1

Metaevolution with Analytic Programming

by

Zuzana **Oplatkova**[1], Roman **Senkerik**[2]

The chapter will explain the differences in the approach of the metaevolution. Metaevolution as well as evolution and symbolic regression are methods of artificial intelligence. The paper will discuss evolutionary techniques for optimization, symbolic regression methods for the task of approximation, fitting and similar tasks. The core will be the discussion what metaevolution is and how it is used in different approaches. Last part of the paper will present applications and their results from the mentioned fields.

Keywords

Metaevolution, evolutionary techniques, symbolic regression, SOMA, differential evolution, deterministic chaos.

Acknowledgments

This work was supported by European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089.

[1] Tomas Bata University in Zlín, Faculty of Applied Informatics, nám. T. G. Masaryka 5555, 760 05 Zlín, Czech Republic, E-mail Address: oplatkova@fai.utb.cz

[2]Tomas Bata University in Zlín, Faculty of Applied Informatics, nam. T. G. Masaryka 5555, 760 01 Zlín, Czech Republic, E-mail Address: senkerik@fai.utb.cz

Introduction

The paper deals with a metaevolution. Metaevolution belongs to the softcomputing methods and is connected with evolutionary algorithms [1]. This is the field where huge computations are done every day in different domains of human activities. Evolutionary algorithms are mainly focused on optimization tasks. Everybody wants to maximize profit and minimize cost. This means optimizing in every task of industry, transportation, medicine, everywhere. For these purposes, we need to have suitable tools, which are able to solve very difficult and complicated problems. As previous years proved, usage of artificial intelligence and soft computing contribute to improvements in a lot of activities.

Metaevolution is computation connected with evolutionary algorithms but in different tasks. The paper has the aim to explain the difference between evolutionary algorithms and metaevolutionary approach.

Evolutionary algorithms are a group of algorithms, which use their special operators as mutation, crossover and others to find an ideal solution. Possible candidates are defined by a cost function which arguments are values of each solution. The best one is in the global extreme – maximum or minimum [1].

These evolutionary algorithms have been known for decades and live through the advancement from the weaker ones to more robust ones, which are used with success in a lot of tasks nowadays. Since their first appearance there is quite long queue of representatives: Genetic Algorithms [2], Differential Evolution [3], Self-Organizing Migrating Algorithm [4], Particle Swarm Intelligence [5], Ant Colony Optimization [6], Artificial Immune system [7]. In optimization, algorithms belong also to some stochastic and deterministic ones: Hill Climbing [8], Simulated Annealing [9], Monte Carlo and a lot of others or their mutations [1].

These techniques promise fast optimization compared to classical mathematical approach. On the other hand, also between these optimization techniques is possible to find better and worse. Their behaviour was described in a lot of references.

And the research in this area is still full of white places. There is wide field of possible applications as tuning of parameters, making of comparisons, trying to find new ones somehow. Some of the mentioned tasks may be used also with metaevolution.

Metaevolution means three main approaches:

choice of optimal evolutionary algorithm, best types of evolutionary operators and settings their parameters for given problem – one evolutionary algorithm tunes another one.

estimation of coefficients in symbolic regression in more difficult cases - one evolutionary algorithms for the main process and the second for the estimation - 2 evolutionary algorithms which helps each other.

evolutionary algorithm with symbolic regression tool breeds a completely new structure (even optimization algorithm of evolutionary character), not only parameters.

As mentioned above, evolutionary algorithms are connected with the domain of symbolic regression. Nowadays, mainly three are known for that – Genetic Programming [10] - [12], Grammatical Evolution [13] - [15] and superstructure of evolutionary algorithms – Analytic Programming [16] - [24]. These techniques can produce a complex formula from basic functions according to required behaviour of function in the case of mathematical data set, of an electronic circuit, trajectory of robots, etc.

Also, some other approaches to the field of symbolic regression can be found – either based only on evolutionary techniques or hybrid ones. Interesting investigations using symbolic regression were showed by Johnson [25] working on Artificial Immune Systems and Salustowicz in Probabilistic Incremental Program Evolution (PIPE) [26] which generates programs from an adaptive probability distribution over all possible programs. To

Grammatical Evolution foreruns GADS, which solves the approach to grammar [27], [28]. Also from evolutionary algorithm artificial immune systems evolved the artificial immune system programming for symbolic regression [29]. Approaches which differ in representation and grammar are described in gene expression programming [30], multiexpression programming [31], meta-modelling by symbolic regression and pareto Simulated Annealing [32]. To the group of hybrid approaches, it belongs mainly numerical methods connected with evolutionary systems, e.g. [33].

Analytic Programming

Basic principles of the AP were developed in 2001 [34], [35]. The core of AP is based on a special set of mathematical objects and operations. The set of mathematical objects is set of functions, operators and so-called terminals (as well as in GP), which are usually constants or independent variables. This set of variables is usually mixed together and consists of functions with different number of arguments. Because of a variability of the content of this set, it is called here "general functional set" – GFS. The structure of GFS is created by subsets of functions according to the number of their arguments. For example GFSall is a set of all functions, operators and terminals, GFS3arg is a subset containing functions with only three arguments, GFS0arg represents only terminals, etc. The subset structure presence in GFS is vitally important for AP. It is used to avoid synthesis of pathological programs, i.e. programs containing functions without arguments, etc. The content of GFS is dependent only on the user. Various functions and terminals can be mixed together [34], [35].

The second part of the AP core is a sequence of mathematical operations, which are used for the program synthesis. These operations are used to transform an individual of a population into a suitable program. Mathematically stated, it is a mapping from an individual domain into a program domain. This mapping consists of two main parts. The first part is called discrete set handling (DSH) (Fig. 1) [34], [35] and the second one stands for security procedures which do not allow synthesizing pathological programs. The method of DSH, when used, allows handling arbitrary objects including nonnumeric objects like linguistic terms {hot, cold, dark…}, logic terms (True, False) or other user-defined functions. In the AP DSH is used to map an individual into GFS and together with security procedures creates the above-mentioned mapping, which transforms arbitrary individual into a program.

AP needs some evolutionary algorithm [34], [35] that consists of population of individuals for its run. Individuals in the population consist of integer parameters, i.e. an individual is an integer index pointing into GFS. The creation of the program can be schematically observed in Fig. 2. The individual contains numbers which are indices into GFS. The detailed description is represented in [34], [35].

AP exists in 3 versions – basic without constant estimation, APnf – estimation by means of nonlinear fitting package in Mathematica environment and APmeta – constant estimation by means of another evolutionary algorithms; meta means meta-evolution.

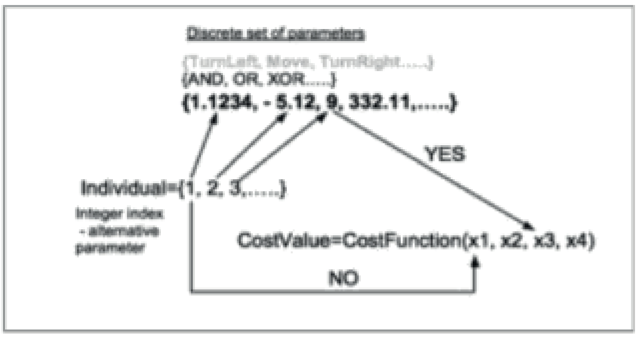

Fig. 1 Discrete set handling

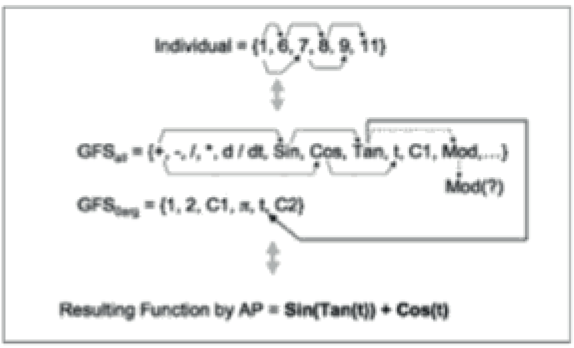

Fig. 2 Lorem Ipsum dolor amet, consectetur

Applications symbolic regression, evolution and metaevolution

The following three applications show the difference between usage of symbolic regression just with an evolutionary algorithm and symbolic regression with metaevolution where a second evolutionary algorithm has to help to estimate coefficients in the found solution. The last application will show the metaevolution in the case of breeding a new optimization algorithm of the evolutionary character – evolution breeds evolution.

Design of electronic circuit

The aim was to find a suitable shape of circuits which would behave according to a given truth table. The even-k-parity problem means that the number of k inputs with value true is even. Even-3, 4, 5, 6-parity problems were carried out. For the symmetry problems the situation is different in that the true values in inputs should be symmetric. Also for k - symmetry problems we did simulations where k was 3, 4, 5 and 6 [35]. Because of the dimensions of the truth tables, there are only 3 – parity (Table 1) and 3 – symmetry problems (Table 2) for illustration. The GFS consists of Input A, Input B,

Table 1 Truth table of 3- parity problem

Input A	Input B	Input C	Output
True	True	True	False
True	True	False	True
True	False	True	True
False	True	True	True
True	False	False	False
False	True	False	False
False	False	True	False
False	False	False	True

Input C and operators like AND, NAND, OR. Nothing else was necessary, no coefficient estimation. The only one evolutionary algorithm helps the AP to find the best option for k-symmetry or k-parity problem.

Table 2 Truth table of 3- symmetry problem

Input A	Input B	Input C	Output
True	True	True	True
True	True	False	False
True	False	True	True
False	True	True	False
True	False	False	False
False	True	False	True
False	False	True	False
False	False	False	True

Final output of the 3 parity problem is e.g. in (1). The first version has the longer output. This extended version contains redundant data like input A AND input A, which is still input A. This was then shortened in the version in (1).

$$\neg C \wedge B \wedge A \vee \neg B \wedge C \wedge A \vee \neg A \wedge C \wedge B \vee \neg C \wedge \neg B \wedge \neg A \quad (1)$$

The all 50 simulations and repetitions were able to find the rule for k-parity and k-symmetry problem. Different evolutionary algorithms for the main process of Analytic Programming were used. More details can be found in [35].

Metaevolution – 2 algorithms helps each other in AP

The study case comes from the field of deterministic chaos. Interests about deterministic chaos increase day by day. To control these kind of systems is not easy and specialists look for the way of effective control tool every day. One possibility is to use some classical optimization techniques but we have used evolutionary techniques for faster and better optimization [36]. Used chaotic systems were Logistic equation, Hénon map and others.

The logistic equation (logistic map) is a one-dimensional discrete-time example of how complex chaotic behaviour can arise from very simple non-linear dynamical equation. This chaotic system was introduced and popularized by the biologist Robert May [37]. It was originally introduced as a demographic model as a typical predator – prey relationship. The chaotic behaviour can be observed by varying the parameter r. At r = 3.57 is the beginning of chaos, at the end of the period-doubling behaviour. At r > 3.57 the system exhibits chaotic behaviour. The example of this behavior can be clearly seen from bifurcation diagram – Fig. 3.

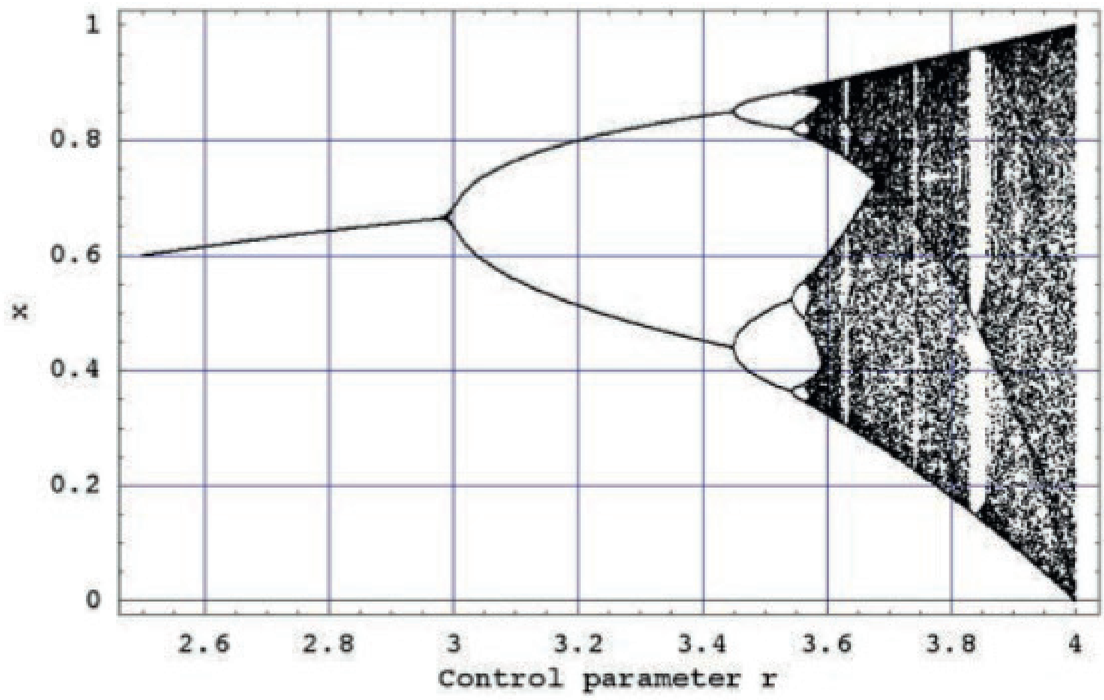

Fig. 3 Bifurcation diagram of Logistic equation

Evolutionary techniques were used in all above-mentioned chaotic systems for optimization of parameters, which can be accessed from Pyragas method: Extended delay feedback control – ETDAS [36]. This is also the case of usage of only one evolutionary algorithm to find the best option for parameters.

There exist also another approach which we wanted to demonstrate – not only to use a predefined control law and to find suitable coefficients but also to synthesize the whole control rule including the values of coefficients [38] - [40] in a similar way, as in the case of data approximation described above. Therefore the second evolutionary algorithm was used for coefficients estimation, this method is called metaevolution [35]. The second algorithm is used in the Analytic Programming to find coefficients and the first algorithm is for the main procedure of AP. The second algorithm helps the first one to find a whole solution.

Examples of synthesized solution are given in following notations (2), (3) and a stabilized system for 1-orbit for the logistic equation is depicted in Fig. 4.

$$F_n = (K_1 + K_2) * (x_{n-1} - x_n + \frac{x_{n-1} - x_n}{1 + x_{n-1}})$$

(2) without Ks estimation

$$F_n = -0.377047 * (x_{n-1} - x_n + \frac{x_{n-1} - x_n}{1 + x_{n-1}})$$

(3) with Ks estimation

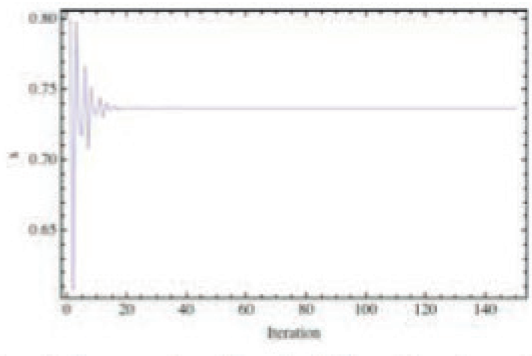

Fig. 4 Example of a stabilized system for 1p-orbit

Metaevolution – one evolutionary algorithm breeds a new evolutionary algorithm

The objective was to try to create a new optimization algorithm, probably of evolutionary character, which could be robust and effective to optimize difficult problems in the world. This is a metaevolution in third context, which was mentioned in this paper. According to previous approaches, metaevolution is determining the optimal evolutionary algorithm, best types of evolutionary operator and their parameter setting for a given problem. It means basically, that one evolutionary algorithm tunes another one [35]. But this approach is different. The metaevolution is used on a higher level for creating a new algorithm completely, not only for setting of its parameters [35].

The simulations used different operators of known evolutionary algorithms like their mutation or crossover operators and found following notations for new algorithms (4) – (7):

SOMAATORandWithoutPRT(SOMAATORandWithPRT(SOMAATORandWithPRT(MutateDECurrentToBest(SelectSOMALeader)))) (4)

SOMAATOWithPRT(SOMAATOWithPRT(SOMAATORandWithPRT(CrossDEBin(SOMAATOWithPRT(SelectSOMARandLeader))))) (5)

CrossDEBin(SOMAATOWithPRT(MutateDECurrentToBest(SelectSOMALeader))) (6)

SOMAATOWithPRT(SelectSOMALeader) (7)

These algorithms then were tested on standard benchmark functions for evolutionary algorithms to find out how effective they are. Here only one table (Table 3) is presented which shows that algorithms were competed not only between themselves but also in dimensions - 2D, 20 D and 100 D.

The numbers are for each benchmark problem as follows: 1- 1st De Jong's function, 2 - 2nd De Jong's function, 3 - 3rd De Jong's function, 4 - 4th De Jong's function, 5 - Rastrigin's function, 6 - Schwefel's function, 7 -

Table 3 Winner for each benchmark function.

	Algorithm 1	Algorithm 2	Algorithm 3	SOMAATO
2 D	1, 3, 4, 5, 6, 8, 10, 11, 12, 15, 16	5, 6, 7, 8, 10, 11, 12, 13, 15, 16	5, 6, 7, 8, 10, 11, 12, 13, 15, 16	2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
20 D	1, 3, 4, 7, 11, 14, 15	5, 6, 8, 10, 12, 13, 16	2	9
100 D	12, 13, 14	1, 2, 3, 5, 7, 8, 15, 16,		4, 6, 9, 10, 11

Griewangk's function, 8 - Sine Envelope Sine Wave function, 9 - Stretched V sine wave function - Ackley, 10 - Ackley test function, 11 - Ackley function, 12 - Egg Holder function, 13 - Rana's function, 14 - Pathological function, 15 - Michalewicz's function, 16 - Master's cosine wave function.

If the same number appears in more cells on the same row it means that algorithms finished in the same cost value.

Conclusion

This paper presents introduction into metaevolutionary techniques connected with softcomputing methods - evolutionary algorithms and symbolic regression. Presented applications demonstrate different fields and different techniques of usage of these techniques. The paper explains the difference between evolutionary and metaevolutionary approach within Analytic Programming. Applications mentioned here are only a small part of simulations and tasks performed out at our department in this field of research. As described, these techniques have a wide range of possibilities of application.

References

[1] Back T., Fogel D. B., Michalewicz Z., Handbook of evolutionary algorithms, Oxford University Press, 1997, ISBN 0750303921

[2] Dawis L., Handbook of Genetic Algorithms, International Thomson Computer Press, 1996, ISBN 1850328250

[3] Price K., Storn R. M., Lampinen J. A., Differential Evolution : A Practical Approach to Global Optimization (Natural Computing Series), Springer; 1 edition , 2005, ISBN: 3540209506

[4] Zelinka I., „SOMA – Self Organizing Migrating Algorithm“, In: Babu B.V., Onwubolu G. (eds), New Optimization Techniques in Engineering, Springer-Verlag, 2004, ISBN 3-540-20167X

[5] Eberhart R., Kennedy J., Swarm Intelligence (The Morgan Kaufmann Series in Artificial Intelligence), Morgan Kaufmann, 2001, ISBN 1558605959

[6] Dorigo M., Ant Colony Optimization and Swarm Intelligence, Springer, 2006, ISBN 3540226729

[7] Farmer J.D., Packard N., Perelson A., "The immune system, adaptation and machine learning", Physica D, vol. 2, pp. 187—204, 1986

[8] Russel S. J., Norvig P., Artificial Intelligence: Modern Approach, Prentice Hall, 1995, ISBN: 0131038052

[9] Kirkpatrick S., Gelatt C. D., Vecchi M. P., Optimization by Simulated Annealing, Science, 13 May 1983, Volume 220, Number 4598, p. 671 – 680

[10] Koza J. R., Genetic Programming, MIT Press, 1998, ISBN 0-262-11189-6

[11] Koza J. R. et al., Genetic Programming III; Darwinian Invention and problem Solving, Morgan Kaufmann Publisher, 1999, ISBN 1-55860-543-6

[12] www.genetic-programming.org

[13] O'Neill M., Ryan C., Grammatical Evolution. Evolutionary Automatic Programming in an Arbitrary Language, Kluwer Academic Publishers, 2003, ISBN 1402074441

[14] www.grammatical-evolution.org

[15] O'Sullivan J., Ryan C., An Investigation into the Use of Different Search Strategies with Grammatical Evolution, Proceedings of the 5th European Conference on Genetic Programming, p.268 - 277, 2002, Springer-Verlag London, UK, ISBN:3-540-43378-3

[16] Zelinka I., Analytic Programming by Means of Soma Algorithm. ICICIS'02, First International Conference on Intelligent Computing and Information Systems, Egypt, Cairo, 2002, ISBN 977-237-172-3

[17] Zelinka I., Analytic Programming by Means of Soma Algorithm. Mendel '02, In: Proc. 8th International Conference on Soft Computing Mendel'02, Brno, Czech Republic, 2002, 93-101., ISBN 80-214-2135-5

[18] Zelinka I., Oplatkova Z., Analytic programming – Comparative Study. CIRAS'03, The second International Conference on Computational Intelligence, Robotics, and Autonomous Systems, Singapore, 2003, ISSN 0219-6131

[19] Zelinka I.,Oplatkova Z, Nolle L., Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, ESM '2004, In: Proc. 18th European Simulation Multiconference, Magdeburg, Germany 2004

[20] Zelinka I.,Oplatkova Z, Nolle L., Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, International Journal of Simulation Systems, Science and Technology, Volume 6, Number 9, August 2005, pages 44 - 56, ISSN: 1473-8031, online

http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-6/No.9/cover.htm, ISSN: 1473-804x

[21] Oplatkova Z., Optimal Trajectory of Robots Using Symbolic Regression, In: CD-ROM of Proc. 56th International Astronautical Congress 2005, Fukuoka, Japan, 2005, paper nr. IAC-05-C1.4.07

[22] Oplatkova Z., Zelinka I., Investigation on Artificial Ant using Analytic Programming, GECCO 2006, Seattle, Washington, USA, 8 – 12 July 2006, ISBN 1-59593-186-4

[23] Zelinka I., Varacha P., Oplatkova Z., Evolutionary Synthesis of Neural Network, Mendel 2006 – 12th International Conference on Softcomputing, Brno, Czech Republic, 31 May – 2 June 2006, pages 25 – 31, ISBN 80-214-3195-4

[24] Zelinka I., Chen G., Celikovsky S., Chaos Synthesis by Means of Symbolic Regression, International Journal of Bifurcation and Chaos, in print

[25] Johnson Colin G., Artificial immune systems programming for symbolic regression, In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, and E. Costa, editors, Genetic Programming: 6th European Conference, LNCS 2610, p. 345-353, 2003

[26] Salustowicz R. P., Schmidhuber J., Probabilistic Incremental Program Evolution, Evolutionary Computation, vol. 5, nr. 2, 1997, pages 123 – 141, MIT Press, ISSN 1063-6560

[27] Paterson N., Genetic Programming with context sensitive grammars, doctoral thesis, University of St. Andrews, 2003

[28] Paterson N., Livesey M., Distinguishing genotype and phenotype in genetic programming, In Koza, Goldberg, Fogel & Riolo, eds. Late Breaking Papers at GP 1996, MIT Press, 1996, ISBN 0-18-201-031-7

[29] Johnson C. G., Artificial Immune System Programming for Symbolic Regression, Lecture notes in Computer Sciences series, Springer, Volume 2610/2003, 2003, ISSN 0302-9743.

[30] Ferreira C., Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence, Springer, 2006, ISBN: 3540327967

[31] Oltean M., Grosan C., Evolving Evolutionary Algorithms using Multi Expression Programming, The 7th European Conference on Artificial Life, September 14-17, 2003, Dortmund, Edited by W. Banzhaf (et al), LNAI 2801, pp. 651-658, Springer-Verlag, Berlin, 2003

[32] Stinstra E., Rennen G., Teeuwen G., Meta-modelling by symbolic regression and Pareto Simulated Annealing, Tilburg University, Netherlands, nr. 2006-15, ISSN 0924-7815

[33] Davidson J.W., Savic D.A., Walters G.A., Symbolic and numerical regression: experiments and applications, Informatics and Computer Science – An international Journal, Vol. 150, Elsevier, USA, 2003, pages 95 – 117, ISSN:0020-0255

[34] Zelinka et al.: Analytical Programming - a Novel Approach for Evolutionary Synthesis of Symbolic Structures, in Kita E.: Evolutionary Algorithms, InTech 2011, ISBN: 978-953-307-171-8

[35] Oplatkova, Z.: Metaevolution: Synthesis of Optimization Algorithms by means of Symbolic Regression and Evolutionary Algorithms, Lambert Academic Publishing, 154 p., 2009, ISBN: 978-3-8383-1808-0

[36] Senkerik, R.: Optimal Control of Discrete Chaotic Systems : Evolutionary Techniques in the Task of Chaos Control Optimization. 1st edition. Germany : Lambert Academic Publishing. ISBN 978-3-8383-1365-8.

[37] May R.M., "Stability and Complexity in Model Ecosystems", Princeton University Press, 2001, ISBN: 0-691-08861-6.

[38] Oplatkova Z., Senkerik R., Zelinka I. Holoska J.: Synthesis of Control Law for Chaotic Logistic Equation - Preliminary study, AMS 2010, Kota Kinabalu, Borneo, Malaysia, IEEE, 6 p., ISBN 978-0-7695-4062-7

[39] Oplatkova Z., Senkerik R., Zelinka I. Holoska J.: Synthesis of Control Law for Chaotic Henon System - Preliminary study, ECMS 2010, Kuala Lumpur, Malaysia, p. 277-282, ISBN 978-0-9564944-0-5

[40] Oplatkova Z., Senkerik R., Belaskova S., Zelinka I.: Synthesis of Control Rule for Synthesized Chaotic System by means of Evolutionary Techniques, Mendel 2010, Brno, Czech Republic, p. 91 - 98, ISBN 978-80-214-4120-0