EVOLUTIONARY SYSTEM TO MODEL STRUCTURE AND PARAMETERS REGRESSION

Tomas Brandejsky[*]

Abstract: This paper discusses features of multilayered evolutionary system suitable to identify various systems including their model symbolic regression. Improved sensitivity allows modeling of difficult systems as deterministic chaos ones. The presented paper starts with a brief introduction to previous works and ideas which allowed to build the presented two abstraction levels system. Then the structure of Genetic Programming Algorithm – Evolutionary Strategy hybrid system is described and analyzed, including such problems as suitability to parallel implementation, optimal set of building blocks, or initial population generating rules. GPA-ES system combines GPA to model development with ES used for model parameter estimation and optimization. Such a hybrid system eliminates many weaknesses of standard GPA. The paper concludes with examples of GPA-ES application to Lorenz and Rösler systems regression and suggests application to Neural Network Model design.

Key words: *Genetic algorithm, genetic programming algorithm, evolutionary strategy, symbolic regression, model development*

Received: December 6, 2011
Revised and accepted: April 30, 2012

1. Introduction

This paper presents a multilayered evolutionary system combining Genetic Programming Algorithm (GPA) and Evolutionary Strategy (ES) designed to symbolic regression of model with reduced risk of so-called blowing (production of overcomplicated solutions), see [1]. The presented work continues in effort to increase structural sensitivity of GPAs; it means it attempts to produce solutions which are not overcomplicated, which are close to the analytical model describing original data, etc.

Standard GPA (see e.g. [2]–[4]) finds both the structure and its parameters (e.g. equation and magnitudes of its constants). A seriousig problem is caused by the

[*]Tomas Brandejsky
Department of Informatics and Telecommunications, Facuty of Transportation Sciences, Czech Technical University in Prague, Konviktska 20, 110 00 Prague 1, Czech Republic

fact that a wrong structure with perfectly fitted parameters might give a better fitness function value than a good structure with poorly estimated parameters. As it was published in [5] and [6], it is possible to find many situations when an imprecise structure with better parameters is evaluated as more suitable than a structure with better evolution potential, or even a perfect structure with wrong parameters. For example, the following list of pairs $(x, f(x))$ (1)

$$((0,0),(0.5,0.2125),(1,0.85),(1.5,1.9125)) \tag{1}$$

is described by (2) and symbolic regression by GPA gives the best solutions (3) and (4).

$$y = 0.85x^2 \tag{2}$$

$$y = 1.25x^2 \tag{3}$$

$$y = 1.25x \tag{4}$$

Even though (3) gives worse fitness function value 0.98, it has a proper structure. Equation (4) gives a smaller sum of error squares equal to 0.34, but since it is a linear function, it is harder to transform it into a proper form, see Fig. 1.

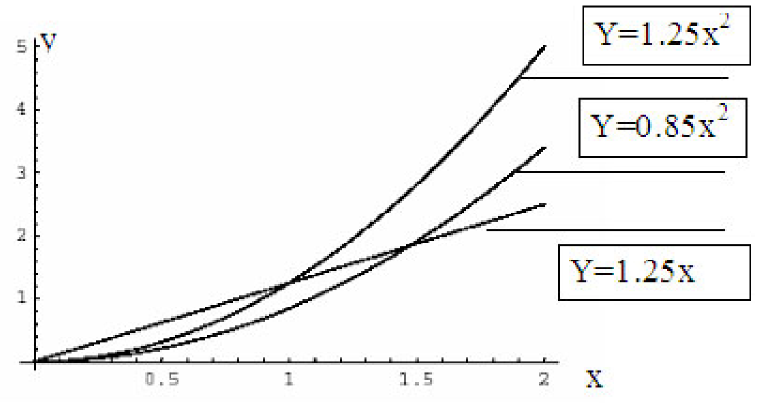

Fig. 1 *Masking of structure by wrongly estimated parameters.*

Since we need to measure structural potential of each solution precisely, the influence of random perturbation of parameters must be eliminated. GPA uses a single fitness function but it evolves both structures and their parameters. Such a situation does not allow separating of perfect structures reliably. In this paper, GPA-ES hybrid system uses GPA to develop structures, but ES (or Genetic Algorithm) system is applied in each cycle for each individual to optimize parameters, especially constants, in algebraic relations. GPA develops only structural operators; it does not care about constant terminals. The application of specific evolutionary system for parameters decreases error of their determination, but it increases computational complexity of the algorithm, as will be discussed later.

In the previous works, many attempts to increase ability of GPA to solve difficult problems were made. Article [7] studied possibility to evolve a qualitative model by GPA and then to improve it by GA into standard differential equations. This way is difficult due to the inherited ambiguity in the qualitative model and its transformation. There are also problems with measurement of distance of qualitative behaviours, because the space of qualitative behaviours is ordered, but without scale. Also improving a qualitative model is a special case of hierarchical evolutionary system building hierarchy of models on different abstraction levels by related hierarchy of evolutionary algorithms, as it was published in [8], [9].

2. Hybrid GPA-ES System

The structure of hybrid GPA-ES system is outlined in Fig. 2. Fig. 3 then illustrates the relationship between individuals in GPA population and vector of ES populations (one population for each GPA individual) in the GPA-ES system. While Fig. 2 outlines control flow of hybrid algorithm, Fig. 3 shows the fundamental data structure of this system.

The left part of Fig. 2 describes a master GPA algorithm, which is implemented in standard Koza's style. It means that genes are represented as tree-like structures and crossover and mutation evolutionary operators are used. The selection of proper genes for crossover operations is replaced by sorting the whole population on the base of fitness magnitudes. When the execution of this algorithm enters into state of individual evaluation, the slave Evolutionary Strategy described in the right part of the figure is called for each GPA individual. At the end of ES execution, fitness of the best individual is reasoned as fitness of the GPA individual.

The first line of boxes in Fig. 3 represents individuals of GPA algorithm. The particular box in the second line corresponds to each individual. These boxes represent population of ES used for parameters optimization. Individuals of this ES population are represented as rows in the related boxes.

The GPA-ES structure is applicable in many modifications. Especially, Evolutionary Strategy might be replaced by other suitable algorithms, like Genetic Algorithms, Self Organizing Migrating Algorithm (SOMA) [10] algorithms, Simulated Annealing [11] or many others suitable to efficiently optimize parameters of structures discovered by the master GPA. These systems are a special case of multilayered evolutionary system working with multiple abreaction levels, see [8] and [9]. GPA-ES uses structural and parameter levels, but it is possible to extend it e.g. by component one. The above mentioned work [5], in contrary, uses two abstraction levels in the form of qualitative and differential model.

3. Parallel Implementation of GPA-ES

From the practical point of view, it is useful to parallelize rather the outer (GPA) loop than the inner ES loop of the presented GPA-ES algorithm. Parallelization of inner loop is efficient only in case of regressed data set, and the regression functions found are extremely complex; thus the evaluation of ES takes much computational time. This is the reason why only parallelization of the GPA loop will be discussed.

From many parallelization schemas, OpenMP library has been chosen, especially due to its efficiency on single processor machines with multicore processors

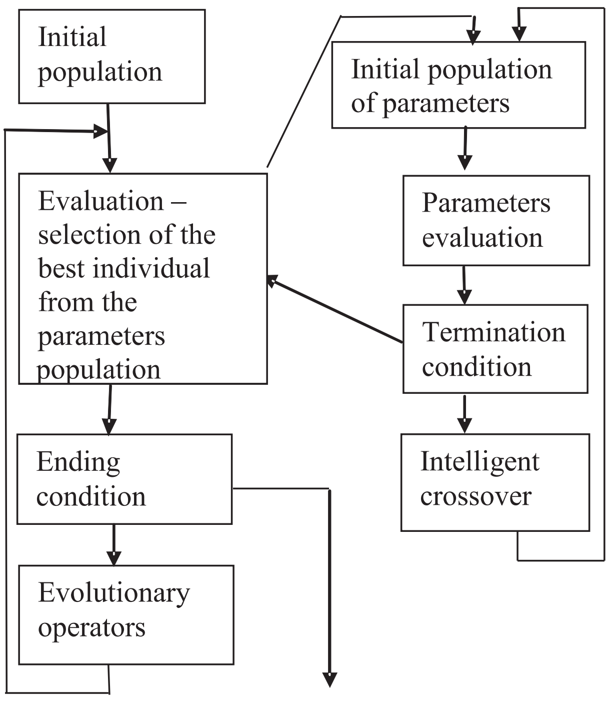

Fig. 2 *Structure of hybrid GPA-ES system.*

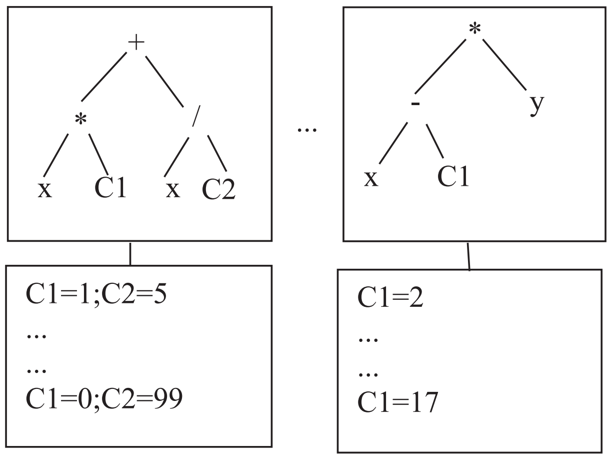

Fig. 3 *Relationship between GPA population individuals and vector of ES populations.*

or common Symmetric MultiProcessor machines (SMP). Then, the implementation is reduced to parallelization of evaluation loop in GPA, which reduces to a single directive and includes the OpenMP library header file, because OpenMP implicitly supports C and Fortran languages, but only C/C++ is applicable in GPA implementation. It is also possible to use e.g. Microsoft PPL library which is much simpler than OpenMP because only cycle parallelization is supported. Even special languages like Chapel or X10 developed within High Performance Computing Initiative can be used; however, the efficiency of these languages on small machines is not great and many features of these languages remain unused, see [12]–[14]. Chapel [12], only a few years old, and until now intensively developed parallel programming language being developed by Cray Inc. as a part of the DARPA-led High Productivity Computing Systems program (HPCS). Within HPCS program, two programming languages are being developed. The second one is IBM X10 [13]. In computer cluster built at the Department of Informatics and Telecommunication, Faculty of Transportation Sciences, CTU in Prague, Chapel compiler brings less compatibility problems than X10 environment, therefore it is used for multilayered evolutionary system development. Both languages support a multithreaded execution model via high-level abstractions for data parallelism, task parallelism, concurrency, and nested parallelism. Chapel is based on GNU compiler set, X10 is JAVA oriented. Both languages are designed to hide fragmented memory space (Non Uniform Memory Access) from the user and to increase programming efficiency.

The following four principles guided the design of Chapel language:

1. General parallel programming
2. Locality-aware programming
3. Object-oriented programming
4. Generic programming

The first two principles were motivated by desire to support general, performance-oriented parallel programming through high-level abstractions. The second two principles were motivated by desire to shorten the distance between high-performance parallel programming languages and mainstream programming.

When the evaluation loop of GPA is parallelized, each ES population of parameters is optimized in parallel with others. Thus, for each individual of the GPA population, a special copy of the ES is created and executed. Because no communication between them is required, the execution runs extremely efficiently without necessity of any synchronization.

Computational complexity of GPA-ES is expressed as (5),

$$O\left(GPAES\right) \cong pqnm\log m + pqn\log n, \tag{5}$$

where
n is the number of GPA individuals
m is the number of ES individuals
l is the complexity of structures created by GPA
k is the average number of constants in GPA genes, where
p is the number of GPA populations
q is the number of ES populations

This complexity estimation of GPA-ES system described by Fig. 2 presumes that computational complexities of building blocks are following:

GPA population initialization complexity is given by (6)

$$O\left(GPAinit\right) = n2^{l} \tag{6}$$

GPA population evaluation complexity

$$O\left(GPAevaluation\right) = n2^{l} \tag{7}$$

Ending complexity

$$O\left(GPAend\right) = 1 \tag{8}$$

Evolutionary operators with population ordering

$$O\left(GPAeval\right) = n\log n + n * 2^{l} + kn + n \tag{9}$$

ES population initialization

$$O\left(ESinit\right) = m * 2^{l} \tag{10}$$

ES population evaluation

$$O\left(ESeval\right) = mk \tag{11}$$

ES termination condition

$$O\left(ESterm\right) = 1 \tag{12}$$

Population ordering and intelligent crossover

$$O\left(ESsort\right) = m\log m + mk \tag{13}$$

In case of hardware threads and parallelization of evaluation cycle of GPA only, relationship 5 changes to (14):

$$O\left(GPAES\right) \cong pq\frac{n}{t}m\log m + pqn\log n. \tag{14}$$

Parallel implementation in the Chapel language requires application of Object Oriented Programming. Inner ES algorithm is implemented as class **CES** with methods described in Tab. I. **CES** object also contains constants and variables used for representation of genes, their fitness measures and data list used for fitness function evaluation.

The GPA-ES evolutionary system uses parallelization of outer GPA loop calling multiple **CES** objects in parallel. **CES** objects are implemented as scalar to increase efficiency of implementation.

The main GPA class named **CGPAES** contains analogical methods outlined in Tab. II.

Method	Description
CES (constructor)	Initializes ES internal data, especially randomly generates the first population.
Evalinit	Evaluates the initial population stored in the Genes array. Due to the application of consolidation, populations created in the following step are stored in Newgenes; a parallel data structure and evaluated by Evalnew. After Evalnew, Consol is applied; it they are better than parents, they replace them.
Sort	Sorts genes stored in the Genes array on the base of magnitudes in the fitness array.
Ending	This function implements condition of computation termination. The CES class implements a simple condition based on the maximal number of iterations and predefined acceptable error. These two conditions are in “OR” relationship.
Newpop	Computes new genes (new population) by intelligent crossover.
Evalnew	Evaluates fitness measure of each member of the new population.
Consol	Compares new and parents individuals, and if new ones are better, it replaces the older ones (parents) by them.

Tab. I *Methods of CES class implementing class of evolutionary strategy used for parameters (constants) optimization.*

4. Optimal Population Sizes

Efficiency of the algorithm depends on sizes of populations. These dimensions influence computational complexity (5) or (14), as it is discussed in the previous chapter, but it also influences an average error of slave ES and consequently an error of the whole GPA-ES. Unfortunately, while the computational time of single GPA-ES cycle increases with sizes of populations, the average error decreases, and thus the number of GPA-ES system cycles decreases too. Because the average number of constants **k** in the developed structures plays the significant role in (5) and (14), the number of ES individuals in ES population depends on expected complexity of resulting structure created by master GPA. Some examples of these magnitudes will be discussed in the GPA-ES application chapter.

5. Suitable Set of Building Blocks

The GPA-ES system is influenced also by many other parameters, especially by the predefined set of building blocks like operators and functions in case of symbolic regression of data set. It is well known that the use of simply interchangeable functions as e.g. sin and cos strongly decreases efficiency of GPA algorithm and it is better to use one of them. On the contrary, a more serious problem occurs when a suitable function or operator is missing. Initial structures in population influence efficiency of evolutionary process too.

Terminals also play a significant role from the efficiency point of view. In case of linear systems, generating terminals in the form (15) is better than generating constants and variables randomly connected by any operator or function. However, even in case of non-linear systems, the structure of initial population influences efficiency, as it will be discussed in the following chapter.

$$const1 * var + const2 \tag{15}$$

Systems might be described as functions of time or previous states. Problems occur when building a block set of GPA allows both descriptions. In such a case, there is risk that the crossover operation will try to combine individual using one description with the opposite one. That situation tends to decrease GPA efficiency due to poor results of such an operation, which will be rejected frequently.

Method	Description
CGPAES (constructor)	Initializes GPA internal data, especially randomly generates the first population and initializes vector of CES objects.
Evalinit	Evaluates the initial population. It calls evaluation of ES objects individuals representing parameters for each individual of GPA population, and the fitness of the best of them serves as fitness measure of the structure represented by related GPA gene.
Sort	Sorts genes stored in the Genes array on the base of magnitudes in the fitness array. GPA individuals are represented in tree-like structures in Koza's style.
Ending	This function implements condition of computation termination. Also a possible number of GPA evolutionary cycles depends on the predefined maximal value of fitness function of the best individual and on the maximal number of cycles.
Newpop	Creates new structures by crossover and mutation operations. In the first evolutionary cycles, mutation is preferred in relation to Koza's suggestions. Then the probability of crossover operation increases.
Evalnew	Evaluates new population. To do so, it calls related CES object to optimize parameters and to select the best of them. Its fitness magnitude serves as magnitude of the structure represented by related GPA gene.
Consolidate	Compares new and parents individuals and if the new ones are better, it replaces the older ones by them.

Tab. II *CGPAES hierarchical evolutionary system main class methods.*

6. GPA-ES Application

Study of chaotic systems has relation to many areas of science and especially to system modeling, as stated in works [15] and [16]. Also, the applicability of evolutionary techniques to complex system modeling has been verified many times as illustrated e.g. by works [17] and [18].

Symbolic regression of Lorenz attractor from the previously computed data set is a good test bed for many techniques applicable to model chaotic system, as it is studied e.g. in [19]. Lorenz attractor is described by a set of three equations (16), where parameter magnitudes are chosen as (17) (there are known many combinations of parameters capable to produce chaotic behavior, and chosen parameters produces behavior outlined at Fig. 4).

$$\begin{aligned} x'[t] &= \sigma(y[t] - x[t]), \\ y'[t] &= x[t](\rho - z[t]) - y[t], \\ z'[t] &= x[t]y[t] - \beta z[t] \end{aligned} \tag{16}$$

$$\begin{aligned} \sigma &= 16 \\ \beta &= 4 \\ \rho &= 45.91 \end{aligned} \tag{17}$$

Hybrid systems as the used GPA-ES work with multiple populations of genes. Presented algorithm works with population of genes representing tree-like description of regressed algebraic relationships. This population was evolved by GPA and it contained 400 genes. In each step of evolution of each gene, the population of 2,000 genes representing parameters of its structure is created and optimized. Fitness function magnitude of the best parameter set then serves as magnitude of fitness function of evolved structure in GPA.

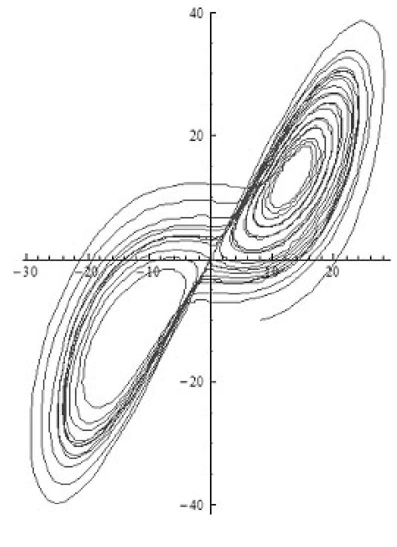

Fig. 4 *Lorenz attractor in phase-space.*

During the experiment, all three forms of terminals were tested – free form, const*var, and const1*var+const2. Because Lorenz attractor equations are non-linear, the free form was expected to be the best one. Results are summarized in the following Tab. III.

Unexpectedly, the best form of terminals from both aspects of convergence (number of iterations) and solution quality is the form const*var. It is given by the fact that the form const1*var+const2 is not present in equations of Lorenz attractor (16), but the const*var form is present many times. The free form of terminals is less efficient too. Quality of results mentioned in Tab. I is in range a) perfect (solution obtained by evolutionary system is identical to original equation), with b) decomposed constants (e.g. on the place of constant 5 the result contains form 2+3) or c) with decomposed constants and variables (e.g. on the place of constant 5 there is structure var+2+3-var). A totally different solution which is not transformable by any algebraic transformation is the last possibility d). No totally different and false structures have been obtained. In all experiments, the limit of preciseness has been reached. Preciseness was expressed as a sum of difference squares on 599 samples (6 seconds of prediction). The limit of this sum was 10^{-8}.

terminal form	number of iterations	quality of result
free form	x: 2 y: 4 z: 3	x: perfect y: decomposed constants z: perfect
const*var	x: 1 y: 4 z: 2	x: perfect y: decomposed constants z: perfect
const1*var+const2	x: 3 y: 35 z: 4	x: perfect y: decomposed constants and variables z: decomposed constants

Tab. III *Influence of terminal form on the number of iterations and quality of result for each Lorenz equation.*

As the second system for verification of GPA-ES capabilities, Rösler attractor has been used. Rösler attractor is described by three non-linear ordinary differential equations and thus its symbolic regression from data is comparably difficult as Lorenz attractor symbolic regression. The following figure reprinted from [20] presents the shape of this system.

The defining equations are:

$$\begin{aligned} x'[t] &= -y - z \\ y'[t] &= x + ay \\ z'[t] &= b + z\,(x - c) \end{aligned} \tag{18}$$

In the experiment, the system originally studied by Rösler with properties $a = 0.2$, $b = 0.2$, and $c = 5.7$ was reasoned. These properties give chaotic behavior to the system of equations (18).

Numerical solution of these equations was used to generate a table of data and GPA-ES system was used for backward reconstruction of equation (18) by methods of symbolic regression without any additional knowledge like parameter magnitudes or operators occurring in (18). Numbers of three-dimensional vectors

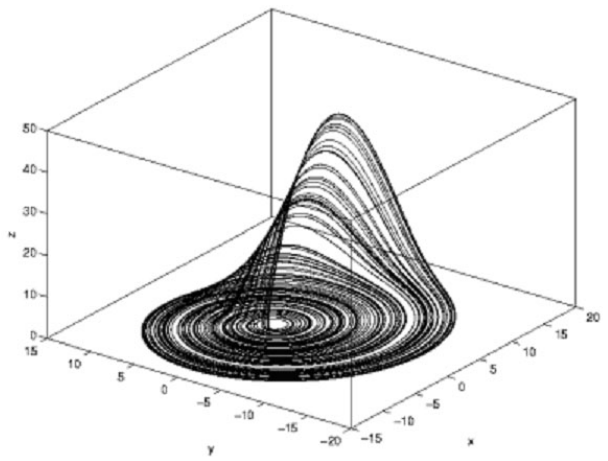

Fig. 5 *Rösler attractor described by equation (18).*

forming the data set were tested on magnitudes 995, 500, 250, 100 and 50. Results are summarized in the following Tab. IV. The biggest problems were caused by regression of variable Z, which, generating function, contains element $X * Z$. In that case it was possible to observe that series shorter than approximately 100 samples tends to identification of alternative solutions which cannot be transformed into original form (18) by any algebraic operation.

Average dependency between data size and quality of result is outlined in the following Fig. 6:

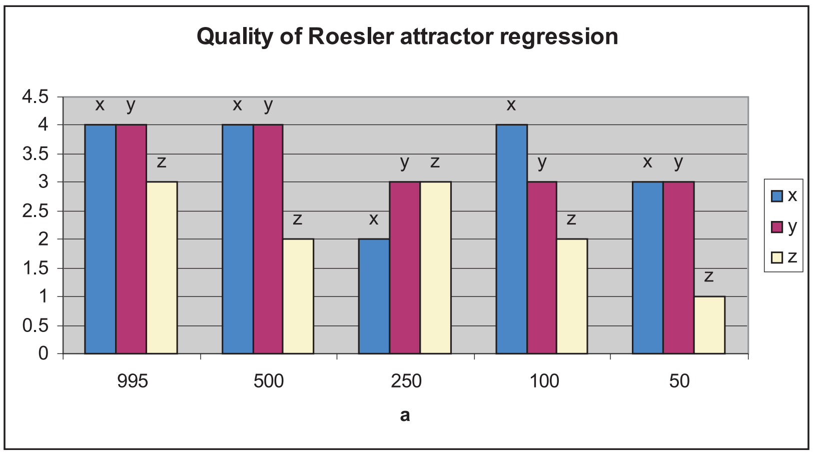

Fig. 6 *Average dependency between data set size and quality of result. Magnitudes have the following meaning – 1 is totally different algebraic form of solution, 2 represents decomposed constants and variables, 3 means decomposed constants and 4 is perfect form (equal to generating function).*

Variable	Data size	Quality of result	Approximation Error
X	995	perfect	1.67179e-18
	500	perfect	9.09656e-19
	250	decomposed constants and variables	3.72294e-19
	100	perfect	1.88232e-19
	50	decomposed constants	5.95561e-20
Y	995	perfect	1.77429e-18
	500	perfect	8.38322e-19
	250	decomposed constants	4.61374e-19
	100	decomposed constants	3.32647e-18
	50	decomposed constants	1.22095e-19
Z	995	decomposed constants	1.40079e-08
	500	decomposed constants and variables	8.73276e-10
	250	decomposed constants	1.30507e-11
	100	decomposed constants and variables	3.45838e-10
	50	Totally different solutions	2.09698e-07

Tab. IV *Results of Rösler attractor data symbolic regression for different sizes of regressed data set.*

Capabilities of the presented evolutionary system allows its application in many difficult problem domains as driver behavior modelling, EEG analysis with respect to micro-sleep detection and prediction or application to development and learning of extended Dynamic neural architecture HONNU [21] (of its activating function). All these application domains are now intensively tested. Because of high required computational capacity of the algorithm and constrained parallelization, it is possible to wait for quantum computer implementation, see e.g. [22], or to use supercomputer with symmetric multiprocessing and high number of processor cores. Unfortunately, such way is not easy, too, for small mixing in large populations and concluding decrease of GPA evolution efficiency.

7. Conclusion

The presented paper describes the Multi-Layered Evolutionary System suitable to symbolic model regression. The described structure of the evolutionary system decreases risk of wrongly identified parameters influence. Convergence of the system is fast and the obtained results have acceptable quality. This fact allows efficient applying of this evolutionary system in such tasks as symbolic regression of deterministic chaos system dataset. In the future, this algorithm is applicable in the area of EEG analysis and EEG signal model symbolic regression, where it is possible to follow works [23—25] applying EEG analysis in the area of driver micro-sleep detection and prediction which is significant for road transport safety improvement.

Acknowledgement

The work is supported by research project of MŠMT ČR No. 6840770043 "Improvement of methods of design and employment of transportation networks from optimization viewpoint".

References

[1] Langdon W. B., Poli R.: Foundations of Genetic Programming. Springer, New York, Heidelberg, Berlin, 1998.

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

[3] Koza J. R.: Genetic Programming II: Automatic Discovery of Reusable Programs. Boston, MA: The MIT Press, 1994.

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

[5] Brandejsky T.: Genetic Programming Algorithm with Parameters Pre-optimization – Problem of Structure Quality Measuring. In: Mendel 2005, Brno, 2005, pp. 138-144. ISBN 80-214-2961-5.

[6] Brandejsky T.: Genetic Programming Algorithm with constants pre-optimization of modified candidates of new population. In: Mendel 2004, Brno, 2004, pp. 34-38.

[7] Brandejsky T.: Qualitative behaviors similarity measure. Neural Network World, **12**, 2002, pp. 105-112.

[8] Brandejsky T.: Structure of Qualitatively-Numerical Evolutionary Algorithm applicable in the area of Conceptual Design, Artificial Intelligence in Design 2002 – 7th International Conference on Artificial Intelligence in Design – Poster Abstracts, University of Cambridge, Cambridge, UK, 2002.

[9] Brandejsky T.: The Application of Analogical Reasoning in Conceptual Design System. In: M. H. Hamza (ed.) Artificial Intelligence and Applications. Anaheim: Acta Press, 2003, **1**, pp. 511-516. ISBN 0-88986-390-3.

[10] Zelinka I., Lampinen J.: Evolutionary Identification of Predictive Models. In: C. Fye (Ed.) International Symposium on Engineering of Intelligent Systems, Paisley, Scotland, UK, ICSC Academic Press International Computer Science Conventions, Canada, Switzerland, ISBN 3-906454-21-5

[11] Kirkpatrick S., Gelatt C. D., Vecchi M. P.: Optimization by Simulated Annealing. Science, No. 220, 1983, pp. 671-680.

[12] Chapel Language Specification, (2010, April 15). [Online]. http://chapel.cray.com/spec/spec-0.795.pdf (URL).

[13] Saraswat V., Bloom B., Peshansky I., Tardieu O., Grove D.: X10 Language Specification. (2011, May 31). [Online]. http://dist.codehaus.org/x10/documentation/languagespec/x10-latest.pdf (URL).

[14] Brandejsky T.: Parallel implementation of evolutionary strategy in Chapel language. In: Mendel 2011, Brno, 2011, pp. 107-111.

[15] Kahng B.: Redefining Chaos: Devaney-chaos for Piecewise Continuous Dynamical Systems, International Journal of Mathematical Models and Methods in Applied Sciences, **3**, 4, 2009, pp. 317-326.

[16] Sahab A. R., Modabbernia M. R., Pastaki A. G.: Synchronization Chaos using OGBM with Genetic Algorithm, International Journal of Mathematical Models and Methods in Applied Sciences, **5**, 2, 2011, pp. 102-109.

[17] Lasheen A. A., El-Garhy A. M., Saad E. M., Eid S. M.: Using hybrid genetic and Nelder-Mead algorithm for decoupling of MIMO systems with application on two coupled distillation columns process, International Journal of Mathematics and Computers in Simulation, **3**, 3, 2009, pp. 146-157.

[18] D. J. and Sandhu K. S.: Excitation Control of Self Excited Induction Generator using Genetic Algorithm and Artificial Neural Network," International Journal of Mathematical Models and Methods in Applied Sciences, **3**, 1, 2009, pp. 68-75.

[19] Abarbanel H. D. I.: Analysis of observed chaotic data. New York, Springer Inc., 1996. ISBN 0-387-94523-7

[20] Cayuela J. S.: (2011, September) Chaos [Online], http://complex.upf.es/∼josep/Chaos.html

[21] Bukovsky I., Bíla J.: Development of Higher Order Nonlinear Neural Units for Evaluation of Complex Static and Dynamic Systems, Proceedings of Workshop 2004, Part A, Special Issue, Czech Technical University, Czech Republic, Prague, March **8**, 2004, pp. 372-373.

[22] Svítek M.: Conditional Combinations of quantum Systems, Neural Network World. 2011, **21**, 1, 2006, pp. 67-73. ISSN 1210-0552.

[23] Faber J., Novák. M., Tichý T., Svoboda P., Tatarinov V.: Driver psychic state analysis based on EEG signals, Neural Network World. **16**, 1, 2006, pp. 25-39. ISSN 1210-0552.

[24] Faber J., Novák M.: Thalamo-cortical reverberation in the brain produces alpha and delta rhythms as iterative convergence of fuzzy cognition in an uncertain environment, Neural Network World, **21**, 2, 2011, pp. 169-192. ISSN 1210-0552.

[25] Jirina M., Bouchner P., Novotný S.: Identification of driver's drowsiness using driving information and EEG, Neural Network World. **20**, 6, 2010, pp. 773-791. ISSN 1210-0552.