Hybrid Evolutionary Algorithm for Solving Global Optimization Problems

Radha Thangaraj[1], Millie Pant [1], Ajith Abraham[2] and Youakim Badr[2]

[1] Department of Paper Technology,IIT Roorkee, India
[2]National Institute of Applied Sciences of Lyon, INSA-Lyon, Villeurbanne, France
t.radha@ieee.org, millifpt@iitr.ernet.in, ajith.abraham@ieee.org,
youakim.badr@insa-lyon.fr

Abstract. Differential Evolution (DE) is a novel evolutionary approach capable of handling non-differentiable, non-linear and multi-modal objective functions. DE has been consistently ranked as one of the best search algorithm for solving global optimization problems in several case studies. This paper presents a simple and modified hybridized Differential Evolution algorithm for solving global optimization problems. The proposed algorithm is a hybrid of Differential Evolution (DE) and Evolutionary Programming (EP). Based on the generation of initial population, three versions are proposed. Besides using the uniform distribution (U-MDE), the Gaussian distribution (G-MDE) and Sobol sequence (S-MDE) are also used for generating the initial population. Empirical results show that the proposed versions are quite competent for solving the considered test functions.

Keywords: Hybrid Algorithm, Differential Evolution, Evolutionary Programming, Global Optimization.

1 Introduction

Evolutionary Algorithms (EAs) are general-purpose stochastic search methods imitating the phenomena of biological evolution. One of the reasons of the success of EAs is their population based strategy which prevents them from getting trapped in a local optimal solution and consequently increases their probability of finding a global optimal solution. Thus, EAs can be viewed as global optimization algorithms. Some frequently used EAs include Evolutionary Programming (EP) [1], Evolution Strategies (ES) [2], Genetic Algorithms (GA) [3], Particle Swarm Optimization [4], and Differential Evolution [5]. These algorithms have been applied successfully to wide range of problems [6] – [9]. Some common features of these algorithms may be given as:

- Start with population of points instead of single point
- Do not depend on initial guess.
- Are able to solve ill-defined or inexplicitly expressed problems as they do not depend on the mathematical properties like continuity or differentiability
- Inherits natural parallelism

Despite having several attractive features, these algorithms also have weaknesses and drawbacks like slow convergence, loss of diversity, stagnation of population leading to a suboptimal performance etc. These problems become more prominent in case of multimodal problems having several local and global optima. Several variants of these algorithms have been proposed in the past to improve their performance. In the present study we propose a hybridized version of DE, which is relatively a newer addition to the class of EA.

DE was proposed by Storn and Price [5] in 1995. It soon became a popular tool for solving global optimization problems because of several attractive features like fewer control parameters, ease in programming, efficiency etc. DE has parameters like mutation, crossover and selection for guiding the population towards the optimum solution similar to GAs. However, it's the application of these operators that makes DE different from GA. The main difference between GAs and DE is that; in GAs, mutation is the result of small perturbations to the genes of an individual while in DE mutation is the result of arithmetic combinations of individuals. At the beginning of the evolution process, the mutation operator of DE favors exploration. As evolution progresses, the mutation operator favors exploitation. Hence, DE automatically adapts the mutation increments (i.e. search step) to the best value based on the stage of the evolutionary process. Mutation in DE is therefore not based on a predefined probability density function.

DE has been successfully applied to solve a wide range of real life application problems such as clustering [10], unsupervised image classification [11], digital filter design [12], optimization of non-linear functions [13], global optimization of non-linear chemical engineering processes [14] and multi-objective optimization [15] etc. Also it has reportedly outperformed other optimization techniques [16] – [18].

However like other EA, DE has certain flaws like slow convergence and stagnation of population. Several modified versions of DE are available in literature for improving the performance of basic DE. One class of such algorithms includes hybridized versions where DE is combined with some other algorithm to produce a new algorithm. DE has been hybridized with ACO, Simulated Annealing, PSO, local search methods like Nelder Mead etc. The hybridized versions have also been used successfully for solving practical application problems [19]-[21].

The present study differs from other hybridized algorithms in two ways; firstly we have hybridized DE with EP, which to the best of our knowledge has not been done before and secondly we use different initializing techniques for generation of random numbers like uniformly distributed random numbers, Gaussian distributed random numbers and random numbers generated using quasi random Sobol sequence. The proposed algorithms are named as Modified Differential Evolution having uniform distribution (U-MDE), having Gaussian distribution (G-MDE), having Sobol distribution (S-MDE). The rationale for using different initialization techniques is that the population based search methods generally use computer generated uniformly distributed random numbers. This technique however is not very efficient as the computer generated random numbers may not cover the search domain effectively. In the present work we used Sobol sequence to generate the initial population. It is a quasi random sequence and covers the search space more evenly in comparison to the

computer generated random numbers. It has given better results in comparison to the algorithms using uniformly distributed random numbers [22] – [24].
The remaining of the paper is organized as follows: in Sections 2 and 3, we give a brief description of DE and EP algorithms respectively. Section 4, describes the proposed MDE algorithm. Experimental settings are given in Section 5. Benchmark problems and corresponding numerical results are given in Section 6. The paper finally concludes with Section 6.

2 Differential Evolution

A general DE variant may be denoted as DE/X/Y/Z, where X denotes the vector to be mutated, Y specifies the number of difference vectors used and Z specifies the crossover scheme which may be binomial or exponential. Throughout the study we shall consider the mutation strategy DE/rand/1/bin [5]. It is also known as the classical version of DE and is perhaps the most frequently used version of DE. DE works as follows: First, all individuals are initialized with uniformly distributed random numbers and evaluated using the fitness function provided. Then the following will be executed until maximum number of generation has been reached or an optimum solution is found.
For a D-dimensional search space, each target vector $x_{i,g}$, a mutant vector is generated by

$$v_{i,g+1} = x_{r_1,g} + F*(x_{r_2,g} - x_{r_3,g}) \quad (1)$$

where $r_1, r_2, r_3 \in \{1,2,....,NP\}$ are randomly chosen integers, must be different from each other and also different from the running index i. F (>0) is a scaling factor which controls the amplification of the differential evolution $(x_{r_2,g} - x_{r_3,g})$. In order to increase the diversity of the perturbed parameter vectors, crossover is introduced. The parent vector is mixed with the mutated vector to produce a trial vector $u_{ji,g+1}$,

$$u_{ji,g+1} = \begin{cases} v_{ji,g+1} & if \ (rand_j \le CR) \ \ or \ \ (j = j_{rand}) \\ x_{ji,g} & if \ (rand_j > CR) \ \ and \ \ (j \ne j_{rand}) \end{cases} \quad (2)$$

where j = 1, 2,……, D; $rand_j \in [0,1]$; CR is the crossover constant takes values in the range [0, 1] and $j_{rand} \in (1,2,.....,D)$ is the randomly chosen index.
Selection is the step to choose the vector between the target vector and the trial vector with the aim of creating an individual for the next generation.

3 Evolutionary Programming

Evolutionary programming (EP) originated from the research of L.J. Fogel in 1962 [25] on using simulated evolution to develop artificial intelligence. The concept of self adaptive EP (SAEP) was introduced by Back and Schwefel [26] and Fogel [27]

and was shown to be more efficient than the normal EP. The computational steps of SAEP are given below:

Step 1: Each individual is taken as a pair of real-valued vectors, (x_i , σ_i) for all i=1,…,M. The x_i's give the ith member's object variables and σ_i's the associated strategy parameters.

Step 2: Evaluate the objective function of each individual.

Step 3: Mutation: Creates a single offspring (x_i' , σ_i'), from each parent (x_i , σ_i) for all i=1,…, M by

$$\sigma_i'(j) = \sigma_i(j) \exp(\tau\, N(0,1) + \tau'\, N_j(0,1))$$
$$x_i'(j) = x_i(j) + \sigma_i'(j)\, N_j(0,1) \text{ for all } j = 1,.....n.$$

where N(0,1) denotes a random number distributed by Gaussian distribution.

The factors τ and τ' are commonly set to $1/\sqrt{2n}$ and $1/\sqrt{2\sqrt{n}}$ respectively.

Step 4: Calculate the objective function value of each offspring (x_i' , σ_i'), for all i = 1,…,M.

Step 5: Selection:

Each individual x from the union of parents (x_i , σ_i) and offspring (x_i' , σ_i'), is evaluated against q other randomly chosen solutions. For each comparison, a "win" is assigned if x is better than its opponent The M solutions with the greatest number of wins are retained to be parents of the next generation. Parameter q allows tuning selection pressure, typically $q = 10$.

Step 6: Stop if the stopping criteria is reached otherwise go to step 3.

4 Proposed MDE Algorithm: A Hybridized Version of DE and EP

The proposed DE-PSO as mentioned earlier is a hybrid version of DE and EP. MDE starts like the usual DE algorithm up to the point where the trial vector is generated. If the trial vector is better than the target vector, then it is included in the population otherwise the algorithm enters the EP phase and generates a new candidate solution using EP based mutation. The method is repeated iteratively till the optimum value is reached. The inclusion of EP phase creates a perturbation in the population, which in turn helps in maintaining diversity of the population and producing a good optimal solution. The proposed MDE algorithm initialize with uniform distribution is called as U-MDE, initialize with Gaussian distribution is called as G-MDE and initialize with Sobol sequence is called as S-MDE. The pseudo code of the MDE Algorithm is:

```
Initialize the population using uniform (/Gaussian/ Sobol
sequence) distributed random numbers
For i = 1 to N (Population size) do
Select r1, r2, r3 ∈ N randomly
// r1, r2, r3 are selected such that r1≠ r2 ≠ r3 //
For j = 1 to D (dimension) do
      Select jrand ∈D
      If (rand () < CR or j = jrand)
// rand () denotes a uniformly distributed random number
between 0 and 1//
```

$$U_{ji,g+1} = x_{r_1,g} + F*(x_{r_2,g} - x_{r_3,g})$$

End if

$$X_{ji,g+1} = \begin{cases} U_{ji,g+1} & \text{if } f(U_{ji,g+1}) < f(X_{ji,g}) \\ TX_{ji} & \text{if } f(TX_{ji}) < f(X_{ji,g}) \quad \text{// EP Phase} \\ X_{ji,g} & \text{otherwise} \end{cases}$$

Where $TX_{ji} = X_{ji,g} + \sigma_i * N_j(0,1)$

$\sigma_i = \sigma_i * \exp(\tau_a * N(0,1) + \tau_b * N_j(0,1))$

End for

End for

5 Experimental Settings and Results

We considered a set of ten unconstrained benchmark functions namely Rastringin (f_1), Spherical (f_2), Griewank (f_3), Rosenbrock (f_4), Ackley (f_5), Generalized penalized function 1 (f_6), Generalized penalized function 2 (f_7), Levy (f_8), Test2N (f_9) and Circle (f_{10}). All the algorithms were implemented using Dev C++ on a PC compatible with Pentium IV, a 3.2 GHz processor and 2 GB of RAM. The following experimental settings have been considered for the present study:
Experimental settings for proposed MDE algorithms and DE:
For dimension 30: Pop=50, run=30, Max Gne=3000
Experimental settings for proposed MDE and DEPSO [28] (Table 4):
Pop: 30, dim: 30, Max Gne.: 12000 for functions f_1, f_3, f_5 and Max Gne.: 4000 for function f_6
Experimental settings for proposed MDE and BBDE [29] (Table 5):
Pop: 30, dim: 30, Max number of function evaluations: 100000
Experimental settings for proposed MDE and DEPSO [30] (Table 6):
Pop: 20, 40 and 80, dim: 30, Max Gne: 2000
In the above mentioned settings, Pop denotes the population size taken; run denotes the number of times an algorithm is executed; Max Gne denotes the maximum number of generations allowed for each algorithm

6.1 Numerical Results

The proposed versions are compared with the basic DE and with two other hybrid versions of DE, called BBDE [29] and DEPSO [28], [30]. In Table 1 we give the comparison of the proposed versions with the basic DE in terms of average fitness function value, standard deviation and the number of generations required to satisfy the given stopping criteria. In terms of average fitness function value all the algorithms gave good performance as it is evident from Table 1, although the proposed versions gave a slightly better performance in some of the test cases. If we compare the standard deviation, then also we can observe that all the algorithms converged to the desired objective function value with small value for standard deviation which is less than zero in almost all the test cases. This tendency shows the stability of the algorithms. However when we compare the proposed versions with the basic DE in terms of number of generations, it can be clearly seen that the proposed

versions converged much faster in comparison to the basic DE. The performance curves of the proposed algorithms with respect to few selected problems are given in Figure 1.

The proposed versions are also compared with three other hybrid versions available in literature. In Table 2, we compare the performance of the proposed algorithms with DEPSO; in Table 3, the comparison is done with BBDE and in Table 4, the comparison is done with DEPSO, another hybridized version of DE. In all the versions taken for comparison, hybridization of DE is done with PSO. The results of other hybridized versions are taken from literature and therefore we have considered the problems which are common to all the algorithms. For example in Table 2, we have only considered test cases f1, f3, f5, f6 because these were the only test cases common to the problems considered for the present study and to the literature. From Tables 2-4, we can easily see that the proposed versions give a much better performance in comparison to other hybridized versions available in literature for almost all the test problems.

Table 1 Average Fitness function Value (AFV), Standard Deviation (STD) and average number of generations for basic DE and the modified versions proposed in the present study

F	DE		U-MDE		G-MDE		S-MDE	
	AFV (STD)	Gen.	AFV (STD)	Gen.	AFV (STD)	Gen.	AFV (STD)	Gen.
f_1	1.22e-05 (8.47e-06)	364	1.09e-05 (5.84e-06)	351	**8.00e-07** (4.3e-06)	316	1.17e-05 (**3.51e-06**)	**282**
f_2	1.12e-05 (6.07e-06)	173	8.66e-06 (**2.17e-06**)	167	6.27e-06 (3.38e-06)	165	**8.75e-07** (5.42e-06)	**135**
f_3	9.86e-09 (1.76e-08)	699	1.05e-10 (**2.14e-10**)	713	7.07e-11 (3.48e09)	140	**9.79e-012** (2.75e-09)	**112**
f_4	0.0953 (**0.0479**)	3000	0.0259 (0.1437)	3000	**0.0044** (0.1723)	3000	0.04857 (1.0967)	3000
f_5	5.39e-05 (1.55e-05)	323	4.90e-05 (1.84e-05)	318	**4.66e-06** (1.51e-05)	250	4.56e-05 (**1.33e-05**)	**239**
f_6	1.04e-05 (5.93e-06)	214	8.92e-06 (4.97e-06)	214	9.59e-06 (4.37e-06)	**116**	**8.59e-06** (**3.37e-06**)	123
f_7	-1.15043 (5.94e-06)	232	-1.15043 (7.46e-06)	232	**-1.15044** (4.75e-06)	167	-1.15043 (**3.29e-06**)	**162**
f_8	-21.2361 (6.00e-06)	223	**-21.5023** (5.48e-06)	215	**-21.5023** (**5.45e-06**)	**200**	**-21.5023** (7.53e-06)	223
f_9	-77.2608 (6.49e-06)	195	**-78.3323** (4.87e-06)	193	**-78.3323** (**2.02e-06**)	**119**	**-78.3323** (4.84e-06)	191
f_{10}	0.0914 (0.0235)	3000	0.0869 (0.0246)	3000	0.0868 (0.00022)	3000	**0.08049** (0.01276)	3000

Table 2 Comparison Results (1): MDE vs DEPSO [28] in terms of average fitness value

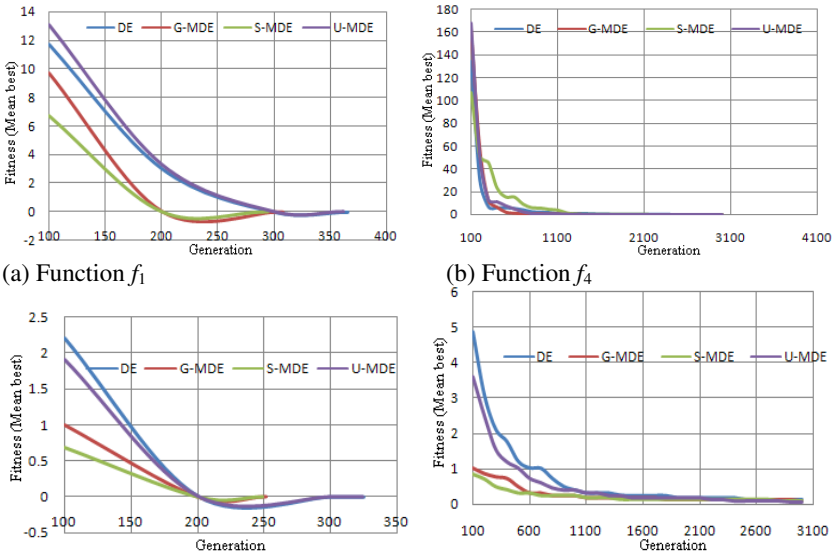
F	DEPSO	U-MDE	G-MDE	S-MDE
f_1	24.216 (6.417)	**0.0000(0.0000)**	**0.0000(0.0000)**	**0.0000(0.0000)**
f_3	6.2e-16 (4.1e-16)	**0.0000(0.0000)**	**0.0000(0.0000)**	**0.0000(0.0000)**
f_5	-0.0002 (0.0002)	**3.69e-15(0.0000)**	**3.69e-15(0.0000)**	**3.69e-15(0.0000)**
f_6	3.9e-20 (4.1e-21)	5.51e-13(0.0000)	5.32e-18(0.0000)	**4.71e-22(0.0000)**

Table 3 Comparison Results (2): MDE vs BBDE [29] in terms of average fitness value

F	BBDE	U-MDE	G-MDE	S-MDE
f_1	72.185 (3.018)	1.73e-13(1.36e-13)	1.99e-18(3.66e-18)	**0.0000(0.0000)**
f_3	0.269e-01 (0.767-02)	2.16e-20(2.65e-20)	**5.42e-21(1.62e-20)**	**5.42e-21(1.62e-20)**
f_4	**14.295 (0.948)**	48.129(25.23)	25.51(0.9108)	25.69(3.91)
f_5	2.1361 (0.159)	2.18e-14(4.03e-15)	1.08e-14(3.55e-15)	**7.25e-15(2.31e-15)**

Table 4 Comparison Results (3): MDE vs DEPSO [30] in terms of average fitness value

F	Swarm size	DEPSO	U-MDE	G-MDE	S-MDE
f_1	20	0.8656	0.1989	**0.0000**	0.3979
	40	0.009950	2.83e-14	1.73e-19	**0.0000**
	80	3.919E-9	1.11e-13	5.20e-19	**0.0000**
f_3	20	0.009073	1.08e-20	**0.0000**	4.33e-20
	40	0.006930	3.25e-20	1.08e-20	**0.0000**
	80	0.005589	2.16e-20	0.0000	**0.0000**
f_4	20	80.8259	43.95	26.64	**25.3198**
	40	66.8730	28.52	24.74	**24.5649**
	80	60.6405	28.7815	24.15	**23.2118**



(a) Function f_1 (b) Function f_4

(c) Function f_5 (d) Function f_{10}

Fig 1 Performance curves of selected benchmark problems

7 Conclusions

In the present study, we propose modified hybridized versions of DE algorithm for solving global optimization problems. The proposed versions used EP type mutation

operator stochastically during the selection of trial vector. The inclusion of EP mutation operator helps in increasing the diversity which in turn helps in better exploration of the search space which finally helps in improving the solution quality and the convergence rate of an algorithm. This is evident from the empirical studies done in the present study. Another experiment that we have done is with the initialization of the population. Besides using the computer generated random numbers in U-MDE, we also initialized the population with Gaussian distribution and with a Quasi Random sequence called Sobol sequence. As expected, initialization with a quasi random sequence gave much better results in comparison to the other distributions. This behavior is quite expected because a quasi random sequence covers the search domain more uniformly in comparison to other distributions. We are continuing our work towards the theoretical development of the proposed algorithms and extending them for solving constrained optimization problems.

References

[1] L. J. Fogel, A. J. Owens, and M. J. Walsh.: Artificial intelligence through a simulation of evolution. In M. Maxfield, A. Callahan and L. J. Fogel, editors, Biophysics and Cybernetic systems. Proc. of the 2nd Cybernetic Sciences Symposium, pp. 131 – 155, Spartan Books, (1965).

[2] I. Rechenberg.: Evolution Strategy: Optimization of Technical systems by means of biological evolution. Fromman-Holzboog (1973).

[3] Holland, J. H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Ann Arbor, MI: University of Michigan Press.

[4] Kennedy, J. and Eberhart, R.: Particle Swarm Optimization. *IEEE International Conference on Neural Networks* (Perth, Australia), IEEE Service Center, Piscataway, NJ, pg. IV: 1942-1948 (1995).

[5] R. Storn and K. Price.: Differential Evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report, International Computer Science Institute, Berkley (1995).

[6] Blesa, M.J.; Blum, C.: A nature-inspired algorithm for the disjoint paths problem. In Proc. Of 20th Int. Parallel and Distributed Processing Symposium, IEEE press, pp. 1 – 8 (2006).

[7] Y. delValle, G. K. V. Moorthy, S. Mohagheghi, J-C. Hernandez, and R. G. Harley.: Particle Swarm Optimization: Basic Concepts, Variants and Applications in Powwer Systems. IEEE Trans. On Evolutionary Computation, Vol 12 (2), pp. 171 – 195 (2008).

[8] C-T. Hsiao, G. Chahine and N. Gumerov.: Application of a Hybrid Genetic/Powell Algorithm and a Boundary Element Method to Electrical Impedence Tomograpghy. Journal of Computational Physics, Vol. 173, pp. 433 – 453 (2001).

[9] S. Kannan, S. M. R. Slochanal and N. P. Pathy.: Application and Comparison of metaheuristic techniques to generation expansion planning problem. IEEE Trans. on Power Systems, Vol 20(1), pp.466 – 475 (2005).

[10] Paterlini. S, Krink. T.: High performance clustering with differential evolution." In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 2, pp. 2004–2011 (2004).

[11] Omran. M, Engelbrecht. A and A.Salman.: Differential evolution methods for unsupervised image classification," In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 2, pp. 966–973 (2005).

[12] Storn. R.: Differential evolution design for an IIR-filter with requirements for magnitude and group delay. Technical Report TR-95-026, International Computer Science Institute, Berkeley, CA (1995).
[13] Babu. B and Angira. R.: Optimization of non-linear functions using evolutionary computation. In: Proceedings of the 12th ISME International Conference on Mechanical Engineering, India, pp. 153–157 (2001).
[14] Angira. R and Babu. B.: Evolutionary computation for global optimization of non-linear chemical engineering processes. In: Proceedings of International Symposium on Process Systems Engineering and Control, Mumbai,pp. 87–91 (2003).
[15] Abbass. H.: A memetic pareto evolutionary approach to artificial neural networks. Lecture Notes in Artificial Intelligence, vol. 2256. Springer, pp. 1–12 (2002a).
[16] Vesterstroem. J and Thomsen. R.: A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. *Proc. Congr. Evol. Comput.*, vol. 2, pp. 1980–1987 (2004).
[17] Andre. J, Siarry. P, and Dognon. T.: An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization. *Advance in Engineering Software 32*, pp. 49–60. (2001).
[18] Hrstka. O and Kucerová. A.: Improvement of real coded genetic algorithm based on differential operators preventing premature convergence. *Advance in Engineering Software 35*, pp. 237–246 (2004).
[19] Ji-Pyng Chiou.: Variable scaling hybrid differential evolution for large-scale economic dispatch problems. Electric Power Systems Research, Vol. 77, Issues 3-4, pp. 212 – 218 (2007).
[20] Feng-Sheng Wang, Tzu-Liang Su, and Horng-Jhy Jang.: Hybrid Differential Evolution for Problems of Kinetic Parameter Estimationand Dynamic Optimization of an Ethanol Fermentation Process. *Ind. Eng. Chem. Res.*, Vol. 40 (13), pp. 2876-2885 (2001).
[21] Changtong Luo, Bo Yu.: Low Dimensional Simplex Evolution—A Hybrid Heuristic for Global Optimization. Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. Pp. 470 – 474 (2007).
[22] S. Kimura and K. Matsumura.: Genetic Algorithms using low discrepancy sequences. In Proc. of GEECO 2005, pp. 1341 – 1346 (2005).
[23] Nguyen X. H., Nguyen Q. Uy., R. I. Mckay and P. M. Tuan.; Initializing PSO with Randomized Low-Discrepancy Sequences: The Comparative Results. In Proc. of IEEE Congress on Evolutionary Algorithms, pp. 1985 – 1992 (2007).
[24] Millie Pant, Radha Thangaraj and Ajith Abraham.: Improved Particle Swarm Optimization with Low-discrepancy Sequences. In Proc. IEEE Cong. on Evolutionary Computation, Hong Kong, pp. 3016 – 3023 (2008).
[25] L.J. Fogel.: Autonomous Automata. Industrial Research. Vol. 4, pp.14–19 (1962).
[26] T. B¨ack and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evol. Comput.*, vol. 1, no. 1, pp. 1–23 (1993).
[27] D. B. Fogel.: Evolutionary Computation: Toward a new Philosophy of Machine Intelligence. IEEE press (1995).
[28] Hao, Z-F, Gua, G-H, Huang, H.; A Particle Swarm Optimization Algorithm with Differential Evolution. Sixth International conference on Machine Learning and Cybernetics, pp. 1031 – 1035 (2007).
[29] Omran, Mohd. G.H., Engelbrecht, A. P., Salman, Ayed.: Differential Evolution based Particle Swarm Optimization. IEEE Swarm Intelligence Symposium (SIS 2007), pp. 112 – 119 (2007).
[30] W-J. Zhang and X-F. Xie.: DEPSO: Hybrid Particle Swarm with Differential Evolution Operator. IEEE International Conference on Systems, Man & Cybernetics (SMCC), Washington D C, USA, pp. 3816-3821 (2003).