ANN Synthesis for an Agglomeration Heating Power Consumption Approximation

PAVEL VAŘACHA, ROMAN JAŠEK
Tomas Bata University in Zlin
Faculty of Applied Informatics
nam. T.G. Masaryka 5555, 760 01 Zlin
CZECH REPUBLIC
varacha@fai.utb.cz http://www.fai.utb.cz

Abstract: This article deals with Analytic Programming (AP) which was proven to be highly effective tool of Artificial Neural Network (ANN) synthesis and optimization. AP is successfully applied here to evolve such ANN which is able to approximate heating power consumption of an agglomeration, in dependence on time and atmospherics temperature, more accurately than standard methods. An experiment described in the paper was realized on real life data from Most agglomeration and heating plant situated in Komořany, Czech Republic. Resulted ANN brings 19% increase of approximation accuracy.

Key-Words: Neural Network, approximation**,** heating plant, Analytic Programming**,** SOMA, optimization

1 Introduction

Correct approximation of heating power consumption in dependence on time and atmospheric temperature is an important presumption for heating plant successful control. Nevertheless precision of standard approximation methods (described in chapter 5) is often insufficient [1]. In these cases Artificial Neural Network (ANN) can be designed to improve approximation accuracy.

This article describes process which synthesis such ANN via symbolic regression working with real life data from Most agglomeration and heating plant situated in Komořany.

There are two well-known methods: Genetic Programming and Grammatical Evolution, which can both symbolically regress using evolutionary algorithm. However, in this article, more recent and flexible procedure called Analytic Programming (AP) is used here.

AP performed well in many separate cases (for example [2, 3]) together with different evolutionary algorithms (EA) as its "engine". Asynchronous implementation of **SOMA** – **S**elf-**O**rganizing **M**igration **A**lgorithm [4] is applied here to boost AP with the ambition to show unusual electivity of such arrangement.

SOMA is based on a self-organizing behavior of groups of individuals in a "social environment". It can also be classified as an evolutionary algorithm, despite the fact that no new generations of individuals are created during the search (due to philosophy of this algorithm). Only positions of individuals in the searched space are changed during one generation, called a "migration loop". The algorithm was published in journals, book and presented at international conferences, symposiums as well as in various invitational presentations, for example [5, 6, 7].

2 Analytic Programming

Main principle (core) of AP is based on discrete set handling (DSH) DSH shows itself as universal interface between EA and a symbolically solved problem. This is why AP can be used almost by any EA.

Briefly put, in AP, individuals consist of non-numerical expressions (operators, functions,…), which are within evolutionary process represented by their integer indexes. Such index then serves like a pointer into the set of expressions and AP uses it to synthesize resulting function-program for Cost Function evaluation.

All simple functions and operators are in so called General Function Set (GFS) divided into groups according to the number of arguments which can be inserted during the evolutionary process to create subsets GFS_{3arg}, GFS_{2arg}...GFS_{0arg}.

Functionality of discrete set handling can be seen on the concrete example in Fig. 1.

Number of actually used pointers from an individual before synthesized expression is closed is called depth.

ISBN: 978-1-61804-004-6

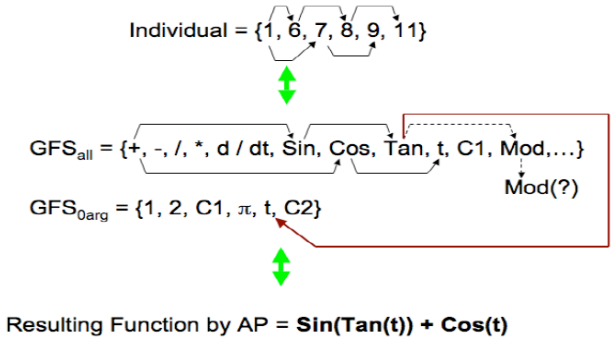

Fig 1, Main principles of AP

2.1 Constant Processing

Synthesized ANN, programs or formulas may also contain constants "K" which can be defined in GFS_0 or be a part of other functions included in GFS_{all}. When the program is synthesized, all Ks are indexed, so K_1, K_2, …, K_n, are obtained and then all K_n are estimated by second EA. In this case the EA is, again, asynchronous implementation of SOMA.

This is especially convenient for an ANN synthesis. K_n can be interpreted as various weights and thresholds and their optimization by SOMA as ANN learning.

3 SOMA All-to-One

Several different versions of SOMA exist, nevertheless, this article is focused on most common **All-to-One** version, which is most suitable for asynchronous parallel implementation. All basic All-to-One SOMA principles important for correct understanding of executed experiment are described below.

3.1 Parameter definition

Before starting the algorithm, SOMA's parameters: Step, PathLength, PopSize, PRT and Cost Function need to be defined. The Cost Function is simply the function which returns a scalar that can directly serve as a measure of fitness. In this case, Cost Function is provided by AP.

3.2 Creation of Population

Population of individuals is randomly generated. Each parameter for each individual has to be chosen randomly from the given range <Low, High>.

3.3 Migration loop

Each individual from population (PopSize) is evaluated by the Cost Function and the Leader (individual with the highest fitness) is chosen for the current migration loop. Then, all other individuals begin to jump, (according to the Step definition) towards the Leader. Each individual is evaluated after each jump by using the Cost Function. Jumping continues until a new position defined by the PathLength is reached. The new position $x_{i,j}$ after each jump is calculated by (1) as is shown graphically in Fig. 1. Later on, the individual returns to the position on its path, where it found the best fitness.

$$x_{i,j}^{MLnew} = x_{i,j,start}^{ML} + (x_{L,j}^{ML} - x_{i,j,start}^{ML})\, t\, PRTVector_j \quad (1)$$

where $t \in$ <0, *by Step to, PathLegth*>
and ML is actual migration loop

Before an individual begins jumping towards the Leader, a random number *rnd* is generated (for each individual's component), and then compared with PRT. If the generated random number is larger than PRT, then the associated component of the individual is set to 0 using PRTVector.

$$if\ rnd_j < PRT\ then\ PRTVector_j = 0\ else\ 1 \quad (2)$$

where $rnd \in$ <0, 1>
and $j = 1, \ldots n_{param}$

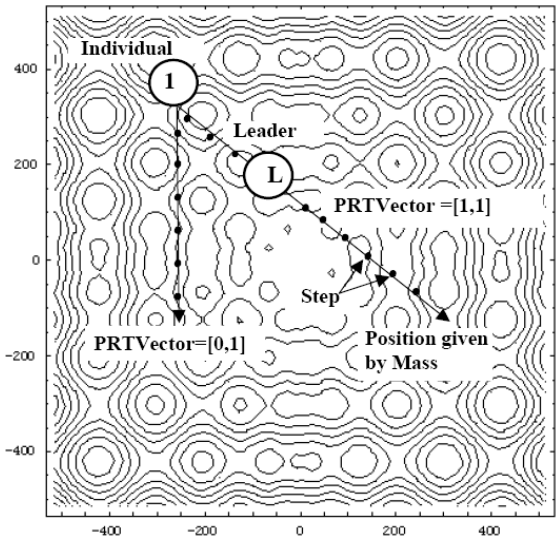

Fig. 2, PRTVector and its action on individual movement [8]

Hence, the individual moves in the N-k dimensional subspace, which is perpendicular to the original space. This fact establishes a higher robustness of the algorithm. Earlier experiments demonstrated that without the use of PRT, SOMA

tends to determine a local optimum rather than global one. [8]

4.4 Test for stopping condition

If a divergence between current Leader and Leader from the last migration loop is less than defined number, stop and recall the best solution(s) found during the search.

4 Neural Network Synthesis

There is a very easy way of using AP for ANN synthesis. [9] The most important part is to define items from which ANN will be composed. In this case GFS contains only three items.

(3)

$$GFS_{all} = \{+, AN, K*x\}$$

Most important item of (3) is Artificial Neuron (AN) (4) with hyperbolic tangent as transfer function (5). Weight of output, steepness and thresholds are computed as K in AP (6).

(4)

$$GFS1 = \{AN\}$$

(5)

$$AN = w * (e^{(2 * \lambda * (input + \phi))} - 1) / (e^{(2 * \lambda * (input + \phi))} + 1);$$

(6)

$$AN = K_1 * (e^{(2 * K_2 * (input + K_3))} - 1) / (e^{(2 * K_2 * (input + K_3))} + 1);$$

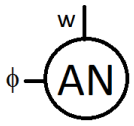

Fig. 3, Graphical example of AN

To allow more inputs into one ANN simple plus operator (7) is used.

(7)

$$GFS_2 = \{+\}$$

Fig. 4, Graphical example of plus operator

Finally, (8) represents weighted input data.

(8)

$$GFS_0 = K*x$$

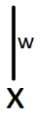

Fig. 5, Graphical example of weighted input

Under such circumstances, translation of an individual to ANN can be easily grasped from Fig. 6.

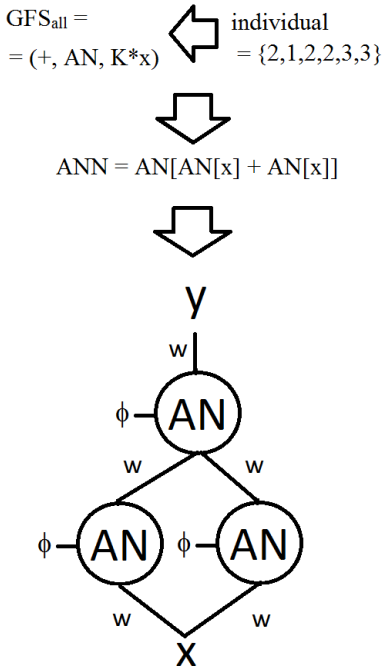

Fig. 6, Translation of an individual to ANN

Whole process is cyclical. Individuals provided by EA are translated to ANNs. ANNs are evaluated in accordance with training data set and their global errors are used to set fitness to these individuals. Consequently, a new generation is chosen and the whole process is repeated in next migration loop.

3.1 Reinforced evolution

If ANN of adequate quality cannot be obtained during AP run, AP puts the best ANN it found as a sub ANN into GFS_0 and starts over. This arrangement considerably improves AP ability to find ANN with desirable parameters.

5 Approximation of Supplied Power

The power supplied into agglomeration by the heating plant is approximated by a sum of two functions. The first function depends on time t and the second depends on external atmospheric temperature ϑ_{ex} [1]:

(9)

$$f_P(t, \vartheta_{ex}) = f_{time}(t) + f_{temp}(\vartheta_{ex})$$

To approximate division of power which depends on time Gauss curve is used:

(10)

$$f_{ETG}(t) = \frac{H^{"}}{\{1 + \lambda_L e^{k_L(t_L - t)}\}^{\left(\frac{t_L}{t}\right)^{\alpha}} + \{1 + \lambda_T e^{k_T(t - t_T)}\}^{\left(\frac{t}{t_T}\right)^{\beta}}}$$

Where:

- $H^{"}$ is a coefficient depended on top height
- λ_L is a coefficient of ascending curve dependence measure on $f_{temp}(0)$
- λ_T is a coefficient which modified ascending curve preliminary to exponential function
- k_L is an increase rate of ascending edge
- k_T is an increase and decrease rate of descending edge
- t_L is an increasing edge's inflection point time
- t_T is a decreasing edge's inflection point time
- α is a parameter which modify increasing edge shape
- β is a parameter which modify decreasing edge shape

Power dependence on actual hour is defined by a following function:

(11)

$$f_{time}(t) = \frac{H^{"}}{\{1 + \lambda_L e^{k_L(t_L - t)}\}^{\left(\frac{t_L}{t}\right)^{\alpha}}} + \frac{H^{"}}{+\{1 + \lambda_T e^{k_T(t - t_T)}\}^{\left(\frac{t}{t_T}\right)^{\beta}}}$$

As two tops occur during one period, final dependence on time is defined by a sum of two empirically transformed Gauss curves:

(12)

$$f_{time}(t) = f_{ETG_1}(t) + f_{ETG_2}(t) = \frac{H_1^{"}}{\left\{1+\lambda_{1L}e^{k_{1L}(t_{1L}-t)}\right\}^{\left(\frac{t_{1L}}{t}\right)^{\alpha_1}}} + \frac{H_1^{"}}{+\left\{1+\lambda_{1T}e^{k_{1T}(t-t_{1T})}\right\}^{\left(\frac{t}{t_{1T}}\right)^{\beta_1}}} + \frac{H_2^{"}}{\left\{1+\lambda_{2L}e^{k_{2L}(t_{2L}-t)}\right\}^{\left(\frac{t_{2L}}{t}\right)^{\alpha_2}}} + \frac{H_2^{"}}{+\left\{1+\lambda_{2T}e^{k_{2T}(t-t_{2T})}\right\}^{\left(\frac{t}{t_{2T}}\right)^{\beta_2}}}$$

To approximate division of power which depends on external atmospheric temperature generalized logistic curve is used:

(13)

$$f_{temp}(\vartheta_{ex}) = A + \frac{K - A}{(1 + Qe^{-B(\vartheta_{ex} - M)})^{\frac{1}{v}}}$$

Where:

- A is a value of lower asymptote
- K is a value of higher asymptote
- Q is a measure of dependence on $f_{temp}(0)$
- B is an increase rate of lower curve
- M is a time of maximal increase if $Q = v$
- v $v > 0$, decide in which asymptote maximal increase occur

An accuracy of described approximation was statistically evaluated on data covering a period from 11. 3. 2009 to 12. 31. 2009 which includes 1416 samples in one hour interval. Resulted NRMSD was calculated on provided data as 4.28% [1].

6 Experiment set up

Used data was normalized into <0; 1> interval and a very simple GFS structure was used for ANN synthesis during the experiment:

(14)

$$GFS_{all} = \{+, AN, K*t, K*h\}$$

If best synthetized ANN does not improve its NRMSD at least for 0,001%, breeding stopped.

(15)

$$RMSD(\theta_1, \theta_2) = \sqrt{\frac{\sum_{i=1}^{n} (ANN(x_i) - y(x_i))}{n}}$$

(16)

$$\mathrm{NRMSD} = \frac{\mathrm{RMSD}}{x_{\max} - x_{\min}}$$

Setting of Asynchronous SOMA used as EA for AP can be seen in table 1.

Number of Individuals	48
Individual Parameters	100
Low	0
High	10
PathLength	3
Step	0,11
PRT	1/ depth
Divergence	0.00001
Period	3

Table 1, Setting of SOMA used as EA for AP

Table 2 described settings of SOMA used for ANN learning:

Number of Individuals	number of K_n * 0.5 (at least 10)
Individual Parameters	100
Low	-10
High	10
PathLength	3
Step	0,11
PRT	1 / number of K_n
Divergence	0.01
Period	6

Table 2, Setting of SOMA used to optimize K_n

7 Results

In 100 cases AP was always able to synthetize ANN with NRMSD = 3.46% however final results vary in number of used AN. One example of typically obtained ANN structure is represented below:

(17)

ANN0 = **AN**[*h*] + *t*

(18)

ANN1 = ***ANN0*** + **AN**[*h*]

(19)

ANN2 = ***ANN1*** + **AN**[**AN**[*h*] + *h* + **AN[AN**[*h*]]]

(20)

ANN3 = **AN**[**AN**[*h*]] + ***ANN2***

(21)

ANN4 = ***ANN3*** + **AN**[**AN**[*t*] + **AN**[*t*] + *h*]

(22)

ANN5 = **AN**[**AN**[**AN**[*h*] + *t*]] + ***ANN4***

(23)

ANN6 = ***ANN5*** + **AN**[*h* + **AN**[*h*] + *t*]

(24)

ANN7 = **AN**[*h* + **AN**[**AN**[*t*]]] + ***ANN6***

(25)

ANN8 = **AN**[**AN**[**AN**[*t*]] + *h*] + ***ANN7***

(26)

ANN9 = **AN**[**AN**[*h*]] + ***ANN8***

(27)

ANN10 = ***ANN9*** + **AN**[*h*]

(28)

ANN11 = **AN**[*t*] **+** ***ANN10*** + **AN*[ANN10***]

(29)

ANN12 = **AN**[*h*] + ***ANN11***

(30)

ANN13 = **AN**[*h*] + ***ANN12***

(31)

ANN14 = **AN**[*h*] + ***ANN13***

(32)

ANN15 = **AN**[*h*] + ***ANN14***

(33)

ANN16 = **AN[***h* + *t*] + ***ANN15***

(34)

ANN17 = ***ANN16*** + **AN**[**AN**[*t* + **AN[***ANN16* + *h* + ***ANN16***] + *h* + **AN**[*h*] + ***ANN16*** + **AN**[*h*] + ***ANN16***]]

(35)

ANN18 = ***ANN17*** + **AN**[**AN**[*h* + *t*]]

In this cases AP used 18 sub ANN to evolve final ANN with 0.001% accuracy.

Final ANN has nontrivial structure nevertheless it can be easily simplified in case of necessity by cutting of later sub ANN with positive influence on ANN computation speed. For example (35) benefit for ANN accuracy is 0.001%.

Fig. 7 shows ANN's first five steps of evolution.

8 Conclusion

AP was able to synthetize ANN with NRMSD 3.46%. This success represents 19% improvement in comparison with standardly used methods.

Synthetized ANN can positively influenced control quality in Komořany heating plant and will be integrated into a wider expert system.

ISBN: 978-1-61804-004-6

AP proves its ability to successfully synthetized ANN within dynamic and irregular environment.

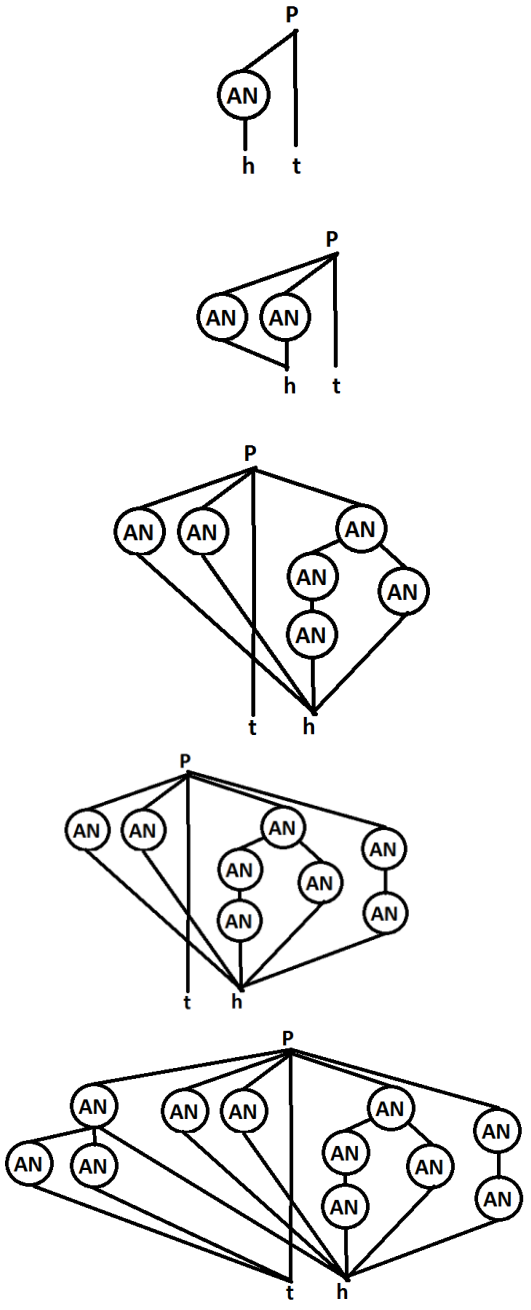

Fig. 7, ANN evolution from (17) to (21)

9 Acknowledgment

The work was performed with financial support of research project NPVII-2C06007, by the Ministry of Education of the Czech Republic.

References:

[1] E. Král, V. Dolinay, L. Vašek, P. Vařacha, Usage of PSO Algorithm for Parameters Identification of District Heating Network Simulation Model. *In 14th WSEAS International Conference on Systems. Latest Trands on Systems.*Volume II, Rhodes, WSEAS Press (GR) , 2010. p. 657-659. ISBN/ISSN: 978-960-474-214-1.

[2] Z. Oplatková, I Zelinka,. Creating evolutionary algorithms by means of analytic programming - design of new cost function. In *ECMS 2007, European Council for Modelling and Simulation*, 2007, p. 271-276. ISBN/ISSN: 978-0-9553018-2-7

[3] Z. Oplatková, I. Zelinka,. Investigation on Artificial Ant using Analytic Programming. In *Genetic and Evolutionary Computation Conference*, 2006, p. 949-950. ISBN/ISSN: 1-59593-186-4.

[4] P. Vařacha, I. Zelinka, Distributed Self-Organizing Migrating Algorithm Application and Evolutionary Scanning. In *Proceedings of the 22nd European Conference on Modelling and Simulation ECMS 2008* Nicosia, 2008. p. 201-206. ISBN/ISSN: 0-9553018-5-8.

[5] M. Červenka, I. Zelinka, Application of Evolutionary Algorithm on Aerodynamic Wing Optimisation. In *Proceedings of the 2nd European Computing Conference*, Venice, WSEAS Press (IT), 2008, ISBN/ISSN: 978-960-474-002-4.

[6] Z. Oplatková, I. Zelinka, Investigation on Shannon - Kotelnik Theorem Impact on SOMA Algorithm Performance. In *European Simulation Multiconference*, 2005, Riga, ESM , 2005. p. 66-71. ISBN/ISSN: 1-84233-112-4.

[7] R. Šenkeřík, I. Zelinka, Optimization and Evolutionary Control of Chemical Reactor. In *10th International Research/Expert Conference Trends in the Development of Machinery and Associated Technology*, TMT, Zenica, Bosna and Hercegovina, 2006, p. 1171-1174. ISBN/ISSN: 9958-617-30-7.

[8] I. Zelinka, Studies in Fuzziness and Soft Computing, New York : Springer-Verlag, 2004.

[9] P. Vařacha, Impact of Weather Inputs on Heating Plant - Agglomeration Modeling. In *Proceedings of the 10th WSEAS Ing. Conf. on Neural Networks*, Athens, WSEAS World Science and Engineering Academy and Science , 2009. p. 159-162. ISBN/ISSN: 978-960-474-065-9.

[10] B. Chramcov, Forecast of heat demand according the Box-Jenkins methodology for specific locality. In *Latest Trends on Systems*, Rhodes, WSEAS Press (GR) , 2010, p. 252-256, ISBN/ISSN: 978-960-474-199-1.