UTILIZATION OF ANALYTIC PROGRAMMING FOR THE STABILIZATION OF HIGH ORDER OSCILLATIONS OF CHAOTIC HÉNON MAP

[1]Zuzana Oplatkova, [1]Roman Senkerik, [2]Ivan Zelinka, [2]Donald Davendra

[1]Tomas Bata University in Zlin , Faculty of Applied Informatics
Nam T.G. Masaryka 5555, 760 01 Zlin, Czech Republic
{senkerik , oplatkova}@fai.utb.cz

[2]Department of Computer Science, Faculty of Electrical Engineering and Computer Science
VB-TUO, 17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
{ivan.zelinka , donald.davendra}@vsb.cz

KEYWORDS
Chaos, Control, Hénon map, Evolutionary computation, Analytic programming, SOMA, Differential Evolution.

ABSTRACT

This research deals with a utilization of tool for symbolic regression, which is analytic programming, for the purpose of the synthesis of a new control law. This synthesized chaotic controller secures the stabilization of high periodic orbit – oscillations between several values of discrete chaotic system, which is Hénon map. The paper consists of the descriptions of analytic programming as well as chaotic system, used heuristic and cost function. For experimentation, Self-Organizing Migrating Algorithm (SOMA) and Differential evolution (DE) were used.

INTRODUCTION

During the past five years, usage of new intelligent systems in engineering, technology, modeling, computing and simulations has attracted the attention of researchers worldwide. The most current methods are mostly based on soft computing, which is a discipline tightly bound to computers, representing a set of methods of special algorithms, belonging to the artificial intelligence paradigm. The most popular of these methods are neural networks, evolutionary algorithms, fuzzy logic, and genetic programming. Presently, evolutionary algorithms are known as a powerful set of tools for almost any difficult and complex optimization problem.

The interest about the interconnection between evolutionary techniques and control of chaotic systems is spread daily. First steps were done in (Senkerik et al., 2006; 2010a), (Zelinka et al., 2009), where the control law was based on Pyragas method: Extended delay feedback control – ETDAS (Pyragas, 1995). These papers were concerned to tune several parameters inside the control technique for chaotic system. Compared to previous research, this paper shows a possibility how to generate the whole control law (not only to optimize several parameters) for the purpose of stabilization of a chaotic system. The synthesis of control is inspired by the Pyragas's delayed feedback control technique (Just, 1999), (Pyragas, 1992). Unlike the original OGY control method (Ott et al., 1990), it can be simply considered as a targeting and stabilizing algorithm together in one package (Kwon, 1999). Another big advantage of the Pyragas method for evolutionary computation is the amount of accessible control parameters, which can be easily tuned by means of evolutionary algorithms (EA).

Instead of EA utilization, analytic programming (AP) is used in this research. AP is a superstructure of EAs and is used for synthesis of analytic solution according to the required behaviour. Control law from the proposed system can be viewed as a symbolic structure, which can be synthesized according to the requirements for the stabilization of the chaotic system. The advantage is that it is not necessary to have some "preliminary" control law and to estimate its parameters only. This system will generate the whole structure of the law even with suitable parameter values.

This work is focused on the expansion of AP application for synthesis of a whole control law instead of parameters tuning for existing and commonly used method control law to stabilize desired Unstable Periodic Orbits (UPO) of chaotic systems.

This work is an extension of previous research (Oplatkova et al., 2010a; 2010b), (Senkerik et al., 2010b) focused on stabilization of simple p-1 orbit – stable state and p-2 orbit. In general, this research is concerned to stabilize p-4 UPO – high periodic orbit (oscillations between four values).

Firstly, AP is explained, and then a problem design is proposed. The next sections are focused on the description of used cost function and evolutionary algorithms. Results and conclusion follow afterwards.

PROBLEM DESIGN

The brief description of used chaotic systems and original feedback chaos control method, ETDAS is given. The ETDAS control technique was used in this research as an inspiration for synthesizing a new

Proceedings 26th European Conference on Modelling and Simulation ©ECMS Klaus G. Troitzsch, Michael Möhring, Ulf Lotzmann (Editors)
ISBN: 978-0-9564944-4-3 / ISBN: 978-0-9564944-5-0 (CD)

feedback control law by means of evolutionary techniques.

Selected chaotic system

The chosen example of chaotic system was the two dimensional Hénon map in form (1):

$$\begin{aligned} x_{n+1} &= a - x_n^2 + by_n \\ y_{n+1} &= x_n \end{aligned} \tag{1}$$

This is a model invented with a mathematical motivation to investigate chaos. The Hénon map is a discrete-time dynamical system, which was introduced as a simplified model of the Poincaré map for the Lorenz system. It is one of the most studied examples of dynamical systems that exhibit chaotic behavior. The map depends on two parameters, *a* and *b*, which for the canonical Hénon map have values of $a = 1.4$ and $b = 0.3$. For these canonical values the Hénon map is chaotic (Hilborn 2000). The example of this chaotic behavior can be clearly seen from bifurcation diagram – Figure 1.

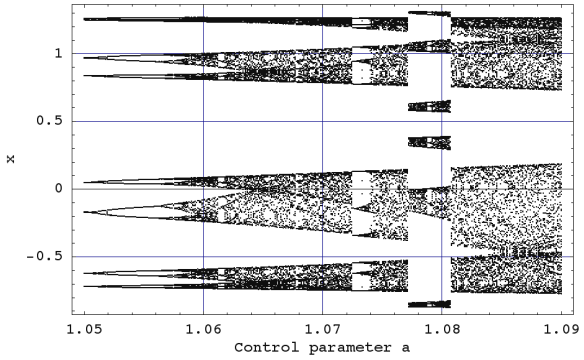

Figure 1: Bifurcation diagram of Hénon Map

Figure 1 shows the bifurcation diagram for the Hénon map created by plotting of a variable *x* as a function of the one control parameter for the fixed second parameter.

ETDAS control method

This work is focused on explanation of application of AP for synthesis of a whole control law instead of demanding tuning of EDTAS method control law to stabilize desired Unstable Periodic Orbits (UPO). In this research desired UPO is only p-2 (higher periodic orbit – oscillation between two values). ETDAS method was obviously an inspiration for preparation of sets of basic functions and operators for AP.

The original control method – ETDAS has form (2).

$$\begin{aligned} F(t) &= K\left[(1-R)S(t-\tau_d) - x(t)\right] \\ S(t) &= x(t) + RS(t-\tau_d) \end{aligned} \tag{2}$$

Where: *K* and *R* are adjustable constants, *F* is the perturbation; *S* is given by a delay equation utilizing previous states of the system and τ_d is a time delay.

The original control method – ETDAS in the discrete form suitable for two-dimensional Hénon map has the form (3).

$$\begin{aligned} x_{n+1} &= a - x_n^2 + by_n + F_n \\ F_n &= K\left[(1-R)S_{n-m} - x_n\right] \qquad (3) \\ S_n &= x_n + RS_{n-m} \qquad (3) \end{aligned}$$

Where: *m* is the period of *m*-periodic orbit to be stabilized. The perturbation F_n in equations (3) may have arbitrarily large value, which can cause diverging of the system outside the interval {0, 1.0}. Therefore, F_n should have a value between $-F_{\max}$, $F_{\max}$. In this preliminary study a suitable $F_{\max}$ value was taken from the previous research. To find the optimal value also for this parameter is in future plans.

Previous research concentrated on synthesis of control law only for p-1 orbit (a fixed point). An inspiration for preparation of sets of basic functions and operators for AP was simpler TDAS control method (4) and its discrete form suitable for logistic equation given in (5).

$$F(t) = K\left[x(t-\tau) - x(t)\right] \tag{4}$$

$$F_n = K\left(x_{n-m} - x_n\right) \tag{5}$$

ANALYTIC PROGRAMMING

Basic principles of the AP were developed in 2001 (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009). Until that time only genetic programming (GP) and grammatical evolution (GE) had existed. GP uses genetic algorithms while AP can be used with any evolutionary algorithm, independently on individual representation. To avoid any confusion, based on use of names according to the used algorithm, the name - Analytic Programming was chosen, since AP represents synthesis of analytical solution by means of evolutionary algorithms.

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 GFS_{all} is a set of all functions, operators and terminals, GFS_{3arg} is a subset containing functions with only three arguments, GFS_{0arg} 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 (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009).

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) (See Figure 2) (Zelinka et al. 2005, Lampinen and Zelinka, 1999) 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.

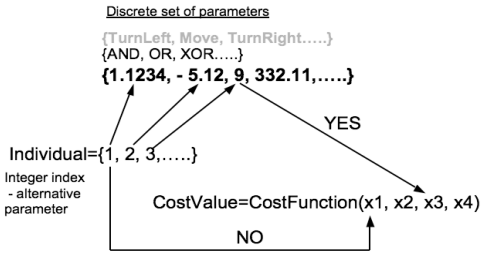

Figure 2: Discrete set handling

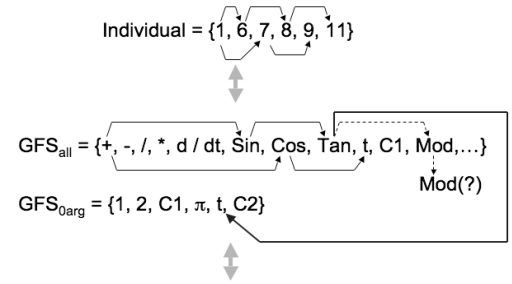

Figure 3: Main principles of AP

AP needs some evolutionary algorithm (Zelinka, 2004) 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 3. The individual contains numbers which are indices into GFS. The detailed description is represented in (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009).

AP exists in 3 versions – basic without constant estimation, AP_{nf} – estimation by means of nonlinear fitting package in Mathematica environment and AP_{meta} – constant estimation by means of another evolutionary algorithms; meta means metaevolution.

COST FUNCTION

Proposal for the cost function comes from the simplest Cost Function (CF). The core of CF could be used only for the stabilization of p-1 orbit. The idea was to minimize the area created by the difference between the required state and the real system output on the whole simulation interval – τ_i.

But another universal cost function had to be used for stabilizing of higher periodic orbit and having the possibility of adding penalization rules. It was synthesized from the simple CF and other terms were added. In this case, it is not possible to use the simple rule of minimizing the area created by the difference between the required and actual state on the whole simulation interval – τ_i, due to many serious reasons, for example: degrading of the possible best solution by phase shift of periodic orbit.

This CF is in general based on searching for desired stabilized periodic orbit and thereafter calculation of the difference between desired and found actual periodic orbit on the short time interval - τ_s (40 iterations) from the point, where the first min. value of difference between desired and actual system output is found. Such a design of CF should secure the successful stabilization of either p-1 orbit (stable state) or higher periodic orbit anywise phase shifted. The CF_{Basic} has the form (6).

$$CF_{Basic} = pen_1 + \sum_{t=\tau 1}^{\tau 2} \left| TS_t - AS_t \right|, \tag{6}$$

where:

TS - target state, AS - actual state

τ_1 - the first min value of difference between TS and AS

τ_2 – the end of optimization interval ($\tau_1 + \tau_s$)

$pen_1 = 0$ if $\tau_i - \tau_2 \geq \tau_s$;

$pen_1 = 10*(\tau_i - \tau_2)$ if $\tau_i - \tau_2 < \tau_s$ (i.e. late stabilization).

USED EVOLUTIONARY ALGORITHMS

This research used two evolutionary algorithms: Self-Organizing Migrating Algorithm (Zelinka, 2004), Differential Evolution (Price, 2005). Future simulations expect a usage of soft computing GAHC algorithm (modification of HC12) (Matousek, 2007) and a CUDA implementation of HC12 algorithm (Matousek, 2010).

Self Organizing Migrating Algorithm - SOMA

Self Organizing Migrating Algorithm (SOMA) is a stochastic optimization algorithm that is modelled on the social behaviour of cooperating individuals (Zelinka, 2004). It was chosen because it has been proven that the algorithm has the ability to converge towards the global optimum (Zelinka, 2004). SOMA works on a population of candidate solutions in loops called *migration loops*. The population is initialized randomly distributed over the search space at the beginning of the search. In each loop, the population is evaluated and the solution with the highest fitness becomes the leader *L*. Apart from the leader, in one migration loop, all individuals will traverse the input

space in the direction of the leader. Mutation, the random perturbation of individuals, is an important operation for evolutionary strategies (ES). It ensures the diversity amongst the individuals and it also provides the means to restore lost information in a population. Mutation is different in SOMA compared with other ES strategies. SOMA uses a parameter called PRT to achieve perturbation. This parameter has the same effect for SOMA as mutation has for genetic algorithms.

The novelty of this approach is that the PRT Vector is created before an individual starts its journey over the search space. The PRT Vector defines the final movement of an active individual in search space.

The randomly generated binary perturbation vector controls the allowed dimensions for an individual. If an element of the perturbation vector is set to zero, then the individual is not allowed to change its position in the corresponding dimension.

An individual will travel a certain distance (called the PathLength) towards the leader in *n* steps of defined length. If the PathLength is chosen to be greater than one, then the individual will overshoot the leader. This path is perturbed randomly. The main principle is depicted in Figures 4 and 5.

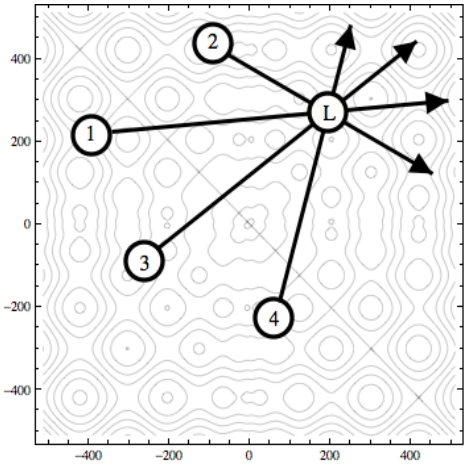

Figure 4: Principle of SOMA, movement in the direction towards the Leader

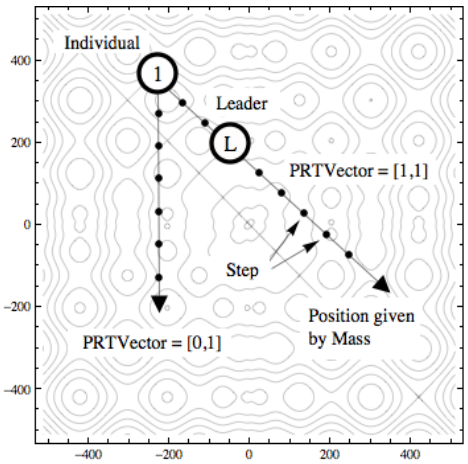

Figure 5: Basic principle of crossover in SOMA, PathLength is replaced here by Mass

Differential evolution

DE is a population-based optimization method that works on real-number-coded individuals (Price, 2005). For each individual $\vec{x}_{i,G}$ in the current generation G, DE generates a new trial individual $\vec{x}'_{i,G}$ by adding the weighted difference between two randomly selected individuals $\vec{x}_{r1,G}$ and $\vec{x}_{r2,G}$ to a randomly selected third individual $\vec{x}_{r3,G}$. The resulting individual $\vec{x}'_{i,G}$ is crossed-over with the original individual $\vec{x}_{i,G}$. The fitness of the resulting individual, referred to as a perturbed vector $\vec{u}_{i,G+1}$, is then compared with the fitness of $\vec{x}_{i,G}$. If the fitness of $\vec{u}_{i,G+1}$ is greater than the fitness of $\vec{x}_{i,G}$, then $\vec{x}_{i,G}$ is replaced with $\vec{u}_{i,G+1}$; otherwise, $\vec{x}_{i,G}$ remains in the population as $\vec{x}_{i,G+1}$. DE is quite robust, fast, and effective, with global optimization ability. It does not require the objective function to be differentiable, and it works well even with noisy and time-dependent objective functions. Description of used DERand1Bin strategy is presented in (7). Please refer to (Price and Storn 2001, Price 2005) for the description of all other strategies.

$$u_{i,G+1} = x_{r1,G} + F \bullet \left(x_{r2,G} - x_{r3,G}\right) \tag{7}$$

RESULTS

As described in section about Analytic Programming, AP requires some EA for its run. In this paper AP_{meta} version was used. Meta-evolutionary approach means usage of one main evolutionary algorithm for AP process and second algorithm for coefficient estimation, thus to find optimal values of constants in the evolutionary synthesized control law.

SOMA algorithm was used for main AP process and DE was used in the second evolutionary process. Settings of EA parameters for both processes were based on performed numerous experiments with chaotic systems and simulations with AP_{meta} (Table 1 and Table 2).

Table 1: SOMA settings for AP

PathLength	3
Step	0.11
PRT	0.1
PopSize	50
Migrations	4
Max. CF Evaluations (CFE)	5345

Table 2: DE settings for meta-evolution

PopSize	40
F	0.8
CR	0.8
Generations	150
Max. CF Evaluations (CFE)	6000

Compared to previous research with stabilization of stable state - p-1 orbit, the data set for AP required only constants, operators like plus, minus, power and output values x_n and x_{n-1}. Due to the recursive attributes of delay equation *S* utilizing previous states of the system in discrete ETDAS (3), the data set for AP had to be expanded and cover longer system output history, thus to imitate inspiring control method for the successful synthesis of control law securing the stabilization of higher periodic orbits.

Basic set of elementary functions for AP:
GFS2arg= +, -, /, *, ^
GFS0arg= $data_{n-11}$ to $data_n$, K

Total number of 30 simulations was carried out. The most simulations were successful and have given new synthesized control law, which was able to stabilize the system at required behaviour (p-4 orbit) within short simulation interval of 200 iterations.
Total number of cost function evaluations for AP was 5345, for the second EA it was 6000, together 32.07 millions per each simulation. See Table 3 for simple CF values statistic.

Table 3: Cost Function values

Min	0.0984
Max	1.0182
Average	0.6130

The novelty of this approach represents the synthesis of feedback control law $F_n(8)$ (perturbation) for the Hénon map inspired by original ETDAS control method.

$$x_{n+1} = a - x_n^2 + by_n + F_n \tag{8}$$

Following Table 4 contains examples of synthesized control laws. Obtained simulation results can be classified into 2 groups, based on the quality and durability of stabilization at real p-4 UPO, which for unperturbed Hénon map has following values: $x_1 = 0.139$, $x_2 = 1.4495$, $x_3 = -0.8595$, $x_4 = 0.8962$. More about this phenomenon is written in conclusion section.

Table 4 covers direct output from AP – synthesized control law without coefficients estimated, further the notation with simplification after estimation by means of second algorithm DE, corresponding CF value, average error between actual and required system output, and identification of figure with simulation results.

Table 4: Simulation results

Control Law	Control Law with coefficients	CF Value	Avg. output error	Figure
$F_n = -K_1 x_{n-8} x_{n-7} x_{n-3}(x_{n-4} - x_n)$	$F_n = -0.527409 x_{n-8} x_{n-7} x_{n-3}(x_{n-4} - x_n)$	0.0984	0.0025	6a
$F_n = \frac{x_{n-2} x_n}{K_1}$	$F_n = 0.011824 x_{n-2} x_n$	0.2230	0.0057	6b
$F_n = \frac{x_{n-5} x_n}{-x_{n-8} - x_{n-4} x_{n-2} + \frac{K_1(-K_3 x_n - K_2)}{\frac{x_{n-1}}{x_{n-3}} - x_{n-8}}}$	$F_n = \frac{x_{n-5} x_n}{-x_{n-8} - x_{n-4} x_{n-2} + \frac{85.45(-65.43 x_n - 58.56)}{\frac{x_{n-1}}{x_{n-3}} - x_{n-8}}}$	0.3052	0.0076	6c
$F_n = \frac{x_{n-7} x_{n-6}(K_2 + x_n)}{K_3 - \frac{x_{n-6}}{K_4}}$ $K_1 + \frac{x_n(K_6 - K_7 + K_8)}{x_{n-3}} - K_5 + x_{n-2}$	$F_n = \frac{x_{n-7} x_{n-6}(10.0667 + x_n)}{59.4863 + \frac{-0.0174 x_{n-6} - 52.191}{\frac{x_n 5.9706}{x_{n-3}} + 39.4742 + x_{n-2}}}$	0.7095	0.0177	6d

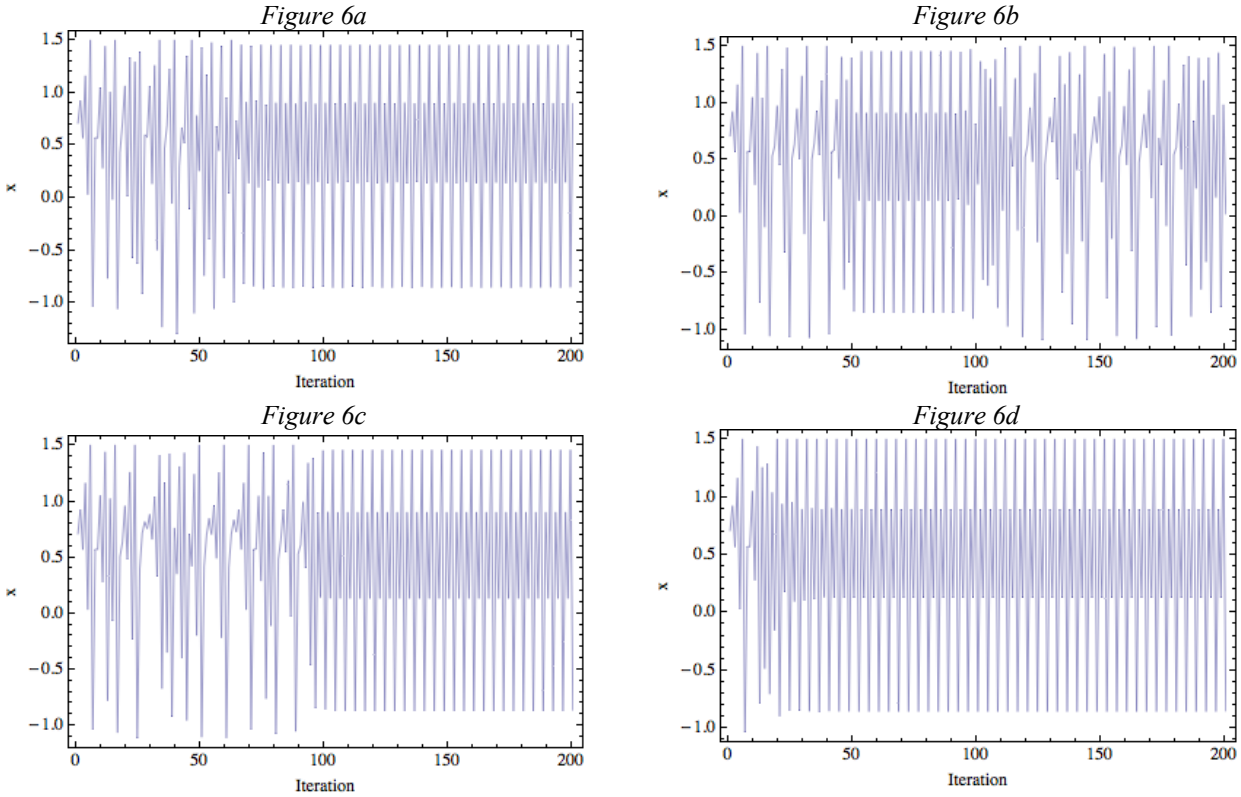

Figure 6: Examples of results – stabilization of p-4 orbit for Hénon map by means of control laws given in Table 4.

CONCLUSION

This paper deals with a synthesis of a control law by means of AP for stabilization of selected chaotic system at high periodic orbit. Hénon map as an example of two-dimensional discrete chaotic system was used in this research.

In this presented approach, the analytic programming was used instead of tuning of parameters for existing control technique by means of EA's as in the previous research.

Obtained results reinforce the argument that AP is able to solve this kind of difficult problems and to produce a new synthesized control law in a symbolic way securing desired behaviour of chaotic system and stabilization.

Presented four simulation examples show two different results. Low CF values indicating precise, but unfortunately slow stabilization and sometimes only temporary, together with simple control law in the first two cases. And according to the higher CF values not very precise, but very fast stabilization and relatively complex notation of chaotic controller in the next two cases. This phenomenon is caused by the design of CF, which was borrowed from the previous research focused on the simpler cases, which were stabilization of stable state and p-2 orbit, and it has given satisfactory results Nevertheless this fact lends weight to the argument, that AP is a powerful symbolic regression tool, which is able to strictly and precisely follow the rules given by cost function and synthesizes any symbolic formula, in the case of this research – the feedback controller for chaotic system. The question of energy costs and more precise and faster stabilization will be included into future research together with development of better cost functions, different AP data set, and performing of numerous simulations to obtain more results and produce better statistics, thus to confirm the robustness of this approach.

ACKNOWLEDGEMENT

This work was supported by the European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089 and project IT4Innovations Centre of Excellence No. CZ.1.05/1.1.00/02.0070.

REFERENCES

Hilborn R.C., 2000. Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers, Oxford University Press, 2000, ISBN: 0-19-850723-2.

Just W., 1999, "Principles of Time Delayed Feedback Control", In: Schuster H.G., Handbook of Chaos Control, Wiley-Vch, ISBN 3-527-29436-8.

Kwon O. J., 1999. "Targeting and Stabilizing Chaotic Trajectories in the Standard Map", Physics Letters A. vol. 258, 1999, pp. 229-236.

Lampinen J., Zelinka I., 1999, "New Ideas in Optimization – Mechanical Engineering Design Optimization by Differential Evolution", Volume 1, London: McGraw-hill, 1999, 20 p., ISBN 007-709506-5.

Matousek R., 2007, „GAHC: Improved GA with HC station“, In WCECS 2007, San Francisco, pp. 915-920. ISBN: 978-988-98671-6-4.

Matousek R., 2010, „HC12: The Principle of CUDA Implementation“. In MENDEL 2010, Mendel Journal series, pp. 303-308. ISBN: 978-80-214-4120- 0. ISSN: 1803- 3814.

Oplatková, Z., Zelinka, I.: 2009. Investigation on Evolutionary Synthesis of Movement Commands, Modelling and Simulation in Engineering, Volume 2009 (2009), Article ID 845080, 12 pages, Hindawi Publishing Corporation, ISSN: 1687-559.

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

Oplatkova Z., Senkerik R., Belaskova S., Zelinka I., 2010b, 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.

Ott E., C. Greboki, J.A. Yorke, 1990. “Controlling Chaos”, Phys. Rev. Lett. vol. 64, 1990, pp. 1196-1199.

Price, K. and Storn, R. (2001), *Differential evolution homepage*, [Online]: http://www.icsi.berkeley.edu/~storn/code.html, [Accessed 29/02/2012].

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

Pyragas K., 1992, “Continuous control of chaos by self-controlling feedback”, Physics Letters A, 170, 421-428.

Pyragas K., 1995. “Control of chaos via extended delay feedback”, Physics Letters A, vol. 206, 1995, pp. 323-330.

Senkerik R., Zelinka I., Navratil E., 2006, “Optimization of feedback control of chaos by evolutionary algorithms”, in proc 1st IFAC Conference on analysis and control of chaotic systems, Reims, France, pp. 97 - 102.

Senkerik R., Zelinka I., Davendra D., Oplatkova Z., 2010a, “Utilization of SOMA and differential evolution for robust stabilization of chaotic Logistic equation”, Computers & Mathematics with Applications, Volume 60, Issue 4, pp. 1026-1037.

Senkerik R., Oplatkova Z., Zelinka I., Davendra D.,Jasek R., 2010b, “Synthesis Of Feedback Controller For Chaotic Systems By Means Of Evolutionary Techniques,”, Proceeding of Fourth Global Conference on Power Control and Optimization, Sarawak, Borneo, 2010,.

Zelinka I., 2004. “SOMA – Self Organizing Migrating Algorithm”, In: *New Optimization Techniques in Engineering*, (B.V. Babu, G. Onwubolu (eds)), chapter 7, 33, Springer-Verlag, 2004, ISBN 3-540-20167X.

Zelinka I.,Oplatkova Z, Nolle L., 2005. *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.

Zelinka I., Senkerik R., Navratil E., 2009, “Investigation on evolutionary optimization of chaos control”, Chaos, Solitons & Fractals, Volume 40, Issue 1, pp. 111-129.

Zelinka, I., Guanrong Ch., Celikovsky S., 2008. Chaos Synthesis by Means of Evolutionary algorithms, International Journal of Bifurcation and Chaos, Vol. 18, No. 4 (2008) 911–942