Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

# New Caledonian crow learning algorithm: A new metaheuristic algorithm for solving continuous optimization problems

Wedad Al-Sorori, Abdulqader M. Mohsen *

*University of Science and Technology, Sana'a, Yemen*

ARTICLE INFO

*Article history:*
Received 13 September 2019
Received in revised form 31 March 2020
Accepted 16 April 2020
Available online 28 April 2020

*Keywords:*
Social learning
Swarm intelligence
Observational learning
Nature-inspired computing
Socio-inspired optimization

ABSTRACT

Several metaheuristic algorithms have been introduced to solve different optimization problems. Such algorithms are inspired by a wide range of natural phenomena or behaviors. We introduced a new metaheuristic algorithm called New Caledonian (NC) crow learning algorithm (NCCLA), inspired by efficient social, asocial, and reinforcement mechanisms that NC-crows use to learn behaviors for developing tools from *Pandanus* trees to obtain food. Such mechanisms were modeled mathematically to develop NCCLA, whose performance was subsequently evaluated and statistically analyzed using 23 classical benchmark functions and 4 engineering problems. The results verify NCCLA's performance efficiency and highlight its accelerated convergence and ability to escape from local minima. An extensive comparative study was conducted to demonstrate that the solution accuracy and convergence rate of NCCLA were better than those of other state-of-the-art metaheuristics. The results also indicate that NCCLA is a promising algorithm that can be applied to solve other optimization and real-world problems.

## 1. Introduction

The main goal of solving optimization problems is to find optimal solutions using an appropriate exact, heuristic, or metaheuristic algorithms. Most real-world activities are optimized under some predefined constraints to achieve particular objectives. When attempting to find optimal solutions to real-world problems, we usually encounter a lack of resources, including time and budget [1,2]. Metaheuristic algorithms have generated high-quality, near-optimal solutions for both discrete and continuous problems. In addition, they exhibit an excellent ability to solve a wide range of real-world problems within a reasonable time frame. The search process of any metaheuristic algorithm is composed of two interchanging stages; exploitation(intensification) and exploration (diversification). Although finding an appropriate balance between these two phases represents the performance key of an algorithm, it is still a challenging task due to the stochastic nature of metaheuristics [1–4].

Nature-inspired metaheuristic algorithms are based on biological, mathematical, physical, vegetative, or social phenomena. Most social phenomena used for optimization are referred to as swarm intelligence (SI) or collective intelligence, representing collaborations between organisms [5,6]. SI is based on the learning processes of a group of individuals, each of whom can perform tasks with others in the same group to increase the overall group fitness. In other words, SI focuses on group effectiveness and its overall contribution to accomplish tasks or important goals [5–7]. Compared with traditional optimization techniques, SI algorithms have many advantages that include the following: (i) they are readily used with only minor revisions in various fields; (ii) they often use memory to retain the best solution achieved so far; (iii) SI preserves the optimization information during iterations; (iv) fewer parameters must be adjusted; therefore, SI algorithms are easy to implement. Accordingly, several SI algorithms have been inspired by collective behaviors such as particle swarm optimization (PSO) based on birds flocking together through space [8], firefly (FF) optimization based on the mutual attraction among fireflies [9], and ant colony optimization (ACO) based on behaviors of foraging-ant groups [10]. Despite several metaheuristic optimization algorithms published in the literature, demonstrating efficiency in tackling difficult optimization problems, both SI and other metaheuristics are limited by premature convergence, trapping at local optima, and lack of balance between exploitation and exploration.

Learning behavior is a concept used to describe a set of complex processes, enabling species to adapt to their environment or situation by continuously adjusting their behaviors [11], whereby individuals employ different social or asocial learning mechanisms, as necessary. In social learning, instead of determining appropriate behavior using numerous trial-and-error iterations, also known as asocial learning, individuals guide their learning

* Corresponding author.
*E-mail addresses:* w.alsrori@ust.edu (W. Al-Sorori), a.alabadi@ust.edu (A.M. Mohsen).

https://doi.org/10.1016/j.asoc.2020.106325
1568-4946/© 2020 Elsevier B.V. All rights reserved.

using the information provided by other group members, typically those with more experience [11,12]. Such learning enables individuals to adapt and evolve complex behaviors found in their societies. Asocial learning generally is more costly than social learning. Social learning can be found in human and animal societies. According to the cost and reliability required to accomplish certain behaviors, animals can conditionally behave, depending on social and asocial information [11].

The earliest attempt at optimizing social-learning mechanisms was made by Montes de Oca and Stützle in 2008 [13], wherein an incremental social-learning framework was proposed and combined with PSO. The authors found that incremental social learning could accelerate optimization and learning processes and improve the quality of solutions and optimization strategies. However, their work does not represent an independent variant of PSO. Later, Cheng and Jin introduced an independent variant of PSO called social learning PSO (SLPSO) that incorporates a social-learning mechanism known as limitation into PSO. Particles in SLPSO learn from better particles known as demonstrators in the swarm instead of learning from historically best particle positions [14].

The learning mechanisms of human beings have attracted the attention of researchers regarding the development of several algorithms such as the teaching–learning based optimization algorithm (TLBO) [15], social learning behavior based on the Bandura' social learning theory (SLA) [16], the socio evolution and learning optimization algorithm (SELO) [17], the simple human learning optimization algorithm (HLO) [18] and diverse human learning optimization (DHLO) [19]. Although the recent studies have assumed that inspiration from human society may yield in developing more efficient metaheuristics, algorithms inspired by the social phenomenon of human beings still produce equivalent or comparable results to those algorithms inspired by social animals and insects [20,21].

Crows are members of a widely known genus of birds, considered among the most intelligent animals in the world. They have demonstrated distinctive behaviors such as: (i) self-awareness in mirror tests; (ii) tool-making and using ability; (iii) faces recognition and warning each other when meeting unfriendly face, (iv) communicate in sophisticated ways; and (v) food hiding [11,12, 22–26]. Such behaviors have attracted significant interest from researchers developing algorithms for solving optimization problems. For example, the crow search algorithm (CSA) was inspired by the food-hiding behavior of crows [27]; it imitates the ability of crows to hide excess food in certain locations and retrieve it as needed. The raven roosting optimization (RRO) algorithm is another crow algorithm inspired by the roosting and foraging behaviors of ravens [28]. In RRO, the leader is the raven that has the best food source. Some ravens follow the leader to obtain a food source, and the remaining ravens fly toward positions of the best food sources that they have identified thus far. However, like most metaheuristic algorithms, CSA and RRO exhibit premature convergence and get stuck at local optima [29,30].

NC-crows are medium in size, all-black members of the *Corvidae* family and have relatively large brains [11,12]. NC-crows are the most well-known interactive birds owing to their intelligence and adaptability, and they have exceptional abilities such as problem-solving and tool development. NC-crows can develop plant-based tools (usually from *Pandanus* trees) including sticks, leaf-based tools, and hooks used to capture prey or grubs hiding in nooks, cracks, or crevices. NC-crows have developed *Pandanus* tool-making skills through both social and asocial (i.e., individual) learning. The NC-crow social system encourages the faithful transmission of information about local tool-design by maximizing vertical learning from the parent to the offspring and minimizing horizontal learning among unrelated individuals [11,12]. NC-crow behavior has many similarities with the optimization process; hence it can be modeled to build an effective optimization algorithm that provides a good balance between exploration and exploitation. To the best of our knowledge, there has been no previous study in the literature inspired by the crow-learning mechanism to solve optimization problems.

Herein, we propose an optimization algorithm inspired by the learning mechanism of NC-crows developing *Pandanus* tools, called new NC-crow learning algorithm (NCCLA). Our motivations of introducing such algorithm are as follows: (i) the intelligent and distinguished behavior of learning mechanism of NC-crows to manufacture *Pandanus* tools; (ii) the previous studies highlighted that the self-improvement through the learning process is more direct and rapid than the natural evolution of genotypes [31,32]. As a result, employing the learning process techniques in NC-crows may lead to developing more effective algorithm than existing SI algorithms; (iii) it can also be noticed from the literature that the recently developed algorithms such as golden ratio optimization method (GROM) [33], the grey wolf optimizer (GWO) [34], the whale optimization algorithm (WOA) [35], the pathfinder algorithm (PFA) [36], the social learning algorithm (SLA) [16], and SLPSO [14] achieved better results and outperformed other traditional algorithms such as GA, ACO and PSO; (iv) the no free lunch (NFL) theorem states that the absolute superiority of an algorithm to solve all optimization algorithms cannot be claimed [37,38]. Accordingly, it keeps this domain of research open to either improve the existing algorithms such as [39–42] or propose new algorithms for higher performance; and (v) the rapid advancement in technology today leads increasingly to many complex optimization problems [43]. Thus, the scientific community is still developing new optimization techniques to solve new and more complex optimization problems to reach a better design.

In NCCLA, the parent and juvenile NC-crows represent agents optimizing the search. Each juvenile in NCCLA tries to improve its own behavior and skill to develop *Pandanus* tools by continuously altering its own behavioral attributes, which can be achieved by observing other demonstrators in its environment, copying some interesting behavioral attributes, and then reinforcing them to habituate better behaviors. In other words, NCCLA consists of learning and reinforcement phases. In learning, a juvenile can improve its behavior by learning socially or asocially (i.e., individually), representing a local or global search, respectively. Social learning is a mechanism that enables juveniles to copy behavioral attributes from their parents or more-experienced siblings. Learning individually is the mechanism that allows juveniles to learn individually by trial and error or by retaining its previous attributes. In the reinforcement phase, some newly learned juvenile attributes and parent's behaviors may be influenced by positive or negative rewards to intensively or extensively search among the search space more effectively. In brief, the proposed work has the following contributions.

(1) The main contribution is the development of a nature-inspired NC-crow learning algorithm. NC-crows' behaviors toward developing *Pandanus* tools to obtain food are studied in detail and are subsequently modeled mathematically to guarantee a good balance between exploration and exploitation.

(2) The updating and searching mechanisms in NCCLA that simulate learning and reinforcement mechanisms of NC-crows. Different mechanisms in learning and reinforcement phases enable the NCCLA to effectively solve different functions of varying complexity, containing 23 benchmark test functions and four classical engineering problems. Experimental results highlight the high performance of NCCLA compared to other algorithms, including natural-phenomenon-based such as GROM; swarm-based such as

GWO, WOA, PFA, and CSA; and social-learning-based such as SLA and SLPSO.

(3) Rigorous and extensive comparative study is conducted with other metaheuristics using statistical analysis and different performance analysis including exploration, exploitation and convergence behavior analysis.

The rest of the paper is organized as follows. Section 2 introduces the inspiration for the current study. The proposed algorithm is described in detail and is implemented in Section 3. Differences between NCCLA and other metaheuristics are discussed in Section 4. In Section 5, NCCLA performance is detailed and compared with those of other state-of-the-art optimization algorithms using optimization problems with different levels of complexity. The conclusions of this study are presented in Section 6 by summarizing the key findings and their significance and by proposing a direction for future research based on the current findings.

## 2. Inspiration

A recent field study by Holzhaider et al. [12] investigated social learning in NC-crows based on the creatures eating habits. A type of corvid family attracted the attention of researchers owing to their exceptional behaviors and tool manufacturing skills. These crows create three different types of tools from the leaves of *Pandanus* trees, i.e. wide, narrow, and stepped tools. These tools enable them to later probe for insects in holes in *Pandanus* or other types of trees. NC-crows were shown to develop their *Pandanus* tool-design skills through both asocial and social learning [11,44]. Fig. 1 illustrates an NC-crow holding a self-produced wide *Pandanus* tool (Fig. 1(a)) and subsequently using it to reach the food within a tree hole (Figs. 1(b, c, d)).

Holzhaider et al. [45] found that the social organization of NC-crows may be suitable for enabling incremental, cumulative technology. That exploits different forms of learning, such as social and asocial learning. Authors also pointed out that juveniles take more than one year to reach adults-like tool-design skills using different learning mechanisms. Learning by trial and error, asocial, is one of the essential learning mechanisms that enable juveniles to develop their skills as well as increase diversity and novelty. In addition, juveniles can improve their skills with social learning. Social learning can be obtained by the observation of others or interacting with others discarded ready-made tools or with the counterparts that already found on *Pandanus* tree leaves. The social system of NC encourages faithful information transformation about local tool-design by maximizing vertical learning, from parent to offspring, and minimizing horizontal learning, between unrelated individuals.

Studies on animal social learning have also highlighted concepts of behavioral imitation, reinforcement, and reward [46–48]. Reinforcement is a source of information feedback that individuals can obtain either from other group members or by simply performing the same behavior as the other members. Reinforcement is conducted either by knowing whether the other members have been rewarded or punished for their behaviors or a reward for conformity. Moreover, social learning theories such as Bandura theory [49] posited that individuals could positively or negatively reinforce learned behaviors either by increasing or decreasing the intensity of the learned attributes, respectively.

## 3. NC-Crow learning optimization algorithm (NCCLA)

Inspired by the aforementioned crow-learning mechanism, a new population-based optimization algorithm called NCCLA was developed based on the following assumptions.

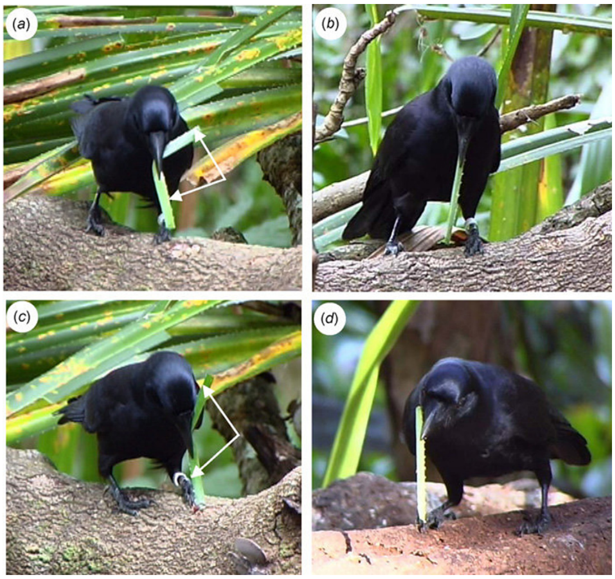

**Fig. 1.** New Caledonian crows holding-wide-pandanus-tools, taken from [50].

(1) Juvenile NC-crows exchange between social and asocial learning to develop their tool-design skills. Because this is not a dangerous behavior, such as dealing with predators, juveniles will depend extensively on social learning and depend slightly on individual or asocial learning.

(2) NC-crows live in a relatively stable environment; thus, they frequently rely on social learning.

(3) NC-crows are selective by nature; based on their selection strategy, they can observe and select the crows and the tool-development behaviors that they will copy.

(4) Each crow has one cognitive design template to develop through incremental evolution before starting to build and use tools.

(5) Parents update their behavioral attributes by exploiting their own experience and knowledge.

Table 1 shows the analogy between crow learning tool-development concepts and optimization concepts.

Fig. 2 illustrates how juvenile $i$ in crow family $F$ can improve its behavior, comprising four social and asocial learning attributes. Juvenile $i$ can socially learn by observing other elder, more-experienced family members. It almost exclusively learns by observing its parents, from whom it may copy specific attributes from parent 1 or 2. As shown from the figure, the juvenile $i$ copies two attributes from his first and second parents, including “$P11$” and “$P22$”, respectively. Fig. 2 also illustrates that juvenile $i$ copies some attributes from its older, more-experienced sibling, as shown with the “$S13$” attribute. Thus, it can copy attributes from sibling one but not from sibling two. By asocial learning, juvenile $i$ can also keep certain attributes representing its historical experience or update them randomly through trial and error, as shown with the $Rand(L, U)$ attribute.

Similar to other population-based algorithms, NCCLA starts with random initial behaviors for each juvenile. Each behavior is represented by a vector of behavioral attributes in a D-dimensional search space. Then, NCCLA improves these behaviors using different learning and reinforcement techniques iteratively. Finally, it returns the best behavior vector when one of the terminal condition is satisfied. The power of NCCLA as a stochastic algorithm is owing to its probabilistic nature, which guarantees that the algorithm does not get trapped at local optima. The following subsections illustrate NCCLA steps in detail.

**Table 1**
Analogy between crow learning tool-development concepts and optimization concepts.

| Tool-development learning concepts | Optimization concepts |
|---|---|
| Crow's behavioral attributes | Design variables |
| Crow's manufacturing behavior | Solution |
| Behavior quality | Objective function |
| Learning process | Solution enhancement |
| Practice or experience | Iterations |
| Crow's adult-like tool-development behavior | Optimal or near-optimal solution |

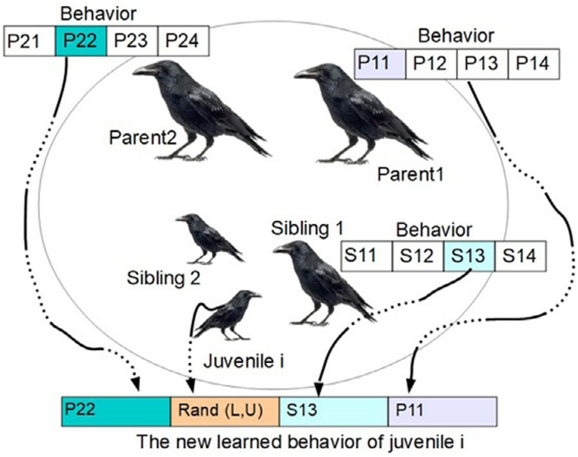

**Fig. 2.** Updating behavioral schema for juvenile *i*.

### 3.1. Initialization

First, algorithm parameters such as number of crows $n$ within a family $F$, reinforcement probability ($Rp_{prob}$), social learning probability ($SL_{prob}$), vertical learning probability ($VSL_{prob}$), first parent selection probability ($P1_{prob}$), trial-and-error probability ($TaE_{prob}$), and the maximum number of iterations are specified. Then, population generation is randomly defined such that there exist $n$ crows in family $F$ representing the population, and behavior of the $i_{th}$ crow is represented by solution vector $X_i$. Behaviors of all the crows in $F$ are represented by a matrix, as shown in Eq. (1).

$$F = \begin{bmatrix} (X_{11} & X_{12} & X_{13} & \cdots & X_{1d}) \Rightarrow X_1 \\ (X_{21} & X_{22} & X_{23} & \cdots & X_{2d}) \Rightarrow X_2 \\ (X_{31} & X_{32} & X_{33} & \cdots & X_{3d}) \Rightarrow X_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (X_{n1} & X_{n2} & X_{n3} & \cdots & X_{nd}) \Rightarrow X_n \end{bmatrix}, \tag{1}$$

where $X_{i,j}$ represents attribute $j$ in behavior $X_i$ of crow $i$. Uniform distribution, as shown in Eq. (2), is used to allocate initial behaviors of each crow.

$$X_{i,j} = x_L + U(0,1) \times (x_U - x_L), \tag{2}$$

where $x_L$ and $x_U$ are lower and upper bounds of behavioral attributes of the $i_{th}$ crow, respectively, and $U(0, 1)$ is a uniformly distributed random number in the range [0, 1].

After that, the fitness (i.e., quality) of each behavior is calculated according to behavioral attributes (i.e., decision variables) for the problem under consideration. The fitness function of the corresponding behavior is calculated. Fitness of each crow's behavior represents the goodness of behavior when compared to corresponding adult-like behavior, i.e., the optimal solution. Accordingly, the crow population is sorted from the best to the worst according to the fitness function. Then, the two best solutions in positions 1 and 2 are selected as parents, denoted $X_1$ and $X_2$, respectively. The rest $n-2$ crows represent the juveniles within the family. Eq. (3) shows the representation of the fitness function.

$$f = \begin{bmatrix} f_1([x_{11} & x_{12} & x_{13} & \cdots & x_{1d}]) \\ f_2([x_{21} & x_{22} & x_{23} & \cdots & x_{2d}]) \\ f_3([x_{31} & x_{32} & x_{33} & \cdots & x_{3d}]) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f_n([x_{n1} & x_{n2} & x_{n3} & \cdots & x_{nd}]) \end{bmatrix} \tag{3}$$

### 3.2. Learning phase

In the learning phase, each individual or crow in the family improves its behavior by employing different learning mechanisms, including social or asocial learning. These mechanisms enable the algorithm to strike a good balance between exploration and exploitation and are illustrated as follows.

During the learning phase, each juvenile $i$ in the family tries to update its behavioral attributes either socially or individually according to specific probabilities, $SL_{prob}$ or $1 - SL_{prob}$, respectively. Each juvenile will try to intensify the search in promising areas, by updating its behavioral attributes socially, and to explore the search space by updating behavioral attributes individually. High $SL_{prob}$ enables search agents of NCCLA to intensify the search in promising areas, and low $SL_{prob}$ leads agents to do more exploration. Owing to characteristics of NC-crows, juveniles extensively rely on social learning, which is the mechanism whereby juveniles learn vertically from their parents or horizontally from their older, more-experienced siblings. Juveniles can also learn individually by trial and error or by retaining previous attributes. Strictly speaking, newly adjusted behavior is composed of attributes copied from several demonstrators and learned individually.

#### 3.2.1. Social learning

When juvenile $i$ chooses to learn socially according to $SL_{prob}$, it can decide to learn vertically from its parents or horizontally from its older, more-experienced siblings according to a predefined probability, $VSL_{prob}$ or $1 - VSL_{prob}$, respectively. High $VSL_{prob}$ enables juveniles to copy more behavioral attributes from their parents, the most experienced demonstrators in the family, thereby leading juveniles to intensify the search more efficiently around the best solutions. High $1 - VSL_{prob}$, on the other hand, enables juveniles to copy more attributes from their older, more-knowledgeable siblings and then explore different promising areas. Crow $i$ (observer) can learn socially by coping one attribute $j$ or more from crow $k$ (demonstrator) according to Eq. (4).

$$\begin{aligned} x_{i,j}(t) =& x_{k,j}(t-1) \quad where \quad j \in 1, 2, \ldots, d \\ & for \quad vertical\_learning: \quad k = 1 \quad or \quad 2 \\ & for \quad horizontal\_learning: \quad 3 \le k \le i-1 \end{aligned} \tag{4}$$

In which $x_{i,j}(t)$ is the new (i.e., copied) attribute, $j$, for observer juvenile $i$ acquired from demonstrator $k$ in iteration $t$. If $(K = 1 \quad or \quad 2)$, crow $i$ will copy attribute(s) from his parents; otherwise it will copy attribute(s) from his siblings which their positions between $3 \quad and \quad i-1$ in the sorted population.

(1) **Vertical Learning:** If juvenile $i$ chooses to learn vertically, it will determine from which parent to learn according to a predefined probability, $P1_{prob}$. Here, $k$ in Eq. (4) represents either the first parent ($X_1$) or second parent ($X_2$) that juvenile $i$ will choose to imitate. Increasing the $X_1$ selection probability, $P1_{prob}$, intensifies the local search. Although high $P1_{prob}$ intensifies the search around the first-best solution, low $P1_{prob}$ intensifies it around the second-best one. It is worth noting that juveniles tend to copy most of their behavioral attributes from the first parent. As a result, this enables juveniles to intensify the search around the best solution in promising areas, thereby accelerating convergence.

(2) **Horizontal Learning:** When the juvenile tends to learn horizontally, it will observe one of its older, more-experienced siblings according to Eq. (4). Here, $k$ represents a randomly chosen sibling whose fitness is better than that of juvenile $i$ as shown in Eq. (5). In other words, each juvenile can learn from others who are more knowledgeable or more skillful.

$$k = 3 + \lfloor rand \times (i - 3) \rfloor, \tag{5}$$

where $rand$ is a uniformly distributed random number in the range [0, 1] and $i$ is the observer juvenile. $k$ is the index of sibling demonstrator that takes a random value in the range $[3, i - 1]$.

Therefore, from Eqs. (4) and (5), each juvenile can learn or improve its behavior from different demonstrators, not from just one demonstrator through either vertical or horizontal learning, which enhances population diversity. Furthermore, it can be observed that in each generation, a crow could act as a demonstrator for different imitators several times. Besides, the best juvenile in the sorted family, i.e. crow in position 3, will imitate just its parents attributes in addition to altering its attributes individually which mean that it cannot learn horizontally. Moreover, all crows in the family (i.e. parents and juveniles) could serve as demonstrators for other observers within the family except the crow with the worst behavior (i.e. juvenile in the position $n$). In this context, parents will never serve as imitators.

### 3.2.2. Asocial (individual) learning

When juvenile $i$ chooses to learn individually, it can choose to update its behavioral attributes either randomly using Eq. (2) according to a predefined probability, $(1 - SL_{prob}) \times TaE_{prob}$, or based on its prior experience according to probability $((1 - SL_{prob}) - TaE_{prob})$. By this mechanism, juvenile $i$ may retain some attributes of the previous behavior, thereby intensifying the search around its previous behavior to find improved behavior in the same search area. Although parameter $TaE_{prob}$ supports global search, it should take an appropriate value to enable the algorithm to explore the search space well while maintaining convergence speed. High values of $TaE_{prob}$ will let NCCLA search agents move more randomly around the search space, thereby losing their ability to converge to the optimal solution.

## 3.3. Reinforcement phase

After completion of the learning phase, some attributes of the learned behavior of the juvenile and parent behaviors may be rewarded based on reinforcement probability $Rp_{prob}$.

(1) **Juvenile reinforcement:** Reinforcement of attributes of updated juvenile behavior is defined by Eq. (6).

$$X_{i,j}(t) = X_{i,j}(t) \pm RW, \tag{6}$$

where $RW$ represents the reward calculated using Eqs. (7) and (8), wherein each attribute, $j$, of juvenile behavior $X_i$ can be either increased or decreased by $RW$ as iteration $t$ continues. When the newly updated attribute, $X_{i,j}$, exceeds its boundaries, crow $i$ will re-observe behavioral attributes of another more knowledgeable demonstrator. Calculation of $RW$ considers components $\alpha$ and $\beta$. The first, $\alpha$, is the difference between the newly copied attribute and the previous one. It represents the effect of autonomous experience, which helps to intensify the search in promising areas and is calculated, as shown in Eq. (7).

$$\alpha = |X_{i,j}(t) - X_{i,j}(t - 1)| \tag{7}$$

The second component, $\beta$, represents social learning effect developed over time, which helps to explore the search space well. Components of $RW$ enable the algorithm to strike a further balance between exploration and exploitation. Calculation of $\beta$ is shown in Eq. (8).

$$\beta = X_{i,j}(t - 1) \times \exp(-lf \times r \times t \times mean(j)), \tag{8}$$

where $X_{i,j}(t - 1)$ is the previous behavioral attribute $j$ of crow $i$, $r$ is a normally distributed random number in the range [0, 1], $mean(j)$ is the mean of attribute $j$ among the population (representing the behavioral attribute shown by the majority of potential demonstrators), $t$ is current iteration, and $lf$ is a learning factor starting from its minimum value and iterating linearly toward its maximum one. The idea behind such adaptation is to achieve a further balance between exploration and exploitation by varying $lf$ dynamically over the algorithm runtime, and $lf$ is calculated as follows:

$$lf = lf_{min} + ((lf_{max} - lf_{min})/max\_t) \times t, \tag{9}$$

where $lf_{min}$ is the minimum value of $lf$, $lf_{max}$ is the maximum value of $lf$, $t$ is the current iteration, and $max\_t$ is the maximum number of iterations. $RW$ is calculated using Eq. (10):

$$RW = \begin{cases} \beta - \alpha, & \text{if} \quad i < n/2, \\ r1 \times ((r2 \times \beta) - \alpha), & \text{otherwise}, \end{cases} \tag{10}$$

where $i$ represents the current crow and $n$ represents the number of crows in the family. Because juveniles in the first half are more experienced, it would be better for them to intensify the search around a promising area by exploiting historical and experiential data. By contrast, juveniles in the second half are more likely to be inexperienced, so it would be better to let them widely explore different areas in the search space effectively. This can be achieved by multiplying the value with a uniform random numbers, $r1$ and $r2$, as shown in Eq. (10). The balance between exploration and exploitation is emphasized in the reinforcement phase. The high value of reinforcement probability $Rp_{prob}$ led to further balance of exploration and exploitation.

(2) **Parents reinforcement:** In addition to juvenile reinforcement, parents $X_1$ and $X_2$ reward their behaviors because their knowledge and experience increase over time according to their past experiences. This can be performed through the reinforcement phase, wherein $X_1$ and $X_2$ update some of their behavioral attributes according to $Rp_{prob}$. Because $X_1$ is more experienced, it would be better for NCCLA to intensify the search around it, which can be conducted by exploiting historical and experiential data of the first parent. Conversely, the second parent has less experience than the first. Therefore, NCCLA will try to increase the intensity of the search in a promising area around the

second parent by exploiting historical and experiential data in addition to exploring the promising area around the first parent, which can be achieved by multiplying the result with a uniform random number, $r2$. Parent's behaviors are updated according to Eq. (11):

$$X_{i,j}(t) = \begin{cases} X_{i,j}(t-1) - [X_{1,j}(t-1) + \exp\{r1 \times (mean(j) \\ \quad - X_{i,j}(t-1))\}], & i = 1. \\ X_{i,j}(t-1) - [r2 \times [X_{1,j}(t-1) - \exp\{r1 \\ \quad \times (mean(j) - X_{i,j}(t-1))\}]], & i = 2 \end{cases} \tag{11}$$

where $r_1$ is a normally distributed random number, and $r_2$ is a uniformly distributed number in the range [0, 1]. Here, $X_{i,j}(t)$ represents the $j_{th}$ attribute for the best solution among the sorted population, at position 1 or 2, during iteration $t$. In addition, $mean(j)$ is the mean of the $j_{th}$ attribute among the population, representing the attribute shown by the majority of individuals in the population.

### 3.4. Sorting and parent selection

Behaviors of the family population are evaluated and sorted from the best to worst according to the crows' fitness values. Then, the two individuals with the best behaviors, $X_1$ and $X_2$ are selected as parents of the family for the next iteration. The rest of the population represents the juveniles of the family.

### 3.5. Check stopping criteria

Steps 3.2, 3.3 and 3.4 are repeated until one of two stopping criteria is met. In this study, stopping criteria are either the maximum number of iterations is reached, or the optimal solution is found. Finally, NCCLA returns the best behavior as the global best solution, $X^*$.

Fig. 3 exhibits the flowchart of NCCLA steps, and the pseudocode of the NCCLA is presented in Algorithm 1.

**Algorithm 1:** NCCLA Algorithm

```
begin
    t=0;
    Initialize the crow's behaviors within the family $F[n] = [X_1, X_2, ...., X_n]$ ;
    Evaluate each behavior $X_i$ in $F$ using $f(X_i)$;
    Rank all behaviors and determine parent's behavior $X_1$ and $X_2$ at positions 1 and 2, respectively.;
    while termination criterion not met do
        for each juvenile i in the family F do
            /* update behavior $X_i$ through either social or asocial learning */
            for each attribute j in the behavior $X_i$ do
                if $rand \leq SL_{prob}$ then
                    if $rand \leq VSL_{prob}$ then
                        $X_{i,j(t)} = X_{pk,j(t-1)}$; /* pk equals either 1 or 2 according to $P1_{prob}$ */
                    else
                        $X_{i,j(t)} = X_{sk,j(t-1)}$; /* sk is any juvenile with a better behavior than i */
                    end
                else
                    if $rand \leq TaE_{prob}$ then
                        $X_{i,j(t)} = X_L + U(0,1) \times (x_U - x_L)$; /* j is a new random attribute */
                    else
                        $X_{i,j(t)} = X_{i,j(t-1)}$; /* keep the previous attribute j */
                    end
                end
            end
            With $Rp_{prob}$, reinforce certain attributes of the modified $X_i$ using Eq. (6) and Eq. (10);
        end
        With $Rp_{prob}$, parent1 $X_1$ and parent2 $X_2$ are reinforced according to Eq. (11) ;
        Rank all behaviors and select the two best behaviors to be new parents $X_1$ and $X_2$;
        $t = t + 1$;
    end
    Return the global best solution or accepted (adult-like) behavior $X^*$ ;
end
```

## 4. Advantages and differences of NCCLA from other existing meta-heuristics

In this section, NCCLA is compared on conceptual grounds with other metaheuristics including GA, PSO, SLPSO, SLA and CSA to highlights the significant differences of the new algorithm from other previous algorithms. This need arose from the emergence of "new"algorithms that sometimes come as an existing algorithm just in a new appearance [51].

First, NCCLA differs from other metaheuristics regarding the idea of inspiration and search process.

(1) The idea of inspiration of GA is a natural evolution, that of PSO is particles collective behaviors, that of SLPSO and SLA is social learning, and that of CSA is a crow hiding and recovering extra food. NCCLA, in contrast, is inspired by NC-crow learning mechanisms used to improve their behaviors for developing *Pandanus* tools to obtain food.

(2) NCCLA search process also differs from those of other meta-heuristics. In GA, exploration and exploitation are achieved by mutation and crossover operators, respectively. In PSO, they are achieved using a single strategy or formula [52]. In classical PSO, they are conducted through movement among particles and by considering global and personal bests. In NCCLA, exploration and exploitation are achieved in two phases – learning and reinforcement – as previously discussed. Exploitation is achieved through either vertical or horizontal social learning probability and by rewarding the first half of juveniles and parents. Exploration is achieved through individual learning probability by trial and error as well as by rewarding the second half of juveniles.

Furthermore, parents' selection strategies and their purpose in NCCLA are different from those in GA. In GA, different selection strategies, such as roulette wheel and tournament, are used to select parents which mate to reproduce offspring. Whereas, in NCCLA, there is no reproduction for new offspring. Simply, in each iteration, NCCLA sorts the population and chooses the best two solutions to represent the parents of the crow family for the next iteration. Then, juveniles (children) interact with their parents through vertical learning to improve their behaviors by copying some attributes to converge to parents behaviors. Similar to other swarm-based algorithms such as PSO and TLBO, the parents in NCCLA play the role of leaders to guide the search within the search space.

Furthermore, NCCLA learning mechanisms differ from those of other social-learning-based algorithms as illustrated below.

(1) NCCLA vs PSO

- In PSO, behavior is learned from historical best positions. In NCCLA, on the other hand, crows learn behaviors by combining different demonstrator behavioral attributes to acquire distinct skills, ensuring that they learn by observing a more-experienced individual (i.e., demonstrator) instead of learning from historical best positions. Thus, NCCLA can better maintain population diversity.

(2) NCCLA vs SLA

- In SLA observer can acquire some behavioral attributes from demonstrators based on a statistically significant test. On the contrary, no need for this complicated process in NCCLA where the observer can copy attributes from any better demonstrator.

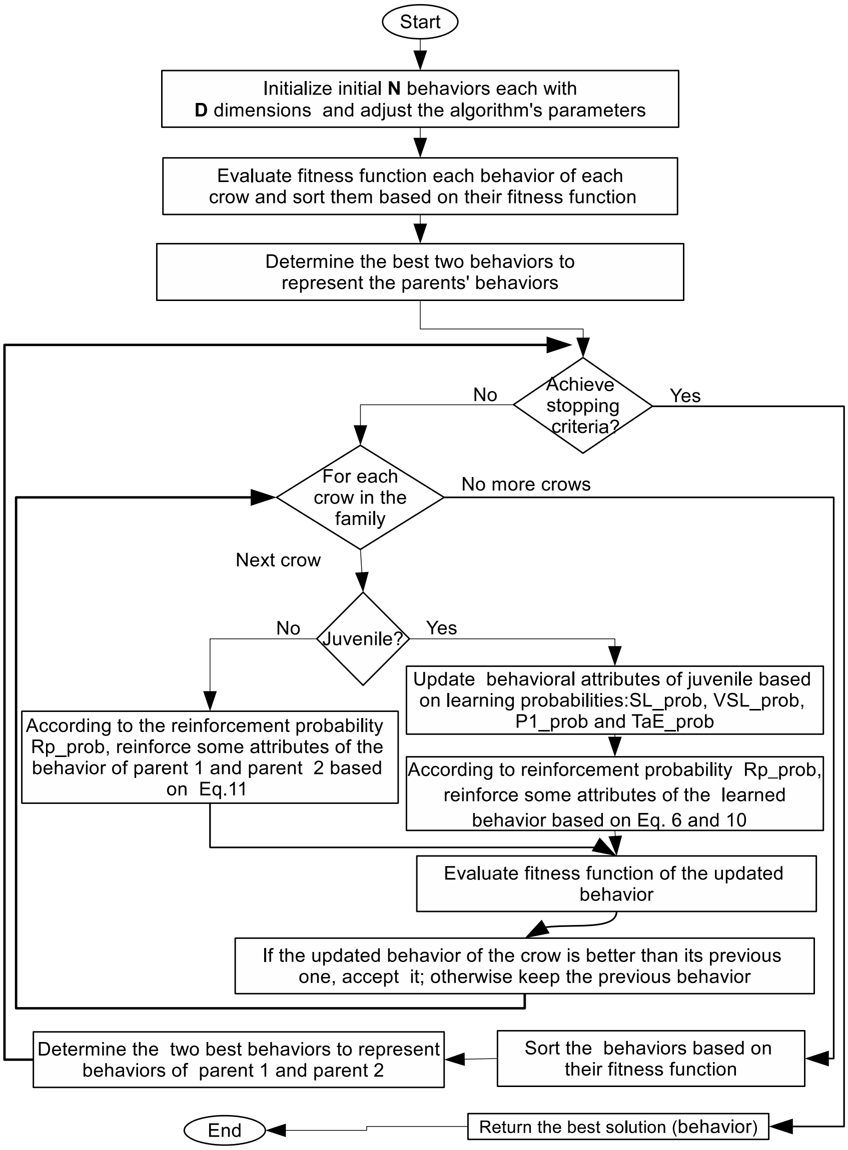
Start
Initialize initial N behaviors each with D dimensions and adjust the algorithm's parameters
Evaluate fitness function each behavior of each crow and sort them based on their fitness function
Determine the best two behaviors to represent the parents' behaviors
Achieve stopping criteria?
No
Yes
For each crow in the family
No more crows
Next crow
Juvenile?
No
Yes
Update behavioral attributes of juvenile based on learning probabilities:SL_prob, VSL_prob, P1_prob and TaE_prob
According to the reinforcement probability Rp_prob, reinforce some attributes of the behavior of parent 1 and parent 2 based on Eq.11
According to reinforcement probability Rp_prob, reinforce some attributes of the learned behavior based on Eq. 6 and 10
Evaluate fitness function of the updated behavior
If the updated behavior of the crow is better than its previous one, accept it; otherwise keep the previous behavior
Determine the two best behaviors to represent behaviors of parent 1 and parent 2
Sort the behaviors based on their fitness function
Return the best solution (behavior)
End

**Fig. 3.** NCCLA flowchart.

- In SLA observers can learn from anyone of demonstrators group without any preference to copy from the best solutions in the population. In NCCLA crows tend to copy more behavioral attributes from its parents, best solutions. Thus, the new solutions are forced to move toward the best solutions and therefore gaining fast convergence.
- Reinforcement is different in SLA and NCCLA as follows. (a) In SLA, reinforcement occurs during the reproduction of new behaviors. Reinforcement in NCCLA, on the contrary, occurs after each juvenile learns behaviors. (b) Unlike in SLA, reinforcement in NCCLA is performed for both juveniles and parents. (c) In SLA, copied dimensions are reinforced through the addition of positive rewards or negative punishments according to significant differences between the copied and previous dimensions. In NCCLA, on the contrary, juveniles can choose the behavioral attributes to be reinforced based on predefined probability, and each attribute has an equal chance to add either positive or negative reinforcements, which is compatible with the stochastic nature of the algorithm which simulates stochastic behavior in animal societies. (d) To calculate reinforcement, SLA uses a stochastic differential ranging from 0 to the value between the demonstrator and the observer. However, NCCLA uses a differential between the demonstrator and observer in addition to the personal experience of each individual. (e)Unlike SLA, juveniles in NCCLA are rewarded depending on the quality of individual behaviors. Less-experienced individuals tend to search more randomly, and more-experienced ones tend to search extensively around promising areas, as illustrated in Eqs. (6) and (10). (f) Similarly, parents in NCCLA are rewarded according to their personal experience, as illustrated in Eq. (11), whereas in SLA, there are no parents in the population.

(3) NCCLA vs SLPSO

- SLPSO is composed of swarm sorting and behavior learning, whereas NCCLA is comprised of learning and reinforcement.

- In SLPSO, the best particle in the current swarm cannot be updated. However, with increasing iterations in NCCLA, parents continue learning by reinforcing some of their attributes according to their knowledge, thereby accelerating convergence.
- The updating procedure in SLPSO takes into account the addition of a behavior correction component to the particle's attribute. Therefore, the behavior-updating mechanism focuses on improving behavior according to social learning and neglects trial-and-error learning concept, which may lead to premature convergence and stuck in local optima. On the contrary, juveniles in NCCLA update their behaviors through the learning phase and the reinforcement phase. In the learning phase, NCCLA either copies several attributes from several demonstrators or updates attributes individually. Then, in the reinforcement phase, NCCLA reinforces (re-update) some attributes based on crows' knowledge and the difference between the current observer's attribute and the mean of this attribute among the population.
- NCCLA takes into account the quality of population behaviors in the calculation of reward values. SLPSO, on the contrary, takes into account the variety of learning motivation between individuals based on the learning probability.

Finally, NCCLA differs from other crow algorithms such as CSA in the following aspects.

(1) CSA is composed of two phases: pursuit (i.e., the movement toward the best solution) and evasion (i.e., random reinitialization of candidate solutions), whereas NCCLA is composed of learning and reinforcement phases, as discussed previously.

(2) CSA searches are controlled by an awareness probability parameter, *AP*, that maintains a balance between exploration and exploitation. In addition, random movement enables CSA to explore the search area. From a practical point of view, CSA prematurely converges and gets stuck at local optima. NCCLA, on the contrary, overcomes such problems by employing different learning and reinforcement mechanisms.

(3) Unlike behavior updating in NCCLA, that in the CSA pursuit phase is based only on personal experience of each crow, and social experience is neglected. Thus, through social-learning probabilities in NCCLA, promising solutions have a high probability of being exploited. Furthermore, behavioral updating in CSA takes into account the entire behavior as a unit. However, NCCLA works at the attribute level, wherein each juvenile can learn behaviors from different demonstrators, not just one, which helps in maintaining population diversity.

The comparison mentioned above between NCCLA and other algorithms shows the significant differences in the proposed algorithm from other known metaheuristic. Also, it underlines that to the best of our knowledge, no metaheuristic algorithm in the literature tries to imitate the effective mechanisms that NC-crows use to learn the behavior of tools development. It thus prompted our attempt to develop a new metaheuristic algorithm inspired by nature, inspired by the NC-crows' biological behavior to address a wider variety of optimization problems.

## 5. Experiment results and discussion

To verify NCCLA performance, we conducted various experiments using benchmark and engineering test functions, and results obtained using NCCLA were compared to those obtained using several other state-of-the-art algorithms

NCCLA was implemented in Java and compiled under Microsoft Windows 10. All the implementations were carried out on a computer equipped with an Intel(R) Core(TM) i5–4210M @ 2.60-GHz central processing unit (CPU).

NCCLA algorithm performance was evaluated using a set of 23 test functions representing classical benchmark functions used by many previous studies [14,16,33–36]. Furthermore, four additional engineering optimization problems were tested to evaluate NCCLA performance in real-world applications. To verify NCCLA scalability, we used test functions with different levels of problem complexity such as unimodal, multi-modal, separable, and non-separable functions in addition to low and high dimensionality. Unimodal functions are typically used to test algorithm convergence ability. However, multi-modal functions are used to validate the ability of an optimization algorithm to escape from local minima and are therefore used to test the exploration ability. It must be noted that non-separable functions are more complex than separable ones owing to the dependency among variables. Problems also become more complex with the increasing number of dimensions, as stated in [27,28,53–58]. Test functions used in the experiments and their characteristics are listed in Table 2.

NCCLA performance was compared with seven of the state-of-the-art algorithms, including a natural-phenomenon-based algorithm (GROM), swarm-based algorithms (GWO, WOA, PFA, and CSA), and social-learning-based algorithms (SLA and SLPSO). All the algorithm source codes and parameter values are available online. All the algorithms were written in MATLAB except for SLA, which was written in C++. To analyze the optimization robustness, NCCLA, and all the other algorithms were executed 30 times for each test function.

### 5.0.1. NCCLA's parameters tuning

To tune and choose the proper combination of NCCLA parameters, the statistical Taguchi experimental design was applied. The Taguchi experimental design is a design method used to identify the effective parameters and their levels with minimum number of experiments [59]. For example, in case of applying classic full factorials experiments, we need ($5^5 = 3125$) experiments for 5 parameters with 5 levels. On the other side, using Taguchi method only 25 experiments (L25) will be needed. The levels of NCCLA's parameters were as follows: (i) $P1_{prob}$ (0.3, 0.5, 0.7, .9, 0.99); (ii) $Rp_{prob}$ (0.1, 0.3, 0.5, 0.7, 0.9); (iii) $TaE_{prob}$ (0.01, 0.2, 0.4, 0.6, 0.8); (iv) $SL_{prob}$(0.3, 0.5, 0.7, .9, 0.99); and (v) $VSL_{prob}$ (0.3, 0.5, 0.7, .9, 0.99). Table 3 shows an appropriate orthogonal array that was produced by Taguchi method using Minitab software. This array offered a balance between the orthogonal index, parameters, and levels. Taguchi classifies parameters into controllable and noise parameters. It is used to determine the optimal levels of important controllable parameters and minimize the effect of noisy parameters.

To conduct the 25 experiments, seven test functions, which represent different levels of complexity, were used with 100 dimensions. These functions are Sphere, Rosenbrock, Ackley, Rastring, Shwefel1.2, Shwefel2.22, and Grainwank. Each function was executed ten times for each combination to increase the reliability of the experiments. The average results of all different observation values were evaluated by transforming the results into signal-to-noise ratio (S/N ratio) in the Taguchi experimental design. The S/N ratios are used to identify the control factor settings that minimize the variability caused by the noise factors. From Table 4, it is clear that the effect of parameters on the S/N ratio had the following order: $Rp_{prob}$ (Delta 110.2594, Rank = 1); followed by $SL_{prob}$ (Delta 46.8329, Rank = 2); then followed by $TaE_{prob}$, $P1_{prob}$, and $VSL_{prob}$. In addition, Fig. 4 emphasizes the effects of the parameters in terms of the S/N ratio

**Table 2**
Unimodal, Multimodal and Fixed-dimension test functions employed in the study.

| $f(x)$ Type | $f(x)$ No. | $f(x)$ Name | $f(x)$ Formula | Range | Opt. Sol. | Dim |
|---|---|---|---|---|---|---|
| Unimodal | **F1** | sphere | $f(x) = \Sigma_{i=1}^{D} x^2$ | −100,100 | 0 | 30 |
| | **F2** | Schwefel2.22 | $f(x) = \Sigma_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$ | −10,10 | 0 | 30 |
| | **F3** | Schwefel1.2 | $f(x) = \Sigma_{i=1}^{D} (\Sigma_{j=1}^{i} x_j)^2$ | −100,100 | 0 | 30 |
| | **F4** | Schwefel2.21 | $f(x) = max(\lvert x_i \rvert, 1 \leq i \geq D)$ | −100,100 | 0 | 30 |
| | **F5** | Rosenbrock | $f(x) = \Sigma_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ | −30, 30 | 0 | 30 |
| | **F6** | step | $f(x) = \Sigma_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$ | −100,100 | 0 | 30 |
| | **F7** | Quartic | $f(x) = \Sigma_{i=1}^{D} i x_i^4 + random[0, 1]$ | −1.28,1.28 | 0 | 30 |
| Multimodal | **F8** | Schwefel | $f(x) = \Sigma_{i=1}^{D} - x_i \sin(\sqrt{x_i})$ | −500,500 | $-418.9829 \times D$ | 30 |
| | **F9** | Rastring | $f(x) = \Sigma_{i=1}^{D} [x_i^2 - 10 \cos(2\pi x_i) + 10D]$ | −5.12,5.12 | 0 | 30 |
| | **F10** | Ackely | $f(x) = -20 \exp(-0.2\sqrt{1/D \Sigma_{i=1}^{D} x^2})$<br>$- \exp(1/D \Sigma_{i=1}^{D} \cos 2\pi x_i) + 20 + \varrho$ | −32 32 | 0 | 30 |
| | **F11** | Griewank | $f(x) = 1/4000 \Sigma_{i=1}^{D} (x_i)^2 - \prod_{i=1}^{D} \cos(x_i/\sqrt{i}) + 1$ | −600, 600 | 0 | 30 |
| | **F12** | Penalized1 | $f(x) = \pi/D\{10 \sin^2(\pi y_1)$<br>$+ \Sigma_{i=1}^{D-1} (y_i - 1)^2 [1 + \sin^2(\pi y_{i+1}) + (y_D - 1)^2]\}$<br>$+ \Sigma_{i=1}^{D} u(x_i, a, k, m)$ | −50,50 | 0 | 30 |
| | **F13** | Penalized2 | $f(x) = 1/10\{\sin^2(3\pi x_1)$<br>$+ \Sigma_{i=1}^{D-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)^2 [1 + \sin^2(2\pi x_D])\}$<br>$+ \Sigma_{i=1}^{D} u(x_i, a, k, m)$ | −50,50 | 0 | 30 |
| Fixed-dimension Multimodal | **F14** | De_Jong5 | $f(x) = (\frac{1}{500} + \Sigma_{j=1}^{25} (\frac{1}{j+} \Sigma_{i=1}^{2} (x_i - a_{ij})))^- 1$ | −65,65 | 1 | 2 |
| | **F15** | kowalik | $f(x) = \Sigma_{i=1}^{11} [a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x + 3 + x_4}]^2$ | −5,5 | 0.00030 | 4 |
| | **F16** | Camel_Six Hump | $f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3} x_1^6 + x_1 x_2 + 4x_2^2 + 4x_2^4$ | −5,5 | −1.0316 | 2 |
| | **F17** | Branin | $f(x) = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2 + 10(1 - \frac{1}{8\pi}) \cos x_1 + 10$ | −5,5 | 0.398 | 2 |
| | **F18** | Goldstein | $f(x) = (1 + (x_1 + x_2 + 1)^2 \times$<br>$(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)) \times$<br>$(30 + (2x_1 - 3x_2)^2$<br>$\times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2))$ | −2,2 | 3 | 2 |
| | **F19** | Hartman3 | $f(x) = \Sigma_{i=1}^{4} c_i \exp(-\Sigma_{i=1}^{3} a_{ij(xj - p_{ij})^2})$ | 1,3 | −3.86 | 3 |
| | **F20** | Hartman6 | $f(x) = \Sigma_{i=1}^{4} c_i \exp(-\Sigma_{i=1}^{6} a_{ij(xj - p_{ij})^2})$ | 0,1 | −3.32 | 6 |
| | **F21** | Shekel5 | $f(x) = -\Sigma_{i=1}^{5} [(X - a_i)(X - a_i)^T + C_i]^- 1$ | 0,10 | −10.1531 | 4 |
| | **F22** | Shekel7 | $f(x) = -\Sigma_{i=1}^{7} [(X - a_i)(X - a_i)^T + C_i]^- 1$ | 0,10 | −-10.4028 | 4 |
| | **F23** | Shekel10 | $f(x) = -\Sigma_{i=1}^{10} [(X - a_i)(X - a_i)^T + C_i]^- 1$ | 0,10 | −10.5363 | 4 |

**Table 3**
Parameter values obtained by the Taguchi method.

| Experiment | P1 | Rp | TaE | SL | VSL |
|---|---|---|---|---|---|
| 1 | 0.3 | 0.1 | 0.01 | 0.3 | 0.3 |
| 2 | 0.3 | 0.2 | 0.2 | 0.5 | 0.5 |
| 3 | 0.3 | 0.4 | 0.4 | 0.7 | 0.7 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 25 | 0.99 | 0.8 | 0.6 | 0.7 | 0.5 |

**Table 4**
Response table for Signal-to-Noise ratios.

| Level | P1 | Rp | TaE | SL | VSL |
|---|---|---|---|---|---|
| 1 | −6.8275 | −54.8397 | 24.4349 | −17.2511 | 1.4942 |
| 2 | 9.3881 | −21.1303 | −0.3735 | −11.3638 | −6.0083 |
| 3 | −10.6570 | 3.3867 | −18.7726 | −8.6649 | 2.041 |
| 4 | 9.8217 | 23.3044 | 3.6947 | 13.8389 | 3.2426 |
| 5 | 4.4154 | 55.4197 | −2.8428 | 29.5817 | 5.3713 |
| Delta | 20.4787 | 110.2594 | 43.2075 | 46.8329 | 11.3796 |
| Rank | 4 | 1 | 3 | 2 | 5 |

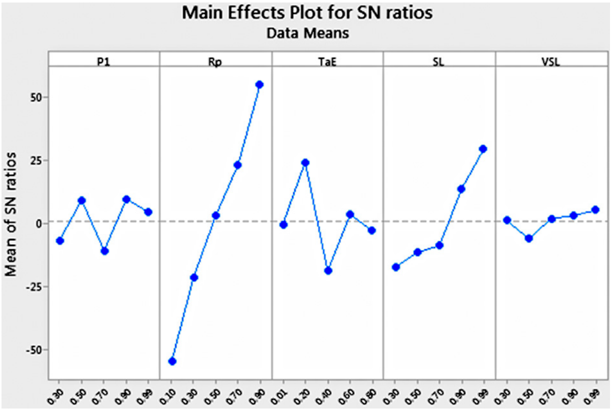

**Fig. 4.** Main effects plot for S/N ratio (higher value is better).

and indicates that $Rp_{prob}$ has the largest effect on the signal-to-noise ratio whereas $VSL_{prob}$ has the smallest effect or no effect on the signal-to-noise ratio. It is clear from the figure that high level values are more suitable for $P1_{prob}$, $Rp_{prob}$, $SL_{prob}$ and $VSL_{prob}$, whereas low level value is suitable for $TaE_{prob}$. However, it is not possible to define specific values for the algorithmic parameters that will be the proper ones for all problems. In addition, changing values of parameters with low sensitivity has little effect on the NCCLA's performance. Accordingly, the better performance and robustness of NCCLA can be obtained when its parameters were adjusted to the values suggested by the Taguchi method with very little change. As shown in Table 5, $P1_{prob}$, $Rp_{prob}$, $TaE_{prob}$, $SL_{prob}$ and $VSL_{prob}$ were set to the values .95, .90, 0.3, 0.99 and 0.99 respectively.

All the NCCLA parameters were fixed during all the benchmark-function simulations, which were conducted using a population of 80 over 500 iterations, representing 40,000 function evaluations ($NFE_s$). The NCCLA algorithm stopping criterion was met when the algorithm reached either the maximum $NFE_s$ or found the best solution.

### 5.1. Performance of NCCLA with classical benchmark functions

**Table 5**
Parameter settings for NCCLA.

| Control parameter | Value |
|---|---|
| No. of crows within a family | 80 |
| Maximum No. of Iterations | 500 |
| Reinforcement Probability ($Rp_{prob}$) | 0.9 |
| Social Learning Probability ($SL_{prob}$) | 0.99 |
| Vertical Learning Probability ($VSL_{prob}$) | 0.99 |
| First Parent Selection Probability ($P1_{prob}$) | 0.95 |
| Trial and Error Probability ($TaE_{prob}$) | 0.3 |
| $lf_{min}$ | 0.0005 |
| $lf_{max}$ | 0.02 |

NCCLA performance obtained over 30 independent runs was tested and evaluated using standard benchmark functions. It was subsequently compared with corresponding performance obtained using seven other state-of-the-art algorithms, including GROM, GWO, WOA, CSA, PFA, SL-PSO, and SLA for the same benchmark functions. Each function was executed 30 times to obtain results recorded in Tables 6–8. Benchmark functions used in this experiment were unimodal, multi-modal, and fixed-dimension multi-modal, as shown in Table 2. The number of scalable-function dimensions, including unimodal and multi-modal was set to 30. It should be noted that unimodal functions are employed to test the ability of the algorithms to exploit the search for good solutions in a promising area of the search space. Whereas, multi-modal functions, either fixed or non-fixed (i.e., scalable), highlight the ability of an algorithm to explore the search space.

#### 5.1.1. Exploitation analysis

Unimodal functions were employed to study the ability of the NCCLA algorithm to exploit a search within a promising search space. Thus, NCCLA was tested among all unimodal functions (F1–F7) listed in Table 2. Results achieved using NCCLA were compared to those obtained using the other state-of-the-art algorithms for the maximum number of iterations, as shown in Table 6. It is noted that NCCLA algorithm was superior for solving unimodal functions. NCCLA effectively reached global optima for the best and average solutions achieved with functions (F1–F6), as shown in Table 6. Moreover, it should be noted that SLA was the second-best algorithm because it reached global optima with functions F2, F4, F5, F6, and F7 in terms of the best and average solutions. However, SLPSO only reached global optima with function F6. NCCLA reached global optima after 11.2, 4.8, 58.2, and 8.333333333 iterations on average with functions F2, F4, F5, and F6, respectively. By contrast, SLA reached global optima after 73.33333333, 22.86666667, 181.4, and 316.4666667 iterations on average for all the runs, respectively.

Although non-separable functions such as Rosenbrock (F5) are more complex, NCCLA, and SLA both achieved the same performance with F5. Because F5 is nonlinear, it is difficult to converge to the minimum with this function. Thus, algorithms must strike a good balance between exploration and exploitation to overcome such difficulty and solve the function. Results listed in Table 6 highlight the ability of NCCLA to provide such balance and solve F5 effectively. NCCLA can reach optima of F5 in fewer than 60 iterations on average for all the runs because exploitation components through $SL_{prob}$, $VSL_{prob}$, and $P1_{prob}$ enabled NCCLA to intensify the search around the best solutions in promising areas. In addition to the juvenile and first-parent rewards that was calculated according to Eq. (10) and Eq. (11), respectively.

Results listed in Table 6, show that NCCLA is superior in solving unimodal functions. As mentioned previously, unimodal functions are suitable for benchmarking exploitation. Thus, these results exhibit the superior exploitation performance of NCCLA.

#### 5.1.2. Exploration analysis

Unlike unimodal functions, multi-modal ones show exploration ability. Thus, multi-modal functions (F8–F13) listed in Table 2 were used to study NCCLA ability to explore the search space

**Table 6**
Results of unimodal functions with 30 dimensions and 500 iterations.

| Alg. Fun. | | F1 | F2 | F3 | F4 | F5 | F6 | F7 |
|---|---|---|---|---|---|---|---|---|
| NCCLA | Best | 0 | 0 | 0 | 0 | 0 | 0 | 2.68E−06 |
| | Avg | 0 | 0 | 0 | 0 | 0 | 0 | 3.76E−04 |
| | S.D. | 0 | 0 | 0 | 0 | 0 | 0 | 6.25E−04 |
| GROM | best | 1.60E−69 | 1.58696E−38 | 1.77E−68 | 1.25E−22 | 17.79997 | 4.98E−08 | 7.02E−05 |
| | Avg | 1.21E−67 | 1.78733E−37 | 3.4E−63 | 2.15E−22 | 19.46449 | 8.51E−07 | 1.97E−04 |
| | S.D. | 1.84E−67 | 1.91481E−37 | 1.46E−62 | 4.77E−23 | 1.225087 | 1.81E−06 | 8.5E−05 |
| GWO | best | 1.46E−40 | 6.922E−24 | 5.02E−14 | 7.16E−11 | 25.21574 | 2.5E−05 | 0.000275 |
| | Avg | 3.58E−38 | 1.07782E−22 | 3.38E−11 | 5.76E−10 | 26.42891 | 0.256525 | 0.000707 |
| | S.D. | 6.08E−38 | 1.38198E−22 | 6.11E−11 | 3.88E−10 | 0.63204 | 0.25359 | 0.000301 |
| SLPSO | best | 8.89E−16 | 1.08E−08 | 2.20E+01 | 2.05E−04 | 2.50E+01 | 0 | 6.38E−03 |
| | avg | 4.80E−15 | 2.52E−08 | 1.27E+02 | 4.78E−04 | 3.96E+01 | 0 | 1.48E−02 |
| | S.D. | 3.62E−15 | 1.10049E−08 | 87.46587 | 0.000198 | 27.83026 | 0 | 0.004444 |
| WOA | best | 8.5E−107 | 8.39818E−63 | 4639.391 | 5.38E−05 | 26.47709 | 0.002962 | 9.89E−05 |
| | Avg | 2.65E−93 | 2.50541E−56 | 23235.36 | 32.14791 | 27.04334 | 0.010499 | 0.001174 |
| | S.D. | 8.6E−93 | 7.63502E−56 | 9198.434 | 32.02524 | 0.405875 | 0.005503 | 0.001367 |
| SLA | best | 0.001383 | 0 | 14118 | 0 | 0 | 0 | 0.485763 |
| | Avg | 0.006361 | 0 | 22071.34 | 0 | 0 | 0 | 0.681824 |
| | S.D. | 0.003879 | 0 | 4173.511 | 0 | 0 | 0 | 0.588307 |
| PFA | best | 9.47E−18 | 4.03067E−11 | 68.53747 | 0.002892 | 23.22998 | 2.4E−06 | 0.000846 |
| | Avg | 7.14E−15 | 6.12057E−10 | 767.8065 | 0.12493 | 24.17964 | 6.46E−06 | 0.004511 |
| | S.D. | 1.22E−14 | 1.096E−09 | 622.2952 | 0.178346 | 0.539809 | 3.15E−06 | 0.00335 |
| CSA | best | 0.328 | 0.603931314 | 22540.602 | 1.558 | 3799282 | 0.277 | 0.009 |
| | Avg | 0.910 | 1.571553678 | 53460.831 | 2.937 | 25103353 | 0.780 | 0.021 |
| | S.D. | 0.409 | 0.516243375 | 16102.663 | 0.760 | 15488639 | 0.374 | 0.009 |

**Table 7**
Results of multi-modal functions with 30 dimensions and 500 iterations.

| Alg.\Fun. | | F8 | F9 | F10 | F11 | F12 | F13 |
|---|---|---|---|---|---|---|---|
| NCCLA | Best | −1.26E+04 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 1.57E−32 | 1.35E−32 |
| | Avg | −1.26E+04 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 1.57E−32 | 1.35E−32 |
| | S.D. | 0.010461176 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 5.47E−48 | 5.47E−48 |
| GROM | best | −9.09E+03 | 0.00E+00 | 8.88E−16 | 0.00E+00 | 3.98E−10 | 1.57E−07 |
| | Avg | −5.52E+03 | 0.00E+00 | 8.88E−16 | 2.63E−03 | 2.42E−02 | 1.41E−01 |
| | S.D. | 1.04E+03 | 0.00E+00 | 9.86E−32 | 4.76E−03 | 6.38E−02 | 2.76E−01 |
| GWO | best | −7.68E+03 | 0.00E+00 | 2.22E−14 | 0.00E+00 | 3.28E−06 | 3.87E−05 |
| | Avg | −6.25E+03 | 1.05E+00 | 3.19E−14 | 3.09E−03 | 2.05E−02 | 2.21E−01 |
| | S.D. | 9.24E+02 | 3.39E+00 | 4.49E−15 | 7.18E−03 | 1.21E−02 | 1.40E−01 |
| SLPSO | best | −1.16E+04 | 8.95E+00 | 8.31E−09 | 1.89E−15 | 2.57E−17 | 7.42E−16 |
| | avg | −1.10E+04 | 4.93E+01 | 2.07E−08 | 5.75E−04 | 1.59E−16 | 3.66E−04 |
| | S.D. | 3.26E+02 | 3.06E+01 | 7.97E−09 | 2.21E−03 | 1.20E−16 | 2.01E−03 |
| WOA | best | −1.26E+04 | 0.00E+00 | 8.88E−16 | 0.00E+00 | 3.10E−04 | 5.35E−03 |
| | Avg | −1.19E+04 | 7.58E−15 | 4.20E−15 | 5.76E−03 | 4.41E−03 | 5.97E−02 |
| | S.D. | 1.13E+03 | 2.47E−14 | 2.63E−15 | 2.33E−02 | 1.26E−02 | 6.22E−02 |
| SLA | best | −1.26E+04 | 2.00E+01 | 7.71E−03 | 5.82E−03 | 4.83E−03 | 1.46E−01 |
| | Avg | −1.26E+04 | 3.11E+01 | 1.35E−02 | 4.75E−02 | 8.80E−01 | 2.77E+00 |
| | S.D. | 1.38E+00 | 5.94E+00 | 4.28E−03 | 6.68E−02 | 1.51E+00 | 2.74E+00 |
| PFA | best | −1.03E+04 | 2.84E−13 | 1.27E−09 | 0.00E+00 | 5.37E−07 | 1.35E−05 |
| | Avg | −8.01E+03 | 3.58E+01 | 1.58E−01 | 1.36E−02 | 4.54E−01 | 2.31E−02 |
| | S.D. | 1.14E+03 | 1.72E+01 | 6.00E−01 | 1.82E−02 | 1.14E+00 | 3.25E−02 |
| CSA | best | −8.98E+03 | 5.812187627 | 1.592209279 | 0.551 | 169.831 | 16130.188 |
| | Avg | −7.20E+03 | 16.01575782 | 2.734991305 | 0.719865114 | 6946.360 | 555412.332 |
| | S.D. | 7.31E+02 | 5.879843778 | 0.459647157 | 0.107465121 | 11416.947 | 583418.412 |

well. Results shown in Table 7 compare all the algorithms' results achieved for the same number of iterations. As shown in Table 7, NCCLA achieved the best results among all the algorithms with all the test functions, reaching the global optimal solutions for functions F8–F11 and the best solutions with functions F12 and F13. NCCLA was able to get the optimal solutions with functions F9, F10, and F11 in terms of best and average solutions.

In addition, the superiority of NCCLA, followed by that of SLA, in solving the Schewefel function (F8) should be noted. This note is due to Schewefel function characteristics. Although Schewefel has several local minima, its global minimum is at the boundary of the search space, unlike most other test functions. In the same context, only the best and average NCCLA and GROM reached global optima with the Rastrigin function (F9). However, NCCLA

**Table 8**
Results of fixed multi-modal Functions with 500 iterations.

| Alg. Fun. | | F14 | F15 | F16 | F17 | F18 | F19 | F20 | F21 | F22 | F23 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NCCLA | Best | 9.98E−01 | 3.07E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 9.98E−01 | 4.83E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.26E+00 | −8.79E+00 | −1.02E+01 | −1.02E+01 |
| | S.D. | 5.30E−11 | 2.69E−04 | 3.79E−08 | 1.69E−04 | 2.75E−14 | 1.44E−06 | 5.94E−02 | 2.25E+00 | 9.54E−01 | 1.35E+00 |
| GROM | best | 9.98E−01 | 3.07E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 1.06E+00 | 3.07E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.30E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | S.D. | 3.56E−01 | 4.33E−19 | 5.83E−08 | 5.87E−06 | 4.44E−16 | 2.66E−15 | 4.76E−02 | 1.78E−15 | 0.00E+00 | 8.88E−15 |
| GWO | best | 9.98E−01 | 3.07E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 3.16E+00 | 2.44E−03 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.26E+00 | −9.14E+00 | −1.02E+01 | −1.05E+01 |
| | S.D. | 3.12E+00 | 5.98E−03 | 7.69E−09 | 1.68E−04 | 4.32E−06 | 2.37E−03 | 6.59E−02 | 2.02E+00 | 9.54E−01 | 4.68E−04 |
| SLPSO | best | 6.48E+00 | 4.78E−04 | −9.47E−01 | 4.89E−01 | 4.93E+00 | −9.01E−01 | −6.94E+02 | −4.54E+00 | −9.94E+00 | −1.04E+01 |
| | avg | 9.38E+01 | 7.34E−01 | 2.67E+00 | 1.40E+01 | 1.64E+02 | −2.69E−01 | −4.70E+02 | −1.54E+00 | −2.17E+00 | −2.38E+00 |
| | S.D. | 1.34E+02 | 1.23E+00 | 1.04E+01 | 2.01E+01 | 2.73E+02 | 2.93E−01 | 1.48E+02 | 1.11E+00 | 2.06E+00 | 1.82E+00 |
| WOA | best | 9.98E−01 | 3.08E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 1.26E+00 | 7.99E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.27E+00 | −9.81E+00 | −8.75E+00 | −8.68E+00 |
| | S.D. | 6.35E−01 | 6.36E−04 | 2.34E−11 | 5.79E−07 | 4.09E−06 | 2.17E−03 | 7.26E−02 | 1.30E+00 | 2.84E+00 | 2.95E+00 |
| SLA | best | 9.98E−01 | 4.87E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.00E−01 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 9.98E−01 | 5.68E−03 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.00E−01 | −3.31E+00 | −5.64E+00 | −4.81E+00 | −4.37E+00 |
| | S.D. | 6.78E−16 | 8.49E−03 | 0.00E+00 | 1.69E−16 | 0.00E+00 | 1.13E−16 | 3.02E−02 | 3.46E+00 | 3.33E+00 | 2.04E+00 |
| PFA | best | 9.98E−01 | 3.07E−04 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 1.03E+00 | 1.31E−03 | −1.03E+00 | 3.98E−01 | 3.00E+00 | −3.86E+00 | −3.25E+00 | −8.96E+00 | −7.56E+00 | −8.60E+00 |
| | S.D. | 1.81E−01 | 3.63E−03 | 2.88E−13 | 2.95E−11 | 2.43E−11 | 4.32E−11 | 5.83E−02 | 2.19E+00 | 3.37E+00 | 3.09E+00 |
| CSA | best | 9.53E−01 | 4.97E−03 | −1.03E+00 | 4.08E−01 | 3.06E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | Avg | 1.03E+00 | 1.73E−02 | −9.57E−01 | 5.05E−01 | 5.65E+00 | −3.86E+00 | −3.32E+00 | −1.02E+01 | −1.04E+01 | −1.05E+01 |
| | S.D. | 2.60E−01 | 9.18E−03 | 5.36E−02 | 9.12E−02 | 2.80E+00 | 2.66E−15 | 8.09E−13 | 1.45E−13 | 6.26E−14 | 3.05E−14 |

reached global optima with this function in around ten iterations on average for all the runs. By contrast, GROM reached global optima with F9 after 53.5 iterations on average for all the runs.

In addition to multi-modal of the functions, a non-separable feature increases the complexity that optimization algorithms may face, such as the complexity encountered when trying to solve the Ackley function (F10). Despite such difficulty, only NCCLA addressed this function perfectly, reaching optima in 15 iterations on average for all the runs.

Overall, NCCLA outperformed all the other algorithms for solving functions F8–F13, achieving average global optima with four of six multi-modal functions. Penalized functions (F12 and F13) are complex and challenging to solve because they are composed of different sinusoidal functions. Despite such a challenge, NCCLA outperformed all the other algorithms, reaching the best-known solutions that all the other algorithms failed to reach. Moreover, low standard deviations achieved with functions F8, F12, and F13, and zero deviation obtained with functions F9, F10, and F11, demonstrate NCCLA robustness owing to exploration achieved through $TaE_{prob}$, thereby enabling NCCLA to explore the search space effectively, and to autonomous experience employed in the reinforcement phase. This effect was also produced by the juvenile and second-parent reward calculated according to Eq. (11), thereby helping to explore the search space well.

Fixed-dimension multi-modal functions (F14–F23) shown in Table 2 have several local optima. Therefore, the exploration ability of the optimization algorithm also should be evaluated by employing such test functions. Results listed in Table 8 show that for functions F14–F23, all the algorithms reached approximately the same best result. NCCLA was able to reach the global optimal solution in terms of best and average with all the fixed-dimension functions, F14–F23 in terms of the best solution, and similar to the global optima with five of ten functions (i.e., F14, F16, F17, F18, and F19) in terms of average solution. Although some of the other algorithms, such as GROM and CSA, achieved better average solutions with Shekel functions (i.e., F21–F23), and with functions (i.e., F19–F20) respectively, NCCLA solutions achieved with these functions were still reasonable. They were comparable with the corresponding solutions produced by the other algorithms. Moreover, NCCLA achieved the best standard deviations among all the algorithms with some of the test functions, i.e., F14 and F16.

Overall, results listed in Tables 7 and 8 show that NCCLA achieved excellent exploration performance and demonstrated good ability to find the best result among different local optima. Although NCCLA became slightly stuck at local optima, its overall performance for exploring optima within search spaces of scalable multi-modal functions was superior to those of all the other algorithms, as shown in Table 7. Moreover, NCCLA provided good results comparable to those obtained by the other algorithms with non-scalable multi-modal functions, as shown in Table 8.

### 5.1.3. NCCLA Convergence behavior

NCCLA convergence behavior was evaluated, and the best, worst, and average convergence behavior achieved for an independent run are plotted in the left column of Fig. 5. Sphere (F1), Rosenbrock (F5), Rastrigin (F9), Ackley (F10), and Goldstein (F18) functions were used to evaluate NCCLA convergence behavior. All these functions have 30 dimensions except the Goldstein function (F18), which is a non-scalable two-dimensional function. It can be noted that NCCLA quickly converged. Furthermore, NCCLA has a high capability to escape from local optima, as shown with function F10, regained its diversity, and converged to global optima, as demonstrated with the Rosenbrock function(F5).

Furthermore, the convergence behavior achieved by each algorithm during the first 100 iterations are plotted in the right column of Fig. 5 to fairly compare the convergence behavior of all the algorithms. It is clear from the figure that NCCLA converged faster than all the other algorithms with all the test functions. Furthermore, NCCLA converged to the well-known solution faster than any other algorithm (i.e., in fewer than 40 iterations) with all depicted functions. Fig. 5 also highlights the superior NCCLA convergence speed.

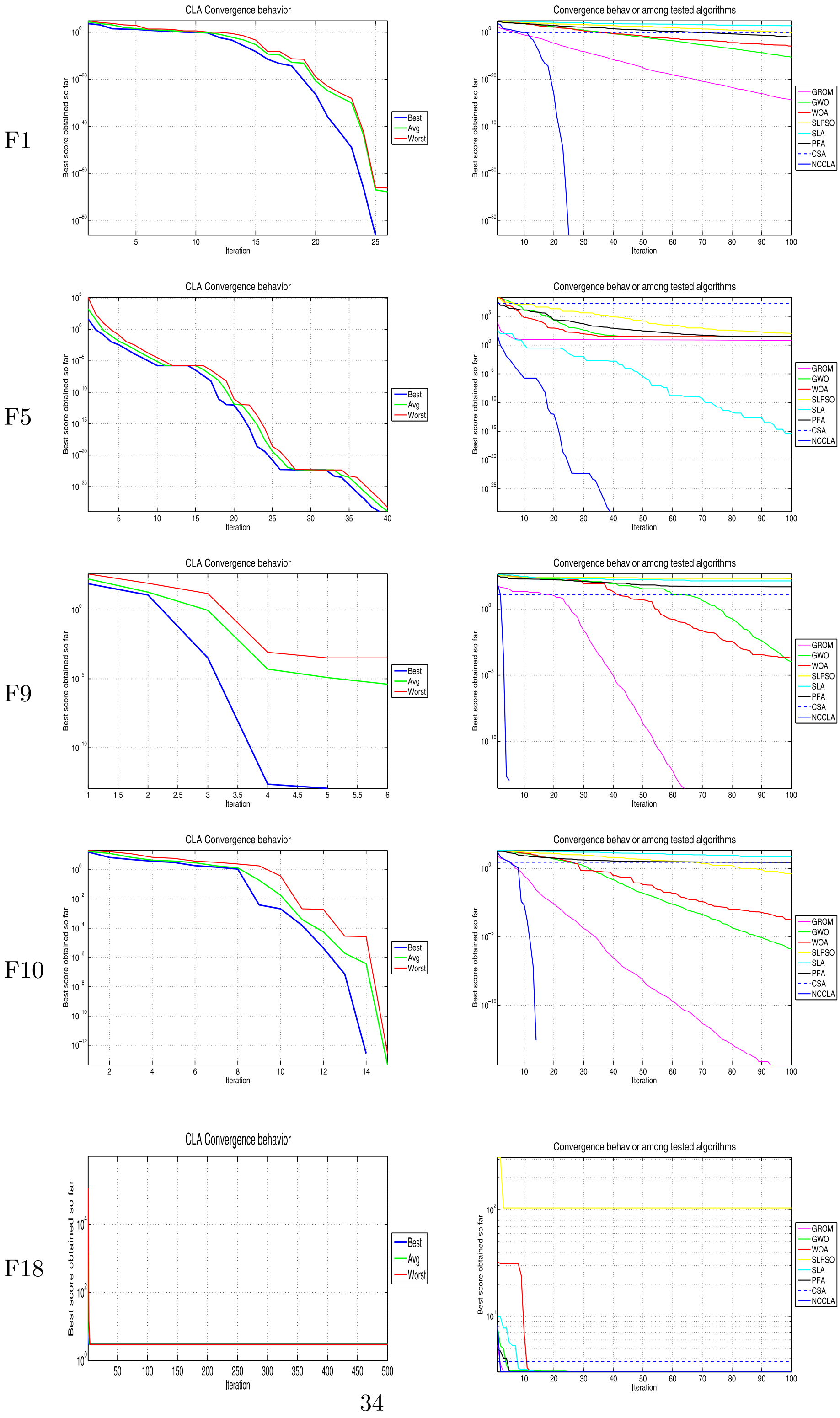

**Fig. 5.** Convergence behavior of NCCLA and other tested algorithms.

### *5.1.4. Discussion of NCCLA performance*

Experimental results revealed interesting NCCLA characteristics. High exploitation and convergence abilities of NCCLA were due to NC behavior-updating mechanisms. Clearly, NCCLA achieved performance comparable or superior to the performance of the other state-of-the-art metaheuristics in terms of best and average values. In terms of the best value, NCCLA reached the optimal solution of the unimodal and multi-modal (fixed and non-fixed) functions with success rates, (Number of optimum achievements/ Number of problems), of 86% and 88% respectively. In general, NCCLA was capable of reaching the optimal solution for 20 functions out of 23; this refers to an average success rate of 87%. For the remaining three functions (F7, F12, and F13), NCCLA found near-optimum solutions that were better than all other compared methods. The second success rate (57%) was achieved by SLA, followed by GROM, GWO, and WOA with a success rate

**Table 9**
Average rankings of Friedman test and $\rho$-values obtained with Holm's post hoc test using the NCCLA as a control algorithm over the 23 tested functions.

| DataSet/Algorithm | Ranking | p | z | Holm |
|---|---|---|---|---|
| NCCLA | 2.5870 | The control method | | |
| GROM | 3.1522 | 0.4339 | 0.7825 | 0.0500 |
| WOA | 4.3696 | 0.0136 | 2.4679 | 0.0250 |
| SLA | 4.5652 | 0.0062 | 2.7388 | 0.0167 |
| GWO | 4.7391 | 0.0029 | 2.9795 | 0.0125 |
| PFA | 4.9130 | 0.0013 | 3.2203 | 0.0100 |
| SLPSO | 5.7391 | 0.0000 | 4.3640 | 0.0083 |
| CSA | 5.9348 | 0.0000 | 4.6349 | 0.0071 |

of 52%. Lastly, CSA with a success rate of 26%. In terms of the average results over 30 runs, NCCLA achieved an average success rate (Number of average optimum achievements/ Number of problems), of 74% followed by GROM and SLA with an average success rate of 43%. In detail, it reached the optimum solution of the unimodal functions with success rates of 86%, which was the better achievement rate over all other methods. For multimodal functions, its achievement was identical to GROM at 56%, followed by SLA with 38%. The standard deviations of NCCLA algorithm were the best for some test functions and competitive to other methods for some other test functions. Moreover, the average of 30 experiments for each test function was identical or very close to the optimal solution. Lower standard deviation values also explain this feature of NCCLA. Notably, it was the most efficient or second-best algorithm in the majority of test functions.

This performance was gained due to the characteristics of NCCLA in its two main phases. Firstly, in the learning phase, different parameters contributed to increasing the exploitation component. The high value of $SL_{prob}$, $VSL_{prob}$ and $P1_{prob}$ led to intensify the search around best solutions. On the other hand, exploration parameters contributed to increasing the exploration component. The probability of horizontal learning $1 - VSL_{prob}$ and the past experience probability $((1 - SL_{prob}) - TaE_{prob})$ led to maintain the diversity of the population. In addition, the low value of the trial and error probability $TaE_{prob}$, which led to exploring the search space well. Moreover, the balance between exploration and exploitation was emphasized in the reinforcement phase of juveniles and parents, as mentioned previously in Section 3.3. The high value of reinforcement probability $Rp_{prob}$ led to further balance of exploration and exploitation. By combining fine parameter tuning and different updating procedures discussed previously in the two phases, NCCLA shows local optima avoidance and high convergence speed simultaneously during the course of the run.

### 5.1.5. Statistical test

To draw fair, meaningful conclusions about algorithm performance, we conducted rigorous statistical analysis on average algorithm solutions achieved using 23 functions. Because results were not normally distributed, as proven by a normality test wherein $\rho = 0.0000(< 0.05)$, a nonparametric test was conducted. Because six algorithms were applied, a multiple comparison test should be used [60]. A Friedman Test examines the null hypothesis that there were no significant differences among all tested algorithms $\chi^2(df = 7) = 35.72463768115928$, $\rho < 0.05$. Results were distributed according to a chi-squared distribution with six degrees of freedom. For the Friedman Test, $\rho = 0.000$, indicating that the results were statistically significant. Therefore, algorithm performance showed statistically significant differences. As noted in column 1 Table 9, the NCCLA algorithm achieved the lowest rank among all the algorithms.

In addition, Holm's post-hoc test was conducted to evaluate the statistical significance of NCCLA performance. Column 5 of Table 9 shows $\rho$ values obtained by applying Holm's post-hoc test with NCCLA as the control algorithm. Results listed in Table 9 show that NCCLA was significantly better than the other algorithms in solving the 23 test functions. Wherein Holm's procedure rejected those hypotheses with $p$ value $\leq 0.05$. The statistical test results support conclusions previously drawn from experimental results. Therefore, NCCLA significantly outperformed all the other algorithms.

### 5.1.6. Large-scale optimization functions

It becomes more challenging for optimization algorithms to solve high-dimensional functions owing to the increased complexity. To evaluate NCCLA performance for solving such functions, unimodal and scalable multi-modal test functions were employed by considering 100 decision variables (i.e., dimensions) for each test function. Test functions were solved by applying various optimization algorithms to a population of 80 for 500 iterations.

Results obtained using different algorithms to solve large-scale unimodal and multi-modal test functions are listed in Table 10. It is worth noting that the best and average NCCLA solutions reached the well-known solution or optima for 6 of the 13 functions and obtained the best solutions for the remaining functions.

NCCLA achieved excellent performance in solving large-scale test functions, even when the same number of iterations was used to solve 30-dimensional functions. This highlights the convergence speed of NCCLA compared to the other algorithms.

The statistical result obtained for the Friedman test applied to NCCLA as the control algorithm was 48.826923076923016 according to a chi-squared distribution with seven degrees of freedom, and the corresponding $\rho$ was 0.0000. NCCLA achieved the best rank, as shown in column 1 of Table 11. In addition, Holm's post-hoc test was conducted to evaluate the statistical significance of NCCLA performance. Table 11 shows $\rho$ obtained by applying Holmâôs post-hoc test to NCCLA as the control algorithm. According to results listed in column 5 of Table 11, NCCLA showed significantly better performance than WOA, GWO, PFA, SLPSO, SLA, and CSA in solving the 13 test functions. In addition, Holm's post-hoc test indicated no statistically significant differences between NCCLA and GROM performance. Holms' procedure rejects hypotheses that have a $p$ value of $\leq 0.025$.

## 5.2. Performance of NCCLA with classical engineering problems

NCCLA was applied to solve four real-world, continuous, constrained engineering design problems: tension/compression spring, welded beam, pressure vessel, and cantilever. Using the penalty functions, all the constrained problems were converted into unconstrained ones.

**Table 10**
Results of large-scale unimodal and multi-modal functions with 80 crows and 500 iterations.

| Alg.\Fun. | | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 | F11 | F12 | F13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Best | 0 | 0 | 0 | 0 | 0 | 0 | 2.69E−05 | −41898.3 | 0 | 0 | 0 | 4.71E−33 | 1.35E−32 |
| NCCLA | Avg | 0 | 0 | 1717.286104 | 0 | 0 | 0.333333 | 0.050863 | −41897.7 | 1.089118 | 0 | 0 | 4.71E−33 | 1.35E−32 |
| | S.D. | 0 | 0 | 2517.348493 | 0 | 0 | 0.745356 | 0.063877 | 2.199551 | 3.445333 | 0 | 0 | 1.37E−48 | 5.47E−48 |
| | best | 1.524E−52 | 3.72E−29 | 1.40146E−57 | 8.81E−22 | 92.49585 | 2.46004 | 5.07E−05 | −12089.6 | 0 | 8.88E−16 | 0 | 0.016125 | 3.981439 |
| GROM | Avg | 2.033E−51 | 9.3E−29 | 6.17247E−51 | 1.24E−21 | 95.02978 | 4.07E+00 | 0.000252 | −9427.55 | 0.033166 | 8.88E−16 | 0 | 0.028937 | 8.438889 |
| | S.D. | 1.439E−51 | 3.83E−29 | 3.08619E−50 | 1.64E−22 | 1.637102 | 0.73171 | 0.000113 | 947.7429 | 0.181655 | 1E-31 | 0 | 0.008464 | 1.768591 |
| | best | 3.397E−18 | 4.5E−11 | 1.451700434 | 0.00076 | 95.60586 | 4.804013 | 0.000466 | −22521.9 | 1.14E−12 | 2.65E−10 | 0 | 0.098504 | 4.248918 |
| GWO | Avg | 1.972E−17 | 8.11E−11 | 23.4384845 | 0.029006 | 97.20363 | 6.939992 | 0.002501 | −17332.3 | 5.670678 | 5.97E−10 | 0.002072 | 0.173875 | 5.312495 |
| | S.D. | 1.528E−17 | 2.44E−11 | 39.45927659 | 0.059322 | 0.875415 | 0.827528 | 0.001129 | 1894.003 | 6.95094 | 2.76E−10 | 0.005507 | 0.047015 | 0.463021 |
| | best | 4.57E−03 | 5.43E−02 | 1.25E+05 | 1.86E+01 | 9.48E+01 | 0 | 1.05E−01 | −3.75E+04 | 8.10E+02 | 1.07E−02 | 2.04E−03 | 7.86E−03 | 3.71E−02 |
| SLPSO | avg | 1.15E−02 | 1.01E−01 | 1.53E+05 | 2.32E+01 | 4.31E+02 | 5.00E−01 | 1.53E−01 | −3.61E+04 | 8.86E+02 | 1.84E−02 | 8.81E−03 | 1.08E−01 | 3.47E−01 |
| | S.D. | 0.0044469 | 0.034931 | 14942.31144 | 3.182139 | 598.9306 | 1.074789 | 0.026586 | 761.8651 | 34.01318 | 0.00408 | 0.005951 | 0.108776 | 0.290593 |
| | best | 2.36E−103 | 6.55E−61 | 482785.5697 | 0.434559 | 96.80322 | 0.30145 | 3.05E−05 | -41898 | 0 | 8.88E−16 | 0 | 0.004648 | 0.151153 |
| WOA | Avg | 6.564E−91 | 7.66E−56 | 733084.0782 | 66.70356 | 97.45627 | 0.710536 | 0.001488 | −39248.1 | 0 | 4.91E−15 | 0.004063 | 0.008356 | 0.803998 |
| | S.D. | 3.188E−90 | 3.68E−55 | 107782.5514 | 29.99599 | 0.402671 | 0.228054 | 0.001918 | 3796.925 | 0 | 2.59E−15 | 0.022254 | 0.002311 | 0.34603 |
| | best | 18050.8 | 21 | 324291 | 0 | 0 | 14774 | 248.776 | −36150.5 | 448.878 | 13.1608 | 163.457 | 41372100 | 1.48E+08 |
| SLA | Avg | 22191.743 | 27.2 | 376425.4667 | 16.6 | 4.11E−32 | 21874.9 | 512.5443 | −34942.2 | 528.652 | 14.0301 | 200.7257 | 85969607 | 2.15E+08 |
| | S.D. | 2416.4059 | 3.585411 | 33389.56471 | 10.75303 | 2.25E−31 | 2404.281 | 151.3742 | 756.3685 | 34.90379 | 0.436331 | 21.74757 | 18691958 | 36416333 |
| | best | 3.762E−10 | 1.7E−07 | 13326.394 | 0.391121 | 94.7561 | 0.113476 | 0.001421 | −34856.5 | 1.23E−09 | 3.45E−06 | 5.94E−11 | 0.002447 | 0.931414 |
| PFA | Avg | 1.767E−08 | 1.03E−06 | 46370.23358 | 2.085942 | 96.57428 | 0.729456 | 0.009169 | −25802.7 | 84.05996 | 1.18E−05 | 0.010471 | 0.178963 | 1.473349 |
| | S.D. | 3.696E−08 | 1.08E−06 | 19607.51271 | 1.705639 | 0.779889 | 0.369062 | 0.005502 | 4402.496 | 106.2562 | 9.8E−06 | 0.017715 | 0.343718 | 0.350448 |
| | best | 206.670 | 12.509 | 337078.344 | 7.613 | 227982609 | 193.728 | 0.125597 | −18970.889 | 144.037 | 3.937 | 2.789 | 591 | 136622 |
| CSA | Avg | 282.319 | 17.576 | 554414.429 | 8.933 | 1493298223 | 276.852 | 0.223 | −16313.652 | 204.265 | 4.940 | 3.519 | 127016 | 2280259 |
| | S.D. | 42.365 | 1.605 | 171619.404 | 0.718 | 4.71E+08 | 3.92E+01 | 0.0417 | 1767.879 | 30.848 | 0.558 | 0.359 | 170994 | 1816890 |

**Table 11**
Average rankings of Friedman test and $\rho$-values obtained with Holm's post hoc test using the NCCLA as a control algorithm over large-scale functions.

| DataSet/Algorithm | Ranking | p | z | Holm |
|---|---|---|---|---|
| NCCLA | 1.6538 | The control method | | |
| GROM | 3.1154 | 0.1282 | 1.5212 | 0.0500 |
| WOA | 3.5385 | 0.0498 | 1.9616 | 0.0250 |
| GWO | 4.1538 | 0.0093 | 2.6021 | 0.0167 |
| PFA | 4.6923 | 0.0016 | 3.1625 | 0.0125 |
| SLPSO | 5.1538 | 0.0003 | 3.6429 | 0.0100 |
| SLA | 6.8462 | 0.0000 | 5.4043 | 0.0083 |
| CSA | 6.8462 | 0.0000 | 5.4043 | 0.0071 |

### 5.2.1. Tension/compression spring design problem

Arora and Belegundu [61,62] introduced the tension/compression spring design problem. The problem has three design variables: mean coil diameter (D), wire diameter (d), and a number of active coils (N), as shown in Fig. 6. Considering these variables, the problem aims to minimize the total weight of the tension/compression spring, which is achieved by considering constraints such as shear stress $g_1(X)$, surge frequency $g_2(X)$, and the minimum deflection $g_3(X)$ and limits on the outside diameter $g_4(X)$. Eq. (12) shows the mathematical formulation of the design problem. In addition, Table 12 compares fitness values and design variables obtained by employing NCCLA and the other algorithms to solve the problem. Tabulated results show that NCCLA achieved the best performance when solving the problem, followed by SLA.

### 5.2.2. Welded beam design problem

Ragsdell and Phillips introduced the welded beam design problem in 1976 [63]. The problem is composed of beam *A*, which must be joint-welded to member *B*, as shown in Fig. 7. The objective is to minimize the fabrication cost of the welded beam by finding a feasible set of problem dimensions required to carry a certain load, *P*. Such dimensions represent the problem's design variables: weld-joint thickness *h*, length of the attached part of the bar *l*, bar height *t*, and bar thickness *b*. Constraints are shear stress $\tau$, bending stress in the beam *s*, bucking load on the bar *P*, and end deflection on the beam *d*. The mathematical formulation of the optimization problem is summarized in Eq. (13).

$$\begin{aligned}
&\text{Consider} \quad \vec{X} = [x_1, x_2, x_3] = [d, D, N] \\
&\text{Minimize} \quad f(\vec{X}) = (x_3 + 2)x_2x_1^2 \\
&\text{Subject to} \quad g_1(\vec{X}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} \le 0 \\
&g_2(\vec{X}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\
&g_3(\vec{X}) = 1 - \frac{(x_2^3x_3)}{71785x_1^4} \le 0 \\
&g_4(\vec{X}) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \\
&\text{Variables range} \quad 0.05 \le x_1 \le 2.00 \\
&0.25 \le x_2 \le 1.30 \\
&2.00 \le x_3 \le 15.0
\end{aligned} \tag{12}$$

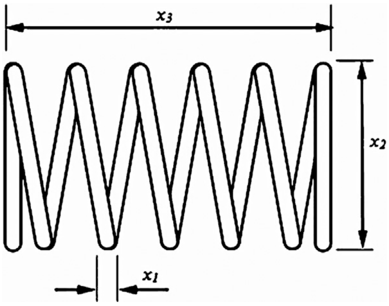

**Fig. 6.** Tension/compression spring design problem, source: [64].

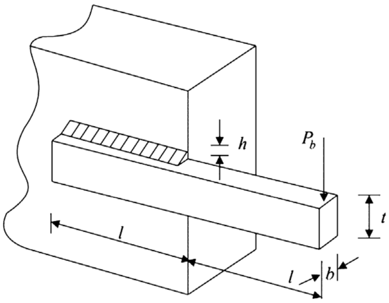

**Fig. 7.** Welded beam design problem, source: [65].

**Table 12**
Results of tension/compression spring design.

| Algo.\var. | f_min | X1 | X2 | X3 |
|---|---|---|---|---|
| NCCLA | 0.009463 | 0.046802 | 0.304659 | 12.17996 |
| GROM | 0.012666 | 0.051654 | 0.355873 | 11.33888 |
| GWO | 0.012681 | 0.051167 | 0.34418 | 12.07236 |
| WOA | 0.012701 | 0.053105 | 0.391739 | 9.496823 |
| SLA | 0.009872 | 0.05 | 0.374433 | 8.54657 |
| SLPSO | 0.012667 | 0.052043 | 0.365284 | 10.80385 |
| PFA | 0.012666 | 0.051843 | 0.36044 | 11.0742 |
| CSA | 0.012665 | 0.05165 | 0.355771 | 11.3447 |

$$\begin{aligned}
&\text{Consider} \quad \vec{X} = [x_1, x_2, x_3, x_4] = [h, l, t, b], \\
&\text{Minimize} \quad f(\vec{X}) = 1.10471x_1^2x_2 + 0.048211x_3x_4(14.0 + x_2), \\
&\text{Subject to} \quad g_1(\vec{X}) = \tau(x) - \tau_{max} \le 0, \\
&g_2(\vec{X}) = \sigma(x) - \sigma_{max} \le 0, \\
&g_3(\vec{X}) = \delta(x) - \delta_{max} \le 0, \\
&g_4(\vec{X}) = x_1 - x_4 \le 0, \\
&g_5(\vec{X}) = P - P_c(x) \le 0, \\
&g_6(\vec{X}) = 0.125 - x_1 \le 0, \\
&g_7(\vec{X}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \le 0 \\
&\text{Variables range} \quad 0.1 \le x_1 \le 2, \\
&0.1 \le x_2 \le 10, \\
&0.1 \le x_3 \le 10, \\
&0.1 \le x_4 \le 2
\end{aligned} \tag{13}$$

where $L = 14in$, $\tau_{max} = 136,000psi$, $\tau(x)$, $\sigma_{max} = 36,600psi$, $s(x)$, $P_c(x)$, $P = 6,000lb$, $d_{max}$, and $d(x)$ represent overhang length,

**Table 13**
Results of welded beam design.

| Algo.\var. | f_min | X1 | X2 | X3 | X4 |
|---|---|---|---|---|---|
| NCCLA | 1.695911 | 0.205285 | 3.260604 | 9.038254 | 0.205733 |
| GROM | 1.724852 | 0.20573 | 3.470489 | 9.036624 | 0.20573 |
| GWO | 1.725995 | 0.205529 | 3.475308 | 9.039084 | 0.205779 |
| WOA | 1.801188 | 0.194855 | 3.920691 | 8.849531 | 0.21452 |
| SLA | 1.69551 | 0.203204 | 3.30138 | 9.03658 | 0.205738 |
| SLPSO | 1.724852 | 0.20573 | 3.470489 | 9.036624 | 0.20573 |
| PFA | 1.724875 | 0.205729 | 3.470483 | 9.036765 | 0.20573 |
| CSA | 1.724856 | 0.20573 | 3.470484 | 9.036649 | 0.20573 |

allowable shear stress of the weld, weld shear stress, allowable yield stress for the bar material, the bending stress, bar bucking load, loading condition, allowable bar end deflection, and bar end deflection, respectively. In addition, $E = 30 \times 10^6$ and terms $\tau(x)$, $\sigma(x)$, $P_c(x)$, and $d(x)$ are calculated as follows.

$$
\begin{aligned}
\tau(\vec{X}) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2},\\
\tau' = \frac{P}{\sqrt{2}x_1x_2},\ \tau'' &= \frac{MR}{J},\ M = P(L + \frac{x_2}{2}),\\
R &= \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2},\\
J &= 2\sqrt{2}x_1x_2[\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2],\\
\sigma(\vec{X}) &= \frac{6PL}{x_4x_3^2},\ \delta(\vec{X}) = \frac{6PL^3}{Ex_3^2x_4},\\
P_c(\vec{X}) &= \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}})
\end{aligned}
\tag{14}
$$

Table 13 compares fitness values and design variables obtained by solving the problem with NCCLA and the other algorithms. Results show that NCCLA achieved the second-best performance (1.695911392) in solving the problem, which is nearly identical to the best performance (1.69551) achieved by SLA.

### 5.2.3. Pressure vessel design problem

The pressure vessel problem model is illustrated in Fig. 8. This problem seeks to minimize the total cost consisting of the material, formation, and welding of a cylindrical vessel capped on both ends, with a hemispherical head [66]. Design variables include shell thickness $T_s$, head thickness $T_h$, inner radius $R$, and length of the cylindrical part without head $L$. The problem is subject to constraints shown in Eq. (15). Results listed in Table 14 demonstrate that most of the algorithms (including SLA, GROM, and PFA) achieved the best results (5885.33, 5885.331, and 5885.365, respectively) with nearly identical performance in solving the problem. In addition, NCCLA achieved a reasonably comparable result of 5887.806 in solving the problem.

### 5.2.4. Cantilever design problem

The cantilever design optimization problem aims to minimize the weight of a beam [67], as shown in Fig. 9. The problem considers five design variables, $x_1$–$x_5$, to reach this aim. The model in Fig. 9 exhibits five hollow square elements showing a constant wall/perimeter thickness. In addition, the first element is supported rigidly, whereas node 5 can hold a vertical load on the free end of the beam. The problem is subject to one constraint, $g(X)$, which imposes vertical displacement, as illustrated in Eq. (16). Almost all tested algorithms showed good performance in solving the problem, as shown by results listed in Table 15. GROM, NCCLA, and SLPSO exhibited comparable performance of 1.339956, 1.339957, and 1.339957377, respectively.

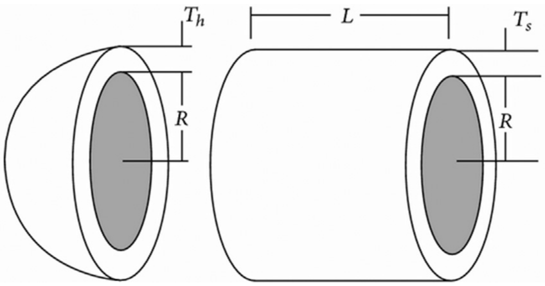

**Fig. 8.** The pressure vessel design problem, source: [68].

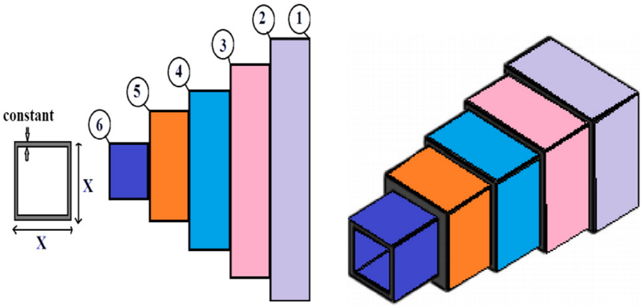

**Fig. 9.** Cantilever design problem, source: [33].

$$
\begin{aligned}
&\text{Consider}\quad \vec{X} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L],\\
&\text{Minimize}\quad f(\vec{X}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2\\
&\qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3,\\
&\text{Subject to}\quad g_1(\vec{X}) = -x_1 + 0.0193x_3 \le 0\\
&g_2(\vec{X}) = -x_3 + 0.00954x_3 \le 0\\
&g_3(\vec{X}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0\\
&g_4(\vec{X}) = x_4 - 240 \le 0\\
&\text{Variables range}\quad 0 \le x_1 \le 99\\
&0 \le x_2 \le 99,\\
&10 \le x_3 \le 200,\\
&10 \le x_4 \le 200
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
&\text{Consider}\quad \vec{X} = [x_1, x_2, x_3, x_4, x_5] = [v_1, v_2, v_3, v_4, v_5]\\
&\text{Minimize}\quad f(\vec{X}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5),\\
&\text{Subject to}\quad g_1(\vec{X}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \le 0,\\
&\text{Variables range}\quad 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100
\end{aligned}
\tag{16}
$$

## 6. Conclusion

In this paper, we introduced an NC-crow learning algorithm, NCCLA, inspired by mechanisms that NC-crows use to learn behaviors for developing *Pandanus* tools to obtain food. The NCCLA algorithm was applied to solve continuous optimization problems, and its ability to solve such problems was tested and

**Table 14**
Results of pressure vessel design.

| Algo./variables | f_min | X1 | X2 | X3 | X4 |
|---|---|---|---|---|---|
| NCCLA | 5887.806 | 0.778698 | 0.385095 | 40.34487 | 199.6833 |
| GROM | 5885.331 | 0.778168 | 0.384649 | 40.31962 | 200 |
| GWO | 5892.678 | 0.779063 | 0.385935 | 40.34745 | 199.6541 |
| WOA | 6018.055 | 0.82361 | 0.422247 | 42.66257 | 169.7694 |
| SLA | 5885.33 | 0.778169 | 0.38465 | 40.3197 | 199.999 |
| SLPSO | 5908.631 | 0.791567 | 0.391272 | 41.01385 | 190.5564 |
| PFA | 5885.365 | 0.778167 | 0.384653 | 40.31964 | 200 |
| CSA | 5886.556 | 0.778525 | 0.384895 | 40.33686 | 199.7718 |

**Table 15**
Results of cantilever beam design.

| Algo./variables | f_min | X1 | X2 | X3 | X4 | X5 |
|---|---|---|---|---|---|---|
| NCCLA | 1.339957 | 6.013778 | 5.311017 | 4.49201 | 3.505521 | 2.151349 |
| GROM | 1.339956 | 6.015849 | 5.308661 | 4.494831 | 3.501864 | 2.152454 |
| GWO | 1.339977 | 6.027633 | 5.316478 | 4.484833 | 3.493066 | 2.151975 |
| WOA | 1.349924 | 6.085676 | 5.215101 | 4.136845 | 3.835008 | 2.360773 |
| SLA | 1.33996 | 6.01699 | 5.32277 | 4.52573 | 3.4861 | 2.12381 |
| SLPSO | 1.339957 | 6.015077 | 5.308306 | 4.496387 | 3.501656 | 2.152236 |
| PFA | 1.33996 | 6.019605 | 5.30383 | 4.500031 | 3.501003 | 2.149244 |
| CSA | 1.339957 | 6.016677 | 5.307885 | 4.494765 | 3.50259 | 2.151751 |

evaluated using 23 different benchmark test functions and four engineering test problems. Experimental results showed that the NCCLA has an excellent ability to solve simple and complex optimization problems. NCCLA also demonstrated good ability to escape from local minima of multi-modal functions. Furthermore, NCCLA delivered superior performance in solving unimodal, most multi-modal, and large-scale functions. All the experimental results indicated that NCCLA is a promising algorithm that can be enhanced and applied to solve other categories of optimization problems. Results also showed that NCCLA significantly outperformed all the other algorithms with most of the 23 benchmarks. Employing learning mechanisms in optimization algorithms resulted in the development of effective optimizers such as NCCLA, SLA, and SLPSO; it accelerated convergence while maintaining a good balance between exploration and exploitation. The following NCCLA characteristics can be identified from the results: (a) NCCLA shows a good balance between exploitation and exploration, so solutions are updated very well, and quickly converge to the best final results. (b) According to the learning and reinforcement procedures, NCCLA is a competitive algorithm. (c) NCCLA algorithm is robust and achieves approximately the same results in different experiments. (d) NCCLA has high performance in solving continuous optimization problems, including classical problems with different dimensions and real engineering problems. As a future study, NCCLA will be applied to solve different discrete optimization and real-world problems. Furthermore, NCCLA performance using multiple families will be studied in our future research. It will also be possible to combine NCCLA with other promising algorithms, including GROM, SLA, and PFA, which have demonstrated a good performance.

## CRediT authorship contribution statement

**Wedad Al-Sorori:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Visualization, Writing - original draft. **Abdulqader M. Mohsen:** Conceptualization, Methodology, Supervision, Validation, Formal analysis, Investigation, Data curation, Visualization, Writing - original draft, Writing - review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

[1] X.-s. Yang, L. Press, Nature-Inspired Metaheuristic Algorithms Second Edition, Luniver Press, 2010.

[2] X.-s. Yang, Nature-Inspired Optimization Algorithms, Elsevier, 2014.

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

[4] L. Lin, M. Gen, Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation, Soft Comput. 13 (2) (2009) 157–168.

[5] A. Neme, S. Hernández, Algorithms inspired in social phenomena, in: Nature-Inspired Algorithms for Optimisation, Springer, 2009, pp. 369–387.

[6] A. Tang, The Collective Intelligence Concept: A Literature Review FromThe Behavioral and Cognitive Perspective, (August), 2016, pp. 23–24.

[7] H. Ma, S. Ye, D. Simon, M. Fei, Conceptual and numerical comparisons of swarm intelligence optimization algorithms (2017) 3081–3100, https://doi.org/10.1007/s00500-015-1993-x.

[8] R. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, Citeseer, 1995, pp. 1942–1948.

[9] X.-S. Yang, Firefly algorithms for multimodal optimization, in: International Symposium on Stochastic Algorithms, Springer, 2009, pp. 169–178.

[10] M. Dorigo, G. Di Caro, Ant colony optimization: a new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 2, IEEE, 1999, pp. 1470–1477.

[11] K.N. Laland, Social learning in animals : Empirical studies and theoretical models, 55 (6) (2005).

[12] J.C. Holzhaider, G.R. Hunt, R.D. Gray, Social learning in new caledonian crows, Learn. Behav. 38 (3) (2010) 206–219, http://dx.doi.org/10.3758/LB.38.3.206.

[13] M.A. Montes de Oca, T. Stützle, Towards incremental social learning in optimization and multiagent systems, in: Proceedings of the 10th Annual Conference Companion on Genetic and Evolutionary Computation, ACM, 2008, pp. 1939–1944.

[14] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inform. Sci. 291 (2015) 43–60.

[15] R.V. Rao, V.J. Savsani, D. Vakharia, Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems, Inform. Sci. 183 (1) (2012) 1–15.

[16] Y.-J. Gong, J. Zhang, Y. List, From the social learning theory to a social learning algorithm for global optimization, in: Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on, IEEE, 2014, pp. 222–227.

[17] M. Kumar, A.J. Kulkarni, S.C. Satapathy, Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology, Future Gener. Comput. Syst. 81 (2018) 252–272.

[18] L. Wang, H. Ni, R. Yang, M. Fei, W. Ye, A simple human learning optimization algorithm, in: International Conference on Life System Modeling and Simulation and International Conference on Intelligent Computing for Sustainable Energy and Environment, Springer, 2014, pp. 56–65.

[19] L. Wang, L. An, J. Pi, M. Fei, P.M. Pardalos, A diverse human learning optimization algorithm, 2016, http://dx.doi.org/10.1007/s10898-016-0444-2.

[20] E.V. Altay, B. Alatas, Performance comparisons of socially inspired metaheuristic algorithms on unconstrained global optimization, in: Advances in Computer Communication and Computational Sciences, Springer, 2019, pp. 163–175.

[21] T.T. Khuat, M.H. Le, A survey on human social phenomena inspired algorithms, Int. J. Comput. Sci. Inf. Secur. 14 (6) (2016) 76.

[22] H.N. Cornell, J.M. Marzluff, S. Pecoraro, Social learning spreads knowledge about dangerous humans among american crows, Proc. R. Soc. Lond. Biol. 279 (1728) (2011) 499–508.

[23] B.G. Galef Jr., L.-A. Giraldeau, Social influences on foraging in vertebrates: causal mechanisms and adaptive functions, Anim. Behav. 61 (1) (2001) 3–15.

[24] J.-F. Gariépy, K.K. Watson, E. Du, D.L. Xie, J. Erb, D. Amasino, M.L. Platt, Social learning in humans and other animals, Front. Neurosci. 8 (2014) 58.

[25] R.L. Knight, D.J. Grout, S.A. Temple, Nest-defense behavior of the american crow in urban and rural areas, Condor 89 (1) (1987) 175–177.

[26] D.F. Caccamise, L.M. Reed, J. Romanowski, P.C. Stouffer, Roosting behavior and group territoriality in American crows, Auk (1997) 628–637.

[27] A. Computers, S. June, A novel metaheuristic method for solving constrained engineering optimization problems : Crow search algorithm, Comput. Struct. 169 (May) (2016) 1–12, http://dx.doi.org/10.1016/j.compstruc.2016.03.001.

[28] A. Brabazon, W. Cui, M.O. Neill, The raven roosting optimisation algorithm, 2015, http://dx.doi.org/10.1007/s00500-014-1520-5.

[29] G.I. Sayed, A.E. Hassanien, A.T. Azar, Feature selection via a novel chaotic crow search algorithm, Neural Comput. Appl. 31 (1) (2019) 171–188.

[30] S. Torabi, F. Safi-Esfahani, Improved raven roosting optimization algorithm (irro), Swarm Evol. Comput. 40 (2018) 144–154.

[31] Y.-j. Gong, J. Zhang, From the social learning theory to a social learning algorithm for global optimization, (January) (2015) 5–8.

[32] M. Fahle, S. Edelman, Long-term learning in vernier acuity: Effects of stimulus orientation, range and of feedback, Vis. Res. 33 (3) (1993) 397–412.

[33] A.F. Nematollahi, A. Rahiminejad, B. Vahidi, A novel meta-heuristic optimization method based on golden ratio in nature, Soft Comput. (2019) 1–35.

[34] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey.wolf. optimizer, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.

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

[36] H. Yapici, N. Cetinkaya, A new meta-heuristic optimizer: Pathfinder algorithm, Appl. Soft Comput. 78 (2019) 545–568.

[37] D.H. Wolpert, W.G. Macready, et al., No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.

[38] Y.-C. Ho, D.L. Pepyne, Simple explanation of the no-free-lunch theorem and its implications, J. Optim. Theory Appl. 115 (3) (2002) 549–570.

[39] Abdulqader M Mohsen, Wedad Al-Sorori, A new hybrid discrete firefly algorithm for solving the traveling salesman problem, Springer, 2017, pp. 169–180.

[40] Wedad Al-Sorori, Abdulqader Mohsen, et al., An improved hybrid bat algorithm for traveling salesman problem, in: International Conference on Bio-Inspired Computing: Theories and Applications, Springer, 2016, pp. 504–511.

[41] Asma M Altabeeb, Abdulqader M Mohsen, Abdullatif Ghallab, An improved hybrid firefly algorithm for capacitated vehicle routing problem, Applied Soft Computing 84 (2019) 105728.

[42] Abdulqader M Mohsen, Annealing ant colony optimization with mutation operator for solving tsp, Computational intelligence and neuroscience 2016 (2016).

[43] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm Evol. Comput. 44 (2019) 148–175.

[44] J.C. Holzhaider, G.R. Hunt, R.D. Gray, The development of pandanus tool manufacture in wild new caledonian crows, Behaviour 147 (5) (2010) 553–586.

[45] J.C. Holzhaider, M. Sibley, A. Taylor, P. Singh, R.D. Gray, G. Hunt, The social structure of new caledonian crows, Anim. Behav. 81 (1) (2011) 83–92.

[46] T.R. Zentall, Perspectives on observational learning in animals, J. Comp. Psychol. 126 (2) (2012) 114.

[47] J.F. Wilson, M.B. Cantor, An animal model of excessive eating: Schedule-induced hyperphagia in food-satiated rats, J. Exp. Anal. Behav. 47 (3) (1987) 335–346.

[48] J. Noble, P.M. Todd, Imitation or something simpler? Modeling simple mechanisms for social information processing, Imitation Animals Artifacts (2002) 423–439.

[49] A. Bandura, R.H. Walters, Social Learning Theory, vol. 1, Prentice-hall Englewood Cliffs, NJ, 1977.

[50] G.R. Hunt, J. Abdelkrim, M.G. Anderson, J.C. Holzhaider, A.J. Marshall, N.J. Gemmell, R.D. Gray, Innovative pandanus-tool folding by new caledonian crows, Aust. J. Zool. 55 (5) (2008) 291–298.

[51] K. Sörensen, Metaheuristics the metaphor exposed, Int. Trans. Oper. Res. 22 (1) (2015) 3–18.

[52] A. Cheraghalipour, M. Hajiaghaei-Keshteli, M.M. Paydar, Tree growth algorithm (tga): A novel approach for solving optimization problems, Eng. Appl. Artif. Intell. 72 (2018) 393–414.

[53] J.M. Dieterich, B. Hartke, Empirical review of standard benchmark functions using evolutionary global optimization, 2012, arXiv preprint arXiv:1207.4318.

[54] M. Jamil, X.-S. Yang, A literature survey of benchmark functions for global optimization problems, 2013, arXiv preprint arXiv:1308.4008.

[55] X.-S. Yang, Test problems in optimization, 2010, arXiv preprint arXiv:1008.0549.

[56] B. Akay, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Inform. Sci. 192 (2012) 120–142.

[57] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Tech. Rep. Technical Report-tr06, Erciyes University, 2005, Engineering faculty, computer.

[58] T. Ganesan, P. Vasant, I. Elamvazuthy, A hybrid pso approach for solving non-convex optimization problems, Arch. Control Sci. 22 (1) (2012) 87–105.

[59] C. Gologlu, N. Sakarya, The effects of cutter path strategies on surface roughness of pocket milling of 1.2738 steel based on taguchi method, J. Mater. Process. Technol. 206 (1–3) (2008) 7–15.

[60] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.

[61] J. Arora, Introduction to optimum design, ny, 1998.

[62] A.D. Belegundu, A study of mathematical programming methods for structural optimization, 1983.

[63] K. Ragsdell, D. Phillips, Optimal design of a class of welded structures using geometric programming, 1976.

[64] L.C. Cagnina, S.C. Esquivel, C.A.C. Coello, Solving engineering optimization problems with the simple constrained particle swarm optimizer, Informatica 32 (3) (2008).

[65] Saeid Barshandeh, Maryam Haghzadeh, A new hybrid chaotic atom search optimization based on tree-seed algorithm and levy flight for solving optimization problems, Engineering with Computers (2020) 1–44.

[66] B. Kannan, S.N. Kramer, An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, J. Mech. Des. 116 (2) (1994) 405–411.

[67] H. Chickermane, H. Gea, Structural optimization using a new local approximation method, Internat. J. Numer. Methods Engrg. 39 (5) (1996) 829–846.

[68] Hongqing Zheng, Yongquan Zhou, A cooperative coevolutionary cuckoo search algorithm for optimization problem, Journal of Applied Mathematics (2013) 1–9.