Modified Seeker Optimization Algorithm for Unconstrained Optimization Problems

Ivona BRAJEVIC
Faculty of Mathematics
University of Belgrade
Studentski trg 16
SERBIA
ivona.brajevic@googlemail.com

Milan TUBA
Faculty of Computer Science
Megatrend University Belgrade
Bulevar umetnosti 29
SERBIA
tuba@ieee.org

Abstract: - Seeker optimization algorithm (SOA) is a novel search algorithm based on simulating the act of human searching, which has been shown to be a promising candidate among search algorithms for unconstrained function optimization. In this article we propose a modified seeker optimization algorithm. In order to enhance the performance of SOA, our proposed approach uses two search equations for producing new population and employs modified inter-subpopulation learning phase of algorithm. This modified algorithm has been implemented and tested on fourteen multimodal benchmark functions and proved to be better on majority of tested problems.

Key-Words: - Unconstrained optimization, Seeker optimization algorithm, Artificial bee colony, Nature inspired heuristics

1 Introduction

Optimization problems have the objective to find minimum or maximum of the function under consideration. Unconstrained optimization problems can be formulated as a *D*-dimensional minimization or maximization problem:

$$\min \text{ (or max) } f(x),\ x = (x_1,...,x_D) \in R^D \qquad (1)$$

where D is the number of the parameters to be optimized.

There are many population based metaheuristics applied to unconstrained optimization problems. Genetic algorithms (GA) [1], particle swarm optimization (PSO) [2], artificial bee colony (ABC) algorithm [3], differential evolution (DE) [4] and recently proposed seeker optimization algorithm (SOA) [5] are among the most popular metaheuristics which employ a population of individuals to solve the problem. The success of a population based method depends on its ability to produce proper balance between exploration and exploitation. The exploitation refers to the ability of algorithm to apply the knowledge of the previous good solutions to better guide the search towards improving regions of the search space. The exploration refers to the ability of algorithm to investigate the unknown regions in the search space to discover the global optimum. A poor balance between exploration and exploitation may result in a weak optimization method which may suffer from premature convergence, i.e. the algorithm may be trapped in a local optimum.

In order to enhance the performance of SOA, we propose a modified seeker optimization algorithm named MSO, which uses two search equations for producing new population and employs modified inter-subpopulation learning phase of algorithm. In each iteration of the proposed algorithm a new possible solution is produced by search equation of artificial bee colony (ABC) algorithm or search equation of seeker optimization algorithm (SOA). The ABC algorithm developed by Karaboga and its modified versions were successfully applied to unconstrained and constrained optimization problems [3], [6], [7], [8]. Seeker optimization algorithm was analyzed with a challenging set of benchmark problems for function optimization, where its performance was compared with the performance of DE and three versions of PSO algorithms. SOA has shown better global search ability and faster convergence speed for the most chosen benchmark problems, especially for unimodal benchmarks. For multimodal test functions the results were not very satisfactory because it was noticed that for this type of problems

This research is supported by Ministry of Science, Republic of Serbia, Project No. 44006

algorithm may be stuck at a local optimum. Since its invention, SOA has been applied to solve the other kinds of problems beside numerical function optimization. In [9], the application of the SOA to tuning the structures and parameters of artificial neural networks is presented, while in [10] SOA-based evolutionary method is proposed for digital IIR filter design. Also, a new optimized model of proton exchange membrane fuel cell (PEMFC) was proposed by using SOA [11]. In this work, our MSO algorithm was applied to multimodal test functions and its performance was compared with the performance of SOA. Modified seeker optimization algorithm showed better performance than SOA for the majority of tested problems.

This paper is organized as follows. Section 2 presents the seeker optimization algorithm. Section 3 describes our proposed approach. Section 4 describes benchmark functions. Section 5 presents the experimental setup adopted and test results. Our conclusion is provided in section 6.

2 Seeker optimization algorithm

Seeker optimization algorithm (SOA) mimics the behavior of human search population based on their memory, experience, uncertainty reasoning and communication with each other. SOA is a population-based heuristic algorithm. The algorithm operates on a set of solutions called search population. Each individual of the population is called a seeker or agent. The total population is categorized into three subpopulations according to the indexes of the seekers. All the seekers in the same subpopulation constitute a neighborhood which represents the social component for the social sharing of information. Each seeker *i* has the following attributes: the current position $x_i(t) = [x_{i1}, x_{i2}, ..., x_{ij}, ..., x_{iD}]$, where j is the j^{th} dimension, D is the dimension size and t is the number of iteration, the personal best position $p_{i,best}$ so far, and the neighborhood best position g_{best} so far. The main characteristic features of this algorithm are the following:

1. The algorithm uses search direction and step length to update the positions of seekers
2. The calculation of the search direction is based on a compromise among egotistic behavior, altruistic behavior and pro-activeness behavior
3. Fuzzy reasoning is used to generate the step length because the uncertain reasoning of human searching could be the best described by natural linguistic variables and a simple if-else control rule: If {objective function value is small} (i.e., condition part), then {step length is short} (i.e., action part)

A search direction $d_{ij}(t)$ and a step length $\alpha_{ij}(t)$ are separately computed for each individual i on each dimension j at each iteration t, where $\alpha_{ij}(t) \geq 0$ and $d_{ij}(t) \in \{-1,0,1\}$.

2.1 Calculation of the search direction

In swarm dynamic there are two extreme types of cooperative behavior: egotistic which is entirely pro-self and altruistic which is entirely pro-group. Every seeker is uniformly egotistic if he believes that he should go toward his personal best position $p_{i,best}$ through cognitive learning. The egotistic behavior of each seeker *i* may be modeled by vector called egotistic direction $d_{i,ego}$ by:

$$d_{i,ego} = p_{i,best} - x_i(t) \tag{2}$$

In altruistic behavior, seekers want to communicate with each other and adjust their behaviors in response to the other seekers in the same neighborhood region for achieving the desired goal. This attitude of each seeker *i* may be modeled by vector called altruistic direction $d_{i,alt}$ by:

$$d_{i,alt} = g_{best} - x_i(t) \tag{3}$$

where *gbest* represents the neighborhood best position so far.

Beside of egoistic and altruistic behavior, the future behavior can be predicted and guided by the past behavior. Due to this, the seeker may be pro-active to change its search direction and exhibit goal-directed behavior according to his past behavior. The pro-active behavior of each seeker *i* may be modeled by vector called pro-activeness direction $d_{i,pro}$ by:

$$d_{i,pro} = x_i(t_1) - x_i(t_2) \tag{4}$$

where $t_1, t_2 \in \{t, t-1, t-2\}$, $x_i(t_1)$ and $x_i(t_2)$ are the best and the worst positions in the set $\{x_i(t-2), x_i(t-1), x_i(t)\}$ respectively.

The expression of search direction for the i^{th} seeker is set to the stochastic combination of egotistic direction, altruistic direction and pro-activeness direction by:

$$d_i(t) = sign(w \cdot d_{i,pro} + \varphi_1 \cdot d_{i,ego} + \varphi_2 \cdot d_{i,alt}) \tag{5}$$

ISBN: 978-1-61804-064-0

where the function sign $(\cdot)$ is a signum function on each dimension of the input vector, ω is the inertia weight and φ_1 and φ_2 are real numbers chosen uniformly and randomly in the range [0,1]. Inertia weight is used to gradually reduce the local search effect of pro-activeness direction $d_{i,pro}$ and provide a balance between global and local exploration and exploitation. Inertia weight is linearly decreased from 0.9 to 0.1 during a run.

2.2 Calculation of the step length

From the view point of human searching behavior, it is understood that one may find the near-optimal solutions in a narrower neighborhood of the point with lower objective function value and on the other hand, in a wider neighborhood of the point with higher objective function value. To design a fuzzy system to be applicable to a wide range of optimization problems, the objective function values of each subpopulation are sorted in descending order and turned into the sequence numbers from 1 to *S* as the inputs of fuzzy reasoning, where *S* denotes the size of the subpopulation to which the seekers belong. The expression is presented as:

$$\mu_i = \mu_{\max} - \frac{S - I_i}{S - 1} \cdot (\mu_{\max} - \mu_{\min}) \quad (6)$$

where I_i is the sequence number of $x_i(t)$ after sorting the objective function values in descending order, $\mu_{\max}$ is the maximum membership degree value which is equal to or a little less than 1.0.

Bell membership function $f(x) = e^{-x^2/2\delta^2}$ is well utilized in the literature to represent the action part of the control rule. Because the membership function value μ_{ij} of α_{ij} beyond $[-3\delta, 3\delta]$ are less than 0.0111, a minimum membership degree $\mu_{\min}$ = 0.0111 is set. Moreover, the parameter δ of the Bell membership function is determined by :

$$\delta_i = \omega \cdot abs(x_{\min} - x_{avg}) \quad (7)$$

In Eq. (7), the absolute value of the input vector as the corresponding output vector is represented by the symbol abs $(\cdot)$, $x_{\min}$ is the position of the best seeker in the subpopulation to which the i^{th} seeker belongs, and x_{avg} is the averaged position of the seekers in the same subpopulation. The inertia weight is used to decrease the step length with increasing time step, which improves the search precision. In order to introduce the randomness in each variable and to improve the local search capability, the following equation is introduced to convert μ_i into a vector with elements as given by:

$$\mu_{ij} = rand(\mu_i, 1), \ j = 1,2,...D \quad (8)$$

The action part of the fuzzy reasoning gives the step length for each seeker *i* by :

$$\alpha_{ij} = \delta_{ij} \cdot \sqrt{-\ln(\mu_{ij})}, \ j = 1,2,...D \quad (9)$$

2.3 Implementation of seeker optimization algorithm

At each iteration the position of each seeker is updated by:

$$x_{ij}(t+1) = x_{ij}(t) + \alpha_{ij}(t) \cdot d_{ij}(t) \quad (10)$$

where $i = 1,2,...SN$; $j = 1,2,...D$ (*SN* is the number of seekers)

Also, at each iteration, the current positions of the worst two individuals of each subpopulation are exchanged for both of the best one in each of the other two subpopulations, which is called inter-subpopulation learning. Short pseudo–code of the SOA algorithm is given below:

1. Generating *s* positions uniformly and randomly in search space;
2. cycle = 0;
3. **Repeat**
4. For i = 1 to s do
   Computing $d_i(t)$ by Eqs. (2), (3), (4) and (5);
   Computing $\alpha_i(t)$ by Eqs. (6), (7), (8) and (9);
   Updating each seeker's position using Eq. (10);
5. End of For
6. Evaluating all the seekers and saving the historical best position;
7. Implementing the inter-subpopulation learning operation;
8. cycle = cycle+1;
9. **Unti**l the termination criterion is satisfied

3 Our proposed approach: MSO

In order to enhance the performance of SOA, MSO algorithm uses two search equations for producing new population: search equation of ABC algorithm and the search equation of seeker optimization algorithm. Also, MSO algorithm implements the modified inter-subpopulation learning using the binomial crossover operator as in [12]. At the first step, MSO algorithm generates a randomly

distributed initial population of *SN* solutions, where *SN* denotes the number of seekers (solutions). Each solution X_i *(i = 1, 2, ..., SN)* is a *D*-dimensional vector and *D* is the number of optimization parameters. Total population is divided into *K* subpopulations according to the indexes of the seekers. After initialization, the population of the solutions is subjected to repeated cycles of the search processes. Any iteration of MSO algorithm can be described as following:

1. Perform an update process for each solution in the search population using randomly selected search equation. MSO chooses between search equation (10) which is used in SOA and the variant of ABC search equation which can be described as:

$$v_{ij} = \begin{cases} x_{ij} + \varphi_i (x_{ij} - x_{kj}), if \;\; R_j \le 0.5 \\ x_{ij} \qquad\qquad\qquad , \quad otherwise \end{cases} \qquad (11)$$

where k is randomly chosen index of solution from the subpopulation to which the *i*th seeker belongs, and *k* has to be different from *i*, $j = 1,2,...,D$ and φ_i is a random number between [-1, 1). The MSO included a new control parameter which is called behavior rate (*BR*) in order to select the search equation in the following way: If a random number between [0, 1) is less then *BR* the SOA search equation is used, otherwise Eq.(11) is performed.

2. Evaluating all the seekers and saving the historical best position.

3. The modified inter-subpopulation learning is implemented as follows: The positions of seekers with the highest objective function values of each subpopulation *l* are combined with the positions of seekers with the lowest objective function values of *(l+t) mod K* subpopulations respectively, where *t=1,2,.. NSC. NSC* denotes the number of the worst seekers of each population which are combined with the best seekers. The appropriate seekers are combined using the following binomial crossover operator as expressed in:

$$x_{l_n j,worst} = \begin{cases} x_{i_j best} \quad , if \quad R_j \le 0.5 \\ x_{l_n j,worst} \quad , \quad otherwise \end{cases} \qquad (12)$$

In Eq. (12), R_j is a uniformly random real number within [0, 1), $x_{l_n j,worst}$ is denoted as the j^{th} variable of the n^{th} worst position in the l^{th} subpopulation, $x_{i_j best}$ is the j^{th} variable of the best position in the i^{th} subpopulation. Additionally, we included a new parameter which we named inter-subpopulation learning increase period (*ILIP*). After *ILIP* iterations the number of the worst seekers of each sub-populations which are combined with the best seekers is increased to *2* NSC*.

We can conclude that in MSO algorithm we have three new control parameters in comparison with the original SOA: the behavior rate (*BR*), the number of seekers of each subpopulation for combination (*NSC*) and the inter-subpopulation learning increase period (*ILIP*). *BR* parameter is used to control which of the search equations for producing new population will be used. In MSO algorithm as in SOA, each subpopulation is searching for the optimal solution using its own information and hence the subpopulation may trap into local optima yielding a premature convergence. Hence the inter-subpopulation learning, which ensures that good information acquired by each subpopulation is shared among the subpopulations, is included in the algorithm. But, in the inter-subpopulation learning of SOA it was noticed that it may not always bring the benefits for multimodal functions since it may attract all agents toward a local optimal solution. In MSO algorithm we increased the diversity of population by using the crossover operator for producing new solutions. In order to provide better balance between exploitation and exploration abilities of algorithm, the diversity of population is additionally increased by *NSC* and *ILIP* parameters, which increase the number of seekers for combining.

4 Benchmark functions

Fourteen benchmark functions from [5] were used to test the performance of our MSO algorithm. The characteristics of benchmark functions are given in Table 1. *D* denotes the dimensionality of the test problem, *S* denotes the ranges of the variables, and $f_{\min}$ is a function value of the global optimum.

Functions f_1 - f_6 (Schwefel, Rastrigin, Ackley, Griewank, Penalized - two functions) are multimodal functions where the number of local minima increases exponentially with the problem dimension. Functions f_7 - f_{14} (Shekel foxholes, Kowalik, Six-hump Camel-Back , Branin, Goldstein-Price, Hartman family - two functions, Shekel) are low-dimensional functions which have only a few local minima. For multimodal functions, the final results are much more important than the convergence rates, since they reflect the algorithm's ability to escape from poor local optima and locate a good near-global optimum.

	D	S	f_{min}
f1	30	$[-500, 500]^D$	-12 569.5
f2	30	$[-5.12, 5.12]^D$	0
f3	30	$[-32, 32]^D$	0
f4	30	$[-600, 600]^D$	0
f5	30	$[-50, 50]^D$	0
f6	30	$[-50, 50]^D$	0
f7	2	$[-65.54, 65.54]^D$	0.998
f8	4	$[-5, 5]^D$	3.075E-4
f9	2	$[-5, 5]^D$	-1.0316
f10	2	$[-5, 15]^D$	0.398
f11	2	$[-2, 2]^D$	3
f12	3	$[0, 1]^D$	-3.86
f13	6	$[0, 1]^D$	-3.32
f14	4	$[0, 10]^D$	-10.54

Table 1 Characteristics of benchmark functions

5 Parameter settings, results and discussion

We applied our modified seeker optimization (MSO) algorithm to the set of benchmark multimodal optimization problems. The proposed algorithm has been implemented in Java programming language. Tests for fourteen benchmark problems were done on a Intel(R) Core(TM) 2 Duo CPU E8500@4-GHz personal computer with 4 GB RAM memory.

The obtained results are compared to the results obtained by seeker optimization algorithm (SOA) and its results are taken from [5]. In all experiments for the both algorithms the same size of population of 100 seekers is used and the algorithms were repeated 50 runs. The maximum number of generations (G) for each tested problem is the same for the both algorithms and their values are listed in Table 2. In our proposed approach the number of subpopulations (*SubpopNum*) is taken 10 and the values of new control parameters are: the behavior rate (*BR)* is 0.4, the number of seekers of each subpopulation for combination (*NSC*) is taken 0.2* *SP/SubpopNum* and the inter-subpopulation learning increase period (*ILIP*) is taken 0.4**G*.

The comparative results of the best, mean values and standard deviations over 50 independent runs of the SOA and MSO algorithm are summarized in Table 2. The dimensions of multimodal functions with many local minima f_1 - f_6 were all set to 30 for both algorithms. As it can be seen from Table 2, MSO algorithm achieved better results for functions f_1, f_5 and f_6, while for functions f_2 and f_3, SOA performed better than MSO algorithm. For function f_4, SOA has better mean value and the same best value as MSO algorithm. For functions f_7 - f_{14}, the number of local minima and the dimension are small.

Function	Stats	SOA	MSO
f1 (G = 9 000)	Best	-11760	-12333
	Mean	-10126	**-11539**
	St. dev	669.5	376.0
f2 (G = 5 000)	Best	0	5.97E-00
	Mean	**0**	1.27E+01
	St. dev	0	0.32E+01
f3 (G = 1 500)	Best	-4.44E-15	7.99E-15
	Mean	**-4.44E-15**	3.49E-14
	St. dev	0	4.12E-14
f4 (G = 2 000)	Best	0	0.0
	Mean	**0**	1.11E-16
	St. dev	0	2.22E-17
f5 (G = 1 500)	Best	3.04E-04	1.67E-31
	Mean	1.28E-02	**4.15E-03**
	St. dev	7.62E-03	2.03E-02
f6 (G = 1 500)	Best	2.77E-03	2.22E-31
	Mean	1.89E-01	**2.46E-02**
	St. dev	1.30E-01	1.13E-01
f7 (G = 100)	Best	0.998	0.998
	Mean	1.199	**1.058**
	St. dev	5.30E-01	3.08E-01
f8 (G = 4 000)	Best	3.0749E-04	3.0749E-04
	Mean	**3.0749E-04**	3.6243E-04
	St. dev	1.58E-09	2.17E-04
f9 (G = 100)	Best	-1.0316	-1.0316
	Mean	**-1.0316**	**-1.0316**
	St. dev	6.73E-06	3.38E-16
f10 (G = 100)	Best	0.39789	0.39789
	Mean	0.39838	**0.39789**
	St. dev	5.14-04	2.21E-14
f11 (G = 100)	Best	3	3
	Mean	3.000 1	**3**
	St. dev	1.17E-04	2.13E-15
f12 (G = 100)	Best	-3.862 8	-3.862 8
	Mean	-3.862 1	**-3.862 8**
	St. dev	6.69E-04	1.09E-16
f13 (G = 200)	Best	-3.321	-3.322
	Mean	**-3.298**	-3.279
	St. dev	0.045	5.71E-02
f14 (G = 100)	Best	-10.54	-10.54
	Mean	- 9.72	**-10.54**
	St. dev	4.72E-01	5.50E-09

Table 2 Comparison of SOA and MSO algorithm

The results from Table 3 showed that SOA is outperformed by MSO algorithm for functions f_7,

f_{10}, f_{11}, f_{12} and f_{14}, while SOA achieved better results for functions f_8 and f_{13}. Also, MSO algorithm is not statistically different from SOA for function f_9.

In [5] it was noticed that for multimodal functions SOA may be stuck at a local optimum. From Table 2 it can be seen that the performance of MSO algorithm is better for the majority of multimodal benchmarks, half of high-dimensional multimodal functions and for the most of low-dimensional multimodal functions.

6 Conclusion

Seeker optimization algorithm has been tested for a challenging set of benchmark problems for function optimization. The simulation results showed that the proposed algorithm is competitive and a promising candidate among swarm algorithms for numerical function optimization. In this paper we presented the modified seeker optimization (MSO) algorithm for unconstrained function optimization. In order to avoid the algorithm to trap at some local attractors MSO algorithm uses two search equations for producing new population and binomial crossover operator in the inter-subpopulation learning phase of algorithm. The experimental results tested on fourteen multimodal benchmark functions show that the proposed approach improved the performance of SOA for majority of tested functions.

References:

[1] J. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, 1992

[2] Kennedy, J., Eberhart, R.C., Particle swarm optimization, *Proceedings of the 1995 IEEE International Conference on Neural Networks*, Piscataway, NJ: IEEE Service Center, 1995, pp. 1942–1948

[3] Karaboga D., Akay B., A comparative study of Artificial Bee Colony algorithm, *Applied Mathematics and Computation*, 2009, Volume 214, Issue 1, pp. 108-132

[4] K.V. Price, R.M. Storn, J.A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, Berlin, Germany, 2005

[5] Dai C., Chen W., Song Y., Zhu Y., Seeker optimization algorithm: a novel stochastic search algorithm for global numerical optimization, *Journal of Systems Engineering and Electronics*, Vol. 21, No. 2, 2010, pp. 300-311

[6] Zhu G, Kwong S., Gbest-guided artificial bee colony algorithm for numerical function optimization, *Applied Mathematics and Computation*, Volume 217, Issue 7, 2010, pp. 3166-3173

[7] Stanarevic N., Tuba M., Bacanin N., Modified artificial bee colony algorithm for constrained problems optimization, *International Journal of Mathematical Models and Methods in Applied Sciences*, Volume 5, Issue 3, 2011, pp. 644-651

[8] Brajevic I., Tuba M., Subotic M., Performance of the improved artificial bee colony algorithm on standard engineering constrained problems, *International Journal of Mathematics and Computers in Simulation*, Volume 5, Issue 2, 2011, pp. 135-143

[9] Dai C., Chen W., Zhu Y., Jiang Z. , You Z. , Seeker optimization algorithm for tuning the structure and parameters of neural networks, *Neurocomputing*, Volume 74, Issue 6 , 2011, pp. 876-883

[10] Dai C., Chen W., Zhu Y., Seeker optimization algorithm for digital IIR filter design, *IEEE Transactions on Industrial Electronics*, Volume 57, Issue 5, 2010, pp.1710-1718

[11] Dai C., Chen W., Cheng Z., Li Q., Jiang Z., Jia J.. Seeker optimization algorithm for global optimization: A case study on optimal modelling of proton exchange membrane fuel cell (PEMFC), *International Journal of Electrical Power and Energy Systems*, 2011, Volume 33, Issue 3, pp. 369-376

[12] B. Shaw, S. Ghoshal, V. Mukherjee, S. P. Ghoshal , Solution of Economic Load Dispatch Problems by a Novel Seeker Optimization Algorithm, *International Journal on Electrical Engineering and Informatics*, Vol. 3, No. 1, 2011, pp. 26-42