Heuristic-Based Firefly Algorithm for Bound Constrained Nonlinear Binary Optimization

M. Fernanda P. Costa[1], Ana Maria A.C. Rocha[2], Rogério B. Francisco[1] and Edite M.G.P. Fernandes[2]

[1]Department of Mathematics and Applications,

Centre of Mathematics, University of Minho, 4710-057 Braga, Portugal

[2]Algoritmi Research Centre, University of Minho, 4710-057 Braga, Portugal

September 22, 2014

Firefly algorithm (FA) is a metaheuristic for global optimization. In this paper, we address the practical testing of a heuristic-based FA (HBFA) for computing optima of discrete nonlinear optimization problems, where the discrete variables are of binary type. An important issue in FA is the formulation of attractiveness of each firefly which in turn affects its movement in the search space. Dynamic updating schemes are proposed for two parameters, one from the attractiveness term and the other from the randomization term. Three simple heuristics capable of transforming real continuous variables into binary ones are analyzed. A new sigmoid '`erf`' function is proposed. In the context of FA, three different implementations to incorporate the heuristics for binary variables into the algorithm are proposed. Based on a set of benchmark problems, a comparison is carried out with other binary dealing metaheuristics. The results demonstrate that the proposed HBFA is efficient and outperforms binary versions of differential evolution (DE) and particle swarm optimization (PSO). The HBFA also compares very favorably with angle modulated version of DE and PSO. It is shown that the variant of HBFA based on the sigmoid '`erf`' function with 'movements in continuous space' is the best, both in terms of computational requirements and accuracy.

1 Introduction

This paper aims to analyze the merit, in terms of performance, of a heuristic-based firefly algorithm (HBFA) for computing the optimal and binary solution of bound constrained nonlinear optimization problems. The problem to be addressed has the form:

$$\begin{array}{rl} \min & f(x) \\ \text{subject to} & x \in \Omega \subset \mathbb{R}^n \text{ (a compact convex set)} \\ & x_l \in \{0,1\} \text{ for } l = 1,\dots,n \end{array} \tag{1}$$

where f is a continuous function. Due to the compactness of Ω, we also have $Lb_l \leq x_l \leq Ub_l, l = 1,\dots,n$ where Lb and Ub are the vectors of the lower and upper bounds respectively. We do not assume that f is differentiable and convex. Instead of searching for any local (non-global) solution we want the globally best binary point. Direct search methods might be suitable since we do not assume differentiability. However, they are only local optimization procedures and therefore there is no guarantee that a global solution is reached. For global optimization, stochastic methods are generally used and aim to explore the search space and converge to a global solution. Metaheuristics are higher-level procedures or heuristics that are designed to search for good solutions, known as near-optimal solutions, with less computational effort and time than more classical algorithms. They are usually non-deterministic and their behaviors do not depend on problem's properties. Population-based metaheuristics have been used to solve a variety of optimization problems, from continuous to the combinatorial ones.

Metaheuristics are common for solving discrete binary optimization problems [4, 6, 13, 14, 15, 18, 21, 23, 24, 29]. Many approaches have been developed aiming to solve nonlinear programming problems with mixed-discrete variables by transforming the discrete problem into a continuous one [5]. The most used and simple approach solves the continuous relaxed problem and then discretizes the obtained solution by using a rounding scheme. This type of approach works well on simple and small dimension academic and benchmark problems but may be somehow limited on some real-world applications.

Recently, a metaheuristic optimization algorithm, termed as firefly algorithm (FA), that mimics the social behavior of fireflies based on the flashing and attraction characteristics of fireflies, has been developed

[30, 31]. This is a swarm intelligence optimization algorithm that is capable of competing with the most well-known algorithms, like ant colony optimization, particle swarm optimization, artificial bee colony, artificial fish swarm, and cuckoo-search.

FA performance is controlled by three parameters: the randomization parameter α, the attractiveness β, and the absorption coefficient γ. Authors have argued that its efficiency is due to its capability of subdividing the population into subgroups (since local attraction is stronger than long-distance attraction), and its ability to adapt the search to problem landscape by controlling the parameter γ [9, 36]. Several variants of the firefly algorithm do already exist in the literature. Based on the settings of their parameters a classification scheme has appeared. Gaussian FA [7], hybrid FA with harmony search [11], hybrid genetic algorithm with FA [8], self-adaptive step FA [36] and modified FA in [27] are just a few examples. Further improvements have been made aiming to accelerate convergence (see, for example, [17, 20, 34]). A practical convergence analysis of FA with different parameter sets is presented in [2]. FA has become popular and widely used in recent years in many applications, like economic dispatch problems [35] and mixed variable optimization problems [10]. The extension of FA to multiobjective continuous optimization has already been investigated [33]. A recent review of firefly algorithms is available in [9].

Based on the effectiveness of FA in continuous optimization, it is predicted that it will perform well when solving discrete optimization problems. Discrete versions of the FA are available for solving discrete NP hard optimization problems [16, 25].

The main purpose of this study is to incorporate some heuristics aiming to deal with binary variables in the firefly algorithm for solving nonlinear optimization problems with binary variables. The binary dealing methods that were implemented are adaptations of well-known heuristics for defining 0 and 1 bit strings from real variables. Furthermore, a new sigmoid function aiming to constrain a real valued variable to the range $[0,1]$ is also proposed. Three different implementations to incorporate the heuristics for binary variables and adapt FA to binary optimization are proposed. We apply the proposed heuristic strategies to solve a set of benchmark nonlinear problems and show that the newly developed HBFA is effective in binary nonlinear programming.

The remaining part of the paper is organized as follows. Section 2 reviews the standard FA and presents new dynamic updates for some FA parameters, and Section 3 describes different heuristic strategies and reports on their implementations to adapt FA to binary optimization. All the heuristic approaches are validated with tests on a set of well-known bound constrained problems. These results and a comparison with other methods in the literature are shown in Section 4. Finally, the conclusions and ideas for future work are listed in Section 5.

2 Firefly Algorithm

Firefly algorithm is a bio-inspired metaheuristic algorithm that is able to compute a solution to an optimization problem. It is inspired by the flashing behavior of fireflies at night. According to [27, 30, 31], the three main rules used to construct the standard algorithm are the following:

- all fireflies are unisex, meaning that any firefly can be attracted to any other brighter one;
- the attractiveness of a firefly is determined by its brightness which is associated with the encoded objective function;
- attractiveness is directly proportional to brightness but decreases with distance.

Throughout this paper, we let $\|\cdot\|$ to represent the Euclidean norm of a vector. We use the vector $x=(x_1,x_2,\ldots,x_n)^T$ to represent the position of a firefly in the search space. The position of the firefly j will be represented by $x^j \in \mathbb{R}^n$. We assume that the size of the population of fireflies is m. In the context of problem (1), firefly j is brighter than firefly i if $f(x^j) < f(x^i)$.

2.1 Standard FA

First, in the standard FA, the positions of the fireflies are randomly generated in the search space Ω, as follows:

$$x_l^i = Lb_l + rand(Ub_l - Lb_l), \text{ for } l=1,\ldots,n$$

where *rand* is a uniformly distributed random number in $[0,1]$, hereafter represented by $rand \sim U[0,1]$. The movement of a firefly i is attracted to another brighter firefly j and is given by:

$$x^i = x^i + \beta(x^j - x^i) + \alpha(rand - 0.5)S, \qquad (2)$$

where $\alpha \in [0,1]$ is the randomization parameter, $rand \sim U[0,1]$, $S \in \mathbb{R}^n$ is a problem dependent vector of scaling parameters, and

$$\beta = \beta_0 \exp\left(-\gamma \|x^i - x^j\|^p\right) \text{ for } p \geq 1 \qquad (3)$$

gives the attractiveness of a firefly which varies with the light intensity/brightness seen by adjacent fireflies and the distance between themselves and β_0 is the attraction parameter when the distance is zero [30, 31, 32, 34]. Besides the presented 'exp' function, any monotonically decreasing function could be used. The parameter γ characterizes the variation of the attractiveness – is the light absorption coefficient

– and is crucial to determine the speed of convergence of the algorithm. In theory, γ could take any value in the set $[0,\infty)$. When $\gamma \to 0$, the attractiveness is constant $\beta = \beta_0$, meaning that a flashing firefly can be seen anywhere in the search space. This is an ideal case for a problem with a single (usually global) optimum since it can easily be reached. On the other hand, when $\gamma \to \infty$, the attractiveness is almost zero in the sight of other fireflies and each firefly moves in a random way. In particular, when $\beta_0 = 0$, the algorithm behaves like a random search method [31, 34]. The randomization term can be extended to the normal distribution N(0,1) or to any other distribution [36]. Algorithm 1 presents the main steps of the standard FA (on continuous space).

```
Data: $k_{\max}$, $f^*$, $\eta$
Set $k = 0$;
Randomly generate $x^i \in \Omega$, $i = 1,\dots,m$;
Evaluate $f(x^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
while $k \le k_{\max}$ and $|f(x^1) - f^*| > \eta$ do
    forall $x^i$ such that $i = 2,\dots,m$ do
        forall $x^j$ such that $j = 1,\dots,i-1$ do
            Compute randomization term;
            Compute attractiveness $\beta$;
            Move firefly $i$ towards $j$ using (2);
    Evaluate $f(x^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
    Set $k = k+1$;
```

Algorithm 1: Standard FA

2.2 Dynamic updates of α and γ

The relative value of the parameters α and γ affects the performance of FA. The parameter α controls the randomness, or, to some extent, the diversity of solutions. Parameter γ aims to scale the attraction power of the algorithm. Small values of γ with large values of α can increase the number of iterations required to converge to an optimal solution. Experience has shown that α must take large values at the beginning of the iterative process to enforce the algorithm to increase the diversity of solutions. However, small α values combined with small values of γ in the final iterations increase the fine-tuning of solutions since the effort is focused on exploitation. Thus, it is possible to improve the quality of the solution by reducing the randomness. Convergence can be improved by varying the randomization parameter α so that it decreases gradually as the optimum solution is approaching [32, 33, 34, 35]. In order to improve convergence speed and solution accuracy, dynamic updates of the parameters α and γ of FA, which depend on the iteration counter k of the algorithm, are implemented as follows.

Similarly to the factor which controls the amplification of differential variations, in differential evolution (DE) metaheuristic [14], the inertial weight, in particle swarm optimization (PSO) [12, 32], and the pitch adjusting rate, in the harmony search (HS) algorithm [19], we allow the value of α to decrease linearly with k, from an upper level $\alpha_{\max}$ to a lower level $\alpha_{\min}$:

$$\alpha(k) = \alpha_{\max} - k\frac{\alpha_{\max} - \alpha_{\min}}{k_{\max}}, \tag{4}$$

where $k_{\max}$ is the maximum number of allowed iterations. To increase the attractiveness with k, the parameter γ is dynamically updated by

$$\gamma(k) = \gamma_{\max} \exp\left(\frac{k}{k_{\max}} \ln(\frac{\gamma_{\min}}{\gamma_{\max}})\right), \tag{5}$$

where $\gamma_{\min}$ and $\gamma_{\max}$ are the minimum and maximum variation of attractiveness respectively.

2.3 Lévy dependent randomization term

We remark that our implementation of the randomization term in the proposed dynamic FA considers the Lévy distribution. Based on the attractiveness β, in (3), the equation for the movement of firefly i towards a brighter firefly j can be written as follows:

$$x^i = x^i + y^i \text{ with } y^i = \beta(x^j - x^i) + \alpha L(x^1)\sigma_x^i, \tag{6}$$

where $L(x^1)$ is a random number from the Lévy distribution centered at x^1, the position of the brightest firefly, with an unitary standard deviation. The vector σ_x^i represents the variation around x^1 (and based on real position x)

$$\sigma_x^i = \left(|x_1^i - x_1^1|, \dots, |x_n^i - x_n^1|\right)^T.$$

3 Dealing with Binary Variables

The standard FA is a real-coded algorithm and some modifications are needed to enable it to deal with discrete optimization problems. This section describes the implementation of some heuristics with FA for binary nonlinear optimization problems. In the context of the proposed HBFA, three different heuristics to transform a continuous real variable into a binary one are presented. Furthermore, to extend FA to binary optimization, different implementations to incorporate the heuristic strategies into FA are described. We will use the term 'discretization' to define the process that transforms a continuous real variable, represented for example by x, into a binary one, represented by b.

3.1 Sigmoid logistic function

This discretization methodology is the most common in the literature when population-based stochastic algorithms are considered in binary optimization, namely PSO [15, 21, 23], DE [6], HS [22, 29], artificial fish swarm [3], artificial bee colony [13, 18, 24].

When x^i moves towards x^j, the likelihood is that the discrete components of x^i change from binary numbers to real ones. To transform a real into a binary number, the following sigmoid logistic function constrains the real value to the interval $[0,1]$:

$$\mathrm{sig}(x_l^i) = \frac{1}{1+\exp\left(-x_l^i\right)} \tag{7}$$

where x_l^i, in the context of FA, is the component l of the position vector x^i (of firefly i) after movement – recall (6) and (3). The equation (7) interprets the floating-point components of a solution as a set of probabilities. These are then used to assign appropriate binary values by using:

$$b_l^i = \begin{cases} 1, \text{ if } rand \leq \mathrm{sig}(x_l^i) \\ 0, \text{otherwise} \end{cases} \tag{8}$$

where $\mathrm{sig}(x_l^i)$ gives the probability that the component itself is 0 or 1 [25] and $rand \sim U[0,1]$. We note that during the iterative process the firefly positions, x, were not allowed to move outside the search space Ω.

3.2 Proposed sigmoid `erf` function

The error function is a special function with a shape that appears mainly in probability and statistics contexts. Denoted by '`erf`', this is a mathematical function defined by the integral

$$\mathtt{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)\,dt,$$

satisfies the following properties

$$\begin{gathered} \mathtt{erf}(0)=0, \mathtt{erf}(-\infty)=-1, \mathtt{erf}(+\infty)=1, \\ \mathtt{erf}(-x) = -\mathtt{erf}(x), \end{gathered} \tag{9}$$

and has a close relation with the normal distribution probabilities. When a series of measurements are described by a normal distribution with mean 0 and standard deviation σ, the `erf` function evaluated at $\frac{x}{\sigma\sqrt{2}}$, for a positive x, gives the probability that the error of a single measurement lies in the interval $[-x,x]$. The derivative of the `erf` function follows immediately from its definition:

$$\frac{d}{dt}\mathtt{erf}(t) = \frac{2}{\sqrt{\pi}}\exp(-t^2), \text{ for } t \in \mathbb{R}. \tag{10}$$

The good properties of the `erf` function are thus used to define a new sigmoid function – the sigmoid `erf` function:

$$\mathrm{sig}(x_l^i) = 0.5(1+\mathtt{erf}(x_l^i)), \tag{11}$$

which is a bounded differentiable real function, defined for all $x \in \mathbb{R}$ and has a positive derivative at each point. A comparison of both functions (7) and (11) is depicted in Figure 1. Note that the slope at the origin of the sigmoid function in (11) is around 0.5641895, while that of function (7) is 0.25, thus yielding a faster growing from 0 to 1.

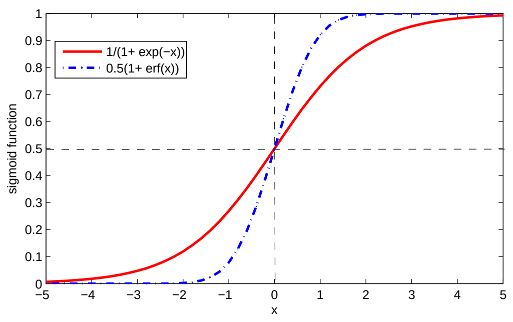

Figure 1: Sigmoid functions

3.3 Rounding to integer part

The simplest discretization procedure of a continuous component of a point into a 0/1 bit uses the rounding to the integer part function, known as floor function, and is described in [26]. Each continuous value $x_l^i \in \mathbb{R}$ is transformed into a binary one, 0 bit or 1 bit, b_l^i, for $l = 1,\ldots,n$ in the following way

$$b_l^i = \lfloor |x_l^i \bmod 2| \rfloor \tag{12}$$

where $\lfloor z \rfloor$ represents the floor function of z, and gives the largest integer not greater than z. The floating-point value x_l^i is first divided by 2 and then the absolute value of the remainder is floored. The obtained integer number is the bit value of the component.

3.4 Heuristics implementation

In this study, three methods capable of computing global solutions to binary optimization problems using FA are proposed.

3.4.1 Movement on continuous space

In this implementation of the previously described heuristics, denoted by 'movement on continuous space' (mCS), the movement of each firefly is made on the continuous space and its attractiveness term is updated considering the real position vector. The real position of firefly i is discretized only after all movements towards brighter fireflies have been carried out. We note that the fitness evaluation of each firefly, for firefly ranking, is always based on the binary position.

Algorithm 2 presents the main steps of HBFA with mCS.

```
Data: $k_{\max}$, $f^*$, $\eta$
Set $k = 0$;
Randomly generate $x^i \in \Omega$, $i = 1,\dots,m$;
Discretize position of firefly $i$: $x^i \to b^i$, $i = 1,\dots,m$;
Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
while $k \le k_{\max}$ and $|f(b^1) - f^*| > \eta$ do
    forall $x^i$ such that $i = 2,\dots,m$ do
        forall $x^j$ such that $j = 1,\dots,i-1$ do
            Compute randomization term;
            Compute attractiveness $\beta$;
            Move position $x^i$ of firefly $i$ towards $x^j$ using (6);
    Discretize positions: $x^i \to b^i$, $i = 1,\dots,m$;
    Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
    Set $k = k + 1$;
```

Algorithm 2: HBFA with mCS

3.4.2 Movement on binary space

This implementation, denoted by 'movement on binary space' (mBS), moves the binary position of each firefly towards the binary positions of brighter fireflies, i.e., each movement is made on the binary space although the corresponding position may fail to be a 0 or 1 bit string and must be discretized before the updating of attractiveness. Here, fitness is also based on the binary positions. Algorithm 3 presents the main steps of HBFA with mBS.

```
Data: $k_{\max}$, $f^*$, $\eta$
Set $k = 0$;
Randomly generate $x^i \in \Omega$, $i = 1,\dots,m$;
Discretize position of firefly $i$: $x^i \to b^i$, $i = 1,\dots,m$;
Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
while $k \le k_{\max}$ and $|f(b^1) - f^*| > \eta$ do
    forall $b^i$ such that $i = 2,\dots,m$ do
        forall $b^j$ such that $j = 1,\dots,i-1$ do
            Compute randomization term;
            Compute attractiveness $\beta$ based on distance $\|b^i - b^j\|^p$;
            Move binary position $b^i$ of firefly $i$ towards $b^j$ using $x^i = b^i + \beta(b^j - b^i) + \alpha L(b^1)\sigma_b^i$;
            Discretize position of firefly $i$: $x^i \to b^i$;
    Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
    Set $k = k + 1$;
```

Algorithm 3: HBFA with mBS

3.4.3 Probability for binary component

For this implementation, named 'probability for binary component' (pBC), we borrow the concept from the binary PSO [15, 23, 28] where each component of the velocity vector is directly used to compute the probability that the corresponding component of the particle position, x_l^i, is 0 or 1. Similarly, in the FA algorithm, we do not interpret the vector y^i in (6) as a step size, but rather as a mean to compute the probability that each component of the position vector of firefly i is 0 or 1. Thus, we define

$$b_l^i = \begin{cases} 1, \text{ if } rand \le \text{sig}(y_l^i) \\ 0, \text{otherwise} \end{cases}, \tag{13}$$

where sig() represents a sigmoid function. Algorithm 4 is the pseudo code of HBFA with pBC.

```
Data: $k_{\max}$, $f^*$, $\eta$
Set $k = 0$;
Randomly generate $x^i \in \Omega$, $i = 1,\dots,m$;
Discretize position of firefly $i$: $x^i \to b^i$, $i = 1,\dots,m$;
Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
while $k \le k_{\max}$ and $|f(b^1) - f^*| > \eta$ do
    forall $b^i$ such that $i = 2,\dots,m$ do
        forall $b^j$ such that $j = 1,\dots,i-1$ do
            Compute randomization term;
            Compute attractiveness $\beta$ based on distance $\|b^i - b^j\|^p$;
            Compute $y^i$ using binary positions (see (6));
            Discretize $y^i$ and define $b^i$ using (13);
    Compute $f(b^i), i = 1,\dots,m$, rank fireflies (from lowest to largest $f$);
    Set $k = k + 1$;
```

Algorithm 4: HBFA with pBC

4 Numerical Experiments

In this section, we present the computational results that were obtained with HBFA – Algorithms 2, 3 and 4, using (7), (11) or (12) – aiming to investigate its performance when solving a set of binary nonlinear optimization problems. Two small 0–1 knapsack problems are also used to test the algorithms' behavior on linear problems with 0/1 variables.

The numerical experiments were carried out on a PC Intel Core 2 Duo Processor E7500 with 2.9GHz and 4Gb of memory. The algorithms were coded in Matlab Version 8.0.0.783 (R2012b).

4.1 Experimental setting

Each experiment was conducted 30 times. The size of the population is made to depend on the problem's dimension and is set to $m = \min\{40, 2^n\}$. Some experiments have been carried out to tune certain parameters of the algorithms. In the proposed FA with dynamic α and γ, they are set as follows: $\beta_0 = 1$, $p = 1$, $\alpha_{\max} = 0.5$, $\alpha_{\min} = 0.01$, $\gamma_{\max} = 10$, $\gamma_{\min} = 0.1$. In Algorithms 2 (mCS), 3 (mBS) and 4 (pBC), iterations were limited to $k_{\max} = 500$ and the tolerance for finding a good quality solution is $\eta = 10^{-6}$. Results reported are averaged (over the 30 runs) of best function values, number of function evaluations and number of iterations.

4.2 Experimental results

First, we use a set of ten benchmark nonlinear functions with different dimensions and characteristics. For example, five functions are unimodal and the remaining multimodal [1, 6, 23, 24]. They are displayed in Table 1. Although they are widely used in continuous optimization, we now aim to converge to a 0/1 bit string solution.

First, we aim to compare with the results reported in [6, 23, 24]. Due to poor results, the authors in [24] do not advice the use of ABC to solve binary-valued problems. The other metaheuristics therein implemented are the following:

- angle modulated PSO (AMPSO) and angle modulated DE (AMDE) that incorporate a trigonometric function as a bit string generator into the classic PSO and DE algorithms respectively;
- binary DE and PSO based on the sigmoid logistic function and (8), denoted by binDE and binPSO respectively.

We noticed that the problems Foxholes, Griewank, Rosenbrock, Schaffer and Step are not correctly described in [6, 23, 24]. Table 2 shows both the averaged best function values (obtained during the 30 runs), f_{avg}, with the St.D. in parentheses, and the averaged number of function evaluations, nfe, obtained with the sigmoid logistic function (see in (7)) and (8), while using the three implementations: mCS, mBS and pBC. Results obtained for these ten functions indicate that our proposal HBFA produces high quality solutions and outperforms the binary versions binPSO and binDE, as well as AMPSO and AMDE. We also note that mCS has the best 'nfe' values on 6 problems, mBS is better on 3 (one is a tie with mCS) and pBC on 2 problems. Thus the performance of mCS is the best when compared with those of mBS and pBC. The latter is the least efficient of all, in particular for the large dimensional problems.

To analyze the statistical significance of the results we perform a Friedman test. This is a non-parametric statistical test to determine significant differences in mean for one independent variable with two or more levels - also denoted as treatments - and a dependent variable (or matched groups taken as the problems). The null hypothesis in this test is that the mean ranks assigned to the treatments under testing are the same. Since all three implementations are able to reach the solutions within the η error tolerance on 9 out of 10 problems, the statistical analysis is based on the performance criterion 'nfe'. In this hypothesis testing, we have three treatments and ten groups. Friedman's chi-square has a value of 2.737 (with a p-value of 0.255). For a 2 degrees of freedom reference χ^2 distribution, the critical value for a significance level of 5% is 5.99. Hence, since $2.737 \leq 5.99$, the null hypothesis is not rejected and we conclude that there is no evidence that the three mean ranks values have statistically significant differences.

To further compare the sigmoid functions with the rounding to integer strategy, we include in Table 3 the results obtained by the '`erf`' function in (11), together with (8), and the floor function in (12). Only the implementations mCS and mBS are tested. The table also shows the averaged number of iterations, *nit*. The results illustrate that implementation mCS (Algorithm 2) works very well with strategies based on (11), together with (8), and (12). The success rate for all the problems is 100%, meaning that the algorithms stop because the f value at the position of the best/brightest firefly is within a tolerance η of the optimal solution f^*, in all runs. Further, mBS (Algorithm 3) works better when the discretization of the variables are carried out by equation (12). Overall, mCS based on (11) produces the best results on 6 problems, mCS based on (12) gives the best results on 7 problems (including 4 ties with the former case), mBS based on (11) wins only on one problem and mBS based on (12) wins on 3 problems (all are ties with mCS based on (12)).

Further, when performing the Friedman test on the four distributions of 'nfe' values, the chi-square statistical value is 13.747 (and the p-value is 0.0033). From the χ^2 distribution table, the critical value for a significance level of 5% and 3 degrees of freedom is 7.81. Since $13.747 > 7.81$, the null hypothesis is rejected and we conclude that the observed differences of the four distributions are statistically significant.

We now introduce in the statistical analysis the results reported in Tables 2 and 3 concerned with both implementations mCS and mBS. Six distributions of 'nfe' values are now in comparison. The Friedman's chi-square value is 18.175 (p-value=0.0027). The critical value of the chi-square distribution for a significance level of 5% and 5 degrees of freedom is 11.07. Thus, The null hypothesis of "no significant differences on mean ranks" is rejected and there is evidence that the six distributions of 'nfe' values have statistically significant differences. Multiple comparisons (two at a time) may be carried out to determine which mean ranks are significantly different. The estimates of the 95% confidence intervals are shown in the graph of Figure 2 for each case under testing. Two compared distributions of 'nfe' are significantly different if their intervals are disjoint and are not significantly different if their intervals overlap. Hence, from the six cases, we conclude that the mean ranks produced by mCS based on (12) is significantly different from those of mBS based on (7) and mBS based on (11). For the remaining pairs of comparison there are no significant differences on the mean ranks.

Table 1: Problems set

Ackley	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$ $n = 30$, $\Omega = [-30,30]^{30}$, $f^* = 0$ at $x^* = (0,\dots,0)$
Foxholes	$f(x) = 1/\left(0.002 + \sum_{j=1}^{25} \frac{1}{j+(x_1-a_{1j})^6+(x_2-a_{2j})^6}\right)$ $[a_{ij}] = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & \dots & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & \dots & 32 & 32 \end{bmatrix}$ $n = 2$, $\Omega = [-65.536, 65.536]^2$, $f^* \approx 13$ at $x^* = (0,0)$
Griewank	$f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$ $n = 30$, $\Omega = [-300,300]^{30}$, $f^* = 0$ at $x^* = (0,\dots,0)$
Quartic	$f(x) = \sum_{i=1}^{n} i x_i^4 + U[0,1]$ $n = 30$, $\Omega = [-1.28,1.28]^{30}$, $f^* = 0 + \text{noise}$ at $x^* = (0,\dots,0)$
Rastrigin	$f(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$ $n = 30$, $\Omega = [-5.12,5.12]^{30}$, $f^* = 0$ at $x^* = (0,\dots,0)$
Rosenbrock2	$f(x) = 100(x_1^2 - x_2)^2 + (1-x_1)^2$ $n = 2$, $\Omega = [-2.048,2.048]^2$, $f^* = 0$ at $x^* = (1,1)$
Rosenbrock	$f(x) = \sum_{i=1}^{n-1} 100(x_i^2 - x_{i+1})^2 + (1-x_i)^2$ $n = 30$, $\Omega = [-2.048,2.048]^{30}$, $f^* = 0$ at $x^* = (1,\dots,1)$
Schaffer	$f(x) = 0.5 + \left(\left(\sin(\sqrt{x_1^2+x_2^2})\right)^2 - 0.5\right) / \left(1 + 0.001(x_1^2+x_2^2)\right)^2$ $n = 2$, $\Omega = [-100,100]^2$, $f^* = 0$ at $x^* = (0,0)$
Spherical	$f(x) = \sum_{i=1}^{n} x_i^2$ $n = 3$, $\Omega = [-5.12,5.12]^3$, $f^* = 0$ at $x^* = (0,0,0)$
Step	$f(x) = 6n + \sum_{i=1}^{n}\lfloor x_i \rfloor$ $n = 5$, $\Omega = [-5.12,5.12]^5$, $f^* = 30$ at $x^* = (0,0,0,0,0)$

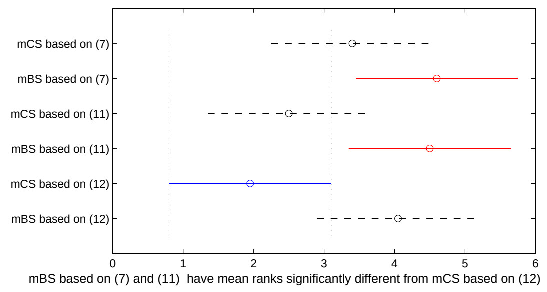

Figure 2: Confidence intervals for mean ranks of nfe

For comparative purposes we include in Table 4 the results obtained by using the proposed Lévy (**L**) distribution in the randomization term, as shown in (6), and those produced by the Uniform (**U**) distribution, using $rand \sim U[0,1]$ as shown in (2). The reported tests use implementation mCS (described in Algorithm 2) with the two heuristics for binary variables: i) the '`erf`' function in (11), together with (8); ii) the floor function in (12). It is shown that the performance of HBFA with Uniform distribution is very sensitive to the dimension of the problem, since the efficiency is good when n is small but gets worse when n is large. Thus, we have shown that the Lévy distribution is a very good bid.

We add to some problems with $n = 30$ from Table 1 – Ackley, Griewank, Rastrigin, Rosenbrock, Spherical – three other functions Schwefel 2.22, Schwefel 2.26 and Sum of Different Power to compare our results with those reported in [29]. Schwefel 2.22 is unimodal and for $\Omega = [-10,10]^{30}$, the binary solution is $(0,\dots,0)$ with $f^* = 0$; Schwefel 2.26 is multimodal and in $\Omega = [-500,500]^{30}$, the binary solution is $(1,1,\dots,1)$ with $f^* = -25.244129544$; Sum of Different Power is unimodal and in $\Omega = [-1,1]^{30}$, the minimum is 0 at $(0,\dots,0)$. For the results of Table 5, we use HBFA based on mCS, with both '`erf`' function in (11), together with (8), and the floor function (12). The table reports on the average function values, average number of function evaluations and success rate (SR). Here, 50 independent runs were carried out to compare with the results shown in [29]. The maximum number of function evaluations therein used was 90000. It is shown that our HBFA outperforms the proposed adaptive binary harmony search (ABHS).

4.3 Effect of problem's dimension on HBFA performance

We now consider six problems with varied dimensions from the previous set to analyze the effect of problem's dimension on the HBFA performance. We test three dimensions: $n = 50$, $n = 100$ and $n = 200$. The algorithm's parameters are set as previously defined. We remark that the size of the population for all the tested problems and dimensions is 40 points.

Table 6 contains the results for comparison based on averaged values of f, number of function evaluations and number of iterations. The 'St.D.' of the f values are also displayed. Since the implementation mCS, shown in Algorithm 2, performs better and shows more consistent results than the other two, we tested only

Table 2: Comparison with AMPSO, binPSO, binDE, AMDE, based on f_{avg} and St.D. (shown in parentheses)

	mCS based on (7)		mBS based on (7)		pBC based on (7)		AMPSO	binPSO	binDE	AMDE
Prob.	f_{avg}	nfe	f_{avg}	nfe	f_{avg}	nfe	f_{avg}	f_{avg}	f_{avg}	f_{avg}
Ackley	8.88e-16	80	8.88e-16	1156	8.88e-16	2168	1.97e01	2.01e01	1.73e01	1.64e01
	(0.00e00)		(0.00e00)		(0.00e00)		(0.57e-01)	(0.49e-01)	(0.31e01)	(0.76e00)
Foxholes	1.27e01	6.1	1.27e01	7.2	1.27e01	6.3	0.53e-10	0.53e-10	1.29e01	5.0e02
	(9.03e-15)		(9.03e-15)		(9.03e-15)		(0.00e00)	(0.97e-14)	(0.86e00)	(0.0e00)
Griewank	0.00e00	80	0.00e00	1332	0.00e00	2300	1.06e02	6.79e01	2.63e02	2.06e02
	(0.00e00)		(0.00e00)		(0.00e00)		(0.44e01)	(0.98e01)	(0.10e02)	(0.39e01)
Quartic	4.55e-01	2012	5.37e-01	1771	4.88e-01	2951	4.15e01	2.09e01	1.49e00	3.55e00
	(3.01e-01)		(3.03e-01)		(2.71e-01)		(0.19e01)	(0.19e01)	(0.66e00)	(0.81e00)
Rastrigin	0.00e00	80	0.00e00	1282.7	0.00e00	2406.7	2.25e02	3.08e02	2.14e02	9.05e01
	(0.00e00)		(0.00e00)		(0.00e00)		(0.35e02)	(0.28e01)	(0.45e02)	(0.31e02)
Rosenbrock2	0.00e00	6.5	0.00e00	5.7	0.00e00	6.1	0.49e-04	0.14e-03	0.20e-05	0.55e-04
	(0.00e00)		(0.00e00)		(0.00e00)		(0.11e-03)	(0.88e-04)	(0.15e-04)	(0.17e-04)
Rosenbrock	0.00e00	180	0.00e00	2190.7	0.00e00	2088	2.20e03	2.24e03	1.81e03	9.14e01
	(0.00e00)		(0.00e00)		(0.00e00)		(0.86e02)	(0.77e02)	(0.16e02)	(0.42e02)
Schaffer	0.00e00	6.1	0.00e00	8	0.00e00	4.9	0.24e-01	0.73e-01	-0.995e00	-1.0e00
	(0.00e00)		(0.00e00)		(0.00e00)		(0.42e-02)	(0.11e-01)	(0.27e-06)	(0.0e00)
Spherical	0.00e00	12	0.00e00	12	0.00e00	11.7	0.30e-03	0.30e-03	0.15e-03	0.20e-04
	(0.00e00)		(0.00e00)		(0.00e00)		(0.00e00)	(0.0e00)	(0.41e-04)	(0.15e-04)
Step	3.00e01	42.7	3.00e01	42.7	3.00e01	48	0.00e00	0.17e-04	0.15e-01	0.25e00
	(0.00e00)		(0.00e00)		(0.00e00)		(0.00e00)	(0.15e-04)	(0.0e00)	(0.60e-01)

Table 3: Comparison mCS *vs.* mBS and (11) *vs.* (12), based on f_{avg}, nfe and nit

	mCS based on (11)			mBS based on (11)			mCS based on (12)			mBS based on (12)		
Prob.	f_{avg}	nfe	nit	f_{avg}	nfe	nit	f_{avg}	nfe	nit	f_{avg}	nfe	nit
Ackley	8.88e-16	80	1	8.88e-16	678.7	16	8.88e-16	80	1	8.88e-16	82.7	1.1
Foxholes	1.27e01	5.7	0.4	1.27e01	8	1	1.27e01	6.1	0.5	1.29e01	404.8	100.2
Griewank	0.00e00	80	1	0.00e00	717.3	16.9	0.00e00	80	1	0.00e00	82.7	1.1
Quartic	2.38e-01	81.3	1.03	4.66e-01	794.7	18.9	9.48e-02	80	1	3.93e-01	80	1
Rastrigin	0.00e00	80	1	0.00e00	702.7	16.6	0.00e00	80	1	0.00e00	80	1
Rosenbrock2	0.00e00	6.4	0.6	0.00e00	5.3	0.3	0.00e00	6.3	0.6	2.33e-01	470.8	116.7
Rosenbrock	0.00e00	80	1	0.00e00	105.3	1.6	0.00e00	80	1	2.90e01	20040	500
Schaffer	0.00e00	6.8	0.7	0.00e00	12.9	2.2	0.00e00	5.1	0.3	7.08e-02	205.1	50.3
Spherical	0.00e00	9.9	0.2	0.00e00	22.7	1.8	0.00e00	10.1	0.3	0.00e00	11.5	0.4
Step	3.00e01	45.9	0.4	3.00e01	62.9	1	3.00e01	40.5	0.3	3.00e01	40.5	0.3

mCS based on (11) and mCS based on (12).

Besides testing significant differences on the mean ranks produced by the two treatments – mCS based on (11) and mCS based on (12) – we also want to determine if the differences on mean ranks produced by problem's dimension – 50, 100 and 200 – are statistically significant at a significance level of 5%. Hence, we aim to analyze the effects of two factors 'A' and 'B'. 'A' is the HBFA implementation (with two levels) and 'B' is the problem's dimension (with three levels). For this purpose, the results obtained for the six problems for each combination of the levels of 'A' and 'B' are considered as replications. When performing the Friedman test for factor 'A', the chi-square statistical value is 1.225 (p-value=0.2685) with 1 degree of freedom. The critical value for a significance level of 5% and 1 degree of freedom in the χ^2 distribution table is 3.84, and there is no evidence of statistically significant differences. From the Friedman test for factor 'B', we also conclude that there is no evidence of statistically significant differences, since the chi-square statistical value is 0.746 (p-value=0.6886) with 2 degrees of freedom. (The critical value of the χ^2 distribution table for a significance level of 5% and 2 degrees of freedom is 5.99.) Hence, we conclude that the dimension of the problem does not affect the algorithm's performance. Only with problem Quartic, the efficiency of mCS based on (11) gets worse as dimension increases. Overall both tested strategies are rather effective when binary solutions are required on small as well as on large nonlinear optimization problems.

4.4 Solving 0–1 knapsack problems

Finally, we aim to analyze the behavior of our best tested strategies when solving well-known binary and linear optimization problems. For this preliminary experiment, we selected two small knapsack problems. The 0–1 knapsack problem (KP) can be described as follows. Let n be the number of items, from which we have to select some of them to be carried in a knapsack. Let w_l and v_l be the weight and the value of item

Table 4: Comparison between Lévy and Uniform distributions in the randomization term, based on f_{avg}, nfe and nit (with St.D. in parentheses)

	mCS based on (11) + **L**			mCS based on (11) + **U**			mCS based on (12) + **L**			mCS based on (12) + **U**		
Prob.	f_{avg}	nfe	nit	f_{avg}	nfe	nit	f_{avg}	nfe	nit	f_{avg}	nfe	nit
Ackley	8.88e-16 (0.00e00)	80	1	8.88e-16 (0.00e00)	1096	26.4	8.88e-16 (0.00e00)	80	1	1.41e00 (1.26e-02)	20040	500
Foxholes	1.27e01 (9.03e-15)	5.7	0.4	1.27e01 (9.03e-15)	8	1	1.27e01 (9.03e-15)	6.1	0.5	1.27e01 (9.03e-15)	6.8	0.7
Griewank	0.00e00 (0.00e00)	80	1	0.00e00 (0.00e00)	2436	59.9	0.00e00 (0.00e00)	80	1	1.27e-01 (2.76e-02)	20040	500
Quartic	2.38e-01 (1.70e-01)	81.3	1.03	6.90e00 (1.16e01)	14692	366.2	9.48e-02 (1.03e-01)	80	1	5.10e-01 (2.90e-01)	10581	263.5
Rastrigin	0.00e00 (0.00e00)	80	1	0.00e00 (0.00e00)	1157	27.9	0.00e00 (0.00e00)	80	1	1.00e-01 (4.03e-01)	18104	451.6
Rosenbrock2	0.00e00 (0.00e00)	6.4	0.6	0.00e00 (0.00e00)	36.8	8.2	0.00e00 (0.00e00)	6.3	0.6	0.00e00 (0.00e00)	5.9	0.5
Rosenbrock	0.00e00 (0.00e00)	80	1	8.22e01 (1.02e02)	14136	352.4	0.00e00 (0.00e00)	80	1	7.42e01 (8.43e01)	17698	441.2
Schaffer	0.00e00 (0.00e00)	6.8	0.7	0.00e00 (0.00e00)	18.5	3.6	0.00e00 (0.00e00)	5.1	0.3	0.00e00 (0.00e00)	5.6	0.4
Spherical	0.00e00 (0.00e00)	9.9	0.2	0.00e00 (0.00e00)	13.3	0.7	0.00e00 (0.00e00)	10.1	0.3	0.00e00 (0.00e00)	14.7	0.8
Step	3.00e01 (0.00e00)	45.9	0.4	3.00e01 (0.00e00)	46.9	0.5	3.00e01 (0.00e00)	40.5	0.3	3.00e01 (0.00e00)	52.3	0.6

Table 5: Comparison of HBFA (with mCS) with ABHS in [29], based on f_{avg}, nfe and SR (%)

	mCS based on (11)			mCS based on (12)			ABHS in [29]		
Prob.	f_{avg}	nfe	SR (%)	f_{avg}	nfe	SR (%)	f_{avg}	nfe	SR (%)
Ackley	8.88e-16	80	100	8.88e-16	80	100	1.56e-01	62350	90
Griewank	0.00e00	80	100	0.00e00	80	100	3.30e-02	79758	38
Rastrigin	0.00e00	80	100	0.00e00	80	100	1.32e01	90000	0
Rosenbrock	0.00e00	80	100	0.00e00	80	100	6.80e02	90000	0
Schwefel 2.22	0.00e00	80	100	0.00e00	80	100	0.00e00	59870	100
Schwefel 2.26	-2.52e01	80	100	-2.47e01	10867	87	-1.195e04	90000	0
Spherical	0.00e00	80	100	0.00e00	80	100	0.00e00	62234	100
Sum Different Power	0.00e00	91	100	0.00e00	168	100	0.00e00	80371	100

l respectively, and let W be the knapsack's capacity. It is usually assumed that all weights and values are non-negative. The objective is to maximize the total value of the knapsack under the constraint of the knapsack's capacity:

$$\begin{array}{ll} \max_x & V(x) \equiv \sum_{l=1}^{n} v_l x_l \\ \text{s.t.} & \sum_{l=1}^{n} w_l x_l \leq W,\ x_l \in \{0,1\}, l = 1,\ldots,n. \end{array}$$

If item l is selected, $x_l = 1$; otherwise, $x_l = 0$. Using a penalty function, this problem can be transformed into

$$\min_x - \sum_{l=1}^{n} v_l x_l + \mu \max\{0, \sum_{l=1}^{n} w_l x_l - W\}$$

where μ is the penalty parameter which was set to be 100 in this experiment.

Case 1: an instance of a 0–1 KP with 4 items. The knapsack's capacity is $W = 6$ and the vectors of values and weights are $v = (40,15,20,10)$ and $w = (4,2,3,1)$. Based on the above mentioned parameters, the HBFA with mCS based on (11) was run 30 times and the averaged results were the following. With a success rate of 100%, items 1 and 2 are included in the knapsack, items 3 and 4 are excluded, with a maximum value of 55 (St.D.= 0.0e00). on average, the runs required 0.8 iterations and 29.3 function evaluations. With a success rate of 23%, the heuristic based on the floor function, thus mCS based on (12), reached $f_{\text{avg}} = 49$ (St.D.= 4.0e00) after an average of 6161.1 function evaluations and an average of 384.1 iterations.

Case 2: an instance of a 0–1 KP with 8 items. The maximum capacity of the knapsack is set to 8 and the vectors of values and weights are $v = (83,14,54,79,72,52,48,62)$ and $w = (3,2,3,2,1,2,2,3)$. The results are averaged over the 30 runs. After 8.7 iterations and 386.7 function evaluations, the maximum value produced by the strategy mCS based on (11) is 286 (St.D. = 0.0e00), with a success rate of 100%. Items 1,4,5,6 are included in the knapsack and the others are excluded. The heuristic

Table 6: Results for varied dimensions ($n = 50, 100, 200$), considering $m = 40$

			mCS based on (11)				mCS based on (12)			
	f^*	n	f_{avg}	St.D.	nfe	nit	f_{avg}	St.D.	nfe	nit
Ackley	0.00e+00	50	8.88e-16	0.00e+00	80	1	8.88e-16	0.00e+00	80	1
	0.00e+00	100	8.88e-16	0.00e+00	80	1	8.88e-16	0.00e+00	80	1
	0.00e+00	200	8.88e-16	0.00e+00	80	1	8.88e-16	0.00e+00	80	1
Griewank	0.00e+00	50	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	100	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	200	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
Quartic	0.00e+00+noise	50	2.24e-01	1.95e-01	82.7	1.1	1.43e-01	1.59e-01	80	1
	0.00e+00+noise	100	4.32e-01	2.73e-01	146.7	2.7	1.73e-01	1.10e-01	80	1
	0.00e+00+noise	200	5.23e-01	2.94e-01	1738.7	42.5	1.96e-01	2.13e-01	81.3	1.03
Rosenbrock	0.00e+00	50	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	100	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	200	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
Spherical	0.00e+00	50	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	100	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
	0.00e+00	200	0.00e+00	0.00e+00	80	1	0.00e+00	0.00e+00	80	1
Step	3.00e+02	50	3.00e+02	0.00e+00	80	1	3.00e+02	0.00e+00	80	1
	6.00e+02	100	6.00e+02	0.00e+00	80	1	6.00e+02	0.00e+00	80	1
	1.20e+03	200	1.20e+03	0.00e+00	80	1	1.20e+03	0.00e+00	80	1

mCS based on (12) did not reached the optimal solution. All runs required 500 iterations and 20040 function evaluations and the average function values was $f_{\text{avg}} = 227$ with St.D.= 3.14e01.

5 Conclusions and Future Work

In this work we have implemented several heuristics to compute a global optimal binary solution of bound constrained nonlinear optimization problems, which have been incorporated into FA, yielding the herein called HBFA. The problems addressed in this study have bounded continuous search space. Our FA proposal uses dynamic updating schemes for two parameters, γ from the attractiveness term and α from the randomization term, and considers the Lévy distribution to create randomness in firefly movement. The performance behavior of the proposed heuristics have been investigated. Three simple heuristics capable of transforming real continuous variables into binary ones are implemented. A new sigmoid '`erf`' function is proposed. In the context of the firefly algorithm, three different implementations aiming to incorporate the heuristics for binary variables into FA are proposed (mCS, mBS and pBC). Based on a set of benchmark problems, a comparison is carried out with other binary dealing metaheuristics, namely AMPSO, binPSO, binDE and AMDE. The experimental results show that the implementation denoted by mCS, when combined with either the new sigmoid '`erf`' function or with the rounding scheme based on the floor function, is quite efficient and superior to the other methods in comparison. The statistical analysis carried out on the results shows evidence of statistically significant differences on efficiency, measured by the number of function evaluations, between the implementation mCS based on the floor function approach and the mBS based on both tested sigmoid functions schemes. We have also investigated the effect of problem's dimension on the performance of our algorithm. Using the Friedman statistical test we conclude that the differences on efficiency are not statistically significant. Another simple experiment has shown that the implementation mCS with the sigmoid '`erf`' function is effective in solving two small 0–1 KP. The performance of this simple heuristic strategy will be further analyzed to solve large and multidimensional 0–1 KP.

Future developments concerning the HBFA will consider its extension to deal with integer variables in nonlinear optimization problems. Different heuristics to transform continuous real variables into integer variables will be investigated. Challenging mixed-integer nonconvex nonlinear problems will be solved.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this article.

Acknowledgments

The authors wish to thank two anonymous referees for their valuable suggestions to improve the paper.

This work has been supported by FCT (Fundação para a Ciência e Tecnologia, Portugal) in the scope of the projects: PEst-OE/MAT/UI0013/2014 and PEst-OE/EEI/UI0319/2014.

References

[1] M.M. Ali, C. Khompatraporn, Z.B. Zabinsky, A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems, *Journal of Global Optimization*, vol. 31, pp. 635–672, 2005.

[2] S. Arora, S. Singh, The firefly optimization algorithm: convergence analysis and parameter selection, *International Journal of Computer Applications*, vol. 69(3), pp. 48–52, 2013.

[3] M.A.K. Azad, A.M.A.C. Rocha, E.M.G.P. Fernandes, A simplified binary artificial fish swarm algorithm for uncapacitated facility location problems, in *Proceedings of World Congress on Engineering 2013*, S.I. Ao, L. Gelman, D.W.L. Hukins, A. Hunter and A.M. Korsunsky (Eds.) Vol. I, pp. 31–36, IAENG London, UK, 2013.

[4] M.A.K. Azad, A.M.A.C. Rocha, E.M.G.P. Fernandes, Improved binary artificial fish swarm algorithm for the 0-1 multidimensional knapsack problems, *Swarm and Evolutionary Computation*, Vol. 14 pp. 66–75, 2014.

[5] S. Burer, A.N. Letchford, Non-convex mixed-integer nonlinear programming: a survey, *Surveys in Operations Research and Management Science*, vol. 17, pp. 97–106, 2012.

[6] A.P. Engelbrecht, G. Pampará, Binary differential evolution strategies, in *Proceedings of 2007 IEEE Congress on Evolutionary Computation*, pp. 1942–1947, 2007.

[7] Sh.M. Farahani, A.A. Abshouri, B. Nasiri, M.R. Meybodi, A Gaussian firefly algorithm, *International Journal of Machine Learning and Computing*, vol. 1(5), pp. 448–453, 2011.

[8] Sh.M. Farahani, A.A. Abshouri, B. Nasiri, M.R. Meybodi, Some hybrid models to improve firefly algorithm performance, *International Journal of Artificial Intelligence*, vol. 8(S12), pp. 97–117, 2012.

[9] I. Fister, I. Fister Jr., X.-S. Yang, J. Brest, A comprehensive review of firefly algorithms, *Swarm and Evolutionary Computation*, vol. 13, pp. 34–46, 2013.

[10] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Mixed variable structural optimization using firefly algorithm, *Computers and Structures*, vol. 89(23–24) (2011), pp. 2325–2336, 2011.

[11] L. Guo, G.-G. Wang, H. Wang, D. Wang, An effective hybrid firefly algorithm with harmony search for global numerical optimization, *The Scientific World Journal*, Volume 2013, Article ID 125625, 9 pages, 2013.

[12] A. R. Jordehi, J. Jasni, Parameter selection in particle swarm optimisation: a survey, *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 25(4), pp. 527–542, 2013.

[13] M.H. Kashan, N. Nahavandi, A.H. Kashan, DisABC: A new artificial bee colony algorithm for binary optimization, *Applied Soft Computing*, vol. 12(1), pp. 342–352, 2012.

[14] M.H. Kashan, A.H. Kashan, N. Nahavandi, A novel differential evolution algorithm for binary optimization, *Computational Optimization and Applications*, vol. 55(2), pp. 481–513, 2013.

[15] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm optimiser, in *Proceedings of IEEE International Conference on Computational Cybernetics and Simulation*, vol. 5, pp. 4104–4108, 1997.

[16] A.N. Kumbharana, G.M. Pandey, Solving travelling salesman problem using firefly algorithm, *International Journal for Research in Science & Advanced Technologies*, Vol. 2(2), pp. 53–57, 2013.

[17] X. Lin, Y. Zhong, H. Zhang, An enhanced firefly algorithm for function optimisation problems, *International Journal of Modelling, Identification and Control*, vol. 18(2), pp. 166–173, 2013.

[18] T. Liu, L. Zhang, J. Zhang, Study of binary artificial bee colony algorithm based on particle swarm optimization, *Journal of Computational Information Systems*, Vol. 9(16), pp. 6459–6466, 2013.

[19] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems, *Applied Mathematics and Computation*, vol. 188, pp. 1567–1579, 2007.

[20] A. Manju, M.J. Nigam, Firefly algorihtm with fireflies having quantum behavior, in *Institute of Electrical and Electronics Engineers*, 2012 ICRCC, India, December 2012, pp. 117–119, 2012.

[21] S. Mirjalili, A. Lewis, S-shaped versus V-shaped transfer functions for binary particle swarm optimization. *Swarm & Evolutionary Computation*, vol. 9, pp. 1–14, 2013.

[22] M. Padberg, Harmony search algorithms for binary optimization problems, in *Operations Research Proceedings 2011*, pp. 343–348, 2012.

[23] G. Pampará, A.P. Engelbrecht, N. Franken, Binary differential evolution, in *Proceedings of 2006 IEEE Congress on Evolutionary Computation*, pp. 1873–1879, 2006.

[24] G. Pampará, A.P. Engelbrecht, Binary artificial bee colony optimization. In *IEEE Symposium on Swarm Intelligence*, IEEE Perth, pp. 1–8, 2011.

[25] M.K. Sayadi, A. Hafezalkotob, S.G.J. Naini, Firefly-inspired algorithm for discrete optimization problems: an application to manufacturing cell formation, *Journal of Manufacturing Systems*, vol. 32(1), pp. 78–84, 2013.

[26] M. Sevkli and A. R. Guner, A continuous particle swarm optimization algorithm for uncapacitated facility location problem, in: M. Dorigo et al. (Eds.), ANTS 2006, *Lecture Notes in Computer Sciences*, Vol. 4150, Springer-Verlag, pp. 316–323, 2006.

[27] S.L. Tilahun and H.C. Ong, Modified firefly algorithm, *Journal of Applied Mathematics*, Volume 2012, Article ID 467631, 12 pages, 2012.

[28] L. Wang, C. Singh, Unit commitment considering generator outages through a mixed-integer particle swarm optimization algorithm, *Applied Soft Computing*, vol. 9, pp. 947–953, 2009.

[29] L. Wang, R. Yang, Y. Xu, Q. Niu, P.M. Pardalos, M. Fei, An improved adaptive binary harmony search algorithm, *Information Sciences*, vol. 232, pp. 58–87, 2013.

[30] X.-S. Yang, Firefly algorithms for multimodal optimization, in O. Watanabe, T. Zeugmann (Eds.) Stochastic Algorithms: Foundations and Applications (SAGA 2009) *Lecture Notes in Computer Sciences*, Vol. 5792, pp. 169–178, 2009.

[31] X.-S. Yang, Firefly algorithm, stochastic test functions and design optimization, *International Journal of Bio-Inspired Computation*, vol. 2(2), pp. 78–84, 2010.

[32] X.-S. Yang, Firefly algorithm, in *Nature-Inspired Metaheuristic Algorithms*, 2nd edition, pp. 81–96, Luniver Press, University of Cambridge, UK, 2010.

[33] X.-S. Yang, Multiobjective firefly algorithm for continuous optimization, *Engineering with Computers*, vol. 29(2), pp. 175–184, 2013.

[34] X.-S. Yang, X. He, Firefly algorithm: recent advances and applications, *International Journal of Swarm Intelligence*, vol. 1(1), pp. 36–50, 2013.

[35] X.-S. Yang, S.S.S. Hosseini, A.H. Gandomi, Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect, *Applied Soft Computing*, vol. 12(3), pp. 1180–1186, 2012.

[36] S. Yu, S. Yang, S. Su, Self-adaptive step firefly algorithm, *Journal of Applied Mathematics*, Volume 2013, Article ID 832718, 8 pages, 2013.