Int. J. Appl. Math. Comput. Sci., *2010*, *Vol. 20*, *No. 2*, *337–347*
DOI: 10.2478/v10006-010-0025-3

RULE WEIGHTS IN A NEURO–FUZZY SYSTEM WITH A HIERARCHICAL DOMAIN PARTITION

KRZYSZTOF SIMIŃSKI

Institute of Informatics
Silesian University of Technology, ul. Akademicka 16, 44–100 Gliwice, Poland
e-mail: Krzysztof.Siminski@polsl.pl

The paper discusses the problem of rule weight tuning in neuro-fuzzy systems with parameterized consequences in which rule weights and the activation of the rules are not interchangeable. Some heuristic methods of rule weight computation in neuro-fuzzy systems with a hierarchical input domain partition and parameterized consequences are proposed. Several heuristics with experimental results showing the advantage of their usage are presented.

Keywords: fuzzy inference system, hierarchical input domain partition, rule weights.

1. Introduction

Neuro-fuzzy systems are widely used due to their ability of knowledge generalisation in the fuzzy domain. They simulate very important features of the brain—the ability to generalise the acquired knowledge and the ability to cope with fuzzy and uncertain knowledge. Humans are able to express the possessed knowledge in the form of fuzzy rules. A very common approach of humans (and many animals) is the assignment of weights to the known rules. In some cases a more specialised rule may have a higher weight than general rules. Such a situation is commonly regarded as an exception to more general rules.

The question of rules in fuzzy systems has not been very widely discussed in the literature. In (Nozaki *et al.*, 1996), weights are applied to rules in a neuro-fuzzy system for classification. The tuning of rule weights is based on the Reward and Punishment (R&P) approach. After presenting a training example, the weights of the rules are modified: the weights of the successful rules are enlarged, whereas the weights of the rules elaborating false results are diminished. The idea of the rule weight was criticised in (Nauck and Kruse, 1998; Nauck, 2000). The authors show that, in Mamdani and TSK neuro-fuzzy systems, rule weights can be substituted by the modification of membership function values in rule premises. The weights applied to rule consequences lead to the obfuscation of the model—the fuzzy sets in consequences are shifted from their original location and their supports are rescaled, which may lead to the lack of interpretability of the rule base. The introduction of rule weights may also lead to negative weights, making the rule base difficult or even impossible to interpret.

The paper (Ishibuchi and Nakashima, 2001) makes profit of the remark expressed in (Nauck and Kruse, 1998) but uses it in an opposite way—instead of modifying the parameters of the membership function, the rule weights are tuned. This approach reduces the number of parameters to tune. Additionally, the domain partition may be piecewise oblique although the rules are orthogonal. The estimation of rule weights is based on rule confidence quality measures (Cordón *et al.*, 1999; Ishibuchi *et al.*, 2001).

A system for binary classification implementing rule weights is presented in (Berg *et al.*, 2002). It is based on probabilistic classification. The weight of a rule is calculated as a difference between the probabilities that the presented examples belong to the first and the second class. The paper (Ishibuchi and Yamamoto, 2005) discusses the assignment of weights upon the confidence measure of the rules. The authors do not describe the method of rule weight tuning. In (Jahromi and Taheri, 2008), the authors do not share the opinion presented in (Nauck and Kruse, 1998) and treat the correspondence between the rule weight and the membership function as an advantage pointing

(i) reduction of the rule's premise parameters from several to one,

(ii) improvement of the classification ability of the system.

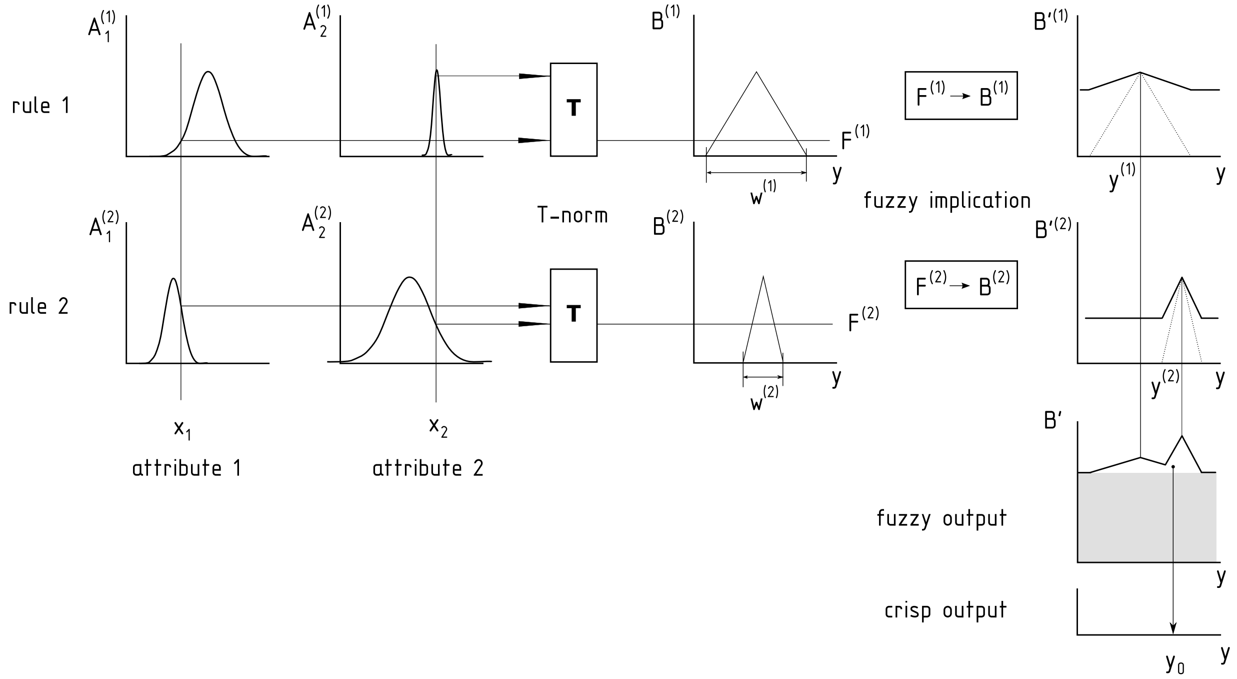

Fig. 1. Scheme of the fuzzy inference system with parametrized consequences with two rules and two-attribute objects.

The authors present a hill-climbing search method for the tuning of the rule weights.

The equivalence of rule weights and the modification of the membership function stated in (Nauck and Kruse, 1998) is not valid for a neuro-fuzzy system with parameterized consequences (Łęski and Czogała, 1997; 1999) In this system the rule weight cannot be shifted into a membership function in the rule premise. This paper discusses the rule weight problem in a neuro-fuzzy system with parameterized consequences.

The paper is organised in the following way. Section 2 describes the fuzzy system with parameterized consequences. Section 3 describes methods of rule weight assignment. Sections 4 describes the experiments and presents their results. Section 5 sums up the elaborated results.

2. Fuzzy system with parameterized consequences

The fuzzy system parameterized consequences (Łęski and Czogała, 1997; Łęski and Czogała, 1999; Czogała and Łęski, 2000) combines the Mamdani-Assilian (Mamdani and Assilian, 1975) and the Takagi-Sugeno-Kang (Takagi and Sugeno, 1985; Sugeno and Kang, 1988) approach. The fuzzy sets in consequences are isosceles triangles (as in the Mamdami-Assilian system) but are not fixed—their location is calculated as a linear combination of attribute values (as the localisation of singletons in the Takagi-Sugeno-Kang system).

The system with parameterized consequences is an MISO one. The rule base comprises fuzzy rules in the form of fuzzy implications:

$$R^{(i)} : \mathbf{x} \text{ is } \mathbf{a}^{(i)} \Rightarrow y \text{ is } b^{(i)}(\mathbf{p}, w), \tag{1}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_N]^{\mathrm{T}}$ and y are linguistic variables, $\mathbf{a}$ and b are fuzzy linguistic terms (values), $\mathbf{p}$ and w are the parameters of the consequence linguistic term. The linguistic variable a_k ($\mathbf{a}$ for the k-th attribute) is described with the Gaussian membership function:

$$\mu_{a_k}(x_k) = \exp\left(-\frac{(x_k - c_k)^2}{2s_k^2}\right), \tag{2}$$

where c_k is the core location for the k-th attribute and s_k is this attribute's Gaussian bell deviation. Each region in the domain is represented by a linguistic variable $\mathbf{a}$. The term b is represented by an isosceles triangle with the base width w, whose height is equal to one. The localisation y of the core of the triangle membership function is determined by a linear combination of input attribute values:

$$\begin{aligned} y^{(i)}(\mathbf{x}) &= \mathbf{p}^{(i)\mathrm{T}} \cdot \left[1, \mathbf{x}^{\mathrm{T}}\right]^{\mathrm{T}} \\ &= \left[p_0^{(i)}, p_1^{(i)}, \ldots, p_N^{(i)}\right] [1, x_1, \ldots, x_N]^{\mathrm{T}}. \end{aligned} \tag{3}$$

Thus the consequence of each rule has two parameters: the width w of the support and the vector $\mathbf{p}$ for localisation.

The firing strength of the i-th rule is a T-norm of memberships of input attributes:

$$F^{(i)}(\mathbf{x}) = \mu_{\mathbf{a}^{(i)}}(\mathbf{x}) = \mu_{a_1^{(i)}}(x_1) \star_T \cdots \star_T \mu_{a_N^{(i)}}(x_N), \tag{4}$$

where $\star_T$ denotes the T-norm and N stands for the number of attributes.

The fuzzy output of the system, fuzzy set B', can be expressed as an aggregation of fuzzy implications (rules):

$$\mu_{B'}(\mathbf{x}) = \bigoplus_{i=1}^{I} \left[\mu_{\mathbf{a}^{(i)}}(\mathbf{x}) \rightsquigarrow \mu_{b^{(i)}}\left(y^{(i)}(\mathbf{x}) \right) \right], \tag{5}$$

where $\bigoplus$ denotes the aggregation, the squiggle arrow ($\rightsquigarrow$) stands for fuzzy implication, i is the rule's index and I is the number of rules.

The result of aggregation, output fuzzy set B', is defuzzified in order to get the crisp output with the MICOG method, which cuts off the noninformative part of the defuzzified set (Czogała and Łęski, 2000). In (Czogała and Łęski, 2000) it was shown that the crisp output can be expressed as

$$y_0(\mathbf{x}) = \frac{\sum_{i=1}^{I} g\left(F^{(i)}(\mathbf{x}), w^{(i)} \right) y^{(i)}(\mathbf{x})}{\sum_{i=1}^{I} g\left(F^{(i)}(\mathbf{x}), w^{(i)} \right)}, \tag{6}$$

where $y^{(i)}(\mathbf{x})$ stands for the location of the core of the consequent fuzzy set (cf. Eqn. (3)), $F^{(i)}$ is the firing strength of the i-th rule, $w^{(i)}$ is the width of the base of the isosceles triangle consequence function of the i-th rule. The function g depends on the fuzzy implication; in the system the Reichenbach one is used, so for the i-th rule function g is

$$g\left(F^{(i)}(\mathbf{x}), w^{(i)} \right) = \frac{w^{(i)}}{2} F^{(i)}(\mathbf{x}). \tag{7}$$

The function g for the Łukasiewicz implication is defined by the formula (8).

Figure 1 (taken from (Łęski, 2008) and modified) shows the fuzzy inference system with parameterized consequences for two-attribute objects and two fuzzy rules. For each rule the Gaussian membership values for attributes are determined. The membership values for attributes are T-normed (in the figure it is a minimum T-norm) in order to determine the rule's firing strength. This value is the premise of the fuzzy implication, the consequent being the fuzzy triangle set. The values of the object's attributes are used to calculate the location of the triangle fuzzy set core. In Fig. 1 these are $y^{(1)}$ for the first rule and $y^{(2)}$ for the second one. The fuzzy sets representing the results of implications are aggregated (cf. Eqn. (5)) and the crisp output is determined with the MICOG procedure (cf. Eqn. (6)). The aggregated fuzzy set has two parts: informative (mountain-like shape in Fig. 1) and non-informative (gray part in the figure). The latter one is discarded as carrying no useful information.

Neuro-fuzzy systems are able to tune the parameters of the model. In this system two different methods are used for tuning. The parameters of the premises (c and s in Eqn. (2)) and the widths of the supports w of the sets in consequences are tuned with the gradient method. The linear coefficients $\mathbf{p}$ for the calculation of the location of the consequence sets are optimised with the recursive least square mean error (LSME) algorithm (Larminat and Thomas, 1983). The minimised criterion in tuning is the root mean square error (RMSE) of output system values.

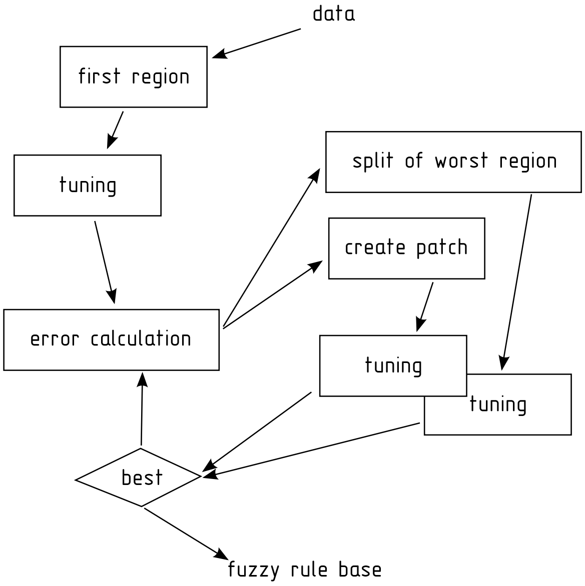

Fig. 2. PAHSID system.

2.1. Domain partition in neuro-fuzzy systems.

Neuro-fuzzy systems differ in the method of rule extraction. Two essential approaches are used: the extraction of rules starts with the construction of premises, or first the consequences are created and then the premises (Sugeno and Yasukawa, 1993). The first approach splits the input domain into regions. The splitting methods can be gathered into three classes (Almeida, 2004; Łęski, 2008):

1. grid split,
2. scatter split, and
3. hierarchical split,

depicted in Fig. 3.

The advantage of the grid split is the partition of the whole domain. There are no areas in the domain with a very low membership to all regions. A weakness is the fact

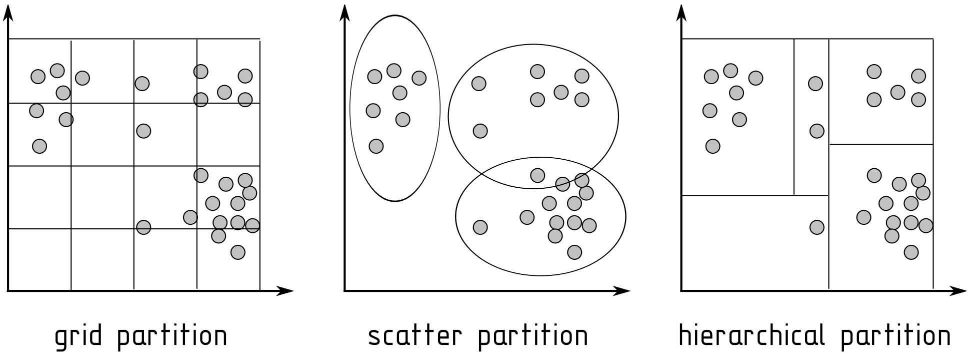

Fig. 3. Three essential ways of domain partitioning: grid, scatter and hierarchical partition.

that some region may have only few or even no examples. Thus the reduction of superfluous, redundant regions is necessary. The major drawback of the grid split is the curse of dimensionality. In this approach each attribute is split into an *a priori* established number of intervals, and thus the number of regions grows exponentially with the number of dimensions. This approach is used in the ANFIS system (Jang, 1993). The grid split is also described in (Wang and Mendel, 1992).

The scatter split (clustering) avoids the curse of dimensionality. But two problems appear: first the question of the number of clusters, then the lack of rules in some part of the input domain (the problem of areas that have very low values of membership to all clusters). The unseen cases may fall into these parts of the domain and the results elaborated by the system may be far from correct. This approach has been used in many systems, such as ANNBFIS (Czogała and Łęski, 2000; Łęski and Czogała, 1999), ANBLIR (Łęski, 2008)—the FCM clustering algorithm (Dunn, 1973); in (Abonyi *et al.*, 2002), the Gath-Geva (Gath and Geva, 1989) a clustering algorithm is applied. Chiu (1994) proposed a subtractive clustering and system based on this partition method. A similar approach was applied by (Priyono *et al.*, 2005). The SANFIS system (Wang and Lee, 2002) applies the MCA clustering algorithm.

The hierarchical approach has the advantages of both clustering and the grid split: an easier way of determining the numbers of clusters, the reduction of low membership regions and no curse of dimensionality. Some attempts have been made to apply the hierarchical domain in neuro-fuzzy systems: LOLIMOT (Nelles and Isermann, 1996; Nelles *et al.*, 2000) and binary space partitioning (Souza *et al.*,2002a; 2002b; Almeida, 2004). In these systems the regions are always split into two equal subregions. The system with a hierarchical split of the input domain, (HSID) splits hierarchically the input domain not necessarily into twin regions (Simiński, 2008a). A patch augmented hierarchically split input domain (PAHSID) neuro-fuzzy system is a developement of HSID. It is a hybrid system that comprises the hierarchical input domain partition with creating patch rules for areas with the highest error (Simiński, 2009a). The algorithm is presented in Fig. 2. The regions the input domain is split into are defined by Eqn. (2). The first action of the partitioning procedure is the creation of an initial region. The rule for this region is tuned. Then the algorithm forks into two branches: either the worst region (with the highest contribution to the error of the model) is split into two subregions or a patch rule is proposed. In both cases the new rule bases are tuned and errors are calculated. For further stages only one model is selected—that with a lower error rate. Some attempts have been made at finding an efficient and robust heuristics for deciding without tuning whether to split the worst region or to add the patch. Many approaches have been tested but none of them has resulted in any reasonable heuristics.

The stopping criterion is set true if the RSME for test data starts increasing (the knowledge generalisation ability declines).

3. Rule weights in a system with parameterized consequences

In systems with parameterized consequences the rule weights are introduced implicitly. The g function in Eqn. (6) is linearly dependent on w in all applied implications (Łęski, 2008; Nowicki, 2006). Thus the support widths w of fuzzy sets in rule consequences can be interpreted as rule weights. In systems described in the papers cited in the introductory section the output of a rule is a function of the product $w \cdot F$ (where w stands for the rule weight and F for the firing strength), which enables switching between the rule weight and its firing strength. This equivalence of rule weights and the modification of the membership function stated in (Nauck and Kruse, 1998) is not valid for a neuro-fuzzy system with parameterized consequences. In this system the rule weight cannot be shifted into a membership function in the rule premise. The formula (6) uses the function g for determining the system output. The

Table 1. Result for time series datasets. 'RMSE' stands for the 'root mean square error' and 'R' is the 'number of rules'. Various methods of rule weight calculation denoted with labels 'linear', 'square' and 'card.' for 'reciprocal cardinality'.

	gas furnace		chaotic		milk	
	RMSE	R	RMSE	R	RMSE	R
ANNBFIS	0.2031	3	0.000648	37	0.6135	10
HSID	0.2166	3	0.010149	40	0.5970	2
PAHSID	0.2079	3	0.000570	40	0.5970	2
linear	0.1782	3	0.000694	40	0.5970	2
card.	0.1759	3	0.000541	38	0.5603	3
square	0.1789	3	0.000647	40	0.6180	2
m. card.	0.1758	3	0.000667	38	0.5258	3
Fib. asc.	0.2175	3			0.5327	4
Fib. des.	0.2176	2			0.5515	3
WPAHSID	0.1992	3			0.5701	3

form of the function g depends on the fuzzy implication applied. The value of the function g is linearly dependent on w. It is non-linearly dependent on the firing strength F in the Łukasiewicz, Fodor, Kleene-Dienes and Gödel implications. For example, for the Łukasiewicz implication the function g is (Łęski, 2008; Nowicki, 2006)

$$g_L\left(F^{(i)}(\mathbf{x}), w^{(i)}\right) = w^{(i)}F^{(i)}(\mathbf{x})\left(1 - \frac{1}{2}F^{(i)}(\mathbf{x})\right), \tag{8}$$

so there is no linear transformation between the rule weight w and its firing strength F. Thus there is no danger of rescaling and dislocating fuzzy sets in the rule premises while using the rule weights. The interpretability of the model is preserved. What is more, the hierarchical domain partition enables some heuristics impossible to apply in the grid split (as in (Ishibuchi *et al.*, 2001)).

Systems with a grid partition need some rule quality measures (as confidence) to estimate the rule importance and its weight (Cordón *et al.*, 1999; Ishibuchi *et al.*, 2001). The hierarchical domain partition may lead to easier rule weight assignment. In this approach a new rule is added when needed. The idea arises to assign a higher weight to the latest rules. The oldest (first created) rule is the most general one and it is to some extent overridden by recent rules.

In ANNBFIS, ANBLIR, HSID and PAHSID, the rules weights (fuzzy set supports w) are initialised with a value of 2. The experiments show that the gradient method used to tune these values modifies the values very little and the final values of set supports w are not far from the initial values. The problem of hardly perceptible modification of rule weights in parameter tuning attracts special attention. A modification of the PAHSID system is proposed. The modification, the WPAHSID (weighted PAHSID) system, tries to modify the rule weights in order to better fit the data. The WPAHSID system splits also the input domain hierarchically and also applies patch rules when needed. Two paradigms of rule weight modifications are applied. One is trivial: the rule weights are not explicitly modified (they are modified in the gradient based procedure of parameter tuning as in PAHSID), the latter being more complicated. For each rule i tuned in the previous iteration, the output error e_i is calculated with the formula

$$e_i = \sqrt{\frac{1}{\hat{K}^{(i)}}\sum_{k=1}^{K}\left[g(\mathbf{x}_k)(y^{(i)}(\mathbf{x}_k) - y_k)\right]^2}, \tag{9}$$

where $y^{(i)}(\mathbf{x})$ is the location of the i-th rule's consequence set for the tuple $\mathbf{x}$ (cf. Eqn. (3)), y_k is the desired output, K is the number of examples in the training set, $\hat{K}^{(i)}$ is the sum of the values of the membership function of all examples for the i-th rule:

$$\hat{K}^{(i)} = \sum_{k=1}^{K}\mu_{\mathbf{a}^{(i)}}(\mathbf{x}_k). \tag{10}$$

The errors are normalised to the values $b \in [0, 1]$, so that the maximum error value is mapped into 1 and the minimum one into 0. The rule weights are then modified using the formula

$$w^{(i)} \leftarrow w^{(i)} + \eta\left(1 - b^{(i)}\right), \tag{11}$$

where $\eta \in [0, 1]$ is a modifying parameter. If the rule has a smaller error e_i (if the rule is better), then its weight is augmented most. The weight of the worst rule is not modified at all.

When the worst region is split into two subregions, they inherit the value of rule weights from the parent region. If the patch rule is applied, its weight is the maximum weight of hitherto existing rules.

In each algorithm iteration, four candidate models are created: with split, patch, split with weight modification and patch with rule weight modification. From these four models only the one with the least error is selected for further development. This makes the algorithm more time consuming in comparison with the PAHSID algorithm. Thus some heuristics are proposed. The following paradigms of assigning the weight of the new rule are only applied to new patch regions. When splitting the region is more advantageous, the weight of the split region is assigned to both new subregions. In this paper some paradigms of weight modification for PAHSID systems are presented:

1. Linear increase in rule weights. The i-th rule has the weight equal to

$$w_i = 1 + \max\{w_1, w_2, \ldots, w_{i-1}\}. \tag{12}$$

Table 2. Results of 10-fold cross-validation for the 'NewSinCos' and 'Hang' datasets.

	NeoSinCos	Hang
	RMSE	RMSE
ANNBFIS	0.008165	0.082576
HSID	0.002792	0.047177
PAHSID	0.002180	0.028677
linear	0.001028	0.031321
card.	0.000948	0.027278
square	0.001715	0.035199
mult. card.	0.000946	0.031404
Fib. asc.	0.002255	0.026794
Fib. des.	0.006393	0.044812

2. Square increase of rule weights. The i-th rule has the weight equal to

$$w_i = (\max\{w_1, w_2, \dots, w_{i-1}\})^2. \qquad (13)$$

3. Reciprocal cardinality of the rule. The fewer examples the rule recognises, the less general it is, so the weight of the rule should be high:

$$w_i = \frac{M}{\sum\limits_{m=1}^{M} F^{(i)}\left(\mathbf{X}_m\right)}, \qquad (14)$$

where F is defined in Eqn. (4).

4. Multiplied reciprocal cardinality of the rule. This is very similar to the above case, but the weight is multiplied by the number of rules:

$$w_i = \frac{iM}{\sum\limits_{m=1}^{M} F^{(i)}\left(\mathbf{X}_m\right)}. \qquad (15)$$

5. Fibonacci search (Ferguson, 1960; Knuth, 1998). This method is used for finding a minimum of the unimodal function in a given interval $[a_1, b_1]$. The precision of the method is assumed as a length of the last interval and is less than ε. The characteristic feature of this method is *a priori* estimation of the required optimising steps. The number of steps k is calculated with the formula $(b_1 - a_1)/F_k \leq \varepsilon$, where F_k is the k Fibonacci number.

   The Fibonacci search narrows the interval $[a_1, b_1]$ to one point. This is done in an iterative way. For more generality, let the interval in the i-th iteration be $[a_i, b_i]$. The iterations are numbered in descending order. Two internal values are calculated, $\beta_{i-1} = a_i + (F_{i-1}/F_i)(b_i - a_i)$ and $\alpha_{i-1} = b_i - (F_{i-1}/F_i)(b_i - a_i)$. Then the values of the optimised function f are calculated. If $f(\alpha_{i-1}) < f(\beta_{i-1})$, then the part $[\beta_{i-1}, b_i]$ of the interval is neglected for the further search, thus $a_{i-1} \leftarrow a_i$ and $b_{i-1} \leftarrow \beta_{i-1}$. Otherwise, interval $[a_i, \alpha_{i-1}]$ is no further analysed and new values are assigned: $b_{i-1} \leftarrow b_i$ and $a_{i-1} \leftarrow \alpha_{i-1}$. The algorithm stops when $i = 3$.

   The weights of the rules are optimised in two ways.

   (a) Ascending. The oldest (more general) rule is optimised first, the latest is optimised as the last one. The initial value of the weight of the first rule is assigned 2. When a new rule is added, only the weight w_i of this rule is optimised within the interval $[w_{i-1}, 2w_{i-1}]$.

   (b) Descending. The latest rule that is optimised is assigned the value $w_I = 1$. The weight of the i-th rule is optimised in the interval $[0, w_{i+1}]$ and finally the oldest rule's weight w_1 is optimised as the last one in the interval $[0, w_2]$.

Table 3. Comparision of the data approximation ability of various systems for the 'gas furnace' data set. The 'R' abbreviation stands for the number of rules.

author/system	R	RMSE
ARMA (Box and Jenkins, 1970)	–	0.8426
Tong (Tong, 1980)	19	0.6848
Lee (Lee *et al.*, 1994)	25	0.6380
Pedrycz (Pedrycz *et al.*, 1995)	25	0.6285
Xu-Lu (Xu and Lu, 1987)	25	0.5727
Pedrycz (Pedrycz, 1984)	81	0.5656
Yoshinari (Yoshinari *et al.*, 1993)	6	0.5468
Box-Jenkins (Box and Jenkins, 1970)	1	0.4494
Sugeno (Sugeno and Yasukawa, 1993)	6	0.4348
Nie (Nie, 1995)	45	0.4111
Joo (Joo *et al.*, 1997)	6	0.4074
Surmann (Surmann *et al.*, 1993)	25	0.4000
EST3 (Gómez-Skarmeta *et al.*, 1999)	2	0.3937
Oh (Oh and Pedrycz, 2000)	4	0.3507
Chen (Chen *et al.*, 1998)	3	0.2678
Lin (Lin and Cunningham, 1995)	4	0.2664
Kim-Park-Ji (Kim *et al.*, 1997)	2	0.2345
Kim-Park (Kim *et al.*, 1998)	2	0.2190
Byun (Byun *et al.*, 2001)	4	0.2141
ANBLIR (Czekalski, 2006)	2	0.1892
ANNBFIS (Simiński, 2008b)	6	0.1537
HSID (Simiński, 2009a)	6	0.1455
Rantala (Rantala and Koivisto, 2002)	5	0.1350
HSID (Simiński, 2009a)	8	0.1344
Czekalski (Czekalski, 2006)	8	0.1280
PAHSID (Simiński, 2009a)	6	0.1247
PAHSID (Simiński, 2009a)	8	0.1044
WPAHSID ($\eta = 0.1$)	8	0.1015

Table 4. Rule weights elaborated by the PAHSID system with different paradigms of rule weights assignment for the 4-th 'NewHang' set used in cross-validation.

PAHSID	linear	card.	square	mult. card.	Fib. asc.	Fib. des.
1.993	3.002	3.296	6.312	6.999	2.033	0.002
1.999	2.005	2.445	8.312	20.300	2.353	0.002
2.005	3.060	4.005	7.305	6.619	2.920	0.004
1.991	8.044	3.313	4.303	6.808	3.513	0.011
2.000	5.031	3.510	9.312	9.092	4.200	0.016
2.002	6.043	4.055	6.312	33.814	5.038	0.049
2.004	9.044	3.359	7.312	21.444	6.043	0.148
2.002	8.037	3.551	8.307	15.894	7.252	0.222
2.002	9.041	3.530	10.311	18.185	8.702	0.667
2.002	10.041	3.344	10.312	32.473	15.664	2.000

4. Experiments

The experiments were conducted on synthetic and real-life datasets. Among them the data series were also used.

4.1. Datasets. The **NewSinCos dataset** is a synthetic two input, one output dataset. One hundred points from the range $x, y \in [0, 1]$ were randomly selected. These points created the tuples $\langle x, y, z \rangle$, where

$$z = 5 + x + y + \sin 5x - \cos 5y. \tag{16}$$

All tuples were divided into 10-fold cross-validation sets.

The **Hang dataset** is described in (Sugeno and Yasukawa, 1993). It is a synthetic dataset with a two-input and one-output value calculated with the formula

$$z = \left(1 + x^{-2} + y^{-1.5}\right)^2, \tag{17}$$

where x and y are evenly distributed grid points from the interval $[1, 5]$. There were 100 $\langle x, y, z \rangle$ prepared points.

The **Chaotic time series** dataset contains the data on concentration x of leukocytes in blood described with the Mackey-Glass equation (Glass and Mackey, 1988)

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \frac{ax(t-\tau)}{1 + (x(t-\tau))^{10}} - bx(t),$$

where a, b and τ are constants. The data set containing 1000 tuples was split into training (1–500) and test (501–1000) sets.

The **Milk production** dataset (Makridakis *et al.*, 1998)[1] describes the monthly milk production per cow over 14 years. The normalised (to the zero mean and unit standard deviation) time series data were organised in the following manner: $[t-5, t-3, t-2, t-1, t, t+5]$. Of 159 tuples, the first 90 were used as a training set and following 69 as a test set (Simiński, 2009b).

Gas furnace is a real life dataset depicting the concentration of methane and carbon dioxide in the gas furnace used by (Box and Jenkins, 1970) and many researchers. It contains 290 tuples organised according to the template $[\mathbf{x}, y] = [y(n-1), \ldots, y(n-4), x(n-1), \ldots, x(n-6), y(n)]$. The tuples are divided into training (tuples 1–100) and test (tuples 101–290) sets for knowledge generalisation tests. For data approximation, all 290 tuples are used in training and test sets.

4.2. Experiments. The results of random 10-fold cross-validation for the 'NewSinCos' i 'Hang' datasets are presented in Table 2. The rule weights for various approaches for the 4-th cross-validation set in the 'Hang' data set are presented in Table 4.

The experiments on the 'chaotic time series', 'milk production' and 'gas furnace' dataset were conducted with separate training and test sets. The results are presented in Table 1. The results for 'gas furnace' represent data approximation (DA), whereas the results for 'chaotic time series' and 'milk production'—knowledge generalisation (KG).

In each experiment the number of iterations is constant and equal to 100 (the only exception with the number of iterations equal to 250 is the data approximation test, whose results are presented in the Table 3, in order to keep the same experiment conditions with other researches).

The selection of the most efficient value of the η parameter (cf. Eqn. (11)) is not easy. The best values are selected after numerous tests. Thus for knowledge generalisation for the 'gas furnace' dataset (Table 1) $\eta = 0.2$, for the 'milk' dataset $\eta = 0.6$. For data approximation (Table 3) the parameter η is 0.1.

Figure 4 presents the expected output values and values elaborated by the PAHSID system with multiplied cardinality rule weight assignment for the 'milk production' dataset. The first 90 examples constitute the training set, the following 69—the test set.

Figure 5 presents the surface elaborated by the PAHSID system with multiplied reciprocal cardinality weight assignment and the original surface defined with the formula (16) ('NewSinCos' dataset).

[1] `http://www.robjhyndman.com/forecasting/`.

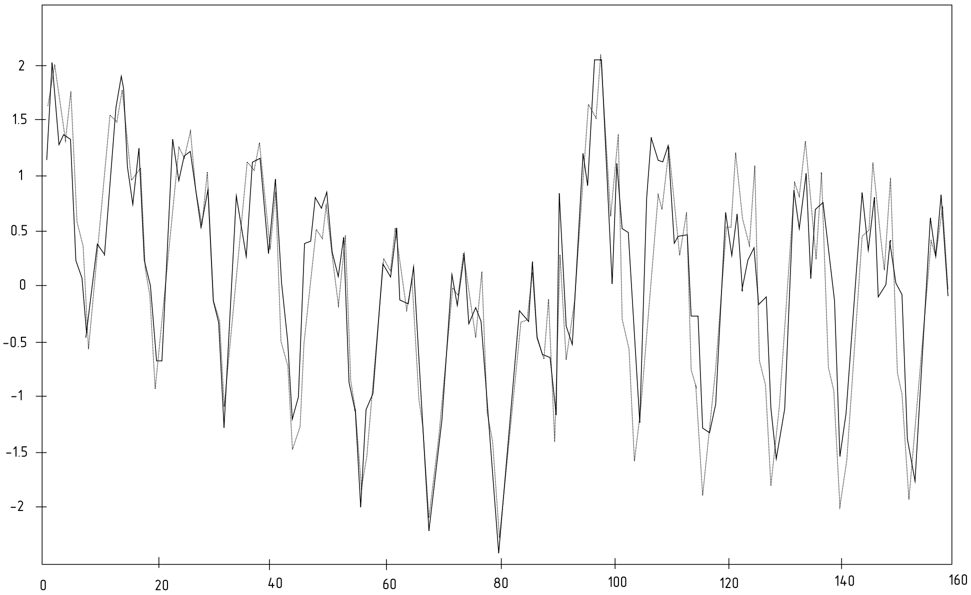

Fig. 4. Values of the output (dotted) and values elaborated by the system (solid) with multiplied cardinality rule weight assignment for training data (1–90) and test data (91–159) of the ‘milk production’ dataset.

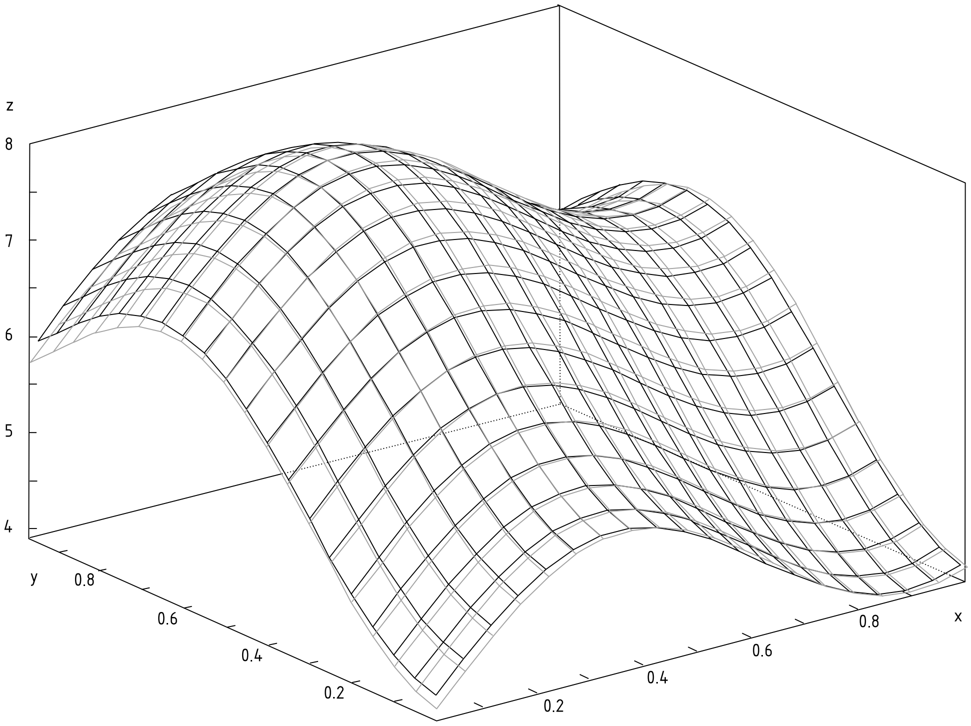

Fig. 5. Surface elaborated for the first rotation of the ‘NewSinCos’ dataset by the PAHSID system with multiplied reciprocal cardinality weight assignment—mesh in black—compared with precise surface defined with the formula (16)—mesh in gray.

5. Discussion

The weights of the rules in a neuro-fuzzy system with parameterized systems are modified very little in the standard gradient method (cf. the first column in Table 4) used in the tuning procedure. So the weights of rules are very similar. The assignment of rule weights can improve both data approximation (cf. the results for 'gas furnace' in Table 1) and knowledge generalisation abilities.

The situations observed in the experiment are as follows:

1. The WPAHSID system can produce more precise models than PAHSID and ANNBFIS systems and PAHSID with simple heuristics. This situation can be observed in Table 3.

2. WPAHSID can create a model better than PAHSID and ANNBFIS and similar in precision to systems with simple heuristics (cf. results for time-series—Table 1).

3. Finally, WPAHSID system is not able to create a better model whereas simple heuristics can produce a model of better precision (cf. results for the 'NewSinCos' and 'Hang' datasets—Table 2).

Table 3 gathers results of data approximation experiments conducted by many researchers in the 'gas furnace' data set. The WPAHSID system outperforms other systems with parameterized consequences (ANNBFIS, PAHSID, HSID). The system creates a model with the RMSE equal to 0.1015, eight rules and the weight modification parameter $\eta = 0.1$. The weights of the rules are $\mathbf{w} = [2.2788, 2.7328, 1.7778, 2.4380, 2.3776, 2.5576, 2.5341, 2.6439]^{\mathrm{T}}$. The modifications of the PAHSID system with simple heuristics achieve poorer results (e.g., for linear weights modification $\mathrm{RMSE} = 0.1219$) than WPAHSID. These heuristics modify the weight too much and the precision of the model decreases.

The cost of better models constructed by the WPAHSID system is the time needed to complete the task. The construction of the model by WPAHSID takes approximately twice as much time as simple heuristics systems (the Fibonacci search being the exception—it is the most time consuming approach).

One more problem needing a better solution is the selection of the best value of the η parameter for weight modification (cf. Eqn. (11)).

The best results in systems with simple heuristics are achieved for reciprocal cardinality. This means that the highest weight is assigned to the rule that covers the least number of cases.

Fibonacci search based rule weight optimisation requires the most calculation power of all methods researched in this paper. There are even not all results for all data sets due to extreme time demands (cf. Table 1). The results obtained using this method are usually not better than those elaborated with systems without any rule weight modification. Only for the 'NewHang' dataset can the Fibonacci search approach have good results, too. This method is suitable for searching the minimum of unimodal functions. Perhaps in this task this assumption is not true.

6. Summary

The paper presents an implementation of rule weights in neuro-fuzzy systems with a hierarchical input domain partition and parameterized consequences. In systems with parameterized consequences, no rule weight can be shifted and incorporated into membership functions in the rule premises, so there is no explicit equivalence between rule weights and the rule firing strength. In the hierarchical domain partition the rules are added when needed—this feature enables assigning higher weights to later, more specialistic rules. This approach enables adding a new feature to coping fuzziness and ability of generalisation—the exception handling.

The results of experiments on synthetic and real-life data show that this approach can create more precise models both in data approximation and knowledge generalisation.

Acknowledgment

The author is grateful to the anonymous referees for their constructive comments that helped to improve the paper.

References

Abonyi, J., Babuška, R. and Szeifert, F. (2002). Modified Gath-Geva fuzzy clustering for identification of Takagi-Sugeno fuzzy models, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* **32**(5): 612–621.

Almeida, M. R.A. (2004). *Hybrid Neuro-Fuzzy-Genetic System for Automatic Data Mining*, Pontifical Catholic University of Rio de Janeiro, (in Portugese).

Berg, J.V.D., Kaymak, U. and van den Bergh, W.-M. (2002). Fuzzy classification using probability-based rule weighting, *FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, Honolulu, HI, USA*, Vol. 2, pp. 991–996.

Box, G. E. P. and Jenkins, G. (1970). *Time Series Analysis, Forecasting and Control*, Holden-Day, Oakland, CA.

Byun, Y. B., Takama, Y. and Hirota, K. (2001). Design of a modified T-S fuzzy model by adding compensation-rules, *Journal of Japan Society for Fuzzy Theory and Systems* **13**(3): 98–109.

Chen, J.-Q., Xi, Y.-G. and Zhang, Z.-J. (1998). A clustering algorithm for fuzzy model identification, *Fuzzy Sets and Systems* **98**(3): 319–329.

Chiu, S.L. (1994). Fuzzy model identification based on cluster estimation, *Journal of Intelligent and Fuzzy Systems* **2**(3): 267–278.

Cordón, O., del Jesus, M. and Herrera, F. (1999). A proposal on reasoning methods in fuzzy rule-based classification systems, *International Journal of Approximate Reasoning* **20**(1): 21–45.

Czekalski, P. (2006). Evolution-fuzzy rule based system with parameterized consequences, *International Journal of Applied Mathematics and Computer Science* **16**(3): 373–385.

Czogała, E. and Łęski, J. (2000). *Fuzzy and Neuro-Fuzzy Intelligent Systems*, Series in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg/New York, NY.

Dunn, J.C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact, well separated clusters, *Journal of Cybernetics* **3**(3): 32–57.

Ferguson, D. E. (1960). Fibonaccian searching, *Communications ACM* **3**(12): 648.

Gath, I. and Geva, A.B. (1989). Unsupervised optimal fuzzy clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**(7): 773–780.

Glass, L. and Mackey, M.C. (1988). *From Clocks to Chaos, the Rhythms of Life*, Princeton University Press, Princeton, NJ.

Gómez-Skarmeta, A.F., Delgado, M. and Vila, M.A. (1999). About the use of fuzzy clustering techniques for fuzzy model identification, *Fuzzy Sets and Systems* **106**(2): 179–188.

Ishibuchi, H. and Nakashima, T. (2001). Effect of rule weights in fuzzy rule-based classification systems, *IEEE Transactions on Fuzzy Systems* **9**(4): 506–515.

Ishibuchi, H. and Yamamoto, T. (2005). Rule weight specification in fuzzy rule-based classification systems, *IEEE Transactions on Fuzzy Systems* **13**(4): 428–435.

Ishibuchi, H., Yamamoto, T. and Nakashima, T. (2001). Determination of rule weights of fuzzy association rules, *10th IEEE International Conference on Fuzzy Systems, Melbourne, Australia*, Vol. 3, pp. 1555–1558.

Jahromi, M.Z. and Taheri, M. (2008). A proposed method for learning rule weights in fuzzy rule-based classification systems, *Fuzzy Sets and Systems* **159**(4): 449–459.

Jang, J.-S.R. (1993). ANFIS: Adaptive-network-based fuzzy inference system, *IEEE Transactions on Systems, Man, and Cybernetics* **23**(3): 665–684.

Joo, Y.H., Hwang, H.S., Kim, K.B. and Woo, K.B. (1997). Fuzzy system modeling by fuzzy partition and GA hybrid schemes, *Fuzzy Sets and Systems* **86**(3): 279–288.

Kim, E., Park, M., Ji, S. and Park, M. (1997). A new approach to fuzzy modeling, *IEEE Transactions on Fuzzy Systems* **5**(3): 328–337.

Kim, E., Park, M., Kim, S. and Park, M. (1998). A transformed input-domain approach to fuzzy modeling, *IEEE Transactions on Fuzzy Systems* **6**(4): 596–604.

Knuth, D.E. (1998). *Art of Computer Programming, Volume 3: Sorting and Searching, 2nd Edition*, Addison-Wesley Professional, Reading, MA.

Larminat, P. and Thomas, Y. (1983). *Control Engineering—Linear Systems*, Wydawnictwa Naukowo-Techniczne, Warsaw, (in Polish).

Lee, Y.-C., Hwang, E. and Shih, Y.-P. (1994). A combined approach to fuzzy model identification, *IEEE Transactions on Systems, Man and Cybernetics* **24**(5): 736–744.

Lin, Y. and Cunningham, G.A., I. (1995). A new approach to fuzzy-neural system modeling, *IEEE Transactions on Fuzzy Systems* **3**(2): 190–198.

Łęski, J. (2008). *Neuro-Fuzzy Systems*, Wydawnictwa Naukowo-Techniczne, Warsaw, (in Polish).

Łęski, J. and Czogała, E. (1997). A new artificial neural network based fuzzy inference system with moving consequents in if-then rules and selected applications, *BUSEFAL* **71**: 72–81.

Łęski, J. and Czogała, E. (1999). A new artificial neural network based fuzzy inference system with moving consequents in if-then rules and selected applications, *Fuzzy Sets and Systems* **108**(3): 289–297.

Makridakis, S.G., Wheelwright, S.C. and Hyndman, R.J. (1998). *Forecasting: Methods and Applications, 3rd Edn.*, Wiley, New York, USA.

Mamdani, E.H. and Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller, *International Journal of Man-Machine Studies* **7**(1): 1–13.

Nauck, D. (2000). Adaptive rule weights in neuro-fuzzy systems, *Neural Computing and Applications* **9**(1): 60–70.

Nauck, D. and Kruse, R. (1998). How the learning of rule weights affects the interpretability of fuzzy systems, *Proceedings of the 1998 IEEE International Conference on Fuzzy Systems, Anchorage, AK, USA*, Vol. 2, pp. 1235–1240.

Nelles, O., Fink, A., Babuška, R. and Setnes, M. (2000). Comparison of two construction algorithms for Takagi-Sugeno fuzzy models, *International Journal of Applied Mathematics and Computer Science* **10**(4): 835–855.

Nelles, O. and Isermann, R. (1996). Basis function networks for interpolation of local linear models, *Proceedings of the 35th IEEE Conference on Decision and Control, Cobe, Japan*, Vol. 1, pp. 470–475.

Nie, J. (1995). Constructing fuzzy model by self-organizing counterpropagation network, *IEEE Transactions on Systems, Man and Cybernetics* **25**(6): 963–970.

Nowicki, R. (2006). Rough-neuro-fuzzy system with MICOG defuzzification, *2006 IEEE International Conference on Fuzzy Systems, Vancouver, Canada*, pp. 1958–1965.

Nozaki, K., Ishibuchi, H. and Tanaka, H. (1996). Adaptive fuzzy rule-based classification systems, *IEEE Transactions on Fuzzy Systems* **4**(3): 238–250.

Oh, S. and Pedrycz, W. (2000). Identification of fuzzy systems by means of an auto-tuning algorithm and its application to nonlinear systems, *Fuzzy Sets and Systems* **115**(2): 205–230.

Pedrycz, W. (1984). An identification algorithm in fuzzy relational system, *Fuzzy Sets and Systems* **13**(2): 153–167.

Pedrycz, W., Lam, P. and Rocha, A.F. (1995). Distributed fuzzy system modelling, *IEEE Transactions on System, Man and Cybernetics* **25**(5): 769–780.

Priyono, A., Ridwan, M., Alias, A.J., Atiq, R., Rahmat, K., Hassan, A. and Mohd.Ali, M.A. (2005). Generation of fuzzy rules with subtractive clustering, *Jurnal Teknologi, Series D* **43**: 143–153.

Rantala, J. and Koivisto, H. (2002). Optimised subtractive clustering for neuro-fuzzy models, *3rd WSEAS International Conference on Fuzzy Sets and Fuzzy Systems, Interlaken, Switzerland*.

Simiński, K. (2008a). Neuro-fuzzy system with hierarchical domain partition, *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA 2008), Vienna, Austria*, pp. 392–397.

Simiński, K. (2008b). Two ways of domain partition in fuzzy inference system with parametrized consequences: Clustering and hierarchical split, *OWD'2008: 10th International Ph.D. Workshop, Wisła, Poland*, pp. 103–108.

Simiński, K. (2009a). Patchwork neuro-fuzzy system with hierarchical domain partition, *in* M. Kurzyński and M. Woźniak (Eds.), *Computer Recognition Systems 3*, Advances in Intelligent and Soft Computing, Vol. 57, Springer-Verlag, Berlin/Heidelberg, pp. 11–18.

Simiński, K. (2009b). Remark on membership functions in neuro-fuzzy systems, *in* K.A. Cyran, S. Kozielski, J.F. Peters, U. Stańczyk and A. Wakulicz-Deja (Eds.), *Proceedings of the International Conference on Man-Machine Interactions ICMMI 2009*, Springer-Verlag, Berlin/Heidelberg, pp. 291–297.

Souza, F.J.D., Vellasco, M.B.R. and Pacheco, M.A.C. (2002a). Load forecasting with the hierarchical neuro-fuzzy binary space partitioning model, *International Journal of Computers, Systems and Signals* **3**(2): 118–132.

Souza, F.J.D., Vellasco, M.M.R. and Pacheco, M.A.C. (2002b). Hierarchical neuro-fuzzy quadtree models, *Fuzzy Sets and Systems* **130**(2): 189–205.

Sugeno, M. and Kang, G.T. (1988). Structure identification of fuzzy model, *Fuzzy Sets and Systems* **28**(1): 15–33.

Sugeno, M. and Yasukawa, T. (1993). A fuzzy-logic-based approach to qualitative modeling, *IEEE Transactions on Fuzzy Systems* **1**(1): 7–31.

Surmann, H., Kanstein, A. and Goser, K. (1993). Self-organizing and genetic algorithms for an automatic design of fuzzy control and decision systems, *Proceedings of the European Symposium on Intelligent Technology and Soft Computing EUFIT'93, Aachen, Germany*, pp. 1097–1104.

Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its application to modeling and control, *IEEE Transactions on Systems, Man and Cybernetics* **15**(1): 116–132.

Tong, R.M. (1980). The evaluation of fuzzy models derived from experimental data, *Fuzzy Sets and Systems* **4**(1): 1–12.

Wang, J.-S. and Lee, C.S.G. (2002). Self-adaptive neuro-fuzzy inference systems for classification applications, *IEEE Transactions on Fuzzy Systems* **10**(6): 790–802.

Wang, L.-X. and Mendel, J. (1992). Generating fuzzy rules by learning from examples, *IEEE Transactions on Systems, Man and Cybernetics* **22**(6): 1414–1427.

Xu, C. W. and Lu, Y.Z. (1987). Fuzzy model identification self-learning for dynamic system, *IEEE Transactions on Systems, Man and Cybernetics* **17**(9): 683–689.

Yoshinari, Y., Pedrycz, W. and Hirota, K. (1993). Construction of fuzzy models through clustering techniques, *Fuzzy Sets and Systems* **54**(2): 157–165.

Krzysztof Simiński received the M.Sc. and Ph.D. degrees in computer science from the Silesian University of Technology (Gliwice, Poland) in 2006 and 2009, respectively. His main interests include data mining, fuzzy reasoning, natural language processing.

Received: 21 January 2009
Revised: 27 August 2009
Re-revised: 10 October 2009