I.J. Intelligent Systems and Applications, **2015, 09, 12-19**
Published Online August 2015 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2015.09.02

Individually Directional Evolutionary Algorithm for Solving Global Optimization Problems- Comparative Study

Łukasz Kubuś
Department of Computer Science Applications, Kielce University of Technology, Kielce, Poland
Email: lkubus@tu.kielce.pl

Abstract— Limited applicability of classical optimization methods influence the popularization of stochastic optimization techniques such as evolutionary algorithms (EAs). EAs are a class of probabilistic optimization techniques inspired by natural evolution process, witch belong to methods of Computational Intelligence (CI). EAs are based on concepts of natural selection and natural genetics. The basic principle of EA is searching optimal solution by processing population of individuals. This paper presents the results of simulation analysis of global optimization of benchmark function by Individually Directional Evolutionary Algorithm (IDEA) and other EAs such as Real Coded Genetic Algorithm (RCGA), elite RCGA with the one elite individual, elite RCGA with the number of elite individuals equal to population size. IDEA is a newly developed algorithm for global optimization. Main principle of IDEA is to monitor and direct the evolution of selected individuals of population to explore promising areas in the search space. The idea of IDEA is an independent evolution of individuals in current population. This process is focused on indicating correct direction of changes in the elements of solution vector. This paper presents a flowchart, selection method and genetic operators used in IDEA. Moreover, similar mechanisms and genetic operators are also discussed.

Index Terms— Individually Directional Evolutionary Algorithm, Evolutionary Algorithm, Evolutionary Computing, Directed Mutation, Global Optimization

I. Introduction

Evolutionary algorithms (EAs) are a class of probabilistic optimization techniques inspired by natural evolution process, witch belong to methods of computational intelligence (CI). Classical optimization methods require that the objective function have some properties such as continuity or are differentiable. These requirements make use of conventional methods of optimization for real-world problems difficult or impossible. Limited applicability of classical optimization methods influence the popularization of stochastic optimization techniques [1,10]. Optimization algorithms such as EAs are used in artificial intelligence (AI) to optimize the structure of neural networks [6, 15] or adapt the weights for fuzzy cognitive map [11 - 13].

EAs includes techniques such as genetic algorithms (GA), evolution strategies (ES), evolutionary programming (EP) and genetic programming (PG). All these techniques have been developed in different research centers. Common name helps to avoid problems with classification of new optimization techniques similar to more than one technique of EAs. Evolutionary Computing (EC) is other common name for EA.

Popularity of EAs has led to a many of modifications of them [2, 8]. These modifications can be divided into several groups: modifications based on intermixing techniques of EAs, modifications independent from representation of solutions and modifications designed to solve specific problem or group of problems. The first group includes modifications such as adding a genetic operator known as a crossover, developed for GA, to ES or using a floating-point representation in GA, known from ES. The second group of modifications contains elements algorithm used by EAs such as alternative selection methods - tournament selection, rank based selection or threshold selection. Specific encoding scheme and genetic operators are in last group of modifications. EAs together with other modifications from the last group are applied to solve travelling salesman problem. These EAs use integer based encoding scheme and genetic operators, that are processing this encoding scheme.

EAs are based on concepts of natural selection and natural genetics. The basic principle of EA is searching optimal solution by processing population of individuals. This process leads to increasing the average value of fitness function of the population in subsequent cycles of the algorithm. Selection methods and genetic operators allow to choose individuals, which represent better solution and exchange information among each other. Selection methods are similar to the natural evolution process and genetic operators are similar to the reproduction of individuals in natural environment. The newly created individuals represent new solutions. Better rated solutions have more chance to transition to the next population. Elite selection methods allow to accelerate the convergence of algorithm. Elite scheme allows the survival of the fittest individuals.

EAs attempt to imitate the process of evolution described by Darwin. The process of function optimization is reduced to the adaptation of a species to the natural environment in successive generations. Improving the value of average fitness function of the population is realized by selection methods and genetic operators. This process assumes that indirectly created

Copyright © 2015 MECS

worse solutions can lead to a better final solution. Subsequent created populations contain the majority of newly created individuals, that prevents monitoring of the evolution of each individual.

IDEA is a newly developed algorithm for global optimization. Main principle of IDEA is to monitor and direct the evolution of selected individuals of population to explore promising areas in the search space. The algorithm uses a proportional selection method with dynamic linear scaling of fitness function and non-uniform mutation uses information about the direction of change also known as direction vector. Applied selection method and mutation operator in IDEA allows to choose the fittest individuals and their modification to explore promising areas in the search space. The idea of IDEA is an independent evolution of individuals in current population. An independent evolution process is focused on indicating correct direction of changes in the elements of solution vector.

The paper is organized as follows. Section 2 contain information about mutation operators applied in EAs. This information relates mainly to the directed mutation operators. Section 3 is description of designed algorithm IDEA. The set of benchmark problems, the compared algorithms and the results of simulation analysis are reported in Section 4. Finally, Section 5 presents a number of conclusions from the present study.

II. Related Work

Classical mutation operator used in the binary coded genetic algorithm (BCGA) is changes randomly selected bits of binary string, that represent each individual. Similar technique is used for RCGA - selected digits of each element of the solution vector randomly changes. However, RCGA allows the use of more sophisticated techniques to modify the current solution than random mutation. Common random mutation operators for RCGA are non-uniform mutation (NUM), uniform mutation (also known as random mutation), power mutation (PM) or boundary mutation (BM).

These operators are purely random - the change of each element is performed independently of previous or other changes. Another type of mutation operators are directional mutation operators (DMO). These operators take into account the impact of mutation on fitness function of individuals. Pointed Directed (PoD) [3] Mutation requires to complement representation of solution by binary direction vector (DV) for each individual. This vector contains a single bit for each gene, which dictates the direction of mutation. The DV is corrected by feedback from the current population. PoD has been modified and named mutation-with-momentum [4]. The mutation-with-momentum algorithm requires additional data to be stored for each gene in an individual's genetic representation. These additional data represent a change of gene value in most recent mutation. The directional mutation operator proposed in [7] is based on distribution of individuals over intervals of each dimension. The statistics information is used to direct the mutation of an individual toward the neighboring interval. Another example of a directional mutation is a mutation based on the α-stable distributions [9]. Adaptive directed mutation (ADM)[14] uses the impact of change values of the solutions vector to value of fitness function of each individual in subsequent generations.

This paper suggest the use of directional non-uniform mutation operator (DNUM). DNUM is similar to NUM operator [8] but direction vector is used to modify individual genes. As a result of DNUM work, a single gene of an individual has been modified in accordance with the direction vector. This technique allows to explore the search space in accordance with direction that produced a better solution in previous generations. In the case when the mutation has not led to the creation of better solution, the corresponding value of DV is adjusted.

III. Individually Directional Evolutionary Algorithm

Individually directional evolutionary algorithm (IDEA) uses characteristic elements of EAs such as proportionate selection (roulette-wheel selection) with dynamic linear scaling of fitness function in pre-selection stage, directional non-uniform mutation operator (DNUM) as genetic operator and elite scheme in post-selection stage. Figure 1 shows the scheme of IDEA.

```
procedure IDEA
begin
   t:=0
   initialization P^0
   evaluation P^0
   P^t:=P^0
   while (not stop condition) do
   begin
         T^t:=pre-selction P^t
         T'^t:= directional non-uniform mutation T^t
         evaluation T'^t
         P^(t+1):=post-selection (T'^t, T^t)
         t:=t+1
   end
   end
```

Fig. 1. IDEA scheme.

Evolution loop of IDEA is initialized in the same way as in other EAs and is made up of 4 steps. In the first step, temporary population T is created from current base population P. Individuals of population T will be subject to mutation. In the second step, another temporary population T' is created by mutations of individuals of temporary population T. Individuals of temporary population T' are evaluated in third step of evolution loop. The last step of evolution loop is to select individuals from temporary population T and T', that will create the next base population P.

Copyright © 2015 MECS

A. Solution encoding

The developed algorithm IDEA uses solution encoding similar to algorithms such as RCGA or ES expanded by the additional direction vector (DV). Each individual is represented by two n-dimensional vectors (1).

$$O = < X, D > \tag{1}$$

where:

$$X = [x_1, x_2, x_3, ..., x_n],$$
$$D = [d_1, d_2, d_3, ..., d_n],$$
$$x_i \in R^n, i = 1,2,3,...,n,$$
$$d_i \in \{-1,1\}, i = 1,2,3,...,n.$$

The vector X represents a potential problem solution. The vector D contains information about mutation direction. This vector is used in the mutation operation.

B. Pre-selection process

Pre-selection procedure used by IDEA is a classic mechanism of proportionate selection with dynamic linear scaling of fitness function [5]. The value of the fitness function is decreased by the value of f_0^t (2).

$$f_0^t = \begin{cases} 0.99 \cdot f_{\min}^t \ \text{for} \ f_{\min}^t > 0 \\ 1.01 \cdot f_{\min}^t \ \text{for} \ f_{\min}^t \le 0 \end{cases} \tag{2}$$

where: $f_{\min}^t$ - fitness function value of "the worst" individual of population P^t.

The accepted method of calculating the value of f_0^t allows obtain a non-zero probability of select the worst individual from population.

C. Directional non-uniform mutation

Directional non-uniform mutation operator is a modified non-uniform mutation operator [8]. In the case of DNUM, the mutation of selected gene is carried out in accordance with the value of corresponding element of directional vector (3).

$$y_i = \begin{cases} x_i - \Delta(t, x_i - l_i) \ \text{for} \ d_i = -1 \\ x_i + \Delta(t, u_i - x_i) \ \text{for} \ d_i = 1 \end{cases} \tag{3}$$

d_i - mutation direction of the i-th gene of the individual,

x_i - current value of the i-th gene of the individual,

y_i - new value of the i-th gene of the individual.

In effect of DNUM is one and only one element of solution vector of any individual that has been modified. This allows to correct the direction of mutation of any gene for each individual in the post-selection process.

D. Post-selection process

Post-selection process is similar to the greedy selection [17,18]. In greedy selection, the new better individual replaces the current worse individual. New individuals are created by the mutation of following genes. In post-selection process, the individual of temporary population T is compared to the corresponding individual from temporary population T'. The corresponding individual from temporary population T' was created by the mutation the individual from temporary population T. Each individual of population T has one and only one corresponding individual in temporary population T'. The difference between these individuals is only the one mutated gene value. One of these individuals, (better rated) is selected for the next base population. If the mutated individual is worse than the corresponding individual, the corresponding element of directional vector of the primary individual will be corrected. The value of the corresponding element of directional vector is corrected by multiplying this value by -1 (4).

$$d_i = d_i \cdot (-1) \tag{4}$$

d_i - mutation direction of the i-th gene of the individual.

This process is called the process of correction of mutation direction and allows to change values of solution vector according to the direction, which previously produced good results.

IV. Simulation Analysis -Comparative Study

The subject of simulation analysis is a global optimization of selected multidimensional benchmark functions by selected EAs.

A typical nonlinear global optimization problem is defined as (5).

$$\min_{\max} f(x), were \ x \in S \ \ and \ \ S \in R^n. \tag{5}$$

The S is the search space. The problem is find value of x^* from S, which minimizes/maximizes $f(x)$. The x^* must comply with the conditions (6) for minimizing or maximizing $f(x)$.

$$\begin{aligned} &\min f(x): f(x^*) \le f(x) \ for \ \forall x \in S \\ &\max f(x): f(x^*) \ge f(x) \ for \ \forall x \in S \end{aligned} \tag{6}$$

Problem of minimizing $f(x)$ can be transformed into a maximizing $f(x)$ (7).

$$x^* = \arg\min_{x \in S}(f(x)) = \arg\max_{x \in S}(-f(x)) \tag{7}$$

The developed algorithm IDEA was compared with other existing algorithms. The number of objective function value calculation is equal for all algorithms. This allows to compare the performance of the algorithms not considering the execution time, programing languages or hardware configuration. Although the calculation of value for benchmark functions is not time-consuming, it can be in the case of real-life problems.

Copyright © 2015 MECS

A. Simulation settings

All compared algorithms use the same parameter values or equivalent values for specific parameters for some algorithm. Basic parameters of the algorithms shall adopt commonly used value [2, 8]. The comparative study contains the comparison of the following algorithms: developed algorithm IDEA, RCGA, elite RCGA with the one elite individual, elite RCGA with the number of elite individuals equal to population size and IDEA without directional vector – scheme of this algorithm is the same as in IDEA but uses mutation operator NUM instead of DNUM. Elite RCGA with the number of elite individuals equal to population size is similar to ES ($\mu + \lambda$) with crossover operator [2]. RCGA and both elite RCGA use uniform crossover operator with exchange probability (p_e) equal to 0.5 and crossing probability (p_c) equal to 0.75. RCGA and both elite RCGA use NUM with mutation probability (p_m) equal to 0.03. The adopted value of p_m is equivalent to "mutation probability" of DNUM used by IDEA, where a single gene is mutated for any individual of population in each algorithm cycle. All these algorithms use a proportionate selection with dynamic linear scaling of fitness function described in section 2. The size of the population and the number of generations are the same for all algorithms. Their values are respectively 100 individuals and 2000 generations.

B. Benchmark functions

The benchmark functions are given in Table 1. Unimodal functions f_1 - f_4 are called: Sphere Model, Schwefel's Problem 2.22, Schwefel's Problem 2.21, Generalized Rosenbrock's Function. Function f_5 is the step function, which has one minimum and is discontinuous. Function f_6 is a noisy quartic function, where random[0, 1) is a uniformly distributed random variable in [0, 1). Multimodal functions are $f_7 - f_9$ and are called: Generalized Rastrigin's Function, Ackley's Function and Generalized Griewank Function.

Table 1. Benchmark functions [8, 16].

f_i	x_i	$f_{\min}$
$f_1(x) = \sum_{i=1}^{30} x_i^2$	$x_i \in [-100,100]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_2(x) = \sum_{i=1}^{30} \lvert x_i \rvert + \prod_{i=1}^{30} \lvert x_i \rvert$	$x_i \in [-10,10]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_3(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le 30\}$	$x_i \in [-100,100]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_4(x) = \sum_{i=1}^{29} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$x_i \in [-30,30]$, $i = 1,2,...,30$	$f_{\min}(1,1,...,1) = 0$
$f_5(x) = \sum_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2$	$x_i \in [-100,100]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_6(x) = \sum_{i=1}^{30} i x_i^4 + random[0,1)$	$x_i \in [-1.28,1.28]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_7(x) = \sum_{i=1}^{30} (x_i^2 - 10\cos(2\Pi x_i) + 10$	$x_i \in [-5.12,5.12]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_8(x) = -20\exp(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2})$ $-\exp(\frac{1}{30}\sum_{i=1}^{30}\cos(2\Pi x_i^2)) + 20 + e$	$x_i \in [-32,32$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$
$f_9(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30}\cos(\frac{x_i}{\sqrt{i}}) + 1$	$x_i \in [-600,600]$, $i = 1,2,...,30$	$f_{\min}(0,0,...,0) = 0$

C. Experimental Results and Analysis

Tables 2-4 present the results of optimization multi-dimensional functions. These tables show obtained value of the best, the worst and average as a result of multiple launches of compared algorithms.

Copyright © 2015 MECS

Based on simulation results (Tab. 2-4) it can be observed that the IDEA gives better results in comparison with other EAs. The best, worst and average result of optimization of benchmark functions is equal or better compared to other EAs for the wide range of selected benchmark functions. Comparison of the results obtained by IDEA and IDEA without vector of direction shows that the use of only IDEA scheme of post-selection does not provide obtain better results than elite RCGA with number of elite individuals equal to population size. The comparison of the worst results obtained by the algorithms shows that in most cases they are better for IDEA, which is confirmed by the average results of the optimization of benchmark functions.

Table 2. The best results achieved.

Function	$f_{\min}$	RCGA	Elite RCGA	Elite RCGA	IDEA without vector of direction	**IDEA**
f_1	0	137,6195	39,34134	0,04354	0,077377	**0,016064**
f_2	0	2,036238	1,281487	0,103552	0,091704	**0,047126**
f_3	0	1,797428	0,654677	0,090355	0,177037	**0,098044**
f_4	0	40659,75	4381,31	82,05972	41,332169	**12,37685**
f_5	0	144	38	0	0	**0**
f_6	0	0,284131	0,479216	0,932808	0,819021	**0,102025**
f_7	0	27,07454	4,841767	0,024082	0,052699	**0,006697**
f_8	0	3,500655	0,898639	0,055616	0,066552	**0,030753**
f_9	0	2,50413	1,424007	0,073187	0,417053	**0,028646**

Table 3. The worst results achieved.

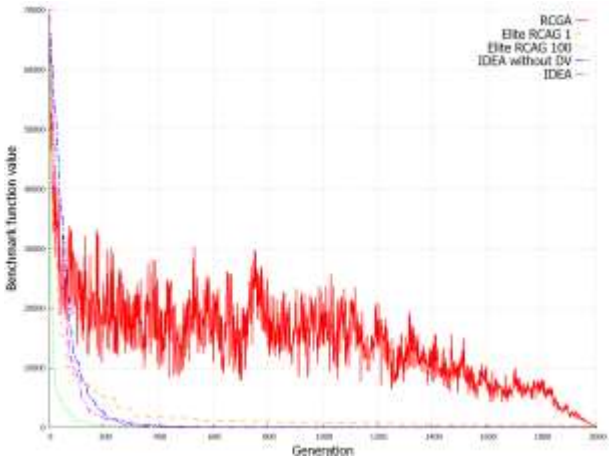
Function	$f_{\min}$	RCGA	Elite RCGA	Elite RCGA	IDEA without vector of direction	**IDEA**
f_1	0	254,5255	75,07664	0,129015	0,178441	**0,036108**
f_2	0	3,514146	1,605774	0,168645	0,190154	**0,084795**
f_3	0	3,346235	1,060122	0,224434	0,550799	**0,257424**
f_4	0	101062,9	10915,08	193,0529	193,6963	**105,1899**
f_5	0	296	99	0	0	**0**
f_6	0	0,918662	0,304626	0,385808	0,257063	**0,761903**
f_7	0	32,97223	9,029178	0,062866	0,112555	**0,014407**
f_8	0	4,438607	1,699093	0,135306	0,15737	**0,049317**
f_9	0	3,819432	1,724608	0,170535	1,018339	**0,178701**

Table 4. The average results achieved.

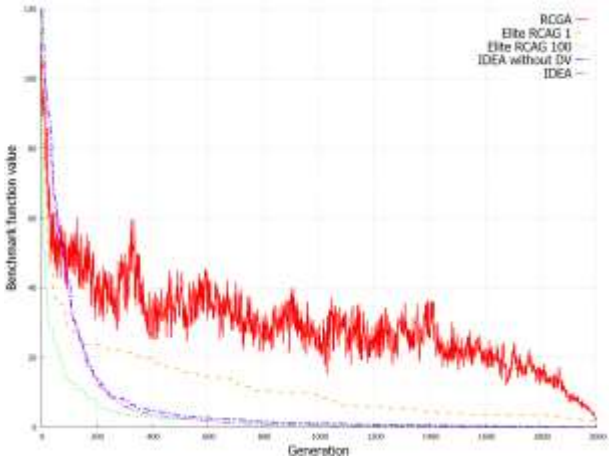
Function	$f_{\min}$	RCGA	Elite RCGA	Elite RCGA	IDEA without vector of direction	**IDEA**
f_1	0	205,4934	63,29672	0,072157	0,136031	**0,023187**
f_2	0	2,864105	1,469042	0,124475	0,143451	**0,059191**
f_3	0	2,658306	0,840783	0,134747	0,30542	**0,145462**
f_4	0	70978,62	7443,365	123,236	125,9332	**78,63267**
f_5	0	222,1	58	0	0	**0**
f_6	0	0,550015	0,545784	0,619994	0,559118	**0,442814**
f_7	0	29,37537	7,165922	0,035986	0,078663	**0,011218**
f_8	0	3,990445	1,343896	0,080027	0,106285	**0,038786**
f_9	0	2,934898	1,515274	0,12185	0,727247	**0,089973**

Copyright © 2015 MECS

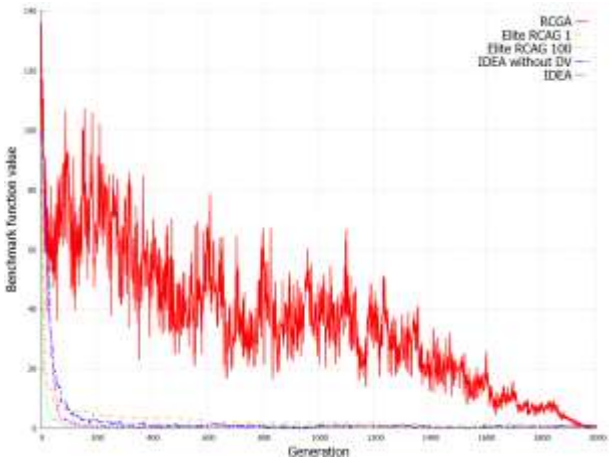
Figure 2. shows the value of benchmark functions for solution represented by the best individual in the subsequent generations. On the basis on graphs from Figure 2 it can be stated that the elite RCGA with number of elite individuals equal to population size achieve better result in initial cycles of algorithm. After few cycles, the IDEA achieves better solution than other EAs and finally returns the best solution in most of these cases. This is clearly visible in Figure 2d and 2e.



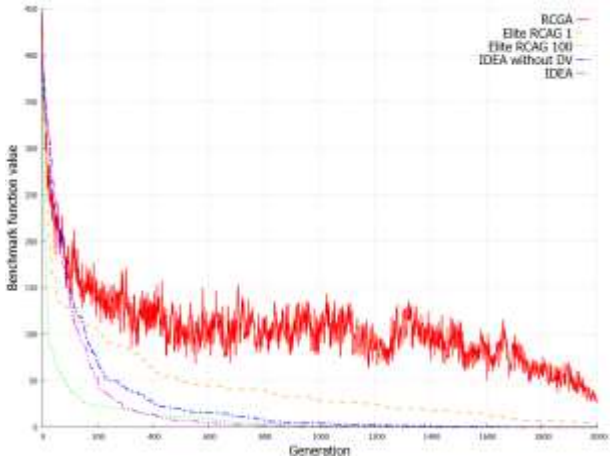
(a) f_1



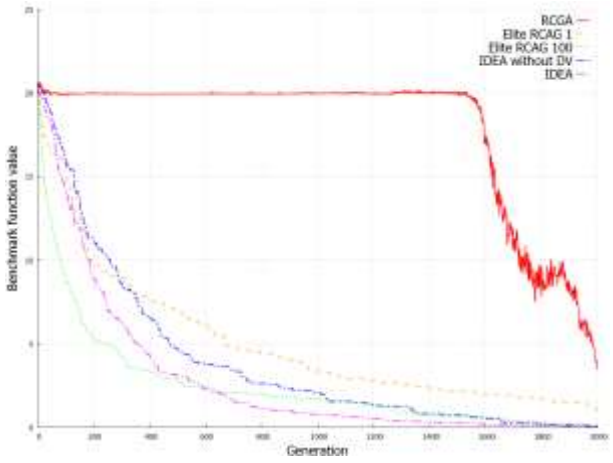
(b) f_2



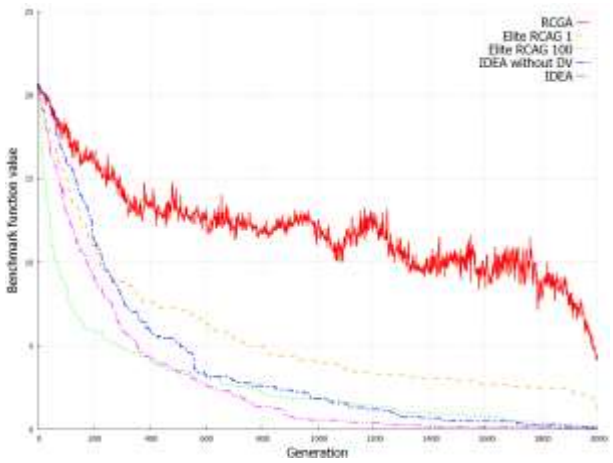
(c) f_4



(d) f_7



(e) f_8



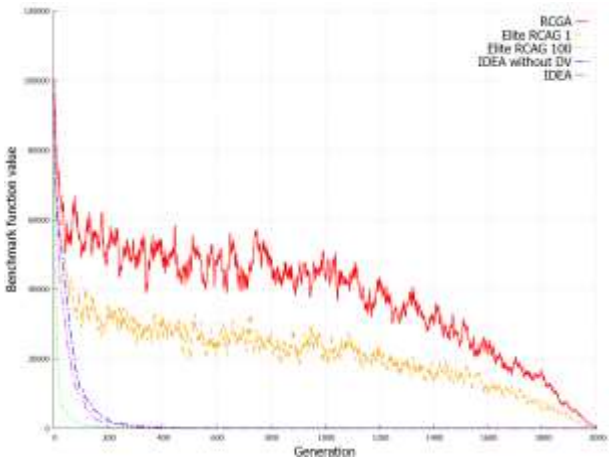
(f) f_9

Fig. 2. Changes of the objective function value in successive generations for best individual.

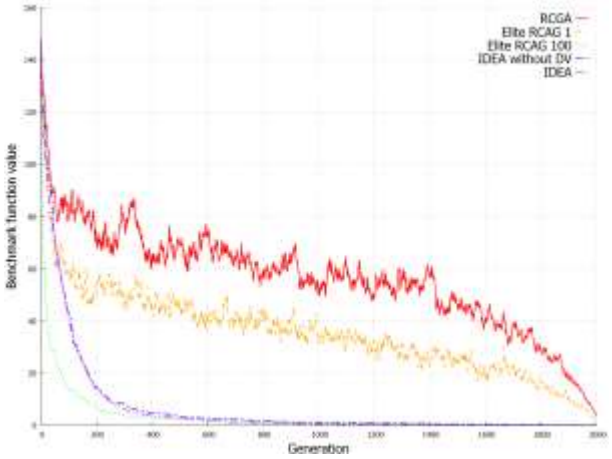
Figure 3. shows the average value of benchmark functions for solution represented by the population in the subsequent generations. Graphs from Figure 3 allow to observe a similar dependency as in case of evolution of best individuals presented on Graphs from Figure 2. IDEA allow to obtain a better best individual, but also obtain population with a higher average value of fitness

Copyright © 2015 MECS

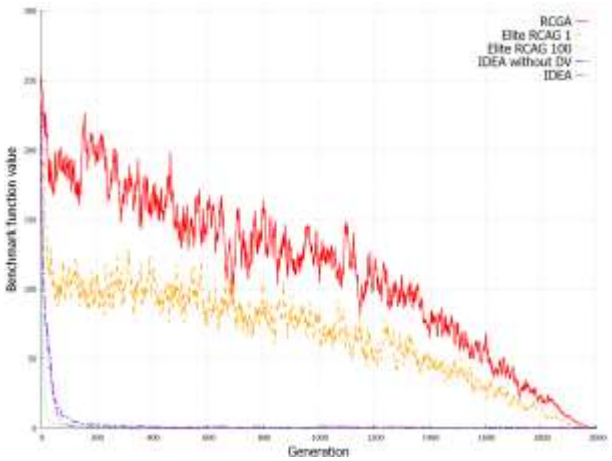
function in successive generations. This can be observed on all graphs from Figure 3 and is clearly visible in Figure 3d and 3e.



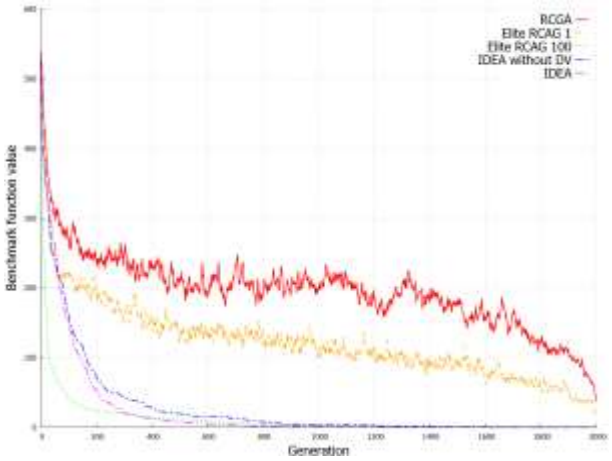
(a) f_1



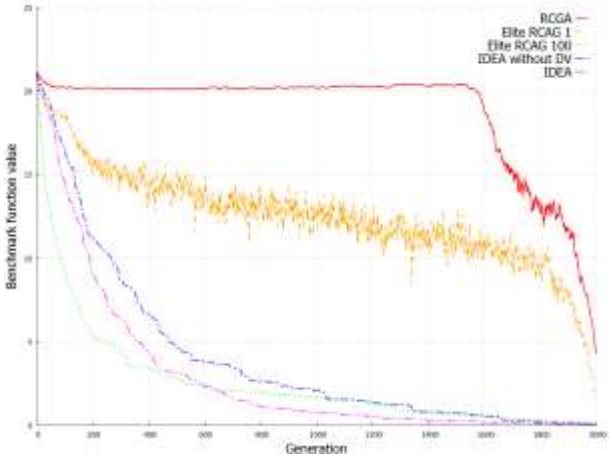
(b) f_2



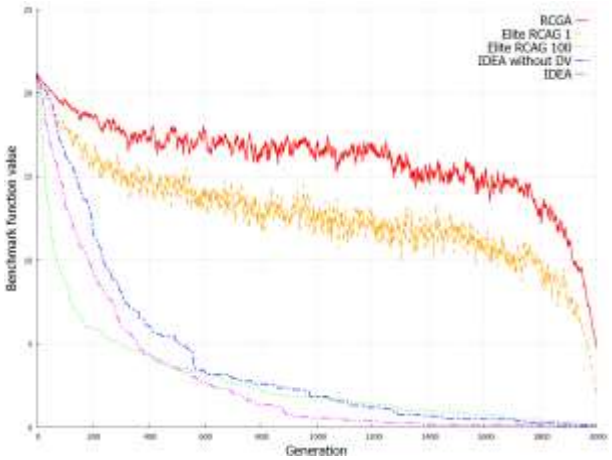
(c) f_4



(d) f_7



(e) f_8



(f) f_9

Fig. 3. Changes of the objective function average value in successive generations for population.

V. Conclusion

This paper presents simulation analysis of developed evolutionary algorithm based on independent evolution of selected individuals. The achieved results of this study show, that the proposed algorithm IDEA provides better results than other popular EAs used in the study. The

Copyright © 2015 MECS

DNUM operator allows to narrow the mutation scale in successive generations. Post-selection process allows to monitor and correct the direction of the independent evolution of individuals. The main idea of algorithm IDEA focuses on monitoring independent evolution of population individuals. Improving the average value of fitness function for population is only additional effect. The improvement of algorithm performance without increase number of calculation objective function value is useful for application of algorithm to solve real-life problems.

REFERENCES

[1] A. F. Ali, Genetic Local Search Algorithm with Self-Adaptive Population Resizing for Solving Global Optimization Problems, International Journal of Information Engineering and Electronic Business, 2014, 6(3): 51-63.
[2] J. Arabas, Lectures on evolutionary algorithms, WNT, Warsaw, 2001 (in Polish).
[3] A. Berry, P. Vamplew, PoD Can Mutate: A Simple Dynamic Directed Mutation Approach for Genetic Algorithms, AISAT 2004: The 2nd International Conference on Artificial Intelligence in Science and Technology, 2004 , 200-205.
[4] A. Berry, P. Vamplew, L. Temby, Accelerating real-valued genetic algorithms using mutation-with-momentum, The 18th Australian Joint Conference on Artificial Intelligence, 2005, 1108-1111.
[5] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, WNT, Warsaw, 1995 (in Polish).
[6] L. Grad, An example of feed forward neural network structure optimisation with genetic algorithm, Bulletin of The Institute of Automation and Robotics, Warsaw, 2006, (23): 27-36 (in Polish).
[7] I. Korejo, S. Yang, C. Li, A Directed Mutation Operator for Real Coded Genetic Algorithms, Applications of Evolutionary Computation, Springer, 2010, (6024): 491-500.
[8] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, WNT, Warsaw,1999, (in Polish).
[9] A. Obuchowicz, P. Prętki, Directional distribution-based mutation for evolutionary algorithms, Advanced Control and Diagnosis - ACD 2009 : 7th Workshop, 2009, [6] CD-ROM.
[10] S. K. Pal, C.S Rai, A. P. Singh, Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non-Linear Optimization Problems, International Journal of Intelligent Systems and Applications, 2012, 4(10): 50-57.
[11] K. Poczęta, Ł. Kubuś, Supervised and Population Based Learning Algorithms for Fuzzy Cognitive Maps – a Comparative Study, Applications of Information Technologies – theory and practice, Radom, 2014, 96-107.
[12] W. Stach, L. Kurgan, W. Pedrycz, M. Reformat, Evolutionary Development of Fuzzy Cognitive Maps, IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2005, Reno, Nevada, USA, May 22-25, 2005, 619-624.
[13] W. Stach, L. Kurgan, W. Pedrycz, M. Reformat, Genetic Learning of Fuzzy Cognitive Maps, Fuzzy Sets Syst., 2005, 153(3): 371-401.
[14] P. Tang, M. Tseng, Adaptive directed mutation for real-coded genetic algorithms, Applied Soft Computing, 2013, 13(1): 600-614 .
[15] Z. Świątnicki, V. Olej, Generation and Optimization of Fuzzy Neural Networks Structure, Bulletin of The Military University of Technology, Warsaw, 2002, 51(9): 125-138 (in Polish).
[16] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation, 1999, 3(2): 82-102.
[17] X. Zhao, X. Gao, A micro evolutionary programming for optimization of continuous space, PPSN VIII workshop on challenges in real world optimization using evolutionary computing, 2004, 17-23.
[18] X. Zhao, X. Gao, Evolutionary programming based on non-uniform mutation, Applied Mathematics and Computation, 2007, 192(1): 1-11.

Author's Profiles

Łukasz Kubuś was born in Kielce, Poland in 1990. He received the B.Sc. and M.Sc. degrees from Kielce University of Technology, in 2013 and 2014, respectively.

In 2014, he joined the Department of Computer Science Applications at Kielce University of Technology, as a PhD student. His field of research comprises application of evolutionary algorithms.

How to cite this paper: Łukasz Kubuś,"Individually Directional Evolutionary Algorithm for Solving Global Optimization Problems-Comparative Study", International Journal of Intelligent Systems and Applications (IJISA), vol.7, no.9, pp.12-19, 2015. DOI: 10.5815/ijisa.2015.09.02

Copyright © 2015 MECS