Modified Bat Algorithm for Nonlinear Optimization

Kazimierz Kiełkowicz and Damian Grela,

Department of Computer Science, Faculty of Electrical and Computer Engineering,
Cracow University of Technology, Cracow, Poland

Summary
Bat Algorithm is recently proposed bio-inspired metaheuristics method for solving hard optimization tasks. It mimics behavior of bats hunting for their prey. The paper introduces some modification to Bat Algorithm. Presented modification changes exploration process as well as introduce different scheme of acceptance of a newly founded solutions. Effects of introduced modifications are tested on standard benchmark functions. The influence of a number of bats used in optimization process is also taken into account.

Key words:
Bat algorithm, swarm intelligence, metaheuristics, optimization.

1. Introduction

In general metaheuristics algorithms can be divided into few groups, e.g. algorithms based on evolutionary approach that models evolutionary process or algorithms exploring phenomena of a Swarm Intelligence [1]. Others approach for evolutionary metaheuristic, such as algorithms for modeling response of a human immune system (e.g. Artificial Immune System algorithms) might be considered as separate category due to their multiplicity of proposed solutions.
Metaheuristics methods which are focused on exploring models of a natural evolution are (mostly but not limited to) as follows: Genetic Algorithms (GA) [2], Genetic Programming (GP) and Differential Evolution (DE) [3]. Algorithms based on Swarm Intelligence are broadly presented by Particle Swarm Optimization (PSO) [4], Ant Colony Optimization (ACO) [5] or some of its modifications.
Recently introduced method, based on population of solutions which explore phenomena of Swarm Intelligence was presented by Yang [6] in 2010, is called Bat Algorithm (BA). In [6] by modeling the behavior of bats hunting for prey and by exploring phenomena of their echolocation capabilities, author managed to incorporate methods for balancing the exploration phase as well as exploitation phase of a modern Swarm Based Algorithms.
Bat Algorithms had already been applied to solve numerous hard optimization problems such as multi-criteria optimization [7] or optimization of topology of microelectronic circuits [8].
Growing popularity of the Bat Algorithm has encouraged researchers to focus their work on its further improvements. Most work has been done within the area of hybridization of Bat Algorithm with other metaheuristics or local search methods [9]. Some other solutions were involved within the area of adding self-adaptability capabilities to algorithm [10]. Some works has also been in adaptation standard Bat Algorithm for binary problems [11].
Unfortunately, most of these modification not only improves the quality of obtained solutions, but also increases the number of control parameters that are needed to be set in order to obtain solutions of expected quality. This makes such solutions quite impractical.
This paper introduces some modifications to Bat Algorithms that results in increment of converge capabilities without the need for any new control parameters.
Paper is organized as follows, in Section 2 basic scheme of the Bat Algorithm is introduced as well as some custom modifications are introduced and briefly discussed, Section 3 covers simulation experiments showing benefits of the proposed modifications, Section 4 summarize presented publication and discuss some concluding remarks.

2. Bat Algorithm

Bat Algorithm is recently proposed bio-inspired metaheuristics method for solving hard optimization tasks. It tries to mimic behavior of bats hunting for their prey. Algorithm was introduced by Yang in 2010 [6]. Bat Algorithm is based on population of bats, which by flying thru solution search space explore it in order to find interesting areas. Each single bat represents one solution in n-dimensional search space. Solutions are evaluated in terms of their fit value by provided fit function.
For example, consider n-dimensional, real valued solution space in which optimization takes place. Each solution, represented as a bat, is evaluated with provided fit function. There are also two real valued n-dimensional vectors associated with each bat in population. First vector is real valued vector representing position of a bat in solution search space. Second vector is real valued vector representing velocity in each of n-dimensional directions. Usually position vector and velocity vector are initialized randomly at the beginning of the algorithm. Main loop of the algorithm consists of iterative improvement in founded

Manuscript received October 5, 2016
Manuscript revised October 20, 2016

solution. At each iteration step fit value is calculated for every member of population of bats by provided fit function, and new velocity vector is calculated based on relative distance from best and current solution in population. Next, position of every bat is updated accordingly to its velocity vector. At the end of each iteration best solution is founded and used as new reference point. Exploring search space continues until some termination conditions are satisfied. Usually these conditions are the maximum number of iterations or improvements in the best solution. As a result, after satisfied stop conditions, the best solution is returned. Pseudo code for Bat Algorithm is listed in Fig.1.

1:	Randomly initialize position x_i and velocity v_i of i-th bat in population
2:	Initialize pulsation frequency $Q_i \in [Q_{min}, Q_{max}]$, pulsation r_i and loudness A_i of i-th bat in population
3:	**while not** termination conditions are satisfied:
4:	**for_each** bat in population:
5:	$v_i(t) = v_i(t-1) + Q_i(x_i(t-1) - x^*)$ $x_i(t) = x_i(t-1) + v_i(t)$
6:	**if rand(0,1)** $> r_i^t$: Generate new solution around current bests solutions
7:	Generate new solution by flying randomly
8:	**if rand(0,1)** $< A_i^t$ **and** $f(x_i) < f(x^*)$: Accept new solution and update pulsation and loudness factors r_i^t and A_i^t as: $A_i^{t+1} \leftarrow \alpha A_i^t;\ r_i^{t+1} \leftarrow r_i^t(1 - \exp(-\gamma t))$
9:	Evaluate bats population using fit function f
10:	Find best bat in population and mark him as x^*

Fig. 1 Bat Algorithm.

where:

$v_i(t)$ - real valued velocity vector of i-th bat,
$x_i(t)$ - real valued position vector of i-th bat,
Q_i - pulsation frequency of i-th bat,
$\alpha, \gamma, Q_{min}, Q_{max}$ - constant.

Equations used for bat position and velocity update, used in algorithm 1 step 5, were introduced in [6]

2.1 Proposed modification to Bat Algorithm

An important aspect of a population-based metaheuristic is balance between exploration and exploitation phase of a search process. Exploration (sometimes called diversification) is responsible for global search capability. While, in contrast, exploitation (sometimes called intensification) response for local search ability of algorithm. As was pointed out in [12] Bat algorithm is powerful at exploitation but has some insufficiency at exploration phase. In our opinion Bat Algorithm also suffer from lack of memory of best solution found during the time of optimization which cause it Bats sometimes tend to escape from promising area of solutions search space. Bat Algorithm also tends to direct bats outside of the solution search space box. Yang in [6] proposed to use upper bound limits on position vector to overcome these limitations. Bat Algorithm also too often tends to accept solution of worse fit value.

In literature few modifications to Bat Algorithm has been proposed. In [13] Inertia Weight Factor Modification relative to current iteration and max iteration and Adaptive Frequency Modification based on relative bat distance to best solution has been introduced. In [14] dynamic and adaptively adjustment of a bat speed and flight direction has been examined. Self-adaptive capability has also been examined in [10].

Bat Algorithm has also been hybridized with Harmony Search Algorithm [12] or with Differential Evaluation Algorithm [9] In [15] Bat Algorithm with self-adaptation of control parameters has been hybridized with different DE strategies as local search heuristics. However there are no systematic solutions to previously mentioned problem hence proposed modification.

Our modifications to Bat Algorithm are introduced in two places: scheme of acceptance of a new solution is modified and velocity equation is modified to overcome mention limitation. Introduced modifications are summarized in pseudo code listing in Fig.2. There is also introduced memory of best solution found during the process of optimization by the algorithm.

1:	Randomly initialize position x_i and velocity v_i of i-th bat in population
2:	Initialize pulsation frequency $Q_i \in [Q_{min}, Q_{max}]$, pulsation r_i and loudness A_i of i-th bat in population
3:	**while not** termination conditions are satisfied:
4:	**for_each** bat in population:
5:	$v_i(t) = \alpha_i v_i(t-1) + Q_i\big(x^* - x_i(t-1)\big) +$ $+ Q_i(x^*_{ever} - x_i(t-1))$ $x_i(t) = x_i(t-1) + v_i(t)$
6:	**if rand(0,1)** $> r_i^t$: $x_i' \leftarrow$ generate new solution around current bat x_i
7:	**if** $f(x_i') < f(x_i)$ **or rand(0,1)** $< A_i^t$: $x_i \leftarrow x_i'$ Update values of pulsation and loudness, respectively r_i^t and A_i^t as: $A_i^{t+1} \leftarrow \alpha A_i^t;\ r_i^{t+1} \leftarrow r_i^t(1 - \exp(-\gamma t))$
8:	Evaluate bats population using fit function f
9:	Find best bat in population and mark him as x^*
10:	**if** $f(x^*) < f(x^*_{ever})$:
11:	$x^*_{ever} \leftarrow x^*$

Fig. 2 Modification of Bat Algorithm.

Our solution modifies bat position and velocity equations. In comparison with equations presented in [6], author use archive component to help direct bats towards area where good solutions were used to be known and also

incorporates some concept of cognition coefficients instead of using upper bounds limits. Equations (1) and (2) shows proposed modification.

$$v_i(t) = \alpha_i v_i(t-1) + Q_i\big(x^* - x_i(t-1)\big) + \\ + Q_i(x^*_{ever} - x_i(t-1)) \quad (1)$$

$$x_i(t) = x_i(t-1) + v_i(t) \quad (2)$$

where:

α_i - cognition coefficient of i-th bat
$x^* - x_i(t-1)$ - social component
$x^*_{ever} - x_i(t-1)$ - archive component
Q_i - pulsation frequency of i-th bat

In comparison to equations proposed by Yang in [6] we can see that modified velocity equation is using cognition coefficients to limit the influence of past direction (taken at time t-1) at the decision taken at current t iteration. There is also archive component that helps bats build social knowledge of the previously, globally found best solution.

Proposed modification to the scheme of acceptance of new solutions are tend to limit the probability of acceptance of worse solution. Comparing original with our modification the worse solution in original approach is accepted with probability A_i where in modified algorithm worse solution is accepted only with probability $(1 - r_i)A_i$. There is obvious relation that, satisfying that $r_i > 0$ and $A_i > 0$, the fallowing relation is true $(1 - r_i)A_i < A_i$. Our modification also includes new form of memory x^*_{ever} of a best solution ever found.

What is important, our introduced modification doesn't change computation complexity of the algorithm in the context of big O notation since proposed modification are linear in nature and are not based on additional computation or evaluation of a fitness function.

3. Simulation experiments

Our modifications has been tested on set of standard and well known in literature benchmark functions. Simulation experiments were performed on a PC computer running Linux on Intel Core i7 2.20GHz with 8GB of RAM. Algorithms has been implemented in C++ and compiled with GCC. Section 3.1 briefly introduce used test functions, Section 3.2 reports obtained results. Simulation were re-run 10 times for every test function, mean and standard deviation are reported. Each of test functions was performed using 10, 20 and 30 dimensions' variants.

3.1 Test Functions

Presented algorithm has been tested on three well known and wildly accepted test function for continues optimization. Used functions was: Sphere, Rastrigin and Rosenbrock [15]. In every equation D will stand for dimension of the function and $\vec{x}$ is real valued vector in search space, $\vec{x} \in \mathcal{R}^D$.

First function was standard test function called Sphere Function. It is convex, unimodal simple test function for metaheuristics (equ. 3), with global solution at the point $\vec{x} = (0,0,\dots,0)$

$$f_{Sphere}(\vec{x}) = \sum_{i=1}^{D} x_i^2 \quad (3)$$

Second function was Rastrigin's Function. It is based on Sphere function (equ.3) by adding sinusoidal modulation we obtain Rastrigin function (equ. 4). It is multimodal non-linear function with global minimum at point $\vec{x} = (0,0,\dots,0)$

$$f_{Rastrigin}(\vec{x}) = 10D + \sum_{i=1}^{D}(x_i^2 - 10\cos(2\pi x_i) \quad (4)$$

Last was Rosenbrock's function (equ. 5) which has its global solution at point $\vec{x} = (0,0,\dots,0)$. Rosenbrock's solution is located in wide parabolic shaped valley. This makes it very complicated point to reach by evolutionary methods. Rosenbrock function is unimodal for D=2,3 while it's multimodal for more dimensions [16]

$$f_{Rosenbrock}(\vec{x}) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \quad (5)$$

3.2 Experiments

Simulation experiments were based on continues optimization of real valued non-linear functions. Optimization task was to find minimum of a test function. Each function was optimized in three dimensionality variation, for D=10, D=20 and D=30. Each test was rerun 10 times and mean solution and standard deviation were reported. Founded solution as well as standard deviation were presented in Tables 1, 2 and 3 for Sphere, Rastrigin and Rosenbrock test function respectively.

Optimized function was limited in D dimensional cube with limitation {-15, 15} on each side. Each test were run for n bats number in population, where $n \in \{10, 20, 50, 100\}$.

Each bat was represented by D-dimension real valued vector $x \in \mathcal{R}^D$. Initial population were generated randomly, loudness and pulsation parameters where set to $A_i = 0.1$, $r_i = 0.9$. Termination conditions were set to 1000 iterations. Method that was used to generate new solution were "random walk" around current solution. Best solutions were underlined.

Table 1: Mean solution found from 10 tries for Sphere function.

	n	***D=10***	***D=20***	***D=30***
Mean value	10	$1800 \cdot 10^{-6}$	$25200 \cdot 10^{-6}$	$1054 \cdot 10^{-4}$
Stand. dev.		$900 \cdot 10^{-6}$	$9370 \cdot 10^{-6}$	$346.8 \cdot 10^{-4}$
Mean value	20	$230 \cdot 10^{-6}$	$5540 \cdot 10^{-6}$	$270 \cdot 10^{-4}$
Stand. dev.		$150 \cdot 10^{-6}$	$1710 \cdot 10^{-6}$	$78.5 \cdot 10^{-4}$
Mean value	50	$0.137 \cdot 10^{-6}$	$353 \cdot 10^{-6}$	$33.3 \cdot 10^{-4}$
Stand. dev.		$6.76 \cdot 10^{-6}$	$127 \cdot 10^{-6}$	$11.1 \cdot 10^{-4}$

Mean value	100	$1.57 \cdot 10^{-6}$	$38.6 \cdot 10^{-6}$	$3.77 \cdot 10^{-4}$
Stand. dev.		$51.80 \cdot 10^{-6}$	$2.8 \cdot 10^{-6}$	$0.91 \cdot 10^{-4}$

Table 2: Mean solution found from 10 tries for Rastrigin function.

	n	***D=10***	***D=20***	***D=30***
Mean value	10	26.021	93.403	231.479
Stand. dev.		9.138	27.073	39.442
Mean value	20	20.188	63.064	112.660
Stand. dev.		10.520	16.898	32.394
Mean value	50	15.329	35.630	73.273
Stand. dev.		5.205	13.781	26.509
Mean value	100	13.035	29.878	59.406
Stand. dev.		5.791	9.729	15.712

Table 3: Mean solution found from 10 tries for Rosenbrock function.

	n	***D=10***	***D=20***	***D=30***
Mean value	10	15.130	92.707	202.422
Stand. dev.		20.321	111.359	216.167
Mean value	20	8.112	62.930	87.279
Stand. dev.		1.639	62.619	66.909
Mean value	50	8.583	29.889	81.823
Stand. dev.		0.938	22.738	57.414
Mean value	100	7.782	17.791	37.268
Stand. dev.		1.163	1.743	15.339

According to Tables 1, 2 and 3, increasing the number of bats used in optimization process results in improvement of quality of found solution and, at the same time, in deceasing of standard deviations. As it can be seen in Tables 1 thru 3 with increasing dimensionality of the search problem, performance of proposed algorithm decrease. The reason for that is as the dimensionality of the optimized function increase search space exponentially increases as well. Second with increasing dimensionality of some optimization problems we might cause change in characteristic of these objective problems. For instance, Rosenbrock function [17], which is unimodal in 2 or 3 dimension, may include multiple local minima for higher number of dimensions [16].

By comparing results reported of modified Bat Algorithm with results reported for standard Bat Algorithm [9] (using the same optimization task and the same test function) proposed modification has positive effect on quality founded by algorithm. In [9] Bat Algorithm was also hybridized with DE. The results for Hybrid BA were similar to results for our modified Bat Algorithm without any hybridizations. Hybridization of modified Bat Algorithm might be interesting path for further research.

The obtained results of modified Bat Algorithm are better that results reported for PSO algorithm in [18] for the same optimization task and the same test function.

4. Conclusions

This paper shows the possibility of modification of Bat Algorithm [6]. Original Bat Algorithm is powerful at exploitation but has some insufficiency at exploration phase. It also tends to direct bats outside of solution search space box. Another important issue is that it too often tends to accept solution of worse fit value. In literature few modifications to Bat Algorithm been proposed to tackle mentioned insufficiency [13][14][10]. Unfortunately there are no systematic solutions to these problem, hence our proposed modification.

Our modification to Bat Algorithm are two fold in nature. First: scheme of acceptance of new solution as well as velocity equation are modified. Acceptance scheme is modified in order to reduce probability of acceptance of worse solution. Velocity update equation is modified by introducing cognitive coefficient and archive component. Last one is form of additional memory which stores best solution ever found during the optimization process.

Comparing original [6] and modified Bat Algorithm it can be seen that in original version worse solution is accepted with probability A_i, where in proposed modification worse solution might be accepted with probability $(1 - r_i)A_i$. There is obvious relation that, satisfying that $r_i > 0$ and $A_i > 0$, the fallowing relation is true $(1 - r_i)A_i < A_i$. Proposed modification also includes new form of memory x^*_{ever} of best solution ever found.

Those modifications were tested in few simulation experiments. Experiments were performed for continues non-linear optimizations problem: minimize real valued test functions. Three well know test functions were used: Sphere, Rastrigin and Rosenbrock [15]. According to numerical experiments proposed modification improves quality of founded solutions.

Reported results of modified Bat Algorithm are close to these reported for Hybridized Bat Algorithm with DE in [4] but without any additional parameters for configuring algorithm. Moreover, our modification doesn't change computation complexity of the algorithm in the context of big O notation since proposed modification are linear in nature and are not based on additional computation or evaluation of a fitness function.

References

[1] R. C. Eberhart, Y, Shi, "Empirical Study of Particle Swarm Optimization",1999.

[2] D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning", Kluwer Academic Publishers, 1989.

[3] R. Storn, K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces", Technical Report, 1995.

[4] J. Kennedy, R. C. Eberhart, "Particle swarm optimization", In Proc. of IEEE Int. Conf. on Neural Networks, vol. 4, pp. 1942–1948, 1995

[5] M. Dorigo, V. Maziezzo, A. Colorni, "The ant system: optimization by a colony of cooperating ants", IEEE Trans. on Systems, Man and Cybernetics B, vol. 26, no. 1, pp. 29–41, 1996.

[6] X. S. Yang, “A new metaheuristic bat-inspired algorithm”, Nature Inspired Cooperative Strategies for Optimization, 2010.
[7] X. S. Yang, “Bat algorithm for multi-objective optimization”, International Journal of Bio-Inspired Computation, 2011.
[8] S. Fong, X. S. Yang, M. Karamanglu, “Bat Algorithm for Topology Optimization in Microelectronic Application”, IEEE, 2012.
[9] I. Fister Jr, D. Fister, X. S. Yang, “A Hybrid Bat algorithm”, Elektrotehniski Vestnik, 2013.
[10] A. Baziar, A. A. Kavoosi-Fard, J. Zare, “A novel Self Adoptive Modification Approach Based on Bat Algorithm for Optimal Management of Renewable MG”, Journal of Intelligent Learning System and Application, 2013.
[11] S. Mirjalili, S. M. Mirjalili, Xin-She Yang, “Binary Bat Algorithm”, Neural Comput & Applic, 2014.
[12] G. Wang and L. Guo, “A Novel Hybrid Bat Algorithm with Harmony Search for Global Numerical Optimization”, Journal of Applied Mathematics, vol. 2013, pp. 21, 2013.
[13] S. Yilmaz and E. U. Kucuksille, “Improved Bat Algorithm (IBA) on Continuous Optimization Problems”, Lecture Notes on Software Engineering, Vol. 1, No. 3, August 2013.
[14] X. Wang, W. Wang, Y. Wang, “An Adaptive Bat Algorithm”, Lecture Notes in Computer Science vol. 7996, 216-223, 2013.
[15] I. Fister Jr., S. Fong, J. Brest, and I. Fister, “A Novel Hybrid Self-Adaptive Bat Algorithm”, The Scientific World Journal, Volume 2014.
[16] W. Gao, S. Liu, “A modified artificial bee colony algorithm”, Computers and Operations Research, pp. 687–697, vol. 39, 2012
[17] Y. W. Shang and Y. H. Qiu, “A note on the extended Rosenbrock function”, Evolutionary Computation, vol. 14, pp. 119-126, 2006.
[18] S. Hossen, F. Rabbi, M. Rahman, “Adaptive Particle Swarm Optimization (APSO) for multimodal function optimization”, International Journal of Engineering and Technology, 2009.