POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

ISSN: 1302-0900 (PRINT), ISSN: 2147-9429 (ONLINE)

URL: http://www.politeknik.gazi.edu.tr/index.php/PLT/index

Comparison of particle swarm and differential evolution optimization algorithms considering various benchmark functions

Parçacık sürüsü ve diferansiyel evrim optimizasyonu algoritmalarının farklı ölçü fonksiyonları kullanılarak karşılaştırılması

Yazar(lar) (Author(s)): *Hasan OZCAN*

Bu makaleye şu şekilde atıfta bulunabilirsiniz(To cite to this article): Ozcan H., "Comparison of particle swarm and differential evolution optimization algorithms considering various benchmark functions", ***Politeknik Dergisi***, 20(4): 899-905, (2017).

Erişim linki (To link to this article): http://dergipark.gov.tr/politeknik/archive

DOI: **10.2339/politeknik.369076**

Comparison Of Particle Swarm And Differential Evolution Optimization Algorithms Considering Various Benchmark Functions

Araştırma Makalesi / Research Article

Hasan OZCAN

Karabuk Universitesi, Muhendislik Fakultesi, Makine Muhendisligi Bolumu, Demir Celik Kampusu78050 Karabuk, Turkiye

(Geliş/Received : 02.10.2016 ; Kabul/Accepted : 11.10.2016)

ABSTRACT

This study aims to compare Particle Swarm Optimization (PSO) and Differential Evolution (DE) methods for various input parameters. Both optimization methods show high performance in optimization of any physical system including simple and complex constraints and objectives. Average and standard values of both methods were evaluated by utilizing 8 benchmark functions and a graphical representation and comparison of corresponding methods was presented for 50x50 and 100x100 population sizes and dimensionalities. It is concluded that DE and PSO show the best fitness value for Sum of Different Powers benchmark function for both number of populations. Approach to the optimum is found to be faster through the PSO method. Both methods are flexible to be used for simple and complex engineering problems with high performances with ease of programming.

Keywords : Particle swarm, differential evolution, optimization, benchmark function.

Parçacık Sürüsü ve Diferansiyel Evrim Optimizasyonu Algoritmalarının Farklı Ölçü Fonksiyonları Kullanılarak Karşılaştırılması

ÖZ

Bu çalışma parçacık sürüsü optimizasyonu ve diferansiyel evrim algoritmalarını farklı giriş parametreleri kullanarak karşılaştırmayı amaçlamaktadır. Bu iki optimizasyon algoritması basit ve kompleks kısıtlar ve sınırlayıcıları da içine alan herhangi bir fiziksel sistemin optimizasyonunda yuksek performans gostermektedirler. Sekiz adet ölçü fonksiyonu kullanılarak iki algoritma için averaj ve standart değerler hesaplanmış ve 50x50 ve 100x100 boyutlarında ve uzaylarında ilgili algoritmaların karşılaştırılması yapılmıştır. İki algoritma da (Farklı kuvvetlerin toplamı) ölçü fonsiyonunda en iyi değeri göstermektedir. Optimum değere parçacık sürüsü optimizasyonu ile daha hızlı ulaşılsa da iki metot da basit ve komplex optmizasyon problemlerinin çözümünde etkin performans ve kolay programlanabilirlik avantajları göstermektedir.

Anahtar kelimeler: Parçacık sürüsü, diferansiyel evrim, optimizasyon, ölçü fonskiyonu.

1. INTRODUCTION

Optimization's aim is to determine the best solution to a problem under a given set of constraints. Mathematically an optimization problem involves a fitness function describing the problem, under a set of constraints representing the solution space for the problem. Because of difficulties in evaluating the first derivatives to locate the optima for many rough and discontinuous optimization surfaces, several derivative free optimization algorithms have emerged [1].

Holland and his colleagues demonstrated Genetic Algorithms (GA) and shown how biological crossovers and mutations of chromosomes can be realized in the algorithm to improve the quality of the solutions over successive iterations [2]. In 1990s Eberhart and Kennedy proposed an alternative solution to the non-linear optimization problems by considering behaviors of bird flocks and called it as Particle Swarm Optimization (PSO) [3]. Price and Storn took an attempt to replace classical crossover and mutation operators in GA by alternative operators and came up with a differential operator to handle the problem. The algorithm they proposed is called Differential Evolution (DE) [4]. Many application of these algorithms for engineering systems are performed. Some recent studies based on these algorithms can be found elsewehere [5-7]. An aircooling systems optimization is performed by [5], where it is concluded that both PSO and DE algorithms shows higher performance than that of Lagrangian methodology (LM) for a simple optimization problem. Comparative evaluation of both optimization methodologies are also discussed in [6], and [7].

**Sorumlu Yazar (Corresponding Author)*
e-posta : *hasanozcan@karabuk.edu.tr*

PSO and DE algorithms do not require gradient information for the function to be optimized and conceptually simple. These algorithms can be implemented in all programming languages and require minimum parameter tuning. In this work, MATLAB is used as computer language for both parameters and a detailed analysis of algorithms for various benchmark functions are represented [8]. A brief information for the corresponding algorithms and benchmark functions are given in next section.

2. EXPLANATION OF ALGORITHMS

In this section, a brief explanation for DE and PSO is done and benchmark functions which are going to be used for testing these optimization methods are shown with their features.

A. Differential evolution

Differential evolution is a method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. Such methods are commonly known as metaheuristics as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as DE do not guarantee an optimal solution is ever found [9].

DE optimizes a problem by maintaining a population of candidate solutions and creating new candidate solutions by combining existing ones according to its simple formulae, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand. In this way the optimization problem is treated as a black box that merely provides a measure of quality given a candidate solution and the gradient is therefore not needed [10]. Stages of differential evolution are briefly given below:

- Initialization
- Fitness evaluation
- Mutation
- Crossover
- Selection

Procedure of the DE algorithm is given as follows [1]:

Input: Randomly initialized position and velocity of the particle $x_i(0)$

Output: Position of the approximate global optima X^*

Begin

Initialize population
Evaluate fitness
For i=0 to max iteration do
Begin
Create difference-offspring;
Evaluate fitness
If an offspring is better than its parent
Then replace the parent by offspring in the next generation;
End If;
End for;

End

B. Particle swarm optimization

Particle swarm optimization is a stochastic population-based metaheuristic inspired from swarm intelligence. It mimics the social behavior of natural organisms such as bird flocking and fish schooling to find a place with enough food. Indeed, in some swarms, a coordinated behavior using local movements emerges without any central control. Originally, PSO has been successfully designed for continuous optimization problems. A template of the particle swarm optimization algorithm is as follows [10,11]:

Random initialization of the whole swarm

Repeat

Evaluate $f(x_i)$;
For all particles I
Update velocities (v_i);
Move to new position
If $f(xi)<f(pbest_i)$ **Then** $pbest_i = x_i$;
If $f(xi)<f(gbest)$ **Then** $pbest = x_i$;
Update (x_i,v_i)

End For

Until stopping Criteria

C. Parameter settings

Input parameters for testing DE and PSO are given in Table 1. N_p is the population size, used for both algorithms and the dimensionality of the problem n is taken to be equal to the size of the population. F and C_r are mutation constant and crossover rate for DE, respectively. *C1* and *C2* represents learning factors for PSO and taken to be equal. *NFC* is maximum number of function calls for both algorithms. Number of runs per algorithm per function is taken as 30 and applied to both algorithms.

Table 1. Input Parameters

Algorithm	***Np=n***	***F***	***C1=C2***	***Cr***	***NFC***
DE	50,100	0.5	-	0.9	5000*n
PSO	50,100	-	2	-	5000*n

D. Benchmark functions

In the field of evolutionary computation, it is common to compare different algorithms using a large test set, especially when the test involves function optimization. However, the effectiveness of an algorithm against another algorithm cannot be measured by the number of problems that it solves better. The ``no free lunch'' theorem shows that, if we compare two searching algorithms with all possible functions, the performance of any two algorithms will be, on average, the same. Table 2

Table 2. Benchmark functions [12]

Function	Definition	Features
F(1) 1st De Jong	$f_1(x)=\sum_{i=1}^{n}x_i^2$ $-5.12\le x_i\le 5.12$ $\min(f_1)=f_1(0,...,0)=0$	Unimodal, scalable, convex, easy function
F(2) Axis Parallel Hyper-Ellipsoid	$f_2(x)=\sum_{i=1}^{n}ix_i^2$ $-5.12\le x_i\le 5.12$ $\min(f_2)=f_2(0,...,0)=0$	Unimodal, Scalable, convex, easy function
F(3) Schwefel's Problem	$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{n}x_j\right)^2$ $-65\le x_i\le 65$ $\min(f_3)=f_3(0,...,0)=0$	Unimodal, Scalable
F(4) Rosenbrock's Valley	$f_4(x)=\sum_{i=1}^{n}[100(x_{i+1}-x_i^2)+(1-x_i)^2]$ $-2\le x_i\le 2$ $\min(f_4)=f_4(0,...,0)=0$	Banana function, non-convex, unimodal,
F(5) Rastrigin's Function	$f_5(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i))$ $-5.12\le x_i\le 5.12$ $\min(f_5)=f_5(0,...,0)=0$	Highly multimodal, location of the minima is regularly distributed
F(6) Griewangk's Function	$f_6(x)=\sum_{i=1}^{n}\frac{x_i^2}{4000}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$ $-600\le x_i\le 600$ $\min(f_6)=f_6(0,...,0)=0$	Many regularly distributed local minima and hard to locate global minimum
F(7) Sum of Different Power	$f_7(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{(i+1)}$ $-1\le x_i\le 1$ $\min(f_7)=f_7(0,...,0)=0$	Unimodal, scalable
F(8) Ackley's Function	$f_8(x)=-20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{n}x_i^2}{n}}\right)-\exp\left(\frac{\sum_{i=1}^{n}\cos(2\pi x_i)}{n}\right)+20+e$ $-32\le x_i\le 32$ $\min(f_8)=f_8(0,...,0)=0$	

shows the benchmark functions for testing DE and PSO with their solution ranges and various features [12].

3. EXPERIMENTAL RESULTS

The Analysis for both 50 and 100 populations are done with MATLAB 7.5.0 programming language on an Intel Core (I5) CPU 520 @2.40 GHz 1.17 GHz computer.

Both DE and PSO codes are integrated in the main algorithm while function files for algorithms and benchmark functions are coded separately. Figures 1-8 represent performance values of corresponding algorithms for a population size and dimensionality of 50 for a 50,000 iteration over 30 runs. Every figure shows the comparison of algorithms for corresponding benchmark function. Figures 9-16 show performance values of DE and PSO for the population size and dimensionality of 100. The elapsed time for 50 and 100 population size is around 33 hours and 60 hours respectively. Since the 3rd benchmark function (Schwefel's Problem) is programmed with a *for loop*, elapsed time for evaluation of this function is almost 4 times more than other functions' time consumptions.

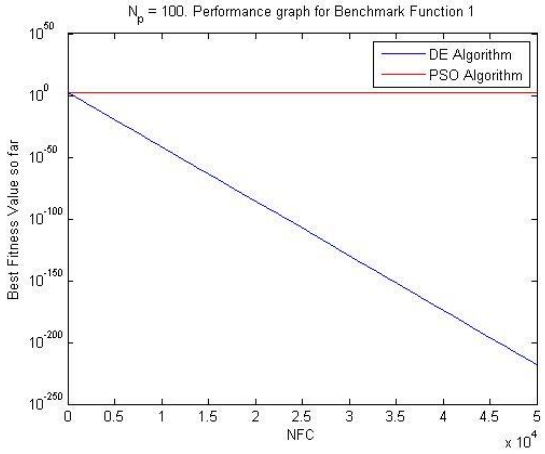

Figure 1. Best fitness value so far for both DE and PSO for function 1 at N_p=50.

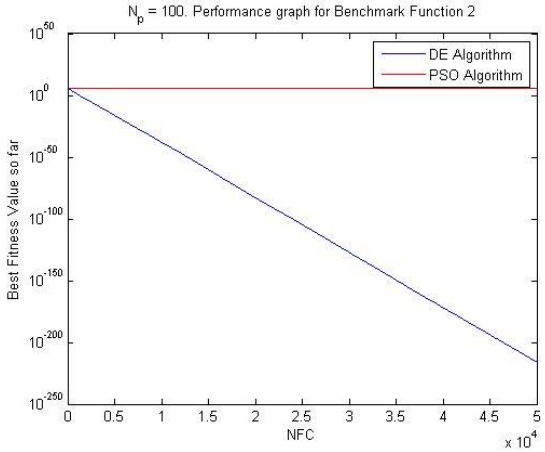

Figure 2. Best fitness value so far for both DE and PSO for function 2 at N_p=50.

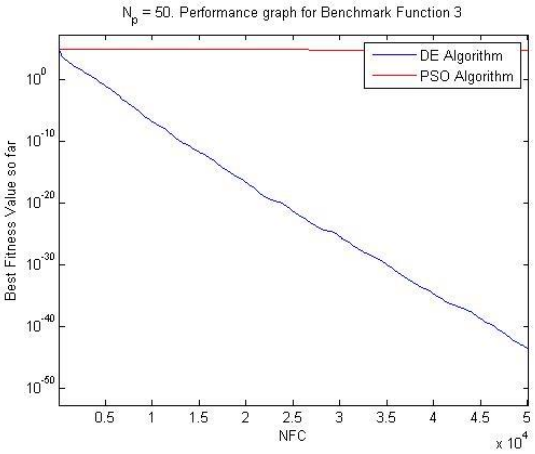

Figure 3. Best fitness value so far for both DE and PSO for function 3 at N_p=50.

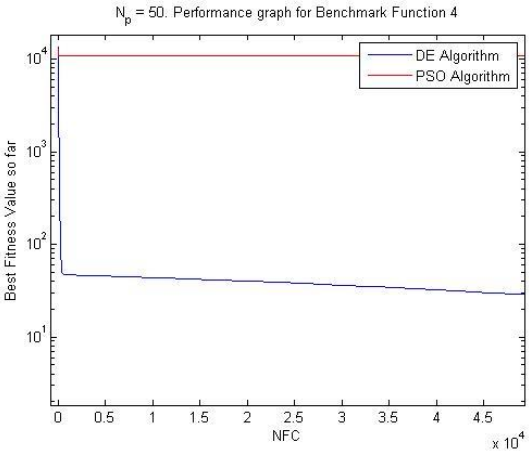

Figure 4. Best fitness value so far for both DE and PSO for function 4 at N_p=50.

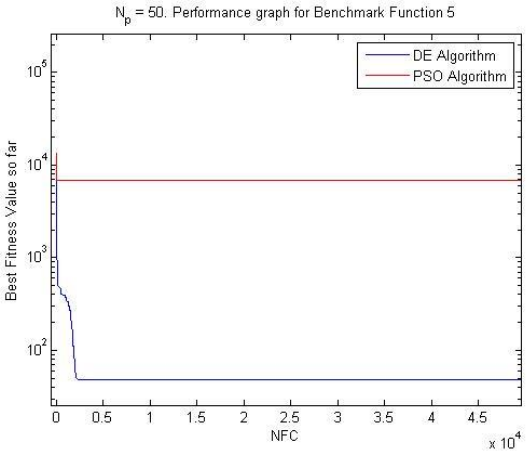

Figure 5. Best fitness value so far for both DE and PSO for function 5 at N_p=50.

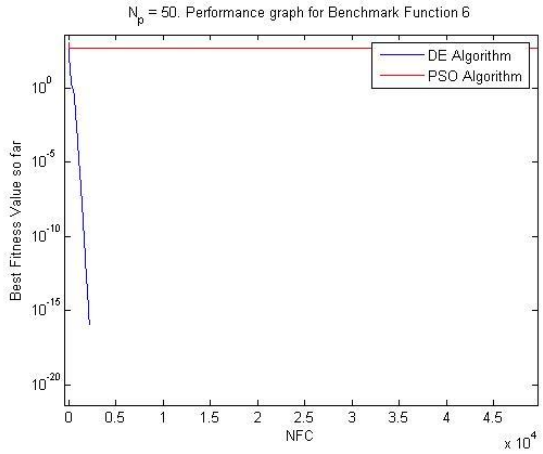

Figure 6. Best fitness value so far for both DE and PSO for function6 at N_p=50.

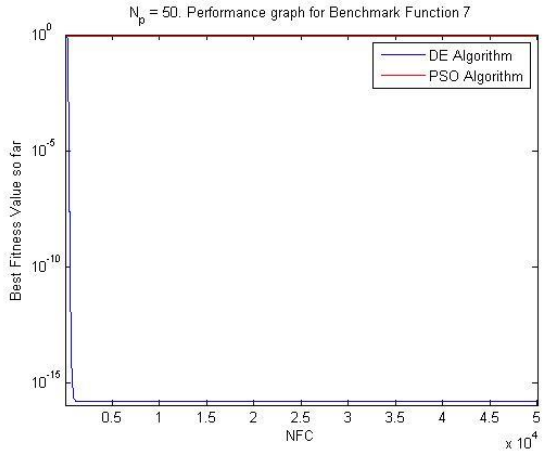

Figure 7. Best fitness value so far for both DE and PSO for function7 at N_p=50.

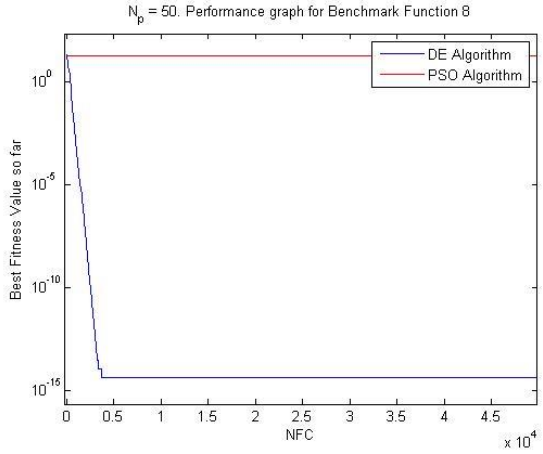

Figure 8. Best fitness value so far for both DE and PSO for function8 at N_p=50.

After 2000 iterations we could not gain any results for DE for function 6, however it has given a very close value to zero as 10^{-15}. For almost all functions PSO has found an optimal value after just a few iterations and then did not change. Figures 8-16 show performance values of DE and PSO for the population size and dimensionality of 100 at 100,000 iteration. If more iteration would be implemented to function 3 and 4, better results could be obtained but also the time span is significantly increasing. Comparison of average and standard values of both algorithms are discussed considering tables 3 and 4.

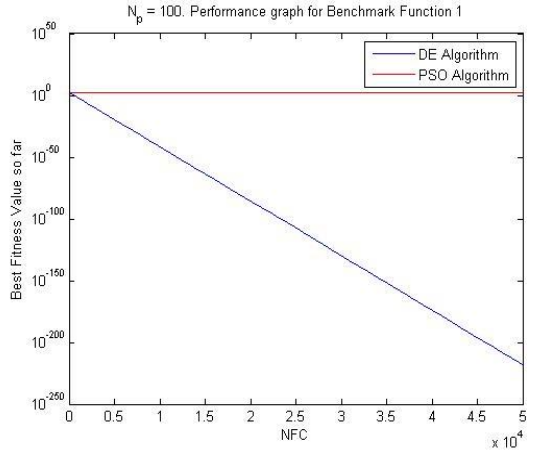

Figure 9. Best fitness value so far for both DE and PSO for function 1at N_p=100

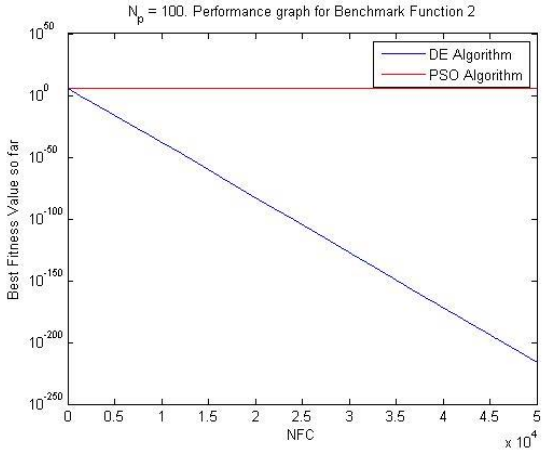

Figure 10. Best fitness value so far for both DE and PSO for function 2 at N_p=100

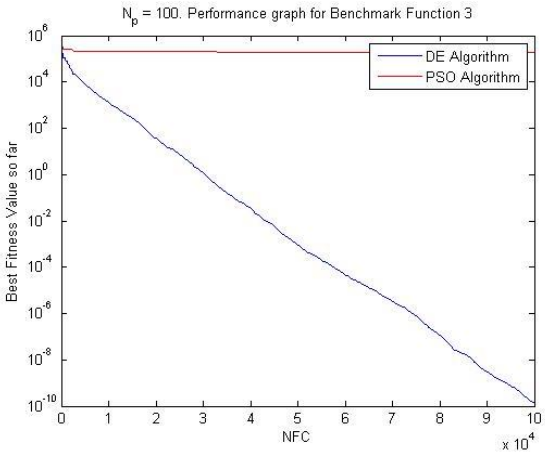

Figure 11. Best fitness value so far for both DE and PSO for function 3 at N_p=100

It is clearly seen from the last eight figures (Figs 9-16) that as the the population, dimensionality and iteration numbers are increased best fitness value for function 3

approaches to the desired value after 75,000 iterations. An exact comparison between N_p=50 and N_p=100 is hard because DE and PSO algorithms create random solutions. If a new evaluation would be assigned to the same software, graphs would not be the same and change as these algorithms are based on approximate algorithms and initiate randomly.

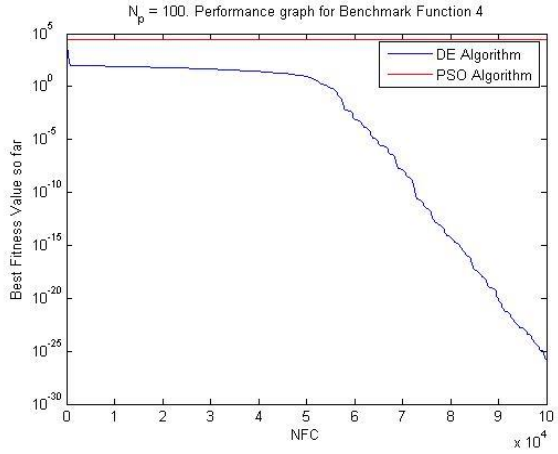

Figure 12. Best fitness value so far for both DE and PSO for function 4 at N_p=100.

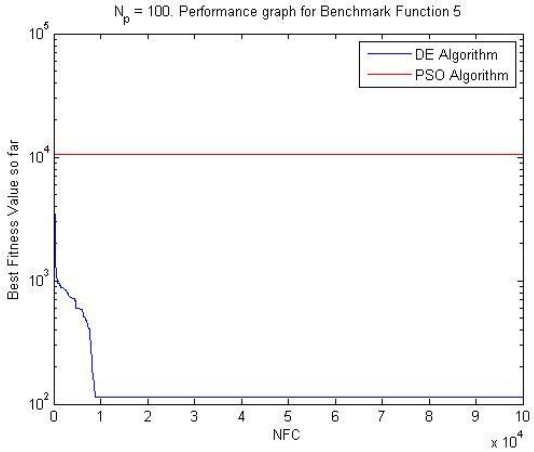

Figure 13. Best fitness value so far for both DE and PSO for function 5 at N_p=100.

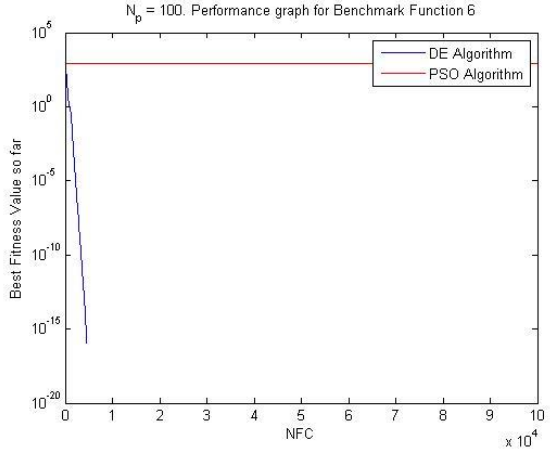

Figure 14. Best fitness value so far for both DE and PSO for function 6 at N_p=100.

PSO evaluation stops changing its value after a few iterations just like in N_p=50 and 50,000 iterations. Best average and standard values are obtained from function 7 and 8. These benchmark functions are proper functions for testing both algorithms.

Average and standard values of best fitness function for all functions considered are given in table 1. Best fitness values are obtained using functions 1,6,7 and 8 for DE and 7 for PSO. Benchmark function 7 has shown the best performance among other functions for implementation of both algorithms. When increasing the run number of algorithms, Average and standard values are slightly approaching to the desired value. However, as soon as we increase the run number, the time span for the solution increases.

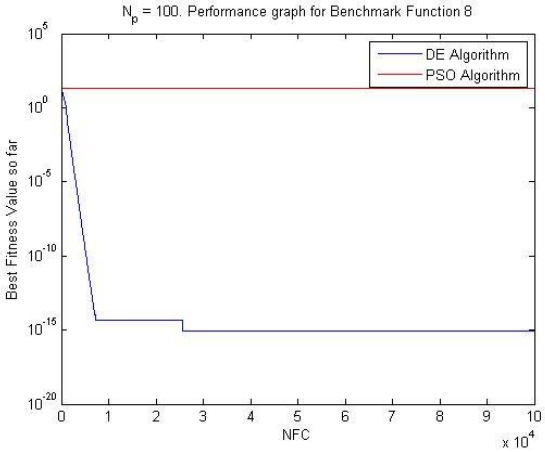

Figure 15. Best fitness value so far for both DE and PSO for function 8 at N_p=100.

Comparing N_p=50 to N_p=100, average values for DE increase at functions 2, 3, 5 and 7; and decrease at functions 1, 4, 6 and 8. Nevertheless standard values increase at functions 1,1,2,3,4,5,6 and 8 and decrease at function 7. When considering the same changes in PSO, average values of PSO increase for all benchmark functions and standar deviation decrease for functions 5,6 and 8. Values of 0 for functions 4 and 7 are due to unchanging evaluation of the PSO algorithm.

Table 3. Avg. and STD. Values OF DE and PSO for all functıons at np=50

Functions	Avg. DE	Std. DE	Avg. PSO	Std. PSO
f(1)	0.88	10.85	140.02	1.77
f(2)	1061.84	12595.14	69464.93	2450.54
f(3)	1825.31	9747.53	58474.32	1021.98
f(4)	114.35	801.11	8438.66	0.00
f(5)	158.88	605.06	4346.03	83.33
f(6)	4.62	53.17	345.95	8.58
f(7)	0.01	0.05	1.11	0.00
f(8)	0.36	2.12	18.19	0.03

Table 4. Avg. and STD. Values OF DE and PSO for all functıons at np=100

Functio ns	Avg. DE	Std. DE	Avg. PSO	Std. PSO
f(1)	0.85	19.30	329.88	1.81
f(2)	1197.6 8	29694.6 3	596506.4 4	2187.51
f(3)	2611.0 5	15201.2 9	194284.2 2	17139.9 2
f(4)	67.14	886.58	30392.43	0.00
f(5)	190.4	717.06	10568.10	55.09
f(6)	2.60	60.98	849.24	4.17
f(7)	1.17	0.00	1.56	0.00
f(8)	0.08	0.99	19.14	0.01

4. CONCLUSIONS

In this study, a comparison between Differential Evolution and Particle Swarm Optimization algorithms has been studied for various benchmark functions. Following results have been extracted from the analysis:

- Results approach to the desired value as iteration number increases.
- When the number of population and dimensionality increase, process time for the evaluation of both algorithms increase as well, however a slight change in results has been observed.
- Best fitness values obtained from PSO algorithm do not change after a few iterations for most benchmark functions.
- The best average and standard fitness values have been obtained via "Sum of Different Power" test function (Function #7).

It has been concluded that population based metaheuristics are promising optimization methods for large search spaced optimization problems and can be initialized with few input parameters. These algorithms can be implemented to various engineering problems specifically for complex system optimization with various parameters affecting whatsoever parameter under consideration. Approach to the optimum is found to be faster through the PSO method although both methods are flexible to be used for simple and complex engineering problems with high performances with ease of programming.

REFERENCES

1) Das S., Abraham A. and Konar A., "Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives", **Studies in Computational Intelligence**, 116: 1-38, (2008).
2) Holland J.H., "Adaptation in natural and artificial systems", ***University of Michigan Press***, Ann Arbor, (1975).
3) Kennedy J.and Elbehart R, "Particle swarm optimization" in proceedings of ***IEEE International Conference on Neural Networks***, 1942-1948, (1995).
4) Storn R.,and Price K., "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces", ***Journal of Global Optimization***, 11(4): 341-359, (1997).
5) Ozcan H., Ozdemir K. and Ciloglu H., "Optimum cost of an air cooling system by using differential evolution and particle swarm algorithms",.***Energy and Buildings***, 65:93-100, (2013).
6) Zhang F., Deb C., Lee S.E., Yang J. and Shah K.W., "Time series forecasting for building energy consumption using weighted Support Vector Regression with differential evolution optimization technique***", Energy and Buildings***, 126: 94-103, (2016).
7) Dezelak K., Bracinik P., Höger M. and Otcenasova A., "Comparison between the particle swarm optimisation and differential evolution approaches for the optimal proportional–integral controllers design during photovoltaic power plants modelling", ***IET Renewable Power Generation***,10(4): 522-530, (2016).
8) MATLAB Version 7.5.0, The Mathworks Inc., 2007.
9) Price K., Storn R.M. and Lampinen J.A., "Differential evolution: a practical approach to global optimization". ***Springer***, London, (2005).
10) Talbi E.G., "Metaheuristics: from design to implementation", ***John Wiley & Sons***, Newyork, (2009).
11) Kennedy J., Elbehart R., "Swarm intelligence", ***Morgan Kaufmann***, San Francisco, (2001).
12) Internet Source: http://www.cs.cmu.edu/afs/cs/project /jair/pub/ volume 24/ ortiz boyer 05a-html/node6.html, accessed on May 2016.