimization problems. Although a few algorithms can solve some optimizations powerfully, these algorithms may perform poorly in other optimization problems. Because of this, the NFL theorem motivates researchers to create novel algorithms, enhance current methods and generate combinative approaches [30]. This is a very vital reason why to develop a novel meta-heuristic algorithm inspired by nuclear reactions.

This research paper aims to introduce a novel and powerful meta-heuristic optimization algorithm called the nuclear reaction optimization (NRO). Based on the analysis of the classic methods of many meta-heuristic algorithms such as DE, SOS and SFS, the excellent strategy of searching is summarized as the process of diversification-interaction-convergence in NRO. Inspired by the nuclear energy freed during the nuclear reaction process, the closed space is full of the various atomic nuclei built for nuclear reactions. First, to establish a framework for NRO, the rules of nuclear fission and fusion processes are presented by analyzing the characteristics of each process. Second, the NRO algorithm can be assumed to perform very well within two crucial phases—nuclear fission and nuclear fusion. The nuclear fission (NFi) phase is inspired by the process of heavy nucleus fission, consisting of three states for fragments of bombardment. For odd nuclei, the Gaussian walk and variants of differential evolution operators are employed to imitate the β decay that can regulate the ability of exploration and exploitation. For even-even nuclei, the Gaussian walk is used to search the neighborhood of current positions. The nuclear fusion (NFu) phase is inspired by the process of light nuclei fusion representing exploitation, which includes the ionization and fusion stages. In the ionization stage, according to the rank of a nucleus' fitness value, the nucleus with a better fitness value is retained as guidance information to enhance the exploitation of the NFu phase and the nucleus with a poor fitness value is employed to improve the exploration under each nucleus variable. In the fusion stage, this procedure performs three states according to the ranking of the nucleus fitness value and employs the variants of differential operators for better exploration of the whole nucleus. To escape from the local optimum, Levy flight is introduced in each stage of the NFu phase. Three types of tests are performed and the results prove the efficiency of our proposed algorithm due to a balance of exploration and exploitation.

The remainder of the paper is organized as follows: Section 2 reviews the nuclear reaction theory. Section 3 depicts how the NFi and NFu models can be used to create a powerful and novel physics-based optimization approach. Then, the comparative experiments on different benchmark tests are presented in Section 4. Finally, in Section 5 we present our conclusion.

II. NUCLEAR REACTION THEORY

Following the discovery of the electron by Joseph John Thomson at the Cavendish laboratory in Cambridge, the discovery of nuclear fission based on numerous events has significantly influenced the contemporary framework [77]. In 1912, the nucleus regarded as the central region of concentrated positive electrical charge, was discovered by Rutherford [78]. The nucleus is a part of the atom; another distinct part is the electron, which can discriminate the different elements. According to the structure of different elements, Rutherford created the artificial transmutation of one element into another in 1919 [79]. The neutron obtained from boron and nitrogen was found and reported by Chaswick [77]. After the discovery of the neutron, from 1934 to 1939, research of the reaction between neutron and uranium was implemented. Since then, the concept of a nuclear reaction started to appear in the human domain [77].

As the techniques in the nuclear field are emerging, although there are potential dangers with nuclear reactors, some scientists believe that nuclear power has become an indispensable part of green energy sources. The discoveries of the neutron and elements artificial transmutation are important foundations of a nuclear reaction, which can be divided into nuclear fission and nuclear fusion [80]. Heavy nucleus fission is an extremely complex process, during which a heated neutron bombards the heavy nucleus and the nucleus of this atom breaks into two or more lighter nuclei as fission products and other byproduct particles. Because of the mass defect that is the result of the difference between the mass of an atom and the sum of the masses of the fission fragments, fission of a heavy nucleus can release a great deal of energy both as gamma rays and as the kinetic energy of other nuclear fragments such as beta and alpha particles [77]. In the process of fission, according to the liquid-drop model, the nucleus in an excited state that absorbs a heated neutron would be deformed gradually, and a part of the excitation energy is rendered into deformation potential energy, which first increases and then decreases with the nucleus stretching gradually. Because of two determinants that include the surface energy of the nuclear force and the Coulomb energy of the electrostatic repulsion, a nucleus can be in the fission by crossing the fission barrier height, which is the energy difference between the ground state and the highest saddle point [81]. Another concept is binding energy, which is relative to the mass defect according to Einstein's mass-energy equation. When the number of protons which must be greater than 56 increases, it is more possible for nuclear fission. The binding energy per mass that represents the nucleus cross the fission barrier height determines whether nuclear fission is executed [77]. After nucleus fission, the fission fragments release neutrons and become primary fission products without β decay or secondary fission products after β decay. Meanwhile, for fission caused by a low-energy neutron, an odd nucleus can be in excess of the fission barrier height and an even-even nucleus cannot reach the fission barrier height. As can be seen in Fig. 1, the heated neutrons bombard the heavy nuclei and generate new neutrons for bombarding other heavy nuclei. This process is called the chain reaction of nuclear fission.

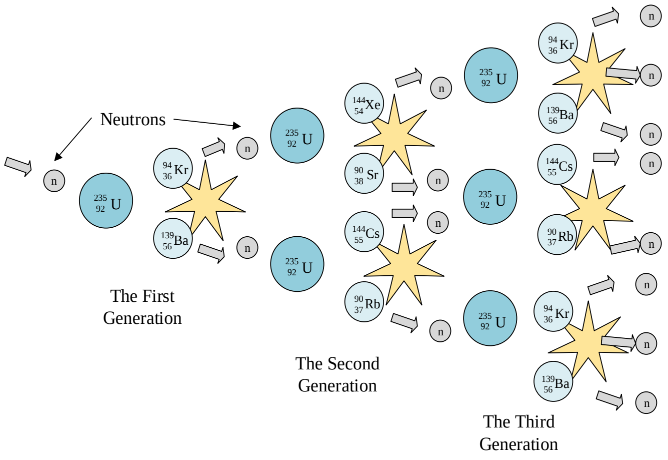


FIGURE 1. *The process of nuclear fission.*

Compared with nucleus fission, nucleus fusion is more difficult to implement, although the energy of fusion is more promising. In 1932, Mark Oliphant achieved firstly the laboratory fusion of hydrogen isotopes [82]. Concurrently, the discovery of the cycle of nuclear fusion in stars was published worldwide. While the number of protons decreases and becomes less than 56, the binding energy per mass decreases. The energy released by the fusion results from the reciprocity of the short-range nuclear force and the Coulomb force. As shown in Fig. 2, fusion can occur under this condition, in which the nuclei get close enough and the strong nuclear force plays a crucial role that binds nuclear particles together and overcomes the Coulomb repulsion force. A fusion reaction could be implemented in a confined volume with appropriate parameters such as the reaction cross-section and the energy distribution of the nuclei [82]. In general, when the nuclei are heated to the plasma state in a confined volume, the high kinetic energy of particles can cause fusion.

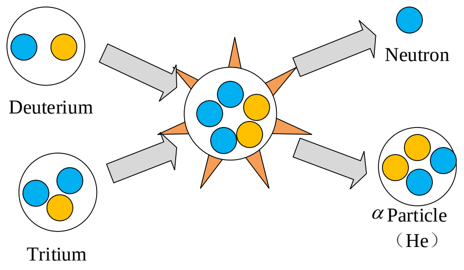


FIGURE 2. **The process of nuclear fusion.**

According to the above illustration, the proposed method has employed both nuclear fission and nuclear fusion theories. In summary, NRO employs several strategic and simple rules to search for an optimal solution:

1. In the nuclear fission process, the nuclei fragments after absorbing heated neutrons become odd nuclei or even-even nuclei. The odd nuclei can be divided into primary fission products that can be employed for exploitation or secondary fission products that can be used for exploration. The even-even nuclei that are not in the fission can search in the neighborhood of the current positions.
2. Each nucleus is heated to the plasma state by absorbing the energy from nuclear fissions.
3. Some nuclei that overcome the Coulomb repulsion force and can be bound together by strong nuclear force can play a role of exploration.
4. Other nuclei that are restrained by the Coulomb repulsion force reduce the approaching velocity for exploitation or repel each other for exploration.
5. The movement energy of each nuclei stems from the heated neutron or energy released in the nuclear reaction.

In this paper, based on above rules, we proposed a novel and powerful physics-based algorithm, called the nuclear reaction optimization. The inspirations for this new algorithm stem from nuclear reactions, including fission and fusion. According to the nuclear reaction theory, there are two phases in the process of optimization—NFi and NFu. To imitate these two pivotal phases in same searching space, there is a pivotal assumption that the heavy nucleus fission (first) and light nucleus fusion (second) occur in a confined volume. Overall, the NFi phase portrays exploitation for seeking a better solution in the vicinity of a current optimal solution, and the NFu phase performs exploration for searching globally. However, each phase can possess different strategies to balance the relation between exploitation and exploration. Because of information sharing between particles in the nuclei population, an optimization algorithm could get caught in the local optimum. Therefore, a fairly effective strategy called Levy flight (which is described in detail in the next section) is presented for assisting NRO in escaping from the local optimum.

III. NUCLEAR REACTION OPTIMIZATION

As discussed previously, a physics-based algorithm inspired by nuclear reactions employs the NFi and NFu phases for exploitation and exploration of a search solution space. Because of the operational principles from nuclear reactions, the algorithm is called nuclear reaction optimization (NRO). The NRO algorithm assumes that whole search space is in a sealed container where all nuclei interact with each other even if nuclear fissions and nuclear fusions cannot function by employing the same nuclei inside the same container. In reality, energy released from nuclear fissions can heat the lighter nuclei to cause nuclear fusions and thermal neutrons generated by nuclear fusions can bombard the heavy nuclei to cause nuclear fission. Hence, NRO assumes that the cycle in which the NFi phase functions first and then the NFu phase is activated could be feasible. As can be seen in Fig. 3, NRO employs the cycle propelled by fission energy and fusion neutrons for searching for the most stable nucleus as an optimal fitness value. In the proposed algorithm, each solution within the search space represents a nucleus characteristic that can include items such as the mass number of the nucleus, the charge property, position, potential energy and kinetic energy. Each item of a nucleus characteristic represents each variable of a solution, which could be somewhat similar to the structure of a molecule as a solution in CRO algorithm [10]. Every nucleus is estimated by the specific binding energy (the binding energy per mass) that represents the stability of the nucleus, and the most stable nucleus is the optimal solution. The NRO algorithm models two pivotal search phases: NFi and NFu. According to the reaction theory, each nucleus is guided by a series of operators that mimic different states of reactions and can escape from the local optimum if possible.

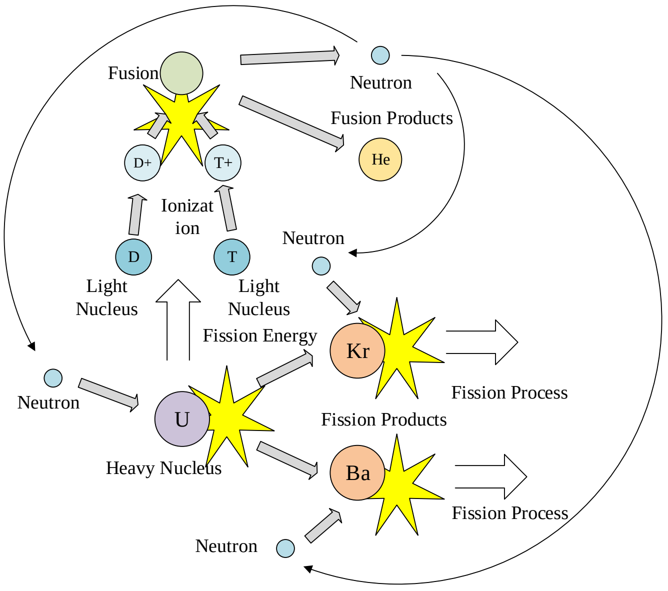


FIGURE 3. The assumption for circulation in processes of nuclear fission and nuclear fusion.

3.1 INITIALIZATION OF THE NUCLEI POPULATION

Initially, similar to other meta-heuristic algorithms, in the NRO algorithm a uniformly distributed population of N nuclei is randomly generated; each nucleus individual $\boldsymbol{X}_i \, (i \in \{1,2,3,...,N\}$ is a D-dimensional vector, where N is the population size. Each nucleus represents the variables in the optimization problem and each nucleus corresponds to a potential solution for the optimization problem within the search space. The ith nucleus in the population is initialized by following formula:

$$\boldsymbol{X}_{i,d} = \boldsymbol{lb}_d + rand \cdot (\boldsymbol{ub}_d - \boldsymbol{lb}_d) \qquad (1)$$

where $\boldsymbol{X}_i$ is the state of the ith nucleus of the population in a confined volume, $\boldsymbol{lb}_d$ and $\boldsymbol{ub}_d$ represent the lower and upper bounds of the dth variable in the search space, respectively, and $rand$ is a uniformly distributed random number between 0 and 1.

3.2 NUCLEAR FISSION (NFi) PHASE

To convert the nuclear fission into a mathematical model, the NFi phase employs a Gaussian walk for creating new nuclei after nuclear fission. Antecedent literature [2][74] has alleged that Gaussian and Levy distributions possess separately distinct advantages. While Gaussian walk can find a more promising global optimal value than Levy flight, Levy flight provides faster convergence than Gaussian walk. Therefore, in the NFi phase, Gaussian walk is mainly employed to simulate the different fission fragments with different states for exploiting a generally promising search space. In the NFu phase, Levy flight will be introduced. According to the theory of nuclear fission reaction, a heated neutron bombards the heavy nucleus to prompt the germination of nuclear fission. However, generally, for chain nuclear fission [83], the reaction is sustained through the continuous neutrons created by a previous reaction. Because of the cycle between nuclear fission and fusion, it is assumed that heated neutrons can be generated by the nuclear fusion of two different random nuclei in the following formula:

$$\boldsymbol{Ne}_i = \frac{\left(\boldsymbol{X}_i + \boldsymbol{X}_j\right)}{2} \tag{2}$$

where Ne_i is the ith heated neutron, and $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ represents the ith nucleus and jth nucleus, respectively.

In general, odd nuclei can undergo fission by absorbing free neutrons and generating secondary fission products for modeling exploitation and primary fission products for mimicking exploration, when $rand \le P_{Fi}$ is true where P_{Fi} is the probability of fission. Conversely, even-even nuclei cannot be generally motivated to produce fission when $rand > P_{Fi}$ is true. Regarding the different states of fragments after bombardment, the characteristics of different products are divided into three states, as shown in Fig. 4.

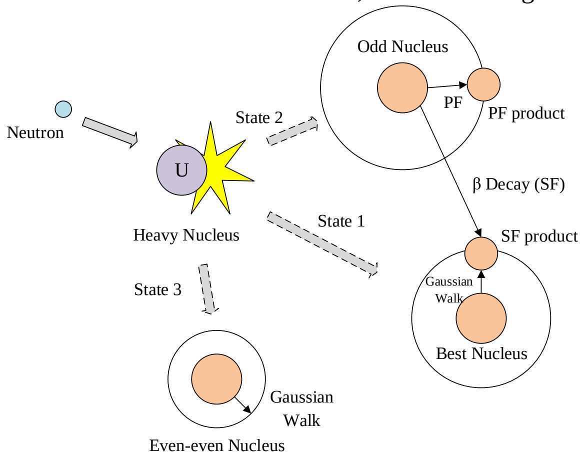


FIGURE 4. The states of fragments after bombardment.

State 1 When odd nuclei undergo fission by absorbing heated neutrons, there are two types of fission products—secondary fission (SF) and primary fission (PF). For the first case, in which the nuclei can produce SF after β decay, these nuclei can be very stable because of the β decay. Therefore, these nuclei employ energy from heated neutrons for activating the nuclei to undergo fissions and approach the more stable state through β decay. Differently, the current ith solution employs the neutron's information and current best solution for producing and searching for a better solution. According to the analysis of Gaussian walk's advantages, the SF product uses Gaussian walk to exploit the neighbor of the current best solution through β decay. Meanwhile, the SF product adopts the difference between the current solution and the heated neutron to exploit the corresponding range. When item $rand \le P_\beta$ is true, in which P_β is the probability of β decay, the creation process of an SF product can be simulated by following:

$$\boldsymbol{X}_i^{Fi} = Gaussian\left(\boldsymbol{X}_{best}, \sigma_1\right) + \left(randn \cdot \boldsymbol{X}_{best} - P_{ne} \cdot \boldsymbol{Ne}_i\right) \tag{3}$$

$$\sigma_1 = \left(\frac{\log(g)}{g}\right) \cdot \left|\boldsymbol{X}_i - \boldsymbol{X}_{best}\right| \tag{4}$$

$$P_{ne} = round\left(rand + 1\right) \tag{5}$$

where X_i^{Fi} is the ith fission product nucleus, $randn$ is a normally distributed random number and X_{best} is the current best nucleus. For the Gaussian distribution, the parameters are X_{best} and σ_1 in which the term $\log(g)/g$ is proposed to balance between exploration and exploitation, where g is the current generation. Meanwhile, the step size σ_1 is calculated by the difference between X_i and X_{best}. Additionally, P_{ne} is a mutation factor that can decide whether the product is to implement exploitation more widely. For P_{ne}, round is nearest integer and rand is uniformly distributed random number between 0 and 1. For **State 1**, the value of P_{ne} in (5) means that SF product can exploit the smaller range of searching.

State 2 A PF product is generated by an odd nucleus after absorbing a heated neutron. For the second state that the fission fragment after absorbing the heated neutron may not undergo β decay, the product nucleus without β decay may not be stable; therefore, its fitness value is lower. The current ith solution employs the information from the Gaussian distribution of X_i to exploit the neighbor of X_i, and adopts the difference between X_{best} and Ne_i to exploit the corresponding range. According to above analysis, the creation of PF can be modeled as follows:

$$\boldsymbol{X}_i^{Fi} = Gaussian\left(\boldsymbol{X}_i, \sigma_2\right) + \left(randn \cdot \boldsymbol{X}_{best} - P_{ne} \cdot \boldsymbol{Ne}_i\right) \tag{6}$$

$$\sigma_2 = \left(\frac{\log(g)}{g}\right) \cdot \left|\boldsymbol{X}_r - \boldsymbol{X}_{best}\right| \tag{7}$$

$$P_{ne} = round\left(rand + 2\right) \tag{8}$$

where the parameters of Gaussian distribution are $\boldsymbol{X}_i$ and σ_2. σ_2 is calculated by the difference between $\boldsymbol{X}_r$ and $\boldsymbol{X}_{best}$. The index r of $\boldsymbol{X}_r$ that is randomly selected from nuclei population is totally different from the index i of current nucleus $\boldsymbol{X}_i$. The difference between $\boldsymbol{X}_r$ and $\boldsymbol{X}_{best}$ is employed for increasing the diversity of nuclei population. For **State 2**, the value of P_{ne} where the term *round* is nearest integer means that PF product can exploit the wider range of searching. As a matter of fact, P_{ne} adjusts the balance between exploitation and exploration to some extent in NFi phase.

State 3 An even-even nucleus cannot be motivated to undergo nuclear fission. Because of the bombardment of the neutron, even if fission has not occurred, the state of the nucleus is changed, i.e., its position and energy. The information of the current nucleus could be retained and stems from the Gaussian walk, which is expressed by the following formula:

$$\boldsymbol{X}_i^{Fi} = Gaussian\left(\boldsymbol{X}_i, \sigma_2\right) \tag{9}$$

Even if there are several possibilities where different nuclei can be created, the corresponding possibility of each product is decided by the value of P_{Fi}. The parameter P_{Fi} of a fission product type is generally determined by the reactants' environment and concentration. However, the value of P_{Fi} is attained through many experiments with the NRO algorithm. The NFi phase assumes that there are two fission nuclei that are lighter than original heavy nucleus if P_{Fi} is a certain possibility according to the general fission process. Another possibility, P_{β}, can determine whether the fission will undergo β decay. Meanwhile, the non-fission product nucleus is assumed to contain the two different states of the current nucleus. For the optimization algorithm, compared with the current nucleus fitness value, the NFi determines whether one of two fission products can replace the current nucleus.

3.3 NUCLEAR FUSION (NFu) PHASE

A nuclear fusion reaction is divided into heated nuclear fusion and cold nuclear fusion based on the reaction conditions. In general, heated nuclear fusion occurs when nuclei are heated to the plasma state and then nuclei could combine to become a heavier nucleus than the original light nuclei. In the first stage, where plasma is created and could be moving at a high velocity, differential operators with different strategies can be applied. In the second stage of fusion, the ions overcome the Coulomb repulsion force and are combined by a strong nuclear force. Meanwhile, other differential operators are employed to search the space and enhance the nuclei population's diversity. Additionally, to avoid the case in which two solutions are similar, Levy flight is employed to assist the current solution in escaping from the local optimal space. With respect to the NFi phase for exploitation, the NFu phase, which can be divided into ionization and fusion stages, primarily makes great contributions to exploration.

3.3.1 IONIZATION STAGE

According to the rules of ionization, this stage is employed for exploration as a part of the NFu phase. In general, the rule assumes that the energy of ionization stems from nuclear fission. For the NRO algorithm, in the ionization stage, the state movement of a nucleus is derived from the thermal energy released from fission. The thermal energy is modeled through two random products—the nuclei and the current nucleus. In this stage, according to ranking of the nuclei's fitness values, the nucleus with a better fitness value is retained as guidance information to enhance the exploitation of the NFu phase and the nucleus with a poor fitness value is employed to improve the exploration.

For the ionization stage, initially, all nuclei are ranked according to their fitness values [41][74]. The ith nucleus is given a probability value that obeys uniform distribution, as follows:

$$Pa_i = \frac{rank\left(fit\boldsymbol{X}_i^{Fi}\right)}{N} \tag{10}$$

where $fit\boldsymbol{X}_i^{Fi}$ is the fitness value of $\boldsymbol{X}_i^{Fi}$, $rank\left(fit\boldsymbol{X}_i^{Fi}\right)$ is considered the rank of the ith nucleus $\boldsymbol{X}_i^{Fi}$ among the other nuclei in the population, and N is the total number of nuclei. The rank of fitness value is from the largest to the lowest. The value Pa determines whether the nucleus is ionized. Pa illustrates that the better nucleus is, the higher its possibility is. In this stage, the nucleus with a poor fitness value possesses a greater possibility of ionization. Alternatively, the nucleus with a better fitness value could have an increasing chance for retaining the good characteristics of this nucleus. For current nucleus $\boldsymbol{X}_i^{Fi}$ in the fusion reaction population, if $Pa_i < rand$, where $rand$ is a uniformly distributed random number between 0 and 1, the dth variable of $\boldsymbol{X}_i^{Fi}$ is ionized based on the following formulas:

$$\boldsymbol{X}_{i,d}^{Ion} = \boldsymbol{X}_{r1,d}^{Fi} + rand \cdot \left(\boldsymbol{X}_{r2,d}^{Fi} - \boldsymbol{X}_{i,d}^{Fi}\right),\ rand \le 0.5 \tag{11}$$

$$\boldsymbol{X}_{i,d}^{Ion} = \boldsymbol{X}_{r1,d}^{Fi} - rand \cdot \left(\boldsymbol{X}_{r2,d}^{Fi} - \boldsymbol{X}_{i,d}^{Fi}\right),\ rand > 0.5 \tag{12}$$

where $\boldsymbol{X}_{i,d}^{Ion}$ represents the dth variable of ith ion after ionization, and $\boldsymbol{X}_{i,d}^{Fi}$, $\boldsymbol{X}_{r1,d}^{Fi}$ and $\boldsymbol{X}_{r2,d}^{Fi}$ represent the dth variables of ith, $r1$th and $r2$th fission nuclei, respectively. According to the value of $rand$, above different strategies that includes (11) and (12) are used for improving the quality of exploration in the case of ionization where $Pa_i > rand$ undergoes. If $Pa_i \le rand$, the fission nucleus with better stability aren't ionized by fission energy and only changes a little or not. In order to improve the performance of exploitation, $\boldsymbol{X}_{i,d}^{Fi}$ is changed by adding a narrow disturbance shown as follows:

$$\boldsymbol{X}_{i,d}^{Ion} = \boldsymbol{X}_{i,d}^{Fi} + round\left(rand\right) \cdot rand \cdot \left(\boldsymbol{X}_{worst,d}^{Fi} - \boldsymbol{X}_{best,d}^{Fi}\right) \quad (13)$$

where $\boldsymbol{X}_{worst,d}^{Fi}$ is the dth variable of the worst fission product nucleus. This stage presents that the nucleus with better fitness value could be retained, and the nucleus with poor fitness value is to be changed by ionization. Overall, the ionization stage can absorb information from different nuclei and can better balance between the exploration and exploitation performance. Meanwhile, if $fit\boldsymbol{X}_i^{Ion} \le fit\boldsymbol{X}_i^{Fi}$ is true for minimum optimization problem, $\boldsymbol{X}_i^{Fi}$ is replaced by $\boldsymbol{X}_i^{Ion}$; otherwise it remains unchanged.

3.3.2 FUSION STAGE

For a heated nuclear fusion reaction, only ionization of light nuclei is not enough; ions with high energy can collide and undergo fusion. Regarding the movement of the variables of the nuclei in an ionization stage, the fusion stage movement is aimed to change the state of an ion combined with information of other ions in the nuclear reaction population. Similar to the ionization procedure, this procedure employs the variants of difference operators for better exploration. Meanwhile, before the fusion stage, all ions obtained from ionization are ranked based on (14), which is similar to (10) and is given as follows:

$$Pc_i = \frac{rank\left(fit\boldsymbol{X}_i^{Ion}\right)}{N} \quad (14)$$

where Pc_i is the probability value of the ith ion and $rank\left(fit\boldsymbol{X}_i^{Ion}\right)$ is considered as the rank of the ith ion $\boldsymbol{X}_i^{Ion}$ among population. The rank of fitness value is from the largest to the lowest. Similar to the first stage, if $Pc_i < rand$ for an ion $\boldsymbol{X}_i^{Ion}$, nuclear fusion which represents **State 1** occurs; otherwise the current characteristics of $\boldsymbol{X}_i^{Ion}$ are updated by **State 2**, where the Coulomb force between ions plays the vital role of balance between exploration and exploitation. The two states are shown in Fig. 5.

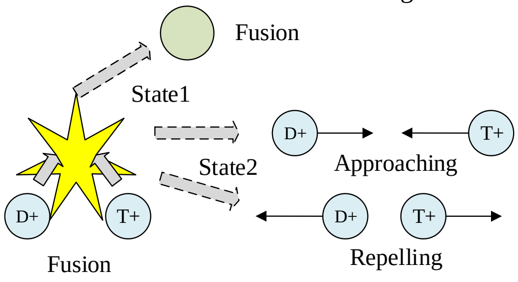


FIGURE 5. Fusion state after ionization.

State 1 Due to the strong nuclear force, the ions of two light nuclei overcome the Coulomb repulsion force and have a collision for fusion. To improve the quality of exploration, different strategies inspired by variants of difference operators are used to mimic the collision and the fusion. Different than the movement of each variable in the ionization stage, the movement of a whole ion in the ionization population employs information of two ions with different indexes. The work of **State 1** is shown in the following formulas:

$$\boldsymbol{X}_i^{Fu} = \boldsymbol{X}_i^{Ion} + rand \cdot \left(\boldsymbol{X}_{r1}^{Ion} - \boldsymbol{X}_{best}^{Ion}\right) + rand \cdot \left(\boldsymbol{X}_{r2}^{Ion} - \boldsymbol{X}_{best}^{Ion}\right) - e^{-norm\left(\boldsymbol{X}_{r1}^{Ion} - \boldsymbol{X}_{r2}^{Ion}\right)} \cdot \left(\boldsymbol{X}_{r1}^{Ion} - \boldsymbol{X}_{r2}^{Ion}\right) \quad (15)$$

where $\boldsymbol{X}_i^{Fu}$ is the ith fusion product, $\boldsymbol{X}_{r1}^{Ion}$ and $\boldsymbol{X}_{r2}^{Ion}$ represent respectively the $r1$th and $r2$th ions, where $r1$ and $r2$ are different and $norm$ is a norm based on population size. The first item $\boldsymbol{X}_i^{Ion}$ in right of (15) represents the basis of X_i^{Fu}, the second item employs the difference between $\boldsymbol{X}_{r1}^{Ion}$ and $\boldsymbol{X}_{best}^{Ion}$, the third item reports the information of another part of fusion through the difference between $\boldsymbol{X}_{r2}^{Ion}$ and $\boldsymbol{X}_{best}^{Ion}$, and the last item with the difference between $\boldsymbol{X}_{r1}^{Ion}$ and $\boldsymbol{X}_{r2}^{Ion}$ means that the ions are to overcome the Comlomb repulsion force. Particularly, in the last item, the exponential coefficient is to increase with distance between $\boldsymbol{X}_{r1}^{Ion}$ and $\boldsymbol{X}_{r2}^{Ion}$ decreasing. This property means the ability of balancing between exploration and exploitation.

State 2 When ions cannot overcome the Coulomb force and fail to fuse together, the Coulomb force is a pivotal factor to manage the characteristic change of the ions. With respect to the fusion formula of **State 1**, to enhance the quality of searching for the optimal solution, this case employs two distinct difference operators to consider both exploration and exploitation. If fusion does not occur, the Coulomb force could reduce the approaching velocity or repulse the opposite movement. The specific mathematic model is proposed as follows:

$$\boldsymbol{X}_i^{Fu} = \boldsymbol{X}_i^{Ion} - 0.5 \cdot \left(\sin\left(2\pi \cdot freq \cdot g + \pi\right) \cdot \frac{G_{\max} - g}{G_{\max}} + 1\right) \cdot \left(\boldsymbol{X}_{r1}^{Ion} - \boldsymbol{X}_{r2}^{Ion}\right), \quad rand > 0.5 \quad (16)$$

$$\boldsymbol{X}_i^{Fu} = \boldsymbol{X}_i^{Ion} - 0.5 \cdot \left(\sin\left(2\pi \cdot freq \cdot g + \pi\right) \cdot \frac{g}{G_{\max}} + 1\right) \cdot \left(\boldsymbol{X}_{r1}^{Ion} - \boldsymbol{X}_{r2}^{Ion}\right), \quad rand \le 0.5 \quad (17)$$

where $\boldsymbol{X}_{r1}^{Ion}$ and $\boldsymbol{X}_{r2}^{Ion}$ represent the $r1$th and $r2$th ions with different indexes, respectively, $freq$ represents the frequency of the sinusoidal function, and g and $G_{\max}$ represent the current generation number and the maximum allowed generation number, respectively. In this case, the $freq$ parameter is set to a value attained by many experiments [84]. In formula (16) and (17), the mutation coefficient of the difference operator employs the non-adaptive sinusoidal adjustment shown in Fig. 6 for mimicking **State 2**. Equation (16) models the reduction of the approaching velocity due to the repulsive force even if two ions are approaching each other. This operator can carry out exploitation for searching a space and converging to the optimal solution. Alternatively, the condition where two ions repel and are far from one another is simulated by (17) for exploration. According to the above analysis of the Coulomb force's function, this mechanism can create a balance between exploration and exploitation. If the fitness value of the new individual $\boldsymbol{X}_i^{Fu}$ is higher than the fitness value of $\boldsymbol{X}_i^{Ion}$, $\boldsymbol{X}_i^{Ion}$ is replaced by $\boldsymbol{X}_i^{Fu}$, otherwise it retains its original variables.

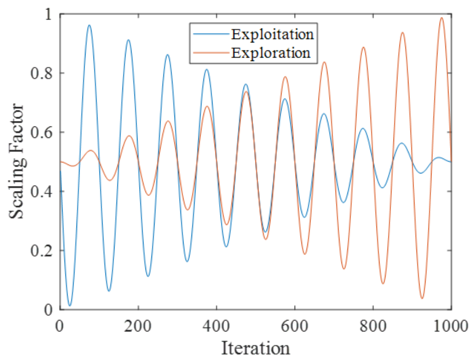


FIGURE 6. Sinusoidal decreasing and increasing adjustment configuration.

3.3.3 LEVY DISTRIBUTION STRATEGY

For an overwhelming majority of optimization algorithm, the precocity conditions where algorithm is stuck in local optimum are the most difficult part during searching procedure. In order to improve the ability of algorithm, it is very necessary to introduce the strategy of jumping out of local optimum. In NFu phase, Levy distribution strategy is introduced to help solution jump out of local optimum. As we known, compared with Gaussian walk, Levy flight whose random step follows a heavy-tailed probability distribution possesses a very important feature that in an uncertain environment individual can search more effectively the space [2]. Due to the advantage of Levy flight, large-scale search of Levy flight is employed to ionization stage and fusion stage in NFu phase. In (11) and (12) of ionization stage, if the item $\boldsymbol{X}_{r2,d}^{Fi} = \boldsymbol{X}_{i,d}^{Fi}$ is true, the Levy flight strategy is described as follows:

$$\boldsymbol{X}_{i,d}^{Ion} = \boldsymbol{X}_{i,d}^{Fi} + \left(\alpha \otimes \text{Levy}(\beta)\right)_d \cdot \left(\boldsymbol{X}_{i,d}^{Fi} - \boldsymbol{X}_{best,d}^{Fi}\right) \quad (18)$$

where α is a scale factor whose value is depended on the scales of problem solved, the $\otimes$ is the entry-wise multiplications, and $\text{Levy}(\beta)$ [3] is the Levy flight step size which is described as

$$\text{Levy}(\beta) = \frac{\mu}{|\upsilon|^{1/\beta}} \quad (19)$$

where μ and υ are respectively obtained from the normal distribution $N\left(0,\sigma_\mu^2\right)$ and $N\left(0,\sigma_\upsilon^2\right)$. σ_μ and σ_υ are designed by follows:

$$\sigma_\mu = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left[(1+\beta)/2\right]\beta 2^{(\beta-1)/2}}\right)^{1/\beta}, \quad \sigma_\upsilon = 1 \quad (20)$$

where Γ is a Gamma function. Meanwhile, $\alpha = 0.01$ and $\beta = 1.5$ are set in [2]. If $\boldsymbol{X}_{r2,d}^{Fi} \neq \boldsymbol{X}_{i,d}^{Fi}$, the Equation (11) and (12) is to be carried out. During the later period of iteration, due to search step is decreasing, it is possible for $\boldsymbol{X}_{i,d}^{Fi} = \boldsymbol{X}_{best,d}^{Fi}$. Therefore, another Levy flight is proposed as

$$\boldsymbol{X}_{i,d}^{Ion} = \boldsymbol{X}_{i,d}^{Fi} + \left(\alpha \otimes \text{Levy}(\beta)\right)_d \cdot \left(\boldsymbol{ub}_d - \boldsymbol{lb}_d\right) \quad (21)$$

For (13) of ionization stage, in order to avoid getting stuck in local optimum, if $\boldsymbol{X}_{worst,d}^{Fi} = \boldsymbol{X}_{best,d}^{Fi}$ is true, (21) is carried out; otherwise Eq. (13) is implemented.

On the other hand, in fusion stage of NFu phase, due to short search step of deuteric exploration, the case of $\boldsymbol{X}_{r1}^{Ion} = \boldsymbol{X}_{r2}^{Ion}$ is likely to appear where algorithm could be easy to be run into local optimum. In order to avoid the condition, a Levy flight strategy is proposed as

$$\boldsymbol{X}_i^{Fu} = \boldsymbol{X}_i^{Ion} + \alpha \otimes \text{Levy}(\beta) \otimes \left(\boldsymbol{X}_i^{Ion} - \boldsymbol{X}_{best}^{Ion}\right) \quad (22)$$

where the parameters of Levy flight are set to the same case as (18) and (21).

In the NRO algorithm, to avoid getting stuck in the local optima and enhance the rate of convergence, an optima storage mechanism during iterations is proposed. In the NFi phase, the fission product nucleus with the best fitness value in the current iteration must be stored as guidance information for the next phase. Meanwhile, in the NFu phase, the fusion nucleus with the best fitness value in the current iteration is stored as the global best solution and is output.

In addition, some individuals of a new population are obtained in the above search processes and may overflow the search boundary. Therefore, the individuals beyond the search boundary are recreated by using the boundary control strategy described as

$$\boldsymbol{X}_{i,d} = \boldsymbol{lb}_d + rand \cdot \left(\boldsymbol{ub}_d - \boldsymbol{lb}_d\right), \quad \boldsymbol{X}_{i,d} < \boldsymbol{lb}_d \; or \; \boldsymbol{X}_{i,d} > \boldsymbol{ub}_d \quad (23)$$

Based on a summary of the combination of the two main nuclear reaction phases, the pseudo code of the NRO algorithm is shown in Algorithm 1. The algorithm flowchart is presented in Fig. 7.

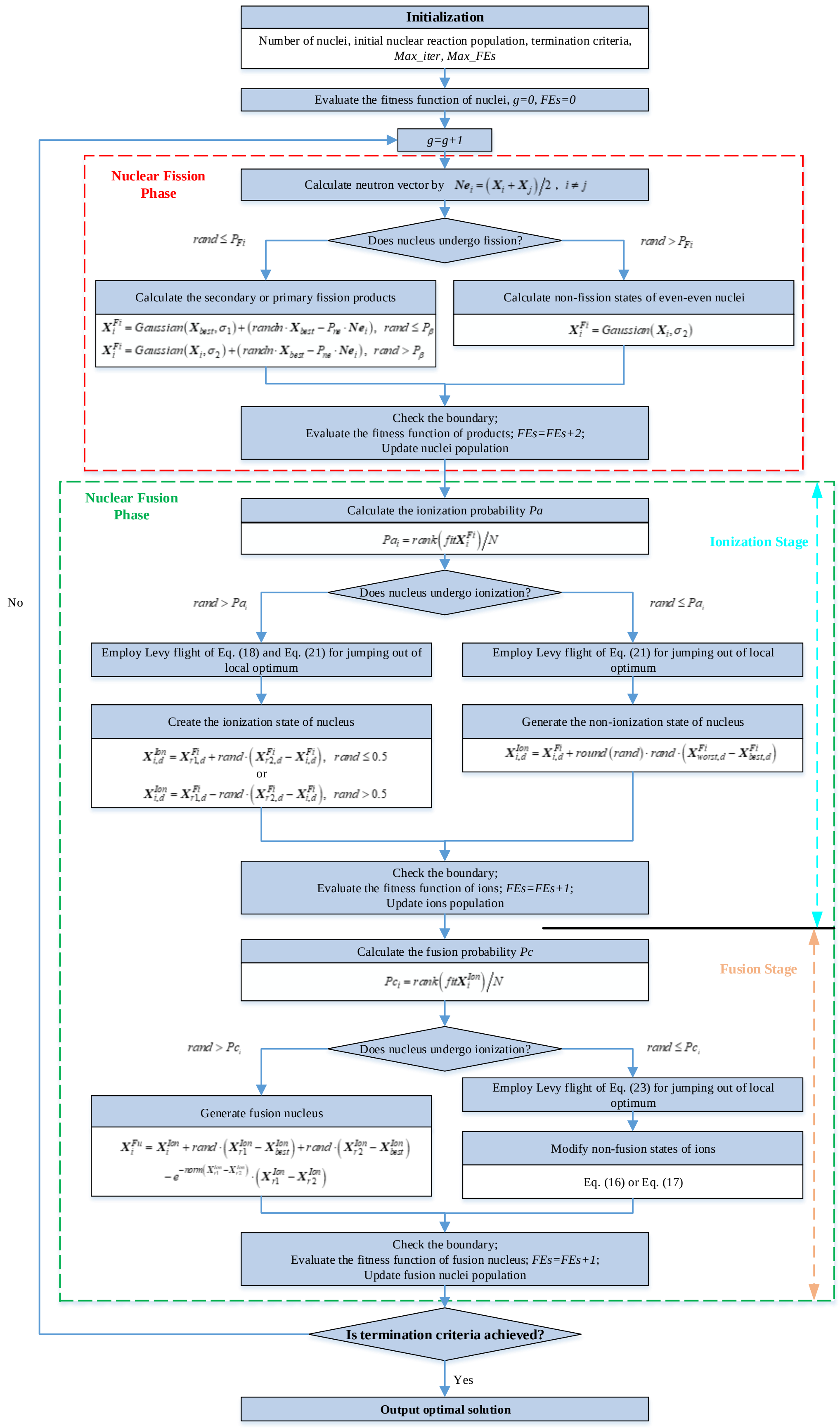

Initialization
Number of nuclei, initial nuclear reaction population, termination criteria, Max_iter, Max_FEs
Evaluate the fitness function of nuclei, g=0, FEs=0
g=g+1
Nuclear Fission Phase
Calculate neutron vector by $Ne_i = (X_i + X_j)/2$, $i \neq j$
Does nucleus undergo fission?
$rand \le P_{Fi}$
$rand > P_{Fi}$
Calculate the secondary or primary fission products
$X_i^{Fi} = Gaussian(X_{best}, \sigma_1) + (randn \cdot X_{best} - P_{ne} \cdot Ne_i)$, $rand \le P_\beta$
$X_i^{Fi} = Gaussian(X_i, \sigma_2) + (randn \cdot X_{best} - P_{ne} \cdot Ne_i)$, $rand > P_\beta$
Calculate non-fission states of even-even nuclei
$X_i^{Fi} = Gaussian(X_i, \sigma_2)$
Check the boundary; Evaluate the fitness function of products; FEs=FEs+2; Update nuclei population
Nuclear Fusion Phase
Calculate the ionization probability Pa
$Pa_i = rank(fitX_i^{Fi})/N$
Ionization Stage
Does nucleus undergo ionization?
$rand > Pa_i$
$rand \le Pa_i$
Employ Levy flight of Eq. (18) and Eq. (21) for jumping out of local optimum
Employ Levy flight of Eq. (21) for jumping out of local optimum
Create the ionization state of nucleus
$X_{i,d}^{Ion} = X_{r1,d}^{Fi} + rand \cdot (X_{r2,d}^{Fi} - X_{i,d}^{Fi})$, $rand \le 0.5$
or
$X_{i,d}^{Ion} = X_{r1,d}^{Fi} - rand \cdot (X_{r2,d}^{Fi} - X_{i,d}^{Fi})$, $rand > 0.5$
Generate the non-ionization state of nucleus
$X_{i,d}^{Ion} = X_{i,d}^{Fi} + round(rand) \cdot rand \cdot (X_{worst,d}^{Fi} - X_{best,d}^{Fi})$
Check the boundary; Evaluate the fitness function of ions; FEs=FEs+1; Update ions population
No
Calculate the fusion probability Pc
$Pc_i = rank(fitX_i^{Ion})/N$
Fusion Stage
Does nucleus undergo ionization?
$rand > Pc_i$
$rand \le Pc_i$
Generate fusion nucleus
$X_i^{Fu} = X_i^{Ion} + rand \cdot (X_{r1}^{Ion} - X_{best}^{Ion}) + rand \cdot (X_{r2}^{Ion} - X_{best}^{Ion}) - e^{-norm(X_{r1}^{Ion} - X_{r2}^{Ion})} \cdot (X_{r1}^{Ion} - X_{r2}^{Ion})$
Employ Levy flight of Eq. (23) for jumping out of local optimum
Modify non-fusion states of ions
Eq. (16) or Eq. (17)
Check the boundary; Evaluate the fitness function of fusion nucleus; FEs=FEs+1; Update fusion nuclei population
Is termination criteria achieved?
Yes
Output optimal solution

FIGURE 7. NRO algorithm flowchart.

Algorithm 1. Pseudo Code of NRO

```
1.  Settings: lb, ub, N, D, Max_iter and g=0;
2.  Initialize the nuclei population according to Eq. (1) and evaluate the fitness function;
3.  While (g<Max_iter) do
4.    g=g+1;
5.      // NFi phase
6.        for i=1 to N do
7.          Calculate neutron vector by Eq. (2);
8.          Update population of fission products according to Eq.(3), (6) or (9);
9.          Check the boundary and evaluate the fitness function for $X_i^{Fi}$;
10          Update $X_i^g$ with $X_i^{Fi}$;
11.       end for
12.     // NFi phase
13.       /* Ionization stage*/
14.         Calculate the Pa through Eq. (10);
15.         for i=1 to N do
16.           for d=1 to D do
17.             Execute the Levy flight strategies by Eq.(18), (21), (22) and the value of Pa;
18.             Update the states of ions according to Eq.(11), (12) or (13);
19.           end for
20.           Check the boundary and evaluate the fitness function for $X_i^{Ion}$;
21.           Update $X_i^{Fi}$ with $X_i^{Ion}$;
22.         end for
23.       /* Fusion Stage */
24.         Calculate the Pc through Eq. (14);
25.         for i=1 to N do
26.           Execute the Levy flight strategy by Eq.(22) and the value of Pc;
27.           Update the population of fusion products according to Eq.(15), (16) and (17);
28.           Check the boundary and evaluate the fitness function for $X_i^{Fu}$;
29.           Update $X_i^{Ion}$ with $X_i^{Fu}$;
30.         end for
31. end while
32. Output the best solution.
```

IV. EXPERIMENTAL STUDY

This section presents the experiments on unconstrained and constrained global optimization problems to integrally evaluate the performance of the proposed algorithm. For unconstrained problems, twenty-three standard benchmark test functions and CEC 2018 test suite are employed. For constrained problems, three commonly used engineering design optimization problems are utilized. All experiments are carried out on a Core(TM) i7-7700HQ 2.80 GHz with 8 GB RAM, and the simulations are performed in the MATLAB 2015a software.

4.1 EXPERIMENT Ⅰ: UNCONSTRAINED CLASSIAL BENCHMARKS

To prove the relative superiority of NRO in solving the unconstrained benchmark functions, 23 commonly used benchmarks referenced in literature [30] are provided in the experiment. Tables S-1, S-2 and S-3 (see supplementary material) describe the details of the various benchmarks. Typically, these benchmark functions can be divided into three categories: (1) unimodal functions presented in Table S-1, (2) multimodal high-dimensional functions described in Table S-2 and (3) multimodal low-dimensional functions presented in Table S-3.

In this section, the type, dimension, range and optimum of various benchmark functions are shown in Tables S-1, S-2 and S-3. The optimal locations are described as follows. For the high-dimensional benchmark functions f_{01}-f_{13}, the optimal locations are located in [0, 0, 0, …, 0] except for f_{05}, f_{08}, f_{12} and f_{13} whose optimal locations occur in [1, 1, 1, …, 1], [420.96, 420.96, …, 420.96], [-1, -1, -1, …, -1] and [1, 1, 1, …, 1], respectively. For the low-dimensional benchmark functions f_{14}-f_{23}, the optimal locations are detailedly described in Table S-4 (see supplementary material) [2]. As unimodal high-dimensional functions, f_{01}-f_{07} are designed to test the convergence rates instead of the final results of optimization. For high-dimensional benchmark functions f_{08}-f_{13}, due to the difficulties of jumping out of local optimum, the final results obtained in experiments are more important than the convergence rates. Meanwhile, final ten low-dimensional multimodal benchmark functions which include f_{14}-f_{23} with fix dimension and a few local minimums are designed to test the performance of solving the fixed and low dimensional optimization problems. The maximum number of function evaluations (MaxFEs) is set to 500 000 for f_{01}-f_{13} and 50 000 for f_{14}-f_{23}, respectively.

In order to prove the meliority of NRO, ten state-of-the-art algorithms proposed in the recent literature are used for comparison. In view of the diversity of compared algorithms and the inspirations of NRO from differential operators and physical phenomena, the ten state-of-the-art algorithms should include EAs, SI algorithms, human behavior-based algorithms and physical systems based algorithms. Therefore, ASO[59], GSA[46], LSA[51], MVO[49], BOA[85], CS[29], SSA[32], HS[86], BBO[87] and CMAES[88] are employed. ASO, GSA, LSA and MVO belong to physical systems based algorithms. BOA, CS and SSA are SI algorithms. HS is thought as a human behavior-based algorithm. BBO and CMAES are considered as evolutionary algorithms. Here, for a fair comparison, the population size of all compared algorithm is set to 100. Meanwhile, a uniformly distributed population generation method is adopted by all compared algorithms. The settings of parameters for all algorithms compared in this analysis are given as follows.

(1) ASO: $a = 50$ and $\beta = 0.2$ as in [59];

(2) GSA: G_0=100 and $a = 20$ as in [46];

(3) LSA: The channel time is set to 10 as in [51];

(4) MVO: $WEP = 0.2 + (1-0.2)\times(iter/iter_{\max})$ and $TDR = 1-(iter/iter_{\max})^{-6}$ as in [49];

(5) BOA: $c = 0.01$, $p = 0.8$ and $a = 0.1$ as in [85];

(6) CS: $\beta = 1.5$ and $p_a = 0.25$ as in [29];

(7) SSA: $c_1 = 2\times\exp\left(-\left(4\times iter/iter_{\max}\right)^2\right)$, $c_2 = rand$ and $c_3 = rand$ as in [32];

(8) HS: The harmony memory consideration rate is set to 0.9, and the pitch adjustment rate is set to 0.1 as in [86];

(9) BBO: The maximum immigration rate I=1 and the maximum emigration rate E=1 have been recommended in [87];

(10) CMAES: $\lambda = N/2$ and $\sigma = 0.3$ as in [88];

(11) NRO: $P_{Fi} = rand$, $P_{\beta} = rand$ and $freq = 0.05$; the settings of parameters for Levy distribution strategy are used as in [2].

For each benchmark function, the process of searching global minimum is performed independently 30 times by all compared algorithms which can use a different initial population at every time.

4.1.1 UNIMODAL FUNCTIONS

In order to show the performance of convergence of proposed NRO algorithm, f_{01}-f_{07} functions are experimented by the compared algorithms and NRO. In the experiment, the statistic results of 30 independent runs for unimodal functions including f_{01}-f_{07} are presented in Table S-5 (see supplementary material). Meanwhile, the ranks among all compared algorithms are visualized in Fig. 8. In order to describe the differences between various algorithms further, the convergence curves of searching and the graphical analysis results of the analysis of variance (ANOVA) tests are shown in Fig. S-1 and S-2 (see supplementary material). As can be seen in Table S-5, in addition to the mean solution obtained by 30 independent runs, the best solution and standard deviation are given. Meanwhile, according to the principle that we rank algorithms from the smallest mean solution to the highest mean solution, the rank of each function among all algorithms, average rank and overall rank are listed. When mean solutions are same among all algorithms, the standard deviation can be used for rank. If standard deviation is same, the best solution is employed for rank. Then, the average rank can be calculated by all ranks of seven functions and the overall rank is obtained by the average rank. In each unimodal function, the results of algorithm with best performance are shown in **boldface**. In Table S-5, NRO provides better results than compared algorithms except f_{05} and f_{07}. Meanwhile, for f_{06}, NRO finds global optimal solution which is 0. For the functions f_{05} and f_{07}, ranks of NRO are 2 and 8 in Fig. 8, respectively. The benchmark function f_{05} called as Rosenbrock is difficult to approach optimal solution. In this test, CMAES exhibits the best performance. For f_{07} which is a noisy quadratic function with a uniformly distributed random number in [0, 1], BBO has the best mean value. However, NRO can possess the best rank for the overall rank. For unimodal benchmark functions, convergence rate is more important than final optimal value. The convergence rate of NRO and compared algorithms is shown in Fig.S-1. For convergence curve, the fitness error value is employed for expressing the accuracy of solution obtained. As can been in these figures, NRO algorithm has faster convergence rate. The ANOVA tests presented in Fig.S-2 manifest that NRO can show its superiority in terms of solution quality and stability.

4.1.2 MULTIMODAL HIGH-DIMENSIONAL FUNCTIONS

Because multimodal functions f_{08}-f_{13} with high dimensions and local minima can test the algorithm's ability of jumping out of poor local optima and obtaining the near-global optimum. As the dimension of f_{08}-f_{13} increases, the number of local minima increases exponentially. As with experiments of unimodal functions, each benchmark function has been run independently 30 times. However, solution quality of multimodal high-dimensional functions is more important. In Table S-6 (see supplementary material), statistic results is summarized. The ranks for each function among different algorithms are given in Fig. 8. Meanwhile, the convergence curves of searching and the graphical analysis results of the ANOVA tests are shown in Fig.S-3 and S-4. According to statistic results and ranks where the best algorithm is **bold**, NRO algorithm outperforms its superiority in terms of quality. NRO can find global optimal solution in f_{08}, f_{09} and f_{11}. Meanwhile, in f_{10}, the best fitness value of 30 independent experiments can achieve global optimal solution. Although NRO and HS have same mean fitness value in f_{08}, NRO shows outperformance in standard deviation. In f_{11}, rank of NRO which is first is same as CMAES. On the whole, according to average rank, NRO can obtain the best rank for overall rank. In Fig.S-3 (see supplementary material), it is observed that the solution quality is best amongst all compared algorithms. As can be seen in Fig.S-4 (see supplementary material), NRO represents the best solution quality and stability. In general, the exploitation performance of NRO is strongly competitive with compared algorithms.

4.1.3 MULTIMODAL LOW-DIMENSIONAL FUNCTIONS

In contrast to the multimodal high-dimensional functions, the last ten benchmark functions including f_{14}-f_{23} with only a few local minima work with fixed and low dimensionality. In this condition, it is difficult to judge the performance of all compared algorithms. For f_{14}-f_{23}, due to their low dimensionalities and a few local minima, it is simpler to search the global optimal solution than f_{08}-f_{13}. The optimization results obtained by above 30 independent experiments for multimodal low-dimensional functions are listed in Table S-7 (see supplementary material). The ranks counted from statistic results of f_{14}-f_{23} are presented in Fig.8. Meanwhile, the convergence curves of searching and the graphical analysis results of the ANOVA tests are shown in Fig.S-5 and S-6 (see supplementary material). As can be found in Table S-7, NRO outperforms its superiority and ranks the first for each multimodal low-dimensional function. Particularly, the solution quality which includes mean solution and standard deviation ranks the first in f_{14}, f_{15}, f_{16}, f_{18}, f_{19}, f_{21}, f_{22} and f_{23}. Meanwhile, in f_{17}, NRO has the first rank that ASO, GSA and LSA have also. In f_{20}, ASO and GSA exert better results and rank the first as with NRO. In detail, although NRO and a few compared algorithms can approach same mean fitness values in f_{14}-f_{20}, f_{22} and f_{23}, there are differences in term of standard deviation. In general, according to the average rank, the overall rank of NRO is the first. In other words, NRO has outstanding exploration ability in f_{14}-f_{23}. ASO and GSA rank second and third, respectively, and BOA exerts the worst performance. Fig.S-5 shows the fitness error value convergence curves of NRO and compared algorithms for multimodal low-dimensional functions. In f_{14}, f_{15}, f_{18}, f_{20} and f_{21}, mean fitness error values of NRO can converge faster than other algorithms. Although the convergence rates of BBO in f_{16}, f_{17} and f_{19} and CMAES in f_{22} and f_{23} outperform respectively, NRO can reach same mean fitness error values as BBO and CMAES. Fig.S-6 describes the ANOVA tests of all algorithms for multimodal low-dimensional benchmark functions. In Fig.S-6, NRO presents better solution quality, which has better mean solution and smaller standard deviation. According to above analysis, the exploitation performance of NRO is strongly competitive with compared algorithms in term of multimodal low-dimensional functions.

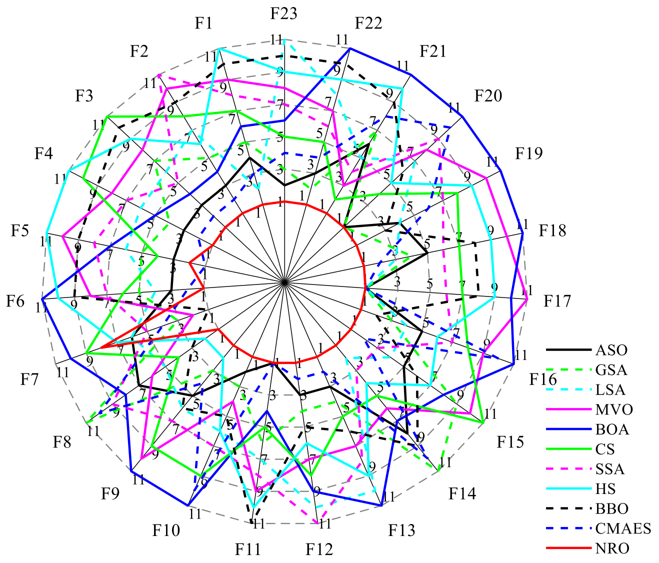


FIGURE 8. Spider charts for ranks among all compared algorithms.

4.1.3 COMPARISON OF CONVERGENCE SPEED AND SR

In order to further explore the convergence performance of different algorithms, we select the number of function evaluations (FEs) required by the algorithm to reach the predefined accuracy level and the success rate (SR) as evaluation indexes [89]. Due to the difference of MaxFEs between high-dimensional functions set and low-dimensional functions set, the results of mean FEs and SR under predefined accuracy level (Threshold) are divided into two tables. The results of mean FEs and SR are summarized in Tables S-8 and S-9 (see supplementary material). Due to the difference of functions, an appropriate threshold value of each test function is selected. For high-dimensional functions with minima of fitness error $f\left(\boldsymbol{x}\right)-f\left(\boldsymbol{x}^*\right)$ at zero, where $f\left(\boldsymbol{x}\right)$ is the best solution fitness attained in experiments and $f\left(\boldsymbol{x}^*\right)$ is the global optimum, the threshold is set to 1.00e-5 [90], except for f_{07} and f_{08}. The thresholds of f_{07} and f_{08} are set to 1.00e-2 and 1.00e-3, respectively. For low-dimensional functions, the threshold is set to 1.00e-5 [90], except for f_{17} and f_{19}, whose thresholds are 1.00e-3 and 1.00e-4, respectively. The stopping criterion is satisfied when the best fitness value reaches to the predefined threshold value or the number of FEs reaches to MaxFEs. Each test function is tested by each compared algorithm 30 times.

Tables S-8 and S-9 summarize the results of the mean number of FEs and SR of each benchmark functions for eleven compared algorithms, where "NA" means that the best fitness value of corresponding algorithm can't reach to the predefined threshold while meeting the MaxFEs. The best FEs and SR among all compared algorithms are presented in **boldface**. As you can see in Table S-8, for high-dimensional functions, BBO algorithm shows a faster convergence speed on f_{07}, and CMAES has a faster convergence speed on f_{04}, f_{06}, f_{12} and f_{13}. Meanwhile, ASO hardly converges to the threshold in f_{05}, f_{08} and f_{09}, GSA can converge to the threshold except for f_{02}, f_{05}, f_{08} and f_{09}, LSA can't reach to the predefined level in f_{03}, f_{05}, f_{08} and f_{09}, MOV converges to the threshold in f_{07}, BOA can converge to the threshold in f_{01}, f_{02}, f_{03}, f_{04} and f_{11}, CS can reach to the predefined level at 100% SR in f_{01}, f_{06} and f_{13}, SSA can converge to the threshold at 100% SR in f_{01}, f_{03} and f_{06} and HS can reach to the threshold at 96.67% SR in f_{02}. Although NRO converges to the predefined level at 43.33% SR in f_{07}, it can reach to the threshold at 100% SR in other functions. NRO can possess the fastest convergence in f_{01}-f_{03}, f_{05} and f_{08}-f_{11}. As seen in total average FEs, NRO costs the fewest FEs to converge to the threshold. The acceleration rates between NRO and other compared algorithms are 3.66, 2.41, 1.61, 4.44, 2.02, 4.68, 4.72, 3.06, 2.81 and 1.21, respectively. In sum, NRO can successfully converge to the predefined level for all functions at highest SR. As seen in Table S-9, for low-dimensional functions, ASO hardly converges to the threshold value in f_{15}, GSA can't reach to the predefined level in f_{14} and f_{15}, LSA can converge to the threshold value at 100% SR in f_{14}-f_{19}, MVO can't reach to the threshold in f_{21}, BOA can merely converge to the predefined level at 43.33% and 3.33% SR in f_{14} and f_{17} respectively, CS can't get to the threshold value in f_{15}, f_{20}-f_{21} and f_{23}, SSA can better converge to the threshold at 100% SR in f_{14} and f_{16}-f_{19}, HS and BBO can't get to the threshold in f_{15}, and CMAES converges to the predefined level at 100% SR in f_{16}-f_{19} and f_{22}-f_{23}. Additionally, BBO shows a faster convergence performance in low-dimensional functions except for f_{15}. Although NRO has merely faster convergence in f_{15}, it obtains better FEs at 100% SR. In total, the mean number of FEs of NRO gets the second rank, which follows BBO algorithm. The acceleration rates between NRO and other compared algorithms are 3.59, 2.33, 1.25, 4.76, 2.78, 2.40, 3.90, 1.29, 0.33 and 2.24, respectively. NRO can successfully converge to the predefined level for all low-dimensional functions at highest SR.

4.1.4 STATISTICAL HYPOTHESIS TEST

For a further detailed comparison, we employ the Wilcoxon signed ranks test to indicate the superiority of NRO over the other ten algorithms compared. Table S-10 (see supplementary material) presents s summary of the statistical results obtained from the Wilcoxon signed rank test [2] at the 95% significance level for twenty-three benchmark functions. As a nonparametric test, the Wilcoxon signed ranks test can examine whether there is a significant difference between two compared algorithms. In Table S-10, NRO is employed to compare with other ten algorithms. In addition, "R+" represents the sum of ranks in the case that NRO outperforms better efficacy over the corresponding competitor. "R-" represents the sum of ranks in the case that NRO exerts the weakness. Meanwhile, "+", "-" and "≈" mean that NRO is significantly better than the competitor, the competitor is significantly better than NRO and the performance of NRO is statistically similar to that of compared algorithm, respectively. As can be observed in Table S-10, although NRO is not significantly better than the compared algorithm in some benchmark functions, NRO obtains more "+" than the competitor. In general, NRO exerts significant outperformance compared to other algorithms statistically in Wilcoxon signed ranks test with the significance level α=0.05.

To further display the difference between NRO and other ten competitors, a non-parametric multiple comparison approach, the Friedman test, is carried out according to the mean values and standard deviations [91]. In Friedman test, Iman-Davenport's procedure is employed as a post hoc procedure. The average ranks in terms of Mean and SD values among all compared algorithms obtained by Friedman test are given in Table 1. According to the Friedman test, the chi-square with ten degrees of freedom are 96.19 and 77.27 in terms of Mean and SD with significance level α=0.05 respectively. The p-values are 3.13891e-16 for Mean and 1.71786e-12 for SD. According to hypothesis of Friedman test, the term p-values<α is true, so the fact that all compared algorithms are different in terms of mean and SD comes into being. Meanwhile, the lower rank indicates that the algorithm is more successful. As you can see in Table 1, NRO ranks first in terms of Mean and SD.

To further determine where the differences actually occur, Iman-Davenport's test is executed [91]. Iman-Davenport's test is a less conservative alternative with statistics distributed according to F-distribution with $(k-1)$ and $(k-1)(N-1)$ degrees of freedom. The F-distribution is given as follows:

$$F_F^2 = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{24}$$

where k is the number of all compared algorithm, and N is the number of test functions. In this paper, k and N are 11 and 23, respectively. The chi-square in (24) is calculated by the previous test. The p-values computed by the Iman-Davenport's test at significance level α=0.05 are 7.64880e-08 and 1.75473e-06 for Mean and SD, respectively. The Nemenyi test, the most commonly used post hoc test for the Friedman statistic, is employed to find the differences among all ranked algorithms by using a critical difference (CD) [41], which can be calculated by following formula:

$$CD = q_\alpha \sqrt{k(k+1)/6N} \tag{25}$$

where q_α is the critical values which can be found in the statistical tables according to F-distribution with 10 and 220 degrees of freedom. In this paper, the critical value is 2.2967, so the CD is 2.2462. The comparisons of all compared algorithms using the CD based on Friedman and Iman-Davenport tests for Mean and SD are shown in Fig. 9. Fig. 9 shows that these compared algorithms with less difference are connected by using CD. From Fig. 9, it can be seen that NRO ranks first in terms of Mean and SD. Meanwhile, from Fig. 9(a), NRO has a similar performance as ASO and CMAES in term of Mean. However, there is significantly difference NRO and the group of the algorithms including GSA, LSA, CS, SSA, BBO, MVO, HS and BOA. From Fig. 9(b), it can be found that there is no significant difference among NRO, ASO and CMAES in term of SD value. However, NRO dominates the other compared algorithms except for ASO and CMAES. On the whole, the proposed NRO exhibits superiority compared to other ten state-of-the-art algorithms for 23 benchmark functions.

4.1.5 SEARCH BEHAVIOR ANAYSIS OF NRO

This subsection describes an experimental analysis of the effectiveness of three search behaviors of NRO, which include fission behavior in NFi phase and ionization and fusion behaviors in NFu phase. Therefore, the proposed algorithm is compared with three different algorithms: (1) NRO without the NFi phase (denoted NRO-Fi), (2) NRO without the Ionization stage of NFu phase (denoted NRO-Ion), (3) NRO without the Fusion stage of NFu phase (denoted NRO-Fu), which have been summarized in Table 2. Meanwhile, because there is only one phase for each algorithm, the population of the three compared algorithms is set to 100. In order to test the ability of convergence rate and accuracy for solution, two unimodal benchmark functions (including f_{01} and f_{05}), two high-dimensional multimodal benchmark functions (including f_{08} and f_{13}) and two low-dimensional benchmark functions (consisting of f_{18} and f_{22}) are selected for comparing NRO with other three algorithms. As similar to preceding experiment settings, each function is tested independently 30 times by all compared algorithms. The statistical results obtained from experiments are presented in Table 3. Meanwhile, Fig.S-7 (see supplementary material) depicts the convergence rates for six functions.

As can be seen in Table 3 and Fig.S-7 for unimodal benchmark functions f_{01} and f_{05}, NRO outperforms other three algorithms in terms of convergence rate and accuracy and the performance of NRO-Fi is worst. For multimodal benchmark functions including f_{08}, f_{13}, f_{18} and f_{22}, NRO shows the best performance, but NRO-Ion performs the worst rank in terms of convergence rate and accuracy. In general, several significant conclusions are tersely drawn as follows: First, NRO outperforms NRO-Fi, NRO-Ion and NRO-Fu in terms of convergence rate and accuracy, which illustrates that combination of fission phase, ionization and fusion stages are very vital. Second, the performance of NRO-Fi is worst in unimodal benchmark functions, which means that the NFi phase has the greatest impact in ability of convergence rate and accuracy of NRO. Third, NRO-Ion performs the worst rank in multimodal benchmark functions, which indicates that the ionization stage has significant impact on performance of NRO. Finally, although the impact of fusion stage in NFu phase is far less than role of other two behaviors, the fusion also owns fairly effects. Because the coexistence of three behaviors impacts the performance of NRO, the algorithm can solve better the above optimization problems. In brief, three behaviors are very important for NRO.

4.1.6 COMPUTATIONAL COMPLEXITY OF NRO

The computational complexity of NRO depends on the nuclei size N, dimension of the problem D, maximum number of iterations T, and sorting behavior of the NFu phase in each run. During each iteration, the time complexity of all nuclei fission and border control strategy in NFi phase is $O(2ND)$. Meanwhile, in the ionization stage of NFu phase, the time complexity of the ionization probability Pa is $O\left((N-1)^2\right)$; the time complexity of the ionization and border control strategy is $O(2ND)$. In addition, in the fusion stage of NFi phase, the time complexity of the fusion probability Pc is $O\left((N-1)^2\right)$; the time complexity of all ions fusion and border control strategy in fusion stage is $O(2ND)$. According to above analysis, the overall computational complexity of NRO is $O\left(2TN^2+5TND\right)$.

In order to display the time complexity of NRO and other algorithms, the time consuming and time ranks for 23 benchmark functions are given in Table S-11 (see supplementary material) and Fig. 10. Table S-11 gives the mean time required in seconds by NRO and compared algorithms for 23 benchmark functions tested 30 independent runs. As you can see in Table S-11 and Fig.10, CMAES requires the least time in the mass. CMAES has the best rank in average time cost and NRO has the forth rank. Although the NRO does not rank the best, it is better than other algorithms except for CS, SSA and CMAES. However, based on the solution quality and time consuming for 23 benchmark functions, the computational complexity of NRO is acceptable.

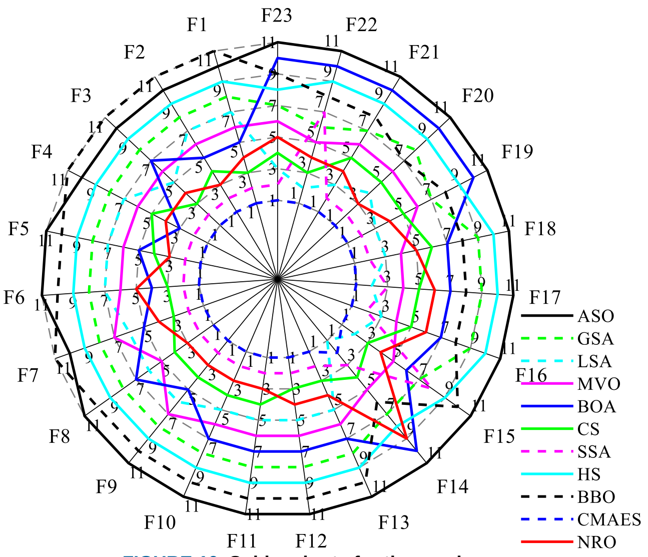


FIGURE 10. Spider charts for time ranks.

4.1.7 PARAMETER ANALYSIS OF NRO

For a novel algorithm like NRO, parameter tuning is one of the most challenging tasks of testing a new algorithm and applying this algorithm for a variety of optimization problems. Setting algorithm parameters static leads to the lack of flexibility in the process of optimization. Due to the disadvantage of setting the parameters static, it is very necessary of analyzing the parameters of NRO algorithm. This section investigates thoroughly the effect of various settings of NRO parameters. NRO parameters consist of nuclear fission probability P_{Fi}, β decay probability P_{β}, frequency of the sinusoidal function *freq*, and nuclei population size N. In preceding experiments for performance of NRO, nuclear fission probability and β decay probability are defined as a random number between 0 and 1. The frequency of the sinusoidal function is 0.05. For population size, N=100 is simply explored. The population size is an important static parameter, which is a common parameter in various algorithms. P_{Fi}, P_{β} and *freq* will be investigated mainly. Meanwhile, the effect of N whose parameter analysis embodies the performance of robustness for an algorithm. In order to evaluate the performance of NRO extensively, an experimental study based on f_{01}, f_{03}, f_{07}, f_{09}, f_{10}, f_{12}, f_{14} and f_{18} is given. These functions are selected from different types according to empirical suggestion and study of previous literatures [92].

The first parameter is nuclear fission probability P_{Fi}, which is analyzed in this section. The nuclear fission probability determines whether the nucleus can produce fission. If $P_{Fi} \geq rand$, the phase can create the fission; otherwise, the even-even nucleus can't be motivated. To understand how various settings for P_{Fi} effect the performance of NRO, different levels of P_{Fi} between 0.1 and 1 are employed. The statistic results are shown in Table S-12 (see supplementary material). Meanwhile, the parallel coordination plots of the corresponding ranks according to mean values and standard deviations obtained from 30 experimental runs by NRO on each selected function are provided in Fig. 11. As you can see in Table S-12, different levels of P_{Fi} are ranked in mean values and standard deviation. In each function, the level parameter with the better statistic values gets higher rank. The obtained ranks of various levels of P_{Fi} in selected functions are stated in Fig.11. At first, P_{Fi} are chosen between 0.1 and 1 except for 0.75. After testing different levels, it can be found that there is a turning point for statistic values of f_{01} and f_{03}. In order to explore further, $P_{Fi}=0.75$ is used. It seems that $P_{Fi}=0.75$ can gain the best rank for f_{01}, f_{07}, f_{09}, f_{10}, f_{12}, f_{14} and f_{18}. Through calculating the average ranks for all levels, the overall ranks demonstrate that $P_{Fi}=0.75$ outperforms in most levels. Hence, $P_{Fi}=0.75$ is the proper setting, compared with other levels of P_{Fi}.

The second parameter is β decay probability P_{β}, which determines whether the nucleus conducts β decay after absorbing the neutron. If $P_{\beta} \geq rand$, the secondary product is created by β decay; otherwise, the phase will create the primary product. In order to analyze whether the β decay probability has an important effect on the performance of NRO, different levels of P_{β} between 0 and 1 are considered. The statistic results are summarized in Table S-13 (see supplementary material). As it is shown in Table S-13, $P_{\beta}=0.0$, $P_{\beta}=0.1$ and $P_{\beta}=0.2$ have efficient results in most conditions. Based on Table S-13, Fig. 12 illustrates the ranking of different P_{β} in selected test functions. As illustrated in Fig. 12, $P_{\beta}=0.1$ obtains the first rank six times. Combined with overall rank of Table S-13, $P_{\beta}=0.1$ has the best rank. Hence, $P_{\beta}=0.1$ is selected as the best level of β decay probability.

The third parameter is the frequency of the sinusoidal function $freq$, which determines the adjustment rate of non-fusion. In order to investigate whether $freq$ has a significant impact on NRO, different levels of the parameter between 0.01 and 1 are employed. These levels include 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1. Statistic results in various values of $freq$ are summed in Table S-14 (see supplementary material). It can be seen that $freq=0.05$ performs better than other levels in f_{01}, f_{03} and f_{07}. The rankings that are calculated by different levels of $freq$ are summarized in Fig. 13. According to Fig. 13, the overall rank indicates that $freq=0.05$ has a significant superiority and ranks the first. Hence, the value of $freq$ is set to 0.05, which has the best performance, compared with other levels of $freq$.

At the end of this section, the number of nuclear reaction population is analyzed. As a common core parameter, N has a sensitive impact on performance of NRO. In order to investigate the effect of N, NRO is tested by above selected functions. For other parameter settings, P_{Fi}, P_{β} and $freq$ will equal to 0.75, 0.1 and 0.05, respectively. Additionally, the MaxFEs of each function are same as previous cases and different levels of N have same MaxFEs. Table S-15 (see supplementary material) lists the statistic results of 30 independent runs on different levels of N. From Table S-15, it can be seen that the performance of NRO algorithm is weakened gradually as N is increasing under the same MaxFEs in most case. With N increasing, the average time cost is decreased roughly. Fig.S-8 (see supplementary material) describes the effect of N on convergence rate for test functions. As shown in Fig. S-8, NRO with $N=30$ has faster convergence rate than other levels. The ranking that are calculated by different levels of N in terms of fitness value and average time cost are summarized in Fig. 14 and 15 respectively. From Fig. 14, it can be found that $N=100$ can gain the best overall rank in term of fitness value. Additionally, $N=50$, $N=30$ and $N=80$ rank in order at the first heels. Fig. 15 shows that $N=30$ ranks the first in term of average time cost. $N=50$, $N=100$ and $N=80$ have the second, third and forth ranks. According to analysis of above results, it is proper to set N as 30-100 due to a trade-off between time cost and convergence rate.

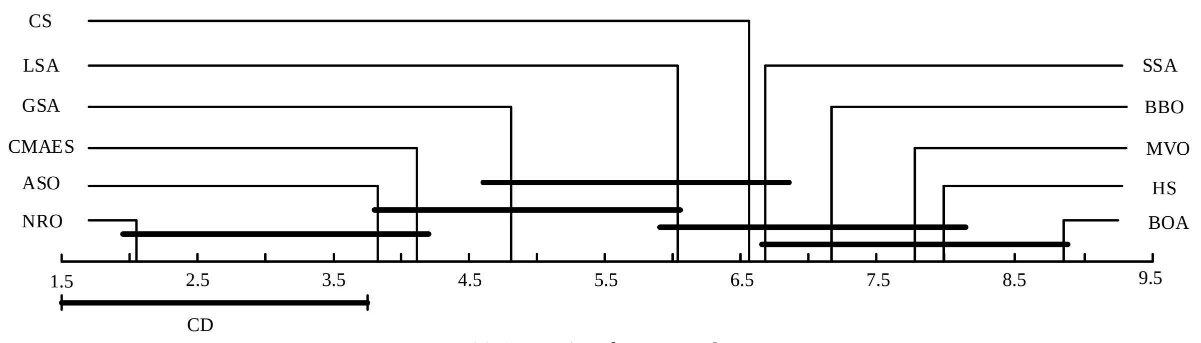
Based on the above analysis, it can be concluded that $P_{Fi}=0.75$, $P_{\beta}=0.1$ and $freq=0.05$ are fairly proper parameter settings. As a common parameter, N value is different for different optimization problems. In general, $N \in [30,100]$ is considered. This section employs a simple method to adjust the parameter. Hence, a comprehensive study on parameter settings of NRO algorithm could be discussed in the future.

TABLE 1.
AVERAGE RANKS FOR NRO AND TEN ALGORITHMS ACCORDING TO THE FRIEDMAN TEST (Α=0.05)

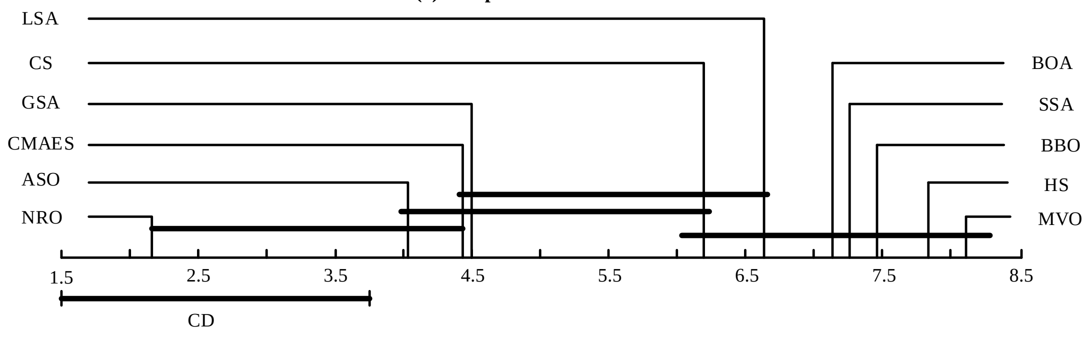
Algorithms	ASO	GSA	LSA	MVO	BOA	CS	SSA	HS	BBO	CMAES	NRO
Mean ranks[a]	3.8261	4.7826	6.0435	7.8261	8.8261	6.5652	6.7826	8	7.1957	4.1087	**2.0435**
SD ranks[b]	4.0435	4.5000	6.6739	8.1304	7.1739	6.2174	7.2609	7.8696	7.4783	4.4783	**2.1739**

a. 1) p-value computed by the Friedman test: 3.13891e-16. Chi-square is 96.19.
2) F-distribution with 10 and 220 degrees of freedom is 15.82.
3) p-value computed by the Iman-Davenport test: 7.64880e-08.

b. 1) p-value computed by the Friedman test: 1.71786e-12. Chi-square is 77.27.
2) F-distribution with 10 and 220 degrees of freedom is 11.13
3) p-value computed by the Iman-Davenport Test: 1.75473e-06.



(a) Comparison for Mean value



(b) Comparison for SD value

FIGURE 9. Algorithm comparison using a post-hoc test; the algorithms are connected using the CD with less difference.

TABLE 2.
DIFFERENCE IN THE BEHAVIOR OF THE FOUR TYPES OF NRO ON ANALYSIS.

Algorithms	Nuclear-fission	Ionization	Fusion
NRO	Yes	Yes	Yes
NRO-Fi	No	Yes	Yes
NRO-Ion	Yes	No	Yes
NRO-Fu	Yes	Yes	No

TABLE 3.
MEANS OBTAINED BY NROS WITH DIFFERENT OPTIMIZATION BEHAVIORS FOR 6 TYPICAL FUNCTIONS.

Function	Algorithms	NRO-Fi	NRO-Ion	NRO-Fu	NRO
f_{01}	Mean	6.0024e-46	5.6682e-172	6.7079e-160	**1.7991e-177**
	SD	6.5829e-46	0	3.3833e-159	**0**
f_{05}	Mean	4.6197e+00	9.3673e+00	2.0837e+00	**2.7068e-23**
	SD	6.1732e-01	6.5224e-01	3.2064e-01	**3.0544e-23**
f_{08}	Mean	-7.9102e+03	-5.7979e+03	-8.0592e+03	**-1.2569e+04**
	SD	2.4842e+02	4.1239e+02	2.9753e+02	**1.8501e-12**
f_{13}	Mean	**1.3498e-32**	9.7977e-07	1.8136e-27	**1.3498e-32**
	SD	**5.5674e-48**	5.2391e-06	1.2152e-27	**5.5674e-48**
f_{18}	Mean	3.0000e+00	3.0000e+00	3.0000e+00	**3.0000e+00**
	SD	9.8269e-16	2.0014e-15	1.7764e-15	**1.2588e-15**
f_{22}	Mean	-1.0403e+01	-1.0403e+01	-1.0403e+01	**-1.0403e+01**
	SD	1.2342e-15	4.0992e-06	3.2986e-16	**1.4378e-16**

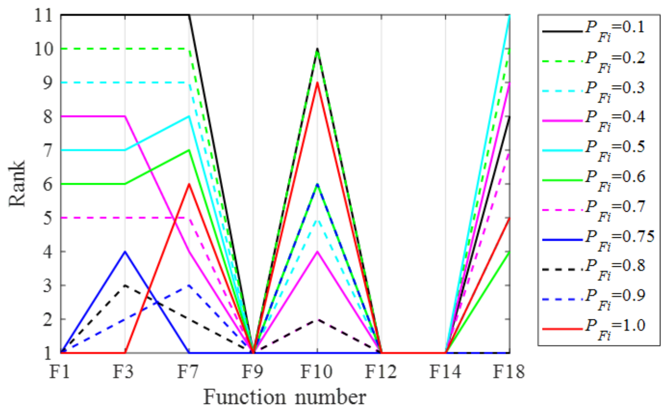


FIGURE 11. Ranking of different levels of P_{Fi} for test functions.

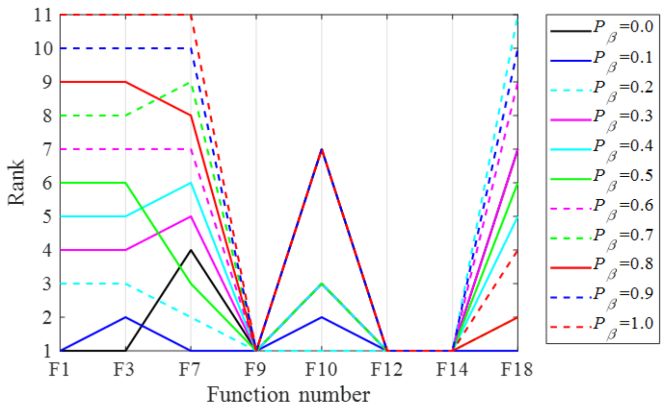


FIGURE 12. Ranking of different levels of P_{β} for test functions.

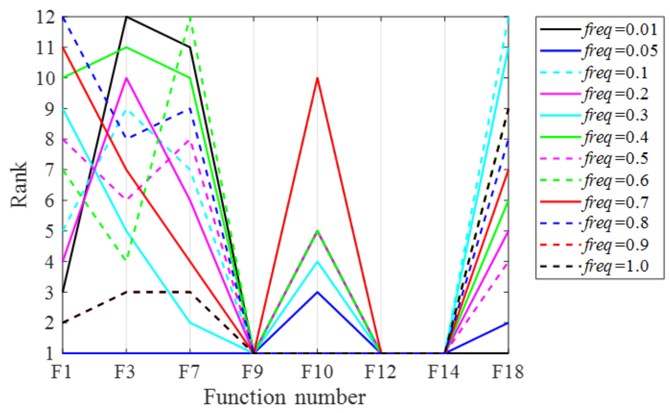


FIGURE 13. Ranking of different levels of *freq* for test functions.

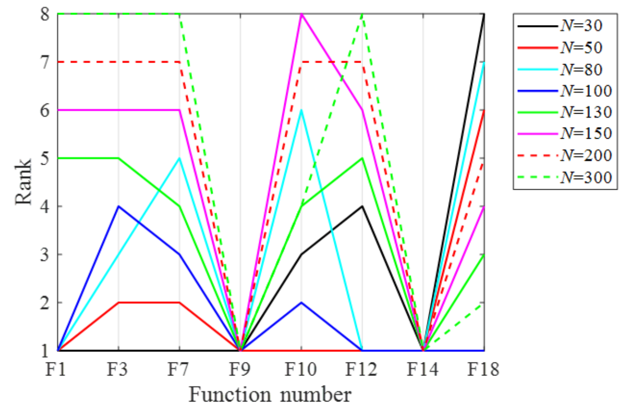


FIGURE 14. Ranking of different levels of *N* for test functions in term of fitness value.

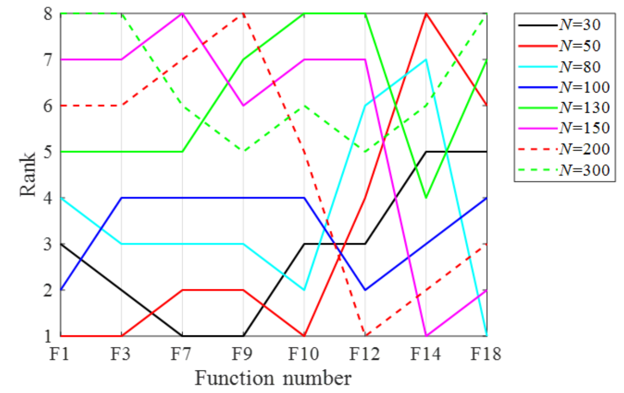


FIGURE 15. Ranking of different levels of *N* for test functions in term of average time cost.

4.2 EXPERIMENT Ⅱ: CEC 2018 TEST SUITE

In the recent years, various kinds of novel optimization algorithms have been proposed to solve real-parameter optimization problems. To evaluate the performance of NRO for solving single-objective real-parameter numerical optimization, the CEC 2018 test suite, including 29 benchmark functions as minimization problem, is employed to carry out the experiment in this section. CEC 2018 test suite which is developed by CEC 2017 test suite consists of 2 unimodal functions, 7 multimodal functions, 10 hybrid functions and 10 composition functions. Unimodal functions include f_{01CEC} and f_{03CEC}, whose optima are 100 and 300, respectively. The optima of f_{04CEC}-f_{10CEC} that belong to multimodal functions equal to 400-1000 respectively. Meanwhile, functions f_{11CEC}-f_{20CEC} are hybrid functions, whose optima are 1100-2000 respectively. For composition functions f_{21CEC}-f_{30CEC}, their optima are 2100-3000 respectively. More details can be found in [93]. For convenience, the same search range is defined as $[-100,100]^D$, where D is the dimensionality of the problem and is set to 10, 30, 50 and 100.

As required by the CEC 2018 organizer, all benchmarks are evaluated with MaxFEs, which is set as $D\times10000$. In this section, the experiments are executed with 50D. For each function, all compared algorithms are performed 51 independent times. As same as classical functions experiment, the solution results of algorithm are recorded by using fitness error, which is defined as $f(\boldsymbol{x})-f(\boldsymbol{x}^*)$, where $f(\boldsymbol{x})$ is the best solution fitness obtained in experiments and $f(\boldsymbol{x}^*)$ is the global optimum of the test function. In CEC 2018 test suite, LSHADE [69], WDE [94], DE [12], LSA [51], CSA [96] and YYPO [95] are employed to compare with NRO. The population size of all compared algorithms is set to $D\times10$, and MaxFEs are set to $D\times10000$. The results consist of average time cost and statistical results that include mean value and standard deviation. At first, a uniformly distributed population generation method is adopted by compared algorithms. The settings of parameters for all algorithms compared in this section are given as follows.

(1) LSHADE: Memory size $H=5$ and archive rate $H=1.4$ as in [69];

(2) WDE: no-parameter algorithm;

(3) DE: $F=0.5$, $CR=0.9$ as in [12] and the DE/rand/2/exp strategy is used;

(4) LSA: The channel time is set to 10 as in [51];

(5) CSA: Awareness probability $AP=0.1$ and flight length $fl=2$ as in [96];

(6) YYPO: $I_{\min}=0.5$, $I_{\max}=0.5$ and $\alpha=0.5$ as in [95];

(7) NRO: $P_{Fi}=0.75$, $P_{\beta}=0.1$ and $freq=0.05$; the settings of parameters for Levy distribution strategy are used as in [2].

The statistical results and computation cost obtained by seven algorithms are listed in Table S-16 (see supplementary material). The **bold data** are the best solution. Meanwhile, based on mean values and standard deviations, we rank algorithms from the best to the worst. At the table, the average ranks are calculated by all ranks of CEC 2018 functions and listed in Table S-16. According to average ranks, the overall ranks are concluded at last row of Table S-16. The ranks of 29 CEC 2018 benchmark functions are described in Fig. 16.

As you can see in Table S-16, NRO has obvious advantages in term of mean values and deviation standards of 16 functions. However, NRO has no significant advantages in term of time cost. Relatively, YYPO algorithm performs well in term of time cost. For unimodal functions f_{01CEC} and

f_{03CEC}, the results of Table S-16 show that LSHADE reaches the best global optimum. Although NRO performs worse than LSHADE in f_{01CEC} and f_{03CEC}, it can have better performance than other algorithms. For multimodal functions, NRO exhibits the best performance in f_{06CEC}, f_{09CEC} and f_{10CEC}, and LSHADE outperforms than the compared algorithms in f_{04CEC}, f_{05CEC}, f_{07CEC} and f_{08CEC}. Additionally, NRO reaches the global optimum of f_{09CEC}. However, NRO ranks the second in other multimodal functions. For hybrid functions, it can be found that the performance of NRO is better than the compared algorithms in f_{11CEC}, f_{12CEC} and f_{15CEC}-f_{19CEC}, and LSHADE performs better than other algorithms in f_{13CEC}, f_{14CEC} and f_{20CEC}. As can be seen from Table S-16, NRO ranks toward the second for f_{13CEC}, f_{14CEC} and f_{20CEC}. For composition functions, as seen from Table S-16, although these results obtained are far away from the global optima, NRO performs better than other compared algorithms in f_{23CEC}-f_{25CEC}, f_{27CEC}-f_{28CEC} and f_{30CEC}. As you can see in Fig. 16, LSHADE ranks toward the top in f_{21CEC} and f_{29CEC}, and CSA has the first ranks in f_{22CEC} and f_{26CEC}. However, NRO has the second, the second, the third and the second ranks in f_{21CEC}, f_{22CEC}, f_{26CEC} and f_{29CEC} respectively. Based on Table S-16 and Fig. 16, NRO ranks the first for overall rankings of fitness values. Meanwhile, NRO has the third rank for overall rankings of average time. Although NRO has no significant advantages in term of time cost, NRO performs the best fitness rank.

For a further detailed comparison, the Wilcoxon signed ranks (α=0.05) is employed to illustrate the efficacy of NRO over the other compared algorithms [97][98]. Table S-17 (see supplementary material) summarizes the results for 51 independent runs to find the best algorithm amongst the all compared algorithms. The **bold data** are considered the best solution according to the Wilcoxon test. When examined the last line of the Table S-17, NRO shows the results with the highest '+' counts amongst all compared algorithms. It means that NRO exhibits the best performance of compared algorithms in this section.

To determine the difference between the seven algorithms in a statistical manner [97], the Friedman test is carried out according to mean values, standard deviations and average time costs. The chi-square with six degrees of freedom is 135.87 for Mean, 86.76 for SD and 163.36 for Time with significance level α=0.05, respectively. The p-values are 7.44287e-27, 1.42277e-16 and 1.15012e-32 for Mean, SD and Time, respectively. The hypothesis is rejected (p-value<α), so the performances of the seven algorithms are different in the related group. The average rankings in terms of Mean, SD and Time values among all compared algorithms obtained by Friedman test are given in Table 4. According to the rule of the Friedman test, NRO ranks the first, the second and the third in terms of Mean, SD and Time, respectively. To further determine the significant performance obtained from NRO, a post hoc Iman-Davenport's test is performed. As can see in Table 4, the F-distributions with 6 and 168 degrees of freedom are 99.78, 27.85 and 429.88 in terms of Mean, SD and Time respectively. Meanwhile, the p-values computed by the Iman-Davenport test are 0, 1.4166e-10 and 0 for Mean, SD and Time respectively. The critical value q_α is 2.4453, so the CD is 1.3872. A comparison of all compared algorithms using the CD based on Mean, SD and Time is shown in Fig.17. As illustrated in Fig.17(a), 7(b) and 17(c), NRO ranks first, second and third in Mean, SD and Time, respectively. As shown in Fig. 17(a), NRO has the first rank in term of Mean, which has a similar performance as LSHADE. In Fig. 17(b), LSHADE ranks first in SD, which has a similar performance as NRO that ranks second. In Fig. 17(c), NRO ranks third, which has a similar performance as LSHADE that ranks second in Time.

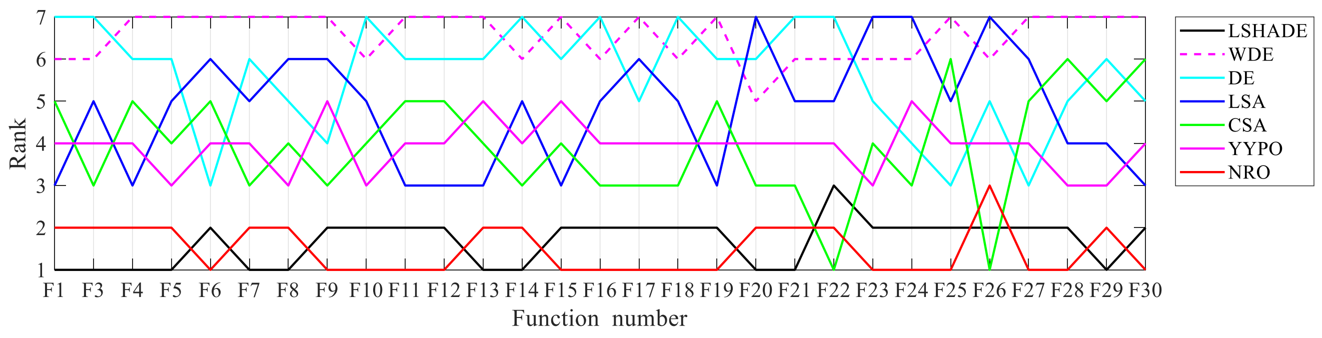


FIGURE 16. Parallel coordination plots of the ranks according to fitness obtained by seven algorithms on each function for 50D test.

TABLE 4.
AVERAGE RANKS OF NRO AND SIX ALGORITHMS FOR CEC 2018 TEST ACCORDING TO THE FRIEDMAN TEST (α=0.05)

Algorithms	LSHADE	WDE	DE	LSA	CSA	YYPO	NRO
Mean ranks[a]	1.6897	6.5862	5.6207	4.8276	3.9310	3.8966	1.4483
SD ranks[b]	1.8621	5.2069	3.9310	5.3793	4.6207	5.1034	1.8966
Time rank[c]	2.4828	6.9310	5	6.0690	3.3448	1	3.1724

a. 1) p-value computed by the Friedman test: 7.44287e-27. Chi-square is 135.87.
2) F-distribution with 6 and 168 degrees of freedom is 99.78.
3) p-value computed by the Iman-Davenport test: 0.

b. 1) p-value computed by the Friedman test: 1.42277e-16. Chi-square is 86.76.
2) F-distribution with 6 and 168 degrees of freedom is 27.85.
3) p-value computed by the Iman-Davenport Test: 1.4166e-10.

c. 1) p-value computed by the Friedman test: 1.15012e-32. Chi-square is 163.36.
2) F-distribution with 6 and 168 degrees of freedom is 429.88.
3) p-value computed by the Iman-Davenport Test: 0.

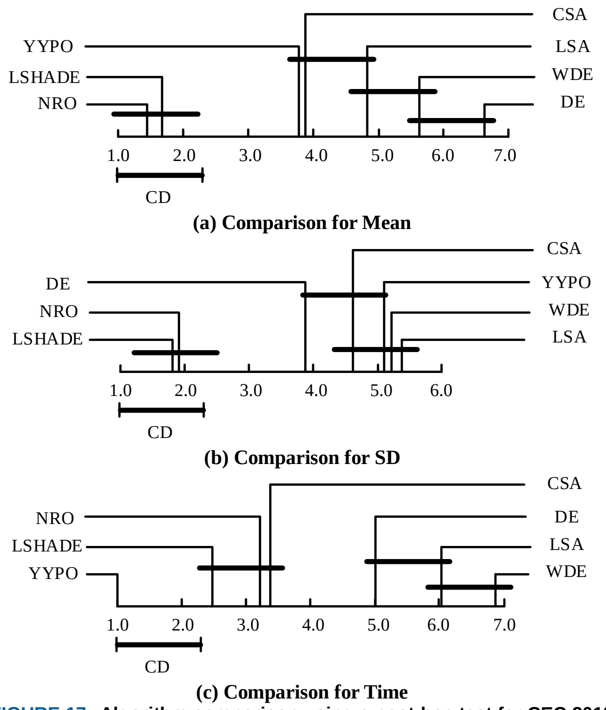


FIGURE 17. Algorithm comparison using a post-hoc test for CEC 2018 test; the algorithms are connected using the CD with less difference.

4.3 EXPERIMENT Ⅲ: CONSRAINED ENGINEERING DESIGN PROBLEMS

To test the ability of solving constrained optimization problem, three constrained engineering design problems, namely, welded beam design problem, pressure vessel design problem and tension/compression spring design problem are employed. The presented benchmark constrained engineering design problems [30] (see Appendix B) include objective functions and various constraints which consist of various numbers of design variables, inequality constraints and equality constraints. Additionally, the same constraints handing mechanism shown as (26) is applied for solving these real-world engineering problems. For constrained optimization problem, the commonly used constraint approaches can consist of six categories [99]: 1). Penalty function method; 2). Machine sorting method; 3). constraint handling method; 4). Multi-objective optimization method; 5). feasibility rule; and 6). Hybrid method. Because the penalty functions are simple, they can low the computational cost of engineering design problems.

$$f_p(\boldsymbol{x}) = f(\boldsymbol{x}) + 10^5\left(\sum_k \max_k\{g_k(\boldsymbol{x}), 0\}\right) \tag{26}$$

where $f_p(\boldsymbol{x})$ is the fitness value with penalty functions, $f(\boldsymbol{x})$ is the objective function value, 10^5 represents the penalty factor, and $g_k(\boldsymbol{x})$ is the kth constraint item.

In constrained optimization problem, AMO [3], WCA [75], LSA [51], GWO [30], SOS [36], TLBO [38], SFS [74], and ABC [27] are employed for comparing with NRO. The population size of all compared algorithms is set to 100, and MaxFEs are set to 50 000 for all algorithms. Optimization results are obtained from 30 independent runs. The results include statistical results, function evaluations and time consuming. A uniformly distributed population generation method is adopted by all compared algorithms. For termination criteria, when the fitness value is not changing, current function evaluation number is counted as one of results. The statistical results consist of variables, worst fitness value, mean fitness value, best fitness value and standard deviation. The settings of parameters for all algorithms in this experiment are given as follows:

(1) AMO: no-parameter algorithm;
(2) WCA: $N_{sr}=8$, $d_{\max}=0.001$ and $\mu=0.1$ as in [75];
(3) LSA: The channel time is set to 10 as in [51];
(4) GWO: $a=2-2\times(iter/iter_{\max})$ as in [30];
(5) SOS: no-parameter algorithm;
(6) TLBO: no-parameter algorithm;
(7) SFS: Maximum Diffusion Number (MDN) is set to 1 as in [74];
(8) ABC: $n_0=0.5\times N_{pop}$, $n_e=0.5\times N_{pop}$, $n_s=1$ and $Limit=n_e\times D$, where N_{pop} is population size of bee colony and D is the dimension of problem. As mentioned before, N_{pop} is set to 100 as in [27];
(9) NRO: $P_{Fi}=0.75$, $P_\beta=0.1$ and $freq=0.05$; the settings of parameters for Levy distribution strategy are used as in [2].

4.3.1 Welded beam design problem

This engineering design problem (see Appendix B.1) firstly proposed by Coello [100], consists of the design thickness of the weld h(denoted as x_1), the length of the clamped bar l(employed as x_2), the height of the bar t(used as x_3) and the thickness of the bar b(used as x_4) which can be seen in Fig. 18. In this problem, the purpose of the welded beam optimization problem is to minimize the overall cost subject to constraints which involve weld stress, buckling load, beam deflection and beam bending stress. This problem can be expounded in Appendix B.1.

As stated earlier, experiments of 30 independent times are performed to compare NRO with other algorithms including AMO [3], WCA [75], LSA [51], GWO [30], SOS [36] and TLBO [38]. The comparisons for best solutions obtained from various algorithms are displayed in Table S-18 (see supplementary material). Meanwhile, the statistical results using the algorithms mentioned above are shown in Table S-19 (see supplementary material). As you can see in Tables S-18 and S-19, NRO apparently performs the first rank among all compared algorithms in terms of solution quality. The solution quality means that the mean and SD values of NRO are best. Meanwhile, the best value is 1.724 852 308 597 365 with 32 100 FEs obtained by NRO. When NRO obtains the best solution, time cost equals to 0.2465s. Although WCA outperforms in the term of the number of FEs, NRO obtains the best fitness value. Though the number of FEs is very important for engineering design problem, the solution quality is more important. Therefore, NRO could offer a competitive best fitness value after less number of FEs. Fig. 19 displays the mean fitness values with respect to the number of FEs for this problem.

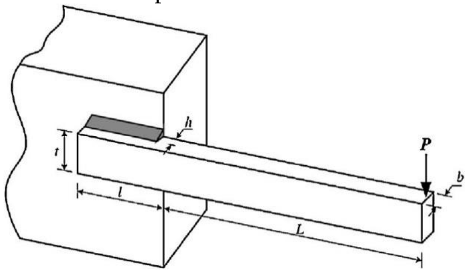


FIGURE 18. Welded beam design problem [100].

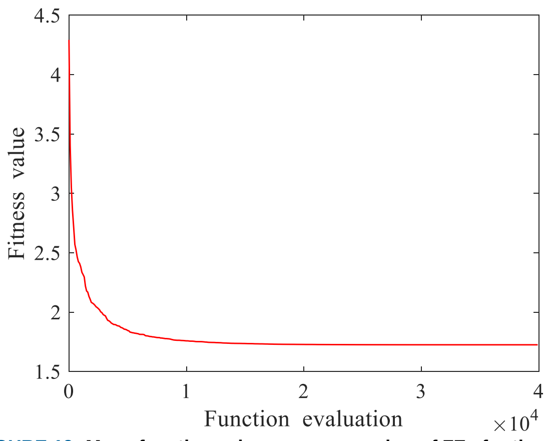


FIGURE 19. Mean function values versus number of FEs for the welded beam design problem.

4.3.2 Pressure vessel design problem

As a constrained benchmark problem, the engineering design problem called as pressure vessel design problem (see Appendix B.2) proposed by Kannan and Kramer [101], is presented for the aim to minimize the overall cost including the cost of material, forming and welding. The pressure vessel design problem whose cost is subject to four different constraints consists of four variables which include the thickness *Ts* (x_1), head's thickness *Th* (x_2), the inner radius *R* (x_3) and the length of cylindrical section *L* (x_4). Meanwhile, among the four design variables, the first and second variables are expected to be integer multiples of 0.0625 *in*, and the rests are continuous variables. Fig. 20 shows a vessel that is cylindrical in shape, whose ends have hemispherical heads. In addition, the problem is formulated in Appendix B.2.

This problem is independently solved 30 times by proposed NRO and previous algorithms including LSA, SFS [74], WCA, SOS, TLBO and GWO. The comparisons in terms of best solutions and statistical results obtained from above algorithms are shown in Tables S-20 and S-21 (see supplementary material). As can only been observed in Table S-20, in terms of quality of best solutions, NRO shows the superiority over other algorithms under same conditions. According to the summaries of Table S-21, the best value obtained from NRO is 5 835.332 773 616 458 after 46 700 FEs. When NRO obtains the best solution, time cost equals to 0.2268s. Although the number of FEs of WCA ranks the first, the accuracy of solution for NRO is first. In overall, NRO presents competitive best solution in less number of FEs than other algorithms. Fig. 21 displays the mean fitness values with respect to the number of FEs for this problem.

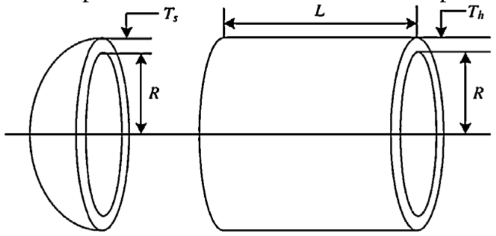


FIGURE 20. Pressure vessel design problem [101].

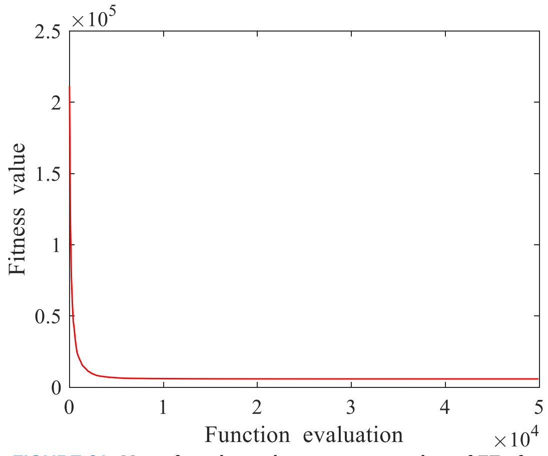


FIGURE 21. Mean function values versus number of FEs for the pressure vessel design problem.

4.3.3 Tension/compression spring design problem

This benchmark engineering problem called as tension/compression spring design problem (see Appendix B.3) proposed by Arora [102], has three variables which consist of the diameter of the wire *w* (x_1), the mean diameter of the coil *d* (x_2) and the length *L* (x_3). As shown in Fig. 22, the main aim of designing a spring is to achieve the minimization of weight which is subject to constraints on minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables. This problem is expounded in Appendix B.3.

The tension/compression spring design problem is independently solved 30 times by proposed NRO and other compared algorithms, including LSA, ABC [27], WCA, SOS, TLBO and GWO. The comparisons of the best solutions and statistical results among reported algorithms are summarized in Tables S-22 and S-23 (see supplementary material). As you can see in Table S-22, NRO performs better rank than other algorithms in the term of solution accuracy. In Table S-23, the best value obtained from NRO is 0.012 665 232 790 408 after 17 300 FEs. When NRO obtains the best solution, time cost equals to 0.2974s. Although NRO does not perform better than WCA in the term of FEs number, the best value of NRO rank the first among all algorithms. In overall, NRO can present competitive best solution in less number of FEs than other algorithms. Fig. 23 depicts the mean fitness values versus the number of FEs for this problem.

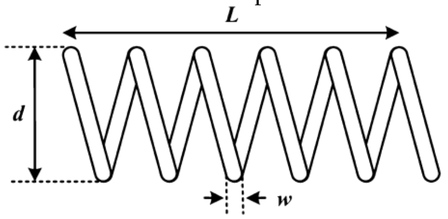


FIGURE 22. Tension/compression spring design problem [102].

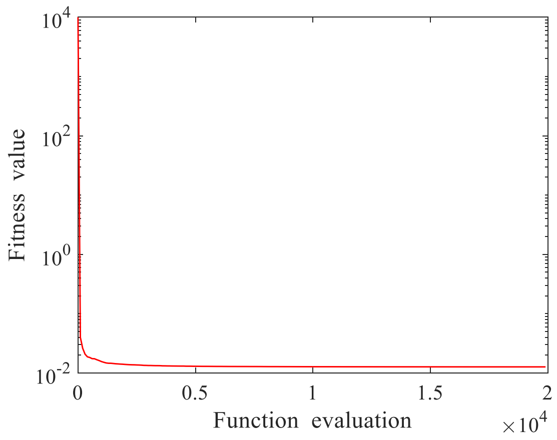


FIGURE 23. Function values versus number of iterations for the tension/compression spring problem

V. CONCLUSIONS

With the development of society, heuristic optimization approaches that have appealing advantages over classic algorithms have gained substantial popularity in applications of data mining, engineering design, decision and control. Currently, a variety of meta-heuristic approaches that imitate various phenomena and features in nature can incrementally reach a global optimum solution by employing operators and behaviors to efficiently search a solution space. Borrowed from the nuclear reaction process in physics, Nuclear Reaction Optimization is proposed as a novel method for global optimization. According to the definitions of nuclear reaction behaviors, the proposed algorithm can be divided into two phases—nuclear fission (NFi) and nuclear fusion (NFu). The NFi phase mainly mimics the process of nuclear fission of nuclei. Based on the types of nuclei and the β decay probability after bombardment, the Gaussian walk and differential operators between the nucleus and neutron have been employed for exploitation and exploration in nuclear fission. The NFu phase mainly simulates the nuclear fusion process of nuclei. In this phase, the NFu can include the ionization and fusion stages. On one hand, NRO uses the differential operator of each variable to generate the ion in the ionization stage, which effectively balances the exploitation and exploration abilities according to the ionization probability. On the other hand, NRO employs the differential operators of whole ions for creating the fusion product in the fusion stage, which mimics the exploitation and exploration based on the fusion probability. Meanwhile, variants of Levy flight are used for random searching to escape the local optima in each stage. NRO is proposed to optimize problems by employing the combination of the two phases.

To verify the superiority of NRO, unconstrained benchmark functions and constrained engineering design problems are tested. For twenty-three unconstrained benchmark functions and CEC 2018 test suite, the statistical results prove that NRO is more accurate, efficient and robust than other state-of-the-art algorithms in most cases. Furthermore, the results from engineering design problems show that NRO can provide the best performance in terms of solution quality. Therefore, the proposed NRO will most likely be a potential focus for searching for an optimal solution in various applications.

Moreover, future work can focus on two aspects of NRO—the variants of NRO and applications. Various operators and strategies can be employed to develop new variants of NRO. In addition to improving the algorithm mechanism, NRO can be hybridized with other optimization algorithms to enhance its performance. For the binary and multi-objective optimization problems, the corresponding versions of NRO can be studied. In terms of applications, focusing on the application of NRO in a wide spectrum of fields is of potential and practical significance. Due to the no free lunch theorem, the work of imitating and simulating the best operators and behaviors inspired by nature to solve optimization problems is endless.

APPENDIX

Appendix A

All codes used in this paper are obtained from below addresses:

ASO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/67011-atom-search-optimization-aso-algorithm

BBO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/52901-biogeography-based-optimization-bbo

BOA:
https://ww2.mathworks.cn/matlabcentral/fileexchange/68209-butterfly-optimization-algorithm-boa

CMAES:
http://dces.essex.ac.uk/staff/qzhang/

CS:
http://www.mathworks.com/matlabcentral/fileexchange/29809-cuckoo-search-cs-algorithm
GSA:
http://www.mathworks.com/matlabcentral/fileexchange/27756-gravitational-search-algorithm-gsa
HS:
https://ww2.mathworks.cn/matlabcentral/fileexchange/52864-harmony-search-hs
LSA:
https://ww2.mathworks.cn/matlabcentral/fileexchange/54181-lightning-search-algorithm-lsa
MVO:
http://www.alimirjalili.com/MVO.html
SSA:
http://www.alimirjalili.com/SSA.html
LSHADE:
http://www.pudn.com/Download/item/id/2840416.html.
DE:
https://ww2.mathworks.cn/matlabcentral/fileexchange/18593-differential-evolution
WDE:
https://ww2.mathworks.cn/matlabcentral/fileexchange/68370-weighted-differential-evolution-algorithm-wde.
CSA:
https://ww2.mathworks.cn/matlabcentral/fileexchange/56127-crow-search-algorithm
YYPO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/65558-yin-yang-pair-optimization-yypo
AMO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/65846-animal-migration-optimizer
WCA:
https://ww2.mathworks.cn/matlabcentral/fileexchange/56339-water-cycle-algorithm-wca.
GWO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/44974-grey-wolf-optimizer-gwo?s_tid=srchtitle.
SOS:
https://ww2.mathworks.cn/matlabcentral/fileexchange/47465-sos-m.
TLBO:
https://ww2.mathworks.cn/matlabcentral/fileexchange/65628-teaching-learning-based-optimization
SFS:
https://ww2.mathworks.cn/matlabcentral/fileexchange/47565-stochastic-fractal-search-sfs?requestedDomain=zh
ABC:
https://ww2.mathworks.cn/matlabcentral/fileexchange/65794-single-objective-artificial-bee-colony-optimization

Appendix B
B.1 Welded beam design problem

$$\min f(x_1,x_2,x_3,x_4)=1.10471x_1^2x_2+0.04811x_3x_4(14.0+x_2)$$

subject to:

$$\begin{cases} g_1(X)=x_1-x_4\le 0\\ g_2(X)=\delta-0.25\le 0\\ g_3(X)=\tau-13600\le 0\\ g_4(X)=\sigma-30000\le 0\\ g_5(X)=0.10471x_1^2+0.04811x_3x_4(14.0+x_2)-5\le 0\\ g_6(X)=0.125-x_1\le 0\\ g_7(X)=6000-F\le 0 \end{cases}$$

where the variables satisfy $0.1\le x_1,x_4\le 2.0$ and $0.1\le x_2,x_3\le 10$.

$$\begin{cases} \sigma=50400/x_3^2x_4\\ Q=6000(14+x_2/2)\\ D=\frac{1}{2}\sqrt{x_2^2+(x_1+x_3)^2}\ \beta=QD/J\\ J=\sqrt{2}x_1x_2\left(x_2^2/6+(x_1+x_3)^2/2\right)\\ \delta=65856/3000x_3^3x_4 \qquad \alpha=6000/\left(\sqrt{2}x_1x_2\right)\\ \tau=\sqrt{\alpha^2+\alpha\beta x_2/D+\beta^2}\\ F=0.61423\times10^6\frac{x_4^3x_3}{6}\left(1-\frac{x_3\sqrt{30/48}}{28}\right) \end{cases}$$

B.2 Pressure vessel design problem

$$\min f(x_1,x_2,x_3,x_4)=0.6224x_1x_3x_4+1.7881x_2x_3^2+3.1661x_1^2x_4+19.84x_1^2x_3$$

subject to:

$$\begin{cases} g_1(X)=-x_1+0.0193x_3\le 0\\ g_2(X)=-x_2+0.00954x_3\le 0\\ g_3(X)=-\pi x_3^2x_4-\frac{4}{3}\pi x_3^2+1296000\le 0\\ g_4(X)=x_4-240\le 0 \end{cases}$$

where the variables satisfy: $0\le x_1,x_2\le 100$ and $10\le x_3,x_4\le 200$.
B.3 Tension/compression spring design problem

$$\min f(x_1,x_2,x_3)=(x_3+2)x_1^2x_2$$

subject to:

$$\begin{cases} g_1(X) = 1 - \dfrac{x_2^2 x_3}{71785 x_1^4} \le 0 \\ g_2(X) = \dfrac{x_2(4x_2 - x_1)}{12566 x_1^3 (x_2 - x_1)} + \dfrac{1}{5108 x_1^2} - 1 \le 0 \\ g_3(X) = 1 - \dfrac{140.45 x_1}{x_2^2 x_3} \le 0 \\ g_4(X) = \dfrac{2(x_1 + x_2)}{3} - 1 \le 0 \end{cases}$$

where the variables satisfy: $0.05 \le x_1 \le 2$, $0.25 \le x_2 \le 1.3$ and $2 \le x_3 \le 15$.

REFERENCES

[1] A. Kaveh and T. Bakhshpoori, “Water Evaporation Optimization: A novel physically inspired optimization algorithm,” *Comput. Struct.*, vol. 167, pp. 69–85, 2016. DOI: 10.1016/j.compstruc.2016.01.008.

[2] Z. Zhang, C. Huang, S. Tang, K. Dong, and H. Huang, “An optimization method: Hummingbirds optimization algorithm,” *Journal of Systems Engineering and Electronics*., vol. 29, No. 2, pp.386-404, 2018. DOI: 10.21629/JSEE.2018.02.19.

[3] X. Li, J. Zhang, and M. Yin, “Animal migration optimization: An optimization algorithm inspired by animal migration behavior,” *Neural Comput. Appl.*, vol. 24, no. 7–8, pp. 1867–1877, 2014. DOI 10.1007/s00521-013-1433-8.

[4] S. J. Nanda and G. Panda, “A survey on nature inspired metaheuristic algorithms for partitional clustering,” *Swarm and Evolutionary Computation*, vol. 16. pp. 1–18, 2014. DOI: 10.1016/j.swevo.2013.11.003.

[5] H. Zang, S. Zhang, and K. Hapeshi, “A review of nature-inspired algorithms,” *J. Bionic Eng.*, vol. 7, no. SUPPL., 2010. DOI: 10.1016/S1672-6529(09)60240-7.

[6] M. Kumar, A. J. Kulkarni, and S. C. Satapathy, “Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology,” *Futur. Gener. Comput. Syst.*, vol. 81, pp. 252–272, 2018. DOI:10.1016/j.future.2017.10.052.

[7] B. Webb, “Swarm Intelligence: From Natural to Artificial Systems,” *Conn. Sci.*, vol. 14, no. 2, pp. 163–164, 2002.

[8] C. P. Lim and L. C. Jain, “Advances in swarm intelligence,” *Studies in Computational Intelligence*, vol. 248. pp. 1–7, 2009.

[9] S. P. Brooks and B. J. T. Morgan, “Optimization Using Simulated Annealing,” *Stat.*, vol. 44, no. 2, p. 241, 1995. DOI:10.2307/2348448.

[10] A. Y. S. Lam and V. O. K. Li, “Chemical-reaction-inspired metaheuristic for optimization,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 381–399, 2010. DOI: 10.1109/TEVC.2009.2033580.

[11] O. Castillo and L. T. Aguilar, “Genetic algorithms,” in *Studies in Fuzziness and Soft Computing*, vol. 373, 2019, pp. 23–39. DOI: 10.1007/978-3-030-03134-3_2.

[12] R. Storn and K. Price, “Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997. DOI: 10.1023/A:1008202821328.

[13] P. J. Angeline, “Genetic programming: On the programming of computers by means of natural selection,” *Biosystems*, vol. 33, no. 1, pp. 69–73, 1994. DOI: 10.1016/0303-2647(94)90062-0.

[14] L. J. Fogel, A. J. Owens, and M. J. Walsh, “Artificial intelligence through a simulation of evolution,” in *Evolutionary Computation: The Fossil Record*, 1998, pp. 230–248. DOI: 10.1109/9780470544600.ch7.

[15] A. M. Andrew, “Evolution and optimum seeking,” *Kybernetes*, vol. 27, no. 8. pp. 975–978, 1998. DOI: 10.1108/k.1998.27.8.975.2.

[16] A. Askarzadeh, “Bird mating optimizer: An optimization algorithm inspired by bird mating strategies,” *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 4, pp. 1213–1228, 2014. DOI: 10.1016/j.cnsns.2013.08.027.

[17] P. Civicioglu, “Backtracking Search Optimization Algorithm for numerical optimization problems,” *Appl. Math. Comput.*, 2013. DOI: 10.1016/j.amc.2013.02.017.

[18] M. Ghaemi, M. Feizi-Derakhshi, “Forest optimization algorithm,” *Expert. Syst. Appl.*, vol. 41, pp. 6676-6687, 2014. DOI: 10.1016/j.eswa.2014.05.009.

[19] A. H. Gandomi, “Interior search algorithm (ISA): A novel approach for global optimization,” *ISA Trans.*, vol. 53, no. 4, pp. 1168–1183, 2014. DOI: 10.1016/j.isatra.2014.03.018 .

[20] R. V. Rao, “Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems,” *International Journal of Industrial Engineering Computations*, vol. 7, pp. 19-34, 2016. DOI: 10.5267/j.ijiec.2015.8.004.

[21] S. Gao, C. Shao, Q. Gao, “Pseudo-collision in swarm optimization algorithm and solution: rain forest algorithm,” *Acta Phys. Sin.*, vol. 62, no. 19, pp. 1-16. DOI: 10.7498/aps.62.190202.

[22] Y. Sharafi, M. A. Khanesar, M. Teshnehlab, “COOA: competitive optimization algorithm,” *Swarm Evol. Comput.*, vol 30, pp. 39-63, 2016. DOI: 10.1016/j.swevo.2016.04.002.

[23] O. Olorunda and A. P. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," in Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on, 2008, pp. 1128-1134.

[24] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," *Evolutionary Computation, IEEE Transactions on*, vol. 9, pp. 126-142, 2005.

[25] R. Eberhart and J. Kennedy, “A new optimizer using PSO,” *Ann. Surg. Oncol.*, vol. 23, no. 10, pp. 3162–3167, 2016. DOI: 10.1245/s10434-016-5403-0.

[26] M. Dorigo and K. Socha, “Ant colony optimization,” in *Handbook of Approximation Algorithms and Metaheuristics*, 2007, pp. 26-1-26–14. DOI: 10.1201/9781420010749.

[27] D. Karaboga and B. Basturk, “An Artificial Bee Colony (ABC) Algorithm for Numeric function Optimization,” *IEEE Swarm Intell. Symp. 2006. Indianapolis, Indiana, USA*, pp. 789–798, 2006. DOI: 10.1007/978-3-540-72950-1_77.

[28] X. S. Yang, “A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization,” *Springer Berlin Heidelb.*, pp. 65–74, 2010. DOI: 10.1007/978-3-642-12538-6_6.

[29] X.-S. Yang, “Cuckoo Search,” in *Nature-Inspired Optimization Algorithms*, 2014, pp. 129–139. DOI: 10.1016/B978-0-12-416743-8.00009-9.

[30] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey Wolf Optimizer,” *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014. DOI: 10.1016/j.advengsoft.2013.12.007.

[31] S. Saremi, S. Mirjalili, and A. Lewis, “Grasshopper Optimisation Algorithm: Theory and application,” *Adv. Eng. Softw.*, vol. 105, pp. 30–47, 2017. DOI: 10.1016/j.advengsoft.2017.01.004.

[32] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, “Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems,” *Adv. Eng. Softw.*, vol. 114, pp. 163–191, 2017. DOI: 10.1016/j.advengsoft.2017.07.002.

[33] X. Qi, Y. Zhu, and H. Zhang, “A new meta-heuristic butterfly-inspired algorithm,” *J. Comput. Sci.*, vol. 23, pp. 226–239, 2017. DOI: 10.1016/j.jocs.2017.06.003.

[34] S. Mirjalili and A. Lewis, “The Whale Optimization Algorithm,” *Adv. Eng. Softw.*, vol. 95, pp. 51–67, 2016. DOI: 10.1016/j.advengsoft.2016.01.008.

[35] M. Jain, V. Singh, and A. Rani, “A novel nature-inspired algorithm for optimization: Squirrel search algorithm,” *Swarm and Evolutionary Computation*, 2018. DOI:10.1016/j.swevo.2018.02.013

[36] M. Y. Cheng and D. Prayogo, “Symbiotic Organisms Search: A new metaheuristic optimization algorithm,” *Comput. Struct.*, vol. 139, pp. 98–112, 2014. DOI:10.1016/j.compstruc.2014.03.007

[37] J. C. Bansal, H. Sharma, S. S. Jadon, and M. Clerc, “Spider Monkey Optimization algorithm for numerical optimization,” *Memetic Comput.*, vol. 6, no. 1, pp. 31–47, 2014. DOI 10.1007/s12293-013-0128-0

[38] F. Zou, L. Wang, X. Hei, and D. Chen, “Teaching-learning-based optimization with learning experience of other learners and its

application," *Appl. Soft Comput. J.*, vol. 37, pp. 725–736, 2015. DOI: 10.1016/j.asoc.2015.08.047.

[39] A. K. Shukla, P. Singh, and M. Vardhan, "Neighbour teaching learning based optimization for global optimization problems," in *Journal of Intelligent and Fuzzy Systems*, 2018, vol. 34, no. 3, pp. 1583–1594. DOI:10.3233/JIFS-169453

[40] M. Kumar, A. J. Kulkarni, and S. C. Satapathy, "Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology," *Futur. Gener. Comput. Syst.*, vol. 81, pp. 252–272, 2018. DOI:10.1016/j.future.2017.10.052

[41] M. Li, H. Zhao, X. Weng, and T. Han, "Cognitive behavior optimization algorithm for solving optimization problems," *Appl. Soft Comput. J.*, vol. 39, pp. 199–222, 2016. DOI:10.1016/j.asoc.2015.11.015

[42] S. J. Mousavirad and H. Ebrahimpour-Komleh, "Human mental search: a new population-based metaheuristic optimization algorithm," *Appl. Intell.*, vol. 47, no. 3, pp. 850–887, 2017. DOI 10.1007/s10489-017-0903-6.

[43] T. T. Huan, A. J. Kulkarni, J. Kanesan, C. J. Huang, and A. Abraham, "Ideology algorithm: a socio-inspired optimization methodology," *Neural Comput. Appl.*, vol. 28, pp. 845–876, 2017. DOI:10.1007/s00521-016-2379-4.

[44] Z. Z. Liu, D. H. Chu, C. Song, X. Xue, and B. Y. Lu, "Social learning optimization (SLO) algorithm paradigm and its application in QoS-aware cloud service composition," *Inf. Sci. (Ny).*, vol. 326, pp. 315–333, 2016. DOI: 10.1016/j.ins.2015.08.004.

[45] S. Satapathy, A. Naik, "Social group optimization (SGO): a new population evolutionary optimization technique," *Complex Intel. Syst.*, vol. 2, No. 3, pp.173 – 203, 2016.

[46] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Inf. Sci. (Ny).*, vol. 179, no. 13, pp. 2232–2248, 2009. 10.1016/j.ins.2009.03.004.

[47] G. Sun, P. Ma, J. Ren, A. Zhang, and X. Jia, "A stability constrained adaptive alpha for gravitational search algorithm," *Knowledge-Based Syst.*, vol. 139, pp. 200–213, 2018. DOI: 10.1016/j.knosys.2017.10.018.

[48] S. Wang, X. Da, M. Li, and T. Han, "Adaptive backtracking search optimization algorithm with pattern search for numerical optimization," *J. Syst. Eng. Electron.*, vol. 27, no. 2, pp. 395–406, 2016. DOI: 10.1109/JSEE.2016.00041.

[49] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 27, no. 2, pp. 495–513, 2016. DOI 10.1007/s00521-015-1870-7

[50] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems," *Appl. Soft Comput. J.*, vol. 13, no. 5, pp. 2592–2612, 2013. DOI:10.1016/j.asoc.2012.11.026

[51] H. Shareef, A. A. Ibrahim, and A. H. Mutlag, "Lightning search algorithm," *Applied Soft Computing Journal*, vol. 36. pp. 315–333, 2015. DOI: 10.1016/j.asoc.2015.07.028

[52] A. Kaveh and T. Bakhshpoori, "Water Evaporation Optimization: A novel physically inspired optimization algorithm," *Comput. Struct.*, vol. 167, pp. 69–85, 2016. DOI:10.1016/j.compstruc.2016.01.008

[53] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: Charged system search," *Acta Mech.*, vol. 213, no. 3–4, pp. 267–289, 2010. DOI 10.1007/s00707-009-0270-4

[54] A. Kaveh and M. Khayatazad, "A new meta-heuristic method: Ray Optimization," *Comput. Struct.*, vol. 112–113, pp. 283–294, 2012. DOI: 10.1109/NaBIC.2011.6089650.

[55] A. Kaveh and V. R. Mahdavi, "Colliding bodies optimization: A novel meta-heuristic method," *Comput. Struct.*, vol. 139, pp. 18–27, 2014. DOI: 10.1016/j.compstruc.2014.04.005.

[56] O. K. Erol and I. Eksin, "A new optimization method: Big Bang-Big Crunch," *Adv. Eng. Softw.*, vol. 37, no. 2, pp. 106–111, 2006. DOI: 10.1016/j.advengsoft.2005.04.005.

[57] A. Kaveh and A. Dadras, "A novel meta-heuristic optimization algorithm: Thermal exchange optimization," *Adv. Eng. Softw.*, vol. 110, pp. 69–84, 2017. DOI:10.1016/j.advengsoft.2017.03.014

[58] A. Kaveh and M. Ilchi Ghazaan, "Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints," *Acta Mech.*, vol. 228, no. 1, pp. 307–322, 2017. DOI: 10.1007/s00707-016-1725-z.

[59] W. Zhao, L. Wang, and Z. Zhang, "A novel atom search optimization for dispersion coefficient estimation in groundwater," *Futur. Gener. Comput. Syst.*, vol. 91, pp. 601–610, 2019. DOI:10.1016/j.future.2018.05.037.

[60] A. Tabari and A. Ahmad, "A new optimization method: Electro-Search algorithm," *Comput. Chem. Eng.*, vol. 103, pp. 1–11, 2017. DOI:10.1016/j.compchemeng.2017.01.046

[61] B. Doɪan and T. Ölmez, "A new metaheuristic for numerical function optimization: Vortex Search algorithm," *Inf. Sci. (Ny).*, vol. 293, pp. 125–145, 2015. DOI:10.1016/j.ins.2014.08.053

[62] B. Alatas, "ACROA: Artificial Chemical Reaction Optimization Algorithm for global optimization," *Expert. Syst. Appl.*, vol. 38, pp. 13170-13180, 2011. DOI: 10.1016/j.eswa.2011.04.126.

[63] V. Muthiah-Nakarajan, M. M. Noel, "Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion," *Appl. Soft. Comput.*, vol. 38, pp. 771-787, 2016. DOI: 10.1016/j.asoc.2015.10.034.

[64] H. Abedinpourshotorban, S. M. Shamsuddin, Z. Beheshti, D. N. Jawawi, "Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm," *Swarm Evol. Comput.*, vol. 26, pp. 8-22, 2016. DOI: 10.1016/j.swevo.2015.07.002.

[65] A. A. Hudaib, H. N. Fakhouri, "Supernova Optimizer: A novel natural inspired meta-heuristic," *Modern Applied Science*, vol. 12, no. 1, pp. 32-50, 2018. DOI: 10.5539/mas.v12n1p32.

[66] A. Tzanetos, G. Dounias, "A new metaheuristic method for optimization: sonar inspired optimization," pp. 417-428, 2017. DOI: 10.1007/978-3-319-65172-9_35.

[67] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," in *Information Sciences*, 2013, vol. 237, pp. 82–117. DOI: 10.1016/j.ins.2013.02.041.

[68] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, 2009. DOI: 10.1109/TEVC.2008.927706.

[69] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, 2014, pp. 1658–1665. DOI: 10.1109/CEC.2014.6900380.

[70] M. Z. Ali, N. H. Awad, P. N. Suganthan, and R. G. Reynolds, "An Adaptive Multipopulation Differential Evolution with Dynamic Population Reduction," *IEEE Trans. Cybern.*, 2017. DOI: 10.1109/TCYB.2016.2617301.

[71] L. Tang, X. Wang, and Z. Dong, "Adaptive Multiobjective Differential Evolution With Reference Axis Vicinity Mechanism," *IEEE Transactions on Cybernetics*, 2018. DOI: 10.1109/TCYB.2018.2849343.

[72] X. G. Zhou and G. J. Zhang, "Abstract Convex Underestimation Assisted Multistage Differential Evolution," *IEEE Transactions on Cybernetics*. 2017. DOI: 10.1109/TCYB.2017.2710626.

[73] X. G. Zhou and G. J. Zhang, "Differential evolution with underestimation-based multimutation strategy," *IEEE Trans. Cybern.*, 2019. DOI:10.1109/TCYB.20182801287.

[74] H. Salimi, "Stochastic Fractal Search: A powerful metaheuristic algorithm," *Knowledge-Based Syst.*, vol. 75, pp. 1–18, 2015. DOI:10.1016/j.knosys.2014.07.025

[75] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm - A novel metaheuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.*, vol. 110–111, pp. 151–166, 2012. DOI:10.1016/j.compstruc.2012.07.010

[76] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, 1997. DOI: 10.1109/4235.585893.

[77] J. E. Fergusson, "The history of the discovery of nuclear fission," *Found. Chem.*, vol. 13, no. 2, pp. 145–166, 2011. DOI 10.1007/s10698-011-9112-2

[78] E. Rutherford, "The scattering of α and β particles by matter and the structure of the atom," *Philos. Mag. 6th Ser.*, vol. 21, pp.669–688, 1911.

[79] E. Rutherford, "Collision of α-particles with light atoms," *Nature*,

vol. 103, no. 2595. pp. 415–418, 1919. DOI:10.1038/103415a0

[80] S. Goriely, “The fundamental role of fission during r-process nucleosynthesis in neutron star mergers,” *European Physical Journal A*, vol. 51, no. 2. pp. 1–21, 2015. DOI: 10.1140/epja/i2015-15022-3.

[81] R. Capote, M. Herman, P. Obložinský, P. G. Young, S. Goriely, T. Belgya, A. V. Ignatyuk, A. J. Koning, S. Hilaire, V. A. Plujko, M. Avrigeanu, O. Bersillon, M. B. Chadwick, T. Fukahori, Z. Ge, Y. Han, S. Kailas, J. Kopecky, V. M. Maslov, G. Reffo, M. Sin, E. S. Soukhovitskii, and P. Talou, “RIPL - Reference Input Parameter Library for Calculation of Nuclear Reactions and Nuclear Data Evaluations,” *Nucl. Data Sheets*, vol. 110, no. 12, pp. 3107–3214, 2009. DOI: 10.1016/j.nds.2009.10.004.

[82] L. R. Grisham, “Nuclear Fusion,” in *Future Energy: Improved, Sustainable and Clean Options for our Planet*, 2013, pp. 199–211. DOI: 10.1016/B978-0-08-099424-6.00010-7.

[83] B. C. Reed, “Energy Release in Nuclear Reactions, Neutrons, Fission, and Characteristics of Fission,” *The Physics of the Manhattan Project*, pp.1-48, 2015.

[84] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, “An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems,” in *2016 IEEE Congress on Evolutionary Computation, CEC 2016*, 2016, pp. 2958–2965. DOI: 10.1109/CEC.2016.7744163.

[85] S. Arora and S. Singh, “Butterfly optimization algorithm: a novel approach for global optimization,” *Soft Comput.*, 2018. DOI:10.1007/s00500-018-3102-4

[86] Z. W. Geem, J. H. Kim, and G. V. Loganathan, “A New Heuristic Optimization Algorithm: Harmony Search,” *Simulation*, vol. 76, no. 2, pp. 60–68, 2001. DOI: 10.1177/003754970107600201.

[87] X. Zhang, Q. Kang, J. Cheng, and X. Wang, “A novel hybrid algorithm based on Biogeography-Based Optimization and Grey Wolf Optimizer,” *Appl. Soft Comput. J.*, vol. 67, pp. 197–214, 2018. DOI:10.1016/j.asoc.2018.02.049.

[88] C. Igel, N. Hansen, and S. Roth, “Covariance matrix adaptation for multi-objective optimization,” *Evol. Comput.*, vol. 15, no. 1, pp. 1–28, 2007. DOI: 10.1162/evco.2007.15.1.1

[89] H. Wang, S. Rahnamayan, H. Sun, and M. G. H. Omran, “Gaussian bare-bones differential evolution,” *IEEE Trans. Cybern.*, 2013. DOI: 10.1109/TSMCB.2012.2213808.

[90] X. G. Zhou, G. J. Zhang, X. H. Hao, and L. Yu, “A novel differential evolution algorithm using local abstract convex underestimate strategy for global optimization,” *Comput. Oper. Res.*, 2016. DOI: 10.1016/j.cor.2016.05.015.

[91] M. D. Li, H. Zhao, X. W. Weng, and T. Han, “A novel nature-inspired algorithm for optimization: Virus colony search,” *Adv. Eng. Softw.*, vol. 92, pp. 65–88, 2016. DOI:10.1016/j.advengsoft.2015.11.004.

[92] R. Moghdani, K. Salimifard, “Volleyball Premier League Algorithm,” *Appl. Soft. Computing*, vol. 64, pp. 161-185, 2018. DOI: 10.1016/j.asoc.2017.11.043.

[93] N.H. Awad, M.Z. Ali, J.J. Liang, B.Y. Qu, P.N. Suganthan, “Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization,” Nanyang Technological University, Jordan University of Science and Technology and Zhengzhou University, Tech. Rep., 2016.

[94] Civicioglu, E. Besdok, M. Gunen, and U Atasever, “Weighted Differential Evolution Algorithm for Numerical Function Optimization: A Comparative Study with Cuckoo Search, Artificial Bee Colony, Adaptive Differential Evolution, and Backtracking Search Optimization Algorithms,” *Neural Comput. Appl.*, In Press. 2018.

[95] V. Punnathanam, P. Kotecha, “Yin-Yang-pair Optimization: A novel lightweight optimization algorithm,” *Eng. Appl. Artif. Intel.*, vol. 54, pp. 62-79, 2016. DOI: 10.1016/j.engappai.2016.04.004.

[96] A. Askarzadeh, “A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm,” *Comput. Struct.*, vol. 169, pp. 1-12, 2016. DOI: 10.1016/j.compstruc.2016.03.001.

[97] S. Das, A. Mandal, and R. Mukherjee, “An adaptive differential evolution algorithm for global optimization in dynamic environments,” *IEEE Trans. Cybern.*, 2014. DOI: 10.1109/TCYB.2013.2278188.

[98] C. Li, T. T. Nguyen, M. Yang, M. Mavrovouniotis, and S. Yang, “An Adaptive Multipopulation Framework for Locating and Tracking Multiple Optima,” *IEEE Trans. Evol. Comput.*, 2016. DOI: 10.1109/TEVC.2015.2504383.

[99] Y. Nesterov, “Nonlinear optimization,” in *Springer Optimization and Its Applications*, 2018. DOI: 10.1007/978-3-319-91578-4_1.

[100] C. A. Coello Coello, “Use of a self-adaptive penalty approach for engineering optimization problems,” *Comput. Ind.*, 2000. DOI: 10.1016/S0166-3615(99)00046-9.

[101] B. K. Kannan and S. N. Kramer, “An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design,” *J. Mech. Des.*, 2008. DOI: 10.1115/1.2919393.

[102] J. S. Arora, *Introduction to Optimum Design*. 2004. DOI: 10.1016/B978-0-12-064155-0.X5000-9.