A Hybrid Approach for Solving Nonlinear Optimization Problems

Ramadan A. ZeinEldin[1], **Ayman M. Senosy**[2]

[1]Operations Research Department, ISSR, Cairo University, Egypt
Email: rzeanedean@yahoo.com
[2]Professional Academy for Teachers, Egypt
Email: ayman_ms8@hotmail.com

Abstract

In this paper, we propose a hybrid optimization approach for solving nonlinear optimization problems, using the ant colony optimization as an initial phase to generate initial solutions then they are improved by using particle swarm optimization. Twelve test functions with lower and upper bounds, eight problems are unconstrained optimization problems and four problems are constrained optimization problems are used to test the proposed approach. We got good results compared with other methods.

Keywords: Nonlinear, Evolutionary Algorithms, Ant colony, Particle swarm, Constrained, Unconstrained

{**Citation:** Ramadan A. ZeinEldin, Ayman M. Senosy. A hybrid approach for solving nonlinear optimization problems. American Journal of Research Communication, 2014, 2(2):7-19} www.usa-journals.com, ISSN: 2325-4076.

1. Introduction

In our real world a lot of problems occurring in applications involve either nonlinear objective functions or nonlinear constraints or both. Nonlinear optimization under nonlinear constraints is usually difficult and there is no a method best for solving all problems [2]. In fact there are different methods for solving nonlinear problems, and there is no one method best for solving all problems. But in recent years, family of stochastic search algorithms, known as evolutionary algorithms, which are inspired by biological evolution are introduced to overcome the drawbacks of classical methods. Evolutionary algorithms have many advantages that make these algorithms better in use than traditional nonlinear programming methods [14].

A nonlinear optimization problem in which the objective function is minimized is defined as the follows [1]:

$$\begin{aligned} \min \quad & f(x) && (1) \\ s.t. \quad & g_i(x) \le 0 \quad , \quad i = 1, \dots, m \\ & h_i(x) = 0 \quad , \quad i = 1, \dots, l \\ & x \in X \end{aligned}$$

Where $f, g_1, \dots, g_m, h_1, \dots, h_l$ are functions defined onR^n , X is a subset of R^n and x is vector of n components$x_1, \dots, x_n$.

Each of the constraints $g_i(x) \le 0$ for $i = 1, \dots, m$ is called inequality constraint, and each of the constrained $h_i(x) = 0$ for $i = 1, \dots, l$ is called an equality constraint. The set X include lower and upper bounds on the variables. A vector $x \in X$ satisfying all the constraints is called a feasible solution to the problem and the point $\bar{x}$ is called an optimal solution if $f(\bar{x}) \le f(x)$ for each feasible point x.

The nonlinear programming problem can be maximization problem. And in case of omitting the constraints, we have an unconstraint problem, and the whole domain of the objective function is feasible set.

In this paper, a hybrid approach is developed for solving nonlinear optimization problems using ant colony optimization as an initial step then the results are improved by using particle swarm optimization. The rest of this paper is organized as follows. Following this introduction, in section 2, a general view about nonlinear is presented. In section 3, the evolutionary algorithms; ant colony optimization and particle swarm optimization are introduced as one of metaheuristics methods. In section 4, the proposed approach is presented with its parameters. In section 5, a number of numerical experiments are performed. The conclusion and future work are discussed in section 6. Finally, we list the unconstrained test problems in appendix A and the constrained test problems in appendix B.

2. Nonlinear Problems

The Problems can be classified according to the nature of the objective function and constraints, the number of variables, the smoothness of the functions (differentiable or non-differentiable), and so on. Nonlinear problems (NLPs) are often difficult to solve and the most efficient algorithms provide no guarantees of success or fast performance over a range of applications [2]. We have two types of methods to solve NLPs; Mathematical methods which are based on calculus and geometry and the other methods are heuristic and metahuristic methods which are based on search heuristic such as genetic algorithms and simulated annealing [3]. The traditional mathematical methods for solving NLPs need a set of conditions to be verified in the problem before solving it such as differentiability and continuity [2]. For that many researches focused on heuristic and metahuristic methods in solving NLPs.

2.1. Unconstrained Problems

Unconstrained optimization problems (UOPs) arise directly in many practical applications, due to its importance, such as data fitting problems, engineering design and process control [4]. This type of problems minimizes or maximizes the function in absence of restriction. It can be formulated as:

$$\begin{aligned} &min \quad f(x) \qquad (2) \\ &s.t. \quad x \in R^n \end{aligned}$$

Where R^n is the n-dimensional space of real numbers. These problems can be distributed between small problems to large problems in the real-world. UOPs arise also as reformulations of constrained optimization problems, in which the constraints are replaced by penalization terms in the objective function that have the effect of discouraging constraint violations [1]. UOPs methods try to start the solution by guessing a solution, and then improve this solution to return many local minimum solutions. The challenge faces these methods is how to find the global minimum solution or at least a solution near to this global solution [4].

2.2. Constrained Problems

Constrained optimization problems (COPs) arise from models that include explicit constraints on the variables. When at least one constrained or the objective function is nonlinear, the problem becomes constrained nonlinear problem. These types of problems tend to arise naturally in the physical sciences and engineering, and are becoming more widely used in management and economic sciences. The problem can be formulated as equation (1) mentioned in section 1.

Solving a constrained optimization problem with inequality, upper bound, and lower bound constraints usually includes mathematical model building and algorithm designing. Many methods have been posed from Newton Method, Linear Search to Evolutionary Computation, which has respective characteristics. The constrained optimization methods often use unconstrained optimization as a step [5].

Sarimveis et al. [14] proposed a new methodology for solving constrained optimization problems. It is based on the formulation of an augmented Lagrange function, which is penalizes the violations of the equality and inequality constraints by including them in the objective function and multiplying them with appropriate penalty weights. The methodology is based on the line-up differential evolution algorithm which is proposed for solving unconstrained problem. The algorithm maintains a population of solutions which is continuously improved. They tested their methodology and claimed that it can be used as a robust and reliable approach for solving constrained optimization problems which are difficult to solve by the traditional optimization algorithms. He et al. [8] proposed a new filled function with one parameter for solving constrained global optimization problems, in which the filled function contains neither exponential term nor fractional term and is easy to be calculated. They perform some numerical experiments for some typical test problems and they reached that their new algorithm is feasible and effective. Yano et al. [18] proposed a hybrid algorithm to obtain more precise solutions. They introduced a neighborhood search algorithm to the standard particle swarm; they applied PSO to determine the minimum values. In the algorithm, when a better solution in the swarm is found, the neighborhood of a certain distance from the solution is searched. Then the algorithm returns to the original PSO search. Mingjun et al. [10] proposed an improved simulated annealing (ISA), a global optimization algorithm for solving the linear constrained optimization problems, of which characteristics are that only one component of current solution is changed based on the Gaussian distribution in each iteration and ISA can directly solve the linear constrained optimization problems. They compared their proposed approach with classical techniques and show that the proposed algorithm is the best one.

3. Evolutionary Algorithms

Evolutionary Algorithms (EAs), which are based on a powerful principle of evolution; survival of the fittest, and which model some natural phenomena. During the last two decades there has been a growing interest in algorithms which are based on the principle of evolution [12]. Classical methods usually are exhaustive in finding the best solution for small spaces but in large spaces special artificial intelligence techniques must employed. The methods of evolutionary computation are among such techniques. There are a few main paradigms of evolutionary computation techniques such as genetic algorithms. Many researchers try to modify evolutionary algorithms either by modifying the values of strategy parameters during the run of the algorithm, or by hybridize methods [9] [15].

In this paper there is hybridization is happened between two EAs; ant colony optimization and particle swarm optimization.

3.1 Ant Colony Optimization

Ant Colony Optimization Algorithm (ACO) is a meta-heuristic optimization method proposed by Dorigo et al. [7] for the solution of discrete combinatorial optimization problems such as the traveling salesman problem and the quadratic assignment problem. The method has been shown to outperform other general purpose optimization algorithms including genetic algorithms (GA) when applied to a number of benchmark combinatorial optimization problems. In the ACO algorithm a colony of artificial ants cooperate in finding good solutions to discrete optimization problems [17]. The foraging behavior of ant colonies has been intensively studied and the bionics corresponding to this behavior finds various applications. The indirect communication between ants is accomplished through pheromones. A pheromone is a chemical substance that triggers a natural response in another member of the same species. There are many types of pheromones deposited by ants, e.g., aggregation pheromones, alarm pheromones, trail pheromones, etc. Here we are interested in trail pheromones, which are deposited by ants with food as they return to their nest; they attract other ants and get them to go along the pheromone trail to find food. The famous bridge experiment shows that ants always can find the shortest path between the colony and food source [8].

The move probability distribution defined probability $P_{i\Psi}^{k}$ to be equal to 0 for all moves which are infeasible, otherwise they are computed by means of formula (3) where α and β are user defined parameters ($0 \leq \alpha, \beta \leq 1$)

$$P_{i\Psi}^{k} = \begin{cases} \dfrac{\alpha_{\tau\iota\psi} + \beta_{\eta\iota\psi}}{\sum_{(\iota\psi)\notin tabu_k}(\alpha_{\tau\iota\psi} + \beta_{\eta\iota\psi})} & if\ (\iota\psi) \notin tabu_k \\ 0 & otherwise \end{cases} \qquad (3)$$

In formula (3) $tabu_k$ is the tabu list of ant k, while parameters α and β specify the impact of trail and attractiveness, respectively. After each iteration t of the algorithm, i.e., when all ants have completed a solution, trails are updated by mean of formula (4)

$$\tau_{\iota\omega}(\tau) = p\, \tau_{\iota\omega}(\tau - 1) + \Delta\tau_{\iota\omega} \qquad (4)$$

Where $\Delta\tau_{\iota\omega}$ represents the sum of the contributions of all ants that used move $(\iota\omega)$ to construct their solution, $0 \leq p \leq 1$, is a user-defined parameter called evaporation coefficient. The ants' contributions are proportional to the quality of the solutions achieved, i.e., the better solution is the higher will be the trail contributions added to the moves it used.

3.2 Particle Swarm Optimization (PSO)

Many scientists have attempted to simulate bird flocking and fish schooling behaviors on computers for a long time. Kennedy and Eberhart suggested an optimization method called particle swarm optimization (PSO) [11]. PSO is derived from two main components; artificial life in swarm, and swarming theory. In PSO, individuals communicate and exchange information with other. The individuals change directions suddenly, scatter, and regroup. The velocity is used to describe the movement of birds. The particle swarm optimization has been found to be robust and fast in solving nonlinear problems [6]. The searching procedures are iterative, and the position and velocity are used to explore the optimum solution in the search space. At any time t, each particle knows its current position x_i^t and remember its personal best position x_{ipBest}^t . And every member in the swarm knows the global best position x_{Gbest}^t which represents the best particle in flock. The velocity v_i^{t+1} of particle i can be calculated according to the following equation:

$$v_i^{t+1} = wv_i^t + c_1 r_1 (x_{iPBest}^t - x_i^t) + c_2 r_2 (x_{GBest}^t - x_i^t) \qquad (5)$$

Where $c_1$and c_2 are constants called acceleration coefficients, the parameters r_1 and r_2 are two independent random numbers in the interval [0 , 1]. The parameter w is called inertial weight. Then, the position x_i^{t+1} of particle i is updated in each generation as shown below:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \qquad (6)$$

4. The Proposed Approach

In the proposed hybrid approach, there are two phases; ACO is started in phase 1 to generate a set of initial points. Then phase 2 is started to improve the results we got from ACO using PSO. We run the proposed approach 50 runs.

4.1 ACO Parameters

In ACO phase, the range of the function f in equation (1) is divided into number of parts (set to be 5 parts) to generate different points along the whole range. Different combined points are collected to form a number of paths. We set the number of ants to be 100 ants, and stopping criteria is where 50% of ants pass a certain path.

4.2 PSO Parameters

In PSO phase, the parameters of PSO set as follows: the number of iterations set to be 500, $W_{max} = 0.9$, $W_{min} = 0.4$, j = 1 , 2 , 3 , ... j_{max} which are used in the equation $w = \frac{(W_{max} - W_{min})}{j_{max}} * j$, Population size (N) set to be equal to the number of satisfied points, c1 and c2 are random numbers between [1 , 2]

4.3 The Proposed Approach steps

4.3.1 Initialization Step

Step 1	:	Define ACO parameters; number of ants , stopping criteria, number of accepted solutions, number of paths
Step 2	:	Divide the range of function into a number of parts, and define the intervals of each part.
Step 3	:	Generate points in each part of range.
Step 4	:	Combine different points from different parts to form different paths.
Step 5	:	Calculate the objective function for each path.
Step 6	:	Arrange the objective function.
Step 7	:	Give weights for each path according to the order of objective function.
Step 8	:	Calculate the probability, expected number of ants and the new trail of each path
Step 9	:	Start calculating next trails to define the best path. Stop when meeting stopping criteria.

Step 10 : Store the points

4.3.2 Main Step

Step 1 : Set the parameters of PSO.
Step 2 : If the problem has constraints select only the points satisfy the constraints, else select all points.
Step 3 : Calculate the objective functions.
Step 4 : Calculate the velocities, then generate new points
Step 5 : Check the feasibility of new points.
Step 6 : Repeat step 2 till stopping criteria.

5. Numerical Experiments

Eight unconstrained problems are listed in Appendix A, these problems are compared by other methods in papers [13] [14] [18] which are listed in table 2, and necessary calculations are performed in order to reach satisfied results which are listed in table 1. Four constrained problems are listed in Appendix B, and the results listed in table 1, we compared our results with those in papers [8] [14] which are listed in table 4. All the numerical experiments are compared from the aspects of global minimum value (best value), mean, global maximum (worst value), standard deviation (SD) and the elapsed time, in order to show the effectiveness of the proposed approach.

Table 1: The unconstrained problems results using the proposed approach

Test Prob.	Pro. 1	Pro. 2	Pro. 3	Pro. 4	Pro. 5	Pro. 6	Pro. 7	Pro. 8
Type	Min	Min	Min	Max	Min	Min	Min	Min
Dim	2	2	2	2	2	2	2	3
Optimal	3	0	-186.73	38.8503	-1.0316	–176.542	–176.1375	0
Best value	-1.388e+05	0	-186.731	38.8503	-1.0316	-176.542	-176.1373	1.4998e-032
Mean	-1.388e+05	24.4921	-186.7309	38.6583	-1.0316	-176.542	-159.8619	1.4998e-032
Worst value	-1.388e+05	48.9843	-186.7309	29.85123	-1.0316	-176.542	-90.8973	1.4998e-032
SD	0.484150	25.8170	7.7251e-014	1.271833	6.7290e-016	2.3675e-014	22.4703	1.3823e-047
Time	3.2850	1.1250	6.3520	9.480	2.9150	10.4550	5.8620	15.5310

Table 2: The results of the proposed approach compared with other methods

Test Prob.	Parsopoulos K. et al [13]	Sarimveis H. et al. [14]	Yano F. et al [18]	The Proposed
Prob. 1	3			-1.388e+05
Prob. 2	0			0
Prob. 3		- 186.73		-186.731
Prob. 4		38.8503		38.8503
Prob. 5			-1.0316	-1.0316
Prob. 6	–176.542			-176.542
Prob. 7	–176.1375			-176.1373
Prob. 8	0			1.4998e-032

In Table 1, eight unconstrained problems are solved. We Ran the algorithm 10 runs for each problem. In all the execution of the algorithm the population consists of 25 solutions, which is the solution of each path we got from phase1. Table 1 summarizes the obtained results including the best and worst solutions out of 50 executions, the average of solutions and standard deviations with respect to the average. The algorithm is compared with papers [13], [14] and [18] and show that the algorithm is good in finding the optimal solution and in some cases we got results better than the optimal such as in problem 1.

Table 3: The constrained problems results using the proposed approach

Prob.	Pro. 9	Pro. 10	Pro. 11	Pro. 12
Type	Min	Min	Min	Min
Dim	2	2	2	2
Best value	-0.97110407	-2.71828183	-5.50801	5.00000
Mean	-0.80163098	-1.76120782	-5.50801	5.00003
Worst value	0	4.49204612	-5.50800	5.00038
SD	0.35497602	1.67802185	1.6961178e-05	6.5799133e-005
Time	4.1340	1.6840	3.3220	2.6360

Table 4: The results of the proposed approach compared with other methods

Test Problem	He, S. et al [8]	Sarimveis H. et al. [14]	Wang, W et al [16]	Zhang, Y et al [19]	Zhang Y et al [20]	The proposed
Prob. 9	-0.97110407			-0.9711041		-0.97110407
Prob. 10	-2.71828183				-2.7183	-2.71828183
Prob. 11	-5.50801327	-5.50801	-5.5079			-5.50801
Prob. 12		5				5.00000

Table 3 summarized the results we have got after 10 runs. The algorithm is compared with papers [8], [14], [16], [19] and [20] shown in table 4. The optimal value is reached within 4 seconds in most cases.

6. Conclusion and future work

By applying the hybrid approach to the test optimization problems, the approach is examined and good results are obtained in comparing with others. Numerical results shows that the proposed approach is better with small scale problems, but with large scale problems (especially constrained problems) it takes a long time to reach a feasible solution. One can extend the work by introducing the idea of decreasing the time consumed in solving constrained large scale problem by converting it into unconstrained problems as first step using other heuristic algorithm, then decreasing the number of paths in searching the feasible solution.

7. Appendix A: (Unconstrained Problems)

Problem 1 (Goldstein-Price): [13]

$$\begin{aligned} min \quad & f(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 + 6x_1x_2 + 3x_2^2)][30 \\ & \qquad + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \\ s.t. \quad & -2 \leq x_i \leq 2 \qquad i = 1, 2 \end{aligned}$$

Problem 2 (Freudenstein-Roth): [13]

$$\min \quad f(x) = \left[-13 + x_1 + \left((5 - x_2)x_2 - 2\right)x_2\right]^2 + \left[-29 + x_1 + \left((x_2 + 1)x_2 - 14\right)x_2\right]^2$$

$$-5 \le x_1 \le 5$$
$$-5 \le x_2 \le 5$$

Problem 3 (Shubert): [14]

$$\min \quad f(x) = \sum_{i=1}^{5} i\cos\left[(i+1)x_1 + i\right] \sum_{i=1}^{5} i\cos\left[(i+1)x_2 + i\right]$$

$$s.t. \quad -10 \le x_1 \le 10$$
$$-10 \le x_2 \le 10$$

Problem 4: [14]

$$\min \quad f(x) = 21.5 + x_1\sin(4\pi x_1) + x_2\sin(20\pi x_2)$$

$$s.t. \quad -3.0 \le x_1 \le 21.1$$
$$4.1 \le x_2 \le 5.8$$

Problem 5 (Six hump camel back): [18]

$$\min \quad f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$$

$$s.t. \quad -3 \le x_1 \le 3$$
$$-2 \le x_2 \le 2$$

Problem 6 (Leavy No. 3): [13]

$$\min \quad f(x) = \sum_{i}^{5}\left[i\cos\left((i-1)x_1 + i\right)\right] \sum_{j}^{5}\left[j\cos\left((j+1)x_2 + j\right)\right]$$

$$s.t. \quad -10 \le x_1 \le 10$$
$$-10 \le x_2 \le 10$$

Problem 7 (Leavy No. 5): [13]

$$\min \quad f(x) = \sum_i^5\left[i\cos\left((i-1)x_1 + i\right)\right]\sum_j^5\left[j\cos\left((j+1)x_2 + j\right)\right] + (x_1 + 1.42513)^2$$
$$+(x_2 + 0.80032)^2$$

$$s.t. \quad -10 \le x_1 \le 10$$
$$-10 \le x_2 \le 10$$

Problem 8 (Leavy No. 8): [13]

$$\min \quad f(x) = \sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2$$

$$where\ y_i = 1 + \frac{x_i - 1}{4}, i = 1, \dots, n$$

$$s.t. \quad -10 \le x_i \le 10 \quad for\ i = 1,2,3$$

8. Appendix B: (Constrained Problems)

Problem 9: [8] [19]

$$\min \quad f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4X_2^2)x_2^2$$

$$s.t. \quad g_1(x) = -\sin(4\pi x_1) + 2\sin^2(2\pi x_2) \le 0$$

$$-1 \le x_i \le 1 \quad i=1,2$$

Problem 10: [8] [20]

$$\min \quad f(x) = -20exp\left(-0.2\sqrt{\frac{|x_1| + |x_2|}{2}}\right) - exp\left(\frac{\cos(2\pi x_1) + \cos(2\pi x_2)}{2}\right) + 20$$

$$s.t. \quad g_1(x) = x_1^2 + x_2^2 \le 300$$

$$g_2(x) = 2x_1 + x_2 \le 4$$

$$-30 \le x_i \le 30 \quad i = 1,2.$$

Problem 11: [8] [14] [16]

$$\min \quad f(x) = -x_1 - x_2$$

$$s.t. \quad g_1(x) = x_2 \le 2x_1^4 - 8x_1^3 + 8x_1^2 + 2$$

$$g_2(x) = x_2 \le 4x_1^4 - 32x_1^3 + 88x_1^2 - 96x_1 + 36$$

$$0 \le x_1 \le 3$$

$$0 \le x_2 \le 4$$

Problem 12: [14]

$$\min \quad f(x) = 0.01x_1^2 + x_2^2$$

$$s.t. \quad g_1(x) = x_1x_2 - 25 \ge 0$$

$$g_2(x) = x_1^2 + x_2^2 - 25 \ge 0$$

$$2 \le x_1 \le 50$$

$$0 \le x_2 \le 50$$

References

[1] Bazaraa, M., Sherali, H., & Shetty, C., (2006) Nonlinear Programming: Theory and Algorithms. 3rd edition. New Jersey: John Wikey & Sons, Inc.

[2] Byrd, R., Nocedal, J., & Waltz, R., (2006) An Integrated Package for Nonlinear Optimization, Springer US , Vol. 83, pp. 35-59.

[3] Coles, A.I., (2007), Heuristics and Metaheuristics in Forward-Chaining Planning, Doctoral dissertation, University of Strathclyde.

[4] Dennis j., & Schnabel, R., (1989) Chapter 1: A view of unconstrained optimization, In : G.L. Nemhauseer, A.H.G. Rinnooy Kan and M.J. Todd, Handbooks in Operations Research and Management Scienc, Elsevier, Vol.1, pp. 1-72, ISSN 0927-0507, ISBN 9780444872845.

[5] Dong, S., (2006), Methods for Constrained Optimization. Spring, Uk.

[6] Dong, Y., Tang, J., Xu, B., & Wang, D., (2004), An application of swarm optimization to nonlinear programming, Sciencedriect Journal, Computers and Mathematics with Applications 49(11) 1655-1668.

[7] Dorigo, M., & Socha, K., (1999) An Introduction to Ant Colony Optimization, IRIDIA, Université Libre de Bruxelles, Technical report number TR/IRIDIA/2006-010, ISSN 1781-3794.

[8] He, S., Chen, W., & Wang, H. (2011), A new filled function algorithm for constrained global optimization problems, Sciencedirect Journal; Applied Mathematics and Computation, Volume 217, Issue 12, 15 February 2011, Pages 5853-5859, ISSN 0096-3003.

[9] Huang, G., & Pan, Z. (2013). A hybrid evolutionary algorithm with importance sampling for multi-dimensional optimization. *arXiv preprint arXiv:1308.5033*.

[10] Ji, M., Jin, Z., & Tang, H., (2006), An improved simulated annealing for solving the linear constrained optimization problems, Sciencedirect Journal; Applied Mathematics and Computation, Volume 183, Issue 1, 1 December 2006, Pages 251-259, ISSN 0096-3003.

[11] Kennedy, j., & Eberhart, R.C., (2001), “Swarm Intelligence”. Morgan Kaufmann Publishers, San Francisco, California, USA. ISBN: 978-1-55860-595-4

[12] Mehta, M.H., (2012) , Hybrid Genetic Algorithm with PSO Effect for Combinatorial Optimisation Problems, International Journal of Advanced Computer Research 2(4), 300-305.

[13] Parsopoulos, K.E., & Vrahtis M.N. (2002), Recent approaches to global optimization problems through Particle Swarm, Natural Computing 1: 235-306, Kluwer Academic Publishers, Netherlands.

[14] Sarimveis, H., & Nikolakopoulos, A., (2005), A line up evolutionary algorithm for solving nonlinear constrained optimization problems, Sciencedirect Journal, Computers & Operations Research 32 (6) 1499-1514.

[15] Schoenauer, M., & Zbigniew, M., (1997), Evolutionary Computation, Published in Control and Cybernetics 26(3) pp. 307-338.

[16] Wang, W.X., Shang, Y.L., Zhang, L.S., A new algorithm for constrained global optimizatin based on filled function, ICALP, Shanghai, China 189-193.

[17] Xu, Z., Zhao, H., Min, F., & Zhu, W. (2013). Ant Colony Optimization with Three Stages for Independent Test Cost

Attribute Reduction, Hindawi Publishing Corportation, *Mathematical Problems in Engineering*.

[18] Yano, F., Shohdohji, T., & Toyoda, Y., (2007), An Improvement of Particle Swarm Optimization with a neighborhood search algorithm, IEMS 6(1), pp. 64-71.

[19] Zhang, Y.L., (2007), The filled function method for solving constrained global optimization, Tianjin: Tianjin University.

[20] Zhang, Y.L., Xu, Y, (2009), A one-parameter filled function method applied to nonsmooth constrained global opimization, Comput. Math. Appl. 58 1230-1238