INTELLIGENT AUTOMATION & SOFT COMPUTING, 2017
http://dx.doi.org/10.1080/10798587.2017.1294811

Check for updates

# Comparative Study of Prey Predator Algorithm and Firefly Algorithm

Hong Choon Ong[a], Surafel Luleseged Tilahun[b], Wai Soon Lee[a] and Jean Meadard T. Ngnotchouye[b]

[a]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Minden, Penang, Malaysia; [b]School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, Private Bag X01, Scottsville, Pietermaritzburg, South Africa

**ABSTRACT**

Metaheuristic algorithms are found to be promising for difficult and high dimensional problems. Most of these algorithms are inspired by different natural phenomena. Currently, there are hundreds of these metaheuristic algorithms introduced and used. The introduction of new algorithm has been one of the issues researchers focused in the past fifteen years. However, there is a critic that some of the new algorithms are not in fact new in terms of their search behavior. Hence, a comparative study in between existing algorithms to highlight their differences and similarity needs to be studied. Apart from knowing the similarity and difference in search mechanisms of these algorithms it will also help to set criteria on when to use these algorithms. In this paper a comparative study of prey predator algorithm and firefly algorithm will be discussed. The discussion will also be supported by simulation results on selected twenty benchmark problems with different properties. A statistical analysis called Mann—Whitney U 2 test is used to compare the algorithms. The theoretical as well as simulation results support that prey predator algorithm is a more generalized search algorithm, whereas firefly algorithm falls as a special case of prey predator algorithm by fixing some of the parameters of prey predator algorithm to certain values.

**KEYWORDS**

Prey predator algorithm; firefly algorithm; comparative study; metaheuristics; swarm intelligence

## 1. Introduction

Many problems in our daily activities involve decision-making or optimization. Identifying the shortest route from a city A to another city B and on how to minimize the electricity consumption for a house, company, a building or even a city are good examples. Optimization is a process of finding a solution to a problem, which satisfies certain limitations also called constraints, and minimizes or maximizes one or more objectives. To solve this kind of problem, many deterministic methods have been proposed including integer programming (Wolsey, 1998), simplex method (Gass, 1985) and dynamic programming (Bellman, 2003). Owing to the complexity of some class of problems, finding an exact solution is not always an easy task. This in turn gives an idea to the researchers to solve the problems heuristically.

Metaheuristic algorithms are algorithms that involve a search process based on educated guess and an incorporated stochastic search component. Most of these algorithms works based on randomly or pseudo randomly generated solutions. In order to improve the accuracy of the results and reduce the computation time of the algorithm, many metaheuristic algorithms are developed. Generally, there are three main inspirations that influence the development of algorithms for problem-solving on digital computers: (i) the human brain (Russel & Norvik, 2010), (ii) the Darwinian evolution (Ahn, 2006), and (iii) the social behavior of insects and other animals (Beekman, Sword, & Simpson, 2008). The first source has led to the emergence of artificial intelligence (AI), the second to evolutionary computation (EC), and the third to swarm intelligence (SI).

Different metaheuristic algorithms have been proposed in the past twenty years. Introducing a new algorithm (Cheng & Prayogo, 2014; Tilahun, Kassa, & Ong, 2012; Yang, 2010; Yang & Hossein Gandomi, 2012), modifying or hybridizing of existing algorithms (Grosan, Abraham, & Ishibuchi, 2007; Farook & Raju, 2013; Kao & Zahara, 2008; Talbi & Belarbi, 2011; Tilahun & Ong, 2013) and performing analysis of these algorithms (Rudolph, 1994; Jin, Yang, & Li, 2012; Pal, Jain, & Pandit, 2011; Qinghai, 2010) have been the central research issues along with applications (Hamadneh, Sathasivam, Tilahun, & Ong, 2012; Ong & Tilahun, 2011, Tilahun & Ong, 2012b). Even though different algorithms are proposed and successfully used in solving problems, there is a critic from the scientific community that some of the algorithms resemble each other and only the scenario they are mimicking is different (Sörensen, 2013). Hence, studies need to be conducted in this direction on the similarity and also unique search behavior of existing algorithms, so that new algorithms will be evaluated based on their search behavior, but not on what they are inspired of. A detailed analysis needs to be done on the search component level when introducing a new algorithm and showing its unique quality or if it is a general case for existing algorithms. It should not be acceptable, if a paper fails to address this issue while introducing a new algorithm.

Prey predator algorithm is a metaheuristic algorithm, which is inspired by the interaction between a predator and its prey (Tilahun & Ong, 2015). It is a swarm based algorithm with some similar search behaviors with another metaheuristic algorithm introduced earlier, called firefly algorithm (Yang, 2009), which later extended to a more general version called modified firefly algorithm (Tilahun & Ong, 2012a). In this paper a comparative study of these two swarm based algorithms will be done to access the relation between the two algorithms. The

**CONTACT** Surafel Luleseged Tilahun surafelaau@yahoo.com

© 2017 TSI® Press

comparison is done based on detailed theoretical level, based on the search components of the algorithm as well as based on simulation. In the theoretical comparison study the asymptotic behavior of the algorithms per their parametric values will be analyzed. In addition to that twenty benchmark problems of dimension varying from two to twenty are used to test the two algorithms. The algorithm parameters are tuned based on recommendations previous studies. A non-parametric statistical test, namely Mann-Whitney U 2 test (Sheskin, 2004), will be used to analyze the simulation result.

In the Section 2, basic concepts including optimization problems and introduction to the two algorithms will be discussed followed by comparison study in Section 3. Section 4 presents simulation results followed by results and discussion in Section 5. In Section 6 a conclusion will be presented.

## 2. Preliminaries

### 2.1. Optimization Problem

An optimization problem has three components: The objective function to optimize, a rule, which is either to maximize or to minimize the objective function, and a feasible region or set of possible actions to choose from. The aim is to optimize (minimize or maximize) the objective function by choosing a value for the variable from the feasible region. The decision variables are the variables whose values are under control and influence the performance of the system. Constraints are the restrictions on the values of decision variables. A standard minimization problem can be described as:

$$Min\ z(x)$$

$$\text{s.t.x}\ \in S \subseteq R^n$$

A solution $x^*$ is the optimal solution if and only if $x^* \in S$ and $z(x) \geq z(x^*)$, for all $x$ in $S$.

### 2.2. Firefly Algorithm

Firefly algorithm is a metaheuristic algorithm for optimization problem inspired by the flashing behavior of fireflies. Randomly generated solutions are considered as fireflies. To update these solutions three rules are used as a construction block of the algorithm. These rules are;

i) All fireflies are unisex, so that one firefly can be attracted to any other firefly regardless of its "sex",
ii) Attractiveness is directly proportional to brightness but inversely to distance. A firefly will move towards a brighter firefly and if there is no brighter firefly to follow then it will move randomly,
iii) The brightness of a firefly is determined based on its performance in the objective function.

The attractiveness of a firefly can be calculated as follows:

$$A(r) = \frac{A_0}{e^{\lambda r^2}} \tag{1}$$

Where $A_0$ is the attractiveness at $r = 0$, $\lambda$ is the coefficient of light absorption of the medium and $r$ is the distance from source or distance of a two fireflies.

If there are two fireflies, firefly $i$ and firefly $j$ at $x_i$ & $x_j$ and suppose firefly $j$ is brighter than firefly $i$, then firefly $i$ will be attracted by firefly $j$ (and it will move towards $j$). The updating procedure for firefly $i$ is given by:

**Table 1.** Modified Firefly Algorithm (It will be the standard firefly algorithm if the brightest firefly is updated using random neighborhood search).

```
Algorithm parameter setup
Generate a set of random solution, {x_1, x_2, ..., x_N}
For Iteration=1:MaximumIteration
  Calculate the intensity for each solution, {I_1, I_2, ..., I_N} and without losing
    generality sort them in brightness from x_1 (dimmer) to x_N (brightest)
  For i=1:N-1
    For j=i+1:N
      Move firefly i towards firefly j using Eq. (2)
    End
  End
  Move the brightest firefly, x_N, in a promising direction using Eq. (3)
End
Return the best result
```

$$x_i \leftarrow x_i + A_0 e^{-\lambda r^2}\left(x_j - x_i\right) + \alpha\varepsilon \tag{2}$$

Where α is a step length and $\varepsilon$ is a random vector. Yang (2010) suggested to assign $A_0 = 1$ for practical usage.

Since the introduction of firefly algorithm, different researches are conducted to improve its performance and to use firefly algorithm on different applications. Currently, there are many variants of firefly algorithm. One of these versions, which shows a promising results is modified firefly algorithm (Tilahun & Ong, 2012a). In the standard firefly algorithm, the brightness of the brightest firefly may decrease through iteration. This in turn gives the algorithm a performance, which is not always improving. This is due to the random movement of the brightest firefly. To avoid this scenario, in the modified firefly algorithm, a modification is proposed by adding a special movement for the brightest firefly, as:

$$x_i \leftarrow x_i + \alpha U \tag{3}$$

Where $U$ is the direction chosen among the randomly generated $m$ unit vectors, which can let the brightness of the brightest firefly to increase if the firefly moves in that direction and α is a step length. If such direction (a direction, which will improve the brightness) does not exist among the randomly generated directions then the brightest firefly will remain in the current position.

Furthermore, the suggestion given in (Yang, 2010), which says "*for practical use, $A_0$, was recommended to be set as 1*," should be replaced by an expression involving the attractiveness or brightness of the fireflies like $e^{I_0' - I_0}$ where $I_0'$ is the intensity at $r = 0$ for firefly $j$ and $I_0$ is the intensity at $r = 0$ for firefly $i$. This will allow the attractiveness of the fireflies to be dependent on the intensity, which in turn depends on the objective function.

The Modified Firefly algorithm can be summarized in Table 1.

### 2.3. Prey-Predator Algorithm

Prey-Predator algorithm is a new metaheuristic algorithm developed by Tilahun and Ong (Tilahun, 2013; Tilahun & Ong, 2015) for handling a complex optimization problem. It has been compared with other algorithms and used in different applications (Hamadneh et. al., 2013; Tilahun & Ong, 2014; Bahmani-Firouzi, Sharifinia, Azizipanah-Abarghooee, & Niknam, 2015; Dai, Liu, & Chai, 2015; Tilahun & Ngnotchouye, 2016; Tilahun, Ong, & Ngnotchouye, 2016). With its success to continuous problems, it has also been modified and used for discrete problems (Tilahun, Goshu, & Ngnotchouye, 2017). It is inspired by the interaction between a carnivorous predator and its prey. The algorithm mimics the ways in which a

predator runs after and hunt its prey where each prey tries to stay within the pack, search for a hiding place, and also run away from the predator.

In the algorithm, a set of initial feasible solutions will be generated and for each solution, xi, a numerical value to show its performance in the objective function called survival value ($SV(x_i)$) is assigned. Better performance in the objective function implies higher survival value. This means for solutions $x_i$ and $x_j$, if $x_i$ performs better than $x_j$ in the objective function, $SV(x_i) > SV(x_j)$. A solution with the smallest survival value will be assigned as predator, $x_{predator}$, and the rest as prey. Among these prey, a prey, say $x_b$, where $SV(x_b) \geq SV(x_i)$, for all $i$, is called the best prey. This means, the best prey is a prey with the highest survival value among the solutions.

Once the prey and the predator are assigned, each prey need to escape from the predator and try to follow other prey with better survival values or find a hiding place. While, the predator hunts the weak prey and scare the others, which contributes to the exploration of the solution space. Exploitation is carried out by the prey, especially the best prey, by using a local search. The best prey is considered as the one who has found a secure place and is safe from the predator. Thus it will only focus on conducting a local search to improve its survival value. However, the other prey will follow the prey population with better surviving values and run away from the predator.

In the updating process of these solutions there are two issues to deal with, the direction and the step length.

(a) Updating direction

In the algorithm, the movement of an ordinary prey (not the best prey) depends on an algorithm parameter called the probability of follow up ($P_f$). If the probability of follow up is greater than or equal to a randomly generated number between zero and one from a uniform distribution, then the prey will follow other prey with better survival value and also does a local search; otherwise it will randomly run away from the predator.

Suppose the probability of follow up larger or equal to a random number and there are prey with better survival value compared to $x_i$, say $x_1$, $x_2$,..., $x_s$. In reality, mostly a group of prey tend to stay in pack and hence it try to be with the nearest pack of prey animals therefore the movement of $x_i$ is highly dependent on the distance between itself and better prey. Therefore, the direction of the movement of $x_i$ can be calculated as follows:

$$y_i = \sum_j e^{SV(x_j)^v - r_{ij}}(x_j - x_i) \tag{4}$$

Where $r_{ij}=||x_i-x_j||$ is a distance between $x_i$ and $x_j$, and $v$ is an algorithm parameter, which plays the role of magnify or diminish the effect of the survival value over the distance. By assigning different values of $v$, it is possible to adjust dependency of the updating direction on the survival value and the distance. If $v$ is too big then the direction favors the performance than the distance of the better prey from the solution, this means the prey try to catch up with the best pack with little consideration how far that pack is. If $v$ is too small, it implies that the prey prefer to follow the nearest better pack. Assigning a large or a small value for $v$ will affect the jump size of $x_i$. Hence, a unit direction will be used to represent the direction as:

$$u_i = \frac{y_i}{||y_i||} \tag{5}$$

Moreover, a local search is done by generating $q$ random directions and taking the best direction, say $y_l$, among these $q$ directions. In order to choose $y_l$ the survival value will be checked if the solution moves in the $q$ directions and the direction, which results the highest survival value will be taken.

If the probability of follow up is smaller than a random number, then the prey will randomly run way from the predator. This is done by generating a random direction $y_{rand}$ and comparing the distance between the predator and the prey if it moves in $y_{rand}$ or $-y_{rand}$, and the direction, which will be take the prey away the predator will be taken.

Unlike the other ordinary prey, the movement of the best prey will perform only to do a local search. It will only move toward the direction, which can improve its survival value, from a randomly generated $q$ directions or stays in its current position if no such direction exist among these $q$ unit directions.

In the algorithm, the main task of the predator is to motivate the prey for exploring the solution space and it also does the exploration of the solution spaces. Thus, it will chase after the prey with the least survival value and also moves randomly in the solution space. Hence the movement direction will have two components; random direction as well as towards the weak prey, $x'$.

(b) Step length

Step length is the other issue related to updating of solutions. In the carnivorous predation, a prey, which is nearer to the predator running faster than the other prey? A similar concept is mimicked in PPA, in such a way that a prey with small survival value runs faster than larger survival valued prey. Therefore, a prey has a step length $\lambda$ that is inversely proportional to its survival value. This can be expressed as the following formulation:

$$\lambda = \frac{\lambda_{max} rand}{|SV(x_i) - SV(x_{predator})|^w} \tag{6}$$

Where *rand* is a random number from a uniform distribution, $0 \leq rand \leq 1$ and $\lambda_{max}$ represents the size of the maximum jump of a prey. Parameter $w$ influences the dependency of the step length on the relative survival value. However, for the practical purpose it is also possible to omit the effect of the survival value in assigning the maximum jump, which eliminates the further study of changes in survival values to choose the parameter $w$ (Tilahun & Ong, 2014; Tilahun & Ngnotchouye, 2016; Tilahun et. al. 2016). Hence, we can simply have;

$$\lambda = \lambda_{max} rand \tag{7}$$

Furthermore, another step length is a step length for local search or exploitation purpose and it is denoted by $\lambda_{min}$. The step length for the exploitation purpose should be smaller than the exploration step length, i.e. $\lambda_{max} > \lambda_{min}$.

Summarizing all the points discussed; the movement of the ordinary prey, the best prey and the predator can be summarized as follows:

Movement of an ordinary prey:

i) If the probability of follow up is larger or equal to a random number,

$$x_i \leftarrow x_i + \left(\frac{y_i}{||y_i||}\right)\lambda_{max} rand + \left(\frac{y_l}{||y_l||}\right)\lambda_{min} rand \tag{8}$$

**Table 2.** Prey Predator Algorithm.

```
Algorithm parameter setup
Generate a set of random solution, {x1, x2, . . ., xN}
For Iteration=1:MaximumIteration
  Calculate the survival value for each solution, {SV1, SV2, . . ., SVN} and without
    losing generality sort them in such a way that SVi ≤ SVi+1 for all i
  Update the predator x1 using Eq. (11)
  For i=2:N-1
    If Pf≤ rand
      For j=i+1:N
        Move solution i towards solution j using Eq. (8)
      End
    Else
      Move solution j using Eq.(9)
    End
  End
  Move the best solution, xN, in a promising direction using Eq. (10)
End
Return the best result
```

ii) If the probability of follow up is smaller than a random number,

$$x_i \leftarrow x_i + \left(\frac{y_r}{||y_r||}\right)\lambda_{max} rand \tag{9}$$

Movement of the best prey:

$$x_b \leftarrow x_b + \left(\frac{y_l}{||y_l||}\right)\lambda_{min} rand \tag{10}$$

Movement of predator:

$$\begin{aligned} x_{predator} \leftarrow x_{predator} &+ \left(\frac{y_r}{||y_r||}\right)\lambda_{max} rand \\ &+ \left(\frac{x_i' - x_{predator}}{\left|\left|x_i' - x_{predator}\right|\right|}\right)\lambda_{min} rand \end{aligned} \tag{11}$$

The Prey-Predator algorithm steps are summarized in Table 2.

## 3. Comparison of Firefly Algorithm and Prey Predator Algorithm

The updating process for an ordinary firefly, excluding the brightest one, in the standard firefly algorithm is the same as in the modified firefly algorithm (Tilahun & Ong, 2012a). Suppose a firefly $x_i$ is attracted by another firefly $x_j$. Then $x_i$ will update its location as follows:

$$x_i \leftarrow x_i + \beta\left(x_j - x_i\right) + \alpha\varepsilon \tag{12}$$

Where $\alpha$ is an algorithm parameter to control the step length of a random direction $\varepsilon$, $\beta$ is the brightness of the firefly $j$ at the location of firefly $i$ and can be computed by $\beta = \beta_0 e^{-\gamma r_{ij}^M}$ for $M \geq 1$ and $\beta_0$ is the brightness of the firefly $j$ at its current position and $r_{ij}$ is the distance between the two fireflies. In most cases $M = 2$, but it should be noted that $M$ can be taken as any value greater or equal to one (Yang, 2010). This results show that the brightness will be inversely proportional to the distance (Yang, 2010).

Now consider the updating process of an ordinary prey, a prey, which is not the best prey, in prey predator algorithm. Suppose the survival value of prey $j$ is better than prey $i$. Then $x_i$ will move towards $x_j$, if the probability of follow up is met, according to:

$$x_i \leftarrow x_i + \frac{e^{SV(x_j)^v}}{e^{r_{ij}}}\lambda_{max} r\,and\left(x_j - x_i\right) + u_l\lambda_{min} r\,and \tag{13}$$

Where *rand* is a random number between 0 and 1 from a uniform distribution, $u_l$ is a vector chosen from $q$ randomly generated unit vectors for local search, $v$ is a parameter to balance the influence of the distance and survival value ($SV$) in the updating process, $\lambda_{max}$and $\lambda_{min}$ are step lengths for the follow up and for the local search, respectively. However, if the probability of follow up is not met, which means if a randomly generated number is greater than probability of follow-up, then $x_i$ will not follow $x_j$, but moves randomly away from the worst solution.

If the probability of follow up is set to be one in PPA, then any solution will follow the better solutions. Suppose that is the case and let us look at Eqs. (12) and (13). The last term of Eq. (12) tells us that the solution will move in a random direction $\varepsilon$ with a step length $\alpha$, and in the last term of Eq. (13), $u_l$ is a unit vector chosen from $q$ random unit vectors and $\lambda_{min}$ is the step length. Hence if $q$ is set to be 1 and since $u_l$ *rand* is a random direction, then the last term of Eq. (13) is a random move with a step length $\lambda_{min}$. The second term of the equations is to follow up for better solutions and is serving the same purpose in the both algorithms.

There is an expression for the term to be dependent on the distance between the solutions, which is in $e^{-r_{ij}^M}$ form, and with setting $\lambda_{max}$=1 the brightness at the initial point of the solution can be computed using $\beta = e^{SV^v}$. The only difference between the two terms will be the expression of *rand* in Eq. (9), but the solutions will behave in similar fashion. Hence, with the probability of follow up being 1, $\lambda_{max}$ being 1 and $q = 1$ an ordinary prey will update itself in a similar way as an ordinary firefly. Therefore, PPA will resemble MFFA and also FFA in terms of ordinary solutions under the tuning of the parameters as described.

Now consider the best solutions in MFFA and PPA. In the case of standard FFA, the brightest firefly will move randomly. However in the case of PPA and MFFA, $q$ unit directions will be generated and a direction will be chosen if it improves the performance of the solution. If such direction doesn't exist among the $q$ random unit directions, the solution will remain in its current position.

In addition to these, the predator in PPA can be considered as a solution, which will follow only the immediate better prey, which is a prey with least performance. However, this role can be omitted in the analysis as FFA and MFFA do not take the least performing solution into account.

Hence, based on the parametric discussion given, PPA will perform in similar fashion as the MFFA when the probability of follow up is one, $\lambda_{max}$ being one, $q = 1$ for the ordinary solution and number of predator is zero. In addition if the number of best prey is zero, PPA will work a similar search behavior with the standard firefly algorithm.

A recently extension of prey predator algorithm based on the group hunting scenario is introduced (Tilahun et. al., 2012). It is the same algorithm, but multiple numbers of predators and best prey. It is named as nm-PPA, where $n$ is the number of predators and $m$ being the number of best prey. Hence, in addition to the discussion and analysis given above the standard firefly algorithm will resemble 00-PPA, for n = 0 and m = 0.

**Table 3.** Benchmark Problems and their Properties According to (Jamil & Yang, 2013).

| Test Function | Function Name | Properties of Function |
|---|---|---|
| 1 | Bird Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 2 | Camel Function- Six Hump | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 3 | Deckkers–Aarts Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 4 | Giunta Function | Continuous, Differentiable, Separable, Scalable, Multimodal |
| 5 | Jennrich-Sampson Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 6 | Mishra 3 Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 7 | Mishra 6 Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 8 | Quadratic Function | Continuous, Differentiable, Non-Separable, Non-Scalable |
| 9 | Bartels Conn Function | Continuous, Non-differentiable, Non-Separable, Non-Scalable, Multimodal |
| 10 | Ackley 2 Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Unimodal |
| 11 | Miele Cantrell Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 12 | Easom Function | Continuous, Differentiable, Separable, Non-Scalable, Multimodal |
| 13 | Egg Crate Function | Continuous, Separable, Non- Scalable |
| 14 | Price 2 Function | Continuous, Differentiable, Non-Separable, Non-Scalable, Multimodal |
| 15 | Stepint Function | Discontinuous, Non-Differentiable, Separable, Unimodal |
| 16 | Salomon Function | Continuous, Differentiable, Non-Separable, Scalable, Multimodal |
| 17 | Sargan Function | Continuous, Differentiable, Non-Separable, Scalable, Multimodal |
| 18 | Trigonometric 1 Function | Continuous, Differentiable, Non-Separable, Scalable, Multimodal |
| 19 | Xin-She Yang (Function 1) | Separable, non-differentiable, Scalable, Stochastic |
| 20 | Zakharov Function | Continuous, Differentiable, Non-Separable, Scalable, Multimodal |

## 4. Simulation

Simulation is also done to check and validate the assumption made in a earlier section, which states that the PPA will perform in a similar fashion as the MFFA when number of predator is zero, the probability of follow up is one, exploration step length being one and $q$ is one for the ordinary solution. It is mentioned that MFFA works in a similar search behavior with standard FFA if in addition the number of best prey is zero (Tilahun & Ong, 2012a; Tilahun, 2013), hence the simulation is done only to compare PPA with MFFA. Twenty benchmark problems of different properties are selected for simulation (Jamil & Yang, 2013). These test functions include multimodal, unimodal, continuous, discontinuous and stochastic problems. For each test problem, the same randomly generated initial solution set is used for both algorithms and the same number of iterations for both algorithms is used. A statistical hypothesis testing is then conducted based on the result generated by both algorithms.

**Table 4.** Parameter Setting.

| Benchmark Function | Feasible region | Problem dimension | ($\lambda_{min}$) |
|---|---|---|---|
| 1 | [-2π, 2 π] | 2 | 4π/5 |
| 2 | [-5, 5] | 2 | 2 |
| 3 | [-20, 20] | 2 | 8 |
| 4 | [-1, 1] | 2 | 0.4 |
| 5 | [-1, 1] | 2 | 0.4 |
| 6 | [-20, 20] | 2 | 8 |
| 7 | [-5, 5] | 2 | 2 |
| 8 | [-10, 10] | 2 | 4 |
| 9 | [-500, 500] | 3 | 225 |
| 10 | [-32, 32] | 4 | 15 |
| 11 | [-1, 1] | 4 | 0.5 |
| 12 | [-100, 100] | 5 | 50 |
| 13 | [-5, 5] | 6 | 4 |
| 14 | [-10, 10] | 8 | 6 |
| 15 | [-5.12, 5.12] | 10 | 3 |
| 16 | [-100, 100] | 12 | 40 |
| 17 | [-100, 100] | 14 | 50 |
| 18 | [0, π] | 15 | π/4 |
| 19 | [-5, 5] | 17 | 2.5 |
| 20 | [-5, 10] | 20 | 3.5 |

### 4.1. Benchmark Problems

For uniformity purposes, all the problems considered are minimization problems taken from Jamil and Yang (2013). Table 3 shows the list of the benchmark problems and their properties.

### 4.2. Mann-Whitney U 2 Sample Test (Also Known As Wilcoxon Rank Sum Test)

The parametric test such as 2 independent $t$-tests cannot be conducted when the assumptions relating to the level of measurement, sample size, normality or equality of variance are not valid (Ott & Longnecker, 2010). Thus, we need to use a non-parametric test. The Mann Whitney test is a non-parametric test, which is useful to study the difference between the magnitudes of the variables in two independent sets of data (Ott & Longnecker, 2010; Sheskin, 2004). The test assumes the two independent samples from two different populations and the samples have the same shape and spread, though they don't have to be symmetric. The Mann-Whitney procedure is a statistical test of the difference between the two medians. Under the null hypothesis, they are equal i.e. the median of the sample results from MFFA and the median of the sample result from PPA are equal. We wish to test the following null hypothesis at α = 0.05 (95% confidence level):

$$H_0: Med_{MFFA} = Med_{PPA}$$

$$H_1: Med_{MFFA} \neq Med_{PPA} \quad (14)$$

The steps for the calculation of the rank sum statistics are as follow:

i. Arrange all the data values for both samples from the smallest to the largest.
ii. Assign number 1 to the smallest data value and $G$ (total number of observations from both samples) to the largest. These assigned values are the ranks of the observations.
iii. If there exists a tie observation, the ranks for the observation is the average rank for the observations.
iv. Let $T$ denote the sum of ranks for the observation from population 1 (MFFA)Under the null hypothesis, the sampling distribution of $T$ has mean and variance given by:

$$\mu_T = \frac{n_1(n_1 + n_2 + 1)}{2} \quad (15)$$

$$\sigma_T^2 = \frac{n_1 n_2 (n_1 + n_2 + 1)}{12}$$

Where $n_1$ is the number of observations from population 1 and $n_2$ is the number of observation from population 2. If $T$ is smaller or larger than $\mu_T$, we have evidence that the null hypothesis is not acceptable under the given confidence interval. In this paper, we use statistical software called MINITAB 16 to carry out the hypothesis testing.

## 5. Results and Discussions

The simulation is done using MATLAB R2011b on Intel® Core™ i5-2,400 CPU @ 3.10 GHz with 32 bit operating system desktop machine. As mentioned earlier, the initial solutions for both algorithms for each of the problems were taken the same random solutions. The algorithm runs with the algorithm parameters fixed based on the discussion in Section 3 and recommendations from literature (Tilahun, 2013; Tilahun & Ong, 2015; Yang, 2010). This means, for PPA the probability of follow up is one, $q$ = 1 and $\lambda_{max}$=1, $v$ = 0, number of predator = 0 and number of best prey = 0. Furthermore, $\alpha$ in MFFA is set to be equal to $\lambda_{min}$ in PPA and given in Table 4.

The termination criterion was set to be maximum number of iterations, which is set to be 30. The experiment was run 30 times with average final results as presented in Table 5.

**Table 5.** Simulation Results.

| Benchmark Functions | Exact optimum solution | Optimum solution generated by MFFA | Optimum solution generated by PPA |
|---|---|---|---|
| 1 | −106.761537 | −106.787718 | −106.787730 |
| 2 | −1.031628 | −1.031617 | −1.031629 |
| 3 | -24777.000000 | -24776.073041 | -24776.518212 |
| 4 | 0.064470 | 0.064471 | 0.064470 |
| 5 | 124.361200 | 124.363321 | 124.362192 |
| 6 | −0.184670 | −0.184780 | −0.184699 |
| 7 | −2.283950 | −2.283946 | −2.283950 |
| 8 | -3873.724300 | -3873.724141 | -3873.724182 |
| 9 | 1.000000 | 1.000091 | 1.000011 |
| 10 | −200.000000 | −199.998555 | −199.999837 |
| 11 | 0.000000 | 0.000000 | 0.000000 |
| 12 | −1.000000 | −0.999970 | −1.000000 |
| 13 | 0.000000 | 0.000019 | 0.000001 |
| 14 | 0.900000 | 0.900000 | 0.900000 |
| 15 | 0.000000 | 0.000000 | 0.000000 |
| 16 | 0.000000 | 0.000019 | 0.000014 |
| 17 | 0.000000 | 0.000000 | 0.000000 |
| 18 | 0.000000 | 0.000018 | 0.000002 |
| 19 | 0.000000 | 0.000000 | 0.000000 |
| 20 | 0.000000 | 0.000001 | 0.000000 |

To check the normality assumption for the data obtained for simulation, Minitab is used and the generated result is shown in Figure 1. From Figure 1, it is clear to see that that probability plot failed to adequately fit the data obtained for MFFA and PPA. Furthermore, the Anderson-Darling test (Dodson, 2006) shown with p-values is less than 0.005, which indicate that the hypothesis, which is "the data is normal" is rejected at 95% of confidence level for both MFFA and PPA. Hence, Mann-Whitney test is suitable for this data.

From Figure 2, the statistical software (Minitab) output provides P-Value is 1.00 for the Mann-Whitney test. Thus at 95% confidence level, the null hypothesis stated in Section 4.2, Mann-Whitney U 2 Sample Test ($H_0$:$Med_{MFFA} = Med_{PPA}$), is accepted with the given confidence level. This means that under the given confidence level, the claim, which says the median of the two experiments are equal is accepted. Hence, there is no significant difference in between the median results based on the functional values of the two algorithms. Hence, the simulated result supports the claim that the performance of MFFA and PPA under the given parametric adjustment is the same.

The simulation result along with the statistical analysis shows that there is no significant difference on the performance of the modified firefly algorithm and prey predator algorithm

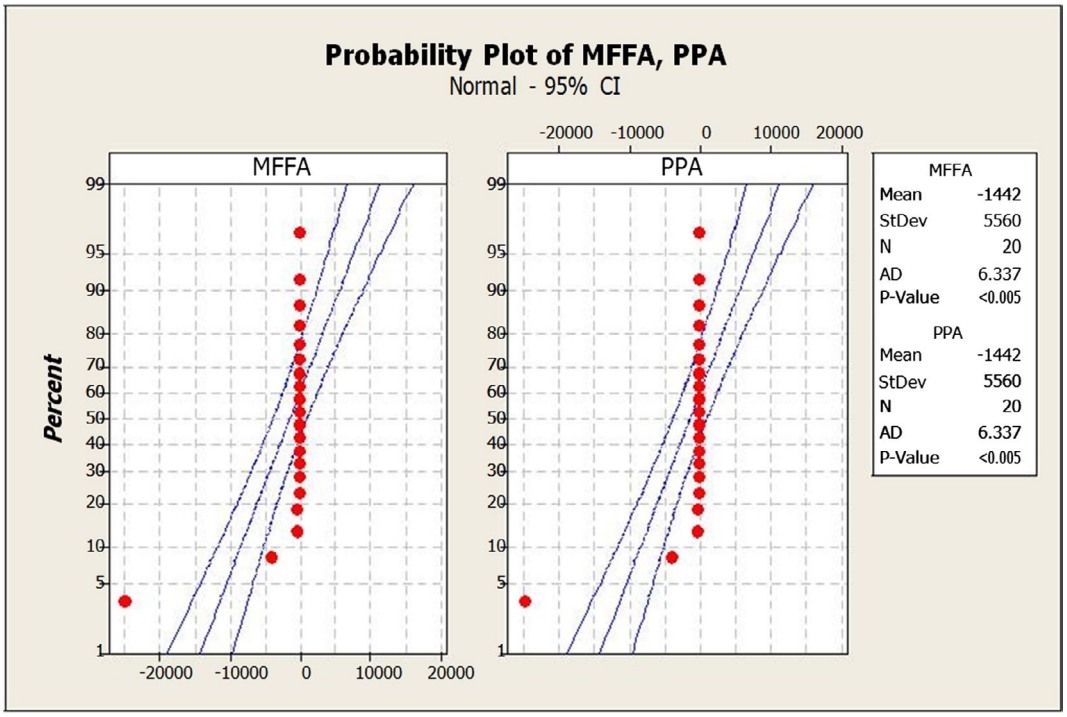



**Figure 1.** Statistical Software (Minitab) Output: Normality Test.

```
Mann-Whitney Test and CI: MFFA, PPA

       N  Median
MFFA  20     0.0
PPA   20     0.0


Point estimate for ETA1-ETA2 is -0.0
95.0 Percent CI for ETA1-ETA2 is (-1.0,0.9)
W = 410.5
Test of ETA1 = ETA2 vs ETA1 not = ETA2 is significant at 1.0000
The test is significant at 1.0000 (adjusted for ties)
```

**Figure 2.** Statistical Software (Minitab) Output: Normality Test: Mann-Whitney Test.

under the given simulation setup and test bed used. This in turn supports the claim in the previous section that MFFA is a special case of PPA. Indeed if the number of predator is zero, the probability of follow up is one and the exploration step length is one along with $q$, which is number of random directions for local search, then PPA become MFFA. In addition, since the standard firefly algorithm is a special case for the modified firefly algorithm, the standard firefly algorithm is also a special case of PPA. That is, in addition to the above conditions, if the number of best prey is set to be zero then PPA will become FFA.

## 6. Conclusion

To satisfy the needs of the experts or scientists in handling complex mathematical optimization problems, various types of algorithms have been developed including firefly algorithm and prey predator algorithm. Each of these algorithms has its own pros and cons. In addition to that, due to a big number of new algorithms, there is a critic that some of the algorithms are not actually contributing to the scientific society as some of the so called new algorithms are in fact known algorithms presented with different names with mimicking different natural scenarios. In this paper a recently introduced prey predator algorithm is compared with the firefly algorithm. Prey predator algorithm becomes a modified firefly algorithm when the number of predators is zero, the probability of follow up, exploration step length and the number of random directions for local search is set to be one. In addition, if the number of best prey is set be one then both prey predator algorithm and modified firefly algorithm will become standard firefly algorithms. A simulation analysis is also done based on twenty well known benchmark problems with a different property as well as dimension varying from two to twenty. A hypothesis testing using Mann—Whitney U2 test shows that there is no significant performance difference between MFFA and PPA under 95% confidence interval. Hence, prey predator algorithm is a general case whereas firefly algorithm falls as a special case.

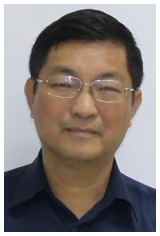
## Disclosure statement

No potential conflict of interest was reported by the authors.

## Funding

This work is supported in part by Universiti Sains Malaysia (U.S.M) Fundamental Research Grant Scheme (F.R.G.S) No. 203/PMATHS/6711319.

## Notes on contributors



***Hong Choon Ong*** received a B.Sc. degree from Universiti Malaya (UM), in 1980 and his M.Sc. and Ph.D. degrees in Mathematics from Universiti Sains Malaysia (USM), Penang, Malaysia in 1996 and 2003 respectively. He is currently an associate professor and a deputy dean (research) in USM. His research interests include data mining, statistical modelling, and machine learning applications.

***Surafel Luleseged Tilahun*** is a postdoctoral researcher in the School of Mathematics, Statistics and Computer Science. He completed his Ph.D. in Computational and Applied Mathematics from Universiti Sains Malaysia in 2013. He also has an M.Sc. in Computational Science, an M.Sc. in Mathematics and a B.Sc. in Mathematics from Addis Ababa University. His research interests are applied and computational mathematics, specifically metaheuristic algorithms, numerical optimization, modeling and simulation including applications of mathematics and theoretical computer science.

***Wai Soon Lee*** received a B.Sc. degree in Applied Statistics from Universiti Sains Malaysia (USM), Penang, Malaysia, in 2013 and his M.Sc. degrees in Statistics from Universiti Sains Malasyia (USM), Penang, Malaysia in 2014. He is currently working in ON Semiconductor Senawang, Malaysia as a Process Engineer.

***Jean Meadard T. Ngnotchouye*** is a lecturer of applied Mathematics at the University of KwaZulu-Natal in South Africa since 2012. He obtained his B.Sc. and M.Sc. from the University of Yaounde 1 in Cameroon. He obtained a postgraduate diploma in Mathematical Sciences from the African Institute for Mathematical Science in 2007 and graduated with a Ph.D. from the University of KwaZulu-Natal in 2011. He has taught a great number of undergraduate and graduate courses for the past eight years. He has successfully supervised more than ten postgraduate students and has published more than 15 research papers in international journals.

## References

Ahn, C.W. (2006). *Advances in evolutionary algorithms: Theory, design and practice*. Heidelberg: Springer-Verlag.

Bahmani-Firouzi, B., Sharifinia, S., Azizipanah-Abarghooee, R., & Niknam, T. (2015). Scenario-based Optimal Bidding Strategies of GENCOs in the Incomplete Information Electricity Market Using a New Improved Prey-predator Optimization Algorithm. *IEEE Systems Journal, 9*, 1485–1495.

Beekman, M., Sword, G.A., & Simpson, S.J. (2008). Biological foundations of swarm intelligence. In C. Blum & D. Merkle (Eds.), *Swarm intelligence: Introduction and applications* (pp. 3–41). Berlin: Springer-Verlag.

Bellman, R. (2003). *Dynamic programming*. USA: Dover Publications.

Cheng, M.Y., & Prayogo, D. (2014). Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Computers & Structures, 139*, 98–112.

Dai, W., Liu, Q., & Chai, T. (2015). Particle size estimate of grinding processes using random vector functional link networks with improved robustness. *Neurocomputing, 169*, 361–372.

Dodson, B. (2006). *The Weibull analysis handbook* (2nd ed.). USA: ASQ, Quality press.

Farook, S., & Raju, P.S. (2013). Evolutionary Hybrid Genetic - Firefly Algorithm for Global Optimization. *International Journal of Computational Engineering & Management, 16*, 37–45.

Gass, S.I. (1985). *Linear Programming: Methods and Applications*. New sYork, NY: McGraw-Hill.

Grosan, C., Abraham, A., & Ishibuchi, H. (2007). *Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews*. Berlin/ Heidelberg: Springer.

Hamadneh, N., Sathasivam, S., Tilahun, S.L., & Ong, H.C. (2012). Learning Logic Programming in Radial Basis Function Network via Genetic Algorithm. *Journal of Applied Sciences, 12*, 840–847.

Hamadneh, N., Tilahn, S.L., Sathasivam, S., & Ong, H.C. (2013). Prey-predator Algorithm as a New Optimization Technique Using in Radia Basis Function Neural Networks. *Research Journal of Applied Sciences, 8*, 383–387.

Jamil, M., & Yang, X-S. (2013). A literature survey of benchmark functions for global optimization problems. *International Journal of Mathematical Modelling and Numerical Optimisation, 4*, 150–194.

Jin, C., Yang, J., & Li, F. (2012). The performance analysis of genetic algorithm based on schema. *Advances in Intelligent and Soft Computing, 168*, 501–506.

Kao, Y.T., & Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. *Applied Soft Computing, 8*, 849–857.

Ong, H.C., & Tilahun, S.L. (2011). Integration fuzzy preference in genetic algorithm to solve multiobjective optimization problems. *Far East Math Sci, 55*, 165–179.

Ott, R.L., & Longnecker, M. (2010). *An introduction to statistical methods and data analysis* (6th ed.). Belmont, CA: Cengage Learning.

Pal, H.K., Jain, K., & Pandit, M. (2011) Performance analysis of metaheuristic techniques for nonconvex economic dispatch. Proceeding of International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2011), 396–402.

Qinghai, B. (2010). Analysis of particle swarm optimization algorithm. *Computer Information Sciences, 3*, 180–184.

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms, IEEE trans. *IEEE Transactions on Neural Networks, 5*, 96–101.

Russel, S.J., & Norvik, P. (2010). *Artificial intelligence: a modern approach*. Upper Saddle River: Prentice Hall.

Sheskin, D.J. (2004). *Handbook of Parametric and Nonparametric Statistical Procedures* 3rd ed. New york, NY: Chapman & Hall/CRC.

Sörensen, K. (2013). Metaheuristics—the metaphor exposed. *International Transactions in Operational Research, 22*, 3–18. doi: 10.1111/itor.12001.

Talbi, N., & Belarbi, K. (2011). Fast hybrid PSO and tabu search approach for optimization of a fuzzy controller. *IJCSI Int. J. Comput. Sci., 8*, 215–219.

Tilahun, S.L. (2013). Prey predator algorithm: A new metaheuristic optimization approach (PhD thesis). Universiti Sains Malaysia.

Tilahun, S.L., & Ngnotchouye, J.M.T. (2016). Prey predator algorithm with adaptive step length. *International Journal of Bio-Inspired Computation, 8*, 195–204.

Tilahun, S.L., & Ong, H.C. (2012a). Modified firefly algorithm. *Journal of Applied Mathematics,* 2012, 1–12.

Tilahun, S.L., & Ong, H.C. (2012b). Fuzzy preference of multiple decision-makers in solving multi-objective optimisation problems using genetic algorithm. *Maejo International Journal of Science and Technology, 6*, 224-237.

Tilahun, S.L., & Ong, H.C. (2013). Vector optmisation using fuzzy preference in evolutionary strategy based firefly algorithm. *International Journal of Operational Research, 16*, 81–95.

Tilahun, S.L., & Ong, H.C. (2014). Comparison between Genetic Algorithm and Prey-predator Algorithm. *Malaysian Journal of Fundamental and Applied Sciences, 9*, 167-170.

Tilahun, S.L., & Ong, H.C. (2015). Prey Predator Algorithm: A new metaheuristic algorithm for optimization problems. *International Journal of Information Technology & Decision Making, 14*, 1–22.

Tilahun, S.L., Kassa, S.M., & Ong, H.C. (2012). A new algorithm for multilevel optimization problems using evolutionary strategy, inspired by natural adaptation. *PRICAI 2012: Trends Artificial Intelligence LNAI 7458,* 577–588.

Tilahun, S.L., Ong, H.C., & Ngnotchouye J.M.T. (2016) Extended prey predator algorithm with a group hunting scenario. *Advances in Operations Research, 2016*, Article ID 7325263, 14.

Tilahun, S.L., Goshu, N.N., & Ngnotchouye, J.M.T. (2017). Prey Predator Algorithm for Travelling Salesman Problem: Application on the Ethiopian Tourism Sites. In P. Vasant, & M. Kalaivanthan (Eds.), *Handbook of Research on Holistic Optimization Techniques in the Hospitality, Tourism, and Travel Industry* (pp. 402–422). Hershey, PA: IGI Global.

Wolsey, L.A. (1998). *Integer Programming*. New York, NY: John Wiley & Sons.

Yang, X.-S. (2009). Firefly algorithms for multimodal optimization, Stochastic Algorithms: Foundations and Applications. *Lecture Notes in Computer Sciences,* SAGA, *5792*, 169–178.

Yang, X.-S. (2010). *Nature inspired metaheuristic algorithms*. Frome, UK: Luniver Press.

Yang, X.-S., & Hossein Gandomi, A.H. (2012). Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations, 5*, 464–483.