Self-adaptive of Differential Evolution using Neural Network with Island Model of Genetic Algorithm

Linh Tao[1] Hieu Pham[2] Hiroshi Hasegawa[3]

[1]D. Functional Control System, Shibaura Institute of Technology, Japan, nb14505@shibaura-it.ac.jp
[2]National Institute of Patent and Technology Exploitation, Vietnam, hieupn@most.gov.vn
[3]D. Functional Control System, Shibaura Institute of Technology, Japan, h-hase@shibaura-it.ac.jp

Abstract

A new evolutionary algorithm called NN-DEGA that using Artificial Neural Network (ANN) for Self-adaptive Differential Evolution (DE) with Island model of Genetic Algorithm (GA) is proposed to solve large scale optimization problems, to reduce calculation cost, and to improve stability of convergence towards the optimal solution. This is an approach that combines the global search ability of DE and the local search ability of Adaptive System with Island model of GA. The proposed algorithm incorporates concept from DE, GA, and Neural Networks (NN) for self-adaptive of control parameters. The NN-DEGA is applied to several benchmark tests with multi-dimensions to evaluate its performance. It is shown to be statistically significantly superior to other Evolutionary Algorithms (EAs), and Memetic Algorithms (MAs).

Keywords: differential evolution, memetic algorithm, migration, neural network, parallel genetic algorithm

1 Introduction

To solve complex numerical optimization problems, researchers have been looking into nature both as model and as metaphor for inspiration. A keen observation of the underlying relation between optimization and biological evolution led to the development of an important paradigm of computational intelligence for performing very complex search and optimization. Evolutionary Computation uses iterative process, such as growth or development in a population that is then selected in a guided random search using parallel processing to achieve the desired end. Nowadays, the field of nature-inspired metaheuristics is mostly continued by the Evolution Algorithms (EAs) (e.g., Genetic Algorithms (GAs), Evolution Strategies (ESs), and Differential Evolution (DE) etc.) as well as the Swarm Intelligence algorithms (e.g., Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), etc.). Also the field extends in a broader sense to include self-organizing systems, artificial life, memetic and cultural algorithms, harmony search, artificial immune systems, and learnable evolution model. The GAs (Goldberg, 1989; Holland, 1992) have been applied to various complex computational problems, and its validity has been reported by many researchers (Goldberg, 1999; Mahfoud, 1992). However, it requires a huge computational cost to obtain stability in convergence towards an optimal solution. To reduce the cost and to improve the stability, a strategy that combines global and local search methods becomes necessary. As for this strategy, current research has proposed various methods (Nasa, 2016). For instance, Memetic Algorithms (MAs) (Ong, 2004; Smith, 2005) are a class of stochastic global search heuristics in which EAs-based approaches are combined with local search techniques to improve the quality of the solutions created by evolution. MAs have proven very successful across the search ability for multi-modal functions with multi-dimensions (Ong, 2004). These methodologies need to choose suitably a best local search method from various local search methods for combining with a global search method within the optimization process. Furthermore, since genetic operators are employed for a global search method within these algorithms, design variable vectors (DVs) which are renewed via a local search are encoded into its genes many times at its GA process. These certainly have the potential to break its improved chromosomes via gene manipulation by GA operators, even if these approaches choose a proper survival strategy. To solve these problems and maintain the stability of the convergence towards an optimal solution for multi-modal optimization problems with multiple dimensions, Hieu Pham et al. proposed evolutionary strategies of Adaptive Plan system with Genetic Algorithm (APGAs) (Pham, 2012). It is shown to be statistically significantly superior to other EAs and MAs. Unlike most other techniques, GAs maintain a population of tentative solutions that are competitively manipulated by applying some variation operators to find a global optimum. For non-trivial problems, this process might require high computational resources such as large memory and search times. To design efficient GAs, a variety of advances by new operators, hybrid algorithms, termination criteria, and more are continuously being achieved. Parallel GAs (PGAs) (Alba, 1999; Cant, 1998; Tanese, 1989) often leads to superior numerical performance not only to faster algorithms. However, the truly interesting observation is that the use of structured population, either in the form of a set of islands or a diffusion grid, is responsible for such numerical benefits. A PGA has the same as a serial GA, consisting in using rep-

resentation of the problem parameters, robustness, easy customization, and multi-solution capabilities. In addition, a PGA is usually faster, less prone to finding suboptimal solutions only, and able of cooperating with other search techniques in parallel. Differential Evolutionary (DE) was recently introduced and has garnered significant attention in the research literature (Storn, 1997). DE has many advantages including simplicity of implementation, reliable, robust, and in general is considered as an effective global optimization algorithm (Price, 2005). DE operates through similar computational steps as employed by a standard EA. However, unlike traditional EAs, the DE variants perturb the current generation population members with the scaled differences of randomly selected and distinct population members. Therefore, no separate probability distribution has to be used for generating the offspring (Das, 2011). Recently, DE has drawn the attention of many researchers all over the world resulting in a lot of variants of the basic algorithm with improved performance such as Improved Self-adaptive Differential Evolution (ISADE) used in (Bui, 2015) and Advanced DE (ADE) (Mohamed, 2011) etc. (Brest, 2006; Liu, 2005; Mohamed, 2011; Noman, 2008; Omran, 2007; Xu, 2009). Compared with and other techniques (Vesterstroem, 2004), it hardly requires any parameter tuning and is very efficient and reliable. In this paper, we purposed a new evolutionary algorithm called NN-DEGA that using Artificial Neural Network (ANN) for Self-adaptive DE with Island model of GA to solve large scale optimization problems, to reduce a large amount of calculation cost, and to improve the convergence towards the optimal solution.

2 New Evolutionary Computation

2.1 Island model parallel distributed in NN-DEGA

Migration PGA, island model, such as those described in Sect. 1, are reported to have greater information compatibility, a stable design and low computational costs because they deal with GAs in parallel. In NN-DEGA, optimization is conducted by applying GA and DE to each subpopulation. The control variables adjust the vicinity of the output constriction factor F between the subpopulations. The candidate control variables and the new solution come from the other subpopulation at the time of immigration, so a diversity of solutions can be expected because the migration destination is determined at random. A schematic diagram of NN-DEGA with PGA migration is shown in Figure 1.

2.2 Self-adaptive using Neural Network

The self-adaptive constriction factor $F(NN)$ is used for data clustering of the GA control variables using NN, which have been determined uniquely to stabilize their variation. From a viewpoint of excellent parallel processing and to ensure compatibility with multi-point search

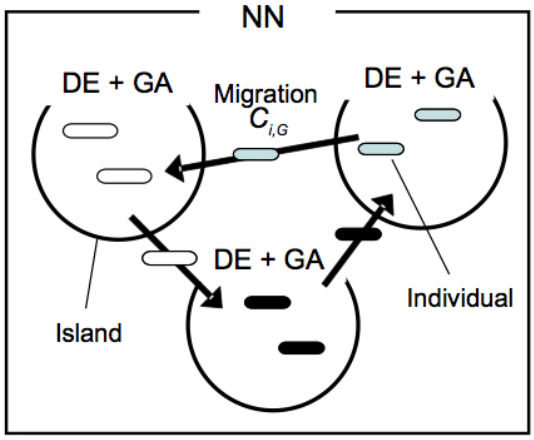

Figure 1. Island Model GA conceptual diagram in NN-DEGA.

methods such as GA and DE are also used in the present method. NN is often used in combination with these techniques (Kobayashi, 2007). NN may be used to cluster and classify the data without using a signal if it is necessary to learn using a teacher signal that is also a NN. In the present method, the GA variable data clustering is controlled using unsupervised learning to determine the output scaling factor change. The initial constriction factor F is set at random and we vary its value based on the NN output. The unsupervised learning method is also a multi-layer NN, so we use NN to perform the feed-forward transfer. The number of layers is determined in a number of search points for each subpopulation. In addition, the NN is configured after it has been sorted in descending order of fitness in the subpopulations to the output side from the input side, where the weight of the transfer equation is as shown in (1). Therefore, many subpopulations have highly adaptive search points with strong effects on other subpopulations. The formulation of the control variable, the transfer equation for each node in the NN and the schematic diagram of the overall NN are as follows.

$$w_{j^n i^n} = y_{nm} / y_{(n-1)m} \tag{1}$$

$$node_t = \sum_{i=1}^{I} SP \cdot w_{i^n j^n} out_i^{n-1} / I \tag{2}$$

$$SP = 2 \cdot C_{i,G} - 1 \tag{3}$$

$$C = [c_{i,j}, \ldots, c_{i,p}]; (0.0 \leq c_{i,j} \leq 1.0) \tag{4}$$

$$F_{i,G+1} = F_{i,G} - \nabla F_i \tag{5}$$

The GA handles control variables (CVs) and C_t is allocated to each search point, which is encoded as a 10-bit string. The order of each search point is allocated to each node of a multi-layer NN, as shown in Figure 2, on the input side and the output side. The weight of the NN, $w_{j^n i^n}$, which is determined from the adaption ratio of the search

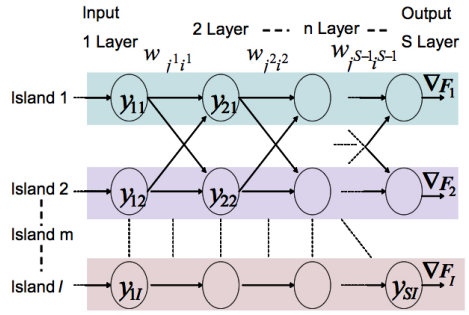

Figure 2. NN-DEGA neural network.

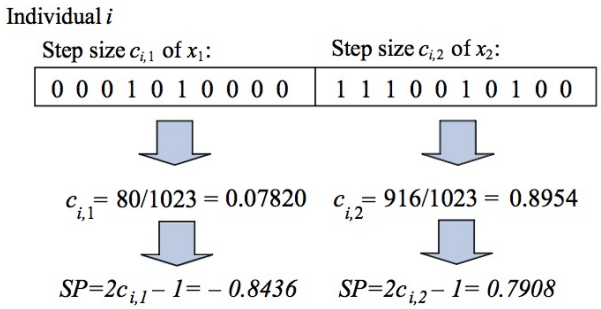

Figure 3. Step size that defined by CVs for controlling a global behavior to prevent it falling into the local optimum.

points, is transmitted between the nodes. C_t is the control variable that determines the step size SP in (3) (as shown in Figure 3) and this element determines the extent of the constraint factor change, ∇F. Therefore, the constriction factor change is an important factor, which determines the width of the overall distribution of the neighborhood of search points. Using the control variable, we can change F adaptively to facilitate more stable solution search and better control of the control variable in the NN. In addition, n is the number of NN hierarchical levels, m is the number of subpopulations, j, i is the number of neurons in NN, t is the number of individuals, S is the number of searches per island and I is the maximum number of islands.

2.3 Reconstruction of differential vector

Each target vector aims at the global optimal solution by updating differential vector based on its best solution has been achieved so far $pbest_{ij}$ and the best solution of all individuals in the population $gbest_j$ (where $j = [1,2,\ldots,D]$, D is the dimension of the solution vector), as following equation:

$$V_{ij,G+1} = gbest_{j,G} + F \cdot \left(pbest_{ij,G} - X_{ij,G}\right) \qquad (6)$$

We carried out the reconstruction of the control variable like considered control variables APGAs (Pham, 2012),

Algorithm 1 The NN-DEGA Pseudocode

```
Initialize population with CVs;
Generate initial DVs;
Evaluate individuals with initial DVs;
while (Termination Condition) do
    Adaptive control of scaling factor F = F(NN) using Neural network;
    Generate DVs via AP with new DE scheme:
    Generate a mutant vector: V_{ij,G+1} = gbest_{j,G} + F(NN) · (pbest_{ij,G} − X_{ij,G});
    Generate a trial vector U_{ij,G+1} through binomial crossover:
    U_{ij,G+1} = { V_{ij,G+1}, (rand_j ≤ CR) or (j = j_rand)
                 { X_{ij,G+1}, (rand_j ≥ CR) and (j ≠ j_rand)   ;
    Evaluate the trial vector U_{i,G};
    if f(U_{i,G}) ≤ f(X_{i,G}) then X_{i,G+1} = U_{i,G} else X_{i,G+1} = X_{i,G};
    end if
    Evaluate individuals with DVs;
    Select parents;
    Recombine to produce offspring for CVs;
    Mutate offspring for CVs;
    if (Restructuring Condition) then
        Restructure chromosome of offspring for CVs;
    end if
end while
```

not only control variable meet the conditions listed below, but also reconstruction of the DE differential vector by keep performing keep the global search of the search point, the appropriate solution search is always performed.

- The same value adaptation accounted for more than 80% for the entire
- The same bit-string chromosome occupies more than 80% for the entire
- The same value of scaling factor accounted for 50% of the total.

2.4 Elite strategy

In this paper, using the diploid genetics is not proper to perform the search using the NN solution (Kouchi, 1992). Generally, GA, information has only a single gene for one individual. However, the structure has a double recessive genetic information that does not appear in the dominant phenotype. Here, in NN, genetic information is treated as a control variable. Information dominance for the NN is elite solution closed to the control variable, as shown in the following equation. With the aim of having a strong influence in the form of dominant inheritance, enhancing the effectiveness of the control variable, advantageously advancing the solution search, elite solution against other sub-populations as the island model of GA.

$$\begin{array}{ll} if\ |eSP - SP_1| - |eSP - SP_2| < 0 & SP = SP_1 \\ if\ |eSP - SP_1| - |eSP - SP_2| > 0 & SP = SP_2 \end{array} \qquad (7)$$

3 Numerical Experiments

In this section, the numerical experiments were performed to compare among strategies. Next, the new algorithm were compared with other techniques for the robustness

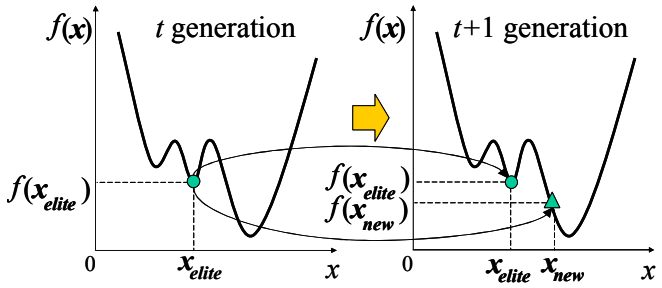

Figure 4. Elite strategy, where the best individual survives in the next generation, is adopted during each generation process.

Table 1. Parameter Settings for Benchmark Tests.

Operator	Control Parameter	Set value
DE	Scaling factor	$F(NN) \in [-1.0, 1.0]$
	Crossover probability	$CR = 0.5$
GA	Selection ratio	1.0
	Crossover ratio	0.8
	Mutation ratio	0.1

The population size: 100

of the optimization approach. These experiments involved 25 independent trials for each function. The parameter settings used in solving the benchmark functions are given in Table 1. The initial seed number are randomly varied during every trial. The scaling factor F takes value in $[-1.0, 1.0]$, and the crossover probability CR is set by 0.5 for the performance of DE. The GA parameters, selection ratio, crossover ratio and mutation ratio are 1.0, 0.8 and 0.01, respectively. The population size has 100 individuals.

3.1 Benchmark Functions

For the NN-DEGA, we estimated the stability of the convergence to the optimal solution by using five benchmarks with 30, and 100 dimensions - Ridge (f_3), Rosenbrock (f_5), Rastrigin (f_9), Ackley (f_{10}), and Griewank (f_{11}). Table 2 lists their characteristics, including the terms epistasis, multi-peaks, and steepness. D denotes the dimensionality of the test problem, design range variables and the global optimum value are summarized. A more detailed description of each function is given in Ref. (Yao, 1999). All functions are minimized to zero, when optimal DVs $X = 0$ are obtained. Note that, it is difficult to search for optimal solutions by applying one optimization strategy only, because each function has specific complex characteristics. The search process is terminated when the search point attains an optimal solution or a current generation process reaches the termination.

$$f_3 = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{8}$$

Table 2. Characteristics of benchmark tests.

Func	Epis	M-peaks	Steepness	Design range	Optimum
f_3	Yes	No	Average	$[-100, 100]^D$	$f(0) = 0$
f_5	Yes	No	Big	$[-30, 30]^D$	$f(0) = 0$
f_9	No	Yes	Average	$[-5.12, 5.12]^D$	$f(0) = 0$
f_{10}	No	Yes	Average	$[-32, 32]^D$	$f(0) = 0$
f_{11}	Yes	Yes	Small	$[-600, 600]^D$	$f(0) = 0$

D denotes the dimensionality of the test problem.
"Epis" stands for Epistatic, "M - peak" stand for Multi - peaks.

$$f_5 = \sum_{i=1}^{D} [100(x_{i+1} + 1 - (x_i + 1)^2)^2 + x_i^2] \tag{9}$$

$$f_9 = 10D + \sum_{i=1}^{D} [x_i^2 - 10cos(2\pi x_i)] \tag{10}$$

$$f_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e \tag{11}$$

$$f_{11} = 1 + \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{12}$$

3.2 Experiment Results

The experiment results, average generations required to reach the global optimum of all benchmark functions with 30 dimensions in term of 150,000 FES by the NN-DEGA are given in Tables 3 - 6. "Mean best" indicates average of optimum values obtained and "Std Dev" stands for standard deviation. The solution of all benchmark functions reach their global optimum solutions, and the success rate of optimal solution is 100%. The effect of island number and immigration rate on the performance of algorithm is reported. From this results via optimization experiments, it can be concluded that either the island number or immigration rate increase, the performance of the NN-DEGA algorithm significantly improves. Additionally, the results show that the NN-DEGA algorithm is effective in all benchmarks for various island number and immigration rate, which suggests that the NN-DEGA is more stable and robust on island model using neural network. We employed the best value of island number and immigration rate for the NN-DEGA are 10, 0.2 respectively. In addition, the experiment results with 100 dimension in term of fixed total evaluation times 500,000 FES are given in Table 7. When the success rate of optimal solution is not 100%, "-" is described. We confirmed that the NN-DEGA could solve multi-modal functions with high probability. As a result, its validity confirms that this strategy can dramatically reduce the computation cost and improve the stability of the convergence to the optimal solution more significantly.

Table 3. Experiment results, average generations required to reach the global optimum over 25 runs ($D = 30$, population size 100, island number 5 and imigration rate 0.05).

Function	Gen. No	NFE	Mean best	Std Dev
f_3	571	57,100	0.000E+000	0.000E+000
f_5	147	14,700	0.000E+000	0.000E+000
f_9	61	6,100	0.000E+000	0.000E+000
f_{10}	101	10,100	4.441E-016	0.000E+000
f_{11}	77	7,700	0.000E+000	0.000E+000

Table 4. Experiment results, average generations required to reach the global optimum over 25 runs ($D = 30$, population size 100, island number 5 and imigration rate 0.2).

Function	Gen. No	NFE	Mean best	Std Dev
f_3	380	38,000	0.000E+000	0.000E+000
f_5	95	9,500	0.000E+000	0.000E+000
f_9	45	4,500	0.000E+000	0.000E+000
f_{10}	71	7,100	4.441E-016	0.000E+000
f_{11}	58	5,800	0.000E+000	0.000E+000

Table 5. Experiment results, average generations required to reach the global optimum over 25 runs ($D = 30$, population size 100, island number 10 and imigration rate 0.05).

Function	Gen. No	NFE	Mean best	Std Dev
f_3	585	58,500	0.000E+000	0.000E+000
f_5	129	12,900	0.000E+000	0.000E+000
f_9	61	6,100	0.000E+000	0.000E+000
f_{10}	99	9,900	4.441E-016	0.000E+000
f_{11}	73	7,300	0.000E+000	0.000E+000

Table 6. Experiment results, average generations required to reach the global optimum over 25 runs ($D = 30$, population size 100, island number 10 and imigration rate 0.2).

Function	Gen. No	NFE	Mean best	Std Dev
f_3	394	39,400	0.000E+000	0.000E+000
f_5	83	8,300	0.000E+000	0.000E+000
f_9	43	4,300	0.000E+000	0.000E+000
f_{10}	66	6,600	4.441E-016	0.000E+000
f_{11}	56	5,600	0.000E+000	0.000E+000

Table 7. Experiment results, average generations required to reach the global optimum over 25 runs in term of 500,000 FES ($D = 100$, population size 100, island number 10 and imigration rate 0.2).

Function	Gen. No	NFE	Mean best	Std Dev
f_3	-	-	-	-
f_5	226	22,600	0.000E+000	0.000E+000
f_9	110	11,000	0.000E+000	0.000E+000
f_{10}	232	23,200	4.441E-016	0.000E+000
f_{11}	154	15,400	0.000E+000	0.000E+000

Table 8. Comparison of DE, jDE, ADE and NN-DEGA algorithm in term of 150,000 FES; Gen. No 1500 ($D = 30$, population size=100).

Func.	Gen. No	DE	jDE	ADE	NN-DEGA
		Mean best (Std Dev)	Mean best (Std Dev)	Mean best (Std Dev)	Mean best (Std Dev)
f_3	1500	1.630860 (0.886153)	0.090075 (0.080178)	-	0.000E+000 (0.000E+000)
f_5	1500	7.8E-09 (5.8E-09)	3.1E-15 (8.3E-15)	3.75E-05 (1) (8.90E-05)	0.000E+000 (0.000E+000)
f_9	1500	173.405 (13.841)	1.5E-15 (4.8E-15)	0.0E+00 (2) (0.0E+00)	0.000E+000 (0.000E+000)
f_{10}	1500	9.7E-08 (4.2E-08)	7.7E-15 (1.4E-15)	6.93E-11 (3.10E-11)	4.441E-016 (0.000E+000)
f_{11}	1500	2.9E-13 (4.2E-13)	0 0	0.0E+00 (3) (0.0E+00)	0.000E+000 (0.000E+000)

(1) Gen. No 3000; (2) Gen. No 5000; (3) Gen. No 2000

3.3 Comparison for Robustness

To evaluate the performance of the NN-DEGA algorithm, we compared to other EAs such as GA, PSO, PS-EA in (Srinivasan, 2010), ABC (Karaboga, 2006), DE (Storn, 1997), jDE (Brest, 2006), and ADE (Mohamed, 2011). Maximum number of generation and the population size, i.e. 100, as in the study presented in (Vesterstroem, 2004; Srinivasan, 2010). The mean and the standard deviations of the function values obtained by these methods are given in Tables 9 and 8. By means of the comparison with other methodologies, the NN-DEGA could certainly achieve optimal solution with low calculation cost. Additionally, the results show that the proposed NN-DEGA algorithm outperformed other techniques in all function. The convergence of the optimal solution could be improved more significantly in the NN-DEGA than that in other methods for the same calculation cost. Therefore, it is desirable to introduce this strategy for global optimization.

4 Conclusions

In this paper, overcome the computational complexity, a new evolutionary strategy that using Artificial Neural Network for Self-adaptive Differential Evolution with Island model of Genetic Algorithm called NN-DEGA is proposed to solve large scale optimization problems, to reduce a large amount of calculation cost, and to improve the convergence to the optimal solution. Then, we verified the effectiveness of the NN-DEGA algorithm by the numerical experiments performed five benchmark tests. Moreover, the NN-DEGA was compared to other EAs, it shown to be statistically significantly superior to other EAs. We confirmed that the NN-DEGA reduces the calculation cost and dramatically improves the convergence towards the optimal solution. Moreover, it could solve large scale optimization problems with high probability. About a solution of the problem of cost reduction, minimum time and maximum reliability, it is a future work. Finally, this study plans to do a comparison with the sensitivity plan of the AP by applying other methods on constrained real-parameters and dynamic optimization problems, and further real-life applications.

Table 9. Comparison of GA, PSO, PS-EA, ABC and NN-DEGA algorithm in term of 100,000 FES; Gen. No 1000 ($D = 30$, population size=100).

Func.	Gen. No	GA	PSO	PS-EA	ABC	NN-DEGA
		Mean best (Std Dev)	Mean best (Std Dev)	Mean best (Std Dev)	Mean best (Std Dev)	Mean best (Std Dev)
f_3	1000	-	-	-	-	0.000E+000 (0.000E+000)
f_5	1000	166.283 (59.5102)	402.54 (633.65)	98.407 (35.5791)	0.219626 (0.152742)	0.000E+000 (0.000E+000)
f_9	1000	10.4388 (2.6386)	32.476 (6.9521)	3.0527 (0.9985)	0.033874 (0.181557)	0.000E+000 (0.000E+000)
f_{10}	1000	1.0989 (0.24956)	1.49E-6 (1.86E-6)	0.3771 (0.098762)	3E-12 (5E-12)	4.441E-016 (0.000E+000)
f_{11}	1000	1.2342 (0.11045)	0.011151 (0.014209)	0.8211 (0.1394)	2.87E-09 (8.45E-10)	0.000E+000 (0.000E+000)

References

E. Alba and J.M. Troya. A Survey of Parallel Distributed Genetic Algorithms, *Journal Complexity*, 4(4): 31–52,1999.

J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Trans. Evol. Comput.*, 10(6): 646–657, 2006.

Ngoc Tam Bui and Hiroshi Hasegawa. Training Artificial Neural Network Using Modification of Differential Evolution Algorithm. *International Journal of Machine Learning and Computing*, 5(1): 1-6, 2015.

E. Cant. A survey of parallel genetic algorithms, *Calculateurs paralleles, reseaux et systems repartis*, vol 10,1998.

S. Das and P.N. Suganthan. Differential evolution - A survey of the State-of-the-Art, *IEEE Transactions on Evolutionary Computation*, 15(1): 4–31, 2011.

D.E. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*, Addison - Wesley, 1989.

D. E. Goldberg and S. Voessner. Optimizing global-local search hybrids, *In Proceedings of 1999 Genetic and Evolutionary Computation Conference*, pages 220–228, 1999.

J. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan 1975, MIT Press, 1992.

D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *Journal Global Optimization*, 39: 459–471, 2006.

K. Kobayashi, T. Hiroyasu, and M. Miki. Mechanism of Multi-Objective Genetic Algorithm for Maintaining the Solution Diversity Using Neural Network, *The Science and Engineering Review of Doshisha University*, 48(2): 24–33, 2007.

M. Kouchi, H. Inayoshi, and T. Hoshino. Optimization of Neural-Net Structure by Genetic Algorithm with Diploydi and Geographical Isolation Model, *Japanese Society for Artificial Intelligence*, 7(3): 509–517, 1992.

J. Liu and J. Lampinen. A fuzzy adaptive differential evolution algorithm, *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 9(6): 448–462, 2005.

S. W. Mahfoud and D. E. Goldberg. Parallel recombinative simulated annealing: A genetic algorithm, *Parallel Computing*, 21(1): 1–28, 1995.

A. Wagdy Mohamed, H.Z. Sabry, and A. Farhat. Advanced Differential Evolution algorithm for global numerical optimization, *In IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pages 156–161, 2011.

N. Noman and H. Iba. Accelerating differential evolution using an adaptive local Search, *IEEE Transactions on Evolutionary Computation*, 12(1): 107–125, 2008.

Nasa Publications `http://ti.arc.nasa.gov/tech/rse/publications/`

M. Omran, A.P. Engelbrecht, and A. Salman. Empirical analysis of self-adaptive differential evolution, *European Journal of Operations Research*, 183(2): 785–804, 2007.

Y.S. Ong and A.J. Keane. Meta-Lamarckian Learning in Memetic Algorithms, *IEEE Transactions on Evolutionary Computation*, 8(2): 99–110, 2004.

K. Price, R. Storn, and J. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, Berlin, 2005.

Hieu Pham, S. Tooyama, and H. Hasegawa. Evolutionary Strategies of Adaptive Plan System with Genetic Algorithm, *JSME Journal of Computational Science and Technology*, 6(3): 129–146, 2012.

D.E. Rumelhart and J.L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, MIT Press, 1986.

D. Srinivasan and T.H. Seow. Evolutionary Computation, *IEEE Congress on Modelling and Simulation*, 2010.

J.E. Smith, W.E. Hart, and N. Krasnogor. *Recent Advances in Memetic Algorithms*, Springer, 2005.

R. Storn and K. Price. Differential Evolution - a simple and efficient Heuristic for global optimization over continuous spaces, *Journal Global Optimization*, 11(4): 341–357, 1997.

R. Tanese. Distributed genetic algorithms, *In Proc. of 3rd Int. Conf. on Genetic Algorithms*, pages 434–439, 1989.

J. Vesterstroem and R. Thomsen. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, *In Proc. IEEE Congr. Evolutionary Computation*, pages 1980-1987, 2004.

Y. Xu, L. Wang, and L. Li. An effective hybrid algorithm based on simplex search and differential evolution for global optimization, *In Proc. ICIC*, pages 341–350, 2009.

X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster, *IEEE Trans. Evol. Comput.*, 3(2): 82–102, 1999.

X. Yao. Evolving artificial neural networks, *In Proceedings of the IEEE*, 87: 1423-1447,1999.