Software Cost Estimation by Fuzzy Analogy for Web Hypermedia Applications

Ali Idri[1], Azeddine Zahi[2] and Alain Abran[3]

[1] Department of Software Engineering, ENSIAS, Mohamed V University, Rabat, Morocco
E-mail: idri@ensias.ma
[2] Department of Computer Science FST, Sidi Mohamed Ben Abdellah University, Fez, Morocco
E-mail: azahi@fst-usmba.ac.ma
[3] École de Technologie Supérieure, 1180 Notre-Dame Ouest, Montreal, Canada H3C 1K3
E-mail: aabran@ele.etsmtl.ca

Abstract. The aim of this paper is to evaluate the accuracy of Fuzzy Analogy for software cost estimation on a Web software dataset. Fuzzy Analogy is based on reasoning by analogy and fuzzy logic to estimate effort when software projects are described by linguistic values such as *low* and *high*. Linguistic values are represented in the Fuzzy Analogy estimation process with fuzzy sets. However, the descriptions given of the Web software attributes used are insufficient to empirically build their fuzzy representations. Hence, we have suggested the use of the Fuzzy C-Means clustering technique (FCM) and a Real Coded Genetic Algorithm (RCGA) to build these fuzzy representations.

1 Introduction

Software cost estimation has been the subject of intensive investigation in the field of software engineering. As a result, numerous software cost estimation techniques have been proposed and investigated. Software cost estimation by analogy is one of the most attractive techniques and is essentially a form of Case-Based Reasoning [13] [12] [7]. It is based on the following assumption: *similar software projects have similar costs*, and it has been deployed as follows. First, each project is described by a set of attributes that must be relevant and independent. Second, we determine the similarity between the candidate project and each project in the historical database. In the third step, known as case adaptation, the known effort values from the most similar historical projects are used to derive an estimate for the new project. There are two main advantages of analogy-based estimation: first, its process is easy to understand and explain to users; and, second, it can model a complex set of relationships between the dependent variable (such as cost or effort) and the independent variables (cost drivers). However, its deployment in software cost estimation still warrants some improvements. Hence, we have developed a new approach referred to as Fuzzy Analogy based on reasoning by analogy and fuzzy logic to estimate effort when software projects are described by linguistic values [7], which is a major limitation of all estimation techniques (categorical data: nominal or ordinal

scale) such as ‘very low’, ‘low’ and ‘high’. Indeed, handling imprecision, uncertainty and partial truth is unavoidable when using these values. As a consequence, Fuzzy Analogy suggests the use of fuzzy sets rather than classical intervals or numbers (as in the classical procedure of cost estimation by analogy) to represent linguistic values. The main motivation behind the theory of fuzzy sets, founded by Zadeh in 1965, is the desire to build a formal quantitative framework that captures the vagueness of human knowledge, since it is expressed via natural language [15].

In an earlier work, we validated Fuzzy Analogy on the COCOMO’81 dataset that contains 63 historical software projects [3]; each of which is described by 12 attributes measured on an ordinal scale composed of six linguistic values: ‘very low’, ‘low’, ‘nominal’, ‘high’, ‘very high’ and ‘extra high’ [7]. The fuzzy representation (fuzzy sets and their membership functions) of the 12 COCOMO’81 cost drivers has been empirically achieved based on their descriptions [5]. The accuracy of Fuzzy Analogy is compared with that of three other models: classical analogy, Intermediate COCOMO’81 and fuzzy Intermediate COCOMO’81. Fuzzy Analogy performs better in terms of accuracy (MMRE=21) and in its adequacy in dealing with linguistic values [7].

The aim of this work is to validate Fuzzy Analogy on a dataset containing 54 Web hypermedia applications [10]. Each application is described by 9 numerical attributes, such as the number of html or shtml files used, the number of media files and team experience (Table 1). Initially, this dataset contains more than 9 software attributes, but some of them may be grouped together. For example, we have grouped together the following three attributes: number of new Web pages developed by the team, number of Web pages provided by the customer and the number of Web pages developed by a third party (outsourced) in one attribute reflecting the number of Web pages in the application (Webpages).

Table 1. Software attributes for the Web dataset

Software attribute	Description
Teamexp	Average number of years’ experience the team has in web development
Devteam	Number of people who have worked on the software project
Webpages	Number of Web pages in the software
TextP	Number of text pages in the software (600 words to a text page)
Imag	Number of images in the software
Anim	Number of animations in the software
Audio/video	Number of audio/video files
Tot-high	Number of high-effort features
Tot-nhigh	Number of low-effort features

The validation of Fuzzy Analogy on the Web dataset requires the determination of the fuzzy sets, and their membership functions, associated with the 9 Web software attributes. However, the descriptions given of these attributes are, unlike the case of COCOMO’81, insufficient to empirically build their fuzzy representations. Hence, we have suggested the use of the Fuzzy C-Means clustering technique (FCM) and a Real Coded Genetic Algorithm (RCGA) to build fuzzy representations for software attributes [8]. So, we apply the FCM-RCGA process to the 9 Web software attributes.

2 Fuzzy Analogy: An Overview

Fuzzy Analogy is a 'fuzzification' of the classical analogy procedure. It is also composed of three steps: identification of cases, retrieval of similar cases and case adaptation [7]:

▪ *Identification of a Case***:** The goal of this step is the characterization of all software projects by a set of attributes. Each software project is described by a set of selected attributes that are measured by linguistic values. Let us assume that we have M attributes, and, for each attribute V_j, a measure with linguistic values is defined (A_k^j). Each linguistic value A_k^j is represented by a fuzzy set with a membership function ($\mu_{A_k^j}$). The fuzzy sets and their membership functions are defined by using: 1) empirical techniques which construct membership functions from expert knowledge; or 2) automatic techniques, which construct membership functions from historical data using clustering techniques.

▪ *Retrieval of Similar Cases*: This step is based on the choice of a software project similarity measure. This choice is very important, since it will influence which analogies are found. We have proposed a set of candidate measures for software project similarity [6]. These measures evaluate the overall similarity $d(P_1, P_2)$ of two projects P_1 and P_2, by combining the individual similarities of P_1 and P_2 associated with the various attributes V_j describing P_1 and P_2, $d_{v_j}(P_1, P_2)$.

$$d(P_1,P_2)=\begin{cases} all\ of\ (d_{v_j}(P_1,P_2)) \\ most\ of\ (d_{v_j}(P_1,P_2)) \\ many\ of\ (d_{v_j}(P_1,P_2)) \\ .... \\ there\ exists\ of\ (d_{v_j}(P_1,P_2)) \end{cases} \quad \textbf{(1)}$$

To evaluate the overall distance of P_1 and P_2, the individual distances $d_{v_j}(P_1, P_2)$ are aggregated using Regular Increasing Monotone (RIM) linguistic quantifiers such as 'all', 'most', 'many', 'at most α', or 'there exists'. The choice of the appropriate RIM linguistic quantifier, Q, depends on the characteristics and needs of each environment. Q indicates the proportion of individual distances that we feel is necessary for a good evaluation of the overall distance.

▪ *Case Adaptation:* The objective of this step is to derive an estimate for the new project by using the known effort values of similar projects. In this step, two issues must be addressed. First, the choice of how many similar projects should be used in the adaptation, and, second, how to adapt the chosen analogies to generate an estimate for the new project. In Fuzzy Analogy, we have proposed a new strategy for selecting projects to be used in the adaptation step. This strategy is based on the distances $d(P, P_i)$ and the definition adopted in the studied environment for the proposition,

‘ P_i is a closely similar project to P.’ For the adaptation formula, the weighted mean of all known effort projects in the data set is used.

3 Building fuzzy sets and their membership functions for the Web software attributes

The use of Fuzzy Analogy to estimate software development effort requires determination of the fuzzy sets, and their membership functions, of the attributes describing software projects. Because the descriptions given of the 9 Web software attributes are insufficient to empirically build their fuzzy representations, we suggest the use of the Fuzzy C-Means clustering technique (FCM) and a Real Coded Genetic Algorithm (RCGA) to build the fuzzy representations of the Web software attributes.
The proposed FCM-RCGA fuzzy set generation process consists of two main steps (Figure 1). First, we use the well-known FCM algorithm and the Xie-Beni validity criterion to decide on the number of clusters (fuzzy sets) [2] [14]. Second, we use an RCGA to build membership functions for these fuzzy sets [4] [11]. Membership functions can be trapezoidal, triangular or Gaussian.
The FCM algorithm is a fuzzy clustering method used to generate a known number of clusters (c) from a set of numerical data $X = \{x_1,...,x_n\}$. The determination of this number is still an open problem in clustering. Often, empirical knowledge or a set of evaluation criteria is used to choose the best set of clusters. In this work, we use the fuzzy cluster validity criterion proposed in [14]. FCM is an iterative algorithm that aims to find cluster centers $(C_j), 1 \le j \le c$ and the matrix $U = (u_{ij}), 1 \le i \le n, 1 \le j \le c$ that minimizes the following objective function:

$$Min\ J_p(U,C) = \sum_{i=1}^{i=n}\sum_{j=1}^{j=c}(u_{ij})^m \left\|x_i - c_j\right\|^2 \text{ subject to } \sum_{i=1}^{c} u_{ij} = 1, \forall j = 1,...,n \qquad (2)$$

where m is the control parameter of fuzziness; $U = (u_{ij})$ is the partition matrix, containing the membership values of all data in all clusters;
After generating fuzzy sets (clusters $(C_j), 1 \le j \le c$) with their partition $U = (u_{ij})$ by means of FCM, we use an RCGA to build membership functions for these clusters [4] [11]; membership functions can be trapezoidal, triangular or Gaussian. Our RCGA consists in building a set of membership functions $(\mu_j), 1 \le j \le c$ that interpolates and minimizes the mean square error, which is defined as follows:

$$MSE(\mu_1,..,\mu_c) = \frac{1}{n}\sum_{j=1}^{j=n}\left\|(\mu_1(x_j),..,\mu_c(x_j)) - (u_{1j},..,u_{cj})\right\|^2 \qquad (3)$$

subject to $\sum_{j=1}^{c} \mu_j(x_i) = 1$, for all x_i and $\mu_j(x_i) = u_{ij}$, $1 \le i \le n$; $1 \le j \le c$

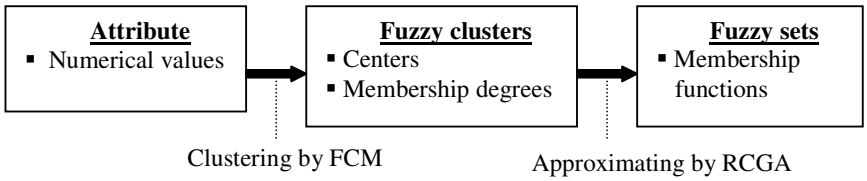

Fig. 1. Fuzzy set generation process

The use of an RCGA to find membership functions μ_j requires the determination of certain parameters, such as the coding scheme, the fitness function and the various genetic operators (selection, crossover and mutation). Concerning the coding scheme, a chromosome in the population of our RCGA, $m_i, 1 \le i \le M$, represents the set of the unknown membership functions, $(\mu_j), 1 \le j \le c$, associated with the c fuzzy sets generated by the FCM. The shape of the membership functions can be trapezoidal, triangular or Gaussian. Thus, each chromosome encodes a set of membership functions in a real vector $(m_i^1, ..., m_i^K)$. The genes m_i^j are obtained from the shape of the membership functions. Figure 2 shows the structure of a chromosome, m_i, encoding trapezoidal membership functions. The fitness function F is obtained using the following formula:

$$F(m_i) = \frac{MSE(m_i)}{\sum_{i=1}^{i=M} MSE(m_i)} ; MSE(m_i) = \frac{1}{n} \sum_{j=1}^{j=n} \left\| \mu(x_j) - y_j \right\|^2 \quad (4)$$

where $\mu(x_j) = (\mu_1(x_j), ..., \mu_c(x_j))$, $y_j = (u_{j1}, ..., u_{jc})$ and M is the size of the population. For the three genetic operators (selection, crossover and mutation), we use those that are specific to RCGAs. Hence, the linear ranking is used as a selection operator [1]. The line recombination method is considered as a crossover operator [11]. The Breeder Genetic Algorithm is used as a mutation operator [11].

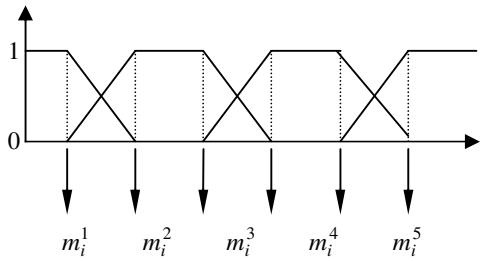

Fig. 2. Structure of a chromosome associated with trapezoidal membership functions

4 Empirical Results

This section presents and discusses the results obtained when applying Fuzzy analogy to the Web dataset. The calculations were made using a software prototype developed with Matlab 7.0 under a Microsoft Windows PC environment. The accuracy of the estimates is evaluated by using the magnitude of relative error, MRE, defined as:

$$MRE = \left| \frac{Effort_{actual} - Effort_{estimated}}{Effort_{actual}} \right| \quad (5)$$

The MRE is calculated for each project in the dataset. In addition, we use the measure prediction level *Pred*. This measure is often used in the literature. It is defined by:

$$\Pr ed(p) = \frac{k}{N} \quad (6)$$

where N is the total number of observations, k is the number of observations with an MRE less than or equal to p. In this evaluation, we use p equal to 0.20. The *Pred*(0.20) gives the percentage of projects that were predicted with an MRE less than or equal to 0.20.

Normally, for the overall distances, each environment must define its appropriate quantifier (Q) by studying its features and its requirements. Because a lack of knowledge concerning the appropriate quantifier for the environment from which Web dataset was collected, we used various α-RIM linguistic quantifiers to combine the individual similarities. An α-RIM linguistic quantifier is defined by a fuzzy set in the unit interval with membership function Q given by:

$$Q(r) = r^{\alpha} \quad \alpha > 0 \quad (7)$$

For each Web software attribute, several experiments were conducted with the FCM algorithm, each time using a different initial matrix U. The desired number of clusters (c) is varied within the interval [2,7]. The parameter m is fixed to 2 in all experiments. As mentioned earlier, we use the Xie-Beni criterion to decide on the number of clusters. For each attribute, we choose the number of clusters that minimizes the value of the Xie-Beni criterion. Table 2 shows the 'best' classification obtained according to the Xie-Beni index.

Table 2: Example of a classification generated by the FCM algorithm.

Attributes	#fuzzy sets	Attributes	#fuzzy sets	Attributes	#fuzzy sets
DevTeam	7	TEXTP	3	Audio	4
Teamexp	7	IMAG	3	Tot-high	5
Webpages	2	ANIM	3	Tot-nhigh	5

After generating the fuzzy sets with the FCM Algorithm, we applied the RCGA algorithm, as designed in the previous section, to these fuzzy clusters to build their membership functions. This algorithm is applied with populations of up to 300, the mutation probability fixed to 0.9, and the number of generations is equal to 200. Figure 3 shows three different shapes of membership functions associated with the fuzzy sets of the IMAG and TEXTP attributes respectively.

By analyzing the results of the validation of the Fuzzy Analogy technique (Table 3 and Figure 4), we noted that the accuracy of the estimates depends on the linguistic quantifier (α) used in the evaluation of the overall similarity between software projects. So, if we consider the accuracy measured by *Pred*(0.20) as a function of α, we can say that, in general, it increases monotonously according to α. This is because our similarity measures decrease monotonously according to α. Indeed, when α tends towards zero, this implies that the overall similarity will take into account fewer attributes among all those describing Web software projects. The minimum number of attributes to consider is one. As a consequence, the overall similarity will be higher because we are more likely to find at least one attribute in the Web dataset for which the associated linguistic values are the same for the two projects. By contrast, when α tends towards infinity, it implies that the overall similarity will take into account many attributes among all the available ones describing the software projects. As a maximum, we may consider all attributes. Consequently, the overall similarity will be minor because we are more likely to find one attribute in the Web dataset for which the associated linguistic values are different for the two projects.

We compared the accuracy of the Fuzzy Analogy when using different shapes of the membership functions. Our findings were the following: Fuzzy Analogy performs better when trapezoidal membership functions are used than when triangular membership functions are used. In addition, it performs better when triangular membership functions are used than when Gaussian membership functions are used. However, the accuracy of Fuzzy Analogy is higher for the three shapes of membership functions than that of threshold precision, which is often used in the software cost estimation literature (Pred(20)≥70).

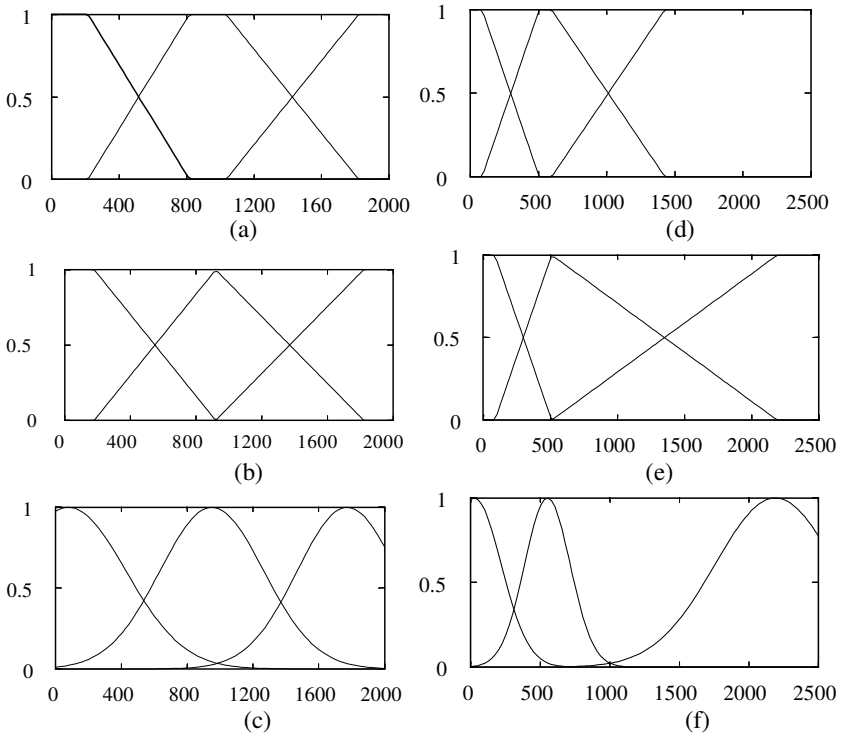

Fig. 3. Membership functions associated with the fuzzy sets of the IMAG and TEXTP attributes respectively: (a) Trapezoidal for IMAG; (b) Triangular for IMAG; (c) Gaussian for IMAG; (e) Trapezoidal for TEXTP; (e) Triangular for TEXTP; and (f) Gaussian for TEXTP.

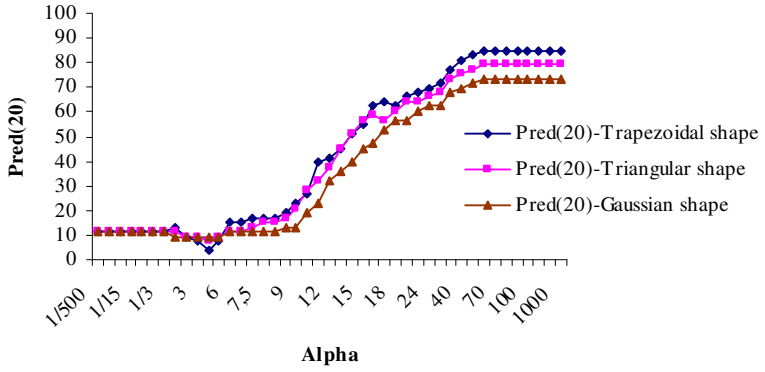

Fig. 4. Relationship between α and the accuracy of Fuzzy Analogy for the three shapes of membership functions.

Table 3. Results of the evaluation of Fuzzy Analogy

	Trapezoidal functions		Trinagulair functions		Gaussian functions	
alpha-RIM	Pred(0.20)	MMRE	Pred(0.20)	MMRE	Pred(0.20)	MMRE
1/10	11.32	717.78	11.32	718.90	11.32	719.54
1/7	11.32	711.12	11.32	712.67	11.32	713.53
1/3	11.32	683.10	11.32	686.37	11.32	688.12
1	13.21	600.72	11.32	607.91	9.43	612.33
3	7.55	434.93	9.43	445.37	9.43	455.48
7	16.98	250.73	13.21	260.14	11.32	272.85
10	26.42	177.59	28.30	184.28	18.87	195.68
15	54.72	119.31	56.60	122.88	45.28	131.48
25	71.70	78.36	67.92	81.12	62.26	87.93
30	77.36	69.91	73.58	73.17	67.92	79.75
40	81.13	62.19	75.47	65.75	69.81	72.40
50	83.02	59.71	77.36	63.30	71.70	70.17
60	84.91	58.92	79.25	62.54	73.58	69.46
70	84.91	58.68	79.25	62.30	73.58	69.28
80	84.91	58.60	79.25	62.23	73.58	69.22

5 Conclusion and future work

In this paper, we have validated the Fuzzy Analogy approach for estimating the cost of Web hypermedia applications. The use of Fuzzy Analogy requires the determination of the fuzzy sets, and their membership functions, of the attributes describing these applications. Because the descriptions given of the Web software attributes are insufficient to empirically build their fuzzy representations, we have suggested the use of the Fuzzy C-Means clustering technique (FCM) and a Real Coded Genetic Algorithm (RCGA) (the FCM-RCGA process) to build these representations for the Web software attributes. The membership functions generated may be trapezoidal, triangular or Gaussian. The results of this validation show that Fuzzy Analogy generates accurate estimates, using trapezoidal, triangular or Gaussian fuzzy representation. It should be noted that our findings favor the use of trapezoidal representation.

We are currently looking at comparing Fuzzy Analogy and classical analogy on the Web dataset, as we have already compared Fuzzy Analogy with three other models (Intermediate COCOMO'81, Fuzzy Intermediate COCOMO'81 and classical analogy) on the COCOMO'81 dataset. We have found that Fuzzy Analogy performs better in terms of accuracy and its adequacy in dealing with linguistic values. Another interesting avenue of research would be to look at the accuracy of Fuzzy Analogy when using the FCM-RCGA process rather than empirical knowledge for building fuzzy sets.

Acknowledgments

We thank Dr. Emilia Mendes for providing us with the data on the 54 Web software projects.

5 Bibliography

1. Baker, J. E. Reducing bias and inefficiency in the selection algorithm. Lawrence Erlbaum Associates, Publishers, ProcICGA 2 (1987), 14-21.
2. Bezdek, J., Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York, (1981).
3. Boehm, B. W., Software Engineering Economics. Prentice-Hall, (1981).
4. Herrera, F., Loazano, M., Sanchez, A. M., A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. Int. J. Intell. Syst. 18(3), (2003) 309-338.
5. Idri, A., Kjiri, L., Abran, A., COCOMO Cost Model Using Fuzzy Logic. Proceedings of the 7th International Conference on Fuzzy Set Theory and Technology, NJ, (2000) 219-223.
6. Idri, A., Abran, A., A Fuzzy Logic-faced Measure for Software Project Similarity: Validation and Possible Improvements. 7th IEEE International Symposium on Software , 4-6 April, London, (2001) 85-96.
7. Idri, A., Abran, A., Khoshgoftaar, T., Estimating Software Project Effort by Analogy Based on Linguistic Values. 8th IEEE International Software Metrics Symposium, Ottawa, June (2002) 21-30.
8. Idri, A., Zahi, A., Abran, A., Generating Fuzzy Term Sets for Software Project Attributes using Fuzzy C-Means and Real Coded Genetic Algorithms, to be published at ICT4M, Malaysia, November 2006.
9. Medasani, S., Kim, J., Krishnapuram, R., An overview of membership function generation techniques for pattern recognition. Internat. J. Approx. Reas., 19 (1998) 391-417.
10. Mendes, E., Triggs, W. C., Mosley, N., Counsell, S., A comparison of Development Effort Estimation Techniques for Web Hypermedia Applications. 8th IEEE International Software Metrics Symposium, Ottawa (2002), 131-140.
11 Mühlenbein, H., Schlierkamp-Voosen, D. Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization," *Evolutionary Computation*, Vol. 1, No. 1 (1993), 25-49.
12 Shepperd, M., Schofield, C., Estimating Software Project Effort Using Analogies. IEEE Transactions on Software Engineering, vol. 23, no. 12 (1997), 736-747.
13. Vicinanza, S., Prietolla, M. J., Case-Based Reasoning in Software Effort Estimation. Proceedings of the 11th Int. Conf. on Information Systems (1990).
14. Xie, X. L., Beni, G., "A validity measure for fuzzy clustering." IEEE Transactions on Pattern Analysis Machine Intelligence, vol. 13, no 8 (1991), 841-847.
15. Zadeh, L. A., Fuzzy sets. Information and Control, vol. 8 (1965), 338-352.