Influence of two different crossover operators use onto GPA efficiency

Tomas Brandejsky
dept. of Electrotechnics ans Informatics
University of Pardubice
Pardubice, Czech Republic
tomas.brandejsky@upce.cz

***Abstract*—Increasing capabilities of today computers, especially size of memory and computational power open new application areas to Genetic Programming Algorithms [1]. Unfortunately, efficiency of these algorithms is not big and decreases with solved problem complexity. Thus, its increase is extremely important for opening of new application domains. There exists three main areas that should potentially influence GPA efficiency. They are algorithms, pseudo-random number generator behaviours and evolutionary operators. Genetic programming algorithms use two basic evolutionary operators – mutation and crossover in the sense of Darwinian evolution. Non-looking to the fact, that it is possible to define additional operators like e.g. application defined operators [2], there are many different implementations of both basic evolutionary operators [3] and each of them is sometimes useful in artificial evolutionary process. Thus, the main question solved in this paper is that it might bring some advance to use two randomly executed different crossover operators in GPA. The study is focused to symbolic regression problem and as GPA is used GPA-ES, because it is capable to eliminate influence of solution parameters (constants) identification and thus to produce more clear results. (*Abstract*)**

Keywords—genetic programming algorithm, evolutionary operators, crossover, mutation, multiple crossover implementations (*key words*)

I. Introduction

Increasing capabilities of today computers, especially size of memory and computational power open new application areas to Genetic Programming Algorithms [1] (GPA). GPAs are special kind of Genetic Algorithms [4] or Evolutionary Strategies [5] which do not optimise set of model parameters but produces models itself (model is frequently described by fitness function, but it is possible to optimize model parameters without algebraically described fitness function using e.g. tournament selection). Unfortunately, efficiency of these algorithms is not big non-looking that it is better than efficiency of many other algorithms and decreases with solved problem complexity. Thus, its increase is extremely important for opening of new application domains. There exists three main areas that should potentially influence GPA efficiency. They are own algorithms, pseudo-random number generator behaviours and evolutionary operators. Genetic programming algorithms use two basic evolutionary operators – mutation and crossover in the sense of Darwinian evolution [6]. Non-looking to the fact, that it is possible to define additional operators like e.g. application defined operators [2], there are many different implementations of both basic evolutionary operators [3] and each of them is sometimes useful in artificial evolutionary process, as it will be discussed later. Thus, the main question solved in this paper is that it might bring some advance to use two randomly executed different crossover operators in GPA. While mutation randomly alters a randomly chosen part of a selected parent model, crossover combines randomly selected parts of predecessors to create offspring. The study is focused to symbolic regression problem and as GPA is used GPA-ES, because it is capable to eliminate influence of solution parameters (constants) identification and thus to produce more clear results.

II. Crossover

This study is focused to crossover operator. Many researches suggest do not use mutation frequently or to limit its application to initial evolutionary cycles or to situations where no progress is observed for many cycles. This is cased by specific feature of mutation which is not capable significantly change structure of the solution but it only optimizes its properties – used operators and referred terminals (constants and variables), see e.g. [1 to 3, 7 to 9]. Studied two versions of crossover operator are one_point crossover [10] (a kind of homologous crossover operator) and standard crossover [1].

There are known also many different modifications of these operators and other crossover operators like uniform crossover [11], context-preserving crossover [12] and size-fair crossover [13]. Standard crossover operator selects two parents from the members of population in the given evolutionary cycle and for the each of them it selects crossover point. Such implementation of crossover allows to use previously developed structure in novel content, but frequently it generates poor individuals, because it swaps sub-trees with different function, purpose. Application of this kind of crossover frequently tens to occurrence of different depth trees, it tends to non-homogeneous populations. Thus we can conclude that standard crossover operator is useful when population is too uniform, loses ability of development, but in other situations in is not efficient much.

Another problem of standard crossover is that it supports bloating – occurrence of extremely complex structures. It is caused possibility to set up crossover points in different depths of parent structures. After crossover operation complexity of one structure might significantly increase and vice versa.

On the opposite side, one_point crossover (and many other homologous crossover operators) searches corresponding sub-trees in both genes. This is done by passing the same trajectory in both genes, if it is possible. In the finish there is big probability, that we get crossover points of the same semantic sense (especially in the latter evolutionary cycles, when the population is more homogeneous). On the opposite side, the application of this kind of crossover operator makes population homogeneous and decreases its adaptability.

In the work [14], two different implementations of the cross-over operator applicable in GPA or GPA-ES algorithms were studied. The paper especially focused to evaluation of practical experiments with symbolic regression of model on the base of pre-computed data generated by Lorenz attractor system model. These experiments did not conclude any significant difference between these two implementations with small extreme near to ratio 1:1 which points that in the case of large populations it is sometime useful to combine both implementations. It herein presented new research the work was focused to GPA algorithm only and to different implementations of cross-over operator.

III. Experiment design

A. Evolutionary operator combination

In the typical situation when it is not possible either to decide between alternative operators nor to create novel one with optimal features is the essential to use all alternative operators together. This the reason why this paper is devoted to mixing of different implementations of crossover evolutionary operator in GPA. Because it is not possible on the today state of art to decide precisely when to apply one version of crossover operator and when other, in this study we will test different probabilities of their use. There are known only above introduced common ideas when to apply standard crossover and when one-point one, but is not easy to measure evolution efficiency, homogeneity of population etc.

B. Used GPA-ES

As it was introduced in the first part of the paper, this study is focused to symbolic regression problem. In the previous years the two level GPA-ES [15] evolutionary algorithm was developed. This algorithm described on Fig. 1 distinguishes between structure development and parameter optimization. First of them is solved by standard GPA algorithm, while parameter optimization is solved by Evolutionary Strategy (ES) algorithm. It gives GPA-ES algorithm to eliminate influence of parameter (constant) identification and measure properties of GPA component with respect to structural identification or development. In the presented paper it enables more precisely identify influence of crossover operator, because experiments results will include only number of GPA cycles needed to identify algebraic equations, see e.g. [16, or 17]. The work [18] brings huge set of references to relevant literature about structures of all typical cases of GPAs.

```
1) FOR ALL individuals DO Initialize() END FOR;
2) FOR ALL individuals DO
       Evaluate()=>fitness END FOR;
3) Sort(individuals);
4) IF Terminal_condition() THEN STOP
   END IF;
5) FOR ALL individuals DO
        SELECT Rand() OF
               CASE a DO Mutate()=> new_individuals;
               CASE b DO Symetric_crossover() =>
                       new_individuals;
               CASE c DO One_point_crossover() =>
                       new_individuals;
               CASE d DO Re-gerating() =>
                       new_individuals;
       END SELECT;
   END FOR;
6) FOR ALL individuals DO Evaluate =>
       new_fitness
   END FOR;
7) FOR ALL individuals DO
       IF new_fitness<fitness THEN
               individual = new_individual;
               fitness = new_fitness;
       END IF;
   END FOR;
8) GOTO 3);
```

Fig. 1. GPA-ES algorithm

Computational complexity of GPA-ES algorithm depends on many parameters with respect to its complexity and it was identified in [15] as follows:
GPA population initialization complexity is given by (1) if maximal building function arity is 2:

$$O(GPAINIT)=n2^l \tag{1}$$

Where

n	is number of GPA individuals
l	is complexity of structures created by GPA

GPA population evaluation complexity is analogically

$$O(GPAevaluation)=n\,O(ES) \tag{2}$$

Complexity of evaluation of ending condition is

$$O(GPAending)=n \tag{3}$$

Complexity of evaluation of individuals on the base of predefined fitness is

$$O(GPAeval)=n2^l \tag{4}$$

and complexity of population ordering is

$$O(GPAorderig) = nlog(n) \tag{5}$$

For the nested ES algorithm used to optimize parameters of each individual, the partial computational complexities are:

$$O(ESinit) = m2^l \tag{6}$$

Where
m is number of ES individuals
Complexity of gene evaluation is

$$O(ESeval) = mk = m2^l \tag{7}$$

where
k is average number of constants in GPA genes, $k = 2^l$.
Complexity of ES termination condition evaluation is

$$O(ESterm) = n \tag{8}$$

Intelligent crossover evolutionary operator complexity is (9):

$$O(EScrossover) = mk = m2^l \tag{9}$$

ES population ordering complexity is

$$O(ESsort) = mlog(m) \tag{10}$$

Resulting complexity of whole GPA-ES algorithm is (11-12):

$$O(ES) = \begin{matrix} O(ESinit) + O(ESeval) + O(ESterm) \\ + O(EScrossover) + O(ESsort) \end{matrix} \tag{11}$$

$$O(GPAES) = \begin{matrix} O(GPAinit) + O(GPAeval) \\ + O(GPAEnding) + O(GPAordering) \end{matrix} \tag{12}$$

C. Problem of GPA experiment evaluation

Because GPAs are stochastic algorithm strongly influenced pseudo-Random Number Generators (pRNGs), it is not possible to stand up conclusions on small number of observations.
SW implemented pRNGs in difference to true RNGs typically implemented in HW generates from single initial state one immutable data array. Different series are generated on the base of initial seed value change. Different situation occurs when pRNG is used in parallel algorithm with multiple threads or tasks. Parallel tasks bring new situation, because there is strong influence of task switching (which is influenced by other processes in computer). It makes generated random series hardly repeatable. Thus experiments with GPA shall be free of multitasking (the use of thread safe implementation of pRNGs is not sufficient) and it is need to use experiments with many different initial seed magnitudes.
In the presented experiments GPA-ES system runs as single-thread application to prevent above described influences. Experiment results are compared on the base of needed computational cycles and not computational times, because speed of processors changes with each novel model.
Lorenz attractor system (13 and 14) was used for comparability with other experiments provided within our research in the are of symbolic regression. To increase comparability of experiments, the large number of test functions is frequently applied by many researchers and benchmark sets were formed. Because it is need to use large number of seed magnitudes for each function and large populations in each experiment, there it is extreme computational power need and experiments are calculated on supercomputer.

$$\begin{aligned} x'(t) &= \sigma(y(t) - x(t)), \\ y'(t) &= x(t)(\rho - z(t)) - y(t), \\ z'(t) &= x(t)\, y(t) - \beta\, z(t) \end{aligned} \tag{13}$$

$$\begin{aligned} \sigma &= 16 \\ \beta &= 4 \\ \rho &= 45.91 \end{aligned} \tag{14}$$

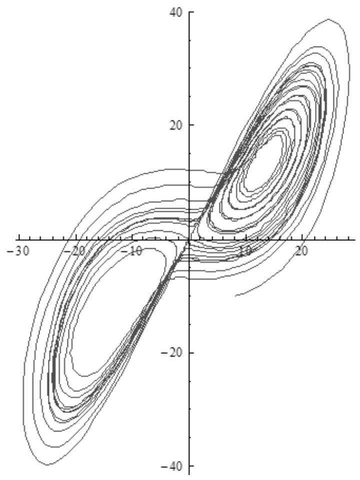

Fig. 2. Lorenz attractor in phase space

Because the Lorenz attractor system equations [19] are simple a they produce in the case of Extremely Accurate Symbolic Regression too simple equations to identify influences of selected crossover operator mix, the test cases ware extended by second, more complicated system – Roessler attractor system [20] (15 and 16).

$$\begin{aligned} x'(t) &= -y(t) - z(t) \\ y'(t) &= x(t) + ay(t) \\ z'(t) &= b + z(t)\left(x(t) - c\right) \end{aligned} \tag{15}$$

$$a=0.1, b=0.1, c=5.7 \tag{16}$$

Data series produced by pRNGs shows very long perturbations, as it is given by long periods of their functions. Unpredictable task switching even these periods many times multiplies. Thus, it is need to use many experiments based on these series with different initial seed magnitudes or the results are confusing. Unfortunately, frequently used GPA benchmarks do not solve this problem, does not suggest requirements to pRNGs, there is not given any limit to quality of solution – residual sum of error squares and many functions forming benchmark set and some functions forming benchmark problem set are very similar as in cases of benchmarks on [17].

If structure of the algorithm is known, it is possible to estimate probability of occurrence of given structure in the population and transpose results from measured experiments to another test case, as it was presented in [17, 13 and 21, where the first monograph presents many ways based especially on the schema theory, the second paper brings approach close to Markov processes. As it was published in [21], if we have sufficient number of experiments and if we solve symbolical regression problems with small residual error, average results depends only on statistical properties of initial population generator and applied evolutionary operators and it is possible to estimate them by application of Markov chain.

IV. Result discussion

From the above described reasons, the experiments were provided with data representing three equations of Lorenz attractor system. There were 500 samples. Experiments stopped when sum of error squares was less than 10^{-7}. The experiments were repeated 10000 times for different initial seed magnitudes of used pRNG (standard rand() function of C++ stdlib function) and for different proportions of probability of standard and one-point crossover application.

Table 1 and Fig. 3 presents computed average numbers of iterations for different proportions of standard and one-point crossover and for variables x, y and z of Lorenz attractor system. There is minimal influence of this proportion for variables x and z, because their generating differential equations are simple and linear. Magnitudes of variable y are computed from non-linear and more complex equation and there might be observed, that mixing of both operators with proportion about 50% to 50% brings increase of GPA algorithm efficiency.

TABLE I. Table 1 Average No of iterations depending on proportion f standard and one-point crossover for Lorenz attractor

	x	y	z
Crossovers 100_0	24.9	241	35.8
Crossovers 75_25	22.4	216	32
Crossovers 50_50	22.3	202	31.3
Crossovers 25_75	23.6	216	34.5
Crossovers 0_100	22.5	225	30.6

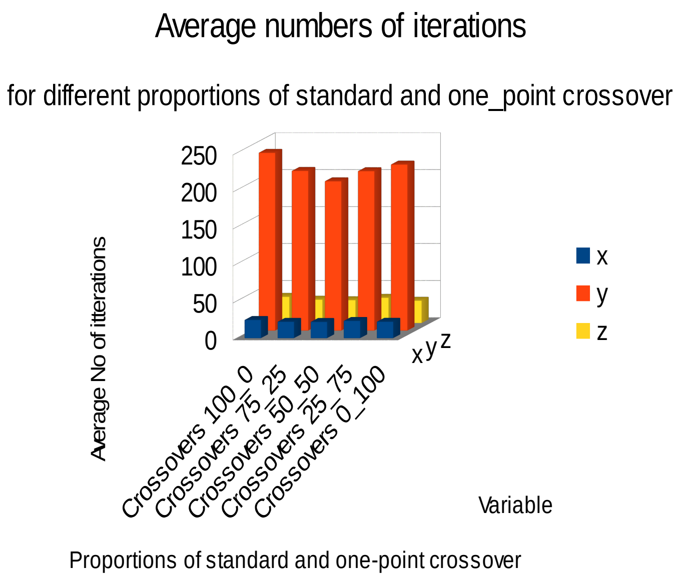

Fig. 3. Average No of iterations depending on proportion of crossover operators for variables x, y and z

Fig. 4 to 6 illustrates change of average iteration number of variables x, y and z respectively for different number of experiments. These graphs illustrates that it is need to use about 10000 experiments to achieve average error less than 1%.

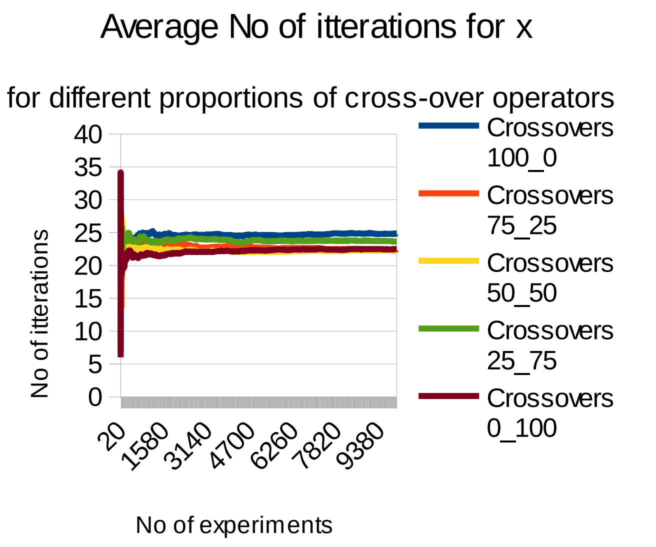

Fig. 4. Average iteration number of variable y for different number of experiments for variable x

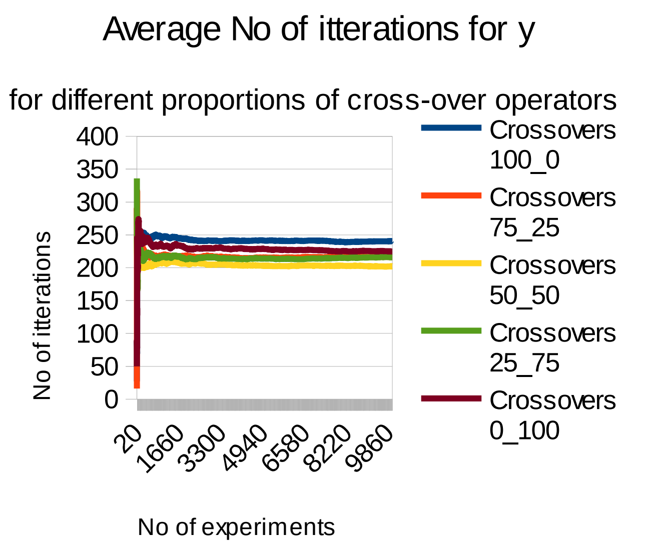

Fig. 5. Average iteration number of variable y for different number of experiments for variable y

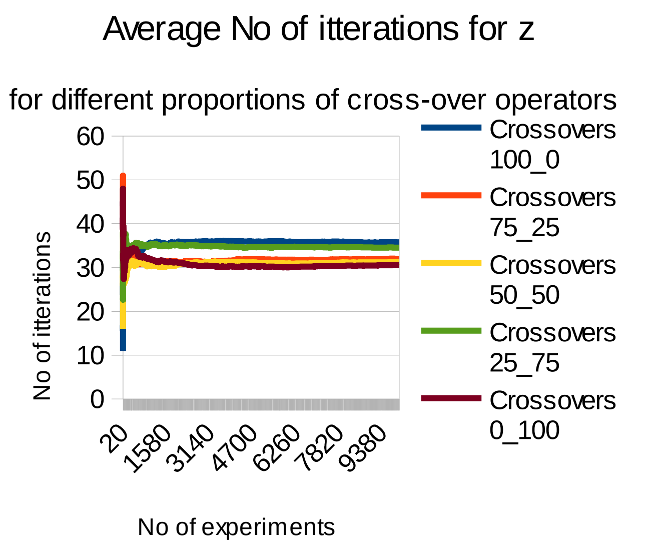

Fig. 6. Average iteration number of variable y for different number of experiments for variable z

Second group of experiments with Roessler attractor system tend to very similar results represented by Table 2 and Fig. 5 containing averages of measured data and graph of average iteration numbers of experiment.

TABLE II. AVERAGE NO OF ITERATIONS DEPENDING ON PROPORTIONS OF STANDARD AND ONE-POINT CROSSOVER FOR ROESLER ATTRACTOR

	x	y	z
Crossovers 10	22.64	226.1	31.9
Crossovers 75	22.44	215.32	31.71
Crossovers 50	22.09	210.58	30.42
Crossovers 25	22.33	212.81	30.28
Crossovers 0_	22.47	225.45	30.53

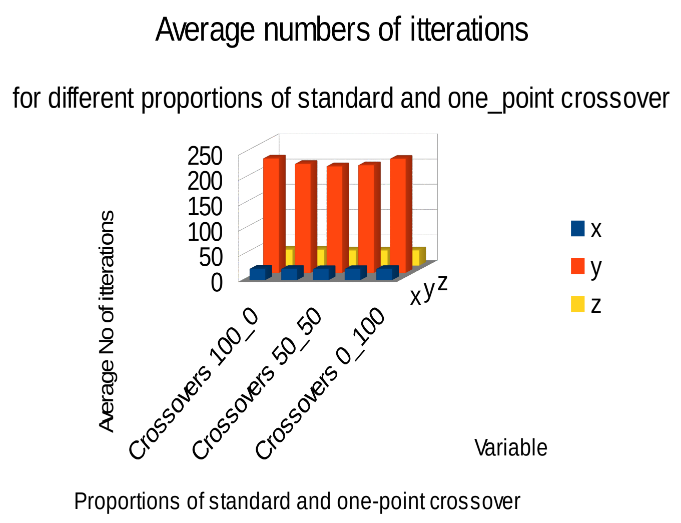

Fig. 7. Average No of iterations depending on proportion of crossover operators for variables x, y and z

V. CONCLUSION

Presented study points that it is impossible to study GPA behaviours on the base of small number of experiments. The paper points that it is possible to improve GPA behaviours by application of two or more different interpretation of crossover operator but additional time expensive test cases must be provided.

The paper brings analysis of used GPA-ES algorithm complexity and presents comparison of simultaneous use of two different crossover operators with different proportion of their application. The experiments were provided on data on two different system differential equations representing Lorenz and Roessler attractor systems. There were used 500 data samples for each system and each equation. Each symbolic regression experiment was 10000 times repeated for different pRNG seed magnitudes to eliminate pRGN influence onto experiment results.

The experiments conclude working hypothesis that it is better to combine (randomly invoke) more different crossover operators than to apply any of them.

ACKNOWLEDGMENT

Computational resources were supplied by the Ministry of Education, Youth and Sports of the Czech Republic under the Projects CESNET (Project No. LM2015042) and CERIT-Scientific Cloud (Project No. LM2015085) provided within the program Projects of Large Research, Development and Innovations Infrastructures.

References

[1] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection. MA: MIT Press, 1992.

[2] J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane, Genetic Programming III: Darwinian Invention and Problem Solving. CA: Morgan Kaufmann Publishers, 1999.

[3] "R. Poli, W. B. Langdon and N. F. McPhee, A field guide to genetic programming. Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008.

[4] D. Goldberg, (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Reading. MA: Addison-Wesley Professional. 1989.

[5] W. G. S. Hines, "Evolutionary stable strategies: a review of basic theory". Theoretical Population Biology. Vol. 31 (2), pp. 195–272, 1987.

[6] P. J. Bowler. Evolution: The History of an Idea (3rd completely rev. and expanded ed.). Berkeley, CA: University of California Press, 2003.

[7] A. S. Bickel and R. W. Bickel, “Tree structured rules in genetic algorithms”, in J. J. Grefenstette, editor, Genetic Algorithms and their Applications: Proceedings of the second International Conference on Genetic Algorithms, MIT, Cambridge, MA, USA, 28-31 July 1987. Lawrence Erlbaum Associates, pp. 77–81, .

[8] N. L. Cramer, “A representation for the adaptive generation of simple sequential programs”, in J. J. Grefenstette, editor, Proceedings of an International Conference on Genetic Algorithms and the Applications, Carnegie-Mellon University, Pittsburgh, PA, USA, 24-26 July 1985. URL http://www.sover.net/~nichael/nlc-publications/icga85/index.html, pp 183–187.

[9] C. Fujiki and J. Dickinson, “Using the genetic algorithm to generate lisp source code to solve the prisoner’s dilemma”, in J. J. Grefenstette, editor, Genetic Algorithms and their Applications: Proceedings of the second international conference on Genetic Algorithms, MIT, Cambridge, MA, USA, 28-31 July 1987. Lawrence Erlbaum Associates, pp. 236–240.

[10] R. Poli and W. B. Langdon, “A new schema theory for genetic programming with one-point crossover and point mutation”, in J. R. Koza, et al., editors, Genetic Programming 1997: Proceedings of the Second Annual Conference, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann, pp. 278–285.

[11] R. Poli and W. B. Langdon, “On the search properties of different crossover operators in genetic programming”, in J. R. Koza, et al., editors, Genetic Programming 1998: Proceedings of the Third Annual Conference, University of Wisconsin, Madison, Wisconsin, USA, 22-25 July 1998. Morgan Kaufmann, pp. 293–301.

[12] P. D’haeseleer, “Context preserving crossover in genetic programming”, in Proceedings of the 1994 IEEE World Congress on Computational Intelligence, volume 1, Orlando, Florida, USA, 27-29 June 1994. IEEE Press, pp. 256–261.

[13] GPA benchmarks, http://www.gpbenchmarks.org/wiki/index.php?title=Problem_Classification, accessed 21st July 2017.

[14] T. Brandejsky, “Two different implementations of the cross-ower operator in GPA algorithm”, in: R. Matousek., ed. 20th International Conference on Soft Computing Mendel 2014. Brno, 25.06.2014 - 27.06.2014. Brno: VUT v Brně, Fakulta strojní. 2014, s. 55-58. ISSN 1803-3814. ISBN 978-80-214-4984-8.

[15] T. Brandejsky, “Multi-layered evolutionary system suitable to symbolic model regression”, in: *Recent Researches in Applied Informatics.* 2nd International Conference on Applied Informatics and ompuiting Theory. Praha, 26.09.2011 - 28.09.2011. Athens: WSEAS Press, 2011, pp. 222-225.

[16] T. Brandejsky, “Genetic Programming Algorithm with Parameters Preoptimization - Problem of Structure Quality Measuring”, in: P. Osmera, ed. *MENDEL 2005.* 11th International Conference on Soft Computing. Brno, 15.06.2005 - 17.06.2005. Brno: Brno University of Technology05, pp. 138-144.

[17] T. Brandejsky, and I. Zelinka, “Specific Bahaviour of GPA-ES Evolutionary System Observed in Deterministic Chaos Regression”, in: I. Zelinka, et al., ed. *Nostradamus: Modern Methods of Prediction, Modeling and Analysis of Nonlinear Systems.* Nostradamus. Ostrava, 05.09.2012 - 07.09.2012. Heidelberg: Springer. 2013, pp. 73-81.

[18] W. B. Langdon and R. Poli, Foundations of Genetic Programming. Springer, New York, Heidelberg, Berlin (1998).

[19] E. N. Lorenz "Deterministic nonperiodic flow". J. Atmos. Sci. 20 (2), 1963, pp. 130–141.

[20] O. E. Rössler "An Equation for Continuous Chaos". Physics Letters 57A (5), 1976, pp. 397–398.

[21] W. B. Langdon, “Size fair and homologous tree genetic programming crossovers”, in W. Banzhaf, et al., editors, Proceedings of the Genetic and Evolutionary Computation Conference, volume 2, Orlando, Florida, USA, 13-17 July 1999, Morgan Kaufmann, pp. 1092–1097.