Artif Life Robotics (2016) 21:112–119
DOI 10.1007/s10015-015-0248-3

ORIGINAL ARTICLE

A hybrid bat algorithm with natural-inspired algorithms for continuous optimization problem

Sakkayaphop Pravesjit[1]

Received: 8 April 2015 / Accepted: 28 October 2015 / Published online: 24 December 2015

Abstract This paper proposes a hybrid bat algorithm with natural-inspired algorithms for continuous optimization problem. In this study, the proposed algorithm combines the reproduction step from weed algorithm and genetic algorithm. The reproduction step is applied to clone each bat population by fitness values and the genetic algorithm is applied in order to expand the population. The algorithm is evaluated on eighteen benchmark problems. The computational results of the proposed algorithm are compared with the methods in the literature which are self-adaptive differential evolution (DE), traditional DE algorithm, intersection mutation differential evolution (IMDE) algorithm, and the JDE self-adaptive algorithm. Findings show that the algorithm produces several solutions obtained by the previously published methods especially for the continuous unimodal function, the quartic function, the multimodal function and the discontinuous step function. In addition, the finding shows that the proposed algorithm can produce optimal solutions efficiently on benchmark instances within short computational time.

Keywords Bat algorithm · Natural-inspired algorithm · Reproduction step · Optimization problem

This work was presented in part at the 20th International Symposium on Artificial Life and Robotics, Beppu, Oita, January 21–23, 2015.

✉ Sakkayaphop Pravesjit
sakkayaphop.pr@up.ac.th

1 School of Information and Communication Technology, University of Phayao, Phayao 56000, Thailand

1 Introduction

Optimization is an attempt of receiving the optimal solution of the problem under the given situation. The main objective of the optimization is to reduce the time or increase the desired benefits. Optimization methods can be defined as the process of attaining optimal solutions that respond to the given objective functions. Recently, many algorithms have been brought to solve the problem. Nature-inspired algorithm as swarm intelligence and evolutionary algorithm is an effective algorithm which has been developed and published by many researchers.

Bat algorithm was developed by Yang [1] in 2010. The main idea of the algorithm is derived from the behavior of bats searching for food/prey. There are three important steps of the algorithm as follows: first, bats search the object by using sound reflection to recognize the distance between food/prey and obstacles. In the second step, while flying (in random) it changes frequency, loudness and pulse emission rate which can be adjusted to find food/prey automatically based on the close proximity of the target. The pulse will start from zero and will increase gradually as the bat approach their food/prey. In the last step, loudness will be changed in various ways when approaching food/prey (i.e., change volume of the highest to the lowest).

In recent years, many studies have been developed in applying the bat algorithm for solving topics in various fields. It proves that bat algorithm works efficiently with a typical quick start [2, 12]. Nevertheless, the gap of improvement could be found and presented to solve the optimization problem in several papers as the following examinations.

Yang and Gandomi [3] present a new metaheuristic method which is modified from bat algorithm for continuous optimization problems, such as dealing with highly nonlinear problem efficiently and finding the optimal solutions accurately in comparison to GA algorithm and PSO algorithm. Tsai et al. [4] developed Evolved Bat Algorithm (EBA) by changing the movement process of original bat algorithm. The accuracy of the algorithm is better than the original bat algorithm for solving numerical optimization problem. Ramesh et al. [5] presented the application of bat algorithm based on mathematical modeling for solving the multi-objective optimization problem in a power system. The application was used as a test on two test cases. The result was much better than the ones from the RGA, SGA, Hybrid GA, and ABC algorithm. Musikapun and Pongcharoen [6] presented bat algorithm that was based on a scheduling tool (BAST) for solving muti-stage, multi-machine and multi-product scheduling problems. Mishra et al. [7] proposed on combining the bat algorithm to update the weights of a functional link artificial neural network (BAT-FLANN) for classification. It used the echo location to find the minimum distance to the objects and moved the bat to a new solution. The result of the algorithm is faster than the PSO algorithm. Yılmaz and Küçüksille [8] proposed an enhanced bat algorithm (EBA). It combines the inertia weight modification (IS1), distribution of the population modification (IS2), and hybridization with invasive weed optimization (IS3). The IS1 method modifies the process of velocity and location similarities with PSO. The (IS2) method proposed on modifying the velocity with the equation $v_i^t = \omega\left(v_i^{t-1}\right) + \left(x_i^t - x_*\right)f_i\zeta_1 + \left(x_i^t - x_k^t\right)f_i\zeta_2$ from the standard velocity of PSO, and the IS3 method improved the local search capability of bat algorithm with IWO. The paper claims that the EBA provides better results than the standard Bat algorithm. Mehrabian and Lucas [9] presented a novel stochastic optimization model in 2006. The basic properties of the process on colonizing behavior of weeds are the following: a limited number of seeds are spread out over the search area, seeds of plants flowering and seed production was depended on the fitness, and seed production was distributed randomly over the searching area, and new plants were grown. With the propagation of weeds in nature, the stronger weed has the opportunity to grow more than the weaker ones. Reproduction step from invasive weed algorithm generated seeds that are randomly over the search space dimension from a normal distribution. Basak et al. [10] presented the hybrid invasive weed optimization algorithm (IWO) and differential evolution called differential invasive weed optimization (DIWO). The DE/rand/1/bin is used in the seed production step for mutation populations. The paper shows that the performance of DIWO is better than the original invasive weed optimization (IWO) and differential evolution (DE). Zhang et al. [11] illustrated a modified invasive weed optimization with a crossover operation (MIWO) for impeding premature convergence to a local optimum. The procedure of crossover operation starts after a spatial dispersal step of invasive weed optimization. The performance of MIWO is better than the standard invasive weed optimization and particle swarm optimization.

The bat algorithm and the improvement of bat algorithm can solve optimization problem efficiently in the literature especially for low-dimensional functions. However, it still has some drawbacks in dealing with the height dimension function. The performance of the algorithm decreased significantly and is easily trapped in the local optima because it tends to initially converge at a very pace [2, 12]. The IWO algorithm and GA have a strong robustness and a fast global searching ability. They provide the way of reproduction and spatial dispersal in competitive exclusion more and are distinctive than other numerical search algorithms Therefore, the IWO and GA are embedded in the traditional bat algorithm to deal with these weakness points. The reproduction concept from the IWO, mutation crossover, and selection concepts from the GA are applied as local search operator in order to expand the searching area and improve the convergence performance of the original bat algorithm. All details are described in the following section.

Section 2 describes the original bat algorithm (BA). Section 3 presents a proposed algorithm with a description of each step. The experimental results are explained in Sect. 4, and the conclusion is given in Sect. 5.

2 Bat algorithm

The metaheuristic Bat algorithm was developed by Yang in 2010. The algorithm is based on the echolocation behavior of micro-bats with varying pulse rates of emission and loudness. The echolocation idea of the bats can be concluded as follows: Each bat flies randomly with a velocity v_i at position (solution) x_i with a wavelength and loudness A_0. In the searching area, the bat changes frequency, loudness and pulse emission rate $r \in [0, 1]$ to find its prey. The best solution will be selected and the iteration will be continued until the stopping criteria is met (Fig. 1).

The structure of bat algorithm consists of the following components:

Step 1 the NB vectors of initial bat populations are generated. All vectors $X_i^B = (x_{1i}, x_{2i}, x_{3i}, \ldots, x_{Di})$ are randomly produced with uniform distribution between 0 and 1, where $i = 1, 2, 3, \ldots, NB$; D is the dimension of the target vector and B is refers to Bth generation. After the

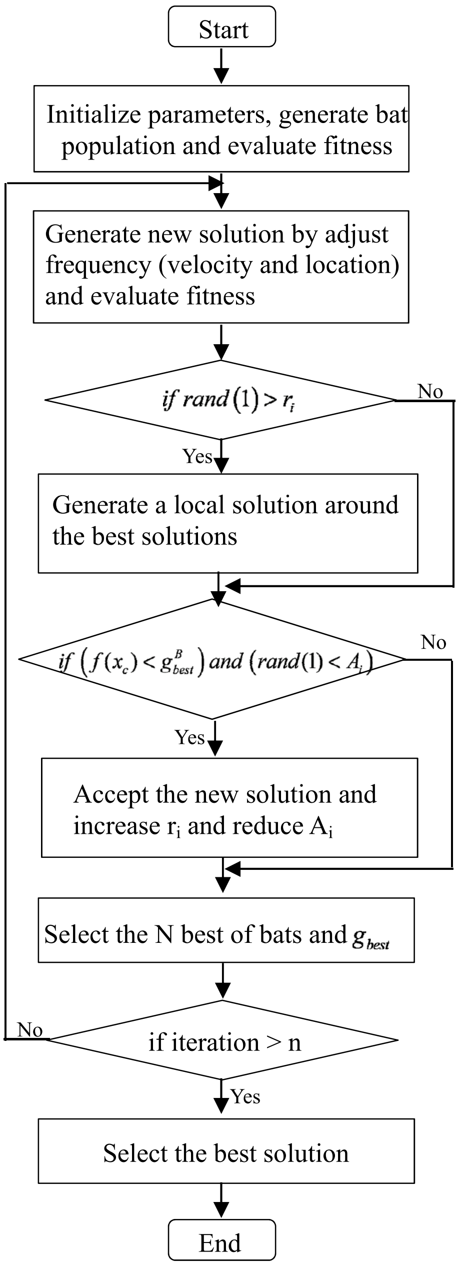
Start
Initialize parameters, generate bat population and evaluate fitness
Generate new solution by adjust frequency (velocity and location) and evaluate fitness
if rand(1) > r_i
No
Yes
Generate a local solution around the best solutions
if (f(x_c) < g^B_best) and (rand(1) < A_i)
No
Yes
Accept the new solution and increase r_i and reduce A_i
Select the N best of bats and g_best
No
if iteration > n
Yes
Select the best solution
End

Fig. 1 Flowchart of the bat algorithm

initial bat population are produced, velocity vector $V_i^B = (v_{1i}, v_{2i}, \ldots, v_{Di})$ is constructed; the maximum frequency ($Q_{\max}$) value and minimum frequency ($Q_{\min}$) value are evaluated. Then pulse rate (r_i) value and loudness A_i^B value are randomly generated.

Step 2 Bat motion step. All bats move from current location, B, to new location, $B + 1$, and turn to be the new solutions by the equation as follows:

$$Q_i^B = Q_{\min} + \left(Q_{\max} - Q_{\min}\right) \times \mathrm{rand}(0,1) \tag{1}$$

$$v_i^{B+1} = v_i^B + \left(x_i^B - x_{best}\right) \times Q_i^B \tag{2}$$

$$x_i^{B+1} = x_i^B + v_i^{B+1} \tag{3}$$

where x_{best} is the best fitness value in the population at generation B.

Step 3 the local search step. Local search is applied to each solution. Random value will be generated for each solution to be compared with the pulse rate. The solution which has less random number than pulse rate will be selected. Local search modifies the current best solution by the following equation:

$$x_i^{B+1} = x_{\text{best}} + \left(\varepsilon A_i^B\right) \times \mathrm{rand}(0,1) \tag{4}$$

where the factor ε is limited the step size of random walks between 0 and 1.

Step 4 after local search step, the pulse rate r_i^B value increases and decreases the loudness A_i^B value by the following equation:

$$A_i^{B+1} = \alpha A_i^B \tag{5}$$

$$r_i^{B+1} = r_i^B \left[1 - \exp^{(-\gamma\varepsilon)}\right] \tag{6}$$

where α and γ are constants. The whole process is repeated until the desired number becomes satisfied.

3 Proposed algorithm

In the previous section, it was found that the local search of BA (Step 3) is likely to stick with the local space. To compare with other metaheuristic algorithms, BA is more complicated [13] because each bat is assigned a set of interacting parameters (e.g., position, velocity, pulse rate, loudness, and frequencies) which affect solution quality and time needed to obtain a solution.

This paper proposes a hybrid bat algorithm with an invasive weed optimization algorithm (IWO) and genetic algorithm (GA) which has the ability for a good exploration as a continuous optimization problem. The procedure for creating the proposed model includes the following steps:

Step 1 initialization of the population. This process is the same as the bat algorithm. It generates the maximum $s_{\max}$ and least $s_{\min}$ value of the seed. After that, the fitness values of the population are to be evaluated.

Step 2 the reproduction step from the invasive weed optimization algorithm is applied to calculate the number of seed. After that, the current population will be

Table 1 The benchmark functions

Functions	D	Range	Iterations	$f_{\min}$
$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	[−100, 100]	1,500	0
$f_2(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	30	[−10, 10]	2,000	0
$f_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	30	[−100, 100]	5,000	0
$f_4(x) = \max_i\{\lvert x_i\rvert, 1 \le i \le D\}$	30	[−100, 100]	5,000	0
$f_5(x) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right) + (x_i - 1)^2\right]$	30	[−30, 30]	20,000	0
$f_6(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5\rfloor)^2$	30	[−100, 100]	1,500	0
$f_7(x) = \sum_{i=1}^{D} ix_i^4 + \mathrm{rand}[0, 1)$	30	[−1.28, 1.28]	3,000	0
$f_8(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	[−500, 500]	9,000	−12,596.5
$f_9(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	[−5.12, 5.12]	5,000	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{D} \cos 2\pi x_i\right) + 20 + e$	30	[−32, 32]	1,500	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600, 600]	2,00	0
$f_{12}(x) = \frac{\pi}{30}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2\left[1 + \sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)\,;$	30	[−50,50]	1,500	0
$f_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1} (x_i - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_{30})\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4);$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0 & -a \le x_i \le a, \\ k(-x_i - a)^m, & x_i < -a \end{cases}$ $y_i = 1 + \frac{1}{4}(x_i + 1)$	30	[−50,50]		0
$f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right]^{-1}$	2	[−65.536, 65.536]	100	0.998004
$f_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5, 5]	4,000	0.0003075
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5, 5]	100	−1.0316285
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[−5, 5]	100	0.397887
$f_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	[−2, 25]	100	3

cloned so that it becomes equal to the number of seed. This is done by the following equation:

$$N_{\mathrm{seed}}^i = \frac{\mathrm{fit}_i^B - \mathrm{fit}_{\min}^B}{\mathrm{fit}_{\max}^B - \mathrm{fit}_{\min}^B}\left(s_{\max} - s_{\min}\right) + s_{\min} \qquad (7)$$

where fit_i^B is the present bat fitness, $\mathrm{fit}_{\max}^B$ and $\mathrm{fit}_{\min}^B$ represent the maximum fitness and minimum fitness of the current population, $s_{\max}$ and $s_{\min}$ represent the maximum and minimum seeds value of a weed.

Step 3 genetic algorithm is applied in order to expand the population as the following procedure:

i. Mutation operation is carried out on all clone population by the equation:

$$G_s^B = x_i^B + \left(\exp^{-j/\max_k}\right) \times \mathrm{rand}(0, 1) \qquad (8)$$

where $j = 1, 2, 3, \ldots, \max_k$, $\max_k$ is the order value of seed from N_{seed}^i and $s = 1, 2, 3, \ldots, \sum N_{\mathrm{seed}}^i$.

Table 2 The parameters used in the experiments

Parameters	Values
Population size, *NB*	50
Loudness, *A*	0.9
Pulse emission, *r*	0.3
Minimum frequency, $f_{\min}$	0
Maximum frequency, $f_{\max}$	1
Minimum number of seeds, $s_{\min}$	1
Maximum number of seeds, $s_{\max}$	5
Mutation parameter, P_m	0.8
Crossover parameter, P_c	0.3

ii. The roulette wheel selection is applied to select the population and then the crossover operation is performed. As a result, $\left(3 \times N^i_{\text{seed}}\right) + 1$ numbers of offspring are generated. Finally, the vector $R^p_c; c = 1, 2, 3, \ldots, \sum \left[\left(3 \times N^i_{\text{seed}}\right) + 1\right]$ will be produced and the fitness values will be evaluated.

iii. After the offspring are created in ii), the new solution will be generated by moving the position from the following equations:

$$Q^p_c = Q^p_{\min} + \left(Q^p_{\max} - Q^p_{\min}\right) \times \text{rand}(0, 1) \tag{9}$$

$$v^{p+1}_c = v^p_i + \left(x^p_c - x^M_{\text{best}}\right) \times Q^p_c \tag{10}$$

Table 3 The computational time of the proposed algorithm

Functions	Iterations [computation time (s)]										Max (s)	Min (s)	Avg. time	SD
	1	2	3	4.	5	6	7	8	9	10				
f_1	7.681	3.368	2.801	2.801	2.797	2.801	2.799	2.798	2.802	2.787	7.681	2.787	3.343	1.534
f_2	22.147	22.626	25.829	33.165	21.970	32.485	30.717	18.478	33.005	20.467	33.165	18.478	26.089	5.719
f_3	10.614	19.664	19.344	10.682	12.942	11.015	20.412	17.794	11.517	12.786	20.412	10.614	14.677	4.106
f_4	2.964	2.924	4.846	3.970	6.315	4.361	4.305	7.646	3.195	6.933	7.646	2.924	4.746	1.684
f_5	31.421	39.145	30.801	38.598	42.428	33.479	26.689	40.582	32.295	39.033	42.428	26.689	35.447	5.162
f_6	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.115	0.000
f_7	14.091	12.280	13.706	6.646	8.238	5.446	13.766	8.358	18.042	15.191	18.042	5.446	11.576	4.142
f_8	16.249	10.619	13.073	12.043	9.596	9.620	10.928	6.363	10.027	10.998	16.249	6.363	10.952	2.569
f_9	3.393	3.613	3.073	6.637	3.367	6.027	9.038	4.879	0.223	0.223	9.038	0.223	4.047	2.738
f_{10}	2.523	1.927	1.807	3.428	4.837	3.681	3.484	4.258	2.135	1.946	4.837	1.807	3.003	1.080
f_{11}	21.150	19.281	24.834	28.096	17.355	17.361	23.213	0.104	0.104	0.104	28.096	0.104	15.160	10.895
f_{12}	26.990	35.186	28.797	34.795	25.377	44.010	31.254	35.360	36.612	30.265	44.01	25.377	32.865	5.478
f_{13}	32.170	36.172	22.327	29.052	36.717	35.494	19.946	18.803	23.130	28.113	36.717	18.803	28.192	6.851
f_{14}	3.854	4.352	3.769	4.277	4.935	2.763	2.238	2.155	2.155	3.603	4.935	2.155	3.410	1.015
f_{15}	1.087	1.114	1.118	1.093	1.466	2.669	2.731	2.530	1.600	1.080	2.731	1.08	1.649	0.710
f_{16}	6.580	6.277	2.506	1.579	2.739	1.765	1.068	1.875	3.040	1.785	6.58	1.068	2.921	1.939
f_{17}	5.410	5.022	5.558	5.069	3.222	1.790	2.958	3.852	2.219	1.692	5.558	1.692	3.679	1.515
f_{18}	2.164	3.816	2.556	2.637	2.223	2.993	2.936	1.823	1.358	1.063	3.816	1.063	2.357	0.816

Table 4 The computation time of the proposed algorithm in comparison with other algorithms

Functions	DE (s)	Evo-DE (s)	Proposed algorithm (s)	Functions	DE (s)	Evo-DE (s)	Proposed algorithm (s)
f_1	3.372	22.390	**2.787**	f_{10}	4.400	63.123	**1.807**
f_2	**2.857**	27.187	18.478	f_{11}	2.238	10.994	**0.104**
f_3	21.321	216.511	**10.614**	f_{12}	**8.066**	66.720	25.377
f_4	6.775	111.102	**2.924**	f_{13}	**9.357**	67.391	18.803
f_5	**12.472**	130.271	26.689	f_{14}	**0.340**	2.208	2.155
f_6	0.238	2.658	**0.115**	f_{15}	5.524	22.269	**1.08**
f_7	11.425	207.209	**5.446**	f_{16}	**0.065**	2.037	1.068
f_8	13.634	373.135	**6.363**	f_{17}	**0.125**	2.028	1.692
f_9	14.371	11.979	**0.223**	f_{18}	**0.133**	2.028	1.063

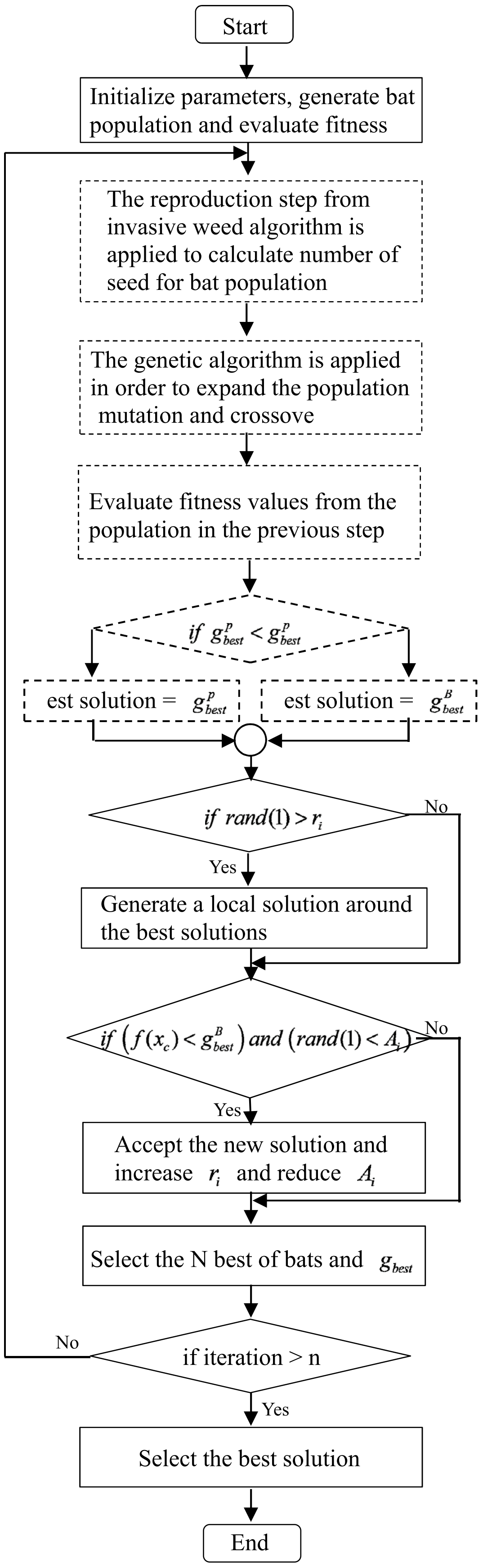
Start
Initialize parameters, generate bat population and evaluate fitness
The reproduction step from invasive weed algorithm is applied to calculate number of seed for bat population
The genetic algorithm is applied in order to expand the population mutation and crossove
Evaluate fitness values from the population in the previous step
if $g^p_{best} < g^p_{best}$
est solution = g^p_{best}
est solution = g^B_{best}
if $rand(1) > r_i$
No
Yes
Generate a local solution around the best solutions
if $(f(x_c) < g^B_{best})$ and $(rand(1) < A_i)$
No
Yes
Accept the new solution and increase r_i and reduce A_i
Select the N best of bats and g_{best}
No
if iteration > n
Yes
Select the best solution
End

Fig. 2 Flowchart of the proposed hybrid algorithm

$$x_c^{p+1} = x_c^p + v_c^{p+1} \tag{11}$$

$$x^M_{\text{best}} = \begin{cases} x^p_{\text{best}} : & \text{if } \text{fit}^p_{\text{best}} < \text{fit}^B_{\text{best}} \\ x^B_{\text{best}} \end{cases} \tag{12}$$

where x^p_{best} is the best population from process ii).

iv. The local search step: The local search is applied to each solution. The random value will be generated for each solution to be compared with the pulse rate. The solution which has less random number than pulse rate will be selected by the following specific equation:

$$x_c^{p+1} = \begin{cases} x^p_{\text{best}} + \left(\varepsilon A_i^B\right) \times \text{rand}(0,1) & \text{if} : x^p_{\text{best}} < x^B_{\text{best}} \\ x^B_{\text{best}} + \left(\varepsilon A_i^B\right) \times \text{rand}(0,1) \end{cases} \tag{13}$$

Step 4 after the local search step, the loudness is compared with the new random number in the range [0, 1] if the random value is less than the loudness value and $f\left(x^p_{\text{best}}\right) < f\left(x^B_{\text{best}}\right)$ then accepts the new solutions and increase r_i and reduce A_i.

Step 5 the selection operator selects X_i^B numbers of individuals from the current generation for progression to the next generation.

All processes will be continued until the stopping condition is met, while the best solution returns.The flowchart of the proposed hybrid algorithm is shown in Fig. 2.

4 Experimental results

The performance of the proposed algorithm is evaluated on 18 benchmark functions from Yao et al. [14] which is demonstrated in Table 1. The benchmark function equations include the continuous unimodal function (functions 1–5), the quartic function (function 7), the discontinuous step function (function 6) and the multimodal function (functions 8–13), and the low-dimensional function (functions 14–18). In evaluating the time and performance of the algorithm, the experiment environments are set in the same way as Jitkongchuen and Thammano [15]. The control parameters in the experiment are presented in Table 2.

The first part of the experiment was to evaluate and compare the computational time employed by the proposed algorithm and other comparative algorithms over 10 runs, using tenfold cross-validation method. The computational time including the maximum time (Max), minimum time (Min), average of computational time (Avg. time) and standard deviation (SD) values of the proposed algorithm are presented in Table 3. The table claims that the proposed algorithm spent just only few computational times to obtain the solution in each function. More significantly, the

Table 5 The experimental results of the proposed algorithm in comparison with other algorithms

Functions	DE	JDE	IMDE 1st process	IMDE 2nd process	Evo-DE	Proposed algorithm
f_1	1.58E − 3	1.1E − 28	2.5E − 32	2.1E − 35	**0**	**0**
f_2	2.84E − 12	1.0E − 23	3.3E − 23	1.7E − 25	**0**	**0**
f_3	0.4110	3.1E − 14	1.2E − 24	7.8E − 29	**0**	**0**
f_4	1.9E − 3	**0**	0.2E − 3	3.4E − 24	**0**	**0**
f_5	8.35E − 27	**0**	**0**	**0**	**0**	**0**
f_6	**0**	**0**	**0**	**0**	**0**	**0**
f_7	2.63E − 3	3.15E − 3	2.4E − 4	3.4E − 4	7.73E − 7	2.49E − 40*
f_8	**−12,569.5**	**−12,569.5**	**−12,569.5**	**−12,569.5**	**−12,569.5**	**−12,569.5**
f_9	15.0430	**0**	**0**	**0**	**0**	**0**
f_{10}	1.5017	7.7E − 15	4.9E − 15	4.6E − 15	4.44E − 16	6.59E − 20*
f_{11}	**0**	**0**	**0**	**0**	**0**	**0**
f_{12}	7.04E − 10	6.6E − 30	1.7E − 32	1.6E − 32	1.57E − 32	2.16709E − 40*
f_{13}	5.07E − 5	5.0E − 29	1.7E − 32	1.3E − 32	1.3E − 32	4.7545E − 30
f_{14}	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**
f_{15}	0.0003075	0.0004	0.0003089	0.0003692	**0.0003075**	0.000369
f_{16}	**−1.0316285**	**−1.0316285**	**−1.0316285**	**−1.0316285**	**−1.0316285**	−1.25711
f_{17}	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**
f_{18}	**3**	**3**	**3**	**3**	**3**	**3**

proposed algorithm able to reduce the computational time in ten benchmark functions (function $f_1, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}$ and f_{15}) is shown in Table 4 where the average time of DE and Evo-DE algorithms are taken from Jitkongchuen and Thammano [15], and all minimum computational times of each algorithm are shown in bold. Because the expansion of population from the reproduction steps from IWO algorithm, permutation and crossover methods from GA can provide an effective searching area. As a result, the algorithm spent a few computational time to be converged into the best solution.

The second part of the experiment intends to evaluate the performance of the proposed algorithm in comparison with other algorithms. The results of the comparative algorithms in Table 5 are taken from Brest et al. [16], Zhou et al. [17], and Jitkongchuen and Thammano [15]. In Table 5, the best results of each problem are presented in bold. The computational results in Table 5 show that the algorithm attains the global minimum of zero in function f_1, f_2, f_3, f_4, f_5, f_6, f_9 and f_{11}. The results are produced in the same value as the Self-adaptive differential evolution algorithm [15], while traditional DE, JDE and IMDE algorithms provide the global minimum of zero for some of these functions. For the functions f_{13}, and f_{15}–f_{18}, the proposed algorithm produces the same results with all of the other algorithms. More essentially, the results follow by * in Table 5 indicate that the proposed algorithm performs with better results than all of the other algorithms for the functions of f_7, f_{10} and f_{12}.

From the computational results, there is an indication that the proposed algorithm works efficiently especially for the continuous unimodal function, the quartic function, the multimodal function, and the discontinuous step function.

5 Conclusions

This paper presents a hybrid bat algorithm with natural-inspired algorithms for continuous optimization problem. In order to improve the performance of the bat algorithm, the searching area is expanded by the reproduction step from the IWO algorithm. To increase a larger coverage area of the search space, the mutation, crossover and selection from GA are applied to the population which is produced by the reproduction method of IWO. After that, each individual from the population will be calculated with the new position and velocity. The 'X_i^B best individuals' from this step will be selected and sent to the next step. All processes will be continued until the stopping condition is met, while the best solution returns. The experimental results claim that the proposed algorithm obtains the optimal and minimal solutions in all eighteen tested functions.

Acknowledgments The authors would like to acknowledge School of Information and Communication Technology, University of Phayao for all resources and highly grateful to Associate Professor Dr. Arit Thammano and Dr. Ravee Phoewhawm for the technical assistance.

References

1. Yang XS (2010) A new metaheuristic bat-inspired algorithm. In: Cruz C, Gonzales JR, Pelta DA, Terrazas G (eds) Nature inspired cooperative strategies for optimization (NISCO 2010) studies in computational intelligence, vol 284. Springer, Berlin, pp 65–74
2. Fister I Jr, Fister D, Yang XS (2013) A hybrid bat algorithm. Elektrotehniski vestnik 80(1–2):1–7
3. Yang XS, Gandomi AH (2012) Bat algorithm: a novel approach for global engineering optimization. Eng Comput 29(5):464–483
4. Tsai PW, Pan JS, Liao BY, Tsai MJ, Istanda V (2011) Bat algorithm inspired algorithm for solving numerical optimization problems. Appl Mech Mater 148–149:134–137
5. Ramesh B, Mohan VCJ, Reddy VCV (2013) Application of bat algorithm for combined economic load and emission dispatch. Int J Electr Eng Telecommun 2(1):1–9
6. Musikapun P, Pongcharoen P (2012) Solving multi-stage multimachine multiproduct scheduling problem using bat algorithm. In: 2nd international conference on management and artificial intelligence (IPEDR), vol 35. IACSIT Press, Singapore, pp 98–102
7. Mishra S, Shaw K, Mishra D (2012) A new meta-heuristic bat inspired classification approach for microarray data. Procedia Technol 4:802–806
8. Yılmaza Selim, Küçüksille EU (2015) A new modification approach on bat algorithm for solving optimization problems. Appl Soft Comput 28:259–275
9. Mehrabian AR, Lucas C (2006) A novel numerical optimization algorithm inspired from invasive weed colonization. Ecol Inf 1:355–366
10. Basak Aniruddha, Maity Dipankar, Das Swagatam (2015) A differential invasive weed optimization algorithm for improved global numerical optimization. Appl Math Comput 219(2013):6645–6668
11. Zhang X, Niu Y, Cui G, Wang Y (2010) A modified invasive weed optimization with crossover. In: Proceedings of the 8th world congress on intelligent control and automation, pp 11–14
12. Fister I Jr, Fister D, Yang XS (2013) A hybrid bat algorithm, CoRR abs/1303.6310
13. Parpinelli RS, Lopes HS (2011) New inspirations in swarm intelligence: a survey. Int J Bio Inspir Comput 3:1–16 **(View at Google Scholar)**
14. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102
15. Jitkongchuen Duangjai, Thammano Arit (2014) A self-adaptive differential evolution algorithm for continuous optimization problems. Artif Life Robot 19:201–208
16. Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 10(6):646–657
17. Zhou Y, Li X, Gao L (2013) A differential evolution algorithm with intersect mutation operator. Appl Soft Comput 13:390–401