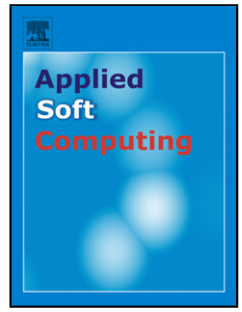
Accepted Manuscript



Title: An artificial algae algorithm with stigmergic behavior for binary optimization

Authors: Sedat Korkmaz, Mustafa Servet Kiran

PII: S1568-4946(18)30006-1
DOI: https://doi.org/10.1016/j.asoc.2018.01.001
Reference: ASOC 4647

To appear in: *Applied Soft Computing*

Received date: 8-10-2017
Revised date: 25-12-2017
Accepted date: 2-1-2018

Please cite this article as: Sedat Korkmaz, Mustafa Servet Kiran, An artificial algae algorithm with stigmergic behavior for binary optimization, Applied Soft Computing Journal https://doi.org/10.1016/j.asoc.2018.01.001

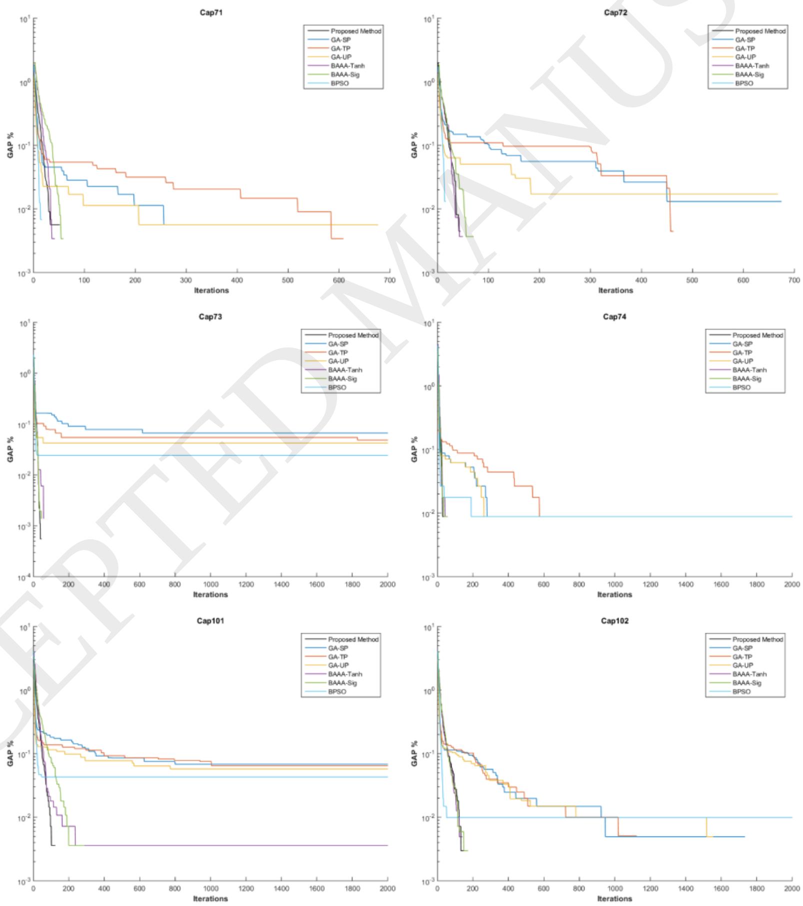
An Artificial Algae Algorithm with Stigmergic Behavior for Binary Optimization

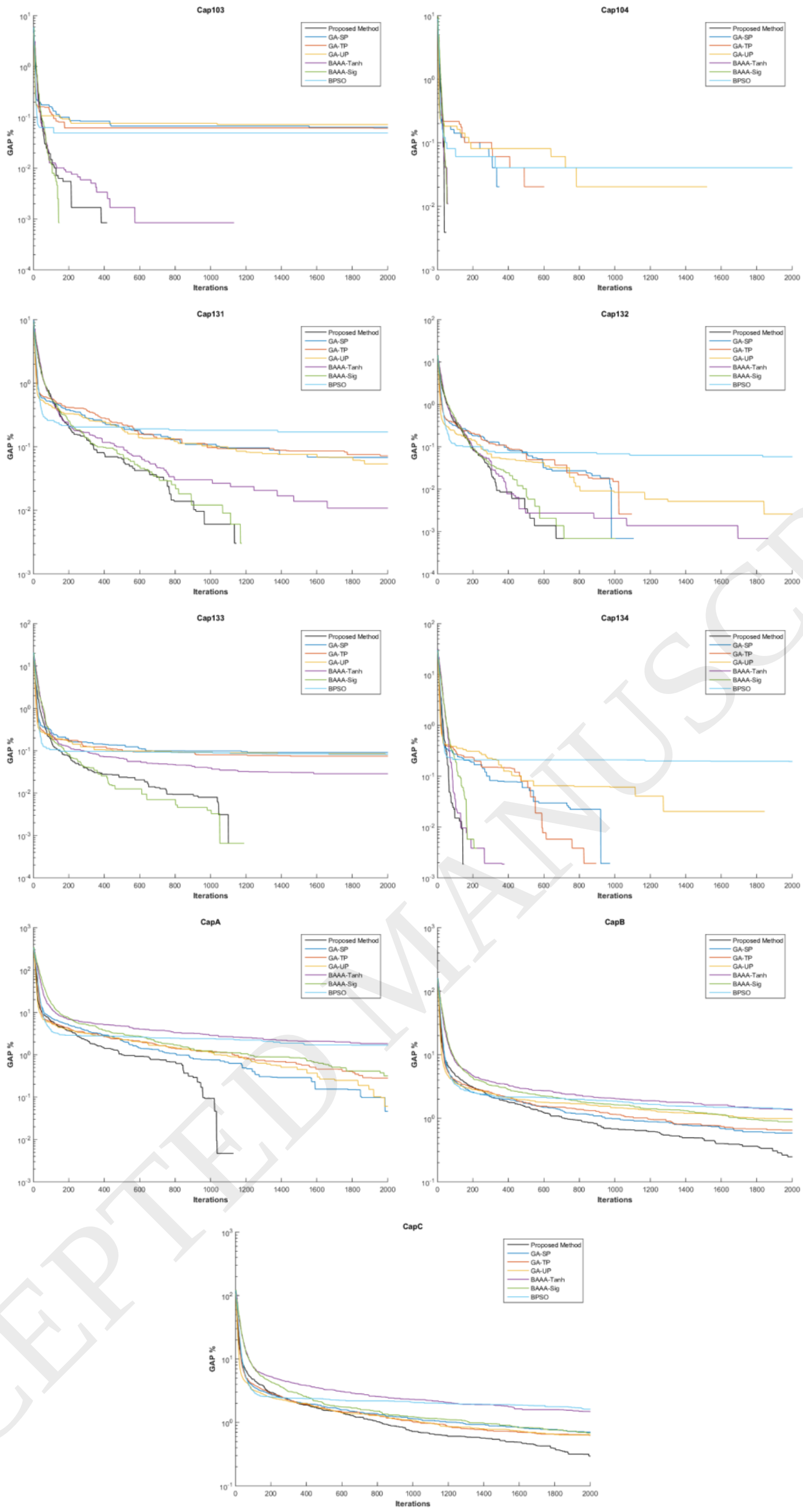
Sedat Korkmaz*, Mustafa Servet Kiran

Department of Computer Engineering, Faculty of Engineering, Selcuk University, 42075, Konya, Turkey

*Corresponding Author.
Tel : 0090 332 223 8644
Fax: 0090 332 241 06 35
E-mail: sedatkorkmaz@selcuk.edu.tr

Graphical Abstract





Highlights

- A binary version of artificial algae algorithm is proposed.
- The position update rule is replaced with xor logic operator.

- The stigmergic behavior is integrated in this version.
- The proposed algorithm is applied to two different problem set.
- Obtained results are compared with state-of-art methods.
- The proposed algorithm is quite competitive and successful in solving binary problems.

Abstract

In this study, we focus on modification of the artificial algae algorithm (AAA), proposed for solving continuous optimization problems, for binary optimization problems by using exclusive-or (xor) logic operator and stigmergic behavior. In the algorithm, there are four processes sequentially realized for solving continuous problems. In the binary version of the algorithm, three of them are adapted in order to overcome the structure of binary optimization problems. In the initialization, the colonies of AAA are set to either zero or one with equal probability. Secondly, helical movement phase is used for obtaining candidate solutions and in this phase, the xor operator and stigmergic behavior are utilized for obtaining binary candidate solutions. The last modified phase is adaptation, and randomly selected binary values in the most starved solution are likened to biggest colony obtained so far. The proposed algorithm is applied to solve well-known uncapacitated facility location problems and numeric benchmark problems. Obtained results are compared with state-of-art algorithms in swarm intelligence and evolutionary computation field. Experimental results show that the proposed algorithm is superior to other techniques in terms of solution quality, convergence characteristics and robustness.

Keywords: artificial algae algorithm, binary optimization, stigmergy, benchmark problem

1. Introduction

In real world, problems such as ambulance station location problem [1, 2], fire station location problem [3, 4], feature selection[5-9], unit commitment problem[10-14], knapsack problems(KP)[15-18] and facility location problems[19-22], encountered in a large number of cases, are modeled as a binary optimization problem. Binary optimization (BO) is a subfield of the discrete optimization and they are generally in class of NP-Hard or NP-complete. Solving this type of problems by using classical methods can be computationally expensive due to their discretely structured large solution space and a huge number of possible solutions. Therefore, we need effective and efficient optimization algorithms to solve this type of problems. Swarm intelligence or evolutionary computation algorithms are alternative solvers in field of discrete optimization and they are problem-independent algorithms. Due to their simple structures, ease of implementation and producing acceptable level of solutions in a reasonable time, they have been attracted interest of the researchers and practitioners.

One of the popular algorithms in evolutionary computation literature is Genetic Algorithms (GAs) [23-25] for solving the optimization problems with discrete or continuously structured solution space. The GA aims to develop solutions using the natural selection, reproduction and mutation operators on the populations of individuals. As the population improves from generation to generation, bad solutions tend to disappear, and good solutions tend to be used to create better solutions because the selection mechanisms defined in GA. Crossover techniques commonly used in the literature for reproduction are single-point, two-point and uniform types [26-28].

The artificial bee colony optimization algorithm (ABC) [29] has been invented by Karaboga in 2005 to solve continuous optimization problems. The motivation of ABC is based on the behaviors of honey bee colonies such as nectar collecting and waggle dance for sharing information about nectar sources. Binary artificial bee colony algorithm (binABC) has been proposed for solving binary problems by Kiran et al. in [30]. They propose xor operator for the solution update mechanism of the basic ABC algorithm. In another binary version of ABC, called as ABC_{bin} [31], suggested a modulo-based transfer function.

Kennedy and Eberhart [32] proposed the particle swarm optimization algorithm (PSO) in 1995, inspired by the food search behavior of birds and fish swarms. According to the PSO algorithm, each particle follows the best solution of the population and personal best solution obtained so far. The binary version of the PSO algorithm called BPSO [33], also investigated by Kennedy and Eberhart in 1997. They used a sigmoid limiting transformation function for converting continuous solutions to discrete binary solutions. In another study, artificial immune systems was used with in the PSO for solving binary optimization problems[34]. The effects of transfer functions for PSO are also analyzed on the numeric benchmark problems in [35-37].

Artificial Algae Algorithm (AAA) [38, 39] has recently been proposed to solve continuous optimization problems and is inspired by the vital behavior of microalgae. The performance of AAA was evaluated on the IEEE-CEC’13[40] and IEEE-CEC’05[41] benchmark sets as real-parameter optimization, IEEE-CEC’11[42] benchmark set and Pressure Vessel Design Optimization Problem[43, 44] as real-world optimization problems.

Test results show that AAA algorithm produces better results compared to other the state of the art algorithms.

AAA, which operates on the continuous solution space and cannot be applied to the problems which operates on the discrete solution space. For this reason the algorithm needs to be adapted to work in discrete solution space. The binary version of AAA, called BAAA [45] is proposed for solving multidimensional knapsack problems, one of the binary optimization problems in the literature. In this study, the authors use logistics functions such as sigmoid and tangent hyperbolic for binarization.

Another popular optimization algorithm, differential evolution algorithm(DE) [46] was proposed Storn and Price in 1997 for solving continuous optimization problems. The binary versions of the DE algorithm were adapted using angle modulation technique to solve binary optimization problems in [47, 48]. In another study, called SabDE[49], authors proposed a new self-adaptive binary version of DE algorithm, based on measure of dissimilarity. They used a probability-based search mechanism among the conventional search strategies in strategy adaptation phase and utilized a logical gate-based operators such as *and*, *or* and *xor* in vector generation phase. Afterwards, they compared SabDE algorithm against the state of the art algorithms such as Simplified Binary Harmony Search Algorithm (SBHS)[50] based on Harmony Search Algorithm (HS) [51], Binary Learning Differential Evolution Algorithm (BLDE)[52], Binary Hybrid Topology Particle Swarm Optimization Algorithm (BHTPSO-QI)[53], Genetic Operators Based Artificial Bee Colony Algorithm (GBABC)[54] and Binary Quantum-Inspired Gravitational Search Algorithm (BQIGSA)[55] based on Gravitational Search Algorithm (GSA) [56].

In this study, we focus on developing a binary version of AAA using a logic operator and stigmergic behavior. In the algorithms and studies given in literature review, the stigmergic behavior is novel for working AAA and it guides to the solution update mechanism in the algorithm. For experimentally validated the performance of the proposed algorithm, a comprehensive test suite and comparisons are conducted.

2. Basic Artificial Algae Algorithm

The AAA is one of the swarm intelligence algorithms, inspired by the food search behavior of the algae colonies. On a D-dimensional solution space, each alga cell corresponds to a dimension of a solution and an algae colony corresponds to a possible solution for an optimization problem. Colonial food search behavior of algae is based on the movement together towards a food source. By modeling this colonial behavior, Uymaz et al. [38]

proposed the AAA to solve continuous optimization problems in 2015. The detailed information on relation between the algorithmic model and the behavior or real algae can be found in [38]. In this algorithm, let N is the number of the algae colonies and D is the number of decision variables (dimensions) of the optimization problem, the algae colonies can be modeled as $N \times D$ matrix, each cell of this matrix corresponds to a possible solution and each row corresponds to a possible solution of the optimization problem. In the basic AAA, there are four phases, called as initialization, helical movement, evolutionary and adaptation phases. Which are sequentially realized. In the initialization of the basic AAA, all the cells are generated by using Eq.1.

$$X_{i,j} = L_{i,j} + r_{i,j} \times (H_j - L_j) \quad i = 1, 2, \ldots, N \ and \ \ j = 1, 2, \ldots, D \tag{1}$$

Where, $X_{i,j}$ is the jth cell of ith algae colony, $r_{i,j}$ is a random number produced in range of [0,1], H_j and L_j are the upper and lower bounds of the search space. The fitness of the algae colonies are calculated by using objective function value of the colonies which is specific for the optimization problem given in Eq.2.

$$fit_i = 1 - \frac{Obj_i - worst(\vec{Obj})}{best(\vec{Obj}) - worst(\vec{Obj})} \tag{2}$$

Where, fit_i is the fitness of the ith algae colony, $best(\vec{Obj})$ and $worst(\vec{Obj})$ are the best and worst objective function values among the algae colonies, respectively. If a minimization is aimed on solving optimization problem, the best solution in the algae colonies is the lowest value among the objective function values and the worst solution in the algae colonies is the highest value among the objective function values.

The sizes of the algae colonies are important in the AAA algorithm because there is an evolutionary process is performed in this algorithm. In the initialization of the algorithm, the sizes of the algae colonies are set to 1 and size of an algae colony is calculated during the iterations given as follows:

$$G_i(t+1) = G_i(t) + \mu_i(t) \times G_i(t) \tag{3}$$

$$\mu_i(t) = \frac{fit_i(t)}{K_i(t) - fit_i(t)} \tag{4}$$

$$K_i(t) = G_i(t) / 2 \tag{5}$$

where, $K_i(t)$ is the substrate half saturation constant of the algal colony, $\mu_i(t)$ is the specific grow rate of ith algae colony.

After the algae colonies are generated, their fitness are calculated and their sizes are obtained, all colonies try to find new food sources by using helical movement. For helical movement, three equations are calculated as follows:

$$X_{i,j}(t+1) = X_{i,j}(t) + (X_{i,j}(t) - X_{k,j}(t)) \times (\Delta - \tau(X_i)) \times \rho \tag{6}$$

$$X_{i,m}(t+1) = X_{i,m}(t) + (X_{i,m}(t) - X_{k,m}(t)) \times (\Delta - \tau(X_i)) \times \sin(\beta) \tag{7}$$

$$X_{i,n}(t+1) = X_{i,n}(t) + (X_{i,n}(t) - X_{k,n}(t)) \times (\Delta - \tau(X_i)) \times \cos(\alpha) \tag{8}$$

where, X_i and X_k denote the ith and kth algae colony, respectively, and i and k indices are different from each other. The indices j, m and n are the cells of X_i and X_k colonies. ρ is a uniform random number produced in range of [-1,1], α and β are the angles produced randomly in range of $[0, 2\pi]$. Δ is the shear force which is a control parameter of AAA and it is set in the initialization of the algorithm. $\tau(X_i)$ is the friction surface of ith colony and calculated as follows:

$$\tau(x_i) = 2\pi\left(\sqrt[3]{\frac{3G_i}{4\pi}}\right)^2 \tag{9}$$

The helical movement is operated by each alga colony until the energy of this colony is end. There are two energy consumption reasons in an alga colony. First is originated from the metabolism of the alga colony or cells and second is originated from the movement. Energy of an alga colony is decreased as follows:

$$E(X_i) = E(X_i) - (e / 2) \tag{10}$$

where, $E(X_i)$ is the total energy of the ith colony and e is a control parameter of the algorithm initialized at the beginning of the algorithm. If there is a movement, the ith alga colony loses some energy and it is calculate using Eq. 10. If new location of the alga is better than the old one, this alga does not lose any amount of energy which is originated from the metabolism because the food sources in the new location are exploited. Otherwise, some amount of energy originated from the metabolism is lost and this is obtained using Eq.10. Until energy of an alga is over, this colony searches for new food sources. If a better quality of solution is not obtained by this colony until energy of this colony is over, the starvation of the colony is increased by 1.

After energy of all the colonies is over, the sizes of the colonies are re-calculated by using Eq. 3. At this time, a simple evolutionary process is performed between the biggest and smallest colonies in accordance with the size of these colonies. In this process, the randomly selected cell from the biggest colony is copied to the smallest colony and it should be noted that the cell locations in the biggest and smallest colonies are the same.

In the adaptation phases of AAA, the most starveling colony is fixed by using starvation counter. This colony is likened to the biggest colony in the algae colonies by using Eq. 11.

$$X_{s,d} = X_{s,d} + r_d \times (X_{b,d} - X_{s,d}) \qquad d = 1, 2, \ldots, D \tag{11}$$

where, $x_{s,d}$ is dth cell of the most starveling colony s, $x_{b,d}$ is the dth cell of the biggest colony and r_d is a random number produced in range of [0,1].

After the all the phases of the algorithm are completed, the best solution or algae colony is determined from the algae colonies in order to report end of the run of the algorithm. The pseudo-code of the AAA is briefly given in Figure 1.

3. Proposed Method

In the present work, two update mechanism is proposed for solving binary structured problems. First approach uses logic XOR operator for producing candidate solutions and second approach is based on stigmergic behavior. Eq.12 is utilized for determining which update mechanism is used for obtain a candidate solution.

$$Update\ mechanism = \begin{cases} UM-1, & if\ (r < UMSP)\ and\ C_{01}(t) \neq 0\ and\ C_{10}(t) \neq 0 \\ UM-2, & otherwise \end{cases} \tag{12}$$

where, *UM-1* is the update mechanism 1, *UM-2* is the update mechanism 2, *UMSP* is the update mechanism selection probability as a control parameter and *r* is a random number produced in range of [0,1]. C_{01} and C_{10} are integer numbers and described in next sections.

3.1. Update Mechanism 1: XOR-based Binary Artificial Algae Algorithm

The first version of the AAA algorithm has been proposed for solving the optimization problems with continuous-structured solution space and it must be modified for solving the binary optimization problems due to the fact that decision variables of these problem can be an element of set {0,1}. In order to address this issue, three points which are initialization of the algorithm, solution update rules (helical movement) and adaptation process in basic AAA are modified. Evaluation phase of the AAA algorithm is suitable for operating with binary values, so it is applied as in the simple version of the AAA algorithm, without any modification. The XOR-based binarization processes is used with some modifications for AAA algorithm as in ABC algorithm[30]. In the initialization phase of the AAA algorithm, Eq.1 of basic AAA is modified given as follows:

$$X_{i,j} = \begin{cases} 0\,, if\ (r_{i,j} < a) \\ 1\,, otherwise \end{cases} \quad , i = 1,2,\dots,N\ and\ j = 1,2,\dots,D \tag{13}$$

where, *a* is the probability value taken as 0.5 for random distribution of the decision variables values, $r_{i,j}$ is a random number produced in range of [0,1] and $X_{i,j}$ is the *j*th cell (dimension) of the *i*th individual (algae colony) in the population. After the initialization of the algae colony, Eq.6, Eq.7 and Eq.8 for helical movement phase are modified as in [30] and given below:

Let $V = X_i$

$$V_j = X_{i,j} \oplus [\, \varphi\, (X_{i,j} \oplus X_{n,j})\,] \tag{14}$$

$$V_k = X_{i,k} \oplus [\, \varphi\, (X_{i,k} \oplus X_{n,k})\,] \tag{15}$$

$$V_l = X_{i,l} \oplus [\, \varphi\, (X_{i,l} \oplus X_{n,l})\,] \tag{16}$$

$$i,n \in \{1,2,\dots,N\}\ ,\ \ i \neq n\ ,\quad j,k,l \in \{1,2,\dots,D\}\ ,\ j \neq k \neq l$$

where, *V* is the candidate solution, *X* is the individual in the population, $\oplus$ is the logic XOR operator, φ is the logic NOT operator with 50% probability, *N* is the population size and *D* is the dimensionality of the problem. Therefore V_j determines the jth dimension, V_k determines the kth dimension and V_l determines the lth dimension of the candidate solution. In the solution update mechanism, three dimensions are updated as in the basic version of the AAA algorithm.

After the helical movement phase, the fitness value of the candidate solution is compared with the old one. If fitness value of the candidate solution is better than the old one, the candidate solution is copied to the old alga colony and the information to be used for the second update mechanism is obtained at this stage. Decision variables of old and new solutions are compared one by one and changed decision variables are determined. If the old value of the decision variable is 1 and the new value of the decision variable is 0, the C_{10} counter is incremented by 1. On the contrary, if the old value of the decision variable is 0 and the new value of the decision variable is 1, the C_{01} counter is incremented by 1. In other words, C_{10} indicates the count of changed value 1 to 0 and C_{01} indicates the count of changed value 0 to 1. C_{01} and C_{10} counters are calculated given as follows:

$$C_{01}(t+1) = \begin{cases} C_{01}(t) + 1\ , Obj(V) < Obj(X_i)\ and\ X_{i,d} = 0\ and\ V_d = 1 \\ C_{01}(t) \qquad\quad , otherwise \end{cases}, \forall \mathrm{d} \in \mathrm{P} \tag{17}$$

$$C_{10}(t+1) = \begin{cases} C_{10}(t) + 1\ , Obj(V) < Obj(X_i)\ and\ X_{i,d} = 1\ and\ V_d = 0 \\ C_{10}(t) \qquad\quad , otherwise \end{cases}, \forall \mathrm{d} \in \mathrm{P} \tag{18}$$

$$P = \{j, k, l\}$$

where, V is the candidate solution, X is the individual in the population, *Obj* is the objective function of the problem. We also mention that, $Obj(V) < Obj(X_i)$ in the equation is for minimization problems and it must be reversed for maximization problems.

In the adaptation phase of the algorithm, another modification is performed for working with the binary decision variables. In the basic version of the AAA, if the random number produced in range of [0,1] is less than the adaptation control parameter, adaptation phase is employed and the most starveling colony is likened to the biggest colony in the algae colonies by using Eq. 11. In our approach, adaptation control parameter is used for all of the cells (dimension) of the biggest and the most starveling colony. The modified equation is given as follows:

$$X_{s,z} = \begin{cases} X_{b,z}\ , if\ (r_z < Ap\ ) \\ X_{s,z}\ , otherwise \end{cases} \quad \forall z \in D\ \ , z = 1,2, \dots , D \tag{19}$$

where, $X_{s,z}$ is zth cell of the most starveling colony *s*, $X_{b,z}$ is the zth cell of the biggest colony *b*, r_z is a random number produced for dimension *z* in range of [0,1] and *Ap* is the adaptation control parameter. For better understanding the method, the pseudo-code of the Update Mechanism 1 is briefly given in Figure 2.

3.2. Update Mechanism 2: Stigmergic Behavior

The concept of stigmergy was first introduced by Grassé[57] in 1959 in the field of entomology. One of the purpose of his work, is to find out how the simple individuals of termites are able to create the grand termite mounds. The French biologist Grassé stated that termites tended to follow very simple rules when constructing their nests[58]. The actions of termites are not coordinated from the beginning to the end with any purposive plan. They exhibit simple behavior depending on the immediate situation of the environment[58]. This instant behavior is called *stigmergy* by Grassé[57]. This behavior is not only observed in termites, but also in ants, bees and many others. In this context, we propose a novel binary approach for AAA algorithm with stigmergic behavior. In this approach, information of the actions performed in the past (C_{01} and C_{10} counters) are used to guide the movements to be performed in the future. Briefly, the information obtained from the processed solution on the problem guides the behavior of algae colonies in the proposed algorithm. UM-1 phase update C_{01} and C_{10} indicators, and in this mechanism these indicators are used to update the solutions in the populations given as follows:

$$p_{01}(t+1) = \frac{C_{01}(t)}{C_{01}(t)+C_{10}(t)} \tag{20}$$

$$p_{10}(t+1) = \frac{C_{10}(t)}{C_{01}(t)+C_{10}(t)} \tag{21}$$

where, $p_{01}(t+1)$ is the probability rate of C_{01} in time *t+1* and $p_{10}(t+1)$ is the probability rate of C_{10} in time *t+1*. These probabilities are used to calculate the candidate solution. In the basic version of AAA algorithm, three dimensions are updated to find new food sources for realizing helical movement. In parallel with this approach, up to three dimensions can be updated for producing candidate solutions. For this purpose, *dimension selection probability (DSP)* method-specific control parameter is added to the method. If random number produced for dimension in range of [0,1] is less than the *DSP* parameter, movement is performed at this dimension.

Let A and B be the index of ones and zeros in candidate solution *V*, respectively. Let a and b be random integers between 1 and sizes of A and B, respectively. Therefore $V_{A(a)}$ indicates random dimension (cell) that have a value of 1 and $V_{B(b)}$ indicates a random dimension that have value of 0. The candidate solution *V* is calculated given as follows:

$$V_{A(a)} = \begin{cases} 0 & , r_1 < p_{10} \\ V_{A(a)} & , \ otherwise \end{cases} \tag{22}$$

$$V_{B(b)} = \begin{cases} 1 & , \ r_1 \geq p_{10} \\ V_{B(b)} & , \ otherwise \end{cases} \tag{23}$$

where, r_1 is a random number produced in range of [0,1]. If r_1 value is less than p_{10} probability value, random decision variable in candidate solution that have a value of 1 ($V_{A(a)}$) is set to 0. If r_1 value is equal or more than the p_{10} probability value, random decision variable in candidate solution that have a value of 0 ($V_{B(b)}$) is set to 1. The working of the update mechanism 2 is briefly given in Fig. 3. After all, for minimization problems, if the objective function value of the candidate solution is less than the selected individual in the population, the candidate solution is copied to this individual and the equation is given as follows:

$$X_i(t+1) = \begin{cases} V(t), & Obj(V(t)) < Obj(X_i(t)) \\ X_i(t), & otherwise \end{cases} \tag{24}$$

4. Experiments

In order to evaluate and compare the performance of the proposed algorithm, two experiments are designed. In first experiment, fifteen UFLPs are considered as test suite and the proposed algorithm are compared with two versions of BAAA, GAs with different crossover operators and BPSO. In second experiments, CEC2015 test suite is used for comparing the performance of the proposed algorithm with SBHS, HS, BLDE, BHTPSO-QI, GBABC, BQIGSA and SabDE. It should be mentioned that the results of compared algorithms on CEC2015 test suite is taken directly from the study of SabDE[49].

4.1. Experimental Material

To better understand the performance of proposed method with the other state-of-the-art algorithms, experiments are performed on two different benchmark sets. Firstly, the performance of algorithms are evaluated on uncapacitated facility location problem (UFLP) taken from OR-Library [59]. The proposed method and other algorithms are re-coded in the MATLAB (Which is trademark of Mathworks Inc.) environment. The second comparison is evaluated on the CEC2015 bound constrained single-objective computationally expensive numerical optimization problems [60]. The proposed method is coded in the MATLAB R2015a environment and the results of the other algorithms are directly taken from [49]. Experiments are performed on a PC with Intel® Core™ i7-3820 3.60GHz CPU and 16GB RAM while running on Windows 10 Pro 64-bit operating system.

4.2. Comparison on UFLPs

In order to perform a fair comparison of the algorithms, the common control parameters of the algorithms are selected equal to each other. The population size is selected as 40 and maximum number of fitness evaluations parameter is set to 80,000 as termination condition. In parallel with maximum number of fitness evaluations, maximum iteration number is set to 2,000 as termination condition for required methods. The crossover and mutation rates are selected as 0.8 and 0.01, respectively for single point crossover, two point crossover and uniform crossover versions of genetic algorithm as used in [23]. For both versions of BAAA [45] algorithm, shear force control parameter is set to 2, energy loss control parameter is set to 0.3 and adaptation rate control parameter is set to 0.5 as used in related work. Changing trend of the curve (*T*) control parameters are selected as 1.5 and 2, respectively for *Tanh(x)* and *Sig(x)* logistic functions. The upper bound of the velocities of the particles (V_{max}) is taken as 6, lower bound of the velocities of the particles (V_{min}) is taken as -6 and positive acceleration constants (c_1, c_2) are taken as 2 for the BPSO algorithm. For proposed method, energy loss control parameter is set to 0.3 and adaptation rate control parameter is set to 0.5 as used in the basic version of AAA[38]. AAA algorithm operates in the continuous solution space and for this reason we don't need shear force control parameter for proposed method. Method-specific control parameters in this approach, named as *Update mechanism selection probability (UMSP)* and *dimension selection probability (DSP)* are selected as 0.5 and 0.66 respectively.

All the algorithms in the comparison are run 30 times with randomly produced seeds and obtained results are reported as mean, standard deviation, hit, mean of running time and gap value. Columns captioned by *Hit*, report the number of successes out of 30 test runs. Columns captioned by *Gap*, report the average gap between the optimal cost value and obtained best cost values which is calculated as follows:

$$Gap_p = \frac{Best_p - Opt_p}{Opt_p} \times 100 \tag{25}$$

where, $Best_p$ is the mean of best solutions obtained from 30 runs for pth problem and Opt_p is the best known solution of the pth problem. The results are given in 4 parts for small sized, medium sized, large sized and ex-large sized problems as in Table 1, Table 2, Table 3 and Table 4, respectively. Sign column in these tables stands for the results of Wilcoxon signed rank test [61] with 0.05 level of p. Likewise, convergence graphs of the proposed method and the other state-of-the-art methods for comparison are given respectively in Fig.4, Fig.5, Fig.6 and Fig.7 for small-sized, medium-sized, large-sized and ex-large sized problems.

On the all problems, except CapB and CapC, the proposed algorithm finds the optimum solution in a reasonable time with respect to compared algorithms. On low dimensional problems, the compared and the proposed algorithms show similar performance in terms of solution quality and robustness. When the dimensionality of the problems is increased, the proposed algorithm is superior to compared algorithms in terms of solution quality. Especially on large-sized problems (CapA, CapB and CapC), these performance differences are clearly seen among the compared algorithms. The similar situations are in the comparisons of the convergence characteristics of the algorithms. Especially, Fig. 7 shows that the proposed algorithm better approaches to the optimum (CapA) or near optimum (CapB and CapC) than the compared algorithms.

The success of an algorithm does not depend on only self-behaviour and algorithmic design but it also depends on dimensionality and characteristics of the optimization problems. Genetic algorithm has a good global search capability and it can produce more quality results on the higher dimensional problems but this algorithm has poor local search or intensification capability. When the chromosomes gather on the similar point on the search space, the algorithm shows stagnation behavior instead of intensification, which is originated from the crossover operator. In order to overcome this problem, the mutation probability should be tuned in accordance with the characteristics and dimensionality of the optimization problem.

This situation is clearly seen from the Table 1, 2, 3 and 4. In accordance with these tables, while the algorithm produces better or comparable solutions on the higher dimensional problems, the algorithm cannot produce good quality or comparable solutions on the lower dimensional problems. Our algorithm is better than the GAs in almost all cases because both exploration and exploitation are provided by binarization process and the origin of the algorithm. In contrast to the behavior GA, the PSO algorithm follows the best solutions (personnel and global) and this is useful behavior on the lower dimensional problems but on the higher dimensional problems, it gets stuck to local minimum because all the solutions including best solutions moves altogether and obtained solutions are similar to each other and exploration capability weakens. Therefore, this algorithm is good at solving lower dimensional problems. When we consider binary variants of AAA, it uses only the information in the population and produces competitive results on all the problems, except CapA, CapB and CapC. The stigmergic behavior provides an advantageous situation for our proposed algorithm because our algorithm uses both information in the population and problem.

In the first part of UFLP experiment, the BPSO, GAs, variants of BAAA and the proposed algorithms are compared to demonstrate the performance of the algorithms. In the second part of this experiment, the results on UFLPs collected from literature are compared with the proposed method. The results of CPSO, DisDE and binDE are directly taken from [62], the results of ABC_{bin} are directly taken from [31], and the results of DisABC and binABC are directly taken from [30]. The implementation details of these algorithms can be found in the given references. Based on these results, the comparison is presented in Table 5. MR and CR stands for mean and corrected rank in this table. The mean rank is calculated as follows: The rank of the algorithm on each problem and then the sum of the ranks is divided by the number of problems. The corrected rank is also used for sorting of the algorithms in accordance with the success.

As seen from the Table 5, the proposed algorithm is first rank and it achieves to the optimal solutions 13 of 15 problems and DisDE is better than the proposed methodology on CapB and CapC problems. DisDE algorithm uses the Jaccard's similarity for generating candidate solutions and its exploration capability is relatively better than the other algorithms. However, this capability causes the intensification of the population found solution and on the lower dimensional problems such as Cap101, Cap103 and Cap133, this exploration behavior in the algorithm prevent the saturation or intensification of the algorithm. Therefore, it can be inferred that the proposed algorithm presents a balanced exploration and exploitation or intensification capability not only lower but also higher dimensional problems. This situation is also seen from the rank measurement and order and valid for comparisons in Table 1, 2, 3 and 4.

4.3. Comparison on Numeric Benchmark Problems

The second experiment has been performed on the CEC2015 bound constrained single-objective computationally expensive numerical optimization problems [60]. On these problems, the proposed algorithm is run 30 times with random seeds and obtained results are compared with the results of state-of-the-art algorithms [49]. For the compared algorithms, experimental result are taken directly from [49] and in order to perform a fair comparison of the algorithms, the common control parameters of the proposed algorithm are selected in accordance with [49]. The population size is selected as 50 and maximum number of fitness evaluations is used for termination condition of the algorithm and it is set to 100,000. The dimension D is set to 10 and the bit size for each dimension is set to 50. Therefore, 500 bits are used for each individual in the population. Since the decision variables of the problems in the benchmark set take on continuous values, each decision variable represented as binary values should be transformed into continuous values before objective function of the benchmark problem is evaluated. This transformation is processed as follows:

$$C_j = L_j + \frac{(H_j - L_j)DecVal_j}{MaxVal} \tag{26}$$

where C_j is the continuous value of the jth dimension, H_j is the upper bound of the jth dimension, L_j is the lower bound of the jth dimension, $DecVal_i$ is the decimal integer value of 50-dimensional binary vector, which represents the jth dimension, and the $MaxVal$ is the maximum decimal integer value of 50 bits. Since *bit size for each dimension* selected as 50, the value of $MaxVal$ is $2^{50} - 1$ accordingly. For the peculiar control parameters of the proposed method, energy loss is set to 0.3 and adaptation rate is set to 0.5 as used in the previous experiment and the basic version of AAA[38]. As the artificial colonies of AAA work on binary-structured solution space, the usage of the shear force parameter of the basic algorithm is not required in the proposed binary version of the algorithm. Method-specific control parameters in this approach, named as *Update mechanism selection probability (UMSP)* and *dimension selection probability (DSP)* are selected as 0.5 and 0.66 respectively.

Under these conditions, obtained results are reported as mean and standard deviation of the runs in Table 6. Additionally the algorithms are ranked in accordance with the mean of runs in order to see the better algorithms at a glance.

As seen from Table 5, the SabDE algorithm produces better quality results than the other algorithms on F4 and F12 functions. On F13 function, the best result is obtained by GBABC algorithm. On the rest of the functions, the proposed algorithm is superior to compared algorithms in terms of solution quality and robustness based on the standard deviations given in the comparison table. In another perspective, when we compared all the algorithms by using ranking, the proposed algorithm shows the best performance among the compared algorithms. This is because the proposed algorithm uses two different update mechanisms based on xor operator and stigmergic behavior. However, when we compare the optimum solution and obtained solutions of these test functions, we see that the binary optimization algorithms need much more efforts in order to achieve optimum or near optimum results.

5. Conclusion and Future Works

Artificial algae algorithm is modified for solving binary optimization problem in the present work. The modification is based on exclusive-or (xor) logic operator and a stigmergic behavior. The helical movement operation in AAA is provided with xor operator due to decision variables of the binary optimization problem and initialization of the algorithm or decision variables are performed by the element of set {0,1}. Stigmergic behavior is included to the algorithm by utilization of two new counters which are affected by working of xor operator and artificial agents in the algorithm. To obtain new candidate solutions or position which correspond the possible solution for the binary optimization problem, not only xor operator but also stigmergic update rule are used in the algorithm. The performance of the proposed approach in AAA algorithm has been investigated on the uncapacitated facility location problem and numeric benchmark functions. The results obtained by the algorithm have been compared with the evolutionary computation and swarm intelligence algorithms and their state-of-art variants. These comparisons show that the proposed algorithm is an effective and efficient algorithm in solving binary optimization problems dealt with the study in terms of solution quality, convergence characteristics and robustness based on the standard deviation. In near future, we will apply this algorithm to solve different binary optimization problems, especially knapsack problems which widely studied and we will also use the stigmergic behavior proposed in the present study in the other swarm intelligence algorithms to solve the binary optimization problems.

Acknowledgements

The authors wish to thank Scientific and Technological Research Council of Turkey and Selcuk University Scientific Projects Coordinatorship for their institutional supports.

References

1. Ingolfsson, A., S. Budge, and E. Erkut, *Optimal ambulance location with random delays and travel times.* Health Care Management Science, 2008. **11**(3): p. 262-274.
2. Andrade, L.A.C.G. and C.B. Cunha, *An ABC heuristic for optimizing moveable ambulance station location and vehicle repositioning for the city of Sao Paulo.* International Transactions in Operational Research, 2015. **22**(3): p. 473-501.
3. Liu, N., B. Huang, and M. Chandramouli, *Optimal siting of fire stations using GIS and ANT algorithm.* Journal of Computing in Civil Engineering, 2006. **20**(5): p. 361-369.
4. Yang, L.L., B.F. Jones, and S.H. Yang, *A fuzzy multi-objective programming for optimization of fire station locations through genetic algorithms.* European Journal of Operational Research, 2007. **181**(2): p. 903-915.
5. Tan, F., et al., *A genetic algorithm-based method for feature subset selection.* Soft Computing, 2008. **12**(2): p. 111-120.
6. Atkinson, J. and D. Campos, *Improving BCI-based emotion recognition by combining EEG feature selection and kernel classifiers.* Expert Systems with Applications, 2016. **47**: p. 35-41.
7. Gunasundari, S., S. Janakiraman, and S. Meenambal, *Velocity Bounded Boolean Particle Swarm Optimization for improved feature selection in liver and kidney disease diagnosis.* Expert Systems with Applications, 2016. **56**: p. 28-47.
8. Shang, L., Z. Zhou, and X. Liu, *Particle swarm optimization-based feature selection in sentiment classification.* Soft Computing, 2016. **20**(10): p. 3821-3834.
9. Shunmugapriya, P. and S. Kanmani, *A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC Hybrid).* Swarm and Evolutionary Computation, 2017.
10. Pavez-Lazo, B. and J. Soto-Cartes, *A deterministic annular crossover genetic algorithm optimisation for the unit commitment problem.* Expert Systems with Applications, 2011. **38**(6): p. 6523-6529.
11. Kamboj, V.K., *A novel hybrid PSO-GWO approach for unit commitment problem.* Neural Computing & Applications, 2016. **27**(6): p. 1643-1655.
12. Khazaei, P., et al., *Applying the Modified TLBO Algorithm to Solve the Unit Commitment Problem.* 2016 World Automation Congress (Wac), 2016.
13. Li, S., et al., *Discrete Chaotic Gravitational Search Algorithm for Unit Commitment Problem.* Intelligent Computing Theories and Application, Icic 2016, Pt Ii, 2016. **9772**: p. 757-769.

14. Saravanan, B., C. Kumar, and D.P. Kothari, *A solution to unit commitment problem using fire works algorithm.* International Journal of Electrical Power & Energy Systems, 2016. **77**: p. 221-227.
15. Haddar, B., et al., *A new Hybrid Heuristic for the 0-1 Knapsack Sharing Problem.* Proceedings of 2013 International Conference on Industrial Engineering and Systems Management (Ieee-Iesm 2013), 2013: p. 12-18.
16. Haddar, B., et al., *A hybrid heuristic for the 0-1 Knapsack Sharing Problem.* Expert Systems with Applications, 2015. **42**(10): p. 4653-4666.
17. Azad, M.A.K., A.M.A. Rocha, and E.M. Fernandes, *Improved binary artificial fish swarm algorithm for the 0–1 multidimensional knapsack problems.* Swarm and Evolutionary Computation, 2014. **14**: p. 66-75.
18. Pavithr, R., *Quantum Inspired Social Evolution (QSE) algorithm for 0-1 knapsack problem.* Swarm and Evolutionary Computation, 2016. **29**: p. 33-46.
19. Beasley, J.E., *A Lagrangian Heuristic for Set-Covering Problems.* Naval Research Logistics, 1990. **37**(1): p. 151-164.
20. Babaoglu, I., *Utilization of Bat Algorithm for Solving Uncapacitated Facility Location Problem.* Intelligent and Evolutionary Systems, Ies 2015, 2016. **5**: p. 199-208.
21. Arulselvan, A., O. Maurer, and M. Skutella, *An incremental algorithm for the uncapacitated facility location problem.* Networks, 2015. **65**(4): p. 306-311.
22. Watanabe, Y., M. Takaya, and A. Yamamura, *Fitness Function in ABC Algorithm for Uncapacitated Facility Location Problem.* Information and Communication Technology, 2015. **9357**: p. 129-138.
23. Holland, J.H., *Genetic Algorithms.* Scientific American, 1992. **267**(1): p. 66-72.
24. Davis, L., *Genetic algorithms and simulated annealing.* 1987.
25. Holland, J.H., *Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence.* Ann Arbor, MI: University of Michigan Press, 1975.
26. Spears, W.M. and K.A. De Jong, *An analysis of multi-point crossover*. 1990, DTIC Document.
27. Syswerda, G., *Uniform Crossover in Genetic Algorithms.* Proceedings of the Third International Conference on Genetic Algorithms, 1989: p. 2-9.
28. Semenkin, E. and M. Semenkina, *Self-configuring Genetic Algorithm with Modified Uniform Crossover Operator.* Advances in Swarm Intelligence, Icsi 2012, Pt I, 2012. **7331**: p. 414-421.
29. Karaboga , D., *An Idea Based on Honey Bee Swarm for Numerical Optimization*. 2005, Erciyes University, Engineering Faculty, Computer Engineering Department ,Kayseri. p. 1-10.
30. Kiran, M.S. and M. Gunduz, *XOR-based artificial bee colony algorithm for binary optimization.* Turkish Journal of Electrical Engineering and Computer Sciences, 2013. **21**: p. 2307-2328.
31. Kiran, M.S., *The continuous artificial bee colony algorithm for binary optimization.* Applied Soft Computing, 2015. **33**: p. 15-23.
32. Kennedy, J. and R. Eberhart, *Particle swarm optimization.* 1995 Ieee International Conference on Neural Networks Proceedings, Vols 1-6, 1995: p. 1942-1948.
33. Kennedy, J. and R.C. Eberhart, *A discrete binary version of the particle swarm algorithm.* Smc '97 Conference Proceedings - 1997 Ieee International Conference on Systems, Man, and Cybernetics, Vols 1-5, 1997: p. 4104-4108.
34. Afshinmanesh, F., A. Marandi, and A. Rahimi-Kian, *A novel binary particle swarm optimization method using artificial immune system.* Eurocon 2005: The International Conference on Computer as a Tool, Vol 1 and 2 , Proceedings, 2005: p. 217-220.

35. Mirjalili, S. and A. Lewis, *S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization.* Swarm and Evolutionary Computation, 2013. **9**: p. 1-14.
36. Saremi, S., S. Mirjalili, and A. Lewis, *How important is a transfer function in discrete heuristic algorithms.* Neural Computing & Applications, 2015. **26**(3): p. 625-640.
37. Mirjalili, S., G.G. Wang, and L.D. Coelho, *Binary optimization using hybrid particle swarm optimization and gravitational search algorithm.* Neural Computing & Applications, 2014. **25**(6): p. 1423-1435.
38. Uymaz, S.A., G. Tezel, and E. Yel, *Artificial algae algorithm (AAA) for nonlinear global optimization.* Applied Soft Computing, 2015. **31**: p. 153-171.
39. Uymaz, S.A., G. Tezel, and E. Yel, *Artificial algae algorithm with multi-light source for numerical optimization and applications.* Biosystems, 2015. **138**: p. 25-38.
40. Liang, J., et al., *Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization.* Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 2013. **201212**.
41. Suganthan, P.N., et al., *Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization.* KanGAL report, 2005. **2005005**: p. 2005.
42. Das, S. and P.N. Suganthan, *Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems.* Jadavpur University, Nanyang Technological University, Kolkata, 2010.
43. Clerc, M., *A method to improve standard PSO.* 2009.
44. Onwubolu, G.C. and B. Babu, *New optimization techniques in engineering*. Vol. 141. 2013: Springer.
45. Zhang, X.D., et al., *Binary artificial algae algorithm for multidimensional knapsack problems.* Applied Soft Computing, 2016. **43**: p. 583-595.
46. Storn, R. and K. Price, *Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces.* Journal of global optimization, 1997. **11**(4): p. 341-359.
47. Pampara, G., A.P. Engelbrecht, and N. Franken, *Binary Differential Evolution.* 2006 Ieee Congress on Evolutionary Computation, Vols 1-6, 2006: p. 1858-+.
48. Engelbrecht, A.P. and G. Pampara, *Binary differential evolution strategies.* 2007 Ieee Congress on Evolutionary Computation, Vols 1-10, Proceedings, 2007: p. 1942-1947.
49. Banitalebi, A., M.I. Abd Aziz, and Z.A. Aziz, *A self-adaptive binary differential evolution algorithm for large scale binary optimization problems.* Information Sciences, 2016. **367**: p. 487-511.
50. Kong, X.Y., et al., *A simplified binary harmony search algorithm for large scale 0-1 knapsack problems.* Expert Systems with Applications, 2015. **42**(12): p. 5337-5355.
51. Geem, Z.W., J.H. Kim, and G.V. Loganathan, *A new heuristic optimization algorithm: Harmony search.* Simulation, 2001. **76**(2): p. 60-68.
52. Chen, Y., W.C. Xie, and X.F. Zou, *A binary differential evolution algorithm learning from explored solutions.* Neurocomputing, 2015. **149**: p. 1038-1047.
53. Beheshti, Z., S.M. Shamsuddin, and S. Hasan, *Memetic binary particle swarm optimization for discrete optimization problems.* Information Sciences, 2015. **299**: p. 58-84.
54. Ozturk, C., E. Hancer, and D. Karaboga, *A novel binary artificial bee colony algorithm based on genetic operators.* Information Sciences, 2015. **297**: p. 154-170.

55. Nezamabadi-pour, H., *A quantum-inspired gravitational search algorithm for binary encoded optimization problems.* Engineering Applications of Artificial Intelligence, 2015. **40**: p. 62-75.
56. Rashedi, E., H. Nezamabadi-Pour, and S. Saryazdi, *GSA: A Gravitational Search Algorithm.* Information Sciences, 2009. **179**(13): p. 2232-2248.
57. Grassé, P.-P., *La reconstruction du nid et les coordinations interindividuelles chezBellicositermes natalensis etCubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs.* Insectes sociaux, 1959. **6**(1): p. 41-80.
58. Crina, G. and A. Ajith, *Stigmergic optimization: Inspiration, technologies and perspectives*, in *Stigmergic optimization*. 2006, Springer. p. 1-24.
59. Beasley, J.E., *Or-Library - Distributing Test Problems by Electronic Mail.* Journal of the Operational Research Society, 1990. **41**(11): p. 1069-1072.
60. Chen, Q., et al., *Problem definition and evaluation criteria for CEC 2015 special session and competition on bound constrained single-objective computationally expensive numerical optimization.* Computational Intelligence Laboratory, Zhengzhou University, China and Nanyang Technological University, Singapore, Tech. Rep, 2014.
61. Wilcoxon, F., *Individual Comparisons by Ranking Methods.* Biometrics Bulletin, 1945. **1**(6): p. 80-83.
62. Kashan, M.H., A.H. Kashan, and N. Nahavandi, *A novel differential evolution algorithm for binary optimization.* Computational Optimization and Applications, 2013. **55**(2): p. 481-513.

Figures

Initialization phase

Determine shear force (Δ), energy loss (e) and adaptation rate (A_p) parameters.
Initialize algal colonies using Eq.1
Evaluate fitness of algal colonies using Eq.2
Evaluate size of algal colonies using Eq.3, Eq.4 and Eq.5

Helical movement phase

For each alga colony

- Select a neighbor via tournament selection
- Select three algal cells(dimensions) randomly
- Modify the colony using Eq.6, Eq.7, Eq.8, Eq.9 and Eq.10 until its energy exhausted.

Evaluation phase

Copy a randomly selected single cell of biggest colony to smallest colony.

Adaptation phase

If the random number is less than the adaptation rate parameter, find the most starveling colony and apply Eq.11 with biggest colony.
If maximum fitness evaluation count is reached; report the best colony.

Fig. 1. The pseudo-code of the AAA algorithm

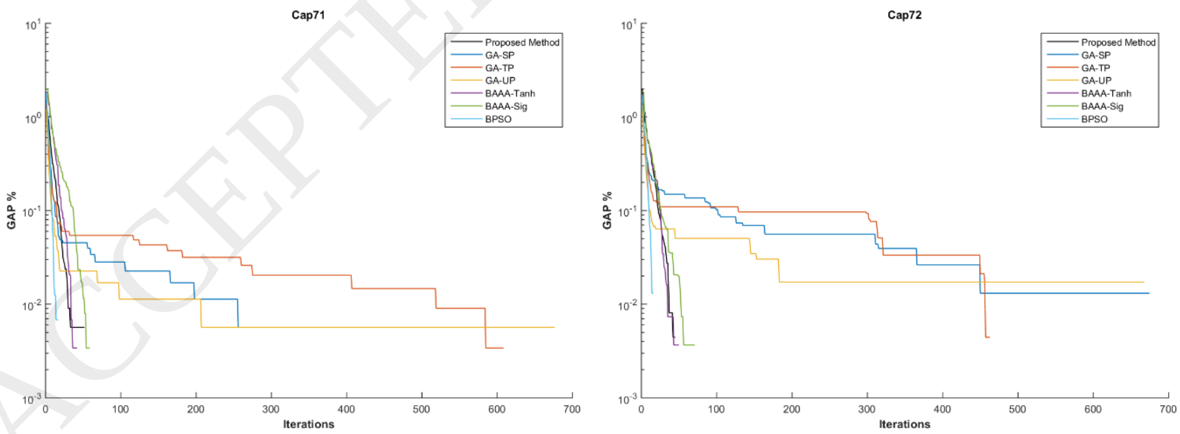
1) Let *n* is index of randomly selected neighbor for *X(i)* using Obj via tournament selection
2) Let *P* be three random dimension indexes between [1,Dim] and different from each other
3) Let *V* is candidate solution
4) φ is the logic NOT operator set as 0.5
5) V=X(i)

```
    For z=1 to 3
      If rand< φ then
        V( P(z) ) = XOR ( X(i, P(z) ) , XOR ( X(i, P(z) ), X(n, P(z) )  ) )
      else
        V( P(z) ) = XOR ( X(i, P(z) ) , □XOR ( X(i, P(z) ), X(n, P(z) )  ) )
      end if
    end for
    If Obj(V)<Obj( X(i) ) then
      For z=1 to 3
        If X(i, P(z) )==0 and V( P(z) )==1 then C01=C01+1 end if
        If X(i, P(z) )==1 and V( P(z) )==0 then C10=C10+1 end if
      end for
      X(i)=V
    end if
```

Fig. 2. The pseudo-code of the Update Mechanism 1.

```
Let V is candidate solution
For d=1 to 3
  If rand < DSP then
    If rand < p10 then
      A=find(V==1)
      a=rand(size of A)
      V( A(a) )=0
    else if
      B=find(V==0)
      b=rand(size of B)
      V( B(b) )=1
    end if
  end if
end for
If Obj(V)<Obj( X(i) ) then
  X(i)=V
end if
```

Fig. 3. The pseudo-code of the Update Mechanism 2



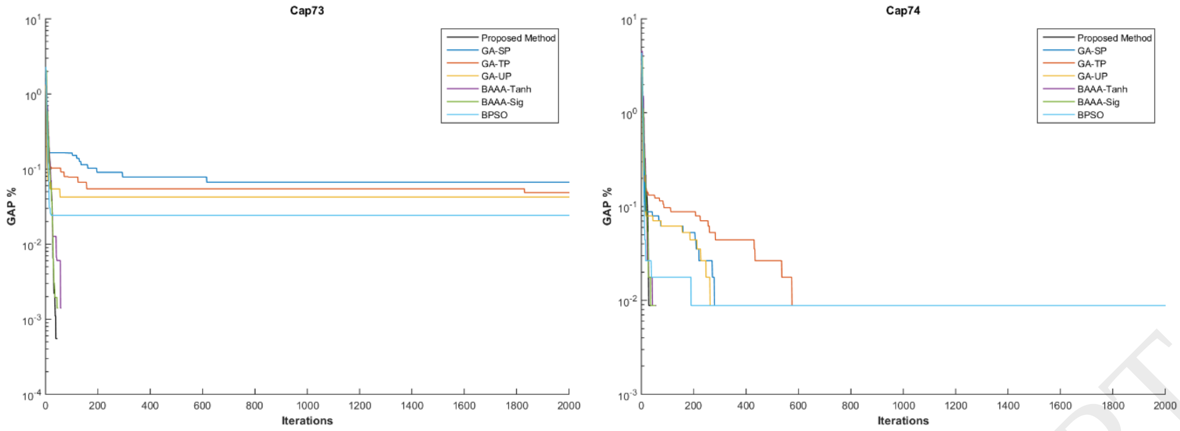


Fig. 4. Convergence graphs of the algorithms on the Cap71, Cap72, Cap73 and Cap74 problems.

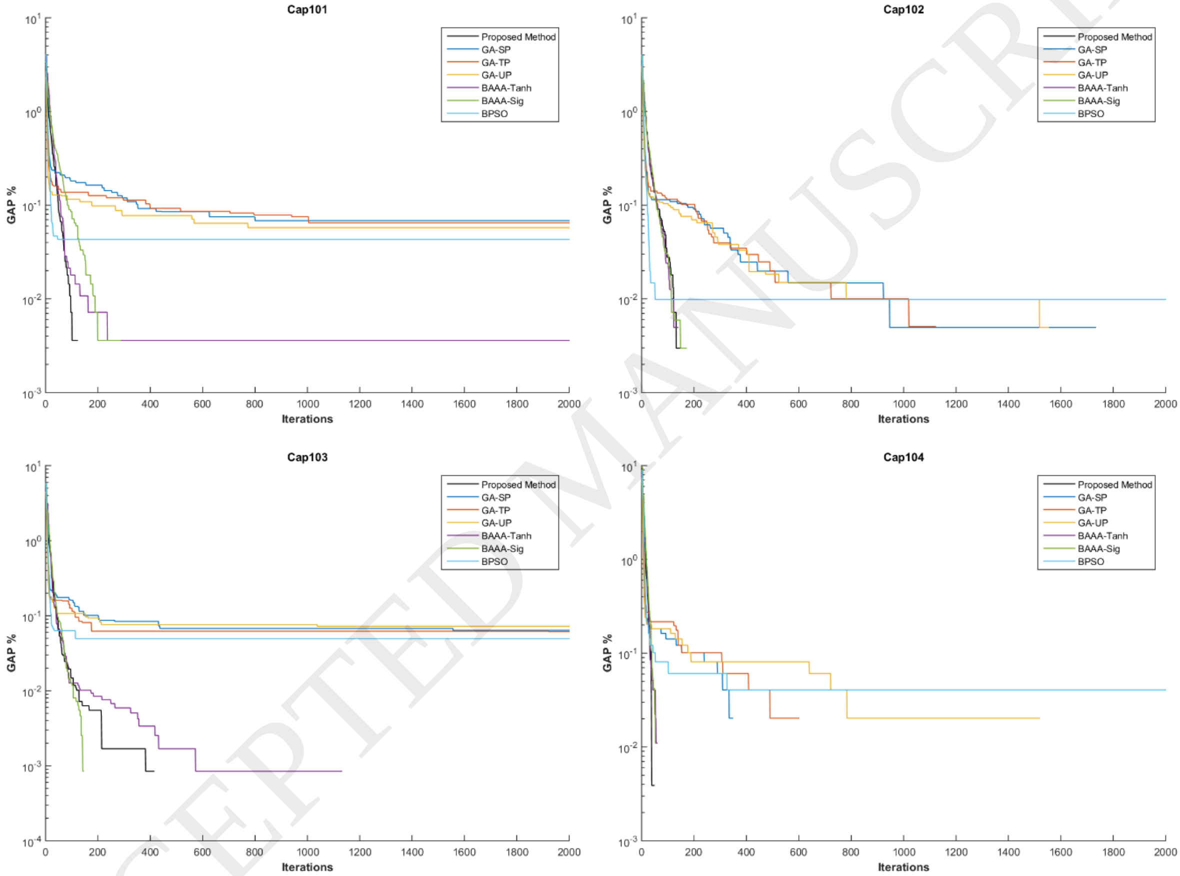


Fig. 5. Convergence graphs of the algorithms on the Cap101, Cap102, Cap103 and Cap104 problems.

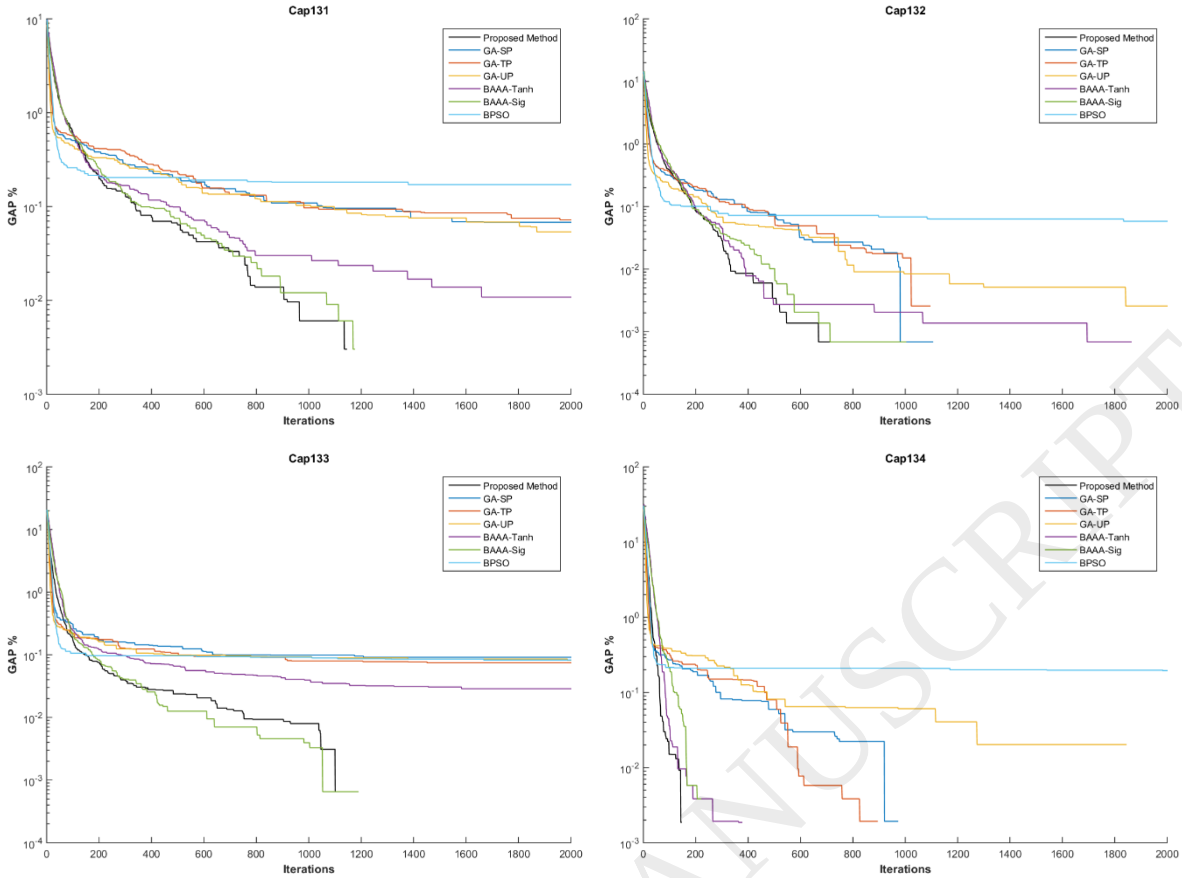


Fig. 6. Convergence graphs of the algorithms on the Cap131, Cap132, Cap133 and Cap134 problems.

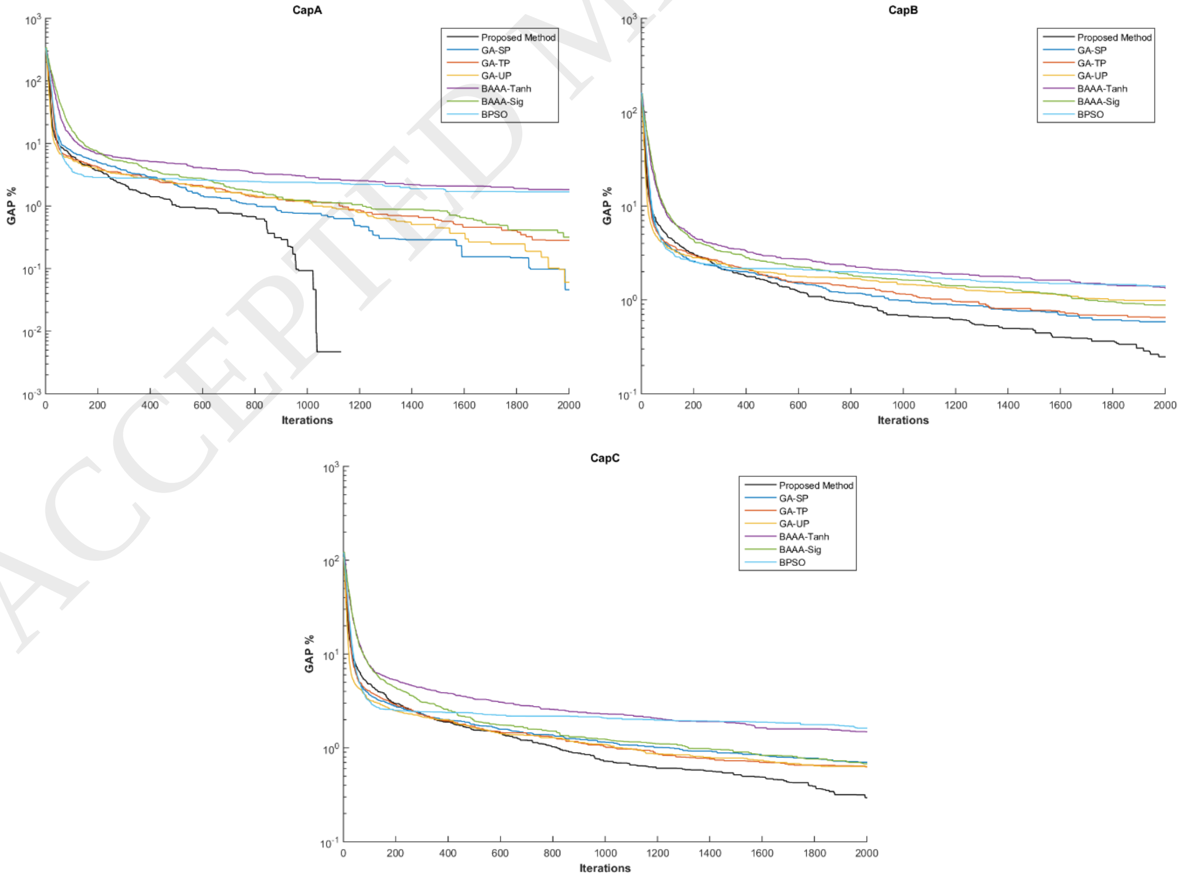


Fig. 7. Convergence graphs of the algorithms on the CapA, CapB and CapC problems.

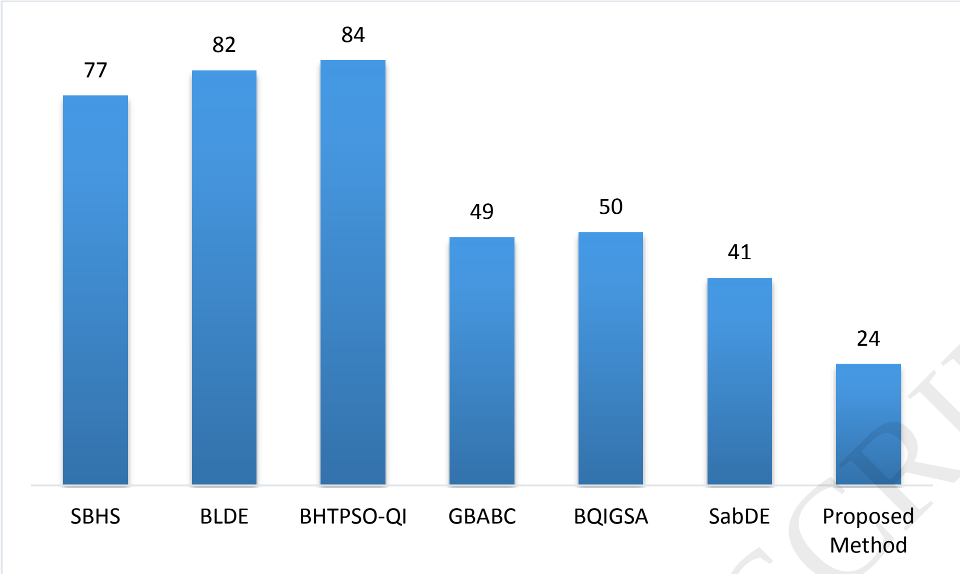


Fig. 8. Comparison of the algorithms according to the ranking mechanism.

Table 1. Comparison of proposed method with the binary structured optimization algorithms on the Cap71, Cap72, Cap73 and Cap74 problems

Methods	Metric	Cap71	Cap72	Cap73	Cap74
GA-SP	Mean	**932615.750**	**977799.400**	1011314.476	**1034976.975**
	Gap	**0.00000**	**0.00000**	0.06659	**0.00000**
	Hit	**30**	**30**	19	**30**
	Std.Dev.	**0.000**	**0.000**	899.650	**0.000**
	Sign.	-	-	+	-
	Avg.Time	**26.957**	**27.893**	27.994	**27.998**
GA-TP	Mean	**932615.750**	**977799.400**	1011130.923	**1034976.975**
	Gap	**0.00000**	**0.00000**	0.04843	**0.00000**
	Hit	**30**	**30**	22	**30**
	Std.Dev.	**0.000**	**0.000**	825.576	**0.000**
	Sign.	-	-	+	-
	Avg.Time	**27.568**	**28.056**	28.050	**28.143**
GA-UP	Mean	**932615.750**	**977799.400**	1011069.739	**1034976.975**
	Gap	**0.00000**	**0.00000**	0.04238	**0.00000**
	Hit	**30**	**30**	23	**30**
	Std.Dev.	**0.000**	**0.000**	789.612	**0.000**
	Sign.	-	-	+	-
	Avg.Time	**27.744**	**28.103**	28.101	**28.095**
BAAA-Tanh	Mean	**932615.750**	**977799.400**	**1010641.450**	**1034976.975**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**

	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Sign.	**-**	**-**	**-**	**-**
	Avg.Time	**25.748**	**26.279**	**28.221**	**27.584**
BAAA-Sig	Mean	**932615.750**	**977799.400**	**1010641.450**	**1034976.975**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Sign.	**-**	**-**	**-**	**-**
	Avg.Time	**25.868**	**26.927**	**28.219**	**27.461**
BPSO	Mean	**932615.750**	**977799.400**	1010886.187	1035068.312
	Gap	**0.00000**	**0.00000**	0.02422	0.00882
	Hit	**30**	**30**	26	29
	Std.Dev.	**0.000**	**0.000**	634.625	500.272
	Sign.	**-**	**-**	**-**	**-**
	Avg.Time	**34.935**	**34.765**	34.950	34.822
Proposed Method	Mean	**932615.750**	**977799.400**	**1010641.450**	**1034976.975**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Avg.Time	**27.901**	**28.078**	**27.528**	**27.239**

Table 2. Comparison of proposed method with the binary structured optimization algorithms on the Cap101, Cap102, Cap103 and Cap104 problems.

Methods	Metric	Cap101	Cap102	Cap103	Cap104
GA-SP	Mean	797193.286	**854704.200**	894351.782	**928941.750**
	Gap	0.06839	**0.00000**	0.06374	**0.00000**
	Hit	11	**30**	6	**30**
	Std.Dev.	421.655	**0.000**	505.036	**0.000**
	Sign.	+	**-**	+	**-**
	Avg.Time	29.372	**28.730**	28.689	**32.706**
GA-TP	Mean	797164.610	**854704.200**	894329.179	**928941.750**
	Gap	0.06479	**0.00000**	0.06121	**0.00000**
	Hit	12	**30**	10	**30**
	Std.Dev.	428.658	**0.000**	540.160	**0.000**
	Sign.	+	**-**	+	**-**
	Avg.Time	29.206	**28.931**	28.907	**32.992**
GA-UP	Mean	797107.258	**854704.200**	894427.382	**928941.750**
	Gap	0.05759	**0.00000**	0.07220	**0.00000**
	Hit	14	**30**	9	**30**
	Std.Dev.	436.524	**0.000**	522.784	**0.000**
	Sign.	+	**-**	+	**-**
	Avg.Time	29.169	**28.870**	28.915	**33.046**
BAAA-Tanh	Mean	796677.114	**854704.200**	**893782.113**	**928941.750**

	Gap	0.00360	**0.00000**	**0.00000**	**0.00000**
	Hit	29	**30**	**30**	**30**
	Std.Dev.	157.066	**0.000**	**0.000**	**0.000**
	Sign.	-	-	-	-
	Avg.Time	27.473	**26.334**	**25.851**	**25.435**
BAAA-Sig	Mean	**796648.438**	**854704.200**	**893782.113**	**928941.750**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Sign.	-	-	-	-
	Avg.Time	**26.836**	**26.215**	**25.926**	**26.963**
BPSO	Mean	796992.553	854788.703	894223.572	929318.098
	Gap	0.04320	0.00989	0.04939	0.04051
	Hit	18	28	14	28
	Std.Dev.	428.658	321.588	521.237	1432.239
	Sign.	+	-	+	-
	Avg.Time	41.814	41.554	41.375	41.177
Proposed Method	Mean	**796648.438**	**854704.200**	**893782.113**	**928941.750**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Avg.Time	**28.177**	**27.923**	**27.838**	**27.592**

Table 3. Comparison of proposed method with the binary structured optimization algorithms on the Cap131, Cap132, Cap133 and Cap134 problems.

Methods	Metric	Cap131	Cap132	Cap133	Cap134
GA-SP	Mean	793980.104	**851495.325**	893891.911	**928941.750**
	Gap	0.06813	**0.00000**	0.09128	**0.00000**
	Hit	16	**30**	10	**30**
	Std.Dev.	720.877	**0.000**	685.076	**0.000**
	Sign.	+	-	+	-
	Avg.Time	34.017	**35.107**	38.143	**36.748**
GA-TP	Mean	794012.905	**851495.325**	893740.954	**928941.750**
	Gap	0.07226	**0.00000**	0.07438	**0.00000**
	Hit	14	**30**	12	**30**
	Std.Dev.	690.560	**0.000**	655.920	**0.000**
	Sign.	+	-	+	-
	Avg.Time	33.885	**35.930**	37.906	**36.609**
GA-UP	Mean	793865.023	851517.200	893808.891	**928941.750**
	Gap	0.05362	0.00257	0.08198	**0.00000**
	Hit	15	29	9	**30**
	Std.Dev.	433.467	119.817	628.654	**0.000**
	Sign.	+	-	+	-
	Avg.Time	33.563	36.329	38.038	**36.588**

BAAA-Tanh	Mean	793525.591	**851495.325**	893333.515	**928941.750**
	Gap	0.01084	**0.00000**	0.02875	**0.00000**
	Hit	27	**30**	16	**30**
	Std.Dev.	262.498	**0.000**	324.451	**0.000**
	Sign.	-	-	+	-
	Avg.Time	45.498	**57.039**	56.239	**56.622**
BAAA-Sig	Mean	**793439.563**	**851495.325**	**893076.713**	**928941.750**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Sign.	-	-	-	-
	Avg.Time	**63.656**	**67.132**	**66.943**	**67.612**
BPSO	Mean	794797.761	851991.551	893816.653	930756.565
	Gap	0.17118	0.05828	0.08285	0.19536
	Hit	10	21	10	18
	Std.Dev.	1505.749	1055.238	690.192	2594.211
	Sign.	+	+	+	+
	Avg.Time	60.083	59.759	59.733	59.516
Proposed Method	Mean	**793439.563**	**851495.325**	**893076.713**	**928941.750**
	Gap	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	Hit	**30**	**30**	**30**	**30**
	Std.Dev.	**0.000**	**0.000**	**0.000**	**0.000**
	Avg.Time	**28.080**	**28.539**	**28.336**	**28.011**

Table 4. Comparison of proposed method with the binary structured optimization algorithms on the CapA, CapB and CapC problems.

Methods	Metric	CapA	CapB	CapC
GA-SP	Mean	17164354.456	13054858.045	11586692.969
	Gap	0.04605	0.58391	0.70486
	Hit	24	9	2
	Std.Dev.	22451.206	66658.649	51848.248
	Sign.	+	+	+
	Avg.Time	741.535	743.370	744.455
GA-TP	Mean	17205089.145	13063527.186	11577797.524
	Gap	0.28348	0.65071	0.62755
	Hit	24	11	0
	Std.Dev.	139690.216	89122.485	46346.052
	Sign.	+	+	+
	Avg.Time	735.939	738.387	741.823
GA-UP	Mean	17166811.915	13107633.077	11578600.532
	Gap	0.06037	0.99053	0.63453
	Hit	24	3	0
	Std.Dev.	35181.974	79714.021	57031.219
	Sign.	+	+	+
	Avg.Time	743.848	748.144	745.128
BAAA-Tanh	Mean	17471223.794	13153617.764	11676427.752
	Gap	1.83470	1.34483	1.48479
	Hit	3	0	0
	Std.Dev.	225123.921	73978.543	101438.607
	Sign.	+	+	+
	Avg.Time	864.755	473.660	479.762
BAAA-Sig	Mean	17210900.533	13093705.559	11583462.068
	Gap	0.31735	0.88322	0.67678
	Hit	16	1	1
	Std.Dev.	90743.456	62168.803	45788.678
	Sign.	+	+	+
	Avg.Time	880.452	475.592	470.911
BPSO	Mean	17446511,870	13161205,473	11692212,797
	Gap	1,69066	1,40329	1,62198
	Hit	8	5	1
	Std.Dev.	319855,431	135326,728	115156,444
	Sign.	+	+	+
	Avg.Time	534,103	539,601	541,035
Proposed Method	Mean	**17156454.478**	**13011234.616**	**11539496.443**
	Gap	**0.00000**	**0.24781**	**0.29466**
	Hit	**30**	**15**	**1**
	Std.Dev.	**0.000**	**39224.744**	**29766.311**
	Avg.Time	**461.906**	**470.094**	**455.470**

Table 5. The comparison of proposed algorithm with the state-of-art binary versions of ABC, DE and PSO by using gap and rank

Problems	CPSO		ABC_{bin}		DisDE		binDE		DisABC		binABC		Proposed Method	
	Gap	Rank	Gap	Rank	Gap	Rank	Gap	Rank	Gap	Rank	Gap	Rank	Gap	Rank
Cap71	5.0E-02	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap72	7.0E-02	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap73	6.0E-02	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap74	7.0E-02	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap101	1.4E-01	3	0.0E+00	1	7.2E-03	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap102	1.5E-01	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap103	1.6E-01	4	5.1E-03	3	8.4E-04	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap104	1.8E-01	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
Cap131	7.5E-01	5	2.0E-01	3	0.0E+00	1	3.6E-03	2	6.2E-01	4	0.0E+00	1	0.0E+00	1
Cap132	7.8E-01	5	2.0E-02	3	0.0E+00	1	5.0E-03	2	9.5E-02	4	0.0E+00	1	0.0E+00	1
Cap133	7.3E-01	7	7.5E-02	5	1.5E-02	3	1.4E-02	2	3.1E-02	4	1.2E-01	6	0.0E+00	1
Cap134	8.9E-01	2	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1	0.0E+00	1
CapA	2.2E+01	7	3.2E+00	6	3.7E-02	2	1.3E+00	4	1.5E-01	3	3.0E+00	5	0.0E+00	1
CapB	1.1E+01	7	2.8E+00	5	6.7E-02	1	1.5E+00	3	3.3E+00	6	2.5E+00	4	2.5E-01	2
CapC	9.7E+00	7	2.0E+00	4	5.8E-02	1	1.6E+00	3	4.7E+00	6	2.6E+00	5	2.9E-01	2
MR	3.93		2.47		1.33		1.67		2.40		2.07		1.13	
CR	7		6		2		3		5		4		1	

Table 6. The comparison of proposed algorithm with the state-of-art methods on CEC2015 test suite

	Metric	SBHS	BLDE	BHTPSO-QI	GBABC	BQIGSA	SabDE	Proposed Method
F1	Mean	3.700E+08	1.087E+11	7.510E+08	2.729E+07	8.419E+07	3.093E+08	**2.140E+07**
	Std	2.271E+08	2.351E+11	2.203E+09	2.267E+07	7.354E+07	1.168E+08	**2.443E+07**
	Rank	5	7	6	2	3	4	**1**
F2	Mean	4.493E+09	4.747E+09	5.132E+09	2.864E+09	7.834E+09	2.541E+09	**1.458E+04**
	Std	1.790E+06	3.209E+08	4.685E+08	2.374E+09	6.527E+09	5.008E+09	**5.211E+03**
	Rank	4	5	6	3	7	2	**1**
F3	Mean	3.203E+02	3.201E+02	3.203E+02	3.202E+02	3.202E+02	3.200E+02	**3.061E+02**
	Std	7.013E-02	1.093E-02	7.426E-02	2.641E+02	2.641E+02	2.044E-02	**9.017E-01**
	Rank	5	3	5	4	4	2	**1**
F4	Mean	4.478E+02	4.378E+02	4.478E+02	4.358E+02	4.389E+02	**4.116E+02**	5.014E+02
	Std	5.171E+00	7.754E+00	1.122E+01	3.599E+02	3.621E+02	**7.606E+00**	6.983E+01
	Rank	5	3	5	2	4	**1**	6
F5	Mean	2.592E+03	1.934E+03	1.804E+03	1.108E+03	1.542E+03	9.330E+02	**5.004E+02**
	Std	1.959E+02	3.088E+02	3.926E+02	9.736E+02	1.275E+03	9.464E+01	**1.445E-01**
	Rank	7	6	5	3	4	2	**1**
F6	Mean	5.312E+04	2.484E+06	6.799E+06	7.442E+06	5.582E+05	4.625E+04	**6.003E+02**
	Std	1.512E+04	7.369E+06	1.186E+07	1.321E+07	6.055E+05	4.076E+04	**9.229E-02**
	Rank	3	5	6	7	4	2	**1**
F7	Mean	8.368E+02	8.472E+02	8.510E+02	7.589E+02	7.392E+02	7.752E+02	**7.002E+02**
	Std	7.981E+00	1.289E+01	1.694E+01	4.668E+02	4.324E+02	4.155E+03	**1.545E-01**
	Rank	5	6	7	3	2	4	**1**
F8	Mean	3.344E+08	7.104E+09	5.407E+07	3.949E+07	2.948E+06	2.395E+07	**8.052E+02**
	Std	1.810E+09	5.651E+09	8.312E+07	2.442E+08	2.469E+06	5.432E+07	**1.475E+00**
	Rank	6	7	5	4	2	3	**1**
F9	Mean	1.189E+03	1.190E+03	1.209E+03	1.017E+03	1.048E+03	1.177E+03	**9.033E+02**
	Std	5.210E+00	1.736E+01	2.050E+01	8.397E+02	8.643E+02	4.102E+01	**2.616E-01**
	Rank	5	6	7	2	3	4	**1**
F10	Mean	9.038E+08	1.326E+10	1.320E+09	9.909E+05	4.426E+04	2.416E+07	**1.293E+04**
	Std	6.596E+08	1.724E+10	2.073E+09	3.659E+06	4.477E+04	8.862E+07	**1.068E+04**
	Rank	5	7	6	3	2	4	**1**
F11	Mean	1.737E+03	1.599E+03	1.632E+03	1.159E+03	1.172E+03	1.114E+03	**1.105E+03**
	Std	1.661E+01	3.313E+01	3.371E+01	9.557E+02	9.669E+02	1.131E+01	**8.947E-01**
	Rank	7	5	6	3	4	2	**1**
F12	Mean	1.445E+03	1.440E+03	1.437E+03	1.264E+03	1.255E+03	**1.224E+03**	1.249E+03
	Std	5.683E-01	3.247E+00	7.552E+00	1.044E+03	1.035E+03	**1.302E+00**	2.176E+01
	Rank	7	6	5	4	3	**1**	2
F13	Mean	1.464E+10	2.882E+10	5.365E+08	**1.446E+03**	1.452E+03	2.815E+09	1.627E+03
	Std	9.749E+09	2.073E+10	2.294E+09	**1.193E+03**	1.197E+03	4.158E+09	5.024E+00
	Rank	6	7	4	**1**	2	5	3
F14	Mean	2.736E+03	4.267E+03	5.005E+03	2.162E+03	3.356E+03	1.727E+03	**1.600E+03**
	Std	2.220E+02	2.569E+03	2.703E+03	2.091E+03	2.869E+03	4.411E+02	**3.715E+00**
	Rank	4	6	7	3	5	2	**1**
F15	Mean	1.700E+03	1.700E+03	1.734E+03	2.012E+03	**1.530E+03**	1.700E+03	1.614E+03
	Std	1.552E-04	2.622E-05	1.851E+02	1.659E+03	**1.262E+03**	2.177E-05	1.257E+02
	Rank	3	3	4	5	**1**	3	2