IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 19, 2020, accepted May 27, 2020, date of publication June 2, 2020, date of current version June 23, 2020.
Digital Object Identifier 10.1109/ACCESS.2020.2999540

Border Collie Optimization

TULIKA DUTTA[1], SIDDHARTHA BHATTACHARYYA[2], (Senior Member, IEEE), SANDIP DEY[3], (Member, IEEE), AND JAN PLATOS[4], (Member, IEEE)
[1]Department of Computer Science and Engineering, University Institute of Technology, Burdwan 713104, India
[2]Department of Computer Science and Engineering, CHRIST (Deemed to be University), Bengaluru 560029, India
[3]Department of Computer Science, Sukanta Mahavidyalaya, Jalpaiguri 735210, India
[4]Department of Computer Science, FEECS, VSB-Technical University of Ostrava, 70800 Ostrava, Czech Republic

Corresponding author: Siddhartha Bhattacharyya (dr.siddhartha.bhattacharyya@gmail.com)

This work was supported by the Student Grant Competition (SGS), VSB-Technical University of Ostrava, Czech Republic, through the Parallel processing of Big Data VII, under Grant SP2020/108 and the European Regional Development Fund (ERDF) "A Research Platform focused on Industry 4.0 and Robotics in Ostrava", No. CZ.02.1.01/0.0/0.0/17_049/0008425.

ABSTRACT In recent times, several metaheuristic algorithms have been proposed for solving real world optimization problems. In this paper, a new metaheuristic algorithm, called the Border Collie Optimization is introduced. The algorithm is developed by mimicking the sheep herding styles of Border Collie dogs. The Border Collie's unique herding style from the front as well as from the sides is adopted successfully in this paper. In this algorithm, the entire population is divided into two parts viz., dogs and sheep. This is done to equally focus on both exploration and exploitation of the search space. The Border Collie utilizes a predatory move called eyeing. This technique of the dogs is utilized to prevent the algorithm from getting stuck into local optima. A sensitivity analysis of the proposed algorithm has been carried out using the Sobol's sensitivity indices with the Sobol g-function for tuning of parameters. The proposed algorithm is applied on thirty-five benchmark functions. The proposed algorithm provides very competitive results, when compared with seven state-of-the-art algorithms like Ant Colony optimization, Differential algorithm, Genetic algorithm, Grey-wolf optimizer, Harris Hawk optimization, Particle Swarm optimization and Whale optimization algorithm. The performance of the proposed algorithm is analytically and visually tested by different methods to judge its supremacy. Finally, the statistical significance of the proposed algorithm is established by comparing it with other algorithms by employing Kruskal-Wallis test and Friedman test.

INDEX TERMS Benchmark test functions, Border Collie optimization, Friedman test, Kruskal-Wallis test, metaheuristic, optimization, swarm intelligence.

I. INTRODUCTION

Optimization is the process of finding the most effective solution to a problem. Due to its versatile scope of application, it is very difficult to provide an exact definition. Mathematically, optimization can be defined as finding a *maxima* or *minima* of a real function [1]. In terms of computing and engineering, optimization can be defined as a system which maximizes the objectives by utilizing fewer resources. Optimization algorithms can be classified into different groups.

Based on the number of objectives, optimization problems can be of two types viz., single objective and multi-objective problems [2]. In real world scenario, most of the problems are multi-objective.

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

Based on the nature of algorithms, optimization algorithms can be classified as deterministic, stochastic and hybrid algorithms. Deterministic algorithms are those which always follow the same steps and produce the same results for a particular problem. Stochastic algorithms on the other hand are random in nature and may produce different results every time. Hybrid algorithms are a combination of deterministic and stochastic algorithms.

Metaheuristic algorithm are special types of stochastic algorithms. They can produce near optimal solutions in comparatively lesser time. Simplicity and efficiency of the algorithms have made them extremely popular among researchers. They are mostly derived from physical phenomena or from behaviors of different living beings. The behavioral study of ants, birds, fishes, wolves are few well known examples which has inspired algorithms like Ant Colony Optimization (ACO) [3], Particle Swarm

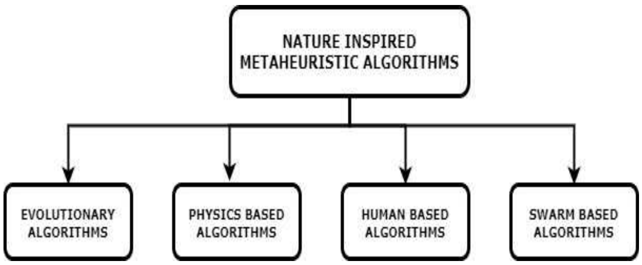


FIGURE 1. Types of nature-inspired metaheuristic algorithms [9].

TABLE 1. List of evolutionary algorithms.

Sr. No.	Algorithm	Year
1.	Evolutionary programming [10] (EP)	1966
2.	Genetic algorithm [11] (GA)	1975
3.	Tabu search [12] (TS)	1986
4.	Co-evolving algorithm [13]	1990
5.	Cultural algorithm [14]	1994
6.	Genetic Programming [15]	1994
7.	Differential evolution [16] (DE)	1997
8.	Quantum evolutionary algorithm [17]	2002
9.	Human evolutionary model [18]	2007
10.	Biogeography-Based Optimization [19]	2008
11.	Differential search algorithm [20]	2011
12.	Evolutionary membrane algorithm [21]	2012
13.	Backtracking optimization algorithm [22]	2013
14.	Stochastic fractal search [23]	2014
15.	Synergistic fibroblast optimization [24]	2018
16.	Physarum-inspired computational model [25]	2019

Optimization (PSO) [4] and Grey Wolf Optimization (GWO) [5] among others. Metaheuristic algorithms are extremely flexible in nature [5]. The same algorithm can be efficiently used for different purposes such as, thresholding of images [6], classification of satellite images [7] as well as optimizing benchmark functions [8], etc. Metaheuristics also have an excellent exploitation capability and local optima avoidance mechanism, thus making them a popular choice for solving optimization problems. Though they are efficient algorithms, yet it has been proved that no metaheuristic is capable of solving all optimization problems.

Metaheuristic algorithms are mostly inspired by natural phenomena. They can be classified based on their sources [9], as depicted in Fig. 1.

1) *Evolutionary Algorithms* - Biological evolution is a gradual process of change and improvement, for the purpose of producing better offsprings. The metaheuristic algorithms based on this mechanism are called *evolutionary algorithms*. They use genetic operators like mutation, natural selection and crossover to produce better evolved generations.
   In Table 1, a timeline of few evolutionary algorithms is presented.
2) *Physics based Algorithms* - These algorithms are inspired from physical phenomena. Optimization is done based on physical laws like gravitational force, magnetic force and others.

TABLE 2. List of physics based algorithms.

Sr. No.	Algorithm	Year
1.	Simulated annealing [26]	1983
2.	Harmony search [27] (HS)	2001
3.	Gravitational search optimization algorithm [28] (GSA)	2003
4.	Charged system search [29]	2010
5.	Electro-magnetism optimization [30]	2011
6.	Water cycle algorithm [31]	2012
7	Multi-Verse Optimizer [32]	2015
8.	Sine cosine algorithm [33] (SCA)	2016
9.	Henry gas solubility optimization [34]	2019

TABLE 3. List of human based algorithms.

Sr. No.	Algorithm	Year
1.	Society and civilization [35]	2003
2.	Human-inspired algorithm [36]	2009
3.	TeachingâĂŞlearning-based optimization [37] (TLBO)	2011
4.	Gaining-sharing knowledge based algorithm [9]	2019

   In Table 2, few widely used physics based metaheuristics are enlisted.
3) *Human based Algorithms* - Metaheuristic algorithms inspired from human behavior fall in this category. The algorithms are based on the physical activities of humans like walking, talking and others, as well as non-physical activities like thinking.
   Few of these optimization algorithms are presented in Table 3.
4) *Swarm based Algorithms* - Swarm based metaheuristics are inspired by the social behavior of insects or animals. In a swarm, each individual has its own intelligence and behavior. The combined behavior of the individuals makes the swarm a powerful tool to solve complex problems.
   In Table 4, few popular swarm based algorithms are presented. Swarm based metaheuristics are capable of achieving more optimal results as compared to other metaheuristics. They are easy to implement and require lesser number of parameters. Complex operators like mutation, elitism and crossover used in the evolutionary algorithms are not required to implement swarms. They often preserve the search space over the iterations and utilize memory to save the best solutions.

In Table 5, a comparative study of few well known metaheuristic algorithms are presented. Every algorithm has its own merits and demerits. Hence one algorithm may perform very well for any particular problem and very poorly for others. To overcome these limitations, three kinds of approaches are adopted. These are (i) improving the existing algorithms, (ii) hybridizing the existing algorithms and (iii) introducing new metaheuristic algorithms.

The improved algorithms are designed using the basic principles of some algorithms, which have already been introduced in the literature. These are basically the improved versions of the said algorithms. In [65], a family genetic algorithm has been proposed, which outperformed the basic GA, with regards to convergence speed. An improved

TABLE 4. List of swarm based algorithms.

Sr. No.	Algorithm	Year
1.	Ant colony optimization [3]	1992
2.	Particle Swarm Optimization [4]	1995
3.	Bacterial foraging [38]	2002
4.	Honey bee swarm optimization algorithm [39]	2005
5.	Artificial bee colony [40] (ABC)	2007
6.	Cuckoo search [41] (CS)	2009
7.	Bat algorithm [42]	2010
8.	Firefly algorithm [43]	2010
9	Fruit fly optimization algorithm [44]	2011
10.	Flower pollination algorithm [45]	2012
11.	Krill herd algorithm [46]	2012
12.	Grey wolf optimizer [5]	2014
13.	Spider Monkey Optimization [47]	2014
14.	Moth-flame optimization algorithm [48]	2015
15.	Ant lion optimizer [49]	2015
16.	Dragonfly algorithm [50]	2015
17.	Bird swarm algorithm [51]	2015
18.	Whale optimization algorithm [52] (WOA)	2016
19.	Crow Search Algorithm [53] (CSA)	2016
20.	Grasshopper optimization algorithm [54] (GOA)	2017
21.	Salp swarm algorithm [55]	2017
22.	Spotted hyena optimizer [56]	2017
23.	Squirrel search algorithm [57]	2019
24.	Harris Hawk optimization [58] (HHO)	2019
25.	Red deer algorithm [59]	2020
26.	Wingsuit Flying Search [60]	2020
27.	Tunicate Swarm Algorithm [61]	2020
28.	Vortex Swarm Optimization [62]	2020
29.	Artificial Cell Swarm Optimization [63]	2020
30.	Orcas Algorithm [64]	2020

DE algorithm with modification in chromosome representation has been developed by Das *et al.* in [66], called the automatic clustering DE (ACDE). The ACDE has a faster convergence speed than the original DE algorithm. An improved version of the HS algorithm has been conceptualized in [67] by Portilla-Flores *et al.* This algorithm increased the exploration and exploitation of the basic HS [27] algorithm, with decreased computational cost. In [68], Liu and Ma developed an improved GSA algorithm based on free search differential evolution, which enhanced the exploitation capability of the GSA algorithm. Wang *et al.* [69], improved the exploitation capability of the sine cosine algorithm using an adaptive probability selection technique. In [70], an improved version of the TLBO algorithm has been proposed to enhance the searching ability and accuracy of the basic TLBO [37] algorithm. This has been achieved by introducing an S-shaped group learning phase instead of the random learning phase.

A Multi-Population Co-Evolution Ant Colony Optimization (ICMPACO) has been developed by Deng *et al.* in [71]. This algorithm increased the population diversity of the basic ACO [3] algorithm. In addition, it also improved the convergence speed of the proposed algorithm. An improved PSO, called the heterogeneous comprehensive learning PSO (HCLPSO) has been proposed in [72]. The exploration and exploitation capabilities of the PSO have also been increased by employing comprehensive learning mechanisms. Zhang and Liu introduced a discrete and improved artificial bee colony (DiABC) algorithm, with enhanced convergence speed in [73]. CS [41] algorithm has a low search efficiency since it uses a single search strategy in the population. Gao *et al.* [74] developed a multi-strategy adaptive cuckoo algorithm (MSACS) to overcome the search efficiency problem. Five different search strategies have been used and compared with previous strategies and control parameters, to perform the optimization process in MSACS. The GWO proposed by Mirjalili *et al.* [5] performed poorly in terms of exploration of the search space. To overcome this limitation, a nonlinear control parameter strategy has been introduced by Long *et al.* [75], to balance the exploration and exploitation capabilities. In [76], an enhanced GWO (EGWO) is proposed for diversifying the population. The introduction of chaotic theory in GWO efficiently increases the balancing between exploration and exploitation of the search space. A lévy flight based variant of WOA has been proposed in [77]. The use of lévy flight based trajectory helped to increase the diversity of the population, restrained it from premature convergence and enhanced the capability of escaping from getting stuck in local optima. Han *et al.* [78] introduced a weight coefficient along with a guidance position and a spiral search mechanism, in CSA. These helped to enhance the balancing between the exploration and exploitation of the search space. By introducing the gravity search operator in [79], the global exploration of the GOA has been improved.

The Krill Herd algorithm [46] has a slow convergence speed and gets stuck in local optima. In [80], three one-dimensional chaotic maps viz., Circle, Sine and Tent are introduced in the Krill Herd algorithm to overcome the limitations. In [81], the fruit fly optimization algorithm is applied to a Support Vector Machine (SVM) for inner parameter optimization. The fruit fly optimization algorithm effectively adjusts the SVM parameters, thus enhancing the generalization capability of the SVM classifier in medical data classification. Wang *et al.* [82] proposed a chaotic moth flame optimization algorithm by introducing chaotic behavior in two steps. Chaotic operation was introduced during population initialization for getting a diverse population. A chaotic disturbance mechanism was also adopted for rescuing the algorithm from falling into local optima. The chaotic moth flame algorithm along with kernel extreme learning machine strategy provided a better classification mechanism and reduced feature subsets in the field of medical diagnosis. Xu *et al.* [83] introduced mutation operators like Gaussian mutation, Cauchy mutation, Lévy mutation or their combination in the moth flame algorithm. The exploration and exploitation capabilities of the moth flame algorithm are greatly enhanced by applying the mutation operators. The Bacterial Foraging Optimization algorithm [38] has several drawbacks like slow convergence speed, getting stuck into local optima and fixed step lengths. To overcome these limitations, an enhanced Bacterial Foraging Optimization algorithm with gaussian mutation, chaotic local search and chaotic chemotaxis step length has been proposed by

TABLE 5. Merits and demerits of popular metaheuristic algorithms.

Metaheuristic Algorithms	Merits	Demerits
GA [11]	1. Capable of reaching global optima faster.	1. Low convergence speed. 2. The crossover and mutation operators are fixed which reduce flexibility.
DE [16]	1. Easy implementation. 2. Few control parameters. 3. Low space complexity.	1. To solve a specific optimization problem, trial and error method of finding the most appropriate strategy is required. 2. High computational cost.
HS [27]	1. Easy implementation. 2. Fast convergence. 3. Good balance between exploration and exploitation.	1. Control parameters are fixed values, so the algorithm easily gets trapped in local optima, if not tuned properly.
GSA [28]	1. Good convergence speed. 2. Accurate solutions.	1. Concentrates more on exploration.
SCA [33]	1. Simplicity, flexibility, and efficiency.	1. Easily converges into local optima.
TLBO [37]	1. Nearly parameter free. 2. Faster convergence.	1. Gets stuck in local optima.
ACO [3]	1. Positive feedback mechanism helps in finding optimal solution faster.	1. Slow convergence speed. 2. Decrease in population diversity due to feedback from pheromone trails.
PSO [4]	1. Faster convergence speed. 2. Lesser number of parameters.	1. Suffers diversity loss near local optimum.
ABC [40]	1. Few parameters and strong exploration abilities.	1. Exploitation of search space is low.
CS [41]	1. Lesser number of parameters.	1. Search efficiency is low for complex problems with multiple peaks.
GWO [5]	1. Ability of avoiding local optima. 2. Good convergence capability.	1. Lacks in extensive exploration of search space. 2. Fails to find global optima.
WOA [52]	1. Lesser number of adjustable parameters. 2. Simple to implement.	1. Slow convergence and gets easily trapped into local optimum when dealing with high dimensional complex problems.
CSA [53]	1. Avoids local optima easily while solving complex, high dimensional. and multimodal problems.	1. Local search strategy is not efficient.
GOA [54]	1. Avoids the stagnation of local optimality to some extent.	1. Poor global search ability. 2. Slower convergence speed.
HHO [58]	1. High-quality solutions. 2. Strong robustness. 3. Smooth transition between exploration and exploitation.	1. The exploration and exploitation phases are unbalanced. 2. Premature convergence. 3. Getting stuck into local optima.

Chen *et al.* [84]. HHO [58] is a relatively new metaheuristic algorithm proposed in 2019. Menesy *et al.* [85] applied ten chaotic functions on the HHO algorithm, to enhance its searching ability and reduce the probability from getting stuck into local optima.

Hybridization of metaheuristic algorithms has been widely adopted by researchers. These kinds of algorithms provide better results by improvising the inherent advantages of the parent algorithms. In [86], a hybrid GA and PSO algorithm has been developed to solve supply chain distribution problem. Ouyang *et al.* [87] combined the Teaching-learning-based algorithm with the HS algorithm to enhance the global search capability and local exploitation capability of the TLBO. In [88], the exploitation capability of simulated annealing [26] has been combined with the exploration capability of WOA. In [89], fuzzy logic has been used to combine the gravitational search algorithm with a local search technique for function optimization. In [90], Bao *et al.* developed a hybrid algorithm by applying HHO and DE in parallel. The proposed algorithm has been found to be a powerful tool for thresholding of color images. In [91], a hybrid algorithm of GWO and CSA has been proposed for function optimization and feature selection. The firefly algorithm has been combined with PSO for automatic clustering in [92].

From the above discussions, we can infer that several methods have been developed so far to minimize the demerits of the existing metaheuristic algorithms. In the literature, numerous algorithms have been successfully designed to handle certain problems. However, no single metaheuristic algorithm has been found to be capable of addressing all the optimization problems successfully.Moreover, improved and hybridized algorithms may suffer from added computational burden. Our aim is to develop a metaheuristic which can overcome this limitation.

In [93], Wolpert and Macready stated that when an algorithm produces effective results for a certain class of problems, it may not perform well for other kinds of problems. Though a lot of researchers are rigorously working on this from past few decades, no such metaheuristic has yet been introduced so far that can efficiently handle all sorts of optimization problems. This is called the "*no free lunch*" theorem [93]. So it can be inferred that new optimization algorithms need to be developed that outperform the existing algorithms, for dealing with certain problems.

This is the main inspiration behind this work to propose a new swarm intelligent metaheuristic algorithm. In this paper, we have proposed a metaheuristic algorithm, called the *Border Collie Optimization* (BCO) by mimicking the herding behavior of the Border Collie dogs.

Border Collies are affectionate, smart, and energetic breed of dogs [94]. They are extremely intelligent, athletic and can be easily trained. These dogs are usually healthy and active, having a normal life span of about 12 to 15 years. It can be said that, watching a border collie herd sheep is like watching a master craftsman at work. Herding is an inherent ability they are born with. Even when a puppy is introduced to the herd for the first time, they demonstrate immense control over the sheep. A representative image of a Border Collie dog is given in Fig. 2.

The intelligent and unique approach of these dogs in herding the sheep has inspired us to introduce a novel metaheuristic, called BCO algorithm based on their herding behavior. The main features of the proposed algorithm are as follows.

- New Swarm based algorithm on Border Collie dogs - Imitating the herding behavior of Border Collie dogs, a new swarm based algorithm has been proposed. To the best of our knowledge, no metaheuristic has been developed so far by mimicking the intelligent behavior of Border Collie dogs.
- Exploration and Exploitation mechanism - The proposed algorithm is designed in such a way that, both exploration and exploitation of the search space can be achieved using the same equations. Proper tuning between exploration and exploitation has a great influence in finding optimal results for metaheuristics. In the proposed algorithm, these two parameters have been efficiently balanced to get optimum results.
- Feedback implementation - Negative and positive feedbacks are two inherent parts of a swarm. Three different herding techniques of the Border Collie dogs are used to achieve the effective feedbacks, that in turn help to find effective results. Negative feedback is achieved by introducing the eyeing mechanism of the Border Collie dogs. Positive feedback is attained by means of gathering and stalking behavior of the dogs.
- Ability to recover from local optima - The eyeing mechanism introduced in the BCO algorithm also serves as an important tool to rescue it from getting stuck into local optima.
- Less Parameters - The algorithm is designed by exploiting mainly two independent parameters.
- Easy Implementation - The algorithm is easy to implement and keeps track of the best solution. These are inherent properties of swarm intelligence.

FIGURE 2. A Border Collie dog [95].

The rest of the paper is organized in the following manner. Section II presents the proposed work comprising biological inspiration, mathematical modeling and the algorithm in details. In Section III, the experimental results and analysis are presented. Section IV draws the conclusion of the paper and provides an insight into the future directions of research.

II. PROPOSED METHODOLOGY

In this section, the biological inspiration of the proposed method is discussed. Thereafter, a mathematical model is drawn and the flow of the algorithm is discussed in details.

A. BIOLOGICAL INSPIRATION

Canis lupus familiaris or the Border Collie is an amazing breed of dog. They have been ranked as the number one dog, in terms of smartness by Stanley Coren in his book "The Intelligence of Dogs" [96]. He also pointed out that they have the ability to obey 95% of human commands. In [97], a study carried out on a nine year old Border Collie, named Rico, established that he could understand around 200 human commands and words. In [98], another Border Collie called Chaser, could understand nouns similar to a human child.

Border Collies, in general are referred to as highly energetic, medium sized, herding dogs. They are a cross between *old Roman dogs* and *Viking spitzes*, according to the American Kennel Club [94]. Both the breeds had been brought to Britain during invasions.

All Border Collies found today can be traced back to a common ancestor, a dog called the *Old Hemp* [99]. He was born to a black sheepdog named Meg and a tri-colored herding dog named Roy in September 1893, in West Woodburn, Northumberland. He was different in physical appearance than the present day Border Collie dogs. He was a tri-colored dog with very less fur. His owner and breeder, Adam Telfer was impressed with his intelligence and herding abilities. He was highly sought after as a stud dog and is said to have as many as 200 pups. He is the *foundation sire* of the Border Collie breed and is enlisted in the stud book of the International Sheepdog Society.

The origin of the word *Collie* is believed to have emanated from the Celtic language, which means *useful*. Another origin of the word is traced back to the colley sheep in the Scottish Highlands. They are noted for their black markings, and *colley* is an old Anglo-Saxon word for the color black. Hence it is believed that, the Border Collie was named based on the black markings on its coat.

In 1880's and 1890's, agriculture based countries like Australia, New Zealand, the United States, Canada, and Argentina exported these expert dogs from the British Isles. A descendant of Old Hemp was gifted to Queen Victoria by John Elliot, which was bred in Scotland.

They usually have double coats with straight furs. They are found in different colors viz., black with or without white, chocolate, blue, gold, tri-colors, sable, merle and others. Nowadays, they are bred more for companionship and can be very good pets. They are also excellent watchdogs.

Border Collies are the best herding dogs of all time and are extremely workaholic. Their ability to judge a situation and to take adaptive decisions has inspired us to develop a metaheuristic algorithm based on their behavior.

1) THE HERDING STYLE OF BORDER COLLIES

These brilliant dogs follow their master's command ardently, but what makes them more appealing is that, they can think and adapt themselves dynamically.

Border Collies adopt a different approach for herding. Instead of approaching from back, they herd sheep from sides and front. They mainly follow three herding techniques, as demonstrated in Fig. 3. The stalking and eyeing behaviors of real Border Collie dogs are presented in Fig. 4.

- *Gathering*: Border Collies control the sheep from sides and front. They tend to gather them and direct them towards the farm. This is known as *gathering*.

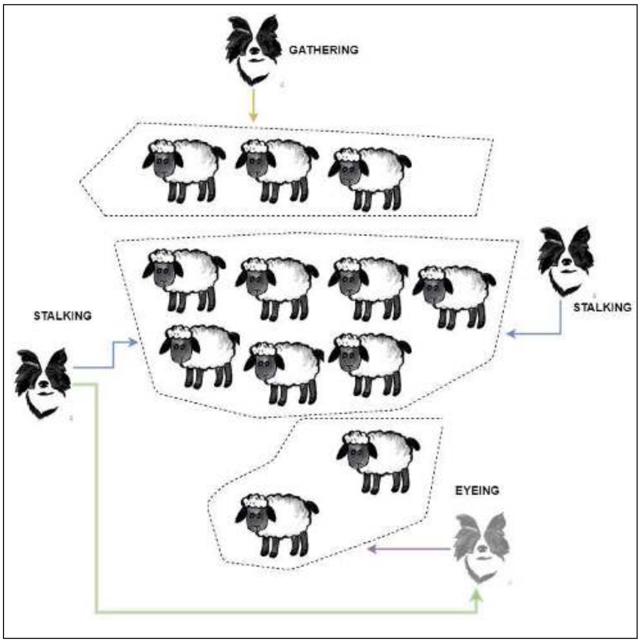


FIGURE 3. Herding techniques of Border Collie.

(a) Stalking [95]

(b) Eyeing [95]

FIGURE 4. Different herding behaviors of Border Collies.

- *Stalking*: Border Collies adopt few wolf-like movements when it comes to controlling the sheep. They crouch down lowering their heads, place their hindquarters high and put their tails down. This behavior is called *stalking*.

- *Eyeing*: Border Collies mimic the victim selection behavior of wolves. This is called *giving an eye* or *eyeing*. When sheep goes astray, these intelligent dogs stare them in the eye. This exerts psychological pressure on the flock to move in the correct direction.

B. MATHEMATICAL MODEL OF HERDING TECHNIQUES

In Subsection II-A, the main herding techniques of Border Collie have been explained. A mathematical model of the herding technique is presented in this subsection, along with an explanation of the algorithm.

In Border Collie Optimization, a population of three dogs and sheep is considered. In real life scenario, a single dog alone is sufficient to control the herd. However, as the search space can be vast for different optimization problems, hence three dogs are considered. A group consisting of three dogs and sheep is visualized while initiating the algorithm. The sheep go out for grazing in different directions and the dogs are responsible for bringing them back to the farm.

The locations of dogs and sheep are initialized with random variables. The dogs - *lead dog*, *left dog* and *right dog* are named so on the basis of their positions. The lead dog controls the herd from the front. The individual with best fitness (fit_f) is hence designated as the lead dog or dog in front of the herd, in every iteration. They are responsible for mainly *gathering*.

The individuals with the 2^{nd} and 3^{rd} best fitness values are chosen as left and right dogs. A tournament selection method is applied to choose the left and right dog. These dogs mainly participate in the *stalking* and *eyeing* of the herd. Their fitness values are referred to as (fit_{le}) and (fit_{ri}), respectively. The remaining population consists of sheep, whose fitness values are less than those of the dogs. The fitness of the sheep is referred to as (fit_s).

The optimum solution is the dogs leading the sheep to the farm. They travel from one point in the field to the farm. The distance covered and direction of the sheep and dogs are controlled by velocity, acceleration and time.

- *Velocity of Dogs*: The velocity of all the three dogs, at time $(t+1)$ is calculated using the following equations.

$$V_f(t+1) = \sqrt{V_f(t)^2 + 2 \times Acc_f(t) \times Pop_f(t)} \quad (1)$$

$$V_{ri}(t+1) = \sqrt{V_{ri}(t)^2 + 2 \times Acc_{ri}(t) \times Pop_{ri}(t)} \quad (2)$$

$$V_{le}(t+1) = \sqrt{V_{le}(t)^2 + 2 \times Acc_{le}(t) \times Pop_{le}(t)} \quad (3)$$

Equations (1), (2) and (3), $V_f(t+1)$, $V_{ri}(t+1)$ and $V_{le}(t+1)$ stand for velocity at time $(t+1)$ for lead, right and left dogs, respectively. Similarly, $V_f(t)$, $V_{ri}(t)$ and $V_{le}(t)$ stand for velocity at time (t) for lead, right and left dogs. $Acc_f(t)$, $Acc_{ri}(t)$ and $Acc_{le}(t)$ stand for acceleration at time (t) for the lead dog, right dog and left dog, respectively. $Pop_f(t)$, $Pop_{ri}(t)$ and $Pop_{le}(t)$ are the positions of the lead dog, right dog and left dog at time (t), respectively. eqnarray (1) updates the velocity of the lead dog. Equations (2) and (3) update the velocities of the right and left dogs, respectively.

- *Velocity of Sheep*: The velocity of the sheep is updated using the three herding techniques.
  - *Gathering*: The sheep which are nearer to the lead dog, move in the direction of the lead dog. Hence, these sheep are only gathered. They are chosen based on their fitness values.

$$D_g = (fit_f - fit_s) - ((\frac{fit_{le} + fit_{ri}}{2}) - fit_s) \quad (4)$$

In (4), if the value of D_g is positive, it indicates that the sheep is nearer to the lead dog. In this case the velocity of the sheep is updated using the following equation.

$$V_{sg}(t+1) = \sqrt{V_f(t+1)^2 + 2 \times Acc_f(t) \times Pop_{sg}(t)} \quad (5)$$

In (5), the velocity of the sheep, V_{sg} is directly influenced by the velocity of the lead dog at time $(t+1)$ and acceleration of the lead dog, at time (t). Pop_{sg} is the present location of the sheep to be gathered.

  - *Stalking*: The sheep which are nearer to the left and right dogs, need to be stalked from the sides to keep them on track. These sheep are those whose D_g values are found to be negative. The velocity of these sheep are more influenced by the velocities of the left and right dogs. The equations for the velocity updation of the stalked sheep are presented below.

$$v_{ri} = \sqrt{(V_{ri}(t+1)tan(\theta_1))^2 + 2 \times Acc_{ri}(t) \times Pop_{ri}(t)} \quad (6)$$

$$v_{le} = \sqrt{(V_{le}(t+1)tan(\theta_2))^2 + 2 \times Acc_{le}(t) \times Pop_{le}(t)} \quad (7)$$

$$V_{ss}(t+1) = \frac{v_{le} + v_{ri}}{2} \quad (8)$$

In (8), the velocity of the stalked sheep, V_{ss} depends on the velocities of the left and right dogs. As the dogs guide the sheep from the sides, hence the *tangent* of the random traversing angles, θ_1 and θ_2 are taken. The value of θ_1 varies from $(1 - 89)$ degrees and that of θ_2 varies from $(91 - 179)$ degrees. The values of θ_1 and θ_2 are chosen randomly.

  - *Eyeing*: The sheep which are totally astray are the ones which need eyeing. Eyeing is implemented, when in consecutive iterations, the fitness of an individual does not improve. In this case, the dog with the least fitness is assumed to go behind the sheep and give them an eye. Hence they are assumed to undergo retardation, which can be presented by the below mentioned equations.

$$V_{se}(t+1) = \sqrt{V_{le}(t+1)^2 - 2 \times Acc_{le}(t) \times Pop_{le}(t)} \quad (9)$$

$$V_{se}(t+1)=\sqrt{V_{ri}(t+1)^2-2\times Acc_{ri}(t)\times Pop_{se}(t)} \quad (10)$$

In (9), $V_{le}(t+1)$ and $Acc_{le}(t)$ are the velocity and acceleration of the left dog, when it has the worst fitness among the three dogs. In (10), $V_{ri}(t+1)$ and $Acc_{ri}(t)$ are the velocity, acceleration of the right dog, when it has the least fitness among the three dogs. Pop_{se} is the present location of the sheep to be gathered. The dog with least fitness is considered because it is assumed that this dog is closest to the sheep.

- *Acceleration of Dogs and Sheep*: The equation for acceleration updation is derived from the most commonly used equation in physics and is mentioned below.

$$Acc_i(t+1)=\frac{(V_i(t+1)-V_i(t))}{Time_i(t)} \quad (11)$$

The acceleration of all the dogs and sheep viz., $Acc_f(t+1), Acc_{ri}(t+1), Acc_{ri}(t+1), Acc_{sg}(t+1), Acc_{ss}(t+1)$ and $Acc_{se}(t)$ are updated using (11). $i \in \{f, le, ri, sg, ss \text{ to } se\}$.

- *Time of Dogs and Sheep*: The time (T) of traversal is updated for each individual using the following equation.

$$Time_i(t+1)=\text{Avg}\sum_{i=1}^{d}\frac{(V_i(t+1)-V_i(t))}{Acc_i(t+1)} \quad (12)$$

where, the average time of traversal of each individual is of dimension (d).

- *Population Updation of Dogs*: The positions of the dogs are updated using the basic physics equation of displacement.

$$Pop_f(t+1)=V_f(t+1)\times Time_f(t+1) +\frac{1}{2}Acc_f(t+1)\times Time_f(t+1)^2 \quad (13)$$

$$Pop_{le}(t+1)=V_{le}(t+1)\times Time_{le}(t+1) +\frac{1}{2}Acc_{le}(t+1)\times Time_{le}(t+1)^2 \quad (14)$$

$$Pop_{ri}(t+1)=V_{ri}(t+1)\times Time_{ri}(t+1) +\frac{1}{2}Acc_{ri}(t+1)\times Time_{ri}(t+1)^2 \quad (15)$$

Equation (13) updates the position of the lead dog, whereas the positions of the left and right dogs are updated using (14) and (15).

- *Population Updation of Sheep*: The positions of the sheep are updated using the following equations, when the sheep belong to the gathering and stalking groups.

$$Pop_{sg}(t+1)=V_{sg}(t+1)\times Time_{sg}(t+1) +\frac{1}{2}Acc_{sg}(t+1)\times Time_{sg}(t+1)^2 \quad (16)$$

$$Pop_{ss}(t+1)=V_{ss}(t+1)\times Time_{ss}(t+1) -\frac{1}{2}Acc_{ss}(t+1)\times Time_{ss}(t+1)^2 \quad (17)$$

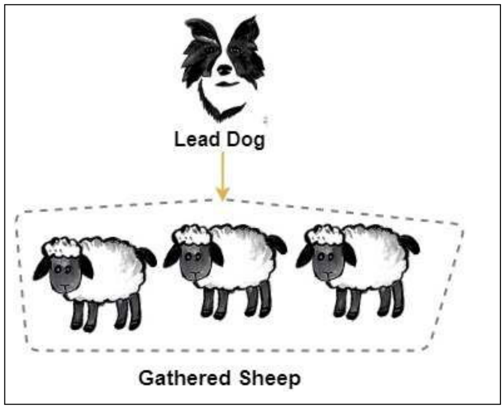


FIGURE 5. Gathering of sheep by Lead Dog.

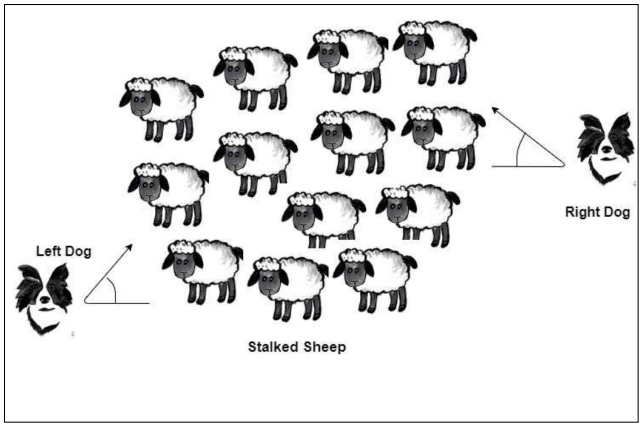


FIGURE 6. Stalking of sheep by Left and Right Dogs.

In case of sheep which are eyed, the below mentioned equation is used.

$$Pop_{se}(t+1)=V_{se}(t+1)\times Time_{se}(t+1) -\frac{1}{2}Acc_{se}(t+1)\times Time_{se}(t+1)^2 \quad (18)$$

The important symbols used and their meanings are presented in Table 6. Figs. 5, 6 and 7 show the different herding techniques.

C. ALGORITHM

The initialization process and the different steps for the proposed optimization algorithm are shown in Algorithm 1.

Dependency of Parameters: The BCO algorithm is designed with the help of mainly four parameters. The updation of the states depends on mainly two independent parameters viz., velocity and time. The other two parameters, acceleration and population are dependent parameters, which can be easily derived from the aforesaid independent parameters. From (11), we derive that $Acc_i(t+1)$ can be obtained if velocity and time are known. Similarly, by substituting the

TABLE 6. Important symbols, their purpose and relevant Equation nos. used in BCO algorithm.

Symbols	Purpose	Equation No.
fit_f	Fitness of Lead dog at time (t)	–
$V_f(t+1)$	Velocity of Lead dog at time $(t+1)$	(1)
$Acc_f(t+1)$	Acceleration of Lead dog at time $(t+1)$	(11)
$Pop_f(t+1)$	Location of Lead dog at $(t+1)$	(13)
$Time_f(t+1)$	Time required by Lead dog to move to $Pop_f(t+1)$	(12)
fit_{ri}	Fitness of Right dog at time (t)	–
$V_{ri}(t+1)$	Velocity of Right dog at time $(t+1)$	(2)
$Acc_{ri}(t+1)$	Acceleration of Right dog at time $(t+1)$	(11)
$Pop_{ri}(t+1)$	Location of Right dog at $(t+1)$	(15)
$Time_{ri}(t+1)$	Time required by Right dog to move to $Pop_f(t+1)$	(12)
fit_{le}	Fitness of Left dog at time (t)	–
$V_{le}(t+1)$	Velocity of Left dog at time $(t+1)$	(3)
$Acc_{le}(t+1)$	Acceleration of Left dog at time $(t+1)$	(11)
$Pop_{le}(t+1)$	Location of Left dog at $(t+1)$	(14)
$Time_{le}(t+1)$	Time required by Left dog to move to $Pop_f(t+1)$	(12)
$V_{sg}(t+1)$	Velocity of gathered sheep at time $(t+1)$	(5)
$Acc_{sg}(t+1)$	Acceleration of gathered sheep at time $(t+1)$	(11)
$Pop_{sg}(t+1)$	Location of gathered sheep at $(t+1)$	(16)
$Time_{sg}(t+1)$	Time required by gathered sheep to move to $Pop_f(t+1)$	(12)
$V_{ss}(t+1)$	Velocity of stalked sheep at time $(t+1)$	(8)
$Acc_{ss}(t+1)$	Acceleration of stalked sheep at time $(t+1)$	(11)
$Pop_{ss}(t+1)$	Location of stalked sheep at $(t+1)$	(17)
$Time_{ss}(t+1)$	Time required by stalked sheep to move to $Pop_f(t+1)$	(12)
$V_{se}(t+1)$	Velocity of eyed sheep at time $(t+1)$	(9) or (10)
$Acc_{se}(t+1)$	Acceleration of eyed sheep at time $(t+1)$	(11)
$Pop_{se}(t+1)$	Location of eyed sheep at $(t+1)$	(18)
$Time_{se}(t+1)$	Time required by eyed sheep to move to $Pop_f(t+1)$	(12)
fit_s	Fitness of sheep at time (t)	–
D_g	Compares the fitness of sheep to fitness of Lead dog and mean fitness of Left and Right dogs	(4)
θ_1	Random angle between Right dog and stalked sheep	(6)
θ_2	Random angle between Left dog and stalked sheep	(7)

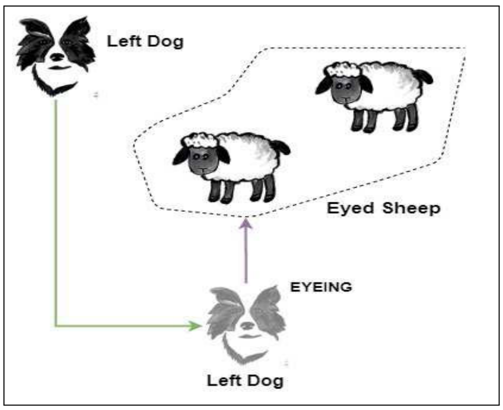


FIGURE 7. Eyeing of sheep by Left Dog.

value of $Acc_i(t+1)$ in (13), we obtain the following equation.

$$Pop_f(t+1) = V_f(t+1) \times Time_f(t+1) + \frac{1}{2}\frac{(V_f(t+1) - V_f(t))}{Time_i(t)} \times Time_f(t+1)^2 \quad (19)$$

or,

$$Pop_f(t+1) = V_f(t+1) \times Time_f(t+1) + \frac{1}{2}(V_f(t+1) - V_f(t)) \times Time_f(t+1) \quad (20)$$

The populations of the left dog, right dog, gathered sheep, stalked sheep and eyed sheep can be obtained in a similar manner, by substituting the value of $Acc_i(t+1)$ in (14), (15), (16), (17) and (18), respectively.

D. AVOIDANCE FROM GETTING STUCK IN LOCAL OPTIMA

In Algorithm 1, at every iteration, the fitness of each sheep is checked to determine whether it is stuck in local optima or not. If the fitness of the sheep doesn't improve in five consecutive steps, the sheep is considered to be stuck in local optima. Then this sheep is eyed by the dog to get it back on track.

E. EXPLORATION AND EXPLOITATION OF BCO ALGORITHM

Exploration and exploitation of the search space play an important role in achieving optimal solutions [72].

Algorithm 1 Border Collie Optimization

Initialize
$Pop_t \rightarrow$ A random population of n individuals having d dimensions each, 3 dogs and $(n-3)$ sheep;
$Acc_t \rightarrow$ Random acceleration for each of the n individuals having d dimensions;
$Time_t \rightarrow$ Random time for each of the n individuals;
$V_t \rightarrow$ Zero velocity for n individuals having d dimensions;
$k = 0$;
while $t < Max\_Iterations$ **do**
  $Eyeing = 0$
  fit_t = Calculate fitness of n individuals
  if $fit_t < fit_{t-1}$ **then**
    $k = k + 1$
  end if
  if $k = 5$ **then**
    $Eyeing = 1$
    $k = 0$
  end if
  LeadDog = Individual with best fitness (fit_f)
  $R = Random\ Number[2, 3]$
  if $R = 2$ **then**
    RightDog = Individual with 2^{nd} best fitness (fit_{ri})
    LeftDog = Individual with 3^{rd} best fitness (fit_{le})
  else
    LeftDog = Individual with 2^{nd} best fitness (fit_{le})
    RightDog = Individual with 3^{rd} best fitness (fit_{ri})
  end if
  Sheep = Rest of the individuals excluding top three (fit_s)
  Update velocity of dogs (using (1), (2) & (3))
  while $i > 3$ *and* $i <= n$ **do**
    if $Eyeing \equiv 1$ **then**
      Update velocity of sheep (using (9))
    else
      if $D_g > 0$ **then**
        Update velocity of sheep (using (5))
      else
        Update velocity of sheep (using (8))
      end if
    end if
  end while
  Update Acceleration of n individuals (using (11))
  Update Time of n individuals (using (12))
  Update Population of Dogs (using (13), (14) & (15))
  while $i > 3$ *and* $i <= n$ **do**
    if $Eyeing \equiv 1$ **then**
      Update Population of sheep (using (18))
    else
      Update Population of sheep (using (16) & (17))
    end if
  end while
end while

The algorithms having the capability to balance between the two, have more chance of being successful in not getting stuck in local optima. Exploration stresses on finding potential solution regions in the search space. The movement of the three dogs viz., *lead dog*, *right dog* and *left dog* controls the exploration capability of the BCO algorithm. They move in different directions and are independent of each others' movement. Hence, they are capable of finding the promising regions in the search space.

On the other hand, exploitation means to focus on refining the search results. The movements of the *gathered sheep* and *stalked sheep* are directly influenced by the three dogs. Hence, they concentrate on finding more optimal solutions in that part of the search space where the dogs are present. Moreover, if the BCO algorithm gets stuck in local optima, the "*eyed sheep*" rescues the algorithm by applying the concept of retardation. Figs. 8 and 9 graphically explain the three herding behaviors of the Border Collie dogs. The farms presented in the images are assumed to be the optima.

F. COMPLEXITY ANALYSIS

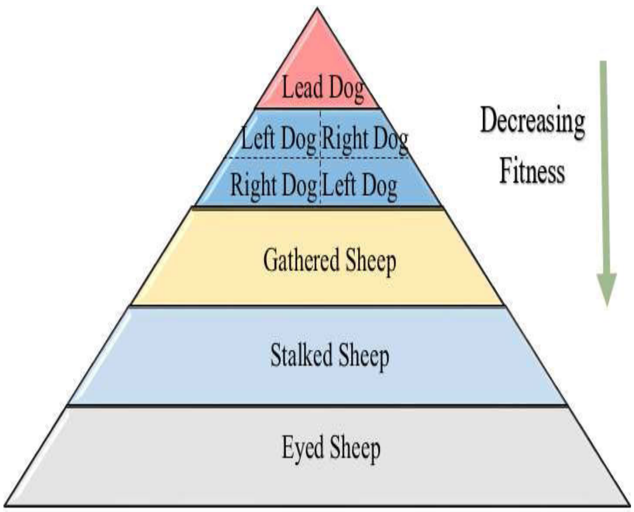
The worst case time complexity of the proposed BCO algorithm is given below.

- In BCO algorithm, the time complexity for producing the initial population is $O(n \times d)$. Here, n is the size of the population and d is the dimension of each of them.
- The fitness of each individual is calculated using different benchmark functions. The time complexity to compute fitness for each generation is $O(n)$.
- The time complexity of velocity updation at every step is $O(3)$.
- The time complexity for updation of time is $O(n)$.
- The algorithm is run for $Max\_Iterations$ number of times. Hence, the time complexity becomes $O(n \times d \times Max\_Iterations)$.

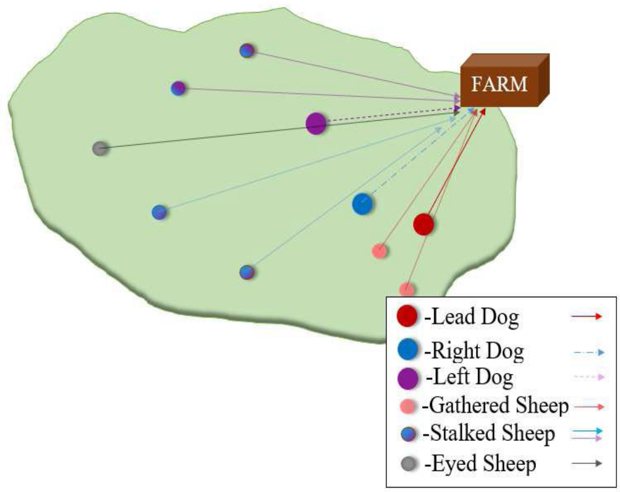
From the above discussion, we can thus state that the overall worst case time complexity for the proposed BCO algorithm is $O(n \times d \times Max\_Iterations)$.

III. RESULTS AND DISCUSSIONS

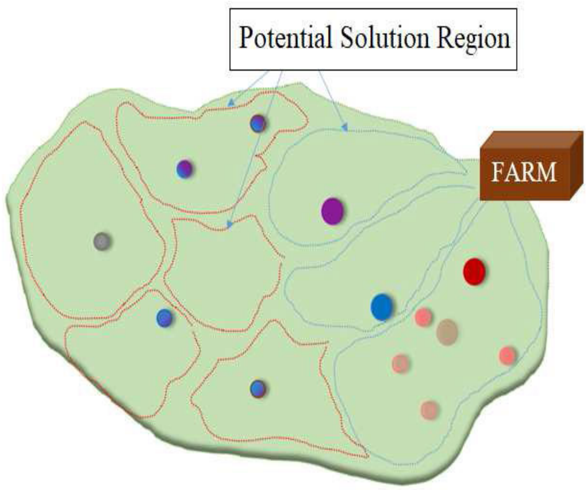
In this section, the results of the BCO algorithm and other associated comparable algorithms are presented. The entire process has been implemented in MATLAB 2019a, on Intel (R) Core (TM) i7 8700 Processor with Windows 10 environment. Nineteen conventional benchmark functions [5] [100] (BF1) and sixteen other functions, taken from CEC'17 benchmark suite [101] (BF2) are used for experimental purpose. The BCO algorithm is compared with seven state-of-the-art metaheuristic algorithms (ACO [3], DE [16], GA [11], GWO [5], HHO [58], PSO [4], WOA [52]) to establish its effectiveness. These algorithms are chosen in such a manner that their distinct characteristics and different advantages help to find out the merits of the proposed BCO algorithm.



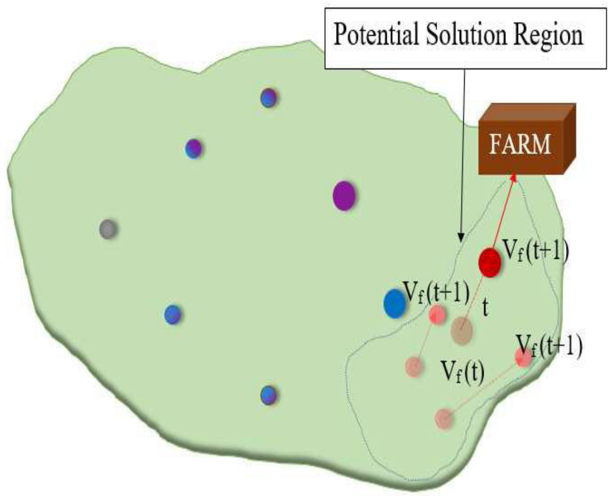
(a) Fitness of Dogs and Sheep



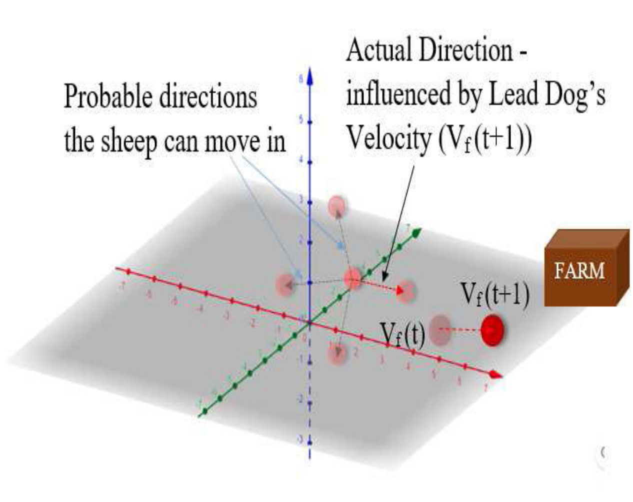
(b) Initial Positions of Dogs and Sheep



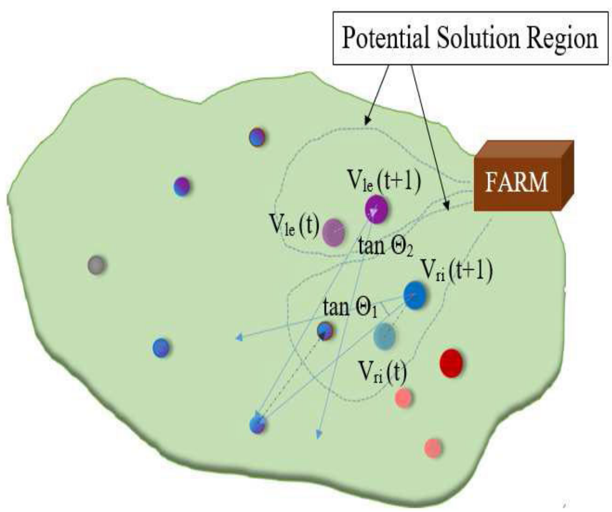
(c) Potential Solution Regions



(d) Lead Dog and Gathering of Sheep

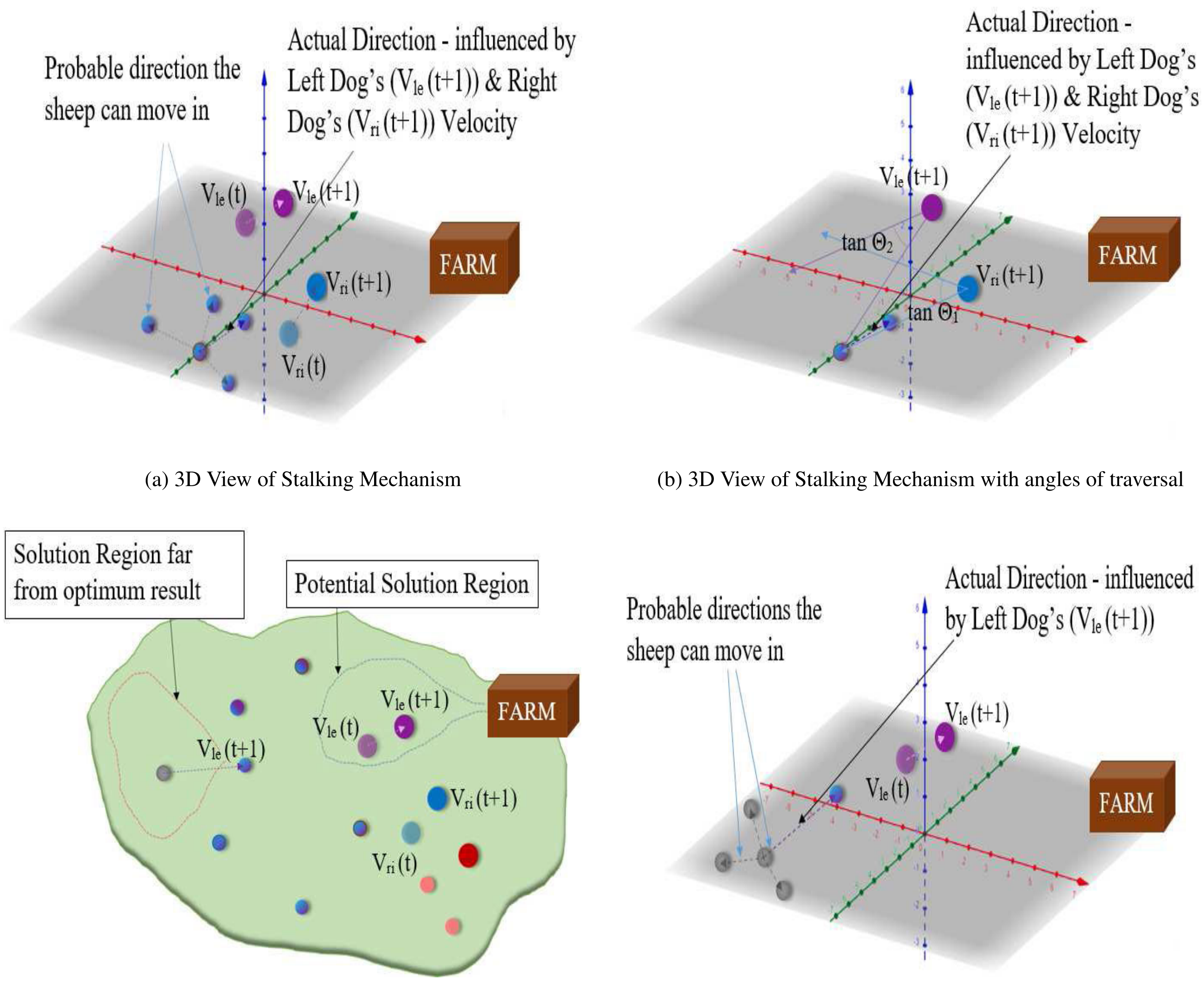


(e) 3D View of Gathering Mechanism



(f) Right and Left Dogs and Stalking of Sheep

FIGURE 8. **(a) Hierarchy of fitness of Border Collie Dogs and Sheep, (b) & (c) Dogs' and sheep's initial positions and potential solution regions, (c) - (f) Herding behaviors and 3D View of Herding behaviors.**



(a) 3D View of Stalking Mechanism

(b) 3D View of Stalking Mechanism with angles of traversal

(c) Eyeing Mechanism

(d) 3D View of Eyeing Mechanism

FIGURE 9. (a) - (d) Herding behaviors and 3D View of Herding behaviors.

To maintain an unbiased approach, all competitive algorithms need to be evaluated using either equal number of fitness evaluations or equal processing time [59]. We have adopted the first aproach to conduct the experiments for all participating algorithms. All participating algorithm are run 50 number of times, having 200 iterations each to ensure a fair comparison. The results of all the eight algorithms are compared on the basis of different statistical tests like mean and standard deviation, to ensure fair analysis. To perceive the overall performance of the algorithms, two popular statistical analysis tests, called Friedman Test [102], [103] and Kruskal - Wallis Test [104] are also conducted among them.

The parameters of ACO [3], DE [16], GWO [5], HHO [58] and WOA [52] are calibrated as mentioned in the original papers. The parameter tuning for GA [11] and PSO [4] is adopted from [65] and [72], respectively for conducting the experiments. The comparable algorithms are chosen in such a way that they possess diverse characteristics that can help us to judge the acceptability of the proposed algorithm based on multiple features. ACO [3], DE [16], GA [11] and PSO [4] are all popular metaheuristics, that usually produce effective results. The other three algorithms viz., GWO [5], HHO [58] and WOA [52] are relatively new popular metaheuristics. The individual features of these algorithms are already discussed in details and presented in Table 5.

A. SENSITIVITY ANALYSIS OF BCO

Every metaheuristic algorithm has a number of uncertain parameters. Their evaluation, accuracy, limitations and scope need to be extensively studied. These uncertainties can be addressed by performing a sensitivity analysis test [105]. The sensitivity analysis test can be conducted by studying one

TABLE 7. Parameters of the compared algorithms.

S.No.	Algorithms	Parameters
1.	ACO [3]	Iterations-200, Runs-50 Priory defined number-0.5
2.	DE [16]	Iterations-200, Runs-50 Crossover Probability-0.95 Mutation Probability-0.05
3.	GA [11]	Iterations-200, Runs-50 Crossover Probability-0.9 Mutation Probability-0.1
4.	GWO [5]	Iterations-200, Runs-50 a value linearly decreases from 2 to 0, with the iterations
5.	HHO [58]	Iterations-200, Runs-50 position vector of hawks (r1, r2, r3, r4) in $[0, 1]$ Equal chance (q) is a random number in $[0, 1]$
6.	PSO [4]	Iterations-200, Runs-50 Inertia Factor(W)-0.9 Cognitive Learning Rate(c_1, c_2)-2, 2
7.	WOA [52]	Iterations-200, Runs-50 a value linearly decreases from 2 to 0, with the iterations r is a random number in $[0, 1]$ l is a random number in $[-1, 1]$ b=1

TABLE 8. Parameter sensitivity analysis result.

Parameters	Range	1^{st} Order Effects [106]	Total Effects [107]	No. Of Dogs	Eyeing Step
Velocity	$\mathbf{(0-1)}$	**0.5322**	1.1363	3	5
	$(1-2)$	0.0751	1.1744	3	5
	$(2-5)$	-1.3402	1.5172	3	5
	$(5-10)$	-0.7826	1.5794	3	5
Velocity (Varying Dogs)	$(0-1)$	**0.5322**	1.1363	**3**	5
	$(0-1)$	0.2799	0.5793	1	5
	$(0-1)$	-0.1028	1.0921	5	5
Velocity (Varying Eyeing Steps)	$(0-1)$	**0.5322**	1.1363	3	**5**
	$(0-1)$	0.1697	1.1060	3	1
	$(0-1)$	0.2675	0.5013	3	3
	$(0-1)$	0.3554	0.6300	3	10
Time	$\mathbf{(0-1)}$	**0.8787**	1.3752	3	5
	$(1-2)$	-0.0515	0.6584	3	5
	$(2-5)$	-0.1346	2.0100	3	5
	$(5-10)$	0.3696	0.6859	3	5
Pop	10	-1.5291	1.7745	3	5
	20	0.0630	0.4658	3	5
	30	**0.4329**	1.2986	3	5
	50	-1.7585	1.5076	3	5
	100	0.3777	0.9562	3	5
Max_Iterations	50	-0.6075	0.8289	3	5
	100	0.2407	0.7258	3	5
	200	**0.7412**	2.1982	3	5
	500	0.6243	0.7977	3	5
	1000	0.2091	0.2805	3	5

parameter at a time *(Local Method)* or by evaluating multiple parameters at a time *(Global Method)*.

To check the robustness of the BCO algorithm, a global sensitivity analysis test is conducted on the independent parameters using the Sobol's sensitivity indices [106]. The sensitivity in this method is measured in terms of conditional variances, as given by

$$St_i = \frac{V_{x_i}\left(E_{x_{\sim i}}(F|x_i)\right)}{V(F)} \tag{21}$$

where, St_i is the *first-order index*. It measures the direct contribution of each input factor to the output variance. F is the output of the model, x_i represents the i^{th} input parameter, E stands for the expected value and V is the variance. It can be noted that higher indices value indicates better effectiveness on the output. The total indices [107] are interpreted as follows

$$St_{Tot_i} = \frac{E_{x_{\sim i}}\left(V_{x_i}(F|x_{\sim i})\right)}{V(F)} \tag{22}$$

The total indices measure the influence of the i^{th} parameter on the output. If the total indices value of a parameter is zero, it indicates that the parameter has no influence on the output. A standard benchmark function for sensitivity calculation, called the Sobol g-function [105] is used for this purpose. Only the independent parameters viz., velocity and time are considered for the purpose of sensitivity analysis along with the population size and number of iterations. The parameters are compared in Table 8. The first order sensitivity indices and total indices are also reported in this table. The best values obtained are marked in boldfaced and are considered as the final parameters.

In this paper, two phenomena viz., population size of dogs and steps to trigger eyeing are analyzed using the Sobol's method. The number of dogs is taken as three. In Table 8, it is found that increasing or decreasing the number of dogs, reduces the efficiency of the algorithm. The eyeing mechanism is optimized by varying the number of steps. Optimum results are obtained when the number of steps for eyeing is taken as 5. Increasing or decreasing the number of steps for eyeing, reduces the efficiency of the algorithm. The velocity parameter is used to tune both the processes.

B. ANALYSIS OF BF1

In this paper, nineteen traditional benchmark functions [5], [100] are used for experimental purpose. The details of these functions are presented in the supplementary document. Table 7 presents the individual parameter settings of the compared algorithms. It can be noted that $F_1 - F_7$ are *Unimodal benchmark* functions. Functions $F_8 - F_{13}$ are *Multimodal benchmark* functions and $F_{14} - F_{19}$ are *Fixed-dimension multimodal benchmark* functions. The 2-D versions of some of these functions are plotted in Fig. 10.

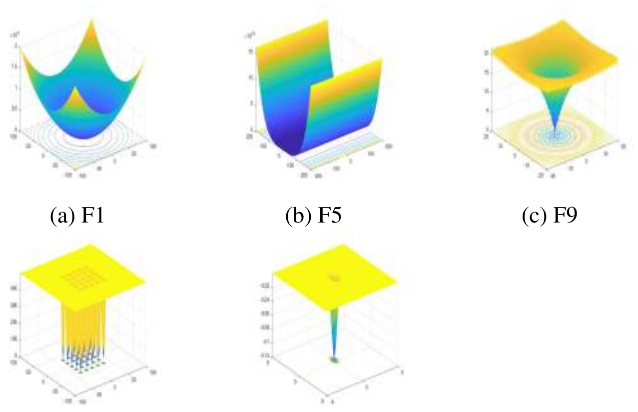
The means, standard deviations (STD), minimum values (Min) and maximum values (Max) obtained by the functions and the minimum time taken to converge are reported in Tables 9 and 10. Kruskal-Wallis test [104] is applied to the results obtained from ACO [3], BCO, DE [16], GA [11], GWO [5], HHO [58], PSO [4], WOA [52]. This statistical test is carried out with 1% significance level for finding the p value. Lower p value indicates higher significance. A p value less than 0.05 represents *"significant"*, whereas less than 0.001 represents *"highly significant"*. The null hypothesis, that all values have same distribution across all the methods, stands rejected. Three representative box plots for the Kruskal-Wallis tests [104] are given in Fig. 11 and rest of them are provided in the supplementary document. The p values are recorded in Table 11.

TABLE 9. Results of BF1 [5], [100].

Fn.	Results	BCO	ACO [3]	DE [16]	GA [11]	GWO [5]	HHO [58]	PSO [4]	WOA [52]
F_1	Mean	2.0928e-58	0.6865	0.1594	0.1215	8.7974e-09	7.8464e-42	2.5120e+04	1.6795e-26
	STD	1.3616e-57	0.2055	0.1307	0.0926	7.8904e-09	3.7385e-41	3.7996e+03	1.0894e-25
	Min	**3.7200e-139**	0.1774	0.0026	0.0023	5.5000e-10	2.2900e-56	1.7194e+04	9.3700e-36
	Max	9.6100e-57	1.1064	0.5430	0.4239	3.3100e-08	2.5200e-40	3.6666e+04	7.7100e-25
	Time	0.00914	8.3400	0.0054	0.0030	0.0313	0.0265	0.0011	0.0121
F_2	Mean	4.0746e-30	1.9961	1.5333	1.0115	7.1530e-06	2.1364e-22	4.9682e+03	1.6111e-19
	STD	1.9532e-29	0.6610	0.7384	0.4304	3.3021e-06	4.8463e-22	3.2034e+04	5.6800e-19
	Min	**3.2400e-53**	0.8265	0.4643	0.0810	1.9500e-06	3.4900e-28	49.0195	1.2700e-23
	Max	1.1100e-28	4.8314	3.9380	1.9236	1.9500e-05	1.9900e-21	2.2669e+05	3.9400e-18
	Time	0.0129	9.0900e-04	0.0032	0.0013	0.0321	0.0274	5.0400e-04	0.0130
F_3	Mean	3.7115e-56	33.5947	28.9469	10.9181	3.2889	3.0943e-26	4.0026e+04	8.3276e+04
	STD	2.1713e-55	18.1414	21.6612	8.4781	3.8946	1.9611e-25	1.3333e+04	1.9343e+04
	Min	**4.0600e-114**	6.6541	3.8184	0.0395	0.1045	3.4000e-45	1.9093e+04	5.0213e+04
	Max	1.5100e-54	93.9601	82.9861	27.7611	15.1990	1.3800e-24	8.1739e+04	1.3649e+05
	Time	0.0275	0.0028	0.0079	0.0043	0.0723	0.1262	0.0020	0.0536
F_4	Mean	4.6042e-31	0.4913	0.0940	0.0950	0.0324	3.8349e-22	61.0419	54.3299
	STD	2.4794e-30	0.0525	0.0460	0.0433	0.0213	1.6379e-21	5.0428	24.6806
	Min	**4.8200e-61**	0.3450	0.0202	0.0187	0.0110	9.1200e-29	50.6114	2.1447
	Max	1.7100e-29	0.6498	0.2256	0.2008	0.1295	1.1300e-20	72.0937	90.8013
	Time	0.0320	0.0012	0.0110	0.0044	0.0313	0.0288	6.1500e-04	0.0104
F_5	Mean	28.8638	94.9885	0.0075	0	27.7533	0.0864	4.0958e+07	28.6349
	STD	0.0624	16.5742	0.0404	0	0.7946	0.1097	1.2862e+07	0.2225
	Min	28.6955	58.3350	**0**	**0**	26.2338	3.3000e-06	7.3394e+07	28.0176
	Max	28.9334	131.2774	0.2838	0	28.8420	0.4820	1.9449e+07	28.8213
	Time	0.0295	8.0700e-04	0.0025	4.5000e-04	0.0379	0.0490	9.3800e-04	0.0154
F_6	Mean	7.4981	10.3380	9.1591	8.7804	1.4065	9.7091e-04	2.4097e+04	1.2543
	STD	0.0133	0.7362	1.0488	0.5902	0.5899	0.0015	4.2658e+03	0.4077
	Min	7.4057	8.8316	7.6741	7.8641	0.3336	**7.8500e-08**	1.5650e+04	0.3826
	Max	7.5000	13.5281	12.9399	10.2276	3.2613	0.0085	3.4457e+04	2.2929
	Time	0.0062	8.0900e-04	0.0129	0.0073	0.0324	0.0362	5.7400e-04	0.0100
F_7	Mean	5.1533e-04	1.7939	0.1597	0.1239	0.0062	3.7104e-04	21.9402	0.0105
	STD	6.4260e-04	0.6713	0.1152	0.1167	0.0035	3.2795e-04	7.2859	0.0102
	Min	**4.8300e-06**	0.5518	0.0107	0.0043	0.0019	6.8610e-04	6.8179	8.1000e-05
	Max	0.0032	3.3288	0.4979	0.4944	0.0177	0.0014	38.2577	0.0396
	Time	0.0203	0.0013	0.0091	0.0030	0.0213	0.0135	4.2200e-04	0.0051
F_8	Mean	-4.0555e+03	-21.2780	-2.2118e+03	-2.1561e+03	-5.9142e+03	-1.2526e+04	-3.0284e+03	-9.7082e+03
	STD	394.2554	0.8149	513.8924	459.3943	920.9345	220.2211	560.9542	1.7885e+03
	Min	-5.3408e+03	-23.7606	-3.5596e+03	-3.0819e+03	-7.8856e+03	**-1.2569e+04**	-4.5224e+03	-1.2569e+04
	Max	-3.4779e+03	-19.7029	-1.2146e+03	-1.4042e+03	-3.0695e+03	-1.1059e+04	-2.2519e+03	-5.6830e+03
	Time	0.0122	0.0121	0.0011	0.0028	0.0020	0.0398	8.8000e-05	0.0127
F_9	Mean	8.8800e-16	1.0861	1.2603	0.5544	1.7523e-05	8.8800e-16	17.0662	2.6822e-14
	STD	2.9731e-31	0.1521	0.5261	0.2933	7.4192e-06	9.9609e-31	0.4474	2.1455e-14
	Min	**8.8800e-16**	0.6271	0.0996	0.0530	6.3600e-06	8.8800e-16	17.0662	4.4400e-15
	Max	8.8800e-16	1.4440	2.5278	1.1309	3.9200e-05	8.8800e-16	19.1058	8.6200e-14
	Time	0.0029	8.7600e-04	0.0070	0.0040	0.0350	0.0203	3.1300e-04	0.0124
F_{10}	Mean	1.6568	1.9417	1.9270	1.8149	0.0969	5.6777e-05	4.8844e+07	0.0815
	STD	0.0598	0.0656	0.1296	0.1431	0.0549	8.7786e-05	2.7480e+07	0.0667
	Min	1.2781	1.8176	1.6825	1.2267	0.0226	**3.9600e-08**	6.7153e+05	0.0176
	Max	1.6690	2.0998	2.2183	2.0241	0.2886	4.6432e-04	1.3270e+08	0.3330
	Time	0.0056	0.0021	0.0140	0.0019	0.0974	0.1704	0.0030	0.0627
F_{11}	Mean	2.0807	0.2350	1.2064e-05	1.3500e-32	1.0727	6.5950e-04	1.4143e+08	1.0566
	STD	0.3071	0.0705	3.4030e-05	8.2941e-48	0.2876	8.4547e-04	5.2552e+07	0.3997
	Min	1.3697	0.0890	**1.3500e-32**	1.3500e-32	0.4625	8.6800e-07	5.4110e+07	0.3574
	Max	2.7318	0.3818	1.6870e-04	1.3500e-32	1.6827	0.0035	3.0141e+08	2.0658
	Time	0.0165	0.0214	0.0020	0.0010	0.0958	0.1776	0.0017	0.0571
F_{12}	Mean	1.9702	12.6705	8.7512	10.4452	5.3734	2.3398	4.9637	4.0163
	STD	0.9299	7.1776e-15	3.6827	3.2749	3.9851	2.1450	4.0187	3.9244
	Min	**0.9980**	12.6705	1.0484	2.3278	0.9980	0.9980	0.9980	0.9980
	Max	4.4181	12.6705	12.6705	12.6705	12.6705	10.7632	21.2757	10.7632
	Time	0.0070	0.0014	0.0058	0.0816	0.0125	0.2291	0.0020	0.0857
F_{13}	Mean	0.0011	6.7514e-04	5.7366e-04	5.7623e-04	0.0043	3.6963e-04	0.0222	7.8839e-04
	STD	5.9279e-04	2.2089e-04	2.1459e-04	1.7673e-04	0.0076	5.2139e-05	0.0273	5.0368e-04
	Min	4.0203e-04	3.4777e-04	3.1660e-04	3.4776e-04	3.0868e-04	**3.0767e-04**	9.5050e-04	3.2001e-04
	Max	0.0039	0.0012	0.0013	0.0012	0.0204	4.8724e-04	0.1079	0.0022
	Time	0.0020	2.4700e-04	9.0200e-04	0.0022	0.0105	0.0263	3.3100e-04	0.0086
F_{14}	Mean	3.2334	600.2789	45.8894	36.3286	3.0002	3.0000	2.8277	3.5433
	STD	0.3278	1.4601	41.3273	31.2902	2.5023e-04	1.4291e-04	3.6343	3.8394
	Min	**3**	600	3.5559	3.2053	3.1	3.12	3.17	3.18
	Max	4.7652	609.7916	229.2187	141.6635	3.0012	3.0010	19.2141	30.1490
	Time	0.0015	3.5400e-04	0.0075	0.0011	0.0032	0.0206	4.0000e-05	0.0040
F_{15}	Mean	-3.8474	-3.8318	-3.7678	-3.7780	-3.8622	-3.8537	-3.7949	-3.8265
	STD	0.0106	0.0191	0.0287	0.0396	0.0014	0.0125	0.0682	0.0676
	Min	**-3.8630**	-3.8603	-3.8350	-3.8595	-3.8628	-3.8628	-3.8626	-3.8628
	Max	-3.8233	-3.7722	-3.7210	-3.7131	-3.8549	-3.8128	-3.5779	-3.5917
	Time	4.2100e-04	0.0013	0.0071	0.0177	0.0136	0.0384	1.6300e-04	0.0112
F_{16}	Mean	-2.9605	-2.8593	-2.8062	-2.8177	-3.2476	-2.9918	-2.9918	-3.2006
	STD	0.1117	0.1626	0.1091	0.0991	0.0766	0.1379	0.1379	0.1030
	Min	-3.1475	-3.1601	-3.0918	-3.0683	**-3.3220**	-3.2471	-3.2471	-3.3170
	Max	-2.7029	-2.2983	-2.6203	-2.5544	-3.0464	-2.6753	-2.6753	-2.9790
	Time	0.0010	2.4000e-04	2.9700e-04	7.1200e-04	0.0172	0.0391	0.0185	0.0108

TABLE 10. Results of BF1 [5], [100].

Fn.	Results	BCO	ACO [3]	DE [16]	GA [11]	GWO [5]	HHO [58]	PSO [4]	WOA [52]
F_{17}	**Mean**	-3.9379	-4.7618	-10.0522	-5.0552	-8.5443	-5.3106	-2.0686	-7.4574
	STD	0.6446	0.3084	0.7145	6.2804e-15	2.8053	1.1493	1.0658	2.7016
	Min	-5.9929	-5.0550	**-10.1532**	-5.0552	-10.1528	-10.0336	-5.7709	-10.1529
	Max	-2.7158	-3.7358	-5.1008	-5.0552	-2.6304	-4.8786	-0.7062	-2.5916
	Time	0.0012	0.0125	4.0500e-04	3.1600e-04	0.0179	0.0312	7.7000e-05	0.0128
F_{18}	**Mean**	-4.2673	-4.7393	-10.2973	-5.0877	-10.2277	-5.1166	-2.3442	-6.6721
	STD	0.9125	0.3684	0.7459	5.3832e-15	1.2108	0.6296	1.0122	2.8095
	Min	-7.1666	-5.0861	**-10.4028**	-5.0877	-10.4025	-9.2565	-5.7236	-10.3843
	Max	-2.8504	-3.6622	-5.1288	-5.0877	-1.8371	-3.6648	-0.9589	-1.8244
	Time	4.5400e-04	0.0108	4.8800e-04	3.4300e-04	0.0200	0.0330	4.3700e-04	0.0126
F_{19}	**Mean**	-4.2199	-4.6324	-10.5363	-5.1285	-9.7585	-5.2009	-2.8460	-6.3036
	STD	1.0368	0.4667	7.1776e-15	- 1.7944e-15	2.3516	1.0702	1.8090	3.3932
	Min	-8.7391	-5.1263	**-10.5363**	-5.1285	-10.5358	-10.2164	-9.0756	-10.5235
	Max	-10.5363	-3.0097	-5.1285	-2.4213	-2.7286	-1.0131	-1.6638	-2.3881
	Time	4.8100e-04	0.0175	2.3700e-04	3.8200e-04	0.0232	0.0405	3.8100e-04	0.0110

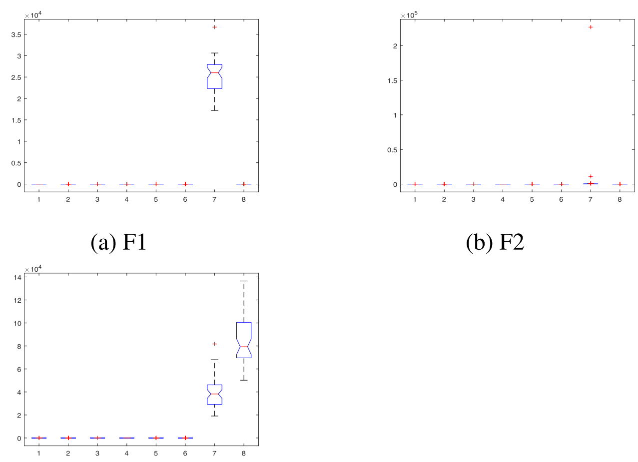


(a) F1 (b) F5 (c) F9 (d) F12 (e) F15

FIGURE 10. 2-D versions of BF1 [5], [100].

TABLE 11. Results of Kruskal-Wallis test [104] on BF1 [5], [100].

Fn.	p-Value	Significance
F_1	1.1928e-79	Highly Significant
F_2	2.5554e-78	Highly Significant
F_3	5.1973e-77	Highly Significant
F_4	6.2110e-78	Highly Significant
F_5	2.3214e-79	Highly Significant
F_6	5.2657e-78	Highly Significant
F_7	3.0626e-76	Highly Significant
F_8	5.9114e-76	Highly Significant
F_9	4.1665e-79	Highly Significant
F_{10}	1.6001e-76	Highly Significant
F_{11}	2.9247e-79	Highly Significant
F_{12}	1.2101e-47	Highly Significant
F_{13}	2.1901e-43	Highly Significant
F_{14}	1.9666e-76	Highly Significant
F_{15}	4.5515e-51	Highly Significant
F_{16}	2.0825e-57	Highly Significant
F_{17}	8.0227e-62	Highly Significant
F_{18}	1.4745e-67	Highly Significant
F_{19}	5.4890e-57	Highly Significant



(a) F1 (b) F2 (c) F3

FIGURE 11. Box plot of Kruskal-Wallis test [104] on BF1 [5], [100].

1) EXPLOITATION CAPABILITY OF BCO ALGORITHM

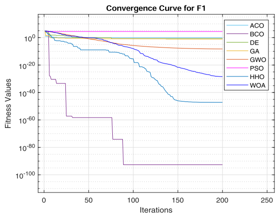
Unimodal functions are useful to compare the exploitation ability of different algorithms, as they have only one global optima. In functions F_1, F_2, F_3, F_4 and $F7$, the BCO algorithm outperforms all the other seven algorithms. The values are recorded in Table 9. The convergence curves are presented in Fig. 12. This clearly indicates that BCO has faster convergence speed and better optimal value finding ability in most cases. This proves that better exploitation of the search space is achieved by the BCO algorithm in most cases as compared to other state-of-the-art algorithms.

2) EXPLORATION CAPABILITY OF BCO ALGORITHM

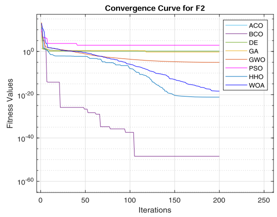
Multimodal functions have the ability to judge the exploration capability of an algorithm. From Tables 9 and 10, the superior performance of the BCO algorithm can be derived. In functions F_9, F_{12}, F_{14} and F_{15}, the proposed algorithm outperforms the others. It outperforms majority of the other algorithms for the rest of the multimodal functions. This proves that the BCO algorithm has good efficiency in terms of exploration of the search space.

C. ANALYSIS OF BF2

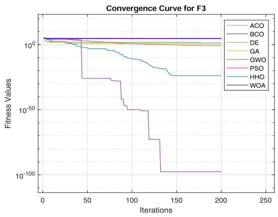
To evaluate the performance of the BCO algorithm, sixteen functions from the CEC'17 Benchmark Suite [101] (BF2) are selected. The details of the functions are provided in the supplementary document. The functions from the CEC'17 Benchmark Suite [101] are chosen in such a manner that *unimodal* functions ($CEC'17-1, 3$), *simple multimodal* functions ($CEC'17 - 4, 5, 6, 7, 9, 10$), *hybrid* functions ($CEC'17 - 11, 16, 18, 20$) and *composition* functions



(a) F1

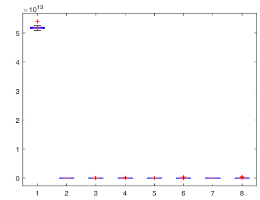


(b) F2

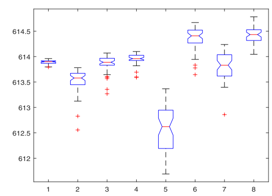


(c) F3

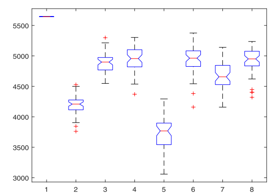
FIGURE 12. **Convergence curves of BF1 [5], [100].**



(a) CEC'17-3

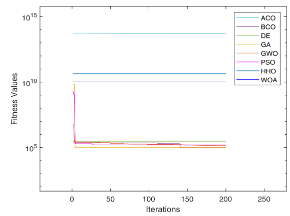


(b) CEC'17-6

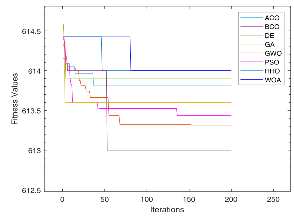


(c) CEC'17-16

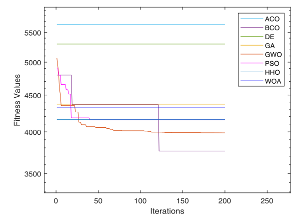
FIGURE 13. **Box plot of Kruskal-Wallis test [104] on BF2 [101].**



(a) CEC'17-3



(b) CEC'17-6



(c) CEC'17-16

FIGURE 14. **Convergence curves of BF2 [101].**

($CEC'17-22, 23, 25, 27$) are all included to conduct rigorous tests on the BCO algorithm. As mentioned in III-B, unimodal functions have a single global optima and multimodal functions have numerous local optima. Hybrid and composition functions are designed by keeping in mind the real-world problems. Hybrid functions are randomly divided into some subcomponents and each subcomponent is a basic function. Composition functions are combinations of basic and hybrid functions. The properties of the sub-functions are merged in a better way, to maintain continuity around the global/local optima in composition functions.

All the CEC'17 Benchmark functions [101] used are minimization problems. The 30D functions present in CEC'17 Benchmark Suite [101] are considered for this purpose. A total of 50 runs and 200 iterations are taken for every algorithm. The BCO algorithm is compared with ACO [3], DE [16], GA [11], GWO [5], HHO [58], PSO [4], WOA [52] algorithms.

The means, standard deviations, minimum values and maximum values obtained by the functions and the minimum time taken to converge are presented in Table 12. Kruskal-Wallis test [104] is applied to the results obtained from ACO [3], BCO, DE [16], GA [11], GWO [5], HHO [58], PSO [4], WOA [52]. This statistical test is carried out with 1% significance level for finding the p value. The p values of the Kruskal-Wallis tests [104] are recorded in Table 14. Three representative box plots are presented in Fig. 13 and rest of them are given in the supplementary document. The null hypothesis, that the values have the same distribution across all the eight methods, stands rejected. The BCO algorithm outperforms all other seven algorithms completely in functions $CEC'17-3, 6, 16$.

The composition functions are extremely challenging functions for testing metaheuristic algorithms. They can simultaneously benchmark exploration and exploitation capabilities. They contain numerous local optima. Hence, they can effectively examine the local optima avoidance capability of any algorithm. The BCO algorithm provides competitive results and outperforms ACO [3], DE [16], HHO [58] and WOA [52] in all the four functions. This shows that the BCO algorithm maintains a good balance between exploration and exploitation of the search space. This also ensures that it effectively avoids getting stuck into local optima.

D. ANALYSIS OF CONVERGENCE CURVES OF BCO

Three representative convergence curves, for each category of BF1 and BF2 functions are presented in Figs. 12 and 14, respectively. The convergence curves of BCO are compared to ACO [3], DE [16], GA [11], GWO [5], HHO [58], PSO [4] and WOA [52] algorithms. In most of the cases, the BCO algorithm converges faster than the other seven algorithms with optimal values. The other convergence curves for BF1 and BF2 are provided in the supplementary document.

A non-parametric test, called the Friedman Test [102], [103] is conducted among the participating algorithms. This method finds the individual rank of each of these algorithms,

TABLE 12. Results of BF2 [101].

Fn.	Results	BCO	ACO [3]	DE [16]	GA [11]	GWO [5]	HHO [58]	PSO [4]	WOA [52]
1	**Mean**	8.16E+10	1.63E+11	1.17E+11	1.3E+11	5.62E+09	1.84E+111	1.14E+11	1.9E+11
	STD	6.52E+09	1.59E+08	7.51E+09	1.49E+10	3.57E+09	2.65E+10	1.98E+10	2.4E+10
	Min	6.45E+10	1.63E+11	9.41E+10	9.65E+10	**3.18E+08**	1.31E+11	7.71E+10	1.37E+11
	Max	6.56E+10	1.63E+11	1.25E+11	1.56E+11	1.59E+10	2.39+11	1.68E+11	2.42E+11
	Time	0.2847	0.1125	0.1082	0.2456	10.8427	0.0021	0.2164	0.0262
3	**Mean**	1.3979e+05	5.1826e+13	3.1628e+08	8.8757e+09	1.0264e+05	2.3253e+10	1.7114e+05	3.5017e+10
	STD	3.1788e+04	5.9566e+11	1.1091e+09	3.4974e+10	2.7380e+04	5.8683e+10	3.7293e+04	9.5749e+10
	Min	**9.3010e+04**	5.0793e+13	9.7975e+04	9.4714e+04	1.0310e+05	1.7882e+05	9.3261e+04	2.2070e+05
	Max	2.2630e+05	5.4023e+13	5.7428e+09	2.2417e+11	1.7384e+05	3.6086e+11	2.5752e+05	4.7866e+11
	Time	0.4618	0.6440	0.5330	0.1601	10.6652	0.0018	0.0536	0.0036
4	**Mean**	2.0494e+04	3.7280e+04	2.2684e+04	2.7124e+04	741.0183	5.0770e+04	2.1762e+04	5.3408e+04
	STD	3.3414e+03	35.1105	3.3649e+03	4.1352e+03	196.0049	1.4197e+04	6.1588e+03	1.4931e+04
	Min	1.2549e+04	3.7205e+04	1.3007e+04	1.6644e+04	**550.6291**	2.4261e+04	8.4718e+03	2.5554e+04
	Max	2.6063e+04	3.7406e+04	2.8840e+04	3.5202e+04	1.5230e+03	8.7075e+04	3.8051e+04	9.7291e+04
	Time	0.1628	0.2797	0.1108	0.2798	10.6988	0.0018	0.1620	0.0018
5	**Mean**	864.7525	938.7560	876.8396	892.8001	618.1538	1.1602e+03	914.5525	1.1488e+03
	STD	14.5293	1.2781	15.1217	20.5061	43.5788	51.3530	36.6880	54.0301
	Min	821.6663	935.7678	833.4017	834.8426	**570.9852**	1.0427e+03	834.1551	1.0348e+03
	Max	893.6781	941.1925	901.3996	930.0059	783.2628	1.2576e+03	1.0132e+03	1.2492e+03
	Time	0.4688	9.5155	0.1095	0.1090	10.0475	0.0018	0.2729	0.0018
6	**Mean**	613.8888	613.5181	613.8612	613.8612	613.5855	614.3542	613.8189	614.4197
	STD	0.0423	0.2334	0.1602	0.1602	0.4425	0.2284	0.2755	0.1636
	Min	**613**	613.8	613.267	613.267	613.3	613.3	613.4	614.0452
	Max	613.7825	613.9607	614.0699	614.0699	613.3659	614.6696	614.2379	614.7821
	Time	0.1171	0.2750	0.1090	0.6130	2.3850	0.0018	0.1077	0.0059
7	**Mean**	2.1391e+03	2.9541e+03	2.5807e+03	2.7303e+03	891.2342	4.5980e+03	2.8466e+03	4.5777e+03
	STD	135.2042	1.9453	127.7943	133.7516	103.4120	568.8132	347.4947	483.3742
	Min	1.7090e+03	2.9492e+03	2.2393e+03	2.3460e+03	**717.2438**	3.2700e+03	2.0961e+03	3.4375e+03
	Max	2.3107e+03	2.9583e+03	2.7234e+03	2.9126e+03	1.1116e+03	5.6533e+03	3.7520e+03	5.5071e+03
	Time	0.4668	0.4668	0.1139	0.1105	11.3414	0.0018	0.4510	0.0019
9	**Mean**	2.6921e+04	1.6609e+04	1.6609e+04	2.0249e+04	3.6627e+03	4.1979e+04	1.7991e+04	4.2252e+04
	STD	1.3561e+03	1.7558e+03	1.7558e+03	2.1874e+03	1.6271e+03	8.6374e+03	4.0856e+03	8.1444e+03
	Min	2.4210e+04	1.0815e+04	1.0815e+04	1.4408e+04	**1.4520e+03**	2.3348e+04	1.0572e+04	2.7201e+04
	Max	2.9740e+04	1.9462e+04	1.9462e+04	2.5997e+04	1.1271e+04	6.0105e+04	2.9619e+04	5.8118e+04
	Time	0.2542	4.0581	0.1147	0.1163	0.8890	0.0022	0.2965	0.0039
10	**Mean**	1.0134e+04	1.5297e+04	1.1589e+04	1.1675e+04	3.3251e+03	1.1648e+04	1.0690e+04	1.1668e+04
	STD	335.3188	5.6047	454.8810	406.3886	1.2262e+03	442.9268	604.8925	539.2197
	Min	9.0098e+03	1.5287e+04	1.0484e+04	1.0807e+04	**1.7038e+03**	9.9003e+03	9.4015e+03	1.0002e+04
	Max	1.0716e+04	1.5308e+04	1.2617e+04	1.2603e+04	6.8214e+03	1.2407e+04	1.2031e+04	1.2665e+04
	Time	0.3827	0.4601	0.2582	0.1253	4.0360	0.0019	0.0569	0.0055
11	**Mean**	3.5296e+04	1.6772e+10	7.4179e+04	3.4191e+06	3.8845e+03	2.1228e+07	7.3174e+04	2.1254e+07
	STD	9.9325e+03	1.3753e+08	4.1481e+04	1.2485e+07	2.7835e+03	6.2660e+07	5.7380e+04	8.5433e+07
	Min	1.4847e+04	1.6561e+10	2.4165e+04	1.9751e+04	**1.2796e+03**	7.4756e+04	1.8869e+04	4.8068e+04
	Max	6.4348e+04	1.7367e+10	2.1618e+05	7.3220e+07	1.4871e+04	2.7240e+08	4.0382e+05	5.9302e+08
	Time	0.7744	3.5994	0.5520	0.7816	45.8855	0.0074	1.0258	0.0075
16	**Mean**	4.1991e+03	5.6439e+03	4.8912e+03	4.9590e+03	3.7080e+03	4.9465e+03	4.6696e+03	4.9184e+03
	STD	157.0791	0.9739	174.6683	200.9164309.9848	238.6925	194.0948	218.9940	
	Min	**3.7616e+03**	5.6420e+03	4.5481e+03	4.3704e+03	3.987e+03	4.1595e+03	4.1582e+03	4.3200e+03
	Max	4.5279e+03	5.6458e+03	5.2988e+03	5303	4.2937e+03	5.3754e+03	5.1411e+03	5.2367e+03
	Time	1.5801	0.5995	0.5717	0.5688	10.4534	0.0093	0.2884	0.0191
18	**Mean**	1.5484e+09	1.6737e+10	6.1640e+09	6.7375e+09	1.4665e+06	8.6647e+09	3.8660e+09	8.6044e+09
	STD	5.1548e+08	1.5063e+07	2.0288e+09	2.5884e+09	4.0120e+06	3.0939e+09	1.9532e+09	3.4217e+09
	Min	5.4621e+08	1.6674e+10	1.2282e+09	1.6446e+09	**1.3588e+04**	2.4964e+09	6.7262e+08	2.0373e+09
	Max	2.7447e+09	1.6773e+10	1.0763e+10	1.6074e+10	1.3461e+07	1.5250e+10	1.0098e+10	1.5903e+10
	Time	5.2322	0.7590	0.6745	0.6813	64.4102	0.3221	0.9691	0.2074
20	**Mean**	4.0145e+03	4.5995e+03	4.3791e+03	4.4283e+03	3.7525e+03	4.4786e+03	4.1604e+03	4.5532e+03
	STD	93.6157	0.6291	134.7850	162.5495	238.1481	183.7676	151.8691	164.0682
	Min	3.6532e+03	4.5977e+03	4.1338e+03	3.9386e+03	**3.3128e+03**	4.1588e+03	3.9273e+03	4.2416e+03
	Max	4.1437e+03	4.6015e+03	4.5949e+03	4.6021e+03	4.0938e+03	4.7643e+03	4.6731e+03	4.8326e+03
	Time	3.0544	1.2140	0.7881	0.7858	35.1844	0.0151	0.8063	0.0847
22	**Mean**	5.7484e+03	6.9757e+03	6.1698e+03	6.1967e+03	4.9656e+03	6.2525e+03	5.9701e+03	6.2762e+03
	STD	106.5333	2.1495	167.4507	167.4194	229.8795	130.0757	139.3917	149.9600
	Min	5.4086e+03	6.9698e+03	5.6880e+03	5.7473e+03	**4.4276e+03**	5.9556e+03	5.6125e+03	5.9694e+03
	Max	5.9017e+03	6.9807e+03	6.5099e+03	6.4550e+03	5.4268e+03	6.4745e+03	6.3034e+03	6.5776e+03
	Time	2.3725	3.3310	1.6240	1.6107	161.4615	0.0579	0.8208	0.0268
23	**Mean**	8.3167e+03	1.3866e+04	9.3807e+03	1.0066e+04	5.9497e+03	1.4396e+04	8.8458e+03	1.3768e+04
	STD	432.1708	16.3279	710.5974	1.1886e+03	67.3679	3.1958e+03	1.1219e+03	2.1859e+03
	Min	7.5198e+03	1.3835e+04	7.7685e+03	8.0145e+03	**5.7972e+03**	8.9192e+03	6.9031e+03	9.2884e+03
	Max	9.1543e+03	1.3909e+04	1.0514e+04	1.2912e+04	6.0712e+03	2.2121e+04	1.1260e+04	1.9243e+04
	Time	2.3570	9.7854	2.1642	2.1283	212.8284	0.0355	3.2755	0.0352
25	**Mean**	6.9875e+03	1.2874e+04	7.9427e+03	9.1812e+03	3.3968e+03	1.4124e+04	7.4507e+03	1.4358e+04
	STD	401.3087	16.1472	725.6574	1.2001e+03	155.7904	2.6182e+03	948.2650	2.2608e+03
	Min	5.6564e+03	1.2833e+04	6.2861e+03	6.7436e+03	**3.0528e+03**	8.4491e+03	5.7298e+03	8.6342e+03
	Max	7.8344e+03	1.2912e+04	9.2093e+03	1.1673e+04	3.8423e+03	2.0836e+04	9.5787e+03	2.0031e+04
	Time	10.1621	17.1394	2.9636	3.1428	268.9320	0.0888	22.2298	0.0414
27	**Mean**	7.9330e+03	1.1195e+04	9.1070e+03	9.5509e+03	5.9585e+03	9.5509e+03	1.0417e+04	1.5009e+04
	STD	1.2620e+03	1.3341e+03	873.6812	1.0111e+03	2.5844e+03	1.0111e+03	2.0329e+03	2.5719e+03
	Min	6.0366e+03	1.0134e+04	6.8744e+03	6.2735e+03	**3.1733e+03**	6.2735e+03	6.2526e+03	8.9355e+03
	Max	9.8966e+03	1.2912e+04	9.9592e+03	1.1244e+04	8.8443e+03	1.1244e+04	1.2652e+04	2.0976e+04
	Time	8.8523	32.0817	8.7271	2.9302	268.7617	0.0460	7.3490	0.0459

TABLE 13. Friedman test [102], [103] on the results of BF1 [5], [100] and BF2 [101].

Fn.	BCO	ACO [3]	DE [16]	GA [11]	GWO [5]	HHO [58]	PSO [4]	WOA [52]
BF1 [5] [100]								
F1	1	7	6	5	4	2	8	3
F2	1	7	6	5	4	2	8	3
F3	1	6	5	4	3	2	7	8
F4	1	6	4	5	3	2	8	7
F5	4	7	1.5	1.5	6	3	8	5
F6	4	7	6	5	3	1	8	2
F7	1	7	6	5	3	2	8	4
F8	4	8	6	7	3	1	5	2
F9	1.5	6	7	5	4	1.5	8	3
F10	4	7	6	5	3	1	8	2
F11	6	4	1.5	1.5	5	3	8	7
F12	1	7.5	6	7.5	4	2.5	5	2.5
F13	5	7	2	3	6	1	8	4
F14	5	8	7	6	3	2	1	4
F15	2	5	8	7	1	3	6	4
F16	2	4	7	6	1	5	8	3
F17	5	6	1	3	7	4	8	2
F18	3	7	1	4	2	6	8	5
F19	6	5	1	2	7	4	8	3
BF2 [101]								
1	2	4	5	6	1	7	3	8
3	1	8	4	5	2	6	3	7
4	2	6	4	5	1	7	3	8
5	2	6	3	4	1	7	5	8
6	1	5	4	6	2	7	3	8
7	2	6	3	4	1	8	5	7
9	2	6	4	5	1	7	3	8
10	2	8	4	5	1	6	3	7
11	2	8	4	5	1	6	3	7
16	1	8	4	6	2	7	3	5
18	2	8	4	5	1	6	3	7
20	1	8	4	5	2	6	3	7
22	2	4	5	6	1	7	3	8
23	2	7	4	5	1	8	3	6
25	2	6	4	5	1	7	3	8
27	1	6	3	4	2	8	6	7
Average rank	2.41	6.44	4.31	4.81	2.66	4.51	5.23	5.41
Overall rank	1	8	3	5	2	4	6	7

which helps to determine the overall comparative performance. In Table 13, the results of this test are presented, which clearly show that the proposed BCO algorithm outperforms others.

IV. CONCLUSION AND FUTURE DIRECTIONS

A novel swarm based optimization algorithm is proposed in this paper. The herding style of the Border Collie dogs is the main inspiration behind the development of the algorithm. Sobol's sensitivity indices are applied on the proposed algorithm for optimal tuning of the parameters. The Sobol g-function is used for implementing the sensitivity analysis. Thirty-five test functions are used to evaluate the performance of the algorithm. The exploration, exploitation and local minima avoidance capabilities of the Border Collie Optimization algorithm are compared with seven state-of-the-art algorithms viz., ACO, DE, GA, GWO, HHO, PSO and WOA. The exploration and exploitation of the search space are evaluated by using unimodal and multimodal functions. Few rotated and shifted functions from CEC'17 benchmark suite are also utilized for evaluating the performance of the BCO. The local minima avoidance capability of the BCO is observed using the eyeing technique. To judge the accuracy and stability of the proposed BCO algorithm, means and standard deviations of all the benchmark functions are reported. The results clearly indicate the superiority of the proposed algorithm in this regard. The BCO algorithm produces competitive results in terms of minimum and maximum fitness

TABLE 14. Results of Kruskal-Wallis test [104] on BF2 [101].

Fn.	p-value	Significance
$CEC'17-1$	3.6525e-58	Highly Significant
$CEC'17-3$	1.1177e-64	Highly Significant
$CEC'17-4$	1.9440e-69	Highly Significant
$CEC'17-5$	2.3735e-71	Highly Significant
$CEC'17-6$	5.4324e-65	Highly Significant
$CEC'17-7$	9.0694e-75	Highly Significant
$CEC'17-9$	3.9243e-60	Highly Significant
$CEC'17-10$	1.4938e-65	Highly Significant
$CEC'17-11$	4.5326e-11	Highly Significant
$CEC'17-16$	1.7205e-64	Highly Significant
$CEC'17-18$	5.8344e-13	Highly Significant
$CEC'17-20$	1.7791e-17	Highly Significant
$CEC'17-22$	2.5725e-33	Highly Significant
$CEC'17-23$	5.1966e-64	Highly Significant
$CEC'17-25$	1.3165e-63	Highly Significant
$CEC'17-27$	1.3165e-63	Highly Significant

values. In addition to this, the convergence times obtained are superior in most cases for the proposed algorithm. A statistical analysis test, called the Kruskal-Wallis test is performed, which proves the superiority of the proposed algorithm in most of the cases. The faster convergence capability of BCO algorithm is established using the convergence curves for all the test functions. The superiority of the BCO algorithm is also established by the Friedman Test.

Methods remain to be investigated to evaluate the performance of the BCO algorithm on other problems. More robust analysis for single objective optimization can be performed in future. The proposed algorithm can be compared with evolved or modified versions of state-of-the-art algorithms. The authors are presently engaged in developing different versions of the BCO algorithm for solving multi-objective problems. Moreover, the development of the extension of BCO algorithm, for dealing with real world and several engineering problems, is of prime interest to the authors. Moreover, evolution of hybrid algorithms can be developed by combining the BCO algorithm with other popular algorithms, may also be an interesting avenue for the researchers.

CODE AND DATA AVAILABILITY

The software code for the proposed algorithm is publicly available at GitHub: https://github.com/Tulika-opt/Border-Collie-Optimization.git.

REFERENCES

[1] T. Weise, M. Zapf, R. Chiong, and A. J. Nebro, "Why is optimization difficult?" in *Nature-Inspired Algorithms for Optimisation* (Studies in Computational Intelligence), vol. 193, R. Chiong, Ed. Berlin, Germany: Springer, 2009, pp. 1–50.

[2] X.-S. Yang, "Review of meta-heuristics and generalised evolutionary walk algorithm," *Int. J. Bio-Inspired Comput.*, vol. 3, pp. 77–84, Apr. 2011.

[3] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, Perth, WA, Australia, vol. 4. Nov./Dec. 1995, pp. 1942–1948.

[5] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[6] A. A. Ewees, M. Abd Elaziz, M. A. A. Al-Qaness, H. A. Khalil, and S. Kim, "Improved artificial bee colony using sine-cosine algorithm for multi-level thresholding image segmentation," *IEEE Access*, vol. 8, pp. 26304–26315, 2020.

[7] F. Xie, F. Li, C. Lei, J. Yang, and Y. Zhang, "Unsupervised band selection based on artificial bee colony algorithm for hyperspectral image classification," *Appl. Soft Comput.*, vol. 75, pp. 428–440, Feb. 2019.

[8] M. A. Awadallah, M. A. Al-Betar, A. L. Bolaji, I. A. Doush, A. I. Hammouri, and M. Mafarja, "Island artificial bee colony for global optimization," *Soft Comput.*, 2020, doi: 10.1007/s00500-020-04760-8.

[9] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, "Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm," *Int. J. Mach. Learn. Cybern.*, pp. 1–29, Dec. 2019, doi: 10.1007/s13042-019-01053-x.

[10] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Intelligent decision making through a simulation of evolution," *Behav. Sci.*, vol. 11, no. 4, pp. 253–272, Jul. 1966.

[11] J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI, USA: Univ. of Michigan Press, 1975.

[12] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, Jan. 1986.

[13] W. D. Hillis, "Co-evolving parasites improve simulated evolution as an optimization procedure," *Phys. D, Nonlinear Phenomena*, vol. 42, nos. 1–3, pp. 228–234, Jun. 1990.

[14] R. G. Reynolds, "An introduction to cultural algorithms," in *Proc. 3rd Annu. Conf. Evol. Program.* San Diego, CA, USA: World Scientific, 1994, pp. 131–139.

[15] J. Koza, "Genetic programming as a means for programming computers by natural selection," *Statist. Comput.*, vol. 4, no. 2, pp. 87–112, Jun. 1994.

[16] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[17] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 6, no. 6, pp. 580–593, Dec. 2002.

[18] O. Montiel, O. Castillo, P. Melin, A. R. Díaz, and R. Sepúlveda, "Human evolutionary model: A new approach to optimization," *Inf. Sci.*, vol. 177, no. 10, pp. 2075–2098, May 2007.

[19] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[20] P. Civicioglu, "Transforming geocentric Cartesian coordinates to geodetic coordinates by using differential search algorithm," *Comput. Geosci.*, vol. 46, pp. 229–247, Sep. 2012.

[21] C. Liu, M. Han, and X. Wang, "A novel evolutionary membrane algorithm for global numerical optimization," in *Proc. 3rd Int. Conf. Intell. Control Inf. Process.*, Dalian, China, Jul. 2012, pp. 727–732.

[22] P. Civicioglu, "Backtracking search optimization algorithm for numerical optimization problems," *Appl. Math. Comput.*, vol. 219, no. 15, pp. 8121–8144, Apr. 2013.

[23] H. Salimi, "Stochastic fractal search: A powerful metaheuristic algorithm," *Knowl.-Based Syst.*, vol. 75, pp. 1–18, Feb. 2015.

[24] T. T. Dhivyaprabha, P. Subashini, and M. Krishnaveni, "Synergistic fibroblast optimization: A novel nature-inspired computing algorithm," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 7, pp. 815–833, Jul. 2018.

[25] X. Chen, Y. Liu, X. Li, Z. Wang, S. Wang, and C. Gao, "A new evolutionary multiobjective model for traveling salesman problem," *IEEE Access*, vol. 7, pp. 66964–66979, 2019.

[26] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 80–671, 1983.

[27] Z. Woo Geem, J. Hoon Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *SIMULATION*, vol. 76, no. 2, pp. 60–68, Feb. 2001.

[28] B. Webster and P. J. Bernhard, "A local search optimization algorithm based on natural principles of gravitation," in *Proc. Int. Conf. Inf. Knowl. Eng. (IKE)*, Las Vegas, NV, USA, Jun. 2003, pp. 255–261.

[29] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: Charged system search," *Acta Mechanica*, vol. 213, nos. 3–4, pp. 267–289, Sep. 2010.

[30] E. Cuevas, D. Oliva, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "Circle detection using electro-magnetism optimization," *Inf. Sci.*, vol. 182, no. 1, pp. 40–55, Jan. 2012.

[31] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.*, vols. 110–111, pp. 151–166, Nov. 2012.

[32] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-verse optimizer: A nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 27, no. 2, pp. 495–513, Feb. 2016.

[33] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.

[34] F. A. Hashim, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, and S. Mirjalili, "Henry gas solubility optimization: A novel physics-based algorithm," *Future Gener. Comput. Syst.*, vol. 101, pp. 646–667, Dec. 2019.

[35] T. Ray and K. M. Liew, "Society and civilization: An optimization algorithm based on the simulation of social behavior," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 386–396, Aug. 2003.

[36] L. M. Zhang, C. Dahlmann, and Y. Zhang, "Human-inspired algorithms for continuous function optimization," in *Proc. IEEE Int. Conf. Intell. Comput. Intell. Syst.*, Shanghai, China, Nov. 2009, pp. 318–321.

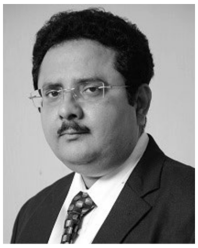
[37] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems," *Inf. Sci.*, vol. 183, no. 1, pp. 1–15, Jan. 2012.

[38] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Syst. Mag.*, vol. 22, no. 3, pp. 52–67, Mar. 2002.
[39] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Eng. Fac., Dept. Comput. Eng., Erciyes Univ., Kayseri, Turkey, Tech. Rep. tr06, 2005, vol. 200, pp. 1–10.
[40] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Oct. 2007.
[41] X. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proc. World Congr. Nature Biologically Inspired Comput. (NaBIC)*, Coimbatore, India, Dec. 2009, pp. 210–214.
[42] X. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)* (Studies in Computational Intelligence), vol. 284, J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds. Berlin, Germany: Springer, 2010, pp. 65–74.
[43] X-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *Int. J. Bio-Inspired Comput.*, vol. 2, no. 2, pp. 78–84, 2010.
[44] W.-T. Pan, "A new fruit fly optimization algorithm: Taking the financial distress model as an example," *Knowl.-Based Syst.*, vol. 26, pp. 69–74, Feb. 2012.
[45] X.-S. Yang, "Flower pollination algorithm for global optimization," in *Unconventional Computation and Natural Computation* (Lecture Notes in Computer Science), vol. 7445. Berlin, Germany: Springer, 2012, pp. 240–249.
[46] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 17, no. 12, pp. 4831–4845, Dec. 2012.
[47] J. C. Bansal, H. Sharma, S. S. Jadon, and M. Clerc, "Spider monkey optimization algorithm for numerical optimization," *Memetic Comput.*, vol. 6, no. 1, pp. 31–47, Mar. 2014.
[48] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[49] S. Mirjalili, "The ant lion optimizer," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, May 2015.
[50] S. Mirjalili, "Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Comput. Appl.*, vol. 27, no. 4, pp. 1053–1073, 2015.
[51] X.-B. Meng, X. Z. Gao, L. Lu, Y. Liu, and H. Zhang, "A new bio-inspired optimisation algorithm: Bird swarm algorithm," *J. Exp. Theor. Artif. Intell.*, vol. 28, no. 4, pp. 673–687, Jul. 2016.
[52] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[53] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Comput. Struct.*, vol. 169, pp. 1–12, Jun. 2016.
[54] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.
[55] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017.
[56] G. Dhiman and V. Kumar, "Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications," *Adv. Eng. Softw.*, vol. 114, pp. 48–70, Dec. 2017.
[57] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: Squirrel search algorithm," *Swarm Evol. Comput.*, vol. 44, pp. 148–175, Feb. 2019.
[58] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019.
[59] A. M. Fathollahi-Fard, M. Hajiaghaei-Keshteli, and R. Tavakkoli-Moghaddam, "Red deer algorithm (RDA): A new nature-inspired metaheuristic," *Soft Comput.*, Mar. 2020, doi: 10.1007/s00500-020-04812-z.
[60] N. Covic and B. Lacevic, "Wingsuit flying search—A novel global optimization algorithm," *IEEE Access*, vol. 8, pp. 53883–53900, 2020.
[61] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization," *Eng. Appl. Artif. Intell.*, vol. 90, Apr. 2020, Art. no. 103541.
[62] A. S. A. Elrahman and H. A. Hefny, "Vortex swarm optimization: New metaheuristic algorithm," in *Proc. Int. Conf. Artif. Intell. Comput. Vis. (AICV)*, in Advances in Intelligent Systems and Computing, vol. 1153. Springer, 2020, pp. 127–136.
[63] S. Chatterjee, S. Dawn, and S. Hore, "Artificial cell swarm optimization," in *Frontier Applications of Nature Inspired Computation* (Springer Tracts in Nature-Inspired Computing). Singapore: Springer, 2020, pp. 196–214.
[64] H. Drias, Y. Drias, and I. Khennak, "A new swarm algorithm based on orcas intelligence for solving maze problems," in *Trends and Innovations in Information Systems and Technologies* (Advances in Intelligent Systems and Computing), vol. 1159. Cham, Switzerland: Springer, 2020, pp. 788–797.
[65] L. Jianhua, D. Xiangqian, W. Sun'an, and Y. Qing, "Family genetic algorithms based on gene exchange and its application," *J. Syst. Eng. Electron.*, vol. 17, no. 4, pp. 864–869, Dec. 2006.
[66] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 38, no. 1, pp. 218–237, Jan. 2008.
[67] E. A. Portilla-Flores, A. Sanchez-Marquez, L. Flores-Pulido, E. Vega-Alvarado, M. B. Calva Yanez, J. A. Aponte-Rodriguez, and P. A. Nino-Suarez, "Enhancing the harmony search algorithm performance on constrained numerical optimization," *IEEE Access*, vol. 5, pp. 25759–25780, 2017.
[68] Y. Liu and L. Ma, "Improved gravitational search algorithm based on free search differential evolution," *J. Syst. Eng. Electron.*, vol. 24, no. 4, pp. 690–698, Aug. 2013.
[69] B. Wang, T. Xiang, N. Li, W. He, W. Li, and X. Hei, "A symmetric sine cosine algorithm with adaptive probability selection," *IEEE Access*, vol. 8, pp. 25272–25285, 2020.
[70] H. Feng and Q. Li, "An improved teaching-learning based optimization algorithm and its application to aero-engine start model adaptation," *IEEE Access*, vol. 7, pp. 136525–136534, 2019.
[71] W. Deng, J. Xu, and H. Zhao, "An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem," *IEEE Access*, vol. 7, pp. 20281–20292, 2019.
[72] N. Lynn and P. N. Suganthan, "Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation," *Swarm Evol. Comput.*, vol. 24, pp. 11–24, Oct. 2015.
[73] S. Zhang and S. Liu, "A discrete improved artificial bee colony algorithm for 0–1 knapsack problem," *IEEE Access*, vol. 7, pp. 104982–104991, 2019.
[74] S. Gao, Y. Gao, Y. Zhang, and L. Xu, "Multi-strategy adaptive cuckoo search algorithm," *IEEE Access*, vol. 7, pp. 137642–137655, 2019.
[75] W. Long, J. Jiao, X. Liang, and M. Tang, "An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization," *Eng. Appl. Artif. Intell.*, vol. 68, pp. 63–80, Feb. 2018.
[76] X. Zhao, X. Zhang, Z. Cai, X. Tian, X. Wang, Y. Huang, H. Chen, and L. Hu, "Chaos enhanced grey wolf optimization wrapped ELM for diagnosis of paraquat-poisoned patients," *Comput. Biol. Chem.*, vol. 78, pp. 481–490, Feb. 2019.
[77] Y. Ling, Y. Zhou, and Q. Luo, "Lévy flight trajectory-based whale optimization algorithm for global optimization," *IEEE Access*, vol. 5, pp. 6168–6186, 2017.
[78] X. Han, Q. Xu, L. Yue, Y. Dong, G. Xie, and X. Xu, "An improved crow search algorithm based on spiral search mechanism for solving numerical and engineering optimization problems," *IEEE Access*, vol. 8, pp. 92363–92382, 2020.
[79] S. S. Guo, J. S. Wang, W. Xie, M. W. Guo, and L. F. Zhu, "Improved grasshopper algorithm based on gravity search operator and pigeon colony landmark operator," *IEEE Access*, vol. 8, pp. 22203–22224, 2020.
[80] S. Saremi, S. M. Mirjalili, and S. Mirjalili, "Chaotic krill herd optimization algorithm," *Procedia Technol.*, vol. 12, pp. 180–185, Jan. 2014.
[81] L. Shen, H. Chen, Z. Yu, W. Kang, B. Zhang, H. Li, B. Yang, and D. Liu, "Evolving support vector machines using fruit fly optimization for medical data classification," *Knowl.-Based Syst.*, vol. 96, pp. 61–75, Mar. 2016.
[82] M. Wang, H. Chen, B. Yang, X. Zhao, L. Hu, Z. Cai, H. Huang, and C. Tong, "Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses," *Neurocomputing*, vol. 267, pp. 69–84, Dec. 2017.
[83] Y. Xu, H. Chen, J. Luo, Q. Zhang, S. Jiao, and X. Zhang, "Enhanced moth-flame optimizer with mutation strategy for global optimization," *Inf. Sci.*, vol. 492, pp. 181–203, Aug. 2019.

[84] H. Chen, Q. Zhang, J. Luo, Y. Xu, and X. Zhang, "An enhanced bacterial foraging optimization and its application for training kernel extreme learning machine," *Appl. Soft Comput.*, vol. 86, Jan. 2020, Art. no. 105884.
[85] A. S. Menesy, H. M. Sultan, A. Selim, M. G. Ashmawy, and S. Kamel, "Developing and applying chaotic Harris Hawks optimization technique for extracting parameters of several proton exchange membrane fuel cell stacks," *IEEE Access*, vol. 8, pp. 1146–1159, 2020.
[86] R. J. Kuo and Y. S. Han, "A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem—A case study on supply chain model," *Appl. Math. Model.*, vol. 35, no. 8, pp. 3905–3917, Aug. 2011.
[87] H. Ouyang, G. Ma, G. Liu, Z. Li, and X. Zhong, "Hybrid teaching-learning based optimization with harmony search for engineering optimization problems," in *Proc. 36th Chin. Control Conf. (CCC)*, Dalian, China, Jul. 2017, pp. 2714–2717.
[88] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, pp. 302–312, Oct. 2017.
[89] K. Qian, W. Li, and W. Qian, "Hybrid gravitational search algorithm based on fuzzy logic," *IEEE Access*, vol. 5, pp. 24520–24532, 2017.
[90] X. Bao, H. Jia, and C. Lang, "A novel hybrid Harris Hawks optimization for color image multilevel thresholding segmentation," *IEEE Access*, vol. 7, pp. 76529–76546, 2019.
[91] S. Arora, H. Singh, M. Sharma, S. Sharma, and P. Anand, "A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection," *IEEE Access*, vol. 7, pp. 26343–26361, 2019.
[92] M. B. Agbaje, A. E. Ezugwu, and R. Els, "Automatic data clustering using hybrid firefly particle swarm optimization algorithm," *IEEE Access*, vol. 7, pp. 184963–184984, 2019.
[93] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[94] akc.org. *American Kennel Club*. Accessed: Dec. 26, 2019. [Online]. Available: https://www.akc.org/dog-breeds/border-collie
[95] pixabay.com. *Pixabay*. Accessed: May 10, 2019. [Online]. Available: https://pixabay.com/photos/search/border%20collie/
[96] S. Coren, *The Intelligence of Dogs*. London, U.K.: Simon & Schuster, 2006.
[97] J. Kaminski, "Word learning in a domestic dog: Evidence for 'fast mapping,'" *Science*, vol. 304, no. 5677, pp. 1682–1683, Jun. 2004.
[98] J. W. Pilley and A. K. Reid, "Border collie comprehends object names as verbal referents," *Behavioural Processes*, vol. 86, no. 2, pp. 184–195, Feb. 2011.
[99] colliepoint.com. *Border Collie History: From Old Hemp to New Beginnings—ColliePoint*. Accessed: May 8, 2020. [Online]. Available: https://colliepoint.com/border-collie-history/
[100] S. Shekhawat and A. Saxena, "Development and applications of an intelligent crow search algorithm based on opposition based learning," *ISA Trans.*, vol. 99, pp. 210–230, Apr. 2020.
[101] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition and special session on constrained single objective real-parameter optimization," Nanyang Technol. Univ., Singapore, Tech. Rep. 2017, Nov. 2016.
[102] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.
[103] M. Friedman, "A comparison of alternative tests of significance for the problem of *m* rankings," *Ann. Math. Statist.*, vol. 11, no. 1, pp. 86–92, Mar. 1940.
[104] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *J. Amer. Stat. Assoc.*, vol. 47, no. 260, pp. 583–621, Dec. 1952.
[105] P. Loubiére, A. Jourdan, P. Siarry, and R. Chelouah, "A sensitivity analysis method aimed at enhancing the metaheuristics for continuous optimization," *Artif. Intell. Rev.*, vol. 50, no. 4, pp. 625–647, Dec. 2018.
[106] I. M. Sobol, "Sensitivity estimates for nonlinear mathematical models," *Math. Model. Comput. Exp.*, vol. 1, no. 4, pp. 407–414, 1993.
[107] T. Homma and A. Saltelli, "Importance measures in global sensitivity analysis of nonlinear models," *Rel. Eng. Syst. Saf.*, vol. 52, no. 1, pp. 1–17, Apr. 1996.

TULIKA DUTTA was born in 1986. She received the B.E. degree in computer science and engineering from the University Institute of Technology, Burdwan Univeristy, India, in 2009, and the master's degree (Diploma) in multimedia and web development and the M.E. degree in multimedia development from Jadavpur University, India, in 2011 and 2013, respectively.

Her research interests include soft computing, quantum computing, and remote sensing.



SIDDHARTHA BHATTACHARYYA (Senior Member, IEEE) received the bachelor's degree in physics, and the bachelor's and master's degrees in optics and optoelectronics from the University of Calcutta, Kolkata, India, in 1995, 1998, and 2000, respectively, and the Ph.D. degree in computer science and engineering from Jadavpur University, Kolkata, in 2008.

He was the Principal with the RCC Institute of Information Technology, Kolkata. He was also a Senior Research Scientist with the Faculty of Electrical Engineering and Computer Science, VSB Technical University of Ostrava, Ostrava, Czech Republic. He is currently a Professor with the Department of Computer Science and Engineering, CHRIST (Deemed to be University), Bangalore, India. He has authored or coauthored more than 280 research publications in international journals and conference proceedings. He has coauthored five books and co-edited 54 books. He holds over three patents. His research interests include soft computing, pattern recognition, multimedia data processing, hybrid intelligence, social networks, and quantum computing.

SANDIP DEY (Member, IEEE) received the bachelor's degree in mathematics, in 1999, the B-level (Equivalent to MCA) from the DOEACC Society and the M.Tech. degree in software engineering from the West Bengal University of Technology, in 2005 and 2008, respectively, and the Ph.D. degree in computer science and engineering from Jadavpur University, India, in 2016.

He was an Associate Professor and the HOD with the Department of Computer Science and Engineering, Global Institute of Management and Technology, Krishnanagar, India. He is currently an Assistant Professor with the Department of Computer Science, Sukanta Mahavidyalaya, Jalpaiguri. He has coauthored one book, coedited two books, and authored or coauthored more than 40 research publications in international journals, book chapters, and conference proceedings. His research interests include soft computing, hybrid intelligence, quantum computing, and image analysis.

JAN PLATOS (Member, IEEE) received the Ph.D. degree in computer science from the VSB-Technical University of Ostrava, in 2006.

He was an Associate Professor in computer science, in 2014. Since 2017, he has been the Head of the Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB Technical University of Ostrava. He is the coauthor of more than 180 scientific articles published in proceedings and journals. His citation report consists of 302 citations and h-index nine on *Web of Science*, 685 citations and h-index 13 on *Scopus*, and 965 citations and h-index 15 on Google Scholar. His research interests include computer science, text processing, data compression, bio-inspired algorithms, information retrieval, data mining, data structures, data prediction, application of different computer science algorithms and methods, and innovative approaches to severe practical problems from any area.