Hindawi Publishing Corporation
Advances in Operations Research
Volume 2016, Article ID 7325263, 14 pages
http://dx.doi.org/10.1155/2016/7325263

Research Article

Extended Prey-Predator Algorithm with a Group Hunting Scenario

Surafel Luleseged Tilahun,[1] Hong Choon Ong,[2] and Jean Medard T. Ngnotchouye[1]

[1]*School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, Private Bag X01, Scottsville, Pietermaritzburg 3209, South Africa*
[2]*School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Pulau Pinang, Malaysia*

Correspondence should be addressed to Surafel Luleseged Tilahun; surafelaau@yahoo.com

Received 25 November 2015; Revised 29 March 2016; Accepted 24 April 2016

Academic Editor: Imed Kacem

Prey-predator algorithm (PPA) is a metaheuristic algorithm inspired by the interaction between a predator and its prey. In the algorithm, the worst performing solution, called the predator, works as an agent for exploration whereas the better performing solution, called the best prey, works as an agent for exploitation. In this paper, PPA is extended to a new version called *nm*-PPA by modifying the number of predators and also best preys. In *nm*-PPA, there will be *n* best preys and *m* predators. Increasing the value of *n* increases the exploitation and increasing the value of *m* increases the exploration property of the algorithm. Hence, it is possible to adjust the degree of exploration and exploitation as needed by adjusting the values of *n* and *m*. A guideline on setting parameter values will also be discussed along with a new way of measuring performance of an algorithm for multimodal problems. A simulation is also done to test the algorithm using well known eight benchmark problems of different properties and different dimensions ranging from two to twelve showing that *nm*-PPA is found to be effective in achieving multiple solutions in multimodal problems and also has better ability to overcome being trapped in local optimal solutions.

1. Introduction

Solving optimization problems using metaheuristic algorithms became useful in different applications [1–5]. Metaheuristic algorithms are algorithms which try to improve the quality of a solution set with iterations using an "educated" trial and error way [6]. Since the development of genetic algorithm in mid 1970s, many metaheuristic algorithms are developed and used. The development of new algorithm, the extension of existing algorithm, and using these algorithms to solve real problems are among the leading research issues along with merging algorithms to improve their performance [7–14]. Most of these algorithms are based on the assumption that the optimal solution for an optimization problem is found near the current best solution among the solutions in hand. Hence, this will play a role in focusing on the exploitation of the solution space more than exploration, as solution tends to do an intensified search around the current best rather than exploring the solution space, especially in swarm based algorithms. However, these approaches become weak for deceiving problems where their global optimum is far from local solutions and the landscape of the objective space gives very small information regarding the global solution. In solving such kind of problems, high degree of exploration needs to be used. In addition, appropriate degree of exploitation needs to be used so that the final result will be a good approximation of the global solution. Hence, balancing between exploration and exploitation property of an algorithm is one of the issues that researchers are trying to deal with [10, 15].

Prey-predator algorithm (PPA) is one of the newly introduced metaheuristic algorithms [6, 16]. Even though it is introduced recently, it has been successfully used in different applications [6, 17–19]. Comparison study also shows that it is a promising algorithm compared to existing algorithms [6, 20]. It is inspired by how a predator hunts its prey and how the

prey tries to run away, hide, and survive in this situation. The algorithm still shares the assumption that optimal solution is found around the current best solution at hand. However, it also focuses on exploring the solution space with a given follow-up probability. Two of the randomly generated feasible solutions will be assigned to be the predator and the best prey. The predator is a solution with the worst performance and the best prey is the solution with the best performance. The predator focuses totally on exploration and forcing the other preys to explore the solution space whereas the best prey totally focuses on exploitation. The rest of the solutions are called ordinary prey and affected by either the predator or the best prey based on algorithm parameter, probability of follow-up. Perhaps it is wise to control the exploration and exploitation rate by adjusting the number of best preys and predators, as group hunting is also common in some species of animals.

Inspired by a group hunting behavior of some animals, in this paper, PPA will be enhanced and extended to another version called *nm*-prey-predator algorithm (*nm*-PPA). *nm*-PPA is a PPA with multiple predators and multiple best preys. In this version of PPA, one can adjust the degree of exploration and exploitation by adjusting the number of best preys and predators. Furthermore, it will be shown that indeed *nm*-PPA is promising for multimodal optimization problems by finding multiple solutions within a single run of the algorithm. Since parameter assignment is another challenging issue in implementing metaheuristic algorithms, a discussion on setting parameter values based on the problem given will be discussed. The other contribution of this paper is introducing a way of measuring the exploration and also exploitation performance of algorithms based on their success in finding multiple solutions for multimodal problems.

In Section 2, a brief introduction on prey-predator algorithm will be presented followed by the proposed version of the algorithm, *nm*-PPA. In Section 3, experimental results will be discussed based on selected eight benchmark problems. A conclusion which contains a summary and possible future works will be discussed in Section 4.

2. Extended Prey-Predator Algorithm

2.1. Prey-Predator Algorithm. Metaheuristic algorithms are inspired by different natural aspects. Learning from nature to solve problems became useful as nature finds solution for problems without being told. The interaction between predators and preys has evolved for long. It means that the predators have managed to capture the weak or unfortunate prey to survive and also the preys have managed to survive. There are different kinds of predators and prey interactions [21]. Among those interactions, the interaction in which the predators run after their prey inspires the prey-predator algorithm. These interactions have motivated different algorithms. For example, motivated by the interaction between a predator and its prey, Haynes and Sen [22] discussed evolving behavioral strategies in predator and preys based on the predator-prey pursuit. The paper considers how to surround a prey in order to capture it in a grid space. The other algorithm is spiral predator-prey approach for multiobjective optimization which was proposed by Laumanns et al. [23]. This algorithm is basically designed for multiobjective optimization in which it uses the concept of graph theory where each prey will be placed in the vertices of the graph and uses $(1 + n)$-evolutionary strategy. The prey-predator algorithm we are discussing is different from these algorithms, even though they are motivated by similar scenario and have similar names.

Prey-predator algorithm is a metaheuristic optimization algorithm introduced by Tilahun and Ong [6, 16]. In the algorithm, randomly generated solutions will be assigned with numerical value called survival value depending on their performance in the objective function. For a maximization problem, survival value of a solution is directly proportional to its functional value in the objective function. Once the survival value is computed for each solution, the solution with the least survival value will be assigned as a predator; a solution with the highest survival value will be assigned as a best prey and the rest as ordinary prey. The updating process of the predator, x_p, is done by running after the weakest prey, where a weakest prey is an ordinary prey with least survival value, with a local search step length, and runs randomly with bigger step length, as given in the following:

$$x_p \coloneqq x_p + \lambda_{\max} u_r + \left(\lambda_{\min} \text{rand}\right) u_c, \tag{1}$$

where $\lambda_{\min}$ and $\lambda_{\max}$ are algorithm parameters to determine the maximum exploitation and exploration step lengths, u_r is a random direction, rand is a random number between 0 and one from a uniform distribution, and

$$u_c = \begin{cases} \dfrac{x' - x_p}{\left\| x' - x_p \right\|}, & \text{if } x' \neq x_p \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

for the weak prey x'. An ordinary prey, x_i, follows better prey in terms of survival value and does a local search by choosing a unit direction from random m directions if the probability of follow-up is met; otherwise, it runs randomly away from the predator, as given in the following:

$$x_i \coloneqq \begin{cases} x_i + \left(\lambda_{\max} \text{rand}\right) u_f + \lambda_{\min} u_l, & \text{if rand} \leq p_f \\ x_i + \lambda_{\max} u_{\text{run}}, & \text{otherwise,} \end{cases} \tag{3}$$

where

$$u_f = \begin{cases} \dfrac{\sum_{f(x_j)<f(x_i)} \left(x_j - x_i\right)}{\left\| \sum_{f(x_j)<f(x_i)} \left(x_j - x_i\right) \right\|}, & \text{if } \displaystyle\sum_{f(x_j)<f(x_i)} \left(x_j - x_i\right) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

and u_l is a local search direction given by $u_l = \text{argmin}_u \{ f(x_i + \lambda_{\min} u) \mid u \in \{u_1, u_2, \ldots, u_q\} \}$, where $u_1, u_2, \ldots, u_q$ are random directions for local search and u_{run} is a random direction away from the predator.

The best prey, x_b, will totally focus on local search. The local search is done by generating q random directions and checking if any of these directions improve the performance of the best prey. If such a direction exists, the best prey will move in that direction; otherwise, it will remain in its current position, as given in the following:

$$x_b := x_b + \lambda_{\min} u_l, \tag{5}$$

where $u_l = \operatorname{argmin}_u \{f(x_b + \lambda_{\min} u) \mid u \in \{u_1, u_2, \ldots, u_q, \vec{0}\}\}$ for $u_1, u_2, \ldots, u_q$ being the random q directions and $\vec{0}$ a zero vector.

This updating of solutions will continue until a termination criterion is met.

Hence, the predator works as an agent for exploration by scaring the other preys and forcing them to explore the feasible region. Unlike the predator, the best prey focuses totally on exploitation. The ordinary prey can tend to focus on either exploration or exploitation based on the probability of follow-up. If probability of follow-up is met, if a random number is at most the probability of follow-up, they will do a local search in addition to following better prey and if the probability of follow-up is not met they will explore the solution space by randomly running away from the predator. Hence, in addition to the predator, the probability of follow-up plays a role in the degree of exploration versus exploitation. This means that if the probability of the follow-up is met, then the ordinary prey tends to focus more on exploitation, otherwise on exploration. As for the best prey, it does only exploitation. However, by increasing the number of best preys and the predators, one can increase both the exploration and exploitation.

2.2. nm-Prey-Predator Algorithm. One of the challenges in optimizing multimodal optimization problems is the problem of being trapped in local optimal solution while searching for global solution. For that purpose, the degree of exploration should be sufficiently large in heuristic search, but again making the degree of exploration too big will affect the convergence behavior of the algorithm. The degree of exploration versus exploitation needs a proper tuning based on the problem at hand.

In the previous section, Section 2.1, it is discussed that the predator not only focuses on exploration but also forces other preys to explore the solution space with appropriate value for follow-up probability. Having multiple predators increases the exploration property of the algorithm. On the other hand, the best prey focuses totally on exploitation. Hence, increasing the number of best preys increases the degree of exploitation. In general, the user can tune the degree of exploration by increasing the number of predators from 1 to n and the degree of exploration by increasing the number of best preys from 1 to m. One of the advantages of tuning the number of predators and best preys to control the degree of exploration and exploitation is that it is possible to increase both the exploration and the exploitation behavior of the algorithm by increasing n and m at the same time. A good explorative behavior with larger n makes the algorithm suitable for deceiving problems and helps it to jump out of local solutions and a good explorative behavior with larger value for m helps algorithm to find multiple local and also global solutions in multimodal problems and also better quality solutions will be generated.

The updating of the predator and the best prey, as well as the termination criterion, will be the same as that given in (1) and (5), respectively. The only modification in the updating process of the ordinary prey is that if the probability of follow-up, p_f, is not met, it will run away from the predator with the worst performance. Hence, the basic steps of the proposed algorithm are as follows:

(1) Set up parameter and generate random N solutions.

(2) Rank the solution according to their performance in the objective functions from the worst to the best; for exmple, $x_1, x_2, \ldots, x_N$ where $f(x_i) \geq f(x_{i+1})$.

(3) Categorize the solutions as predators $(x_1, x_2, \ldots, x_n)$, ordinary prey $(x_{n+1}, x_{n+2}, \ldots, x_{N-m})$, and best prey $(x_{N-m+1}, x_{N-m+2}, \ldots, x_N)$.

(4) Move the predators randomly and towards x_{n+1}.

(5) Move the ordinary prey x_i towards $x_{i+1}, x_{i+2}, \ldots, x_N$ if rand $\leq p_f$ for a random number rand; otherwise, move x_i randomly away from x_1.

(6) Move each best prey x_b based on random q and in an improving direction.

(7) If termination criterion is met, end; otherwise, go back to step (2).

The algorithm is summarized as in Figure 1.

The termination criterion can be the maximum number of iterations; no improvement is recorded in consecutive generations or a given level of tolerance is archived if the optimal solution is known.

3. Experimental Results

A simulation is done on selected eight benchmark problems.

3.1. Benchmark Problems. The benchmark problems are selected from different categories such as being differentiable, nondifferentiable, continuous, discontinuous, separable, partially separable, nonseparable, scalable, and nonscalable and are also with dimensions from two up to twelve. All of the selected problems are multimodal and hence nonconvex.

(1) Shubert function [24] is as follows:

$$\begin{aligned} \min \quad & f_1(x) \\ & = \left(\sum_{i=1}^{5} i \cos\left((i+1)x_1 + i\right)\right)\left(\sum_{i=1}^{5} i \cos\left((i+1)x_2 + i\right)\right) \\ \text{s.t.} \quad & -10 \leq x_i \leq 10, \quad \forall i. \end{aligned} \tag{6}$$

It is a multimodal problem with 760 optimal solutions among which the eighteen global solutions are

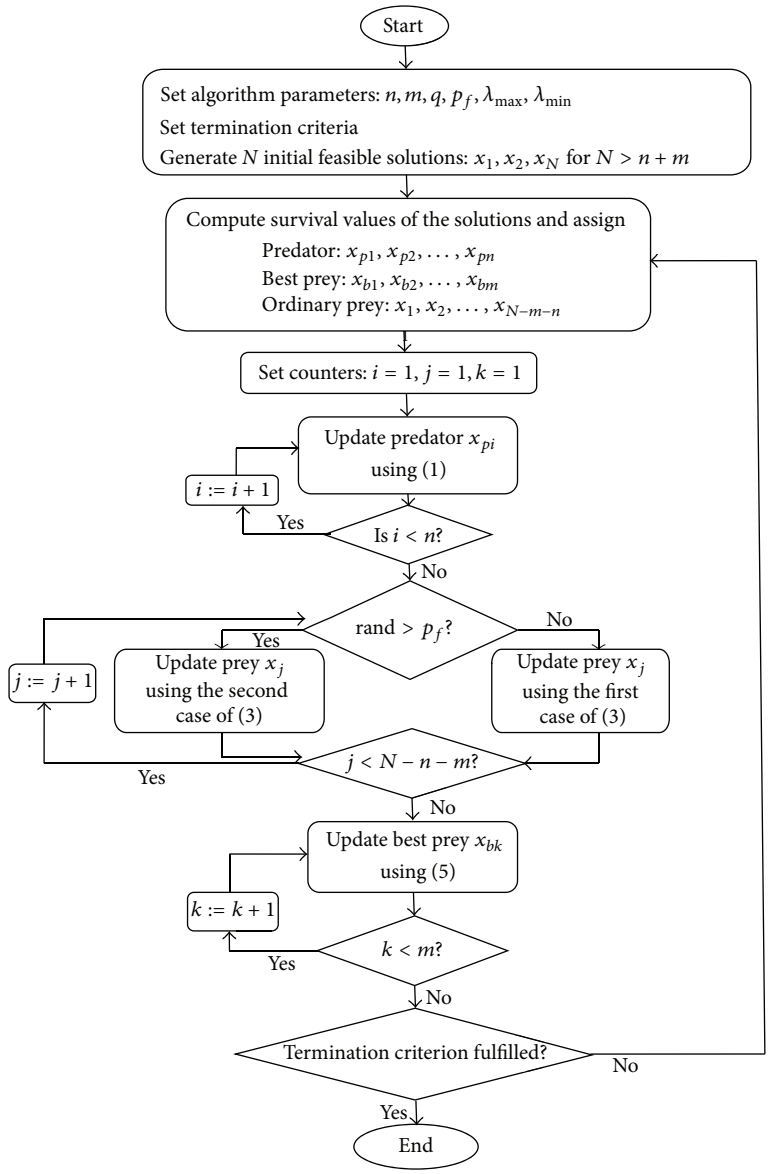


Figure 1: The enhanced prey-predator algorithm (nm-PPA).

located at (7.0835, 4.8580), (−7.0835, −7.7083), (−1.4251, −7.0835), (1.4251, −0.8003), (−7.7083, −7.0835), (−7.7083, −0.8003), (0.8003, −7.7083), (−0.8003, 4.8580), (5.4828, −7.7083), (5.4828, −1.4251), (4.8580, −0.8003), (4.8580, −7.0835), (1.4251, 5.4828), (−0.8003, −1.4251), (−7.7083, 5.4828), (7.0835, −1.4251), (4.8580, 5.4828), and (5.4828, 4.8580) with optimum functional value of −186.7309.

(2) Tripod function [25] is as follows:

$$\begin{aligned} \min \quad & f_2(x) \\ & = p(x_2)\left(1 + p(x_1)\right) \\ & \quad + \left|x_1 + 50p(x_2)\left(1 - 2p(x_1)\right)\right| \\ & \quad + \left|x_2 + 50\left(1 - 2p(x_2)\right)\right| \\ \text{s.t.} \quad & -100 \le x_i \le 100, \quad \forall i, \end{aligned} \tag{7}$$

where

$$p(x) = \begin{cases} 1 & \text{if } x \ge 0 \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

The global optimal value which is $f^* = 0$ is found at $(0, -50)$.

(3) Problem three is as follows: We have constructed a discontinuous, nondifferentiable, multimodal test

problem to be the third test problem, which is given in the following:

$$\begin{aligned} \min \quad & f_3(x) = \sum_{i=1}^{D} \left(p(1) \lfloor x_i \rfloor^4 + p(2) \lfloor x_i \rfloor^3 + p(3) \lfloor x_i \rfloor^2 + p(4) \lfloor x_i \rfloor + p(5) \right) \\ \text{s.t.} \quad & 0 \le x_i \le 12, \quad \forall i, \end{aligned} \tag{9}$$

where $p = [0.3779 - 0.84056.0000 - 14.42007.1340]$. The problem has infinitely many global solutions of $(x_1, x_2, \ldots, x_D)$ with $2 \le x_i \le 3$, $\forall i$. The global optimal value is $f^* = -3.82536D$. Furthermore, the local solutions of the problem are found at $(x_1, x_2, \ldots, x_D)$ for $9 \le x_i \le 10$, $\forall i$, with functional value of $f = -1.43031D$. For the simulation, we set $D = 3$; hence, $f^* = -11.47608$.

(4) Shekel 10 function [26] is as follows:

$$\begin{aligned} \min \quad & f_4(x) = \sum_{i=1}^{10} \frac{1}{\sum_{j=1}^{4} \left(\left(x_j - a_{ij} \right)^2 + c_j \right)} \\ \text{s.t.} \quad & 0 \le x_i \le 10, \quad \forall i, \end{aligned} \tag{10}$$

where

$$A = \left[a_{ij} \right] = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}, \tag{11}$$

$$C = \left[c_{ij} \right] = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{pmatrix}.$$

The global minimum is found at $x = (4, 4, 4, 4)$ with approximate minimum functional value of -10.5319.

(5) Generalized price function is as follows: Based on price function 1 in [27], we construct a generalized price function as given in the following:

$$\begin{aligned} \min \quad & f_5(x) = \sum_{i=1}^{D} \left(|x_i| - t \right)^2 \\ \text{s.t.} \quad & -r \le x_i \le r, \quad \forall i, \end{aligned} \tag{12}$$

where $t \in [0, r)$. The generalized price function has 2^D optimal solutions evenly spaced in the solution space and are found at $(a_1, a_2, \ldots, a_D)$, where $a_i \in \{-t, t\}$, $\forall i$. Here, D, t, and r are set to be 5, 5, and 100, respectively.

(6) Hartman 6 function [28] is as follows:

$$\begin{aligned} \min \quad & f_6(x) = -\sum_{i=1}^{4} c_i e^{-\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2} \\ \text{s.t.} \quad & 0 \le x_i \le 1, \quad \forall i, \end{aligned} \tag{13}$$

where

$$\begin{aligned} \left[a_{ij} \right] &= \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \\ \left[c_i \right] &= \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}, \\ \left[p_{ij} \right] &= \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5586 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}. \end{aligned} \tag{14}$$

The global solution is found at (0.201690, 0.150011, 0.476874, 0.275332, 0.311652, 0.657301) with functional value of -3.32236.

Table 1: Properties of the benchmark problems.

	Dimension	Continuous	Differentiable	Separable	Scalable
f_1	2	Yes	Yes	Partially	No
f_2	2	No	No	No	No
f_3	3	No	No	Yes	No
f_4	4	Yes	Yes	No	Yes
f_5	5	Yes	No	Yes	No
f_6	6	Yes	Yes	No	No
f_7	7	Yes	Yes	No	No
f_8	12	Yes	Yes	Yes	Yes

Table 2: Parameter setting for the simulation.

	p_f	$\lambda_{\max}$	$\lambda_{\min}$	q	m	n	N	$MaxGen$
f_1	0.85	6	3	8	36	10	60	30
f_2	0.70	60	30	12	20	15	50	30
f_3	0.70	3.5	2	16	20	15	50	50
f_4	0.65	3	1.2	28	20	20	60	75
f_5	0.60	70	40	40	25	25	70	85
f_6	0.55	0.35	0.15	52	40	30	100	100
f_7	0.40	75	30	66	250	90	250	400
f_8	0.25	1	0.3	130	100	75	250	300

(7) Pathological function [29] is as follows:

$$\min \quad f_7(x) = \sum_{i=1}^{D-1} \left(0.5 + \frac{\sin^2\sqrt{100x_i^2 + x_{i+1}^2} - 0.5}{1 + 0.001\left(x_i^2 - 2x_i x_{i+1} + x_{i+1}^2\right)^2} \right) \quad (15)$$

$$\text{s.t.} \quad -100 \le x_i \le 100, \quad \forall i.$$

The global optimal solution is found at $(a_1, a_2, \ldots, a_D)$, where $a_i = 0$, $\forall i$, with optimum functional value of 0. D set to be 7.

(8) Deb 3 function [25] is as follows:

$$\min \quad f_8(x) = \frac{1}{D}\sum_{i=1}^{D} \sin^6\left(5\pi\left(x_i^{3/4} - 0.05\right)\right) \quad (16)$$

$$\text{s.t.} \quad -1 \le x_i \le 1, \quad \forall i.$$

The feasible region in the reference was given for each x_i to be between −1 and 1; however, having a negative value for x results in having a complex number involved. Hence, a modification needs to be made either on the feasible set, which means making x_i nonnegative, or replacing $x_i^{3/4}$ by $x_i^{4/3}$. If the second case is taken, we will have $10D$ solutions, but for our case we choose to set all the variables as nonnegative. The problem has D dimensions with $5D$ optimal solutions located at $(a_1, a_2, \ldots, a_D)$, where $a_i \in \{((4t+1)/20)^{4/3} : t \in \{0,1,2,3,4\}\} = \{0.0184, 0.1575, 0.3449, 0.5631, 0.8052\}$, with functional value of zero. For the simulation, we set D to be 12.

All the problems are nonconvex and multimodal. The properties of the functions are summarized in Table 1.

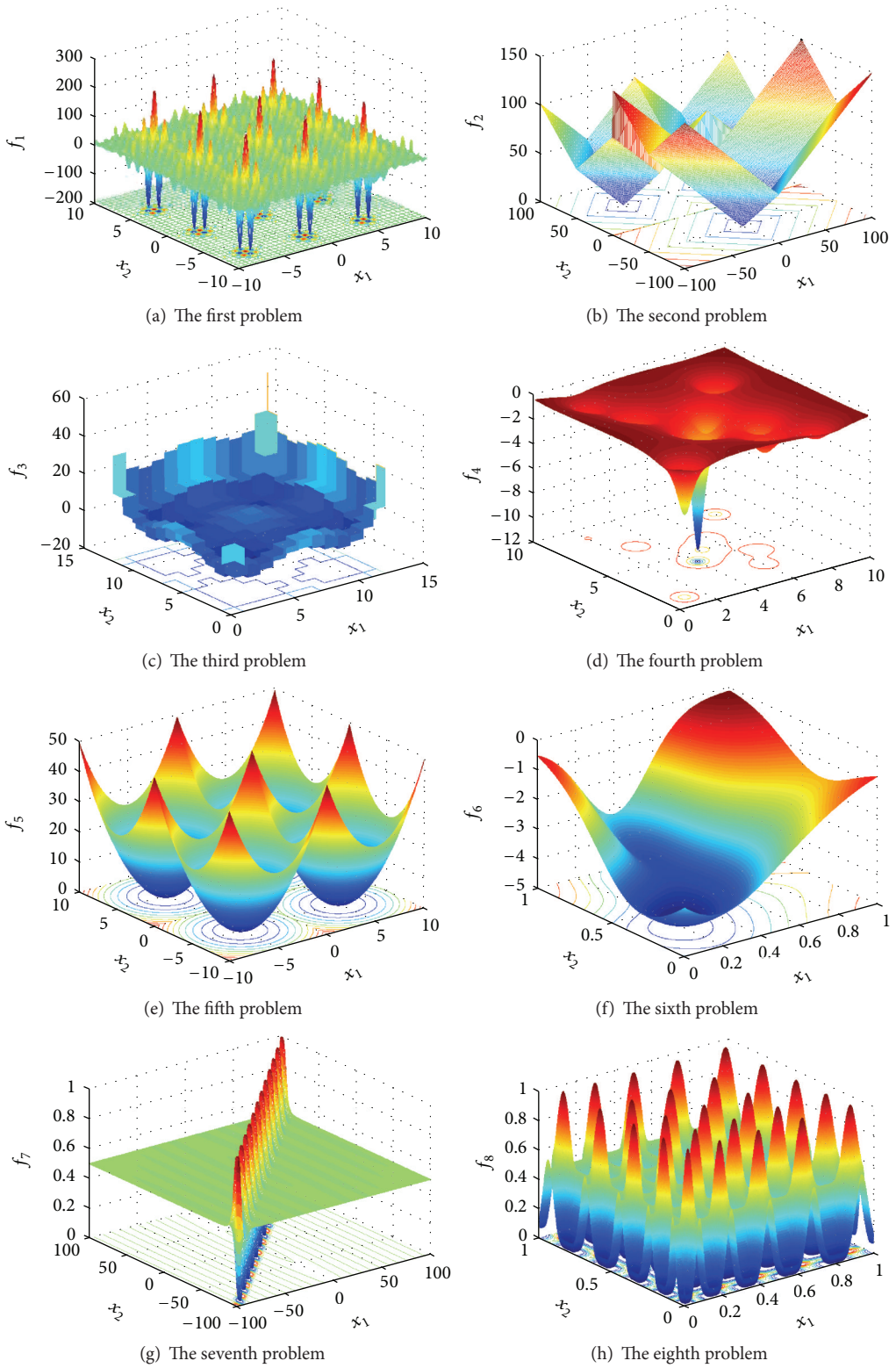
The three-dimensional graphs of the functions, the function with two variables, are given in Figure 2. In drawing the graphs, in cases of the test problems having matrices and vectors, like in problem (6), the appropriate components are taken. For the fifth test problem, the feasible regions are set to be between negative ten and ten so that the behavior of the optimal solutions can be seen clearly.

In addition, the contour graph, which is the image of the objective function on the two-dimensional decision space, is presented in Figure 3. Since the contour numbers, which are the objective function values, are included in the contour graph, it properly demonstrates the landscape of the objective function based on the values in the decision space.

3.2. Simulation Results. To run *nm*-PPA for the selected problems, algorithm parameters need to be set as appropriate as possible, since the performance of the algorithm depends on the parameters. Choosing appropriate values for the parameters is an optimization problem which needs to be dealt with. For instance, if a problem has k optimal solutions, both local and global, assigning m best preys where $m < k$ will not help to find all the k solutions, but if we assign $m > k$, there is a possibility of getting the majority of the solutions, if not all. Higher dimension needs more exploration in general; hence, the probability of follow-up and the dimension will have an inverse relationship as $p_f \prec 1/D$. The number of initial solutions also needs to increase with the dimension and also with the size of the feasible region. The local and exploration step lengths should be determined based on the size of the feasible region, as too big step length causes the solutions to jump out of the feasible region and too short step lengths will affect the speed of convergence. Furthermore, the number of random directions, q, should be directly proportional to the dimension of the problem. The number of best preys, m, and predators, n, should satisfy $n + m < N$, and assigning these values depends on the number of solutions existing in the problem and the number of solutions we are interested to find. In addition to that, it also depends on the degree of explorations and exploitation that we are interested to apply. The termination criterion is set to be the maximum number of iterations, *MaxGen*, and has been set to increase with the dimension, which again gives more chance for the solutions to explore the solution space.

Considering all of the ideas discussed and the recommendations in [6, 16], the algorithm parameters for the simulation have been set as given in Table 2.

3.2.1. Based on Single Trial. It is clear that metaheuristic algorithms are probabilistic solution methods and one can get different results in different runs of the algorithm, even with using the same parameter setting. However, in this section, we will demonstrate the results with a single run of the algorithm to demonstrate how the algorithm works. A detailed analysis with multiple runs will be given in the next section. Hence, after running the algorithm once for the specified number of



(a) The first problem

(b) The second problem

(c) The third problem

(d) The fourth problem

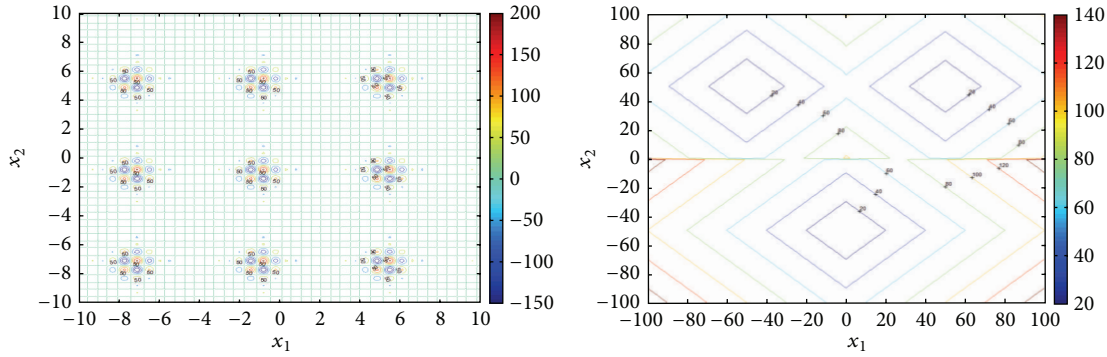
(e) The fifth problem

(f) The sixth problem

(g) The seventh problem

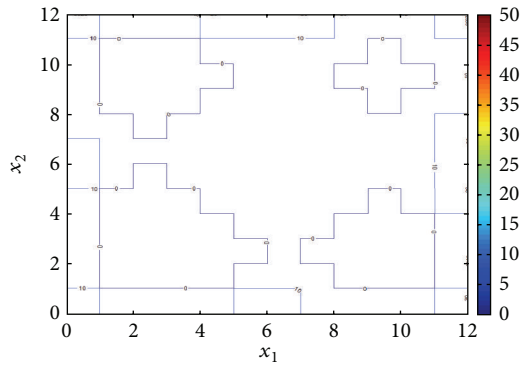
(h) The eighth problem

Figure 2: The benchmark problems in 2 variables.

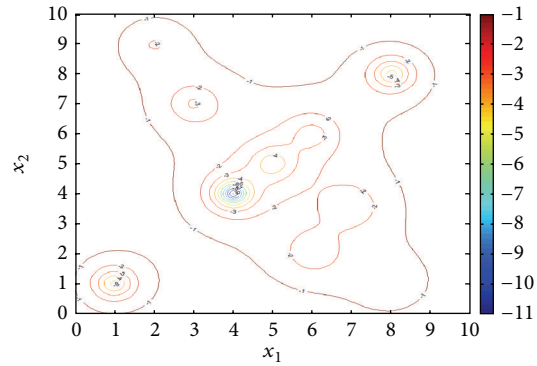


(a) The first problem

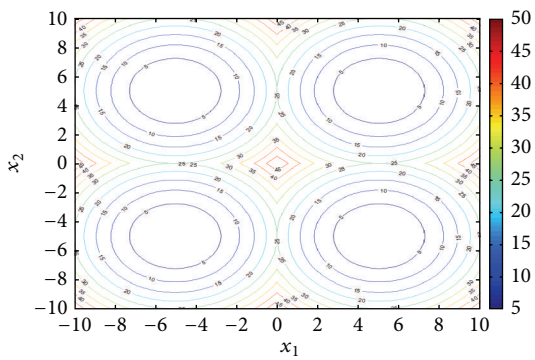
(b) The second problem



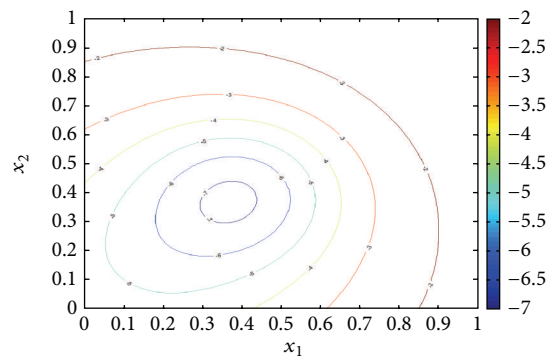
(c) The third problem



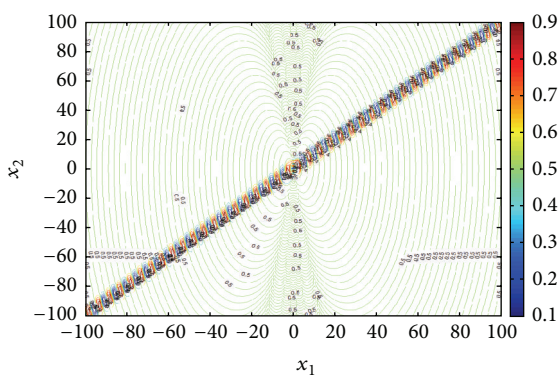
(d) The fourth problem



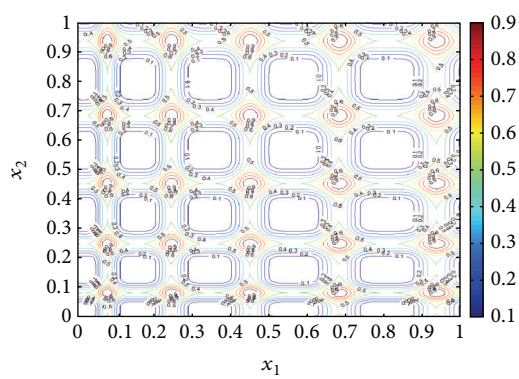
(e) The fifth problem



(f) The sixth problem



(g) The seventh problem



(h) The eighth problem

Figure 3: The contour graph of the benchmark problems in 2 variables.

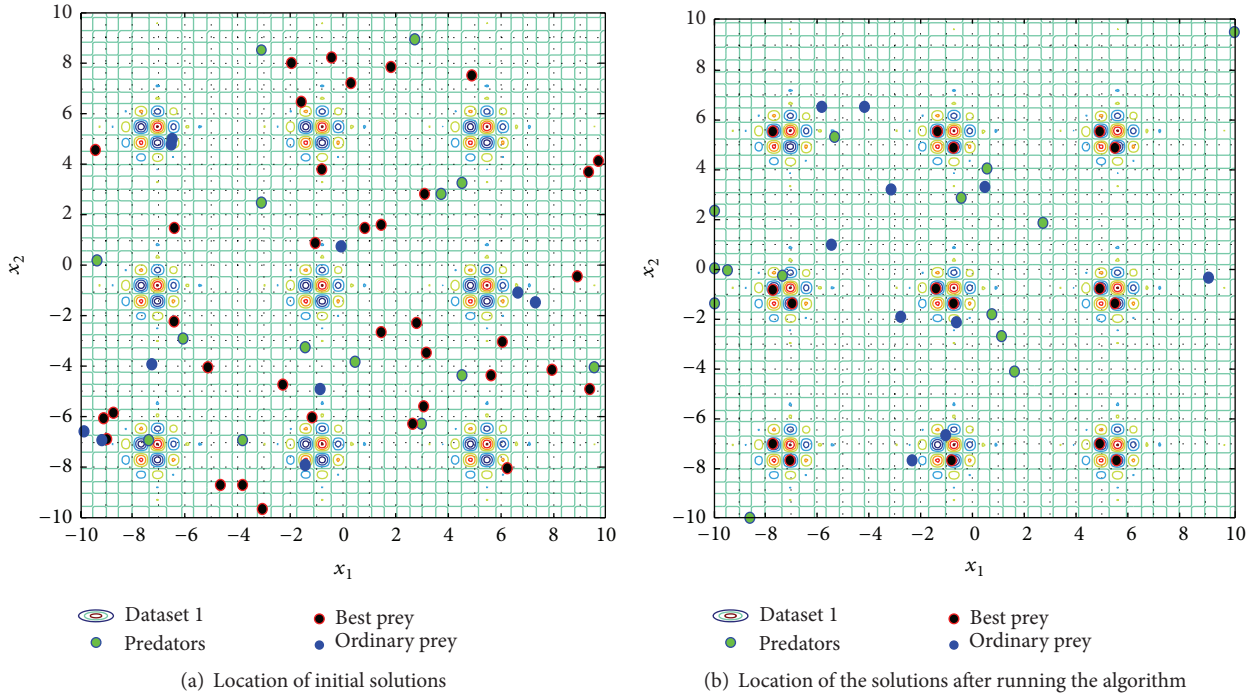


FIGURE 4: Location of solutions at the beginning and the end of running *nm*-PPA for problem one.

iterations, the solutions for each of the problems have been recorded as follows:

(1) First problem: after running *nm*-PPA with the specified values for the parameter, the best solution is found to be $(-0.8009, 4.8578)$ with functional value of -186.7301. Furthermore, thirty-six best preys converge to sixty out of eighty global solutions. The absolute difference between the best functional value and the list among the best solutions is 0.3078. The locations of the solutions at the beginning and at the end of the algorithm run are given in Figure 4.

(2) Second problem: $(0.0001, -0.5004)$ is found to be the best solution with a functional value of 0.0005. *nm*-PPA has also managed to approximate three solutions, one global optimum and two local solutions. Nine of the initial solutions converge around the global solution with a maximum absolute functional difference of 0.0009. Eight of the solutions approximate the local solution $(-50, 50)$ with best approximation of $(-50.1635, 50.0332)$ with a functional value of 1.1966. The other three of the best preys converge around the local solution $(50, 50)$ with the best one being $(50.0046, 49.8839)$ with a functional value of 2.1208. The locations of the solutions at the beginning and the end of the run of the algorithm are given in Figure 5.

(3) Third problem: twenty of the solutions have converged to the flat surface which contains the global optimum solutions, $2 \leq x_i < 3$, with functional value of -11.47608.

(4) Fourth problem: the optimum solution after running *nm*-PPA is found at $(4.0023, 3.9991, 3.9980, 4.0007)$ with functional value of -10.5356, which is even better than the recorded solutions on literatures. Furthermore, ten of the solutions converge around the global optimum which is $(4, 4, 4, 4)$ with the functional value range of $[-10.5356, -10.5313]$. One of the solutions converges to $(7.9981, 7.9958, 8.0008, 8.0004)$ with functional value of -5.1752 which is almost the local minimum of -5.17518 found at $(8, 8, 8, 8)$. Eight of the other solutions converge to the local solution of $(1, 1, 1, 1)$ with the best and worst functional values of -5.1752 and -5.1262, respectively.

(5) Fifth problem: the best optimum solution found is at $(5.0068, -5.0126, 5.0195, 4.9856, -4.9971)$ with functional value of 0.0008. In addition to this, twenty-four of the solutions converge to nineteen global solutions with an absolute maximum functional difference from the best solution with 0.000085.

(6) Sixth problem: the optimum value is found to be at $(0.2028, 0.1496, 0.4742, 0.2759, 0.3114, 0.6540)$ with a functional value of -3.3066. Furthermore, forty of the solutions are aggregated around the global solution with absolute functional value difference from the optimum ranging between 0.00001 and 0.002.

(7) Seventh problem: the optimum functional value found is 0.0000009 at $(0.00004, -0.00001, -0.00001, 0.00000, 0.00002, -0.00001, -0.00004)$. Furthermore, 250 of the solutions give solutions whose functional values are at most 0.00114 and are around the optimal solutions.

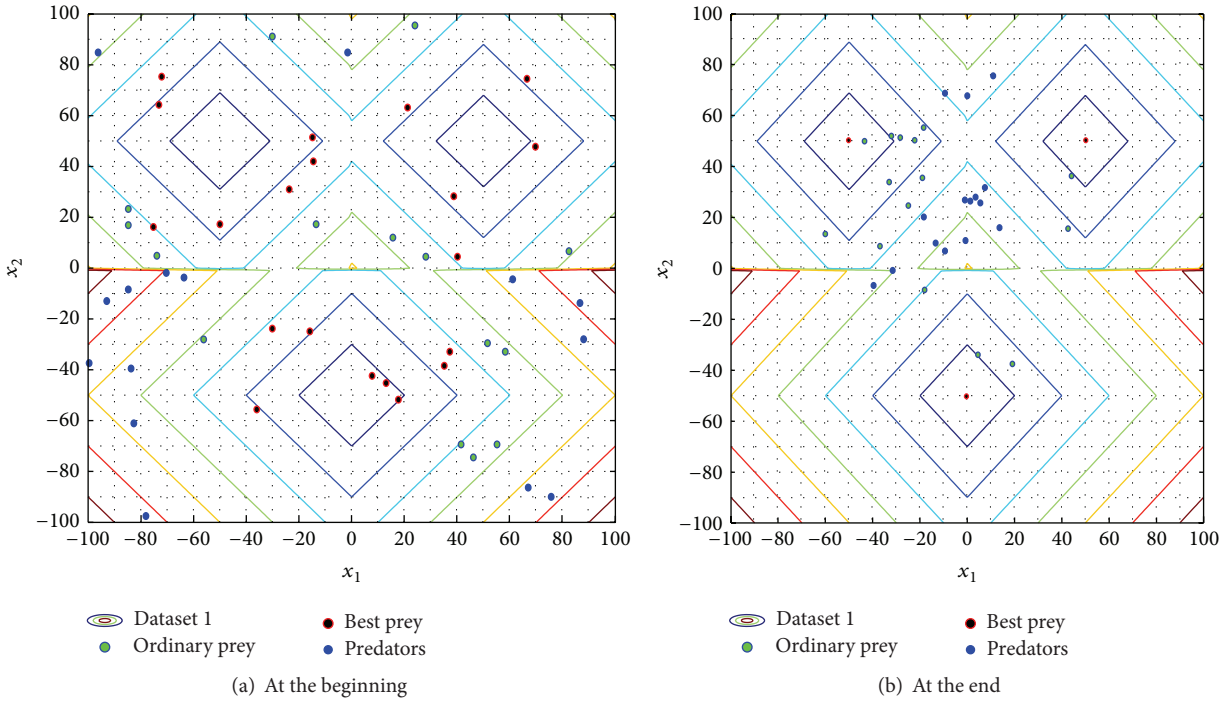


Figure 5: Location of solutions at the beginning and the end of running *nm*-PPA for problem two.

(8) Eighth problem: the optimum functional value after running *nm*-PPA is found to be 10^{-15} at (0.0183, 0.0182, 0.3442, 0.0188, 0.0188, 0.0181, 0.0182, 0.0183, 0.3451, 0.0181, 0.1564, 0.3435). Furthermore, the hundred solutions converge to the eighty-seven global solutions with a functional absolute error of $9.55(10^{-12})$.

The performance and progress of the algorithm per iteration are given in Figure 6.

3.2.2. Based on Multiple Trial. In addition to the single trial, *nm*-PPA has run 100 times for each of the problems and the average and standard deviation of functional value and also number of optimal solutions discovered is recorded and studied. For problems with finite global and local solutions, the performance of the algorithm is measured on how many of the solutions it converges to. Finding multiple solution is a result of good exploration property of the algorithm and how good the solutions are approximated depends on the property of the exploitation. In order to measure the performance of the algorithm, let us define measurement metrics called global success rate and total success rate. The global success rate (GS) is the number of global solutions approximated per the total number of global solutions, as given in the following:

$$\text{G.S.} = \frac{a'}{\min\{a, m\}} 100, \tag{17}$$

where a' is the number of global solutions found under the run of the algorithm, m is the number of best preys, and a is the number of global solutions existing. Similarly, the total success rate (TS) can be measured using the following, where b' is the number of solutions, both local and global, found and b is the total number of solutions existing, both global and local:

$$\text{T.S.} = \frac{b'}{\min\{b, m\}} 100. \tag{18}$$

Furthermore, under multiple trials, the success rate may vary from trial to trial; hence, the average success rate and also the standard deviation of these success rates will be discussed. However, the success rate can be subjective as "how near should a solution be to the global solution so that it will be counted in a'?" may have different answers. Hence, the success rate should be accompanied with a tolerance so that if the absolute error of the approximate solution from the exact solution is at most the given tolerance (Tol(f)), then it will be counted as a success. For the simulation purpose, Tol(f) is set based on the single trial simulation results in the previous section. In addition, a criterion needs to be set to distinguish between two solutions under the tolerance and to take them as different solutions or the same approximate solution. This will be done by setting a distance tolerance between solutions, Tol(x). If the distance between two solutions is under Tol(x), then they will be considered as being approximating the same solution; however, if the distance between them is larger than Tol(x), then the two solutions will be considered as being approximating different solutions. For the simulation purpose, Tol(x) is chosen based on the simulation in the previous section and also by considering the size of the feasible region.

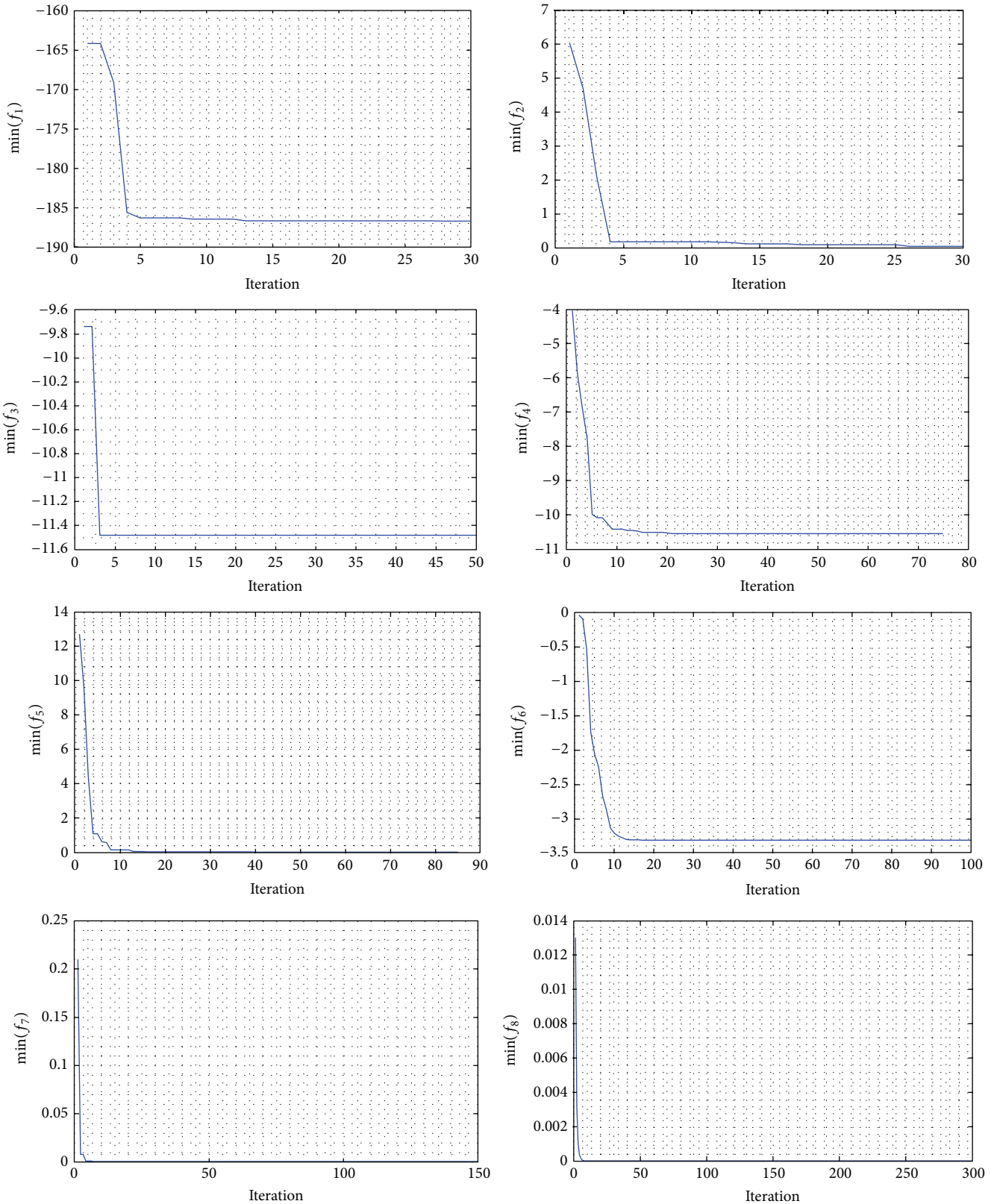


Figure 6: Performance of *nm*-PPA with the iteration on the selected eight benchmark problems.

Table 3: Simulation results with maximum number of iterations as termination criterion based on 100 trials (NK stands for not known and NE for not estimated, i.e., if the number of local solutions is not known, then it is not possible to estimate TS).

	Tol(f)	Tol(x)	Functional value	Number of global solutions found	Number of global solutions	Number of local solutions	G.S.	T.S.
f_1	0.3078	0.8	-186.7302 ± 0.0001	14.98 ± 1.8640	18	742	82.11%	94.33%
f_2	0.0009	1	$(5.3316 \pm 3.4518)10^{-2}$	1.0 ± 0.0	1	2	100%	100%
f_3	0.0001	0.5	$-11.47608 \pm (2)10^{-15}$	19.65 ± 3.0997	Infinite	Infinite	98.25%	98.25%
f_4	0.0043	0.5	-10.5360 ± 0.0006	1.0 ± 0.0	1	NK	100%	NE
f_5	0.0001	0.7	$(1.6174 \pm 0.9143)10^{-3}$	17.8500 ± 8.7255	32	0	71.4%	71.4%
f_6	0.0020	0.1	-3.3066 ± 0.0051	1.0 ± 0.0	1	NK	100%	NE
f_7	0.0011	1	$(0.9330 \pm 1.3130)10^{-6}$	1.0 ± 0.0	1	NK	100%	NE
f_8	0.0001	0.01	$(3.1250 \pm 2.9164)10^{-16}$	67.64 ± 8.6776	5^{12}	0	67.64%	67.64%

Table 4: Simulation results with tolerance level as termination criterion.

	Tol.	Iteration number for *nm*-PPA	CPU time for *nm*-PPA	Iteration number for PPA	CPU time for PPA
f_1	0.3078	6.36 ± 2.0769	$(8.9063 \pm 8.6741)10^{-3}$	17.53 ± 11.4693	$(1.50 \pm 0.31)10^{-2}$
f_2	0.0009	23.7 ± 6.6894	$(9.5312 \pm 2.7720)10^{-2}$	55.05 ± 42.8467	$(4.3594 \pm 4.2207)10^{-1}$
f_3	0.0001	3.1 ± 0.6407	$(7.0317 \pm 7.9752)10^{-3}$	4.40 ± 1.3140	$(1.4063 \pm 0.8633)10^{-2}$
f_4	0.0043	21.99 ± 5.7004	$(9.7222 \pm 8.3540)10^{-3}$	54.40 ± 21.6188	$(1.2083 \pm 0.7069)10^{-2}$
f_5	0.0100	40.84 ± 9.0899	$(5.3125 \pm 7.4768)10^{-3}$	78.76 ± 11.8003	$(1.5938 \pm 0.6689)10^{-2}$
f_6	0.0020	16.85 ± 1.6631	$(2.1875 \pm 0.7853)10^{-2}$	36.10 ± 34.3019	$(3.2813 \pm 0.4809)10^{-1}$
f_7	0.0011	8.10 ± 5.2606	$(5.6328 \pm 0.62185)10^{-1}$	10.35 ± 7.9291	$(7.8438 \pm 0.7580)10^{-1}$
f_8	0.0001	6.1667 ± 1.0532	$(3.5729 \pm 0.4745)10^{-1}$	10.87 ± 3.8393	$(4.3385 \pm 0.2119)10^{-1}$

Hence, based on this, Table 3 presents the simulation results of the 100 trials of the eight benchmark problems, where GS and TS are estimated based on the expected values.

From Table 3, the proposed approach manages to multiple global and local solutions. The success of finding multiple global solution is 70% implying that the algorithm manages to converge to a number of global solutions. For instance, in the first test problem, on average, it has converged to 15 of the 18 global solutions with about 18 local solutions. Hence, 33 of the best preys have converged to different solutions. In the second, the fourth, the sixth, and the seventh problems, the algorithm converges to all local as well as global solutions. In the case of the third problem, the 20 best preys have converged to 20 of the global solutions, on average. Similarly, in the fifth and eighth problems, 18 and 68 of the global solutions are found.

The termination criterion is changed from maximum number of iterations to level of achievement based on a given tolerance. This means that if a tolerance, Tol, is given, then the algorithm will terminate when the functional value becomes at most f^* + Tol, where f^* is the functional value of the global solution. The simulation result, under this termination criterion with hundred trials using *IntelCore™i3-3110M* laptop machine with 2.4 GHz 64-bit operating system, is given in Table 4.

The simulation results show that *nm*-PPA outperforms PPA in all the test problems with a slightly higher processing time. Since the number of best preys increases, the time needed to determine the best direction increases which in turn increases the overall time of the algorithm. In addition, *nm*-PPA converges with smaller iteration number compared to PPA in all the test problems. Hence, the proposed approach is a promising approach in escaping local solutions and also achieving multiple solutions within a single run.

4. Conclusion

In this paper, the recently introduced metaheuristic algorithm called prey-predator algorithm (PPA) has been extended to a new version called *nm*-PPA. PPA is inspired by how a predator hunts its prey and the prey runs away and tries to survive in this situation. This algorithm has been extended by incorporating group hunting behavior. Hence, in *nm*-PPA, there will be multiple predators and best preys as well. By adjusting the number of best preys and predators, it is possible to control the focus of the search mechanism between exploration and exploitation of the search space. Eight benchmark problems are selected with different properties as continuous, discontinuous, differentiable, nondifferentiable, separable, partially separable, nonseparable, scalable, and nonscalable. The simulation results show that ended *nm*-PPA is a promising algorithm for multimodal optimization problems and has an advantage of not being trapped in local solutions; rather, it achieves multiple solutions, both global and local, within a single run. In addition, the way of measuring the global as well as total success of algorithms is introduced and used. Global success measures the success of

an algorithm in finding global optimum solution and total success measures the success of an algorithm in finding global as well as local solution of a problem. It is mentioned that the degree of exploration and exploitation can be adjusted by either varying the number of predators and best preys or adjusting the probability of follow-up. The advantage and disadvantage of these two methods need further study. The other possible future works are to diversify best preys so that they will be forced to converge to global as well as local solutions all over the solution space quickly. Parallel *nm*-PPA also needs exploration as there are higher dimensional and also challenging problems which need high degree of exploration and at the same time a good degree of exploitation. Modifying *nm*-PPA for multiobjective and multilevel optimization problems is also another possible issue along with testing real problems and also hybridization of the algorithm.

Competing Interests

The authors declare that they have no competing interests.

References

[1] M. Villasana and G. Ochoa, "Heuristic design of cancer chemotherapies," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 6, pp. 513–521, 2004.

[2] S. L. Tilahun and H. C. Ong, "Bus timetabling as a fuzzy multiobjective optimization problem using preference-based genetic algorithm," *ROMET-Traffic & Transportation*, vol. 24, no. 3, pp. 183–191, 2012.

[3] H. C. Ong and S. L. Tilahun, "Integrating fuzzy preference in genetic algorithm to solve multiobjective optimization problems," *Far East Journal of Mathematical Sciences*, vol. 55, pp. 165–179, 2011.

[4] M. I. Fraidin, "Decision-making in dependency court: heuristics, cognitive biases and accountability," *Cleveland State Law Review*, vol. 60, pp. 913–975, 2013.

[5] R. Valerdi, "Heuristics for systems engineering cost estimation," *IEEE Systems Journal*, vol. 5, pp. 91–98, 2010.

[6] S. L. Tilahun, *Prey predator algorithm: a new metaheuristic optimization approach [Ph.D. thesis]*, School of Mathematical Sciences, Universiti Sains Malaysia, Penang, Malaysia, April 2013.

[7] N. Hamadneh, S. Sathasivam, S. L. Tilahun, and O. H. Choon, "Learning logic programming in radial basis function network via genetic algorithm," *Journal of Applied Sciences*, vol. 12, no. 9, pp. 840–847, 2012.

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

[9] X.-S. Yang, "A new metaheuristic bat-inspired Algorithm," *Studies in Computational Intelligence*, vol. 284, pp. 65–74, 2010.

[10] X.-S. Yang, "Metaheuristic optimization: algorithm analysis and open problems," in *Experimental Algorithms*, P. M. Pardalos and S. Rebennack, Eds., vol. 6630 of *Lecture Notes in Computer Science*, pp. 21–32, 2011.

[11] S. L. Tilahun, S. M. Kassa, and H. C. Ong, "A new algorithm for multilevel optimization problems using evolutionary strategy, inspired by natural adaptation," in *PRICAI 2012: Trends in Artificial Intelligence: 12th Pacific Rim International Conference on Artificial Intelligence, Kuching, Malaysia, September 3–7, 2012. Proceedings*, P. Anthony, M. Ishizuka, and D. Lukose, Eds., vol. 7458 of *Lecture Notes in Computer Science*, pp. 577–588, Springer, Berlin, Germany, 2012.

[12] C. Blum, J. Puchinger, G. R. Raidl, and A. Roli, "A brief survey on hybrid metaheuristics," in *Proceedings of 4th International Conference on Bioinspired Optimization Methods and their Applications (BIOMA '10)*, B. Filipic and J. Silc, Eds., pp. 3–16, Ljubljana, Slovenia, 2010.

[13] S. L. Tilahun and H. C. Ong, "Vector optimisation using fuzzy preference in evolutionary strategy based firefly algorithm," *International Journal of Operational Research*, vol. 16, no. 1, pp. 81–95, 2013.

[14] C. Blum, C. Cotta, A. J. Fernàndez, J. E. Gallardo, and M. Mastrolilli, "Hybridizations of metaheuristics with branch & bound derivates," *Studies in Computational Intelligence*, vol. 114, pp. 85–116, 2008.

[15] X.-S. Yang, *Nature-Inspired Metaheuristics Algorithm*, Luniver Press, Frome, UK, 2nd edition, 2010.

[16] S. L. Tilahun and H. C. Ong, "Prey-predator algorithm: a new metaheuristic optimization algorithm," *International Journal of Information Technology & Decision Making*, vol. 13, pp. 1–22, 2014.

[17] W. Dai, Q. Liu, and T. Chai, "Particle size estimate of grinding processes using random vector functional link networks with improved robustness," *Neurocomputing*, vol. 169, pp. 361–372, 2015.

[18] B. Bahmani-Firouzi, S. Sharifinia, R. Azizipanah-Abarghooee, and T. Niknam, "Scenario-based optimal bidding strategies of GENCOs in the incomplete information electricity market using a new improved prey—predator optimization algorithm," *IEEE Systems Journal*, vol. 9, no. 4, pp. 1485–1495, 2015.

[19] N. Hamadneh, S. L. Tilahun, S. Sathasivam, and O. H. Choon, "Prey-predator algorithm as a new optimization technique using in radial basis function neural networks," *Research Journal of Applied Sciences*, vol. 8, no. 7, pp. 383–387, 2013.

[20] S. L. Tilahun and H. C. Ong, "Comparison between genetic algorithm and prey-predator algorithm," *Malaysian Journal of Fundamental and Applied Sciences*, vol. 9, no. 4, pp. 167–170, 2014.

[21] C. J. Krebs, *Ecology*, Pearson Education, San Francisco, Calif, USA, 6th edition, 2009.

[22] T. Haynes and S. Sen, "Evolving behavioral strategies in predators and prey," in *Adaption and Learning in Multi-Agent Systems: IJCAI'95 Workshop Montréal, Canada, August 21, 1995 Proceedings*, vol. 1042 of *Lecture Notes in Computer Science*, pp. 113–126, Springer, Berlin, Germany, 1996.

[23] M. Laumanns, G. Rudolph, and H.-P. Schwefel, "A spatial predator-prey approach to multi-objective optimization: a preliminary study," in *Parallel Problem Solving from Nature—PPSN V: 5th International Conference Amsterdam, The Netherlands September 27–30, 1998 Proceedings*, vol. 1498 of *Lecture Notes in Computer Science*, pp. 241–249, Springer, Berlin, Germany, 1998.

[24] M. Molga and C. Smutnicki, *Test Functions for Optimization Needs*, 2005.

[25] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013.

[26] J. Opacic, "A heuristic method for finding most extrema of a nonlinear functional," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 1, pp. 102–107, 1973.

[27] W. L. Price, "A controlled random search procedure for global optimisation," *The Computer Journal*, vol. 20, no. 4, pp. 367–370, 1977.

[28] J. K. Hartman, *Some Experiments in Global Optimization*, School United States Naval Postgraduate, 1972.

[29] S. Rahnamayan, H. R. Tizhoosh, and N. M. M. Salama, "A novel population initialization method for accelerating evolutionary algorithms," *Computers & Mathematics with Applications*, vol. 53, no. 10, pp. 1605–1614, 2007.