Synthesis of artificial neural networks by the means of evolutionary scanning - preliminary study

Pavel Varacha and Ivan Zelinka
Department of Applied Informatics
Tomas Bata University in Zlin
Nad Stranemi 4511, Zlin, 760 05, Czech Republic
E-mail: {varacha,zelinka}@fai.utb.cz

KEYWORDS

neural network, symbolic regression, analytic programming, evolutionary searching

ABSTRACT

This work deals with a problem of synthesis of the artificial neural networks using the evolutional scanning method. The basic task to be solved is to create a symbolic regression algorithm on principles of analytic programming, which will be capable of performing a convenient neural network synthesis. The main motivation here is the computerization of such synthesis and discovering so far unknown solutions.

INTRODUCTION

A shape of the Artificial Neural Network (ANN) which successfully solves concrete problem depends on many factors. A designer who deals with ANN designing pays attention to complexion of problem in the first instance, for example a number of inputs and outputs. However some other aspects such as ANN topology or neurons transfer function are considered in the same time. Concurrently, final shape of ANN is heavily influenced by learning algorithm which is going to be implemented. (Bose and Liang 1996)

ANN designing operates with list of rules and heuristics to ensure correct application of chosen learning algorithm for created ANN. This technique generates relatively small set of possible ANN.

Nevertheless, if the rules of classic ANN theory are disobeyed and personal experience is replace by clever algorithm, scanned set which includes suitable ANN becomes infinite. This paper shows how symbolic regression can be used as tool for successful scanning of such huge sets in order to find and to learn appropriate ANN.

The particular method of symbolic regression in this case Analytic Programming (Zelinka and Oplatkova 2003), an algorithm developed by our faculty on a long-term basis. This algorithm proves a potential to find among a set of functions **F**, the class of functions **f*** (uneducated neural network), which is feasible to be reduced-optimized by a suitable educational method to the specific function **f** (educated neural network) having the quality to solve the particular problem successfully. Such attempt enables to further enhance the set **F** by functions of a non-neuronal character and carry a research on so far into quite unexplored territory between two purely neural and purely mathematical frameworks in such way.

The goal of this research is to build-up a symbolic regression algorithm, which will, using the evolutionary scanning method, successfully find (synthesize) suitable solutions $\mathbf{f} \in \mathbf{F}$ of the particular problem *P*.

EVOLUTIONARY SCANNING

Clause: *Let there be a set of all neural networks with forward running propagation* $\boldsymbol{ANN_{all}} = \{ANN_1, ANN_2, ..., ANN_i, ...\}$ *and a set of all functions* $\boldsymbol{F_{all}} = \{\boldsymbol{f_1}, \boldsymbol{f_2}, ..., \boldsymbol{f_k}, ...\}$. *Then for each ANNi* $\in \boldsymbol{ANN_{all}}$ *exists a function* $\boldsymbol{f_k} \in \boldsymbol{F_{all}}$, *alternatively a set of functions* $\boldsymbol{F_k} \subset \boldsymbol{F_{all}}$ *such, that holds* $ANN_i \Leftrightarrow \boldsymbol{f_k}$, *alternatively* $ANN_i \Leftrightarrow \boldsymbol{F_k}$.

Kolmonogorov theorem (Bose and Liang 1996) further shows also validity of the inverse clause: *For every continuous function* $\boldsymbol{f_k} \in \boldsymbol{F_{all}}$ *exists* $ANN_i \in \boldsymbol{ANN_{all}}$ *such, that holds* $\boldsymbol{f_k} \Leftrightarrow ANNi$.

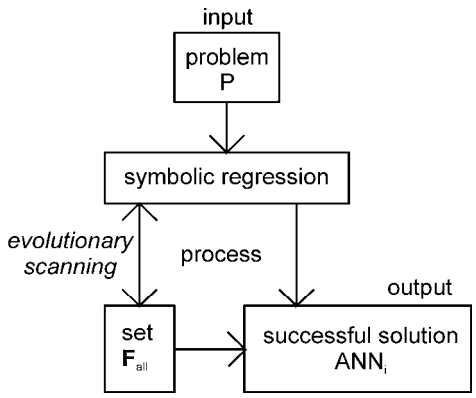
Task: Design an algorithm, which will by the means of the symbolic regression methods, evolutionary scan a set $\mathbf{F_{all}}$ in order to find:

a) $\mathbf{f_k} \Leftrightarrow \mathrm{ANN_i}$

b) $\mathbf{f_k}$, whose at least some subfunctions $\{\mathbf{f_1}, \mathbf{f_2}, ...\} \Leftrightarrow \{\mathrm{ANN_n}, \mathrm{ANN_m}, ...\}$

which solve the particular problem *P* with global error $E_T < \xi$, where ξ is the user defined biased tolerance threshold.

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)



Figures 1: Principle of the Evolutionary scanning

ANALYTIC PROGRAMMING

Algorithm referred to as analytic programming (AP) proved itself by providing the solution to a variety of problems of the symbolic regression as: synthesis of trigonometric functions (Zelinka 2005a), polynomial functions (Zelinka and Oplatkova 2003), functions of boolean parity (Zelinka and Oplatkova 2004), functions of boolean symmetry (Zelinka at al. 2004), solution of differential equations (Zelinka 2002b) and optimization of the artificial ant tour (Zelinka and Oplatkova 2006). Those accomplishments have led to the motivation to utilise an AP also in the ANN synthesis.

AP has been built-up on the sets of functions, operators and so called terminals, which are mostly invariables or independent variables. These mathematical objects form a set, out of which the AP strives to synthesize a suitable solution – so called *General Function Set (***GFS**). Members of **GFS** are then the principal functions **gf**. (That is, we can easily understand the operators and terminals as functions.) **GFS** = {$\mathbf{gf}_1$, ... **gf***i*, ... **gf***n*}.

The basic principle of AP is based on the handling with Discrete Sets (*Discrete Set Handling*, DSH). DSH forms the interface between the Evolutionary Algorithm (*Evolutionary Algorithm*, EA) (Zelinka, 2004), a problem, which should be solved symbolically.

Thanks to that, almost any EA can be applied in the AP. In such a way the EA in effect works as an engine actuating the AP. From the EA nature further results to be rational to define the Cost Function (**CF**), whose value is necessary to be minimized.

Note: *AP does not use as terminals particular, however, general constants K1 to Kn, whose particular value is determined by means of nonlinear interlacing as closely as possible near the assessment of the relevant* ***CF****.*

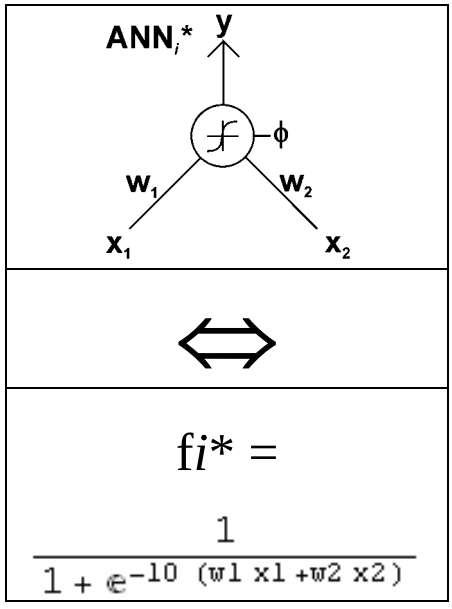
SYNTHESIS OF NEURAL NETWORKS

To accomplish by the means of an AP the successfully synthesize of the ANN, it is necessary at first to appropriately choose the elements of **GFS** and correctly define the **CF**, applicable for the problem being solved. To fulfill both those steps is, however, necessary to comprehend the relationship of equivalence between the ANN and functions – namely an AP itself does not understand the ANN conception, it is capable to synthesise only functions, eventually algorithms.

The relation between an ANN and mathematical functions

The connection between an ANN and mathematical functions is often neglected in the professional literature, at the same time each ANN can be expressed as to its equivalent class of functions **f***. The ANN synthesis process is thus a process of searching of a suitable **f* in** the set of class of functions **F**, which contains the classes accrued by combinations of (basic) base functions **b** of a set **B**. The constitution of elements of **B** then differs according to individual types of ANN. The process of ANN education can then be expressed as searching for convenient values of invariables, which differentiate particular functions **f** in **f*** from each other. The resulting function **f** is then equivalent to the searched for ANN.

Note: *Let's introduce an agreement – if generally speaking about the neural networks, which are already successfully educated and/or it is not important if they are or are not educated at the moment, it would be prudent to continue using the* ANN *shortcut. If however one wishes, wish to accent, that the matter, that it is so far uneducated network the short form* ANN* *can be utilized.*



Figures 2: Equivalence between ANNi* and fi*

That previous described feature of ANN together with aforementioned stated characteristics of an AP, makes from the AP an ideal algorithm for the ANN synthesis.

The most important task to solve, is to select a suitable **B**, which will be used as a **GFS.**

Conception of used GFS

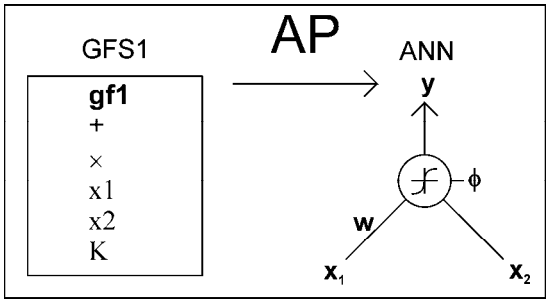
From the preceding text it is clear, that suitable selection of individual **gf** by using **GFS**, is an absolutely crucial question. As a fundamental base stone - function **gf1** – is synthesised ANN, and the choice then becomes **sigmoid** – the transfer function of two input arguments ***arg1***, ***arg2***:

$$gf1 = \frac{1}{1 + e^{-\lambda\,(arg1 + arg2)}}$$

Sigmoid function is in the world of ANN the often used transfer function (Master 1993), which is derived from behavior of real neurons in human brain. For a parameter λ the invariable value 10 was selected, so that the sigmoid nears rather to the unipolar binary function, which is suitable for solutions of classification problems. Against the binary function it , however, has the advantage, that there exists a derivation of it in its whole interval and that it is possible to apply the algorithm dependent on such derivation of function, as it is , for example, just the nonlinear interlacing used in the AP, on the structures, which are thus being created.

Beyond **gf1** there must be an integral part of, as constituted by this research, **GFS** also input independent variables x1 and x2, which represent coordinates of individual points on the classified plane, and then invariables K (K1 to Kn), which shall figure the thresholds ϕ and weights w of the network. To be able to tie up properly the weights and thresholds to the network, it is necessary to add into **GFS** next two **gf,** + (plus) and × (times).

By that one finally arrives at **GFS1** = {**gf1**, +, ×, x1, x2, K}. From the point of view of the problem it handles about so-called **complete GFS,** consequently **GFS1** has by the means of AP guaranteed the potential of solving the ANN syntheses – for the set **B,** through combination of which rises already known solution ANN of the particular problem, is valid $\mathbf{B} \subseteq \mathbf{GFS1}$ - so it comprises all elements, which build and compose the relevant **f**.



Figures 3: Principle of ANN synthesis from **GFS1**

Such definition of **GFS1** is an attractive one because an AP can, thanks to it synthesize not only classical neural structures, known from the rigorous theory of ANN, however, also the networks of pseudo-neural character, eventually quite non-neural functions. These properties from the world of neural functions shall, thanks to AP involuntarily combine with further mathematical transformations.

Effect of GFS1 on the synthesized ANN topology

The sub-functions can arise from the set **GFS1** under favorable conditions during cultivation of an individual in AP, which will compose the winner individual, and on which can be looked on as neurons in ANN. Examples of such arrangement of individual $\mathbf{gf} \in \mathbf{GFS1}$ into convenient subfunctions will be introduced in future studies.

Definition of CF

The correct definition of **CF** is unique for every problem to be solved. **CF** should in itself namely include the whole testing set of the synthesized ANN. **CF** works in such cases on the basis of testing the ANN responses on particular elements of the testing set. What, however, remains always same is the chosen philosophy, under which the relevant **CF** is created.

As a fundamental model of CF1, the staircase conception **CF** was selected for the synthesis. Its operation is, that at the beginning of scoring the **CF** of particular individual (ANN) is valid **CF** = 0. Step by step all elements of the tested set are passed through. If there shows at any tested element, that the response of ANN is divergent to the requirements of the testing set an increment of **CF** = **CF** + 1 is performed. By that way the CF1 can gather the discrete values {1,2, ..., n}, where n is number of elements in the testing set. CF1 thus calculates on how many elements of testing set replied ANN with the inadmissible reply.

By all of that it is assumed, that at the output of ANN there is inserted one more neuron, which standardizes the replies greater than 0.8 to 1 and replies less than 0.2 damps to 0.

SOLVING THE XOR PROBLEM

In the following a row of problems is selected, which should be solved successfully using the introduced methods. And the problem is a classification according to a logical function XOR. That basic classification problem ill-famed by the halting of neural networks development in 30 years of the 20th century. Because of the capacity of neural networks two variants are displayed. These are denoted as variants ANN1 and ANN2.

Table 1: Classification XOR

coordinate x1	coordinate x2	class
1	1	1
-1	-1	1
1	-1	0
-1	1	0

Description of formulas used and graphs

Structure of ANN* - suitable structure of ANN found by the means of AP with invariables K prepared for education

Structure of ANN – structure of ANN, where the constants K (weights and thresholds) were determined (educated) by the means of nonlinear interlacing methods (*nonlinear fitting*).

The border between classes of ANN – shows the classification border of ANN, if the reply is greater than 0.5 falls the classification in the class 1 on the contrary in the class 0.

3D graph of function ANN - displays ANN as a function in $\mathbf{E}^3$.

Topology of ANN – shows the topological relations **b** in ANN, transferred into neuronal shapes.

Example of a solution of ANN1

Structure of ANN1*:

$$x1\left(x1+x2+x1\,x2\,K[\![1]\!]^2\left(\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\left(\frac{x1}{1+e^{-10\,(x1+K[\![3]\!])}}+K[\![2]\!]\right)}}+K[\![4]\!]\right)}}\right)}}+K[\![5]\!]\right)(x2+K[\![6]\!]+x1^2\,K[\![7]\!])\right)$$

Structure of ANN1:

$$x1\left(x1+x2+2.6485\times10^{-26}\left(0.674245+\frac{1}{1+e^{-10\left(1.+\frac{1}{1+e^{-10\left(1.+\frac{x1}{1+e^{-10\,(1.+x1)}}\right)}}\right)}}\right)x1\,x2\,(0.783028+0.783028\,x1^2+x2)\right)$$

Structure of ANN2*:

$$\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\,(x1+x2+K[\![1]\!])}}+\frac{x1^2}{1+e^{-10\,(x2+K[\![3]\!])}}+x1\,K[\![2]\!]^2+x1\,(x2+K[\![4]\!])\right)}}+\frac{x1}{(1+e^{-10\,(x1+K[\![5]\!])})\,(1+e^{-10\,(x1+x2+K[\![6]\!])})}\right)}}+x1\,x2$$

Structure of ANN2:

$$\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\left(\frac{1}{1+e^{-10\,(1.+x1+x2)}}+1.\,x1+\frac{x1^2}{1+e^{-10\,(1.+x2)}}+x1\,(1.+x2)\right)}}+\frac{x1}{(1+e^{-10\,(1.+x1)})\,(1+e^{-10\,(1.+x1+x2)})}\right)}}+x1\,x2$$

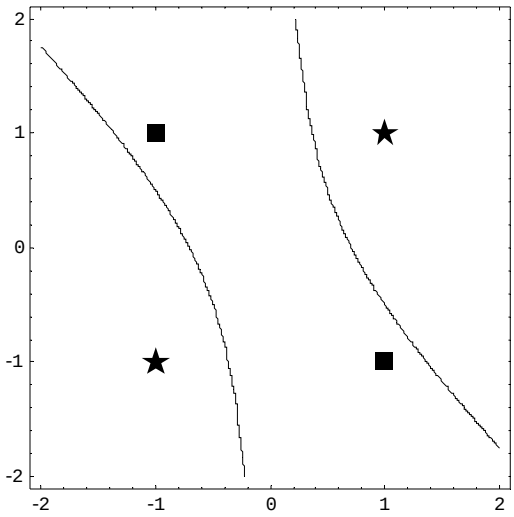


Figure.7: Border between classes of ANN1

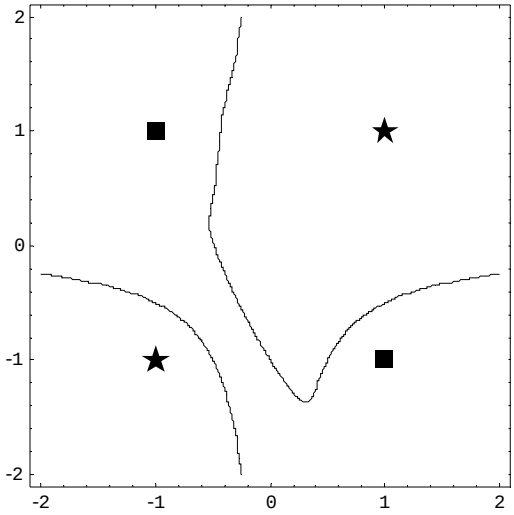


Figure.8: Border between classes of ANN2

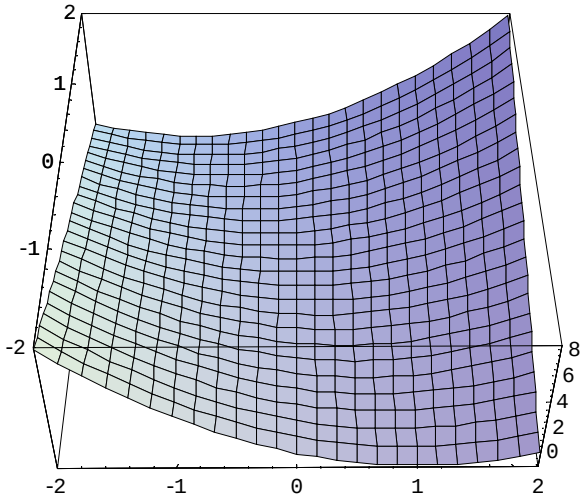


Figure 9: 3D graph of function ANN1

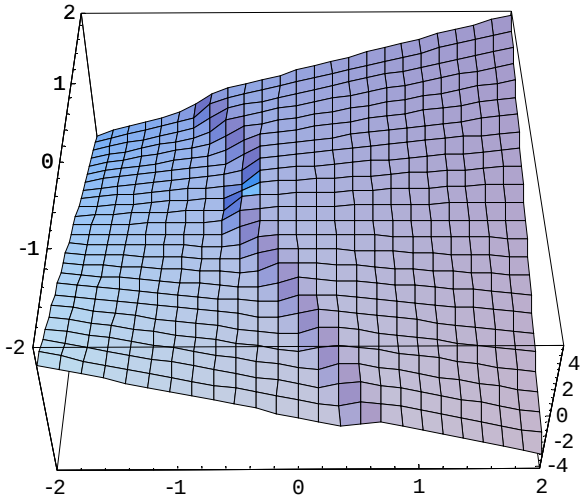


Figure 10: 3D graph of function ANN2

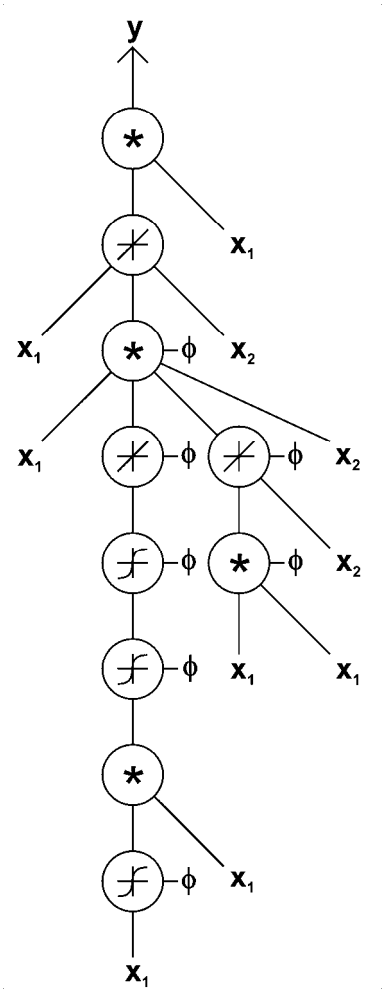


Figure 11: Topology of ANN2

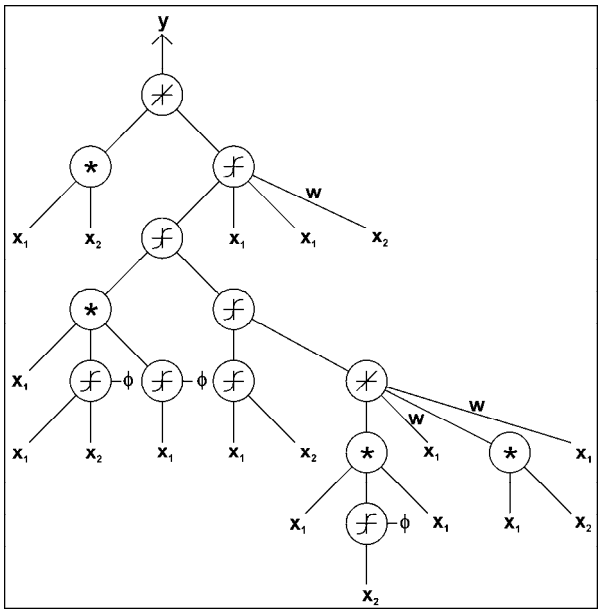


Figure 12**:** Topology of ANN2

CONCLUSION

Analytic programming has been successfully used for different types of problems. In this research a simple initial study on complete ANN synthesis by means of AP is done. Positive results has shoved that AP can be used in this way. In the future more complex study on ANN synthesis are going to be done by means of another evolutionary algorithms and levels of application.

REFERENCE

BOSE, B.K., LIANG, P. 1996. *Neural Network Fundamentals with Graphs, Algorithms, and Aplications*. McGraw-Hill Series in Electrical and Computer Engineering, 478 p. ISBN 0-07 006618-3.

MASTER T., 1993, Practical Neural Networks Recipes in C++, London : Academi Press, 1993, 490 s. ISBN 0-12-479040-2.

ZELINKA I. 2002a. Analytic programming by Means of Soma Algorithm. Mendel ’02, *In: Proc. 8th International Conference on Soft Computing Mendel’02*, Brno, Czech Republic, 93-101., ISBN 80-214-2135-5.

ZELINKA I. 2002b. Analytic programming by Means of Soma Algorithm. ICICIS’02, *First International Conference on Intelligent Computing and Information Systems*, Egypt, Cairo, ISBN 977-237-172-3

ZELINKA I., OPLATKOVÁ Z. 2003. Analytic programming – Comparative Study. *CIRAS’03, The second International Conference on Computational Intelligence, Robotics, and Autonomous Systems*, Singapore, ISSN 0219-6131.

ZELINKA I., OPLATKOVÁ Z. 2004. Boolean Parity Function Synthesis by Means of Arbitrarry Evolutionary Algorithms - Comparative Study, *In: 8th World Multicon-ference on Systemics, Cybernetics and Informatics (SCI 2004)*, Orlando, USA, in July 18-21.

ZELINKA I., OPLATKOVÁ Z., NOLLE L. 2004. Boolean Symmetry Function Synthesis by Means of Arbitrarry Evolutionary Algorithms - Comparative Study, *In: 18th European Simulation Multiconference (ESM 2004)*, Magdeburg, Germany, June 13-16, 2004, ISBN 3-936150-35-4.

ZELINKA I. 2004. *SOMA* - Self Organizing Migrating Algoritm, kap. 7, str. 33, in B.V. Batu, G. Onwubolu (eds), *New Optimization Techniques in Engineering*, Springer-Verlag.

ZELINKA I, OPLATKOVÁ 2006. Z. Investigation on Artificial Ant using Analytic Programming, *GECCO 2006*, Seatle, WA, USA

AUTHOR BIOGRAPHIES

PAVEL VARACHA was born in Czech Republic and went to Tomas Bata University in Zlin, where he studied technical cybernetic and obtained her degree in 2006. He is now Ph.D. student and lecturer (theory of information) at the same university. His e-mail address is: varacha@fai.utb.cz

IVAN ZELINKA was born in Czech Republic, and went to the Technical University of Brno, where he studied technical cybernetic and obtained his degree in 1995. He obtained Ph.D. degree in technical cybernetics in 2001 at Tomas Bata University in Zlin. Now he is associated professor (artificial intelligence, theory of information) and head of department. His e-mail address is: zelina@fai.utb.cz and his Web-page can be found at http://www.ft.utb.cz/people/zelinka/hp