Decision Making: Applications in Management and Engineering
ISSN: 2560-6018
eISSN: 2620-0104
DOI:_https://doi.org/10.31181/dmame0306102022r

A COMPARATIVE STUDY OF METAHEURISTICS ALGORITHMS BASED ON THEIR PERFORMANCE OF COMPLEX BENCHMARK PROBLEMS

Tithli Sadhu[1,2], Somanth Chowdhury[3], Shubham Mondal[4], Jagannath Roy[5], Jitamanyu Chakrabarty[1] and Sandip Kumar Lahiri[3*]

[1] Department of Chemistry, National Institute of Technology Durgapur, West Bengal, India
[2] Department of Biochemistry, School of Agriculture, SR University, Hanumakonda, Telangana, India
[3] Department of Chemical Engineering, National Institute of Technology Durgapur, West Bengal, India
[4] Department of Computer Science and Engineering, Institute of Engineering and Management Kolkata, West Bengal, India
[5] Department of Mathematics, National Institute of Technology Warangal, Telangana, India

Received: 16 February 2022;
Accepted: 18 September 2022;
Available online: 6 October 2022.

Original scientific paper

Abstract: *Metaheuristic approaches with significant improvements are very promising in the solution of intractable optimization problems. The objective of the present study is to test the capability of applications and compare performance of the four selected algorithms from "classical" (simulated annealing (SA), genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE)) and "new generation" (firefly algorithm (FFA), krill herd (KH), grey wolf optimization (GWO), and symbiotic organism search (SOS)) each by solving selected benchmark problems. SOS and KH algorithm successfully solved most of all the selected problems by achieving the best solution in minimum execution time. On the other hand, DE, and PSO also effectively attained the optimal solution which were very close to the best one. Therefore, no firm conclusion can be done about the universally best algorithm and their performance may be varied for different benchmark problems. However, "new generation" algorithm exhibited the most promising result and great potential than the "classical" one. This study gives some insights to use SOS and KH as best performing algorithm to the novice user who can easily get lost by the plethora of large number of optimization algorithms.*

* Corresponding author.
E-mail addresses: sadhu.tithli@gmail.com (T. Sadhu), sc.19ch1103@phd.nitdgp.ac.in (S. Chowdhury), shubhammondal1999@outlook.com (S. Mondal), jroy@nitw.ac.in (J. Roy), jitamanyu.chakrabarty@ch.nitdgp.ac.in (J. Chakrabarty), sandipkumar.lahiri@che.nitdgp.ac.in (S.K.. Lahiri)

Key words: *Metaheuristic, Algorithms, Optimization, Performance, Benchmark problems*

1. Introduction

Optimization in every field of Science and Technology recently gets attention to the researchers as resources, energies are getting limited day by day. Optimization refers to a procedure to get a balance between two or more conflicting objectives with respect to design variables under several conditions and restrictions on them (Rangaiah, 2010). Due to the complexity of modern technologies, the objective function and associated constraints are very complex in real-life applications. Gradient-based classical optimization algorithms often trap in local optima of this complex objective function and cannot find the global optima. To overcome the limitation of gradient-based algorithms, metaheuristic, stochastic approaches are introduced. These algorithms have a random component included in their execution which helps them to escape from local minima. There is an iterative computational technique at the heart of these metaheuristic algorithms that selects an optimal solution iteratively and tries to enhance a candidate solution with regard to a particular measure of quality (Wang & Chen, 2013). In the recent studies, it is referred as nature inspired "metaheuristic" that means the higher level of heuristic that are applied to solve a wide variety of optimization problems. The main advantage of using this metaheuristic algorithm is that it allows the decision makers to obtain near optimal solutions within a relatively shorter period of time even for large size complex problems because of their efficient performance ability (Dokeroglu et al., 2019). Many complex optimization problems with a large variance, ranging from single to multi-objective, constrained to unconstrained, continuous to discrete, can be solved by a practical and elegant way using metaheuristic approaches (Dokeroglu et al., 2019).

The majority of state-of-the-art metaheuristic has been created prior to the year 2000 and in this article these can be termed as "classical" metaheuristic algorithms (Dokeroglu et al., 2019). Some of the major examples of classical algorithms are: Simulated Annealing (SA) (Kirkpatrick et al., 1983), Genetic Algorithm (GA) (Goldberg, 1989), Ant Colony Optimization (Dorigo & Di Caro, 1999), Particle Swarm Optimization (Kennedy & Eberhart, 1995), Differential Evolution (DE) (Storn & Price, 1997), Chaos Optimization Method (COM) (Li & Jiang, 1997), Variable Neighborhood Search (VNS) (Mladenović & Hansen, 1997), Genetic Programming (GP) (Banzhaf et al., 1998), Tabu search (TS) (Glover & Laguna, 1998), Greedy Randomized Adaptive Search Procedure (GRASP) (Marques-Silva & Sakallah, 1999), etc. These algorithms are widely used in almost every field of science and technology and proven to be very versatile. The main advantages of the "classical" metaheuristic algorithms as reported in literature are: (1) low execution time (processing time of the program) for large and complex problems, (2) their ability to escape local optima and high probability to getting global optima solutions (Beheshti & Shamsuddin, 2013). Despite the huge success rate of classical metaheuristic algorithms in diverse field, new generation evolutionary algorithms also developed magnificently in last twenty years improving their performance and execution time. Recently, researcher find out that nature itself has many efficient optimization processes. Various species in nature such as, birds, fish etc. possesses very effective optimization capabilities. From year 2000 to till now, researchers focused on evolutionary or behavioral processes seen in nature and try to mimic them in mathematical algorithms. These give rise the development of new generation, nature-inspired metaheuristic algorithms that can solve all complex real-

world problems and acquire the more practical optimal solution in very short execution time compare to "classical" one for some unsolved benchmark problem sets in all perspective, even for very large problem size (Dokeroglu et al., 2019). Some major examples of "new generation" metaheuristic algorithms are Harmony Search (HS) (Geem et al., 2001), Artificial Bee Colony (ABC) (Karaboga, 2005), Bacterial Foraging Optimization (BFO) (Das et al., 2009), Cuckoo Search (CS) (Yang & Deb, 2009), Firefly Algorithm (FFA) (Yang, 2010a), Bat Algorithm (BA) (Yang, 2010b), Krill Herd (KH) (Gandomi & Alavi, 2012), Grey Wolf Optimization (GWO) (Mirjalili et al., 2014), Symbiotic Organism Search (SOS) (Cheng & Prayogo, 2014), Whale Optimization (WOA) (Mirjalili & Lewis, 2016) etc. A schematic representation of development of both "classical" and "new generation" metaheuristic optimizations (year-wise) is represented in Figure 1(a) and (b), respectively.

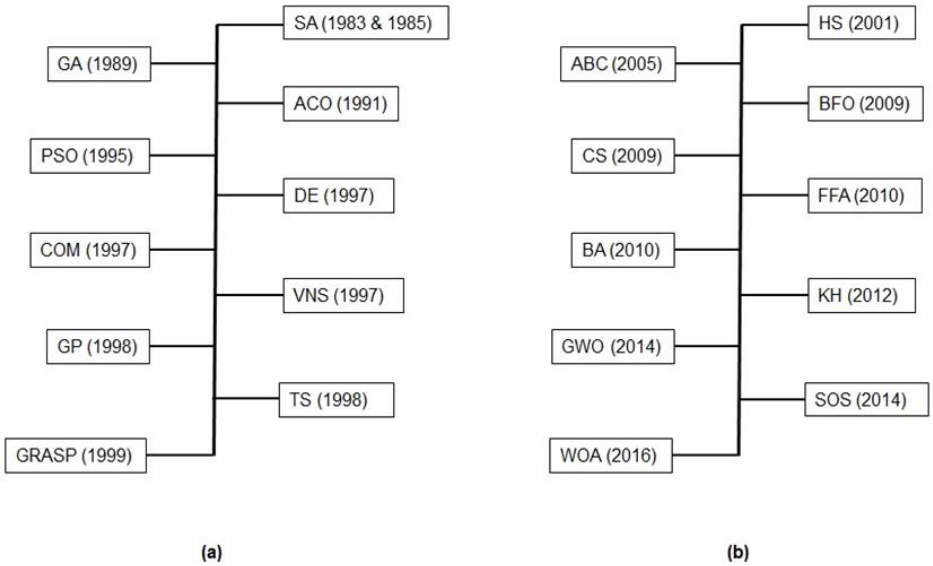

Figure 1. Year-wise development of (a) "classical" and (b) "new generation" metaheuristic algorithms

Due to the availability of large number of metaheuristic algorithms, the novice researchers get confused about the selection of efficient algorithm to solve hard and complex problems. Now-a-days, new algorithm is developed every month and is claimed to be the best among the existing one. However, when the algorithms are applied to a new objective function, this claim is not substantiated. This necessitates making comparison of different old and new generation metaheuristic algorithms on benchmark problem sets.

This study aims to address the above-mentioned issues by following manners:

- First, seven benchmark problems were selected from the published literature and at the time of selection, attention was given that the objective function and associated constraints are very complex with multiple local optima and used before in literature for algorithm testing purpose.
- Out of fifty major "classical" and "new generation" algorithms, four "classical" and four "new generation" algorithms were chosen from published papers (in last 40 years) on the basis of their potential and applications in diverse fields.

- Finally, the main objective of this paper is fulfilled by testing their capability of applications by solving the selected benchmark problems and compare the performance of recent next generation promising algorithms against old generation "classical" algorithms.

This paper is organized as follows: Section 2 and Section 3 describe a brief description of the selected "classical" and "new generation" algorithms, respectively. In result and discussion, Section 4.1 covers the detailed information about the selected benchmark problems. The performance of the selected algorithms on benchmark problems and concluding remarks about performance are presented in Section 4.2 and 5, respectively. The ultimate concluding remarks about this study are provided in Section 6.

2. Classical algorithms

Classical or old generation algorithms refer to all metaheuristic algorithms published before the year 2000. Because of their versatility over 40 years, these algorithms are applied in diverse field of science, engineering and medical and proven to be very robust. A schematic representation of major old generation algorithms is given in Figure 1(a). Due to brevity, in this section comprehensive information describing the main mechanism for optimization, metaheuristic behind execution and pseudocode are provided about the selected "classical" algorithms.

2.1. Simulated annealing (SA)

SA is a stochastic optimization tool for approximating the global optimum of a given function (Rao, 2019). This versatile and successful method of optimization is very beneficial for locating global optima when there are a lot of local optima. SA mimics the certain thermodynamics principals of producing ideal crystal. The term "Annealing" refers to a thermodynamic analogue, specifically the cooling and annealing of metals. This algorithm simulates the slow cooling process of molten metal by controlling the parameter such as temperature to get the minimum value of a function in a minimization problem. A selection of cooling system, i.e., the technique of reducing temperature is discussed in detail in Rangaiah (2010).

Let an unconstrained nonlinear minimization problem but with bounds on variables is defined as:

$\min F(x)$ (1)

Subject to

$x_i^l \leq x_i \leq x_i^{up}; i = 1, \dots, p$ (2)

Basic SA algorithm to solve the above-mentioned problem (Eq. 1 and 2) is illustrated in Figure 2. Eq. 3 and conditions (1) and (2) in the figure are:

$\Delta F^k = F(x^k) - F(x^{k-1})$ (3)

$\Delta F^k \leq 0$ (Condition 1)

$\Delta F^k > 0$ and P $> RND$ (Condition 2)

Where, P= probability of accepting a feasible point

RND =random number from uniform distribution between 0 and 1.

SA approach is also used for many non-linear problems with continuous variables. SA optimization process is commonly used in chemical and process engineering, especially for combinatorial problems or discrete-valued optimization problems. Therefore, SA is applied in discrete, but very large configuration spaces and has a wide range of applications that are still being investigated (Rangaiah, 2010).

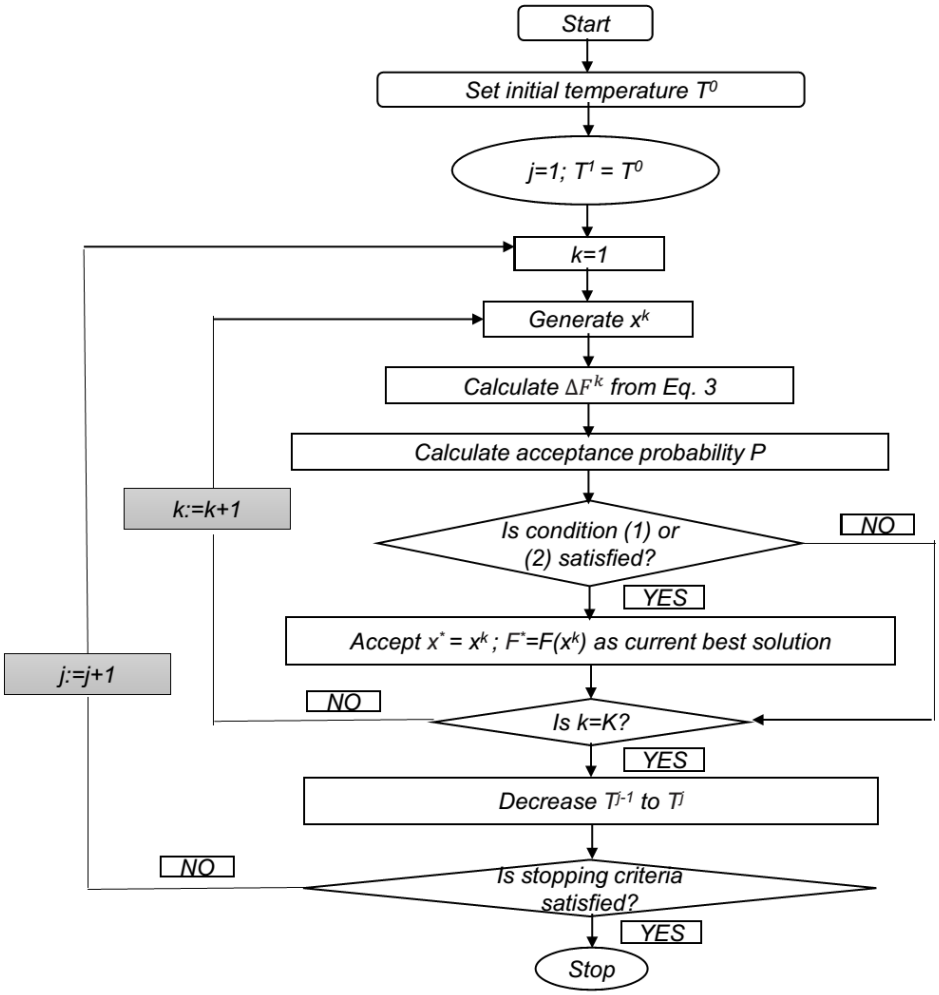

Figure 2. Flowchart of basic SA algorithm

2.2. Genetic algorithm (GA)

GA is the first algorithm in the field of metaheuristics designed by natural inspiration in the search optimization and machine learning processes. GA works by combining the 'survival of the fittest' principle of natural evolution with genetic propagation of characteristics (Lahiri & Ghanta, 2010). The benefit of this principle is that it intelligently exploits the random search provided by previous data for exploring the better performance region of the solution space. As a result, this algorithm is commonly used to provide high-quality solutions to optimization and search problems. The details of the procedures of GA are described in Lahiri and Ghanta (2009, 2010).

The pseudo code for GA algorithm is represented in Figure 3.

```
Generate random population of P solutions (chromosomes);
For each individual i∈P: calculate fitness (i);
        For i=1 to number of generations;
             Randomly select an operation (crossover or mutation);
             If  crossover;
                  Select two parents at random i_α and i_β;
                  Generate an offspring i_δ=crossover (i_α and i_β);
             Else if mutation;
                  Select one chromosome i at random;
                  Generate an offspring i_δ=mutate (i);
             End if;
             Calculate the fitness of the offspring i_δ;
             If i_δ is better than the worst chromosome then replace the worst chromosome by i_δ;
        Next i;
   Check if termination=true;
End;
```

Figure 3. Pseudo code of GA algorithm

2.3. Particle swarm optimization (PSO)

Particle swam optimization (PSO) is another metaheuristic global optimization algorithm which developed from the principle of swarm intelligence and based on the research on bird and fish flock movement behavior. This stochastic optimization technique is very easy to implement and very few particles need to be tuned. PSO is rapidly applied to solve complex optimist problems because the most optimist solution is worked out by the cooperation of each individual (Bai, 2010). The position of the most optimist particle during its movement (individual experience) and also in its surrounding (near experience) affects the position of each particle in the swarm. By applying its previous experience with flying and nearby particles, each particle adjusts its velocity to locate a better solution. The modification of the process in positions and velocities of the particles is illustrated in detail in Lahiri et al. (2012) and in this article shortly depicted in Figure 4.

Easy implementation procedure and simplicity of the algorithm make helpful to wide and successful application of it in many areas such as model classification, function optimization, neural network training, the signal procession, automatic adaptation control, fuzzy system control etc.

Step 1. Initialize optimization

1.1 Initialize algorithm constants t_{max}, P
1.2 Initialize randomly all particles positions x_t^i *and velocities* v_t^i
1.3 Evaluate objective function values as $f(x_t^i)$
1.4 Assign best positions $p_t^i = x_t^i$ *with* $f(p_t^i) = f(x_t^i), i = 1, \dots, P$
1.5 Find $f_t^{best}(p_t^{best}) = min\ \{f(p_t^1), \dots, f(p_t^i), \dots, f(p_t^p)\}$
And initialize $p_t^g = p_t^{best}$ *and* $f(p_t^g) = f_t^{best}(p_t^{best})$

Step 2. Perform optimization

While $(t \le t_{max})$
2.1 Update particle positions x_t^i *and velocities* v_t^i *of all P particles*
2.2 Evaluate objective function values as $f(x_t^i)$
2.3 Update particle best positions if $f(p_t^i) > f(x_t^i)$ *then* $p_t^i = x_t^i$ *with* $f(p_t^i) = f(x_t^i), i = 1, \dots, P$
2.4 Find $f_t^{best}(p_t^{best}) = min\ \{f(p_t^1), \dots, f(p_t^i), \dots, f(p_t^P)\}$
If $f(p_t^g) > f(p_t^{best})$ *then* $p_t^g = p_t^{best}$ *and* $f(p_t^g) = f_t^{best}(p_t^{best})$
2.5 Increment iteration count $t = t + 1$
End while

Step 3. Result

Report best solution p_g *of the swarm with objective function value* $f(p_g)$

x_t^i = the current position of particle i in solution space and subscript t denotes an iteration count
p_t^i = best found position of particle i up to iteration count t and represents the cognitive contribution to the search velocity v_t^i
p_t^g = the global best found position among all particles in the swarm up to iteration count t and forms the social contribution to the velocity vector

Figure 4. Pseudo code of PSO algorithm

2.4. Differential evolution (DE)

DE is another most popular old generation metaheuristic algorithms (Storn & Price, 1997), and has been proved its effectiveness in addressing various real-life optimization problems. The only difference between the GA and DE is that self-adaptive DE uses mutation as its primary search methods and creates new solution strings using non-uniform crossover and tournament selection operators (Enitan & Adeyemo, 2011). DE only requires few control variables that are usually derived from numerical interval with a well-defined range. DE is easy to use because it generates new vectors without relying on an external probability density function with yet to be determined mean and standard deviation. The detail approaches of DE for solving the problem are described in Storn and Price (1997).

The pseudo code of DE is illustrated in Figure 5.

Because of ease to use nature, promising result in real-world problems DE is frequently praised in industrial environments, especially in projects where no optimization specialists are present.

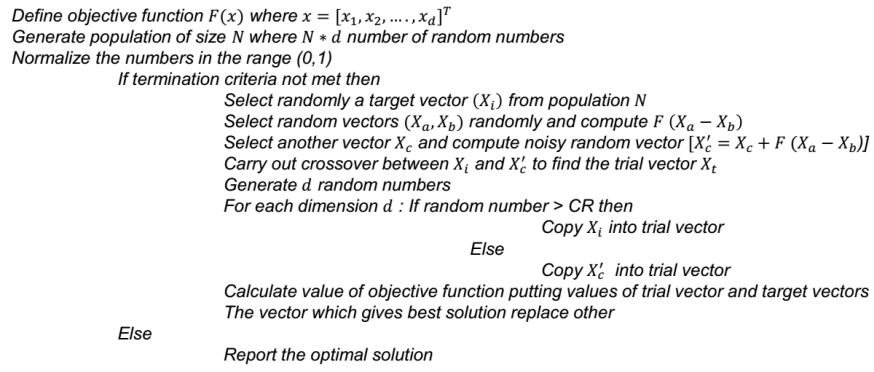

Figure 5. Flowchart of DE algorithm

3. New generation algorithms

Despite the successes of the "classical" metaheuristic algorithms, new generation metaheuristic techniques give the best solution for some unsolved hard real world problems by evolutionary or behavioral approaches (Dokeroglu et al., 2019). All major new generation algorithms developed in last 20 years are summarized schematically in Figure 1(b). In this section, a brief information about the selected algorithms is provided. All of these algorithms are population-based and take inspiration from the characteristics of the natural evolution.

3.1. Firefly algorithm (FFA)

The Firefly algorithm, which mimics the short and idealized flashing behavior of fireflies is proposed by Yang (2010a). The rhythmic flashing character can be expressed as a function that can be used to optimize combinatorial algorithms. The following principles idealize the flashing behaviors (Yang, 2010a).

(1) All fireflies are attracted to other fireflies by flashing irrespective of their sex.

(2) The attractiveness is directly related to the brightness of the firefly and both decrease with the increase in their distance.

(3) The brightness or light intensity of a firefly is affected by the search space of the optimized objective function.

The mathematical formulations are explained in detail by Yang (2010a). The pseudocode of the FFA algorithm is represented in Figure 6.

FFA has established itself as the most promising new generation algorithm and it can strike a delicate balance during the optimization process between the exploration and exploitation of search space. From the literature, it is found that it can solve a wide range of optimization problems in a versatile field.

```
Define objective function f(x) where x = (x_1, x_2, .... x_d)^T
Define γ
Generate population of fireflies whose location is x_i where the fireflies i = 1,2, … , n
while (t <Maximum Generation) do
        while (i < n) do
                while (j < n) do
                        Determine I_i at x_i with the help of f(x_i)
                        If (I_i < I_j) then
                                Firefly i moves towards firefly j in all d dimension
                        Attractiveness varies with distance
                        Calculate new solution and update intensity
        Find the best solution
```

Figure 6. Pseudocode of FFA algorithm

3.2. Krill Herd (KH)

Gandomi and Alavi proposed the KH metaheuristic in 2012. This biologically inspired novel algorithm is based on a simulation of krill herding behavior. The main advantage of the KH is that they can construct large groups. When predators (e.g., penguins, seals, or sea birds) attack a herd, they can eat individual krill and remove them from the herd, resulting in reducing the density of the krill herd and the distance of the krill from the location of the food (Gandomi & Alavi, 2012). The herding of the krill individuals is a multi-objective global optimization problem that includes two main goals: (1) Density-dependent attraction of krill (increasing krill density) and (2) reaching food (areas of high food concentration) (Gandomi & Alavi, 2012). The following actions decide the time-dependent position of an individual krill in 2D surface:

i) Motion induced by other krill individuals
ii) Foraging for food
iii) Random diffusion

A Lagrangian model is used to be able to search the whole space with n dimensions:

$$\frac{dX_i}{dt} = N_i + F_i + D_i \quad (4)$$

Where, N_i =the motion effected by other individuals, F_i =the act of foraging motion, D_i =the physical diffusion of the i-th krill individual ($i = (1,2, \dots, n)$.

The position of the individual krill depending on the above-mentioned actions is illustrated in detail in Gabdomi and Alavi (2012).

The pseudocode of KH algorithm is represented in Figure 7.

KH is widely applied to solve the global numerical optimization problems (Dokeroglu et al., 2019; Bolaji et al., 2016).

```
Describe the simple bounds, determine the algorithm parameters;
Create the initial population in the search space randomly;
Evaluate the fitness value of each krill individual according to its position;
While (Stopping criterion is not satisfied) do
        Motion effected by the presence of other krill;
        Foraging motion;
        Physical diffusion;
        Implement the genetic operators (Crossover and mutation);
        Update the krill individual position in the search space:
End
```

Figure 7. Pseudocode of KH algorithm

3.3. Grey Wolf Optimization (GWO)

Mirjalili et al. (2014) developed GWO which is a stochastic and metaheuristic optimization methodology. This bionic optimization algorithm stimulates the rank-based mechanisms and attacking behaviour of the grey wolf pack. The lead wolf helps the other wolves to capture the prey through the surrounding, haunting, and attacking process. This large-scale search methodology centred on 3 best grey wolves, but there is no elimination mechanism. The optimization technique is different from others in terms of modelling. It constitutes a strict hierarchical pyramid. The group size is 5-12 on average. α layer, consisting of a male and a female leader, is the strongest and most capable individual for deciding the team's predation actions and other activities. β and δ layers are the second and third layers respectively in the hierarchy, responsible for assisting α in the behaviour of group organizations. The bottom ranking of the pyramid is occupied by the majority of the total, named ω. They are mainly responsible for satisfying the entire pack by balancing the internal relationship of the populations, looking after the young, and maintaining the dominance structure (Mirjalili et al., 2014). The social hierarchy, encircling, hunting, attacking prey (exploitation), and searching for prey are the main key elements of the GWO model (exploration). The detail mathematical modelling of GWO has been described by Mirjalili et al. (2014).

This metaheuristic approach is applied in various real world problems because of its efficient and simple performance ability by tuning the fewest operators (Emary et al., 2016; Kohli & Arora, 2018; Mirjalili et al., 2016; Mittal et al., 2016; Qin et al., 2019). Recent researches in this regard look forward to the further development of the optimization algorithm (Niu et al., 2019). The detail of the GWO algorithm is depicted in Figure 8.

Generate the initial grey wolf populations X_i (i=1,2,...,n)
Give initial values randomly to a, A and C
Compute the fitness values of each search agent in the population
X_α= the best search agent in the population
X_β= the second best search agent in the population
X_δ= the third best search agent in the population
while (t < Maximum number of iterations)
for each search agent
Update the position of the current search agent by equation 7
end for
Update the values of a, A and C
Compute the fitness values of all search agents
Update X_α, X_β and X_δ
t =t+1
end while
return X_α

Figure 8. Pseudocode of GWO algorithm

3.4. Symbiotic organism search (SOS)

The SOS algorithm mimics the interactive behavior among different species of organisms in nature. The Greek word "symbiosis" means "living together". In nature, symbiosis defines the reliance-based interaction between any two distinct species, which may be either obligatory or facultative (Cheng & Prayogo, 2014). Therefore, in nature, symbiosis relationships can be classified as mutualism, commensalism, and parasitism. Mutualism means a symbiotic interaction between two different species that benefits them. Commensalism is a symbiotic connection that defines one can get

an advantage, and the other is neutral between two species. In parasitism, one benefits and the other is deliberately harmed (Dokeroglu et al., 2019).

The optimization approach of the SOS algorithm in the mutualism, commensalism, and parasitism phase is elaborated by Cheng and Prayogo (2014).

The pseudocode of the SOS algorithm is represented in Figure 9**.**

The new simple and powerful metaheuristic algorithm, SOS is a most potential candidate for solving hard optimization problems despite using fewer control parameters than other competing algorithms. The three phases of the SOS algorithm are simple to operate, with only simple mathematical operations to code. It can generate better solutions significantly than other metaheuristic algorithms.

```
Initialize the population
While termination criterion is not met do
        Mutualism;
        Commensalism;
        Parasitism;
End
```

Figure 9. Pseudocode of SOS algorithm

4. Result Discussion

4.1. Test problems

The selected "classical" and "new generation" metaheuristic algorithms were tested on a number of benchmark problems of mixed integer non-linear programming (MINLP) and non-linear programming (NLP) used in literature for algorithm testing purpose (Dokeroglu et al., 2019; Shopova & Vaklieva-Bancheva, 2006). While selecting the benchmark problems, attention was given that the objective function and associated constraints are very complex with multiple local optima and tried to solve before in published literature. In this study, seven of them were chosen and illustrated below. The detailed equation of objective function and associated constraint of the problems are represented in Table 1.

Table 1. Details of test problem for algorithm testing

Test problem no. 1	Floudas et al. (1989) and Summanwar et al. (2002): $MIN\ (y_1-1)^2+(y_2-2)^2+(y_3-1)^2-\log(y_4+1)+(x_1-1)^2+(x_2-2)^2+(x_3-3)^2$ Subject to the constraints: $y_1+y_2+y_3+x_1+x_2+x_3\le 5$ $y_3^2+x_1^2+x_2^2+x_3^2\le 5.5$ $y_1+x_1\le 1.2$ $y_2+x_2\le 1.8$ $y_3+x_3\le 2.5$ $y_4+x_1\le 1.2$ $y_2^2+x_2^2\le 1.64$ $y_3^2+x_3^2\le 4.25$ $y_2^2+x_3^2\le 4.64$ $x_i\ge 0, i=1\ldots.3$ $y_i\in\{0,1\}, i=1\ldots.4$
Test problem no. 2	Summanwar et al. (2002): $MIN\ 5.35785x_3^2+0.83569x_1x_5+37.2932x_1-40792.14$ Subject to the constraints: $85.3344+0.0056858x_2x_5+0.0006262x_1x_4-0.0022053x_3x_5\ge 0$ $85.3344+0.0056858x_2x_5+0.0006262x_1x_4-0.0022053x_3x_5\le 92$ $80.5125+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^2\ge 90$ $80.5125+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^2\le 110$ $9.300961+0.0047026x_3x_5+0.0012547x_1x_3+0.0019085x_3x_4\ge 20$ $9.300961+0.0047026x_3x_5+0.0012547x_1x_3+0.0019085x_3x_4\le 25$ $78\le x_1\le 102$ $33\le x_2\le 45$ $27\le x_i\le 45, i=3,\ldots,5$
Test problem no. 3	Summanwar et al. (2002): $MIN x_1^2+x_2^2+x_1x_2-14x_1-16x_2+(x_3-10)^2+4(x_4-5)^2+(x_5-3)^2+2(x_6-1)^2+5x_7^2+7(x_8-11)^2+2(x_9-10)^2+(x_{10}-7)^2+45$ Subject to the constraints: $105-4x_1-5x_2+3x_7-9x_8\ge 0$ $-10x_1+8x_2+17x_7-2x_8\ge 0$ $8x_1-2x_2-5x_9+2x_{10}+12\ge 0$ $-3(x_1-2)^2-4(x_2-3)^2-2x_3^2+7x_4+120\ge 0$ $-5x_1^2-8x_2-(x_3-6)^2+2x_4+40\ge 0$ $-x_1^2-2(x_2-2)^2+2x_1x_2-14x_5+6x_6\ge 0$ $-0.5(x_1-8)^2-2(x_2-4)^2-3x_5^2+x_6+30\ge 0$ $3x_1-6x_2-12(x_9-8)^2+7x_{10}\ge 0$ $0\le x_i\le 10, i=1,\ldots,10$
Test problem no. 4	Michalewicz (1995) and Deb (2000): $MIN 5\sum_{i=1}^{4}x_i-5\sum_{i=1}^{4}x_i^2-\sum_{i=5}^{13}x_i$ Subject to the constraints: $2x_1+2x_2+x_{10}+x_{11}\le 10$ $2x_1+2x_3+x_{10}+x_{12}\le 10$ $2x_2+2x_3+x_{11}+x_{12}\le 10$ $-8x_1+x_{10}\le 0$ $-8x_2+x_{11}\le 0$ $-8x_3+x_{12}\le 0$

	$-2x_4 - x_5 + x_{10} \leq 0$ $-2x_6 - x_7 + x_{11} \leq 0$ $-2x_8 - x_9 + x_{12} \leq 0$ $0 \leq x_i \leq 1, i = 1, \ldots, 9$ $0 \leq x_i \leq 100, i = 10,11,12$ $0 \leq x_{13} \leq 1$
Test probl em no. 5	Michalewicz (1995) and Deb (2000): $MIN x_1 + x_2 + x_3$ Subject to the constraints: $1 - 0.0025(x_4 + x_6) \geq 0$ $1 - 0.0025(x_4 + x_7 - x_4) \geq 0$ $1 - 0.01(x_8 + x_5) \geq 0$ $x_1 x_6 - 833.33252 x_4 - 100 x_1 + 83333.333 \geq 0$ $x_2 x_7 - 1250 x_5 - x_2 x_4 + 1250 x_4 \geq 0$ $x_3 x_8 - x_3 x_5 + 2500 x_5 + 1250000 \geq 0$ $100 \leq x_1 \leq 10000$ $1000 \leq x_2, x_3 \leq 10000$ $10 \leq x_i \leq 1000, i = 4, \ldots, 8$
Test probl em no. 6	Michalewicz (1995) and Deb (2000): $MIN (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11.0)^2 + 10 x_5^6 + 7 x_6^2 + x_7^4 - 4 x_6 x_7 - 10 x_6 - 8 x_7$ Subject to the constraints: $127 - 2 x_1^2 - 3 x_2^4 - x_3 - 4 x_4^2 - 5 x_5 \geq 0$ $282 - 7 x_1 - 3 x_2 - 10 x_3^2 - x_4 + x_5 \geq 0$ $196 - 23 x_1 + 3 x_1 x_2 - 2 x_3^2 - 5 x_6 + 11 x_7 \geq 0$ $10 \leq x_i \leq 10, i = 1, \ldots, 7$
Test probl em no. 7	Floudas et al. (1989) and Summanwar et al. (2002): $MAX - 2x_1 - 3x_2 - 1.5y_1 - 2y_2 + 0.5y_3$ Subject to the constraints: $x_1^2 + y_1 = 1.25; x_2^{1.5} + 1.5 y_2 = 3$ $x_1 + y_1 \leq 1.6; 1.333 x_2 + y_2 \leq 3$ $-y_1 - y_2 + y_3 \leq 0$ $x_1, x_2 \geq 0$ $y_1, y_2, y_3 \in \{0,1\}$

4.1.1. Test problem 1

It is a MINLP minimization problem that includes nine inequality constraints and four binary and three continuous variables (Floudas et al., 1989).

4.1.2. Test problem 2

It is an NLP minimization problem which comprises six inequality constraints and five continuous variables (Summanwar et al., 2002).

4.1.3. Test problem 3

It is an NLP minimization problem containing eight inequality constraints and ten continuous variables (Summanwar et al., 2002).

4.1.4. Test problem 4

It is a relatively easy problem of NLP minimization with nine inequality constraints and thirteen variables (Deb, 2000; Michalewicz, 1995).

4.1.5. Test problem 5

This NLP minimization problem comprises of six inequality constraints and eight variables (Michalewicz, 1995).

4.1.6. Test problem 6

It is an NLP minimization problem that consists four non-linear constraints and seven variables (Michalewicz, 1995).

4.1.7. Test problem 7

It is a MINLP maximization problem having two equality and three inequality constraints as well as three binary and two continuous variables (Summanwar et al., 2002).

4.2. Performance of the selected algorithms on test problems

In this study, four old generation (SA, GA, PSO, and DE) and four new generation (FFA, KH, GWO, and SOS) algorithms were chosen on the basis of their potential in diverse field that are published in large number of research papers in last 40 years. Table 2 represents the lower limit and upper limit of decision variables of the test problems. Codes are developed in MATLAB for each of eight algorithms and run in MATLAB R2017a platform. Herein, due to stochastic nature of each algorithm, each benchmark problem was run for at least 200 times to obtain the best result and maintain accuracy. The population solutions for 200 runs are summarized in a box plot (Figure 10) for test problems 1, 3, 5, and 7.

All the simulations were performed on Pentium i7 processor. The performance of selected algorithms against the tested problems is presented in Table 3 (a), (b), and (c), respectively. Their performance indices were chosen to compare their effectiveness: (1) the minimum or maximum objective function value and its proximity with reported global solution, (2) number of constraints violation (ideally all constraints should be obeyed i.e., the value should be zero), (3) required execution time to attain the optimal solution (less time is preferable).

There are many meta parameters of the individual evolutionary algorithms which need to set according to the specific problem. Judicious selections of these meta parameters improve the solution quality of individual algorithms. However, in this work we have selected the default values of these parameters as suggested by literatures (Kirkpatrick et al., 1983; Goldberg, 1989; Kennedy & Eberhart, 1995; Storn & Price, 1997; Yang, 2010a; Gandomi & Alavi, 2012; Mirjalili et al., 2014; Cheng & Prayogo, 2014). It may possible to improve the final solution reached by the individual algorithms by optimizing these meta parameters. However, this was not tried in the present study as this study focuses on the evaluation of different optimization algorithms at their default parameter settings.

Table 2. Lower limit and upper limit of decision variables

Test problem number	Limit	Decision variables												
		X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}
1	Lower	0	0	0	0	0	0	0	–	–	–	–	–	–
	Upper	1	1	1	1	2	2	2	–	–	–	–	–	–
2	Lower	78	33	27	27	27	–	–	–	–	–	–	–	–
	Upper	102	45	45	45	45	–	–	–	–	–	–	–	–
3	Lower	0	0	0	0	0	0	0	0	0	0	–	–	–
	Upper	10	10	10	10	10	10	10	10	10	10	–	–	–
4	Lower	0	0	0	0	0	0	0	0	0	0	0	0	0
	Upper	10	10	10	10	10	10	10	10	10	10	10	10	10
5	Lower	100	1000	1000	10	10	10	10	10	–	–	–	–	–
	Upper	10000	10000	10000	1000	1000	1000	1000	1000	–	–	–	–	–
6	Lower	−10	−10	−10	−10	−10	−10	−10	–	–	–	–	–	–
	Upper	10	10	10	10	10	10	10	–	–	–	–	–	–
7	Lower	0	0	0	0	0	–	–	–	–	–	–	–	–
	Upper	1	1	1	10	10	–	–	–	–	–	–	–	–

Table 3(a). Performance of algorithms for MINLP minimization problem

Problem No.	Best solution reported	Result of tested algorithm				
	Best fitness value	Algorithm	Fitness value	No. of constraints violated	Time required (sec)	% of deviation from the best value
1	**4.5795**	SA	3.5527	0	5.734	1.63
		GA	4.0780	0	0.375	16.66
		PSO	3.6381	1	39.50	4.08
		DE	3.5161	0	0.219	0.59
		FFA	3.4971	0	3.422	0.04
		KH	3.6305	0	0.844	3.86
		GWO	3.9040	0	0.391	11.68
		SOS	**3.4956**	**0**	**0.250**	**0.00**

Table 3(b). Performance of algorithms for NLP minimization problem

Problem No.	Best solution reported	Result of tested algorithm				
	Best fitness value	Algorithm	Fitness value	No. of constraints violated	Time required (sec)	% of deviation from the best value
2	**-30665.41**	SA	-30517.33	0	5.875	0.48
		GA	**-30665.50**	**0**	**0.203**	**0.00**
		PSO	-30665.55	1	33.93	0.0001
		DE	**-30665.50**	**0**	**0.203**	**0.00**
		FFA	**-30665.50**	**0**	**0.203**	**0.00**
		KH	-30427.22	0	0.453	0.77
		GWO	-30536.47	0	0.391	0.42
		SOS	-30665.54	0	0.262	0.0001
3	**24.3062**	SA	39.3890	0	8.047	58.07
		GA	32.1862	0	0.234	29.16
		PSO	25.3479	0	1.722	1.72
		DE	26.8428	0	0.250	7.72
		FFA	32.1862	0	29.164	29.16
		KH	106.9755	0	0.6406	329.30
		GWO	51.5601	0	0.4218	106.91
		SOS	**24.9187**	**0**	**0.2968**	**0.00**
4	**-15**	SA	-7.6917	0	9.703	61.03
		GA	-14.9450	0	0.938	24.29
		PSO	-14.6471	2	33.32	25.80
		DE	-14.9959	0	0.234	24.03
		FFA	99.2648	2	2.984	402.871
		KH	**-19.7396**	**0**	**3.203**	**0.00**
		GWO	-7.4812	0	0.640	62.10
		SOS	-14.9999	0	0.250	24.01
5	**7049.3309**	SA	3327.859	0	7.422	156.28
		GA	16410.230	0	0.969	1163.75
		PSO	2100.000	0	28.766	61.72
		DE	2100.000	0	0.172	61.72
		FFA	8038.478	0	2.625	519.04
		KH	**1298.535**	**0**	**0.484**	**0.00**
		GWO	2100.000	0	0.391	61.72
		SOS	2100.000	0	0.250	61.72
6	**680.6300**	SA	682.014	0	4.719	0.20
		GA	760.213	0	0.891	11.69
		PSO	**680.646**	**0**	**30.094**	**0.00**
		DE	681.521	0	0.188	0.13
		FFA	680.906	0	2.626	0.04
		KH	691.964	0	0.609	1.66
		GWO	689.110	0	0.359	1.24
		SOS	680.697	0	0.297	0.01

Table 3(c). Performance of algorithms for MINLP maximization problem

Problem No.	Best solution reported	Result of tested algorithm				
	Best fitness value	Algorithm	Fitness value	No. of constraints violated	Time required (sec)	% of deviation from the best value
7	**-7.66**	SA	-8.401	0	5.203	9.60
		GA	-0.798	1	1.359	89.59
		PSO	-0.793	1	36.469	89.65
		DE	-7.675	1	0.203	0.13
		FFA	-0.783	1	3.453	89.78
		KH	**-7.665**	**0**	**0.516**	**0.00**
		GWO	-0.797	1	0.344	89.60
		SOS	-8.416	0	0.250	9.80

4.2.1. Performance for Test problem 1

The best solution for this MINLP minimization problem was 4.97 that reported by Deb's method in the year 2000 and further Summanwar et al. (2002) has obtained the best solution of 4.5795 with modified constraints and more complicated algorithm. From the Table 3(a), and Figure 10(a), it is observed that in this study the obtained optimum solution is 3.4956 by SOS with short time among the eight selected algorithms which is far better than the reported solution. FFA also attains very close value to the optimum (3.4971) but with high execution time. Among the "classical" algorithms, DE shows the most promising result (3.5161) even with shortest execution time.

4.2.2. Performance for Test problem 2

Deb solved this NLP minimization problem with only GA and the best solution was reported -30664.99 and -30665.41 by Deb (2000) and Summanwar et al. (2002), respectively. Herein, we have obtained the best solution of -30665.50 with GA, DE and FFA that is commensurable than reported ones (Table 3(b)). Even all the three algorithms achieve the optimum value with same execution time. "New generation" metaheuristic, SOS also achieves optimal solution (-30665.54) which has very close proximity with the best one within short execution time.

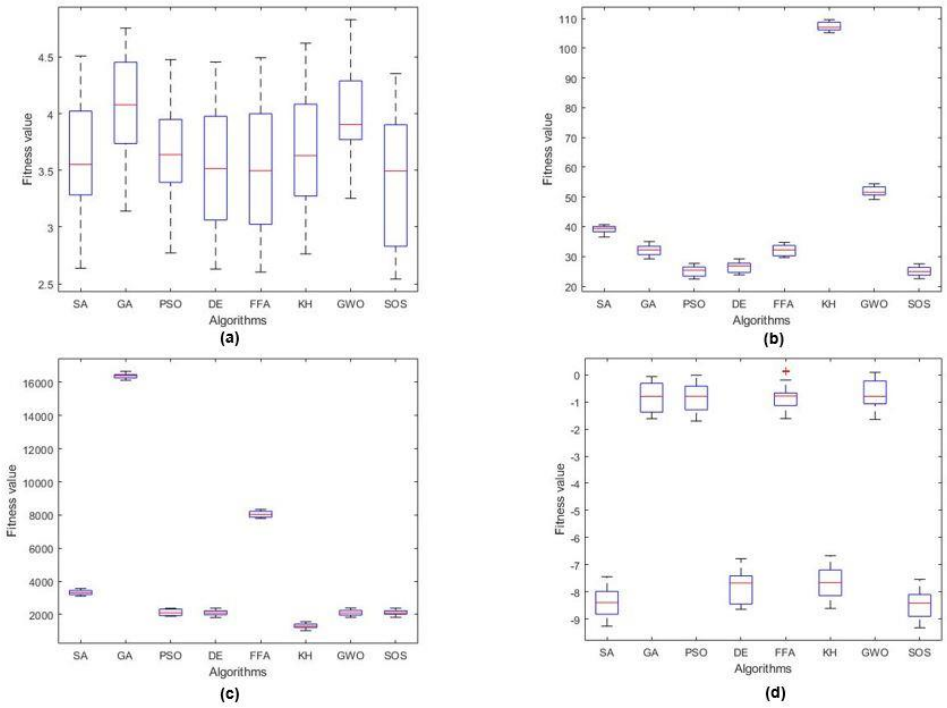

Figure 10. Box plot of solutions for (a) Test problem 1; (b) Test Problem 3; (c) Test problem 5; and (d) Test problem 7

4.2.3. Performance for Test problem 3

Deb (2000) and Summanwar et al. (2002) solved this benchmark problem and obtained the optimum solution of 24.34 and 24.366453, respectively. However, the global optimum reported of the problem is 24.3062. In this study, the best solution of 24.9187 is found by SOS algorithm with minimum time and it is also approximately around the reported value (Table 3(b) and Figure 10(b)). The old algorithm, PSO obtains the second-best optimum solution of 25.3479, followed by DE among the eight algorithms.

4.2.4. Performance for Test problem 4

Michalewics (1995) stated that all constrained handling methods used to solve this problem have found the optimal solution. In the year 2000, Deb found the optimum solution of -15 (Deb, 2000). In this study, Table 3(b) presents that for Test problem 4 the optimum solution of -19.7396 is obtained by KH algorithm in 3.203 sec that is improved sufficiently than the reported best optimum solution. Among the eight algorithms, SOS obtains the second-best optimum solution (-14.9999) at 0.250 sec. The "classical" one, DE also attain the optimum value of -14.9959 (very close proximity with second-best value) even with minimum execution time (0.234 sec).

4.2.5. Performance for Test problem 5

Michalewics reported the optimum solution of 7377.976 by solving the NLP minimization problem and further, in the year 2000, Deb attained the best solution of 7060.221 with GA. However, till now, the reported global optimum solution is 7049.3309. However, in the present study, among the eight selected algorithms, KH performs best and reaches the optimum solution of 1298.535 within a very short time (0.484 sec). Other algorithms, DE and PSO from "classical" one, and GWO and SOS from "new generation" algorithms, obtain the second-best solution, the value of 2100.00 with different execution time. Among those, DE takes minimum execution time followed by SOS and GWO, whereas, PSO requires highest execution time (28.766 sec). The detail of result is presented in Table 3(b) and Figure 10(c).

4.2.6. Performance for Test problem 6

The reported best solution for this NLP minimization problem is 680.6300. Michalewics found the best result of 680.642 in 1995 with penalty function approach (Michalewicz, 1995) and Deb reported the best solution as 680.634 by constrained handling method (Deb, 2000). Herein, for Test problem 6, Table 3(b) presents that the true optimal solution of 680.646 is obtained by PSO algorithm that is very close to the reported best one but the execution time is too high. In contrast, SOS and FFA attain the fitness values of 680.697 and 680.906 respectively, which are very close to the best one even with very short time. Among the "classical" algorithm, DE reaches the optimum value of 681.521 (little bit high than the best value) within very less execution time compare to PSO.

4.2.7. Performance for Test problem 7

The best solutions obtained by Deb's method and Summanwar et al. are -7.66718 and -7.667178, respectively for this MINLP maximization problem. Therefore, the global optimum of -7.66 is reported as best-known solution. Eight selected algorithms perform very well against this maximization problem and among all, KH improves the result most and reaches the optimum solution of -7.665 in execution time of 0.516. All other algorithms except SA, and SOS, fail to solve this problem as they violate one constraint. On the other hand, SA and SOS unable to obtain the optimum solution far better than the reported one. The detail performance of the algorithms and related box plot are presented in Table 3(c) and Figure 10(d), respectively.

5. Overall observation on performance

While comparing the performance of various optimization algorithms on test problems, we give priority of objectives as per below:

Priority 1: Algorithms attain the lowest (or highest) objective function value.

Priority 2: If two or more algorithms attain the optimum value simultaneously, then their lower execution time given priority.

Priority 3: If any algorithm fails to obey the any constraint that will make it ineligible candidate solution.

Based on the above criteria, performances of eight algorithms on the seven selected benchmark problems are compared that are presented in Table 4 and following concluding remarks are provided:

1) Comparing the best results of optimal values by eight selected algorithms, it can be observed that there is no single optimization algorithm which universally performs best on all the seven selected benchmark problems.

2) From the Table 4, it is shown that among the selected eight metaheuristic algorithms, KH has achieved the best solution for three benchmark problems. Whereas, SOS has performed best for two and FFA, GA, DE, and PSO each have executed best result for one test problem.

3) While comparing the performance of "classical" and "new generation" metaheuristic algorithms, new generation algorithms performed much better (selected as best performer for six times among seven benchmark problem).

4) Among four "Classical" algorithms, DE, and PSO effectively attains the optimal solution which are very close to the best one. In spite of its accurate performance ability, PSO takes much more time than any other algorithms. DE, PSO and SA performed best for four, two and one selected test problems, respectively.

5) Among the selected "new generation" algorithm, SOS and KH, each performed best for three benchmark problems and FFA showed its best performance for one. Therefore, SOS and KH are most promising among new generation algorithms.

6) If top two global performer are considered then out of 14 performers, the ranking based on their performance is as follows:

SOS (5 times)>KH (3 times)>FFA=DE=PSO (2 times)>GA (1 time)

Though the KH achieved the best solution for one more benchmark problem compared to SOS (already mentioned in observation number 2), but the ranking proved that the consistency of SOS performance is better than KH.

However, these observations are not universal and the performance of the algorithms may change for other benchmark problems.

Table 4. Overall performance of the selected metaheuristic algorithms

Test Problem Number	Top two performer of "classical" algorithms*	Top two performer of "new generation" algorithms*	Top two global performer among "classical" and "new generation" algorithms*
1	DE, SA	SOS, FFA	SOS, FFA
2	DE, GA	FFA, SOS	FFA, DE, GA**
3	PSO, DE	SOS, FFA	SOS, PSO
4	DE, GA	KH, SOS	KH, SOS
5	DE, PSO	KH, SOS	KH, DE
6	PSO, DE	SOS, FFA	PSO, SOS
7	SA	KH, SOS	KH, SOS

*Arrange them in decreasing order on the basis of performance
**All are in same order of sequence

From the results of all algorithms on all the test problems, it is evident that there is no single optimization algorithm that universally performs best on all the seven selected benchmark problems. It supports the facts of No Free Lunch Theorem (NFLT) (Wolpert & Macready, 1997).

According to the recently discovered No Free Lunch Theorem (NFLT) (Adam et al., 2019; Ho & Pepyne, 2001, 2002; Wolpert & Macready, 1997), no strategy can be predicted to outperform another if we are unable to make any previous assumptions about the optimization problem we are attempting to solve. In other words, there is

no such thing as a general-purpose universal optimization approach. One technique may outperform another only if it is tailored to the problem at hand.

What can we assume about likely problem instances, what structural qualities the assumptions suggest, what technique is best matched to that structure, and how sensitive our strategy is to the assumptions are the fundamental questions in optimization practice. Therefore, we still need to conduct a great deal of research and develop a better understanding of the implications of "No free lunch Theorem" to find a universal metaheuristic optimization algorithm.

6. Conclusions

In recent years, metaheuristic algorithms were successfully being applied for solving the intractable optimizing problems. The majority of state-of-the-art metaheuristic have been developed before the year 2000 ("classical" algorithms) and then they are become more and more advanced improving their performance and execution time ("new generation" algorithms). The novelty of the present study was to test the capability of applications and compare their performance of the four selected algorithms from "classical" and "new generation" each by solving a number of selected benchmark problems that are used in the literature for algorithm testing purpose and ultimate aim was to find out the universally best algorithm among the selected eight metaheuristic algorithms. However, there was no such universally best algorithm which will perform best in different problem statement. The "new generation" SOS and KH algorithm successfully solved most of all the selected test problems and achieved the best solution for most of them. Among four "Classical" algorithms, DE, and PSO effectively attained the optimal solution which were very close to the best one. Based on the result obtained on the present study, new generation performed much better than old generation. It can be concluded on the basis of the performance of different algorithms that both SOS and KH exhibited the most promising result and great potential with respect to execution time also. This study gives some insights to use SOS and KH as best performing algorithm to the novice user who can easily get lost by the plethora of large number of optimization algorithms.

Author contributions: Tithli Sadhu: Formal analysis, Methodology, Writing-original draft. Somnath Chowdhury: Conceptualization, Methodology, Writing-review & editing. Shubham Mondal: Methodology, Formal analysis. Jagannath Roy: Writing-review & editing. Jitamanyu Chakrabarty: Writing-review & editing. Sandip Kumar Lahiri: Conceptualization, Writing-review & editing, Supervision. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgement: Tithli Sadhu and Somnath Chowdhury would like to thank NIT Durgapur for their Ph.D. fellowship and educational support.

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

References

Adam, S. P., Alexandropoulos, S. A. N., Pardalos, P. M., & Vrahatis, M. N. (2019). No free lunch theorem: A review. In Demetriou, I., Pardalos, P. (Eds.), Approximation and Optimization. Springer Optimization and Its Applications (pp. 57–82). Springer, Cham.

Bai, Q. (2010). Analysis of particle swarm optimization algorithm. Computer and information science, 3(1), 180–184.

Banzhaf, W., Nordin, P., Keller, R. E., & Francone, F. D. (1998). Genetic programming: an introduction: on the automatic evolution of computer programs and its applications. (1st ed.). Morgan Kaufmann Publishers Inc, (Chapter 5).

Beheshti, Z., & Shamsuddin, S. M. H. (2013). A review of population-based meta-heuristic algorithms. International Journal of Advances in Soft Computing and its Applications, 5(1), 1–35.

Bolaji, A. L. A., Al-Betar, M. A., Awadallah, M. A., Khader, A. T., & Abualigah, L. M. (2016). A comprehensive review: Krill Herd algorithm (KH) and its applications. Applied Soft Computing, 49, 437–446.

Cheng, M. Y., & Prayogo, D. (2014). Symbiotic organisms search: a new metaheuristic optimization algorithm. Computers & Structures, 139, 98–112.

Das, S., Biswas, A., Dasgupta, S., & Abraham, A. (2009). Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. In Abraham, A., Hassanien, AE., Siarry, P., Engelbrecht, A. (Eds.), Foundations of Computational Intelligence (pp. 23–55). Springer, Berlin, Heidelberg.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. Computer methods in applied mechanics and engineering, 186(2–4), 311-338.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. Computers & Industrial Engineering, 137, 106040.

Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic, in Proceedings of the 1999 congress on evolutionary computation-CEC99. IEEE, 1470–1477.

Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. Neurocomputing, 172, 371–381.

Enitan, A. M., & Adeyemo, J. (2011). Food processing optimization using evolutionary algorithms. African Journal of Biotechnology, 10(72), 16120–16127.

Floudas, C., Aggarwal, A., & Ciric, A. (1989). Global optimum search for nonconvex NLP and MINLP problems. Computers & Chemical Engineering, 13(10), 1117–1132.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: a new bio-inspired optimization algorithm. Communications in nonlinear science and numerical simulation, 17(12), 4831–4845.

Geem, Z.W., Kim, J. H., & Loganathan, G.V. (2001). A new heuristic optimization algorithm: harmony search. Simulation, 76(2), 60–68.

Glover, F., & Laguna, M. (1998). Tabu Search. In Du DZ., & Pardalos P.M. (Eds.), Handbook of Combinatorial Optimization (pp. 2093–2229). Springer, Boston, MA.

Goldberg, D.E. (1989). Genetic algorithms in search, Optimization, and Machine Learning. (1st ed.). Addison-Wesley Publishing Company, (Chapter 3).

Ho, Y. C., & Pepyne, D. L. (2001, December). Simple explanation of the no free lunch theorem of optimization, in Proceedings of the 40th IEEE Conference on Decision and Control. IEEE, 4409–4414.

Ho, Y. C., & Pepyne, D. L. (2002). Simple explanation of the no-free-lunch theorem and its implications. Journal of optimization theory and applications, 115(3), 549–570.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization, in Proceedings of ICNN'95-international conference on neural networks. IEEE, 1942–1948.

Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P. (1983). Optimization by simulated annealing. Science, 220(4598), 671–680.

Kohli, M., & Arora, S. (2018). Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of computational design and engineering, 5(4), 458–472.

Lahiri, S. K., & Ghanta, K. C. (2009). Development of a hybrid artificial neural network and genetic algorithm model for regime identification of slurry transport in pipelines. Chemical Product and Process Modeling, 4(1), Article 22. **DOI:** 10.2202/1934-2659.1343.

Lahiri, S. K., & Ghanta, K. C. (2010). Artificial neural network model with parameter tuning assisted by genetic algorithm technique: study of critical velocity of slurry flow in pipeline. Asia-Pacific Journal of Chemical Engineering, 5(5), 763–777.

Lahiri, S. K., Khalfe, N. M., & Wadhwa, S. K. (2012). Particle swarm optimization technique for the optimal design of shell and tube heat exchangers. Chemical Product and Process Modeling, 7(1), Article 14. **DOI:** 10.1515/1934-2659.1612.

Li, B., & Jiang, W. (1997). Chaos optimization method and its application. Control theory and application, 14(4), 613–615.

Marques-Silva, J. P., & Sakallah, K. A. (1999). GRASP: A search algorithm for propositional satisfiability. IEEE Transactions on Computers, 48(5), 506–521.

Michalewicz, Z. (1995). Genetic algorithms, numerical optimization, and constraints. In Eshelman, L. (Eds.), Proceedings of the Sixth International Conference on Genetic Algorithms (pp. 151–158). Morgan Kauffman, San Mateo.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. Advances in engineering software, 95, 51–67.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46–61.

Mirjalili, S., Saremi, S., Mirjalili, S. M., & Coelho, L. D. S. (2016). Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. Expert Systems with Applications, 47, 106–119.

Mittal, N., Singh, U., & Sohi, B. S. (2016). Modified grey wolf optimizer for global engineering optimization. Applied Computational Intelligence and Soft Computing, 2016, 7950348.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. Computers & operations research, 24(11), 1097–1100.

Niu, P., Niu, S., & Chang, L. (2019). The defect of the Grey Wolf optimization algorithm and its verification method. Knowledge-Based Systems, 171, 37–43.

Qin, H., Fan, P., Tang, H., Huang, P., Fang, B., & Pan, S. (2019). An effective hybrid discrete grey wolf optimizer for the casting production scheduling problem with multi-objective and multi-constraint. Computers & Industrial Engineering, 128, 458–476.

Rangaiah, G.P. (2010). Stochastic Global Optimization: Techniques and Applications in Chemical Engineering. (1st ed.). World Scientific, (Chapter 3).

Rao, S. S. (2019). Engineering optimization: theory and practice. (5th ed.). Wiley, (Chapter 13).

Shopova, E. G., & Vaklieva-Bancheva, N. G. (2006). BASIC- a genetic algorithm for engineering problems solution. Computers and Chemical Engineering, 30(8), 1293–1309.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), 341–359.

Summanwar, V., Jayaraman, V., Kulkarni, B., Kusumakar, H., Gupta, K., & Rajesh, J. (2002). Solution of constrained optimization problems by multi-objective genetic algorithm. Computers & Chemical Engineering, 26(10), 1481–1492.

Wang, F. S., & Chen, L. H. (2013). Heuristic optimization. In Dubitzky, W., Wolkenhauer, O., Cho, KH., Yokota, H. (Eds.), Encyclopedia of Systems Biology (pp. 885–885). Springer, New York.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1(1), 67–82.

Yang, X. S. (2010a). Firefly algorithm, stochastic test functions and design optimisation. International journal of bio-inspired computation, 2(2), 78–84.

Yang, X. S. (2010b). A new metaheuristic bat-inspired algorithm. In González J.R., Pelta D.A., Cruz C., Terrazas G., & Krasnogor N. (Eds.), Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) (pp. 65–74). Springer, Berlin, Heidelberg.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights, in 2009 World congress on nature & biologically inspired computing (NaBIC). IEEE, 210–214.