Available online at www.sciencedirect.com
SCIENCE DIRECT®
Applied Mathematics and Computation 176 (2006) 308–316
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Ant colony optimization for finding the global minimum

M. Duran Toksari

Erciyes University, Engineering Faculty, Industrial Engineering Department, 38039 Kayseri, Turkey

Abstract

The ant colony optimization (ACO) algorithms are multi-agent systems in which the behaviour of each ant is inspired by the foraging behaviour of real ants to solve optimization problem. This paper presents the ACO based algorithm to find global minimum. Algorithm is based on that each ant searches only around the best solution of the previous iteration. This algorithm was experimented on test problems, and successful results were obtained. The algorithm was compared with other methods which had been experimented on the same test problems, and observed to be better.

Keywords: Ant colony optimization; Global minimum; Metaheuristics; Random optimization

1. Introduction

Many heuristic methods such as random search technique (ARSET) [1], heuristic random optimization (HRO) [2] and David–Fletcher method were developed to find global minimum. In this paper, ACO based algorithm will be suggested to find global minimum. ACO belong to class of biologically inspired heuristics. The basic idea of ACO is to imitate the cooperative behaviour of ant colonies.

The function should have many local minimum points, but only one of them is the global minimum. If $F(x_{\min}) \leqslant F(x)$ for all x values, $x_{\min}$ value is defined as the point makes the function minimum. If $F(x)$ is continuous and differentiable, the minimum value can be found on the point $\frac{dF}{dx}$. However, wherever the function is not differentiable, it could prove more advantageous to utilize stochastic methods instead of deterministic ones [1].

The paper is organized as follows. First, ACO will be explained shortly. The proposed ACO based algorithm to find global minimum will be detailed in Section 3. In Section 4, the algorithm will be solved on five benchmark problems. Finally, the proposed algorithm will be compared with other heuristic methods.

2. Ant colony optimization (ACO)

The idea of imitating the behaviour of ants for finding good solutions to combinatorial optimization problems was initiated by Dorigo [3]. The principle of these methods is based on the way ants search for food and

E-mail address: dtoksari@erciyes.edu.tr

0096-3003/$ - see front matter
doi:10.1016/j.amc.2005.09.043

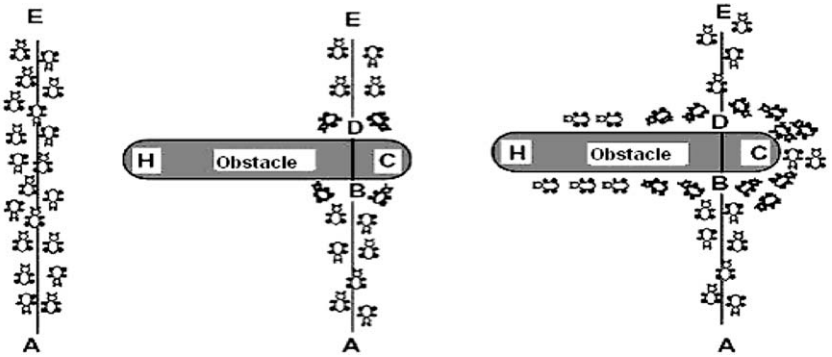

Fig. 1. Ants finding the shortest path around an obstacle.

Step 1. Initialization
- Initialize pheromone trail

Step 2. Solution construction
- For each ant Repeat
  Solution construction using the pheromone trail

Step 3. Update the pheromone trail
  Until stopping criteria

Fig. 2. A generic ACO algorithm.

find their way back to the nest. During trips of ants a chemical trail called pheromone is left on the ground. The role of pheromone is to guide the other ants towards the target point. For one ant, the path is chosen according to the quantity of pheromone.

As illustrated in Fig. 1, when facing an obstacle, there is an equal probability for every ant to choose the left or right path. As the left trail is shorter than the left one and so required less travel time, it will end up with higher level of pheromone. More the ants will take the right path, higher the pheromone trail is. This fact will be increased by the evaporation stage.

The general ACO algorithm is illustrated in Fig. 2. The procedure of the ACO algorithm manages the scheduling of three activities [4,5]: The first step consists mainly in the initialization of the pheromone trail. In the iteration (second) step, each ant constructs a complete solution to the problem according to a probabilistic state transition rule. The state transition rule depends mainly on the state of the pheromone. The third step updates quantity of pheromone; a global pheromone updating rule is applied in two phases. First, an evaporation phase where a fraction of the pheromone evaporates, and then a reinforcement phase where each ant deposits an amount of pheromone which is proportional to the fitness of its solution. This process is iterated until a stopping criterion.

3. Ant colony optimization to find global minimum

The ACO algorithm has been used to find global minimum. In the proposed algorithm, first, number of m ants being associated with m random initial vectors (x^k_{initial}, $(k = 1, 2, \ldots, m)$) (or all of them may be set to the same value) (Fig. 3a).

Then, modifications based on the pheromone trail are then applied to each vector. In the proposed ant colony based algorithm, quantity of pheromone (τ_t) only intensifies around the best objective function value obtained from the previous iteration and all ants turned towards there to search a solution (Fig. 3b). The solution vector of each ant is updated at the beginning of each iteration using the following formula:

$$x^k_t = x^{\text{best}}_{t-1} \pm \mathrm{d}x \quad (t = 1, 2, \ldots, I), \tag{3.1}$$

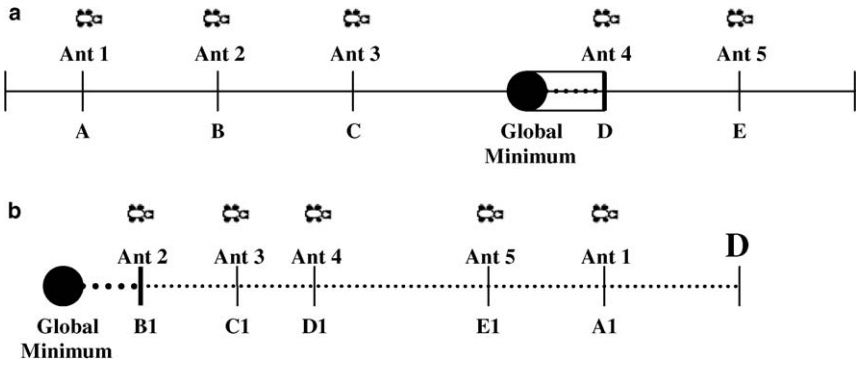

Fig. 3. (a) Number of 5 ants being associated with 5 random initial vectors (the best vector is on the D point. So, quantity of pheromone only intensifies between global minimum point and D point). (b) The first iteration. (Searching was only done between global minimum point and D point. At the end of the iteration 1, the best vector is on the B1 point. So, quantity of pheromone only intensifies between global minimum point and B1 point.)

where x_t^k is solution vector of the kth ant at iteration t, x_{t-1}^{best} is the best solution obtained at the iteration $t-1$ and dx is a vector generated randomly from $[-\alpha, \alpha]$ range to determine the length of jump. At the end of each iteration, quantity of pheromone (τ_t) is updated. First, quantity of pheromone (τ_t) is reduced to simulate the evaporation process with the following formula:

$$\tau_t = 0.1 \times \tau_{t-1}. \tag{3.2}$$

Initialization

Set the initial values:

Stopping criterion: maximum iteration $n \times \sqrt{I}$,

The interval dx vector is to be generated $\alpha_{initial}$,

Generate m random initial vectors $\left(x_{initial}^{k}, \quad (k = 1,2,...,m)\right)$ (or all of them may be set to the same value), each one associated to an ant and assign this as current solution $\left(f(x)_{initial}^{k}\right)$,

Determine $x_{initial}^{best}$,

Define the direction of movement by equation (3.4).

Solution Manipulation

FOR $k = 1$ TO n

FOR $i = 1$ TO $k \times \sqrt{I}$

FOR all ants

Generate dx random vector within $[-\alpha, \alpha]$ range,

Calculate new solution of each ant by equation (3.1),

IF $f\left(x_t^{best}\right) \le f\left(x_{t-1}^{best}\right)$ THEN $x^{global\min} = x_t^{best}$

ELSE $x^{global\min} = x_{t-1}^{best}$

Update of the Number of Pheromone

by equations (3.2) and (3.3).

END

END

$\alpha_k = 0.1 \times \alpha_{k-1}$

END

Fig. 4. Ant colony optimization for the global minimum.

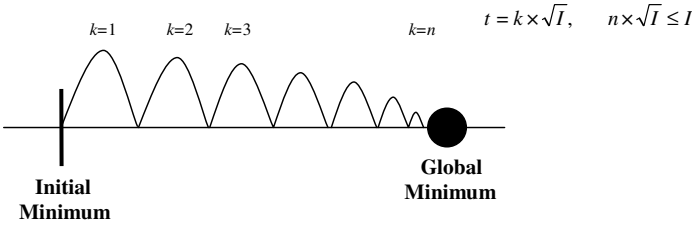

Fig. 5. Jumping movements.

Then, it is only increased around the best objective function value obtained from the previous iteration.

$$\tau_t = \tau_{t-1} + \left(0.01 \times f\left(^{\text{best}}_{t-1}\right)\right). \tag{3.3}$$

This process is iterated until number of maximum iteration (I).

In formula (3.1), (+) sign uses when point x_t^k is on the left of global minimum on the x-coordinate axis. On the other hand, (−) sign uses when point x_t^k is on the right of global minimum on the x-coordinate axis. The direction of movement is defined by Eq. (3.4)

$$\bar{x}^{\text{best}}_{\text{initial}} = x^{\text{best}}_{\text{initial}} + (x^{\text{best}}_{\text{initial}} \times 0.01). \tag{3.4}$$

If $f(\bar{x}^{\text{best}}_{\text{initial}}) \leqslant f(x^{\text{best}}_{\text{initial}})$, (+) sign is used into (3.1). Otherwise, (−) sign is used. (±) sign defines direction of movement that appears after initial solution.

Steps of the proposed algorithm are as Fig. 4.

Setting $\alpha = 0.1 \times \alpha$ at the end of each iteration $\sqrt{I}$ to not pass over global minimum (I is number of maximum iteration). Thus, the length of jumping will gradually decrease. It would be explained in Fig. 5.

4. Benchmark problems

In this section, performance of the proposed ACO based algorithm is examined on five test problems. Studied problems in fact are the ones studied by numerous authors before. Solutions obtained by using the algorithm are compared with previous solutions obtained for these problems.

4.1. Benchmark problem 1

The result obtained by using proposed ACO based algorithm for a function which is one of the problems solved with ARSET [1] and HRO [2]. Objective function of the problem is given in Eq. (4.1), and Fig. 6 shows the graph of function

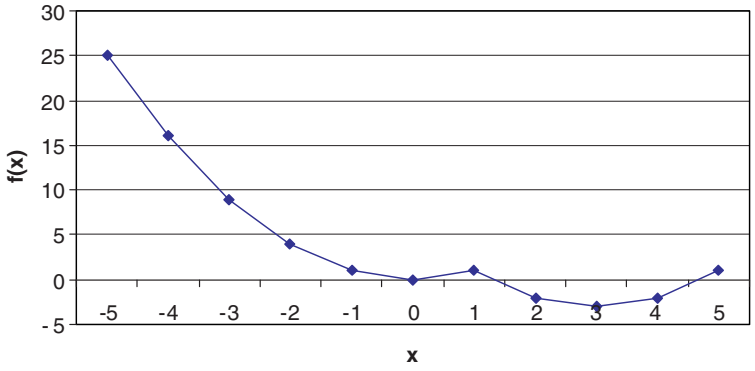

Fig. 6. Objective function of benchmark problem 1.

Table 1
Results obtained by proposed algorithm for benchmark problem 1

	Best x	Best $f(x)$
Epoch number = 500	3	−3

Table 2
Comparison between the proposed ACO based algorithm and HRO and ARSET algorithms for benchmark problem 1

Algorithms	Best x	Best $f(x)$	Epoch number
HRO	3.000324	−2.9999998	1000
ARSET	**3**	**−3**	1000
Proposed ACO based algorithm	**3**	**−3**	**500**

$$\min f(x) = \begin{cases} x^2, & \text{if } x \leqslant 1, \\ (x-3)^2 - 3, & \text{if } x > 1. \end{cases} \tag{4.1}$$

Function has two minimums, one being on $x = 0$ and the other one on $x = 3$. The point on $x = 0$ denotes the local minimum, while the point on $x = 3$ denotes the global minimum.

The proposed ACO based algorithm is initiated with the initial solutions $x^k_{\text{initial}} = 0.5$, $\alpha = \frac{1}{\text{random}(10)}$, $m = 5$ and $I = 100$, and run 10 times. Epoch number is $m \times I = 500$. Best solution obtained is shown in Table 1.

As the best value of x, HRO and ARSET algorithms are obtained $x = 3.000324$ and $x = 3$ with epochs 1000, respectively. Best value of x obtained by using the proposed ACO based algorithm is $x = 3$ with epochs 500. The comparison between HRO, ARSET and proposed ACO based algorithm is shown in Table 2. It is clear that the proposed ACO based algorithm outperforms both HRO and ARSET.

4.2. Benchmark problem 2

Benchmark problem is taken from Li and Rhinehart [2]. The result obtained by using proposed ACO based algorithm for a function which is one of the problems solved with ARSET and HRO. Objective function of the problem is given in Eq. (4.2), and Fig. 7 shows the graph of function

$$\left[x \times \sin\left(\frac{1}{x}\right)\right]^4 + \left[x \times \cos\left(\frac{1}{x}\right)\right]^4. \tag{4.2}$$

It is clear that the minimum value of the function is on $x = 0$.

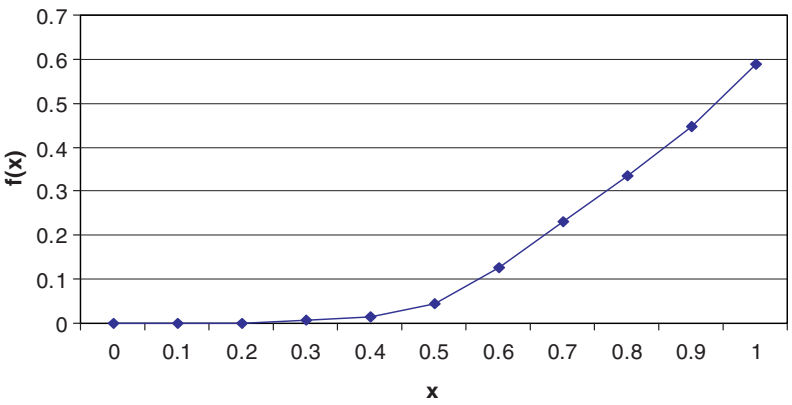

Fig. 7. Objective function of benchmark problem 2.

Table 3
Comparison between the proposed ACO based algorithm and ARSET algorithms for benchmark problem 2

Algorithms	Best x	Best $f(x)$	Epoch number
ARSET	1.90E−006	6.58E−024	10000
Proposed ACO based algorithm	**3.36E−10**	**8.16E−039**	
ARSET	4.39E−008	1.85E−030	30000
Proposed ACO based algorithm	**−1.57E−11**	**5.47E−044**	
ARSET	−2.53E−011	2.21E−043	50000
Proposed ACO based algorithm	**7.79E−012**	**1.40E−045**	

The proposed ACO based algorithm is initiated with the initial solutions, $m = 10$, 30 and 50 and $I = 1000$, and run 10 times, taking the initial values as $\alpha = \frac{1}{\text{random}(10)}$ and $x^k_{\text{initial}} = 1$. Epoch numbers are $m \times I = 10000$, 30000 and 50000, respectively.

As the best value of x, HRO algorithm is obtained $x = 0.000024$. In Table 3, best values of x obtained by using the proposed ACO based algorithm are compared with ARSET. Table 3 shows obviously that the proposed ACO based algorithm outperforms both HRO and ARSET.

4.3. Benchmark problem 3

A function with two variables is shown in Eq. (4.3) as the third benchmark problem [6]

$$f(x,y) = \frac{(x-3)^8}{1+(x-3)^8} + \frac{(y-3)^4}{1+(y-3)^4}. \tag{4.3}$$

Function has a minimum that is $[x \quad y]^{\mathrm{T}} = [3 \quad 3]^{\mathrm{T}}$. The point on $[x \quad y]^{\mathrm{T}} = [3 \quad 3]^{\mathrm{T}}$ denotes the global minimum, and $f(x, y) = 0$ is on it. Fig. 8 shows the graph of function.

Applying David–Fletcher method, Schilling and Harris state that the best outcome of the problem is $[x \quad y]^{\mathrm{T}} = [3.0196706 \quad 3.0004380]^{\mathrm{T}}$.

ARSET algorithm is run 10 times for with various combinations of N and E. The best values obtained by using the proposed ACO based algorithm are compared with ARSET in Table 4.

The proposed ACO based algorithm is initiated with the initial solutions, $m = 10$, 30 and 50 and $I = 1000$, and run 10 times, taking the initial values as $\alpha = \frac{1}{\text{random}(10)}$ and $x^k_{\text{initial}} = [0 \quad 0]$. Epoch numbers are $m \times I =$ 10000, 30000 and 50000, respectively. The proposed algorithm found global minimum point.

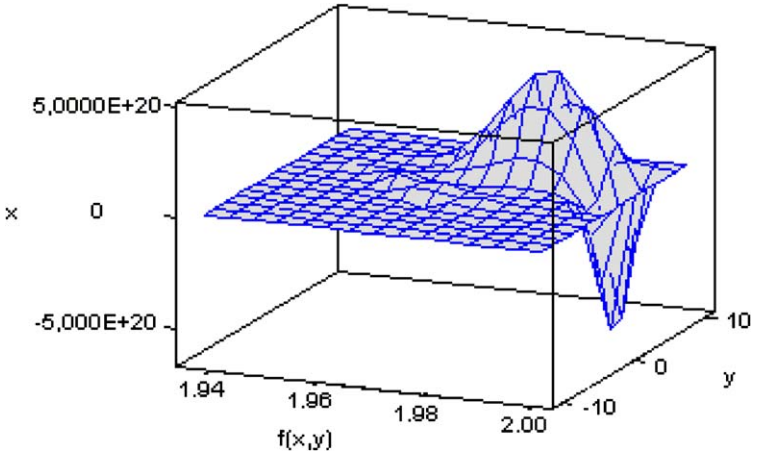

Fig. 8. Objective function of benchmark problem 3.

Table 4
Comparison between the proposed ACO based algorithm and ARSET algorithms for benchmark problem 3

Algorithms	Best x	Best y	Best $f(x, y)$	Epoch number
ARSET	3.0157	2.9999	3.71E−015	10000
Proposed ACO based algorithm	**3 × 2066E−09**	**3 × 2384E−09**	**2.62E−021**	
ARSET	3.0072	3	7.32E−018	30000
Proposed ACO based algorithm	**3**	**3**	**0**	
ARSET	3.0015	3	5.04E−023	50000
Proposed ACO based algorithm	**3**	**3**	**0**	

4.4. Benchmark problem 4

Rosenbrock function, also known as Rosenbrock's Banana Function, is taken as the fourth benchmark problem [7]. The minimum points of the function are $x = 1$, $y = \pm 1$ and $f(x, y) = 0$ within the range $[0 \quad 6]$. The graph of function generates shown in Fig. 9. Objective function of the problem is given follows:

$$f(x,y) = \left(100 \times (x - y^2)^2\right) + (1 - x)^2. \tag{4.4}$$

ARSET algorithm is run with various combinations of N and E [1]. The best values obtained by using the proposed ACO based algorithm are compared with ARSET in Table 5, taking the initial values as $\alpha = \frac{1}{\text{random}(10)}$ and $x^k_{\text{initial}} = [-1.9 \quad 2]$.

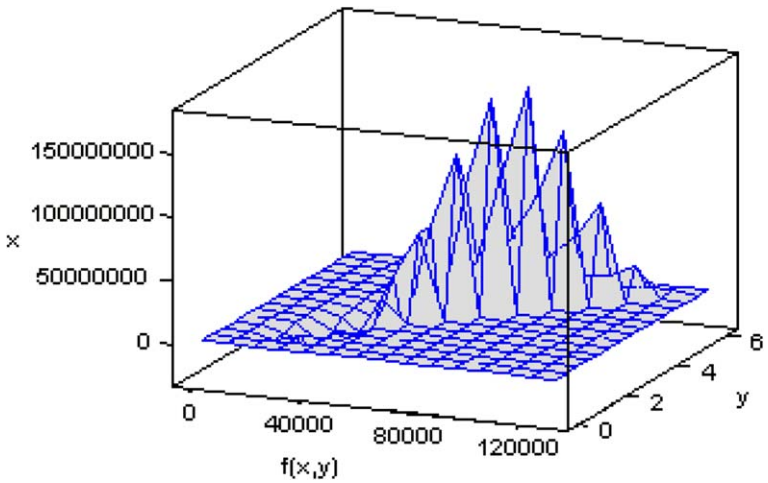

Fig. 9. Objective function of benchmark problem 4.

Table 5
Comparison between the proposed ACO based algorithm and ARSET algorithms for benchmark problem 4

Algorithms	Best x	Best y	Best $f(x, y)$	Epoch number
ARSET	0.99401	0.997	3.58E−005	10000
Proposed ACO based algorithm	**1.00021**	**1.00004**	**1.73E−006**	
ARSET	1.0001	1.0001	2.03E−008	30000
Proposed ACO based algorithm	**1**	**1**	**5.68E−12**	
ARSET	1	1	4.02E−16	50000
Proposed ACO based algorithm	**1**	**1**	**0**	

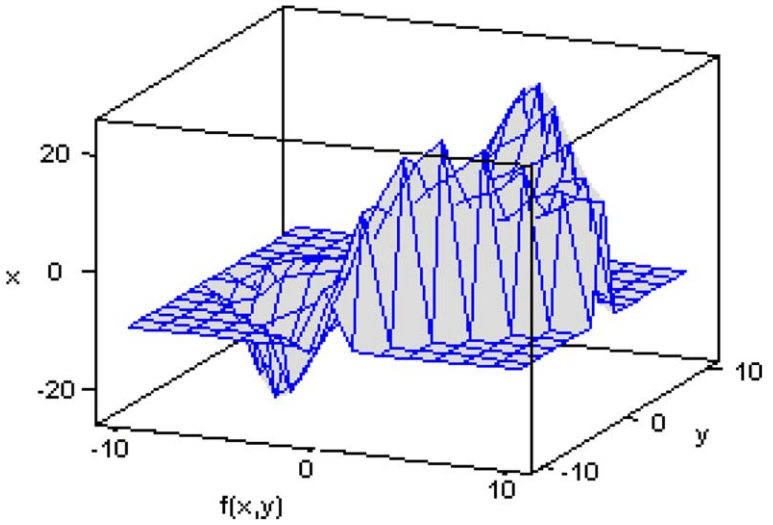

Fig. 10. Objective function of benchmark problem 5.

Table 6
Comparison between the proposed ACO based algorithm and ARSET algorithms for benchmark problem 5 within the range $[-10 \quad 10]$

Algorithms	Best x	Best y	Best $f(x, y)$	Epoch number
ARSET	−9.9968	3.46E−009	−9.9968	10000
Proposed ACO based algorithm	**−9.9989**	**2.01E−004**	**−9.9989**	
ARSET	−9.9996	−2.08E−018	−9.9996	30000
Proposed ACO based algorithm	**−9.9999**	**−6.05E−008**	**−9.9999**	
ARSET	−10	6.67E−008	−10	50000
Proposed ACO based algorithm	**−10**	**8.07E−011**	**−10**	

Both ARSET algorithm and the proposed ACO based algorithm find the global minimum that is point $[x \quad y]^{\mathrm{T}} = [1 \quad 1]^{\mathrm{T}}$.

4.5. Benchmark problem 5

An indifferentiable function is studied as the fifth benchmark problem. It is given in Eq. (4.5). Graph within the range $[-10 \quad 10]$ is shown in Fig. 10. Results of the proposed ACO based algorithm are compared with ARSET in Table 6, taking the initial values as $\alpha = \frac{1}{\text{random}(10)}$ and $x^k_{\text{initial}} = [9 \quad 9]$.

$$f(x,y) = \frac{x}{1+|y|}. \tag{4.5}$$

5. Conclusions and future work

In this paper, a powerful and robust algorithm which is based on ant colonies, is proposed to find global minimum. When compared with previous methods to find global minimum such as ARSET, HRO and David–Fletcher method, results shows that proposed algorithm has a noticeable performance. The proposed ACO based algorithm found exactly global minimum of four benchmark problems, so, it is obvious that the best values obtained by using the proposed ACO based algorithm are the best found values for tested benchmark problems.

Future works include an application of the proposed ACO based algorithm to find global maximum and using hybrid approaches of popular metaheuristics such as genetic algorithm, tabu search, simulated annealing, neural networks, etc. to find global minimum faster.

References

[1] C. Hamzacebi, F. Kutay, A heuristic approach for finding the global minimum: adaptive random search technique, Applied Mathematics and Computation (2005).
[2] J. Li, R.R. Rhinehart, Heuristic random optimization, Computers and Chemical Engineering 22 (3) (1998) 427–444.
[3] M. Dorigo, Optimization, learning and natural algorithms, Ph.D. Thesis, Politecnico di Milano, Italy, 1992.
[4] M. Dorigo, G. Di Caro, Ant colony optimization: a new meta-heuristic, in: Proceeding of the 1999 Congress on Evolutionary Computation, vol. 2, 1999, pp. 1470–1477.
[5] E.G. Talbi, O. Roux, C. Fonlupt, D. Robillard, Parallel ant colonies for the quadratic assignment problem, Future Generation Computer Systems 17 (2001) 441–449.
[6] R.J. Schilling, S.L. Harris, Applied Numerical Methods for Engineers, Brooks/Cole, Pacific Grove, 2000.
[7] R.L. Fox, Optimization Methods for Engineering Design, Addison-Wesley Publishing Company, California, 1971.