Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics. Volume I: Foundations

New Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics Volume I: Foundations

Editors

Krassimir T. Atanassov
Michał Baczyński
Józef Drewniak
Janusz Kacprzyk
Maciej Krawczak
Eulalia Szmidt
Maciej Wygralak
Sławomir Zadrożny

Editors

Krassimir T. Atanassov
Michał Baczyński
Józef Drewniak
Janusz Kacprzyk
Maciej Krawczak
Eulalia Szmidt
Maciej Wygralak
Sławomir Zadrożny

SRI PAS IBS PAN

New Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics Volume I: Foundations

Systems Research Institute
Polish Academy of Sciences

New Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics
Volume I: Foundations

Editors

Krassimir T. Atanassov
Michał Baczyński
Józef Drewniak
Janusz Kacprzyk
Maciej Krawczak
Eulalia Szmidt
Maciej Wygralak
Sławomir Zadrożny

IBS PAN **SRI PAS**

Systems Research Institute
Polish Academy of Sciences
Newelska 6, 01-447 Warsaw, Poland
www.ibspan.waw.pl

ISBN 83-894-7540-5

Dedicated to Professor Beloslav Riečan on his 75th anniversary

Generalized net model for parallel optimization of multilayer perceptron with conjugate gradient backpropagation algorithm

Sotir Sotirov[1] and Maciej Krawczak[2, 3]
[1]Prof. Asen Zlatarov University, Bourgas-8000, Bulgaria
ssotirov@btu.bg
[2] Systems Research Institute, Polish Academy of Sciences,
Newelska 6, Warsaw, Poland
[3] Warsaw School of Information Technology,
Newelska 6, Warsaw, Poland
krawczak@ibspan.waw.pl

Abstract

The used generalized net will give us a possibility for parallel optimization of multilayer perceptron based on assigned training pairs with conjugate gradient backpropagation algorithm. For changing the number of the neurons in the hidden layer we use "Golden section" rul.

Keywords: generalized nets, modelling, neural network, conjugate gradient backpropagation.

1 Introduction

In a series of papers the process of functioning and the results of the work of different types of neural networks are described by Generalized Nets (GNs, see [2], [19]). Here, we shall discuss the possibility for training of feed-forward Neural Networks (NN, see, e.g. [9]) by backpropagation algorithm. The GN will optimize the NN-structure on the based of connections of limit parameter.

The different types of NNs can be implemented in different ways [10], [15], [16] and can be learned by different algorithms [7], [13], [14].

The proposed generalized net model introduces parallel work in the training of two NNs with different structures. The difference between them is in the number of neurons in the hidden layer, which directly reflects on the all

New Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics. Volume I: Foundations *(K.T. Atanassov, M. Baczyński, J. Drewniak, J. Kacprzyk, M.Krawczak, E.Szmidt, M. Wygralak, S. Zadrożny, Eds.), IBS PAN - SRI PAS, Warsaw,2012.*

network's properties. Through increasing their number the network is learned with fewer numbers of epochs achieving its purpose. On the other hand, the great number of neurons complicates the implementation of the NN and makes it unusable in structures with elements' limits [7].

In the many-layer NNs, one layer exit become entries for the next one. The equations describing this operation are:

$$a^3=f_3(w_3f_2(w_2f_1(w_1p+b_1)+b_2)+b_3), \tag{1}$$

where:

- a^m is the exit of the m-th layer of the NN for m =1, 2, 3;
- w is a matrix of the weight coefficients of each of the entries;
- b is neuron's entry bias;
- f_m is the transfer function of the m-th layer.

The neuron in the first layer receives p outside entries.

The neurons' exits from the last layer determine the number a of NN's exits. A couple numbers is submitted (an entry value and an achieving aim – on network's exit) to the algorithm, since it belongs to the training methods with teacher:

$$<p_1, t_1>, <p_2 , t_2>, ..., <p_Q , t_Q>, \tag{2}$$

where $Q \in \{1,...,n\}$, n – numbers of learning couple, where p_Q is the entry value (on the network entry), and t_Q is the exit's value corresponding to the aim. Every network's entry is preliminary established and constant, and the exit has to correspond to the aim. The difference between the entry values and the aim is the error: $e = t-a$.

The "back propagation" algorithm [9] uses least-squarter error:

$$\hat{F} = (t-a)^2 = e^2 \tag{3}$$

In training the NN, the algorithm recalculates the network's parameters (W and b) so to achieve least-mean square error.

The "back propagation" algorithm for the i-th neuron, for k+1-th iteration uses equations:

$$w_i^m(k+1) = w_i^m(k) - \alpha\frac{\partial \hat{F}}{\partial w_i^m}; \tag{4}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\frac{\partial \hat{F}}{\partial b_i^m}, \tag{5}$$

where:

- α - learning rate for neural network;
- $\frac{\partial \hat{F}}{\partial w_i^m}$ - relation between the changes of mean square error and changes of the weights;

- $\frac{\partial \hat{F}}{\partial b_i^m}$ - relation between the changes of mean square error and changes of the biases.

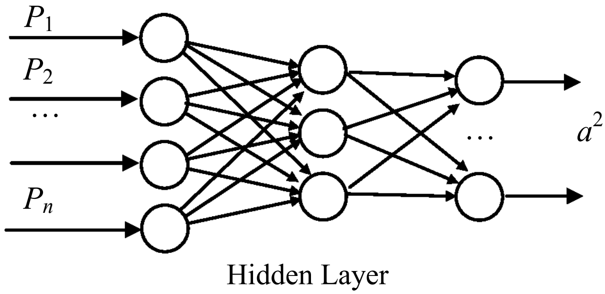


Figure 1: Multilayer Perceptron

In steepest descent algorithm, the learning rate is held constant throughout the training. The performance of the algorithm is very sensitive to the proper adjustment of the learning rate. If the learning rate is set too high, the algorithm may oscillate and become unstable. If the learning rate is too low, the algorithm will take too long to converge. It is not practical to determine the optimal setting for the learning rate before training, and, in fact, the optimal learning rate changes during the training process, as the algorithm moves across the performance surface [1], [5], [12], [17], [18].

The conjugate gradient is a numerical optimization algorithm [20], [21]. The other known methods are very different: steepest descent algorithm is a simplest algorithm, but is very slow in convergence; Newton method is much faster, but requires that the Hesian matrix and its inverse be calculated.

The conjugate gradient algorithm is something of compromise. It doesn't require the calculation of second derivatives and has the quadratic convergence property. This algorithm called conjugate gradient backpropagation (CGBP).

The CGBP algorithm are:

1. Select the firs search direction p_0 to be negative of the gradient:

$$p_0 = -g_0 \tag{6}$$

where

$$g_k = \nabla F(x)\big|_{x=x_k} \tag{7}$$

and $\nabla F(x)$ is a quadratic function.

2. Take a step selecting the learning rate α_k to minimizing the function along the search direction:

$$x_{k+1} = x_k + \alpha_k p_k \tag{8}$$

for the iteration number k+1.

3. Select the next search direction

$$p_k = -g_k + \beta_k p_{k-1} \tag{9}$$

where

$$\beta_k = \frac{\Delta g_{k-1}^T g_k}{\Delta g_{k-1}^T p_{k-1}}, \tag{10}$$

or

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}, \tag{11}$$

or

$$\beta_k = \frac{\Delta g_{k-1}^T g_k}{\Delta g_{k-1}^T g_{k-1}}, \tag{12}$$

4. If the algorithm has no converged, continue with step 2.

The network is trained when

$$e_2 < E_{max}, \tag{13}$$

where E_{max} is the maximum mean square error.

For this case study, a subject has been used as an example but there would be no essential algorithmic difference if the evaluation is related to a program form or a degree of education.

2 The golden sections algorithm

The question for the changes the number of neurons in hidden layer we propos to use the golden section algorithm.

Let the natural number N and the real number C be given. They correspond to the maximum number of the hidden neurons and the lower boundary of the desired minimal error.

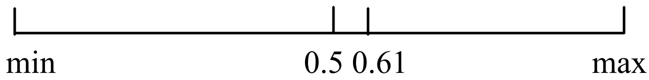
Let real monotonous function f determine the error $f(k)$ of the NN with k hidden neurons.

Let function c : $R \times R \rightarrow R$ be defined for every $x, y \in R$ by:

$$c(x,y) = \begin{cases} 0; & \text{if } \max(x;y) < C \\ \frac{1}{2}; & \text{if } x \le C \le y \\ 1; & \text{if } \min(x,y) > C \end{cases} \tag{14}$$

Let $\varphi = \frac{\sqrt{5}+1}{2} = 0.61..$ be the Golden number.

Initially, let we put: $L = 1$; $M = [\varphi^2 : N] + 1$, where $[x]$ is the integer part of the real number $x \geq 0$.



The algorithm is the following:

1. If $L \geq M$ go to 5.
2. Calculate $c(f(L), f(M))$. If

$$c(x,y) = \begin{cases} 1 \text{ to go } 3 \\ \frac{1}{2} \text{ to go } 4 \\ 0 \text{ to go } 5 \end{cases} \tag{15}$$

3. $L = M + 1$; $M = M + [\varphi^2.(N\text{-}M)] + 1$ go to 1.
4. $M = L + [\varphi^2.(N\text{-}M)] + 1$; $L = L + 1$ go to 1.
5. End: final value of the algorithm is L.

3 GN-model

All definitions related to the concept "GN" are taken from [1]. The network, describing the work of the neural network learned by "Backpropagation" algorithm [5], is shown on Fig.2.

The below constructed GN-model is reduced one. It does not have temporal components, the priorities of the transitions, places and tokens are equal, the place and arc capacities are equal to infinity.

Initially the following tokens enter in the generalized net:

- in place S_{STR} - α-token with characteristic

  x_0^α = "number of neurons in the first layer, number of neurons in the output layer";
- in place S_e – β-token with characteristic

  x_0^β = "maximum error in neural network learning Emax";
- in place S_{Pt} - γ-token with characteristic

x_0^{γ} = "{p_1, t_1}, {p_2 , t_2}, ..., {p_Q , t_Q}";

- in place S_F - one δ-token with characteristic

x_0^{δ} = "f^1,f^2,f^3".

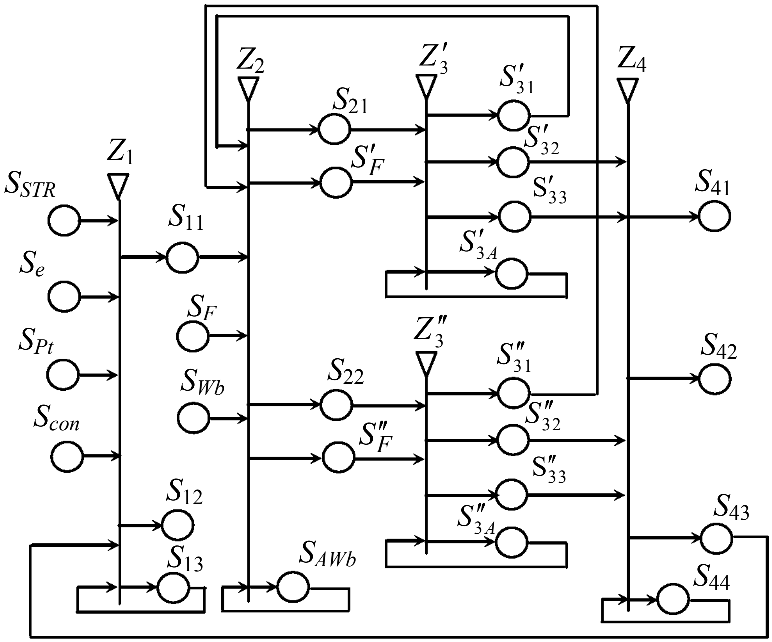


Figure 2: Generalized net model for Parallel Optimization of Multilayer Perceptron with Conjugate gradient Backpropagation Algorithm

The token splits into two tokens that enters respectively in places S'_F and S''_F; in place S_{Wb} - ε-token having characteristics x_0^{ε} = "w, b"; in place S_{con} - ξ-token with a characteristics x_0^{ξ} = "maximum number of the neurons in the hiden layer in the neural network - C_{max}".

Generalized net is presented by a set of transitions:

$$A = \{Z_1, Z_2, Z'_3, Z''_3, Z_4\},$$

where transitions describe the following processes:

- Z_1 – Forming initial conditions and structure of the neural networks;
- Z_2 – Calculating a_i using (1);
- Z'_3 – Calculating the backward of the first neural network;
- Z''_3 – Calculating the backward of the secound neural network;
- Z_4 – Checking for the end of all process.

Transitions of GN-model have the following form. Everywhere:

- p - vector of the inputs of the neural network,
- a - vector of outputs of neural network,
- a_i - output values of the i neural network, $i = 1, 2$,
- e_i – square error of the i neural network, $i = 1, 2$,
- $E_{\max}$ – maximum error in the learning of the neural network,
- t – learn target;
- w_{ik} – weight coefficients of the i neural networks $i = 1, 2$ for the k iteration;
- b_{ik} – bias coefficients of the i neural networks $i = 1, 2$ for the k iteration.

$$Z_1 = \langle \{S_{STR}, S_e, S_{Pt}, S_{\text{con}}, S_{43}, S_{13}\}, \{S_{11}, S_{12}, S_{13}\}, R_1, \wedge(\vee(\wedge(S_e, S_{Pt}, S_{\text{con}}), S_{13}), \vee(S_{STR}, S_{43}))\rangle,$$

$R_1 =$

	S_{11}	S_{12}	S_{13}
S_{STR}	*False*	*False*	*True*
S_e	*False*	*False*	*True*
S_{Pt}	*False*	*False*	*True*
S_{con}	*False*	*False*	*True*
S_{43}	*True*	*False*	*False*
S_{13}	*True*	$W_{13,12}$	*True*

,

and $W_{13,12}$= “it is not posible to devide current interval to the subintervals”.

The token that enters in place S_{11} on the first activation of the transition Z_1 obtain characteristic

$$x_0^{\theta'} = "pr_1 x_0^{\alpha}, [1; x_0^{\xi}], pr_2\, x_0^{\alpha}, x_0^{\gamma}, x_0^{\beta}".$$

Next it obtains the characteristic

$$x_{cu}^{\theta'} = "pr_1 x_0^{\alpha}, [l_{\min}; l_{\max}], pr_2\, x_0^{\alpha}, x_0^{\gamma}, x_0^{\beta}",$$

where $[l_{\min}; l_{\max}]$ is the current characteristics of the token that enters in place S_{13} from place S_{43}.

The token that enters place S_{12} obtains the characteristic $[l_{\min}; l_{\max}]$.

$$Z_2 = \langle \{ S'_{31}, S''_{31}, S_{11}, S_F, S_{Wb}, S_\gamma, S_{AWb}\}, \{S_{21}, S'_F, S_{22}, S''_F, \}, R_2, \vee(\wedge(S_F, S_{11}), \vee(S_{AWb}, S_{Wb}), (S'_{31}, S''_{31}))\rangle,$$

$R_2 =$

	S_{21}	S'_F	S_{22}	S''_F	S_{AWb}
S'_{31}	*True*	*False*	*False*	*False*	*True*
S''_{31}	*False*	*False*	*True*	*False*	*True*
S_{11}	*True*	*False*	*True*	*False*	*False*
S_F	*True*	*True*	*True*	*True*	*False*
S_{Wb}	*True*	*False*	*True*	*False*	*False*
S_{12}	*True*	*False*	*True*	*False*	*False*
S_{γ}	*True*	*False*	*True*	*False*	*False*
S_{AWb}	*True*	*False*	*True*	*False*	*False*

,

The tokens that enter places S_{21} and S_{22} obtain the characteristics respectively:

$$x_{cu}^{\eta'} = "x_{cu}^{\varepsilon'}, x_0^{\gamma}, x_0^{\beta'}, x_0^{\sigma}, a_1, pr_1\, x_0^{\alpha}, [l_{\min}], pr_2 x_0^{\alpha}"$$

and

$$x_{cu}^{\eta''} = "x_{cu}^{\varepsilon'}, x_0^{\gamma}, x_0^{\beta''}, x_0^{\sigma}, a_2, pr_1\, x_0^{\alpha}, [l_{\max}], pr_2 x_0^{\alpha}".$$

$$Z'_3 = <\{S_{21},\ S'_F,\ S'_{3A}\},\ \{S'_{31},\ S'_{32}, S'_{33},\ S'_{3A}\},\ R'_3,\ \wedge(S_{21},\ S'_F,\ S'_{3A})>,$$

$R'_3 =$

	S'_{31}	S'_{32}	S'_{33}	S'_{3A}
S_{21}	*False*	*False*	*False*	*True*
S'_F	*False*	*False*	*False*	*True*
S'_{3A}	$W'_{3A,31}$	$W'_{3A,32}$	$W'_{3A,33}$	*True*

,

and

$W'_{3A,31}$ = "$e_1 > E_{max}$";

$W'_{3A,32}$ = "$e_1 < E_{max}$";

$W'_{3A,33}$ = "$e_1 > E_{max}$ and $n_1 > m$";

where:

n_1 – current number of the first neural network learning iteration,

m – maximum number of the neural network learning iteration.

The token that enters place S'_{31} obtains the characteristic "first neural network: $w(k+1)$, $b(k+1)$". The λ'_1 and λ'_2 tokens that enter place S'_{32} and S'_{33} obtain the characteristic

$$x_0^{\lambda'_1} = x_0^{\lambda'_2} = "l_{\min}".$$

$$Z''_3 = <\{S_{22},\ S''_F,\ S''_{A3}\},\ \{S''_{31},\ S''_{32},\ S''_{33},\ S''_{A3}\},\ R''_3,\ \wedge(S_{22},\ S''_F,\ S''_{A3})>,$$

$$R''_3 = \begin{array}{c|cccc} & S''_{31} & S''_{32} & S''_{33} & S''_{A3} \\ \hline S_{22} & False & False & False & True \\ S''_{3F} & False & False & False & True \\ S''_{A3} & W''_{A3,31} & W''_{A3,32} & W''_{A3,33} & True \end{array},$$

and

$W''_{3A,31}$ = "$e_2 > E_{max}$",

$W''_{3A,32}$ = "$e_2 < E_{max}$",

$W''_{3A,33}$ = "$e_2 > E_{max}$ and $n_2 > m$",

where:

n_2 – current number of the second neural network learning iteration,

m – maximum number of the neural network learning iteration.

The token that enters place S''_{31} obtains the characteristic "second neural network: $w(k+1)$, $b(k+1)$". The λ''_1 and λ''_2 tokens that enter place S''_{32} and S''_{33} obtain respectively $x_0^{\lambda''_1} = x_0^{\lambda''_2} = "l_{max}"$.

$$Z_4 = \langle \{S'_{32}, S'_{33}, S''_{32}, S''_{33}, S_{44}\}, \{S_{41}, S_{42}, S_{43}, S_{44}\}, R_4, \wedge(S_{44} \vee (S'_{32}, S'_{33}, S''_{32}, S''_{33}))\rangle,$$

$$R_4 = \begin{array}{c|cccc} & S_{41} & S_{42} & S_{43} & S_{44} \\ \hline S'_{32} & False & False & False & True \\ S'_{33} & False & False & False & True \\ S''_{32} & False & False & False & True \\ S''_{33} & False & False & False & True \\ S_{44} & W_{44,41} & W_{44,42} & W_{44,43} & True \end{array},$$

and

$W_{44.41}$ = "$e_1 < E_{max}$" & "$e_2 < E_{max}$";

$W_{44.42}$ = "$e_1 > E_{max}$ and $n_1 > m$" & "$e_2 > E_{max}$ and $n_2 > m$";

$W_{44.43}$ = "($e_1 < E_{max}$ and ($e_2 > E_{max}$ and $n_2 > m$)) or ($e_2 < E_{max}$ and ($e_1 > E_{max}$ and $n_1 > m$))".

The token that enters place S_{41} obtains the characteristic "Both NN satisfied conditions – for the solution is used the network who wave smaller numbers of the neurons".

The token that enters place S_{42} obtain the characteristic “There is no solution (both NN not satisfied conditions)”.

The token that enters place S_{44} obtains the characteristic “The solution is in interval [l_{min}; l_{max}] – the interval is changed using the the golden sections algorithm”.

4 Conclusion

The proposed GN-model introduces the parallel work in the training of two NNs with different structures. The difference between the nets is in the number of neurons in the hidden layer and that affects directly the properties of the whole network.

On the other hand, the great number of neurons complicates the implementation of the NN. The constructed GN-model allows simulation and optimization of the architecture of the NNs using Conjugate Gradient Backpropagation algorithm.

References

[1] Kostopoulos, A. E., D.G. Sotiropoulos and T.N. Grapsa, A new efficient variable learning rate for Perry's spectral conjugate gradient training method, Proceedings of the 1st International Conference: From Scientific Computing to Computational Engineering, 2004

[2] Atanassov, K., Generalized nets, World Scientific, Singapore, New Jersey, London 1991.

[3] Atanassov, K., S. Sotirov, A. Antonov, Generalized net model for parallel optimization of feed-forward neural network, Advanced studies in contemporary Mathematics, 2007.

[4] Bellis, S., K. M. Razeeb, C. Saha, K. Delaney, C. O'Mathuna, A. Pounds-Cornish, G. de Souza, M. Colley, H. Hagras, G. Clarke, V. Callaghan, C. Argyropoulos, C. Karistianos, G. Nikiforidis, FPGA Implementation of Spiking Neural Networks - an Initial Step towards Building Tangible Collaborative Autonomous Agents, FPT’04, International Conference on Field-Programmable Technology, The University of Queensland, Brisbane, Australia, 6-8 December, 2004, 449-452.

[5] Sotiropoulos, D.G., A.E. Kostopoulos and T.N. Grapsa, A spectral version of Perry's conjugate gradient method for neural network training, In: Proceedings of 4th GRACM Congress on Computational Mechanics, 27-29 June, University of Patras, Greece, (2002)

[6] Gadea, R., F. Ballester, A. Mocholi, J. Cerda, Artificial Neural Network Implementation on a Single FPGA of a Pipelined On-Line Backpropagation, 13th International Symposium on System Synthesis (ISSS'00), 225-229.

[7] Hagan, M., H. Demuth, M. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996.

[8] Haykin, S., Neural Networks: A Comprehensive Foundation, NY: Macmillan, 1994.

[9] Neural Network Frequently Asked Questions (FAQ), The information displayed here is part of the FAQ monthly posted to comp.ai.neural-nets, 1994. http://www.fi.uib.no/Fysisk/Teori/NEURO/neurons.html

[10] Krawczak, M., Generalized Net Models of Systems, Bulletin of Polish Academy of Science, 2003

[11] Maeda, Y., Toshiki Tada, FPGA Implementation of a Pulse Density Neural Network With Training Ability Using Simultaneous Perturbation, IEEE Transactions on Neural Networks, Vol. 14, No. 3, 2003.

[12] Mandic, D., Chambers J., Towards the Optimal Learning Rate for Backpropagation. Neural Processing Letters 11(1): 1-5, 2000.

[13] Rumelhart, D., G. Hinton, and R. Williams, Training representation by back-propagation errors, Nature, Vol.323, 1986, 533-536.

[14] Sotirov, S., A method of accelerating neural network training, Neural Processing Letters, Springer Science+Business Media B.V., Formerly Kluwer Academic Publishers B.V., Volume 22, Issue 2, Oct 2005, 163 – 169.

[15] Sotirov, S., Modeling the algorithm Backpropagation for training of neural networks with generalized nets – part 1, Proceedings of the Fourth International Workshop on Generalized Nets, Sofia, 23 September 2003, 61-67.

[16] Sotirov, S., Krawczak M. Modeling the algorithm Backpropagation for training of neural networks with generalized nets – part 2, Issue on Intuitionistic Fuzzy Sets and Generalized nets, Warsaw, 2003.

[17] Vogl, T. P., J. K. Mangis, A. K. Zigler, W. T. Zink, and D. L. Alkon, Accelerating the convergence of the back-propagation method. Biological Cybernetics, vol. 59, 1988, 257-263

[18] Plagianakos, V.P., D.G. Sotiropoulos and M.N. Vrahatis, Automatic adaptation of learning rate for backpropagation neural networks In: Recent Advances in Circuits and Systems, N.E. Mastorakis ed.,pp.337-341, World Scientific Publishing Co. Pte. Ltd., (1998).

[19] Atanassov, K., On Generalized Nets Theory, “Prof. M. Drinov”Academic Publishing House, Sofia, 2007.

[20] Scales, L. E., Introduction of Non-Linear Optimization, New York: Springer-Verlag, 1985.

[21] Gill, P. E., W. Murray and M. H. Wright, Practical Optimization, New York,Academic Press, 1981.

The papers presented in this Volume 1 constitute a collection of contributions, both of a foundational and applied type, by both well-known experts and young researchers in various fields of broadly perceived intelligent systems.
It may be viewed as a result of fruitful discussions held during the Tenth International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGN-2011) organized in Warsaw on September 30, 2011 by the Systems Research Institute, Polish Academy of Sciences, in Warsaw, Poland, Institute of Biophysics and Biomedical Engineering, Bulgarian Academy of Sciences in Sofia, Bulgaria, and WIT - Warsaw School of Information Technology in Warsaw, Poland, and co-organized by: the Matej Bel University, Banska Bystrica, Slovakia, Universidad Publica de Navarra, Pamplona, Spain, Universidade de Tras-Os-Montes e Alto Douro, Vila Real, Portugal, and the University of Westminster, Harrow, UK:

Http://www.ibspan.waw.pl/ifs2011

The consecutive International Workshops on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGNs) have been meant to provide a forum for the presentation of new results and for scientific discussion on new developments in foundations and applications of intuitionistic fuzzy sets and generalized nets pioneered by Professor Krassimir T. Atanassov. Other topics related to broadly perceived representation and processing of uncertain and imprecise information and intelligent systems have also been included. The Tenth International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets (IWIFSGN-2011) is a continuation of this undertaking, and provides many new ideas and results in the areas concerned.

We hope that a collection of main contributions presented at the Workshop, completed with many papers by leading experts who have not been able to participate, will provide a source of much needed information on recent trends in the topics considered.

ISBN-13 9788389475404
ISBN 838947540-5
9 788389 475404