Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 13, Number 9 (2017), pp. 4765-4786
© Research India Publications
http://www.ripublication.com

A Combined Genetic Algorithms-Modified Local Search scheme for Nonlinear Programming problems

Moustafa M. Salama[1,2], A. A. Mousa[1,3] , M. A. El-Shorbay[3]

[1] *Mathematics and Statistics Department, Faculty of Sciences, Taif University, KSA.*
[2]*Department of Computer-Based Engineering Applications, Informatics Research Institute, City of Scientific Research and Technology Applications, Egypt*
[3]*Basic Engineering Science Department, Faculty of Engineering, Menofia University, Egypt.*

Abstract

Evolutionary optimization algorithms provides robust and efficient techniques for solving complex real-world applications. The aim of this paper is to present an enhanced genetic algorithm (GA) with advanced particle swarm optimization (PSO) for solving nonlinear optimization problems (NLOPs), Our proposed technique is made of three stages, firstly, stageI is a classical GA, which based on the ideas of repair strategy and co-evolution. Secondly in stageII, PSO is implemented. During the procedures of PSO, a chaotic constriction factor (CCF) is used to accelerate the convergence property of PSO and retain the feasibility of the particles. Finally stageIII, Based on chaotic mapping, the chaotic search (CS) was proposed. The proposed approach can obtain the global optimal results quickly, due to fast globally converging characteristics of GA and the effective ability of PSO for locating optimal solution, also the local search ability of CS. The algorithm was tested on a set of well-known benchmark problems "CEC’05 and a Static Power Scheduling application. The performance of our algorithm is compared with some of the state of-the-art approaches. The results give an idea that the proposed modified algorithm enriches performance of the standard swarm algorithm and converges more quickly with less time to produce optimum solution.

1. INTRODUCTION

Evolutionary algorithms have received a lot of attention regarding their potential as optimization approaches for complex real optimization problems [1-2]. However, they have not made a significant breakthrough in the area of constraint NLOP [3] due to the fact that they have not addressed the issue of handling nonlinear constraints, also there is a fact that evolutionary algorithms may find only near-optimal solutions. On the other hand, many optimization problems involve inequality and/or equality constraints are thus posed as constrained optimization problems [4-5]. In trying to solve constrained problems using evolutionary algorithms or classical optimization techniques, penalty function methods have been the most popular approach [6], because of their simplicity and ease of implementation. However, since the penalty approaches are generic and applicable to any type of nonlinear constraint, their performance is not always satisfactory. Thus, several methods for handling unfeasible solutions have emerged recently [7].

Evolutionary approaches [8-12] are powerful computing systems to deal with large-scale problems. However, they require time consuming, and they are very poor in terms of convergence performance. On the other hand, local search strategy can converge quickly to local minima and get stuck in a local optimum solution, which is far away from the global optimal. The integration of global and local search procedures should offer the advantages of both optimization systems while offsetting their disadvantages. This paper presents a combined genetic algorithms-local search engine for constrained NLOPs.

In the recent years, chaos theory has been applied to many fields of the optimization. It was initially presented by Hénon and Lorenz [13-14]. It has many applications that include sociology, meteorology, engineering, physics ,economics, biology, philosophy, etc.… . Chaos is a common nonlinear phenomenon in nature, where it is fully reflects the complexity of the system that will be useful in optimization. Chaotic maps can easily be implemented and avoid entrapment in local optimal [15-19]. Chaos optimization algorithms have attracted much attention, which were all based on Logistic map. The inherent characteristics of chaos can enhance optimization algorithms by enabling it to escape from local solutions and increase the convergence to reach to the global solution. For instance, in [20] an experimental analysis on the convergence of EAs is proposed; where the effect of introducing chaotic sequences instead of random ones during all the phases of the evolution process is investigated. This approach is based on the substitution of chaotic sequences with the random number generator.

Many researchers proposed integration between chaos theory and optimization methods to improve the algorithm performance. The authors in [21] presented hybrid chaos-PSO algorithm for the vehicle routing problem with time window. While, in [22] chaotic genetic algorithm based on Lorenz chaotic system for optimization problems was proposed. In [23] an improved quantum EA is presented based on PSO and chaos to avoid the disadvantage of easily getting into the local optional solution in the later evolution period. In [24] a new PSO methods that use chaotic maps for parameter adaptation is presented; where eight chaotic maps have been analyzed in the benchmark functions and twelve chaos-embedded PSO methods have been proposed. While, Zelinka et al. [25] discussed the mutual intersection of two interesting fields of research, i.e. evolutionary computation and deterministic chaos; where evolutionary computation are explored, and deterministic chaos is investigated as a behavioral part of EAs. Moreover, in [26] an effective self-adaptive differential evolution algorithm based on Gaussian probability distribution, gamma distribution and chaotic sequence (DEGC) for solving continuous global optimization problems is proposed. On the other hand, to increase the global search ability, chaotic sequences are applied in [27] to generate candidate solutions and a new searching mechanism is used to generate new solutions. In addition, a CLS method is used to help bee colony optimization (BCO) overcome the drawback of premature convergence and increase the local exploitation capability.

In this paper, we present an enhanced genetic algorithm with advances particle swarm optimization for solving NLOP, The proposed algorithm is composed of three stages: (1) classical genetic algorithm, which based on the ideas of repair strategy and co-evolution, (2) Particle swarm optimization is implemented, and (3) the chaotic search (CS) was implemented. The algorithm was tested on a set of a well-known benchmark problems of 25 test functions "CEC'05". the obtained performance is compared against some of the state of-the-art approaches. The simulation results of various numerical studies have been demonstrated the superiority of the proposed approach to finding the global optimal solution.

This paper is organized as follows. In section 2, the General Nonlinear optimization problem is described. Section 3, we offer some well-known chaotic maps. In Section 4, the proposed algorithm is presented. Experimental results are discussed in Section 6. Finally, Section 7 presents our conclusion and notes for future work.

2- GENERAL NONLINEAR OPTIMIZATION PROBLEM (NLOP)

The general NLOP [2] is to find $\overline{x}$ such that:

$$Min \quad f\left(\overline{x}\right), \overline{x} = \left(x_1, \ldots.., x_n\right) \in R^n; \tag{1}$$

where $\overline{x} \in F \subseteq S$. The set $S \subseteq R^n$ defines the search space and the set $F \subseteq S$ defines a feasible section of the search space. Usually, the search space S is defined as n-dimensional rectangle in R^n (domains of variables defined as lower and upper bounds): $\text{left}(i) \le x_i \le \text{right}(i),\ 1 \le i \le n$ Whereas the feasible set F is defined by the search space S and an additional set of *m* equality and inequality constraints:

$$g_i\left(\overline{x}\right) \le 0, \quad for \qquad i = 1,.........,k \tag{2}$$

$$h_j\left(x\right) = 0, \quad for \qquad j = k+1,....m \tag{3}$$

3. CHAOTIC MAPS

Chaos theory studies the behavior of systems that follow deterministic laws but appear random and unpredictable. Chaos being radically different from statistical randomness, especially the inherent ability to search the space of interest efficiently, could improve the performance of optimization procedure. Chaotic map is a map (evolution function) that exhibits some sort of chaotic behavior. Maps may be parameterized by a discrete-time or a continuous-time parameter. Discrete maps usually take the form of iterated functions such as: Sinusoidal map [28], Chebyshev map [29], Singer map [30], Tent map [31], Sine map [32], Circle map [33], Piecewise map [34], Gauss map [33], Logistic map [35], Intermittency map [36], Liebovitch map [29] and Iterative map [34].

4. THE PROPOSED OPTIMIZATION SYSTEM

In this section, an enhanced genetic algorithm (GA) with advances particle swarm optimization (PSO) for solving nonlinear optimization problems (NLOPs) is proposed. It is has three stages. In stage I is a classical GA, which based on the ideas of repair strategy and co-evolution. While in stage II, chaotic particle swarm optimization (CPSO) is implemented. Finally in stage III, Based on chaotic mapping, the chaotic search (CS) is proposed. The details of the proposed algorithm is described as follows:

Stage I: GA

Step 1. Initialize a population of individuals (particles) randomly on *n*-dimensions in the problem space.

Step 2. Check the feasibility of all individuals. Repair the unfeasible individuals, The algorithm needs at least one feasible reference point to enter the evolution process (i.e., complete the algorithm procedure).

Step 3. Evaluate the desired optimization fitness function in *n* variables for each individuals.

Step 4. Using genetic operators, generate a new generation.

Step 5. while stopping criteria for GA not met, Go to step 2.

Stage II: CPSO

Step 1. Set P_{best} of each particle (obtained from stage I) equal to its current position, and set G_{best} equal to the position of best initial particle.

Step 2. The velocity and position of each particle is updated according to Equations 5 and 6.

Step 3. In the standard PSO, the new position x_i^{k+1} depends on the velocity v_i^{k+1} according to equation (6). The velocity v_i^{k+1} may be losing the feasibility of the particle. So, we introduce a chaotic constriction factor χ to accelerate the convergence property and to keep the feasibility of the particles. The chaotic constriction factor is evaluated using a well-known logistic equation, where it exhibits chaotic dynamics:

$$\chi_{n+1} = \mu \cdot \chi_n (1 - \chi_n),\ \chi_0 = 10^{-6}, \mu = 4,\ n = 0,1,2....; \tag{7}$$

where, n is the age of the infeasible particle (How long it's still unfeasible?). Finally, the new modified position of the particle is computed as:

$$x_i^{k+1} = x_i^k + \chi\, v_i^{k+1} \tag{9}$$

The pseudo code of the proposed chaotic constriction factor is shown in Fig.1.

```
Procedure make ( POPULATION = { p_i = {x_i, v_i} : (i = 1,2,...pop_size)} )
Begin
i ← 1
    While (i < pop-size) do
    χ_0 = 10^-6
        While p_i = {x_i, v_i} unfeasible
        x_i^{k+1} = x_i^k + χ v_i^{k+1}
         Check feasibility
         χ_{n+1} = μ · χ_n (1 − χ_n),
        End
    End
```

Fig. 1. Pseudo code of chaotic constriction factor

Step 4. Evaluate the desired optimization fitness function in *n* variables for each particle.

Step 5. For each particle, compare its current position x_i^{k+1} with its P_{best}. If it is better than P_{best}, then update P_{best} as $P_{best} = x_i^{k+1}$. In addition, the best particle in the entire current population is determined. If it is better than G_{best}, then update G_{best} with this best particle.

Stage 3: Chaotic search (CLS)

Optimization of the above-formulated objective function using CPSO yields an approximated optimal solution $x^* = \left(x_1^*, x_2^*, ..., x_n^*\right)$. Chaos search (CS) has the ability to perturb x^*; where local region of x^* will be explored. The detailed description of CLS is described as follows:

Step 1. Determine the range of CS $[a_i, b_i]$, $i = 1, 2, .., N$ by $x_i^* - \varepsilon < a_i$, $x_i^* + \varepsilon > b_i$; where ε is specified radius of chaos search.

Step 2. In this step, a chaotic random numbers z^L is generated by the Logistic map; where $z^0 = 10^{-3}$ by using the following equation.

$$z^L = \mu z^{L-1}\left(1 - z^{L-1}\right),\ z^0 \in (0,1),\ z^0 \notin \{0.0,\ 0.25,\ 0.50,\ 0.75,\ 1.0\},\ L = 1, 2, ...; \quad (16)$$

where *L* is the CS iterations.

Step 3. Map the chaos variable z^L into the variance range of optimization valuable $[a_i, b_i]$ by:

$$x_i^L = a_i + (b_i - a_i) z^L \quad (14)$$

which lead to

$$x_i^L = x_i^* - \varepsilon + 2\varepsilon z^L \quad \forall i = 1, ..., n \quad (15)$$

Step 4. If $f\left(x^L\right) < f\left(x^*\right)$ then set $x^* = x^L$, otherwise break the iteration.

Step 5. If $f\left(x^*\right)$ is not improved for all L iterations, stop Chaos search process and put out x^* as the best solution.

The pseudo code of the proposed algorithm is illustrated in Fig. 2, while the flow chart is shown in Fig. 3.

Initialize the population

Phase I:

While (generation number< N1←GAiteration) do

repair

Evaluation

Keep the best

Recombination

End

Phase II:

$POPULATION = \{p_i = \{x_i, v_i\} : (i = 1, 2, \ldots pop_{size})\}$

While (swarm navigation< N2←PSOiteration) do

Evaluation

Set P_{best}, G_{best}

Update $v_i^{t+1} = \psi(x_i^t, v_i^t, P_{best}, G_{best}),\ x_i^{t+1} = \xi(x_i^t, v_i^{t+1})$ for all particles

Velocity restriction

End

Phase III:

CS Procedure, given $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$, ε **and** z^0.

While: $f(x^*)$ **is improved**

$L \leftarrow 1$

Generate z^k **using Logistic map**

$x_i^L = x_i^* - \varepsilon + 2\varepsilon z^L \quad \forall i = 1, \ldots, n$

If $f(x^L) < f(x^*)$ **then** $x^* = x^k$

Else if $f(x^L) \geq f(x^*)$ **continue,**

End if

If termination criteria satisfied,

Break

End if

$L \leftarrow L + 1$

End while

End

Fig. 2. The pseudo code of the proposed algorithm

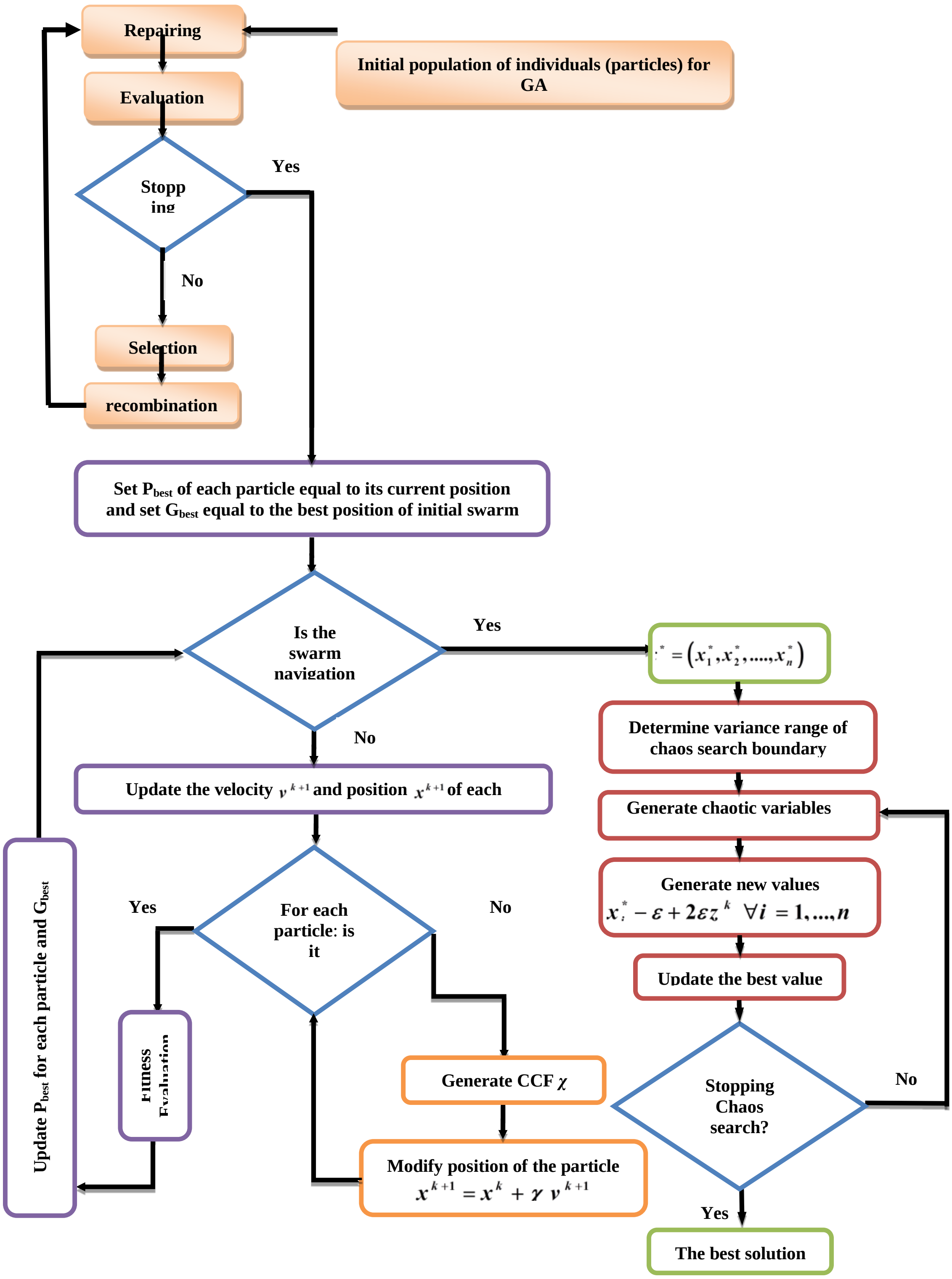

Fig. 3. The flow chart of the proposed algorithm

5. COMPUTATIONAL EXPERIMENT

The proposed algorithm is tested by the set of CEC'05 [37] and **Static Power Scheduling (Static)**

5.1. the set of 25 test functions "CEC'05"

We code the algorithm by using MATLAB 7.0 while the simulations are run on a Pentium 4 CPU 900 MHz with 512 MB memory capacity. the parameters that using in the simulation are listed in Table 1.

Table 1. Proposed algorithm parameters setting

Population size	Varying for each problem
Cognitive parameter	2.8
Social parameter	1.3
Maximum velocity of particles	Vmax=Xmax-Xmin
Inertia weight	0.6
Initial constriction factor χ_0 Navigation	Varying for each problem
Chaos search iteration	1E02
Specified neighborhood radius	1E-6
Crossover probability	0.9
Mutation probability	0.02
Selection operator	Roulette Wheel
Crossover operator	Single point
Mutation operator	Polynomial mutation

RESULTS

The comparison consist of the comparison of performance between the average error obtained by the proposed algorithm, and 9 continuous optimization algorithms [38-47] solved the set of 25 test functions "CEC'05". All the algorithms have been run 50 times for each test function of the "CEC'05". Each run stops either when the error obtained is less than 10^{-8}, or when the maximal number of evaluations (10^5) is achieved. Table 2 shows a comparison between the average errors obtained by the proposed algorithm and the 9 continuous optimization algorithms [38-47].

Table 2. Average error of the 25 CEC'05 benchmark functions obtained by our approach and 9 continuous optimization algorithms

Function	PSO [38]	IPOP-CMA-E [39]	CHC [40,41]	SSGA [42,43]	SS-BLX [44]	SS-Arit [45]	DE-Bin [46]	DE-Exp [46]	SaDE [47]	Present study
F1	$1.234\cdot10^{-4}$	0.000	2.464	$8.420\cdot10^{-9}$	$3.402\cdot10$	1.064	$7.716\cdot10^{-9}$	$8.260\cdot10^{-9}$	$8.416\cdot10^{-9}$	0.000
F2	$2.595\cdot10^{-2}$	0.000	$1.180\cdot10^{2}$	$8.719\cdot10^{-5}$	1.730	5.282	$8.342\cdot10^{-9}$	$8.181\cdot10^{-9}$	$8.208\cdot10^{-9}$	0.000
F3	$5.174\cdot10^{4}$	0.000	$2.699\cdot10^{5}$	$7.948\cdot10^{4}$	$1.844\cdot10^{5}$	$2.535\cdot10^{5}$	$4.233\cdot10$	$9.935\cdot10$	$6.560\cdot10^{3}$	$1.439\cdot10$
F4	2.488	$2.932\cdot10^{3}$	$9.190\cdot10$	$2.585\cdot10^{-3}$	6.228	5.755	$7.686\cdot10^{-9}$	$8.350\cdot10^{-9}$	$8.087\cdot10^{-9}$	0.000
F5	$4.095\cdot10^{2}$	$8.104\cdot10^{-10}$	$2.641\cdot10^{2}$	$1.343\cdot10^{2}$	2.185	$1.443\cdot10$	$8.608\cdot10^{-9}$	$8.514\cdot10^{-9}$	$8.640\cdot10^{-9}$	0.000
F6	$7.310\cdot10^{2}$	0.000	$1.416\cdot10^{6}$	6.171	$1.145\cdot10^{2}$	$4.945\cdot10^{2}$	$7.956\cdot10^{-9}$	$8.391\cdot10^{-9}$	$1.612\cdot10^{-2}$	0.000
F7	$2.678\cdot10$	$1.267\cdot10^{3}$	$1.269\cdot10^{3}$	$1.271\cdot10^{3}$	$1.966\cdot10^{3}$	$1.908\cdot10^{3}$	$1.266\cdot10^{3}$	$1.265\cdot10^{3}$	$1.263\cdot10^{3}$	1.7189
F8	$2.043\cdot10$	$2.001\cdot10$	$2.034\cdot10$	$2.037\cdot10$	$2.035\cdot10$	$2.036\cdot10$	$2.033\cdot10$	$2.038\cdot10$	$2.032\cdot10$	$2.002\cdot10$
F9	$1.438\cdot10$	$2.841\cdot10$	5.886	$7.286\cdot10^{-9}$	4.195	5.960	4.546	$8.151\cdot10^{-9}$	$8.330\cdot10^{-9}$	0.000
F10	$1.404\cdot10$	$2.327\cdot10$	7.123	$1.712\cdot10$	$1.239\cdot10$	$2.179\cdot10$	$1.228\cdot10$	$1.118\cdot10$	$1.548\cdot10$	9.9496
F11	5.590	1.343	1.599	3.255	2.929	2.858	2.434	2.067	6.796	1.6769
F12	$6.362\cdot10^{2}$	$2.127\cdot10^{2}$	$7.062\cdot10^{2}$	$2.794\cdot10^{2}$	$1.506\cdot10^{2}$	$2.411\cdot10^{2}$	$1.061\cdot10^{2}$	$6.309\cdot10$	$5.634\cdot10$	0.5145
F13	1.503	1.134	$8.297\cdot10$	$6.713\cdot10$	$3.245\cdot10$	$5.479\cdot10$	1.573	$6.403\cdot10$	$7.070\cdot10$	0.4764
F14	3.304	3.775	2.073	2.264	2.796	2.970	3.073	3.158	3.415	2.9918
F15	$3.398\cdot10^{2}$	$1.934\cdot10^{2}$	$2.751\cdot10^{2}$	$2.920\cdot10^{2}$	$1.136\cdot10^{2}$	$1.288\cdot10^{2}$	$3.722\cdot10^{2}$	$2.940\cdot10^{2}$	$8.423\cdot10$	$8.104\cdot10$
F16	$1.333\cdot10^{2}$	$1.170\cdot10^{2}$	$9.729\cdot10$	$1.053\cdot10^{2}$	$1.041\cdot10^{2}$	$1.134\cdot10^{2}$	$1.117\cdot10^{2}$	$1.125\cdot10^{2}$	$1.227\cdot10^{2}$	$1.313\cdot10^{2}$
F17	$1.497\cdot10^{2}$	$3.389\cdot10^{2}$	$1.045\cdot10^{2}$	$1.185\cdot10^{2}$	$1.183\cdot10^{2}$	$1.279\cdot10^{2}$	$1.421\cdot10^{2}$	$1.312\cdot10^{2}$	$1.387\cdot10^{2}$	$1.269\cdot10^{2}$
F18	$8.512\cdot10^{2}$	$5.570\cdot10^{2}$	$8.799\cdot10^{2}$	$8.063\cdot10^{2}$	$7.668\cdot10^{2}$	$6.578\cdot10^{2}$	$5.097\cdot10^{2}$	$4.482\cdot10^{2}$	$5.320\cdot10^{2}$	$8.000\cdot10^{2}$
F19	$8.497\cdot10^{2}$	$5.292\cdot10^{2}$	$8.798\cdot10^{2}$	$8.899\cdot10^{2}$	$7.555\cdot10^{2}$	$7.010\cdot10^{2}$	$5.012\cdot10^{2}$	$4.341\cdot10^{2}$	$5.195\cdot10^{2}$	$8.000\cdot10^{2}$
F20	$8.509\cdot10^{2}$	$5.264\cdot10^{2}$	$8.960\cdot10^{2}$	$8.893\cdot10^{2}$	$7.463\cdot10^{2}$	$6.411\cdot10^{2}$	$4.928\cdot10^{2}$	$4.188\cdot10^{2}$	$4.767\cdot10^{2}$	$8.100\cdot10^{2}$
F21	$9.138\cdot10^{2}$	$4.420\cdot10^{2}$	$8.158\cdot10^{2}$	$8.522\cdot10^{2}$	$4.851\cdot10^{2}$	$5.005\cdot10^{2}$	$5.240\cdot10^{2}$	$5.420\cdot10^{2}$	$5.140\cdot10^{2}$	$3.000\cdot10^{2}$
F22	$8.071\cdot10^{2}$	$7.647\cdot10^{2}$	$7.742\cdot10^{2}$	$7.519\cdot10^{2}$	$6.828\cdot10^{2}$	$6.941\cdot10^{2}$	$7.715\cdot10^{2}$	$7.720\cdot10^{2}$	$7.655\cdot10^{2}$	$7.505\cdot10^{2}$
F23	$1.028\cdot10^{3}$	$8.539\cdot10^{2}$	$1.075\cdot10^{3}$	$1.004\cdot10^{3}$	$5.740\cdot10^{2}$	$5.828\cdot10^{2}$	$6.337\cdot10^{2}$	$5.824\cdot10^{2}$	$6.509\cdot10^{2}$	$4.251\cdot10^{2}$
F24	$4.120\cdot10^{2}$	$6.101\cdot10^{2}$	$2.959\cdot10^{2}$	$2.360\cdot10^{2}$	$2.513\cdot10^{2}$	$2.011\cdot10^{2}$	$2.060\cdot10^{2}$	$2.020\cdot10^{2}$	$2.000\cdot10^{2}$	$2.000\cdot10^{2}$
F25	$5.099\cdot10^{2}$	$1.818\cdot10^{3}$	$1.764\cdot10^{3}$	$1.747\cdot10^{3}$	$1.794\cdot10^{3}$	$1.804\cdot10^{3}$	$1.744\cdot10^{3}$	$1.742\cdot10^{3}$	$1.738\cdot10^{3}$	$4.300\cdot10^{2}$

Also, for each problem, we rank the different algorithms according to the obtained average error as shown in Tables 3 and 4. As a result from Table 4, our approach having the first rank 14 times and having the second rank three times; this means that our approach found better solutions than all 9 algorithms [38-47] on average. So, we can say that the proposed more converges to the optimal solution and performs well on the test problems used for this study.

Table 3. Ranking of the average error of the 25 CEC'05 according to the minimum value

Function	PSO [38]	IPOP-CMA-E [39]	CHC [40,41]	SSGA [42,43]	SS-BLX [44]	SS-Arit [45]	DE-Bin [46]	DE-Exp [46]	SaDE [47]	Present study
F1	Rank(6)	Rank(1)	Rank(8)	Rank(5)	Rank(9)	Rank(7)	Rank(2)	Rank(3)	Rank(4)	Rank(1)
F2	Rank(6)	Rank(1)	Rank(9)	Rank(5)	Rank(7)	Rank(8)	Rank(4)	Rank(2)	Rank(3)	Rank(1)
F3	Rank(6)	Rank(1)	Rank(10)	Rank(7)	Rank(8)	Rank(9)	Rank(3)	Rank(4)	Rank(5)	Rank(2)
F4	Rank(6)	Rank(10)	Rank(9)	Rank(5)	Rank(8)	Rank(7)	Rank(2)	Rank(4)	Rank(3)	Rank(1)
F5	Rank(10)	Rank(2)	Rank(9)	Rank(8)	Rank(6)	Rank(7)	Rank(4)	Rank(3)	Rank(5)	Rank(1)
F6	Rank(8)	Rank(1)	Rank(9)	Rank(5)	Rank(6)	Rank(7)	Rank(2)	Rank(3)	Rank(4)	Rank(1)
F7	Rank(2)	Rank(6)	Rank(7)	Rank(8)	Rank(10)	Rank(9)	Rank(5)	Rank(4)	Rank(3)	Rank(1)
F8	Rank(10)	Rank(1)	Rank(5)	Rank(8)	Rank(6)	Rank(7)	Rank(4)	Rank(9)	Rank(3)	Rank(2)
F9	Rank(9)	Rank(10)	Rank(7)	Rank(2)	Rank(5)	Rank(8)	Rank(6)	Rank(3)	Rank(4)	Rank(1)
F10	Rank(6)	Rank(10)	Rank(1)	Rank(8)	Rank(5)	Rank(9)	Rank(4)	Rank(3)	Rank(7)	Rank(2)
F11	Rank(9)	Rank(1)	Rank(2)	Rank(8)	Rank(7)	Rank(6)	Rank(5)	Rank(4)	Rank(10)	Rank(3)
F12	Rank(9)	Rank(6)	Rank(10)	Rank(8)	Rank(5)	Rank(7)	Rank(4)	Rank(3)	Rank(2)	Rank(1)
F13	Rank(3)	Rank(2)	Rank(10)	Rank(8)	Rank(5)	Rank(6)	Rank(4)	Rank(7)	Rank(9)	Rank(1)
F14	Rank(8)	Rank(10)	Rank(1)	Rank(2)	Rank(3)	Rank(4)	Rank(6)	Rank(7)	Rank(9)	Rank(5)
F15	Rank(9)	Rank(5)	Rank(6)	Rank(7)	Rank(3)	Rank(4)	Rank(10)	Rank(8)	Rank(2)	Rank(1)
F16	Rank(10)	Rank(7)	Rank(1)	Rank(3)	Rank(2)	Rank(6)	Rank(4)	Rank(5)	Rank(8)	Rank(9)
F17	Rank(9)	Rank(10)	Rank(1)	Rank(3)	Rank(2)	Rank(5)	Rank(8)	Rank(6)	Rank(7)	Rank(4)
F18	Rank(9)	Rank(4)	Rank(10)	Rank(8)	Rank(6)	Rank(5)	Rank(2)	Rank(1)	Rank(3)	Rank(7)
F19	Rank(8)	Rank(4)	Rank(9)	Rank(10)	Rank(6)	Rank(5)	Rank(2)	Rank(1)	Rank(3)	Rank(7)
F20	Rank(8)	Rank(4)	Rank(10)	Rank(9)	Rank(6)	Rank(5)	Rank(3)	Rank(1)	Rank(2)	Rank(7)
F21	Rank(10)	Rank(2)	Rank(8)	Rank(9)	Rank(3)	Rank(4)	Rank(6)	Rank(7)	Rank(5)	Rank(1)
F22	Rank(10)	Rank(5)	Rank(9)	Rank(4)	Rank(1)	Rank(2)	Rank(7)	Rank(8)	Rank(6)	Rank(3)
F23	Rank(9)	Rank(7)	Rank(10)	Rank(8)	Rank(2)	Rank(4)	Rank(5)	Rank(3)	Rank(6)	Rank(1)
F24	Rank(8)	Rank(9)	Rank(7)	Rank(5)	Rank(6)	Rank(2)	Rank(4)	Rank(3)	Rank(1)	Rank(1)
F25	Rank(2)	Rank(10)	Rank(7)	Rank(6)	Rank(8)	Rank(9)	Rank(5)	Rank(4)	Rank(3)	Rank(1)

Table 4. Ranking statistics.

Method	Rank(1)	Rank(2)	Rank(3)	Rank(4)	Rank(5)	Rank(6)	Rank(7)	Rank(8)	Rank(9)	Rank(10)
Present study	14	3	2	1	1	0	3	0	1	0
PSO [38]	0	2	1	0	0	5	0	5	7	5
IPOP-CMA-ES [39]	6	3	0	3	2	2	2	0	1	6
CHC [40,41]	4	1	0	0	1	1	4	2	6	6
SSGA [42,43]	0	2	2	1	5	1	2	9	2	1
SS-BLX [44]	1	3	3	0	4	7	2	3	1	1
SS-Arit [45]	0	2	0	4	4	3	6	2	4	0
DE-Bin [46]	0	5	2	8	4	3	1	1	0	1
DE-Exp [46]	3	1	8	5	1	1	3	2	1	0
SaDE [47]	1	3	7	3	3	2	2	1	2	1

5.2. Static Power Scheduling (Static)

This application involves two electrical generators connected to a net with three nodes. Variables x1 and x2 are the real power outputs from the generators; x3 and x4,represent the reactive power outputs; x5, x6 and x7 are the voltage magnitudes at the nodes of the electrical network; and finally, x8 and x9 are the voltage phase angles at two of the nodes. The constraints of the model, other than the simple bounds on the variables, are the real and reactive power balance equations, the constraint stating that the power flowing into a node must balance the power flowing out. The mathematical model, described in Bartholomew-Biggs [48,49] , Hock and Schittkowski (1981)[50], Andrei (2003, p. 347)[51] , and Pant et al. (2009b)[52]. The mathematical model is to Minimize:

Minmize $f(x)=3000x_1 + 2000x_2 + 666.667x_2^3$

subject to

$$0.4-x_1 + 2cx_5^2 + x_5x_6\left(d\sin(-x_8)-c\cos(-x_8)\right)$$
$$+ x_5x_7\left(d\sin(-x_9)-c\cos(-x_9)\right) = 0,$$

$$0.4-x_2 + 2cx_6^2 + x_5x_6\left(d\sin(x_8)-c\cos(x_8)\right)$$
$$+ x_6x_7\left(d\sin(x_8-x_9)-c\cos(x_9-x_8)\right) = 0,$$

$$0.8+2cx_7^2 + x_5x_7\left(d\sin(x_9)-c\cos(x_9)\right)$$
$$+ x_6x_7\left(d\sin(x_9-x_8)-c\cos(x_9-x_8)\right) = 0,$$

$$0.2-x_3 + 2dx_5^2 + x_5x_6\left(c\sin(-x_8)+d\cos(-x_8)\right)$$
$$-x_5x_7\left(c\sin(-x_9)+d\cos(-x_9)\right) = 0,$$

$$0.2-x_4 + 2dx_6^2 - x_5x_6\left(c\sin(-x_8)+d\cos(x_8)\right)$$
$$-x_6x_7\left(c\sin(x_8-x_9)+d\cos(x_8-x_9)\right) = 0,$$

$$-0.337 + 2dx_7^2 - x_5x_7\left(c\sin(x_9)+d\cos(x_9)\right)$$
$$-x_6x_7\left(c\sin(x_9-x_8)+d\cos(x_9-x_8)\right) = 0,$$

Where

$$c = \frac{48.4}{50.176}\sin(0.25), \quad d = \frac{48.4}{50.176}\cos(0.25)$$

The simple bounds on variables are:

$$x_i \geq 0, \quad i = 1,2,$$
$$0.90909 \leq x_i \geq 1.0909, \quad i = 5,6,7.$$

Results

All nonlinear equations $h_j(x) = 0$ $(for\ j = k+1,...,m)$ are replaced by pair of inequalities: $-\varepsilon \leq h_j(x) \leq \varepsilon$ with additional parameter (ε) to define the precision of the system [53]. **Table 5** lists the parameter setting. Figures 4-7 Shows the convergence analysis of the algorithm with the generation counter with different value of precision parameter ε^* .

Table 5: Proposed algorithm parameters setting

Population size	100
Cognitive parameter	2.8
Social parameter	1.3
Maximum velocity of particles	$V_{max}=X_{max}-X_{min}$
Inertia weight	0.6
Initial constriction factor χ_0 Navigation	Varying for each problem
Chaos search iteration	1E02
Specified neighborhood radius	1E-6
Crossover probability	0.9
Mutation probability	0.02
Selection operator	Roulette Wheel
Crossover operator	Single point
Mutation operator	Polynomial mutation

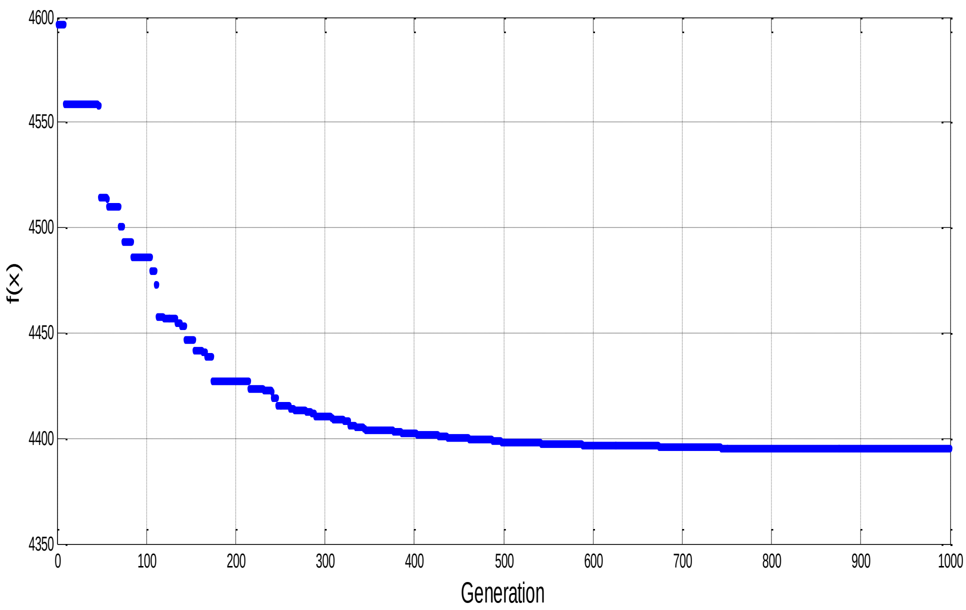

Fig. 4 Convergence diagram for ε=1E-04

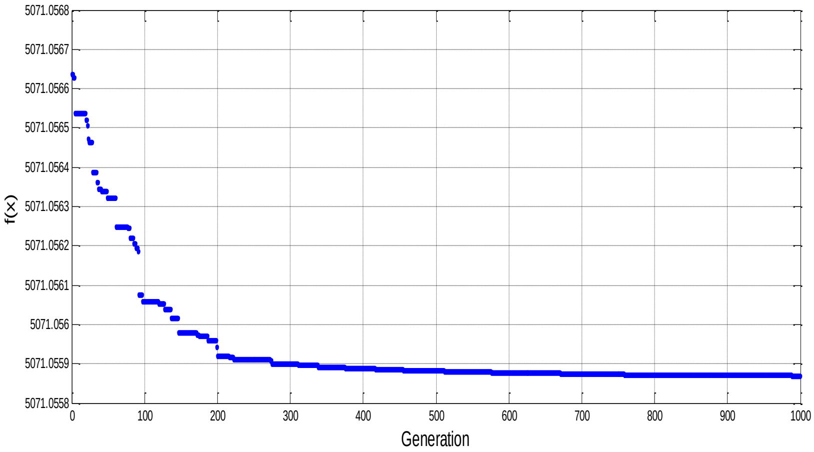

Fig 5 Convergence diagram for ε=1E-06

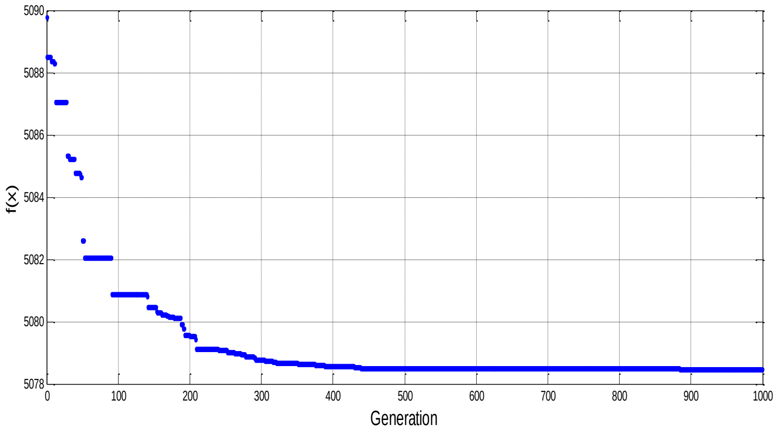

Fig 6 Convergence diagram for ε=1E-08

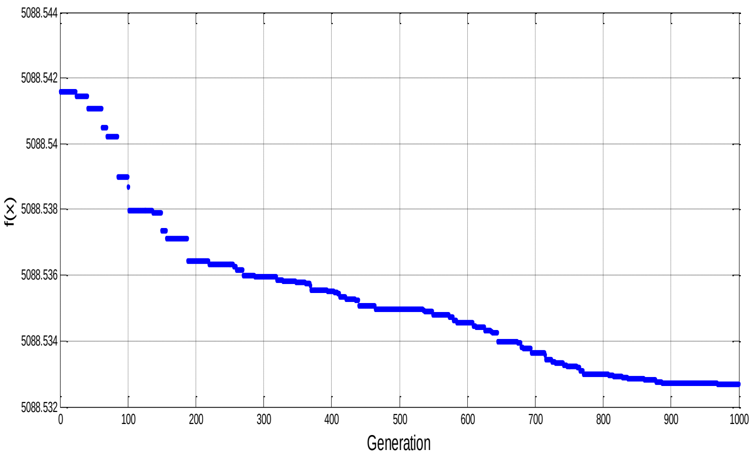

Fig 7 Convergence diagram for ε=1E-10

6- DISCUSSION

In the view of our results on benchmark problems "CEC'05 , we rank the different algorithms according to the obtained average error as in Tables 3 and 4. As a result from Table 4, our approach having the first rank 14 times and having the second rank three times; this means that our approach found better solutions than all 9 algorithms [38-47] on average. So, we can say that the proposed more converges to the optimal solution and performs well on the test problems used for this study.

On the other hand, in the view of our results on Static Power Scheduling application, the algorithm converges to the optimal solution as in figure, where the Cost of generating is converges to the optimal value. we were deeply interested in evaluating the performances of the nonlinear optimization software CONOPT, KNITRO, MINOS, SNOPT and the proposed algorithm for different system precision "ε"

Table 6: Performances of the nonlinear optimization software CONOPT, KNITRO, MINOS, SNOPT and the proposed algorithm for different system precision "ε"

Algorithm	$f(x)$
CONOPT [54]	5055.01
KNITRO[54]	5055.01
MINOS[54]	5055.01
SNOPT[54]	5055.01
Proposed algorithm, ε=1E-04	4391.23
Proposed algorithm, ε=1E-06	5071.05
Proposed algorithm, ε=1E-08	5079.23
Proposed algorithm, ε=1E-10	5088.53

7. CONCLUSIONS AND RECOMMENDATION

In this Project, an enhanced genetic algorithm (GA) with advances particle swarm optimization (PSO) for solving nonlinear optimization problems (NLOPs). It is has three stages. Stage I is a classical GA, which based on the ideas of repair strategy and co-evolution. While, in stage II, PSO is implemented. During the procedures of PSO, a chaotic constriction factor (CCF) is used to accelerate the convergence property of PSO and retain the feasibility of the particles; where it controls the movement velocity of each particle so as to improve search engine visibility. Finally in stage III, Based on chaotic mapping, the chaotic search (CS) was proposed. The proposed approach can obtain the global optimal results quickly, due to fast globally converging characteristics of GA and the effective ability of PSO for locating optimal solution, also the local search ability of CS. The algorithm was tested on a set of well-known benchmark problems "CEC'05" and a Static Power Scheduling application. The performance of our algorithm is compared with some of the state of-the-art approaches. The following observations reveal benefits of our approach:

1. It has been used to increase the solution quality by combing the merits of GA, PSO algorithm and chaos theory.
2. Unlike classical techniques, the proposed algorithm search from a population of points. Therefore it can provide a globally optimal solution.
3. It uses only the objective function information, not derivatives or other auxiliary knowledge. Therefore it can deal with all types of functions which are actually existed in practical optimization problems.
4. The simulation results prove superiority of it to those reported in the literature, where it completely better than the other approaches.
5. Due to procedure simplicity, the reality of using our algorithm to handle complex problems of realistic dimensions has been approved.

Finally, the intelligent algorithms have great important in real life applications. So, the following will be researched in our future works:

a) Solving more and larger scale examples to research and demonstrate the efficiency of our approach.
b) Introduce an extension of our approach to solve the multi-objective problems.
c) Provide new constriction factors to save the solutions feasibility and accelerate the convergence to optimal solutions.

REFERNCES

[1] J. Zhang, P. Liu, Z. Zhou, L. Ma, Z. Li, W. Ni, A mixed-integer nonlinear programming approach to the optimal design of heat network in a polygeneration energy system, Applied Energy 114 (2014) 146-154

[2] M.S. Osman, M.A. Abo-Sinna, A.A. Mousa, A solution to the optimal power flow using genetic algorithm, Applied Mathematics and Computation 155 (2) (2004) 391-405.

[3] R. Sarathy, B. Shetty, A. Sen, A constrained nonlinear 0–1 program for data allocation, European Journal of Operational Research 102 (3) (1997) 626-647.

[4] Y.I. Bontzios, M.G. Dimopoulos, A.A. Hatzopoulos, An evolutionary method for efficient computation of mutual capacitance for VLSI circuits based on the method of images, Simulation Modelling Practice and Theory 19 (2) (2011) 638-648.

[5] S.S. Rao, Engineering Optimization: Theory and Practice, 3rd Edition NY, USA: Wiley, 2003.

[6] C.A. Floudas, Deterministic Global Optimization, Kluwer Academic, Boston, Mass, USA, 1999.

[7] J. Dou, X. Wang, An Efficient Evolutionary Programming, in: International Symposium on Information Science and Engineering 2 (2008) 401–404.

[8] A.A. Mousa, Study on multiobjective optimization using improved genetic algorithm: methodology and application, ISBN 978-3-8465-4889-9, Lambert academic publishing GmbH&Co.kG, Berlin, 2011.

[9] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, Ponnuthurai Nagaratnam Suganthan, Qingfu Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation 1 (1) (2011) 32-49.

[10] M.A. El-Shorbagy, A.A. Mousa, W.F. Abd-El-Wahed, Hybrid Particle Swarm Optimization Algorithm for Multi-Objective Optimization, ISBN 978-3-8473-1149-2, Lambert academic publishing GmbH&Co.kG, Berlin,2011.

[11] M.A. El-Sisy, W.F. Abd-El-Waheed, A.A. Mousa, Ant colony algorithm to support multi-criteria decision making", ISBN 978-3-659-32175-7, Lambert academic publishing GmbH&Co.kG, Berlin, 2013.

[12] M.S. Osman, M.A. Abo-Sinna, A.A. Mousa, A Combined Genetic Algorithm-Fuzzy Logic Controller (GA-FLC) In Nonlinear Programming, Journal Of Applied Mathematics & Computation (AMC) 170 (2005) 821-840.

[13] X.H. Shi, Y.C. Liang, H.P. Lee, C. Lu, L.M. Wang, An improved GA and a novel PSO-GA-based hybrid algorithm, Information Processing Letters 93 (2005) 255–261.

[14] M.A. Panduro, C.A. Brizuela, L.I. Balderas and D.A. Acosta, A Comparison of Genetic Algorithms, Particle Swarm Optimization And The Differential Evolution Method for The Design of Scannable Circular Antenna Arrays, Progress In Electromagnetics Research B 13 (2009) 171-186.

[15] O. Uysal, S. Bulkan, Comparison of Genetic Algorithm and Particle Swarm Optimization for Bicriteria Permutation Flowshop Scheduling Problem,

International Journal of Computational Intelligence Research 4 (2) (2008) 159–175.

[16] C.-l. Sun, J.-c. Zeng, J.-s. Pan, An improved vector particle swarm optimization for constrained optimization problems, Information Sciences 181 (2011) 1153–1163.

[17] A. Khare, S. Rangnekar, A review of particle swarm optimization and its applications in Solar Photovoltaic system, Applied Soft Computing 13 (2013) 2997–3006.

[18] M. Iwana, R. Akmeliawatib, T. Faisala and H. M A A Al-Assadi, Performance Comparison of Differential Evolution And Particle Swarm Optimization In Constrained Optimization, Procedia Engineering 41 (2012) 1323 – 1328.

[19] M. Thakur, A new genetic algorithm for global optimization of multimodal continuous functions, J. Comput. Sci. (2013), http://dx.doi.org/10.1016/j.jocs.2013.05.005.

[20] W.F. Abd-El-Wahed, A.A. Mousa, M.A. El-Shorbagy, Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems, Journal of Computational and Applied Mathematics 235 (2011) 1446–1453.

[21] R.J. Kuo, Y.S. Han, A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem—a case study on supply chain model, Applied Mathematical Modelling 35 (8) (2011) 3905–3917.

[22] Y. Kao, E. Zahara, A hybrid genetic algorithm and particle swarm optimization for multimodal functions, Applied Soft Computing 8 (2008) 849–857.

[23] Y. Hu, Y. Ding, K. Hao, An immune cooperative particle swarm optimization algorithm for faulttolerant routing optimization in heterogeneous wireless sensor networks, Mathematical Problems in Engineering, Article ID 743728 (2012) 19 pages.

[24] A. Yadav, K. Deep, An efficient co-swarm particle swarm optimization for non-linear constrained optimization, J. Comput. Sci. (2013), http://dx.doi.org/10.1016/j.jocs.2013.05.011.

[25] M. Nezhad, R.A. Shandiz, A.E. Jahromi, A particle swarm–BFGS algorithm for nonlinear programming problems, Computers & Operations Research 40 (2013) 963–972.

[26] L. Costa, I. A.C.P. E. Santo, E. M.G.P. Fernandes, A hybrid genetic pattern search augmented Lagrangian method for constrained global optimization,Applied Mathematics and Computation 218 (2012) 9415–9426.

[27] C.-H. Lin, A rough penalty genetic algorithm for constrained optimization, Information Sciences 241 (2013) 119–137.

[28] Y. Li, S. Deng and D. Xiao, A novel Hash algorithm construction based on chaotic neural network, Neural Comput. Appl. 20 (2011) 133–141.

[29] M.S. Tavazoei and M. Haeri, Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms, Appl. Math. Comput. 187 (10) (2007) 76–85.

[30] H. Peitgen, H. Jurgen and D. Saupe, Chaos and Fractals, Springer-Verlag, Berlin, Germany (1992).

[31] E. Ott, Chaos in Dynamical Systems, Cambridge University Press, UK, Cambridge (2002).

[32] R.L. Devaney, An Introduction to Chaotic Dynamical Systems, Addison-Wesley (1987).

[33] R.C. Hilborn, Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers, second ed., Oxford Univ. Press, New York (2004).

[34] R.M. May, Simple mathematical models with very complicated dynamics, Nature 261(4) (1976) 59–67.

[35] J.S. Arora, O.A. Elwakeil, A.I. Chahande and C.C. Hsieh, Global optimization methods for engineering application: a review, Struct. Optim. 9 (1995) 137–159.

[36] A. Erramilli, R.P. Singh and P. Pruthi, Modeling Packet Traffic With Chaotic Maps, Royal Institute of Technology, Stockholm-Kista, Sweden (1994).

[37] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC'2005 special session on real parameter optimization, Nanyang Technological University, Tech. Rep., 2005, Available in http://www.ntu.edu.sg/home/epnsugan/index_files/cec-05/Tech-Report-May-30-05.pdf.

[38] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IV IEEE International Conference on Neural Networks, Piscataway, New Jersey (1995) 1942–1948.

[39] A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in: Proceedings of the 2005 IEEE Congress on Evolutionary Computation (2005) 1769–1776.

[40] L.J. Eshelman, The CHC adaptative search algorithm: How to have safe search when engaging in nontraditional genetic recombination, in: G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, California. (1991) 265–283.

[41] L.J. Eshelman, J.D. Schaffer, Real-coded genetic algorithms and interval schemata, in: D. Whitley (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, California (1993) 187–202.

[42] C. Fernandes, A. Rosa, A study of non-random matching and varying population size in genetic algorithm using a royal road function, in: Proceedings of the 2001 Congress on Evolutionary Computation, Piscataway, New Jersey (2001) 60–66.

[43] H. Mülenbein, D. Schlierkamp-Voosen, Predictive models for the breeding genetic algorithm in continuous parameter optimization, Evolutionary Computation 1 (1993) 25–49.

[44] F. Herrera, M. Lozano, D. Molina, Continuous scatter search: An analysis of the integration of some combination methods and improvement strategies, European Journal of Operational Research 169 (2) (2006) 450–476.

[45] M. Laguna, R. Marti, Scatter Search. Methodology and Implementation in C, Kluwer Academic Publishers (2003).

[46] K.V. Price, M. Rainer, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag (2005).

[47] A.K. Qin, P.N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, in: Proceedings of the 2005 IEEE Congress on Evolutionary Computation 2 (2005) 1785–1791.

[48]M. C .B-Biggs , “A Numerical Comparison between Two Approaches to the Nonlinear Programming Problem”, In: L. C. W. Dixon and G. P. Szego, eds., Towards Global Optimization 2, Amsterdam, Holland: North Holland Publishing Company, pp. 293 – 312, (1978).

[49] W. Hock, & K. Schittkowski Test examples for nonlinear programming codes (Lecture notes in economics and mathematical systems, Vol. 187). Berlin: Springer (1981).

[50] N .Andrei, Models, test problems and applications for mathematical programming. Bucharest: Technical Press. (2003).

[51] M.Pant, R. Thangaraj, & V. P. Singh, Particle swarm optimization with crossover operator and its engineering applications. IAENG International Journal of Computer Science,36(2), 112–121, (2009).

[52] D. Thierens, and D. E. Goldberg, Convergence Models of Genetic Algorithm Selection, Parallel Problem Solving from Nature - PPSN , pp. 119-129,DOI: 10.1007/3-540-58484-6_256, (1994).

[53] A.A. Mousa And I.M. eldesoky," GENLS: Co-evolutionar algorithm for nonlinear system of equations", Journal of Applied Mathematics & Computation (AMC) Vol. 197(2) pp. 633-642, (2008).

[54] Neculai Andrei, Nonlinear Optimization Applications Using the GAMS Technology, Springer Science & Business Media,2010.