25–29 July, 2004 • Budapest, Hungary

Painting Algorithms for Fuzzy Classification

Daniel Gómez, Javier Montero, Javier Yáñez
Department of Statistics and Operational Research
Complutense University of Madrid, Spain
E-mail: dagomez@estad.ucm.es

Carmelo Poidomani
Department of Mathematics and Computer Science
University of Catania
Catania, Italy

Abstract— **Land cover analysis by means of remotely sensing images quite often suggest the existence of fuzzy classes, where no clear borders or particular shapes appear. In this paper we present an image classification aid algorithm which shows as its main output a processed image where each pixel is being colored according to the degree of similitude to their respective surrounding pixels. Such a processed image is therefore suggesting possible classes, to be implemented in a more sophisticated image classification process. A key underlying argument for this approach is the relevance of painting techniques in order to help decision makers to understand complex information relative to fuzzy image classification.**

I. Introduction

Classical crisp image classification can be assimilated to a search for *objects*. These objects have clear boundaries, and a sudden change is expected in their *borders*. Moreover, we frequently have some *a priori* information about their *shape*. This is not the case in many remotely sensing classification problems (see, e.g., Jensen [11]), where the images quite often suggest fuzzy classes. When dealing with land cover problems, for example, most *natural* classes do not show neither clear borders or particular shape. Classes show gradation between adjacent pixels, with no pattern in their shape (see, e.g., Bezdek [6] and Bezdek-Harris [7]). In fact, applications of Fuzzy Sets Theory to remote sensing classification problems is becoming an active research field (see, e.g., Foody [8], Kerre-Natchtegael [12] and Pal *et al.* [16]).

Still, many different *fuzzy* approaches have been tried in remote sensing. For example, Amo *et al.* [2], [5] considered a fuzzy classification model based upon an outranking model developed in Pearman *et al.* [14] (see also Perny-Roy [18] and Siskos [21]). When applied to a particular remote sensing image, Amo *et al.* [2], [5] produced as an output an informative picture showing the existence of gradation between different regions, each one being associated to one color (see Amo *et al.* [5]).

In general, we should not be expecting that a regular decision maker will be able to manage fuzzy classes if they are shown just by means of a bunch of numbers. These numbers need to be *organized* in order to be managed, perhaps by means of an unifying formula. But even simple mathematical expressions are difficult to be managed by decision makers. For example, it is not so easy to define and understand a Ruspini's Fuzzy Partition (see Ruspini [20]). Developing fuzzy representation techniques is an absolute need so decision makers can get a good enough understanding of fuzzy information. Appropriate strategies for their decision making should always be based upon such an understanding.

In particular, in this paper we present an unsupervised fuzzy coloring algorithm, described as the successive application of a basic binary procedure. A hierarchical structure of colors is then produced, and each pixel is assimilated to a place within such a structure of colors. A family of connected pixels will be suggesting a class meanwhile color does not change and such a class is considered *homogeneous* enough. Improvements leading to more sophisticated algorithms based upon this technique can be also tried, always pursuing some automatic representation of images suggesting homogeneous regions and transition zones. Main arguments about fuzzy classification systems, as considered in Amo *et al.* [3], [4], should be also taken into account in a later supervised analysis, where additional information may exist.

Our main objective in this paper is to produce an unsupervised fuzzy coloring algorithm, so a summarizing picture can be offered to decision makers in order to identify possible classes. A main argument, at least within land cover problems (see [1], [2], [5]), is that each fuzzy class has a core of connected pixels, and surrounding pixels may show some degree of similitude to that core. Hence, classification is made pixel by pixel, but behavior of surrounding pixels has a strong influence at every stage.

II. The Image

The image under study, I, is understood as a bidimensional structure of pixels, each one being connected to its natural neighborhood on the earth surface. Of course, each pixel (*information unit*) will be characterized by a fixed number of measurable attributes. These attributes can be, for example, the values of the three bands of the visible spectrum (red, green and blue), the whole family of spectrum band intensities, or any other family of physical measures.

The information about our image is in this way summarized

0-7803-8353-2/04/$20.00 © 2004 IEEE

as

$$I = \{(x^1_{i,j}, ...x^b_{i,j}) \ / \ (i,j) \in P\} \quad (1)$$

in case each pixel is being characterized by b numerical measures, and the $r \times s$ matrix P identifies the associated set of pixels in which the image I is divided,

$$P = \{(i,j) \ / \ i = 1, \ldots, r, \ j = 1, \ldots, s\} \quad (2)$$

meaning that we are dealing with a rectangular picture with $r \times s$ pixels.

Hence, our set of pixels P is being described by means of their cartesian coordinates { } and { }, in such a way that pixels (i, j) and (i', j') cannot be linked if

$$|i - i'| + |j - j'| > 1 \quad (3)$$

so two pixels can not be linked unless they share one coordinate and the other one is contiguous (see figure 1).

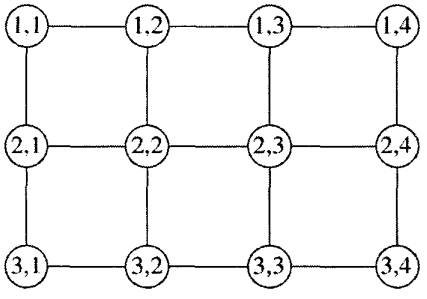


Fig. 1. Pixel network

Given such an image I, a standard crisp classification problem pursues a *nice* partition, each region being a subset of pixels, to be considered a candidate for a class (not many classes with big degree of explanation and low variance within each class are standard conditions for a nice partition, see [3]).

Quite often, the first classification stage implies a classification pixel by pixel, measuring some distance between pixels and certain patterns previously defined. Each pixel is then associated to those patterns. But in a second classification stage we impose some *structural* property on pixels so *manageable* classes can be identifies (for example, we may impose that a certain class should be a convex set of pixels).

In this way, a key tool under our approach will be also a distance

$$d \ : \ P \times P \longrightarrow [0 \ \infty) \quad (4)$$

between the measured properties of pixels. Such distance can be based, for example, upon the Euclidean distance in ^{b} (of course, any other *ad hoc* distance can be alternatively taken into account). Obviously, the classification process will be strongly dependent on this distance d, to be carefully chosen taking into account image features and classification purposes.

Anyway, once such a particular distance has been fixed, we can proceed to define a fuzzy graph (see, e.g., [13], [15], [19]) showing distances between numerical description of contiguous pixels, and then evaluate similitude between contiguous pixels.

Let us remind that given

$$\tilde{G} = (V, \tilde{E}) \quad (5)$$

a fuzzy graph, V being its node set and $\tilde{E}$ being its fuzzy edges, such a fuzzy graph is then characterized by a matrix $\mu = (\mu_{ij})_{i,j \in V}$ where

(6)

and

$$\mu_{\tilde{E}} : V \times V \longrightarrow M \quad (7)$$

is the associated membership function. Each element $\mu_{ij} \in M$ represents the intensity level of the edge $\{i,j\}$ for any $i, j \in V$. The set M is linearly ordered in such a way that $\mu_{i\,j} \prec \mu_{i\,,j}$ means that the intensity level of edge $\{\ ,j\}$ is lower than the intensity level of edge $\{i', j'\}$, allowing intensity gradation. For example, intensity leves can be restricted to three degrees: null "n", low "l" or high "h" ($M = \{n, h\ \}$).

Such a fuzzy graph can be also denoted as

$$\tilde{G} = (V, \mu \quad) \quad (8)$$

Notice that in case $M = \{0, 1\}$, our fuzzy graph $\tilde{G}$ becomes a classical crisp graph:

$$\begin{cases} 1 & \text{if } \{i,j\} \in E \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

Coming back to our image classification problem, we can then denote by

$$\widetilde{G(I)} = (P, \tilde{E}) \quad (10)$$

the graph associated to our image I, where $M = [0, \infty)$ is the domain of the distance function d and

$$\tilde{E} = \left\{ d((i,j);(i',j')) \in M / \vee \begin{bmatrix} i = i' & |j - j'| = 1 \\ j = j' & |i - i'| = 1 \end{bmatrix} \right\}$$

Given an image I and a distance d between the measured properties of pixels, the *pixels fuzzy graph* is defined as the pair

$$\widetilde{G(I)} = (P, \tilde{E})$$

Hence, our pixels fuzzy graph can be also characterized by the set P plus two $r \times s$ matrices, D^1 and D^2, where

$$D^1_{i,j} = d((i,j),(i+1,j)) \quad (11)$$

for all (,j) { , , } { , , } and

$$D^2 \quad d((i\ j)\ (i\ j)\nparallel 1 \quad (12)$$

for all $(i,j) \in \{1, \ldots, r\} \times \{1, \ldots, s-1\}$.

We can therefore denote our *pixels fuzzy graph* $\widetilde{G(I)}$ by

$$(r, s D^{\ 1}, D^2)$$

III. THE BASIC COLORING ALGORITHM

A c-coloring (see, e.g., [17]) of a classical crisp graph

$$G \quad (\mathcal{V}E \ )$$

is a mapping

$$C \ : \ V \ \longrightarrow \ \{1, \ldots, c\} \tag{13}$$

such that

$$C(v) \neq C(v')$$

whenever $\{w, \ '\} \in E$. Any c-coloring induces a crisp classification of the nodes set , being each class associated to one color:

$$V_C(k) = \{v \in V \ / \ C(v) = k\} \qquad \forall k \in \{1, \ldots, c\} \tag{14}$$

In this way, a partition is defined on the set of pixels,

$$P = \cup_{k=1}^{c} A_k \tag{15}$$

with $_i \ _j \ ^{\prime}, \ldots \ / \ j$.

Our objective here is to translate into a fuzzy context the classical crisp procedure in order to obtain a classification of pixels through a c-coloring C of the *pixels fuzzy graph* $\widetilde{G(I)}$, in such a way that each pixel

$$p = (i, j) \in P \tag{16}$$

will be classified as $k \in \{1, \ldots, c\}$ whenever its associated color is

$$C(p) = C(i, j) = k$$

In order to color a fuzzy graph, the family of crisp α-cuts are considered:

$$E_\alpha = \{(p, \ ') \ / \ \mu_{p,p'} \geq \alpha\}$$

In particular, we propose to consider a basic crisp binary coloring process, to be successively applied according to its natural nested structure, leading in this way to a hierarchical coloring of the image. The first binary coloring analyzes the pixels set P, classifying each pixel p either as 0 or as 1. A second binary coloring can be then applied separately to both that subgraph generated by those pixels colored as 0 (to obtain the classes 00 and 01), and to that subgraph generated by those pixels colored as 1 (to obtain the classes 10 and 11). In this way, a c-coloring C will be defined on $\widetilde{G(I)}$: if $C(p) = k$, with $k = 6$ for instance, then the binary representation of $k - 1 = 5$ is 101, *i.e.*, the pixel p will be binary colored three times, respectively as 1, 0 and 1 (meaning that we have considered three binary coloring levels).

A binary coloring of a graph

$$G = (\mathcal{V}E \ )$$

is a particular case of a 2-coloring

$$col \ : \ V \ \longrightarrow \ \{0, 1\}$$

The binary coloring procedure we propose as the basic replicated procedure classifies two adjacent pixels as 0 and 1 depending on the distance between them, when compared to a prescribed threshold . Notice that a standard crisp approach assigns the same class to any two pixels whenever such a distance is *small*, no matter if they are not adjacent.

In order to define de first binary coloring procedure, we fix a value α.

Lets G_α denote the α-cut of the fuzzy graph $\widetilde{G(I)}$,

$$_\alpha \ (\mathcal{V}, \ \alpha)$$

Let

$$col \ P \longrightarrow \ \{0, 1\}$$

be a binary coloring of G_α. The first binary coloring can be then obtained by assigning an arbitrary color ("0" or "1") to an arbitrary pixel, and fixing the order in which pixels will be colored. We can start, for example, with pixel $(1, 1)$ in the top-left corner of the image, and then pixels can be colored from left to right and from up to down, in the following way:

$$col(i+1, j) = \begin{cases} col(i, j) & if \ d^1_{i,j} < \alpha \\ 1 \quad l(i \ j) & if \ d^1 \ \geq \end{cases} \tag{17}$$

for all $(i, j) \in \{1, \ldots, r-1\} \times \{1, \ldots, s\}$, and

$$col(i, j)+1 \qquad \begin{cases} col(i, j) & if \ d^2_{i,j} < \alpha \\ 1 - col(i, j) & if \ d^2_{i,j} \geq \alpha \end{cases} \tag{18}$$

for all $(i, j) \in \{1, \ldots, r\} \times \{1, \ldots, s-1\}$.

Given a colored pixel (i, j), the adjacent pixels $(i+1, j)$ and $(i, j)+1$ can be then colored in a similar way. But notice that since pixel $(i+1, j)+1$ can be colored either from pixel $(i+1, j)$ or from pixel $(i, j)+1$, both coloring processes may not lead to the same color (we can call it an *inconsistent* coloring). This means of course that our binary coloring procedure is also dependent on the particular ordering we have chosen for coloring.

In the algorithm we propose, all pixels are being colored at first, and then we look for a value α^* assuring consistency. Pixels are being classified either into a class "0" or a class "1". We then proceed to get a more precise color for both classes (class "0", for example, will switch either into "00" or "01"). This is being done separately, by alternatively activating only one of the classes already colored in a previous stage. Same will apply in subsequent stages, in such a way that the above binary coloring process is applied to those activated pixels under consideration, i.e., a subset of pixels $P' \subset P$. This subset of pixels P' getting a more precise color at each stage can be also characterized by a matrix act such that

$$act(i, j) = 1, \forall (i, j) \in P' \tag{19}$$

and

$$act(i, j) \quad 0, \forall (i, j) \notin P \tag{20}$$

Initially, $act(i, j) = 1, \forall (i, j) \in P$.

The classification process induced by the previous binary coloring can be refined, in order to classify pixels of class 0 in two subclasses, by applying an analogous procedure to each matrix act, defined as

$$act(i,j) = \begin{cases} 1 & \forall (i,j) \in P \not col \quad (i,j) = 0 \\ 0 & otherwise \end{cases} \tag{21}$$

If the key procedure, which we called *bincol*, is applied with a *consistent* value α (see [10]), the activated pixels will be again classified as 0 or 1, in such a way that two new classes are defined ("00" and "01"). Analogously, another two classes, 10 and 11, will be defined when this second coloring process is applied to class 1, In this way, four classes are being obtained: "00", "01", "10" and "11", and classified pixels can be then identified by the mapping:

$$C \ : \ P \longrightarrow \{0,1,2,3\} \tag{22}$$

where $C(i,j)$ is the integer number associated to a binary description.

The same coloring procedure can be successively applied to each family of pixels belonging to any previously defined color, meanwhile there are adjacent pixels with potential inconsistencies. Hence, if this coloring process is successively applied t times, we shall be obtaining 2^t classes.

It is important to point out that our coloring process is equivalent to a hierarchical classification procedure, where a set of nested clusters is obtained as output.

IV. Computational complexity analysis

Indeed, the above algorithm has no polinomial complexity.

In the worst case, the number it of iterations attains the value $r \times s - 1$ and all color classes except one are void.

In the limit case, when $t \to \infty$, the coloring

$$C \ : \ P \longrightarrow \{0,1,\ldots,2^t - 1\}$$

thus obtained can be understood as a continuous $(0,1)$-coloring. This continuous coloring can be viewed as a *grey* (black and white) coloring of the image, with the property that the intensity difference of the color of two adjacent pixels will be proportional to the distance between them.

Anyway, the above coloring algorithm is very inefficient from a computational point of view, when handling real images. An appropriate decreasing scheme of parameter α, allowing an acceptable ratio of inconsistent pixel squares, will be the core of the relaxed coloring algorithm we introduce in the next section.

V. A relaxed coloring algorithm

Since we should be expecting the existence of quite a number of inconsistencies when dealing with medium size images, it may be the case that our decreasing search for a *consistent* value of α reaches the value 0. Consequently, pixels cannot be classified. In order to avoid such a problem, in this section we propose to relax the constraint of consistency in a new relaxed binary coloring procedure, allowing some few inconsistencies (lets say one per cent of squares). Computational complexity will also be addressed by fixing a small the number of iterations it (4 or 5, for example).

Let $incratio$ be defined as the ratio of inconsistencies of a binary coloring process, i.e., the number of inconsistent pixels divided by the total number of squares subject to inconsistency, $(r-1)(s-1)$. On one hand, large values of $incratio$ are associated to a low number of classes (the value α does not needs to be decreased). On the other hand, the greater this value $incratio$, the greater the number of pixels with a *wrong* color. Hence, we can look for some compromise between these two arguments. Such a compromise can be attained for small inconsistency ratios, lets say 0.01. Each inconsistent pixel should be then *isolated* so that its inconsistency does not induces inconsistency of adjacent pixels.

Given col be any binary coloring, then we proceed to assign new colors to every pixel, following a sequence still according to the same order already considered in the previous binary algorithm (left to right and up to down).

Again, coloring process of the first row, from left to right, does not produce inconsistencies, neither first vertical coloring. Inconsistencies, if any, will appear when a *horizontal* arc is being checked with an already *vertically* colored arc. Let $ninc$ be the total number of those inconsistent pixels.

It can be checked that for each activated pixel $(i,j) \in P$ we can associate the value $act(i,j) = 2$ whenever such a pixel has not been yet colored, and $act(i,j) = 1$ otherwise. Initially, $act(i,j) = 2$ for all activated pixels (i,j). Once a pixel (i,j) is colored, $act(i,j) = 1$ if it is consistent); otherwise, $act(i,j) = -1$. As pointed out above, these inconsistent pixels must be isolated in order to avoid contamination of adjacent pixels.

Any inconsistent pixel can be arbitrarily colored. Subsequent iterations will smooth the effects of such arbitrary coloring.

In order to avoid the exponential growth of computations of the coloring algorithm, the parameter it can be bounded (in this way we control the number of binary partitions). Let it_M be such a bound (a small value of it_M will be allowing few inconsistent squares but computational time will increase).

Once the value of it_M has been fixed, different values of α must be selected: let

$$\{\alpha_{it}\}_{it}$$

the family of these selected values, $it \in \{1,\ldots,it_M\}$. A procedure we denoted as $vecalpha$ will return these values into the vector

$$(\alpha_1,\ldots,\alpha_{it_M})$$

The associated pseudocode for our relaxed coloring algorithm will be based upon the successive relaxed binary coloring at different levels α_{it}.

Details of all referenced and related procedures can be obtained from the authors under request [10] (see also [9]).

VI. Application to a particular image

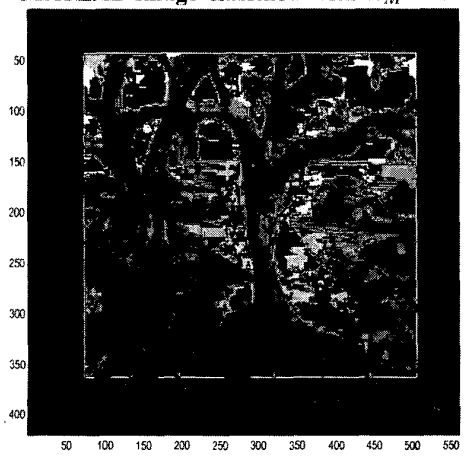
The above algorithm has been applied to a standard image, taken from the *MATLAB* package (a water colored tree, already considered by in [2], reproduced here as a *grey* picture (see figure below).

Original MATLAB image

Hence, a sequence of binary classifications has been obtained, so a number is associated to each pixel and the image is divided into homogeneous regions of adjacent pixels, all of them with the same associated number. In order to be able to visualize these homogeneous regions, we have associated a color RGB to each one of them, obtained as the mean of the original color of pixels in each region. This mean color represents an homogeneous class (notice that such a procedure can be extended to other characteristics not being a color meanwhile the number we deal with three characteristics at most). In this way we can *paint* our successive classifications.

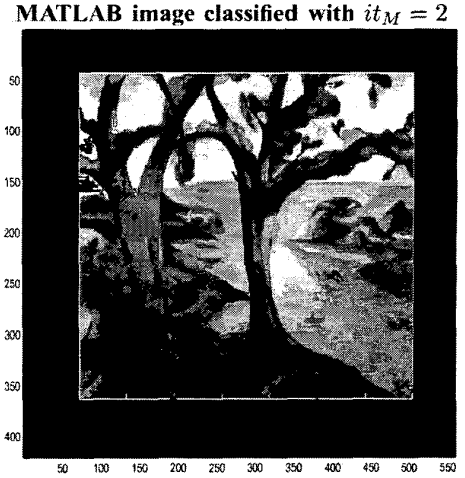
In figure below it is shown results with only one iteration, i.e., allowing just two colors (please take into account that these are the *grey* reproductions of the original pictures).

MATLAB image classified with $it_M = 1$



In this particular MATLAB image main features seem to be captured with only two iterations, i.e., four colors (see figure below).

MATLAB image classified with $it_M = 2$



The above algorithm has been applied to several alternative images, in general obtaining very good classification behavior with few iterations. Although we are aware of potential misbehavior with extremely *smooth* images, these results suggest no practical relevance when we introduce a bound on the parameter *it* (number of iterations) of the above relaxed algorithm. Since all basic procedures are polinomial, whenever the number of coloring procedures is bounded (e.g., $it \leq 10$), computational complexity of the above relaxed algorithm becomes polinomial.

VII. Final comments

The coloring algorithm we propose in this paper provides as a main output a picture of the image, where a set of different colors shows a structured family of classes that can be easily understood by decision makers and help them to define classes. The sequential application of a basic binary coloring procedure brings crisp techniques into complex images, allowing a first approach to a *fuzzy* picture of fuzzy classes, taking very much into consideration behavior of adjacent pixels. As any heuristic approach, our algorithm is not free of some potencial misbehavior, which will suggest future improvements (for example, consideration of additional rings surrounding each pixel, and not only adjacent pixels, can be tried when we face extremely *smooth* images with very small variation between adjacent pixels).

Such a coloring algorithm should be understood as a piece in a more sophisticated image classification process, producing a first unsupervised classification that may give interesting hints about the structure of the image under study. In this sense, we think that fuzzy coloring techniques should play a key role in the future, in order to help decision makers to capture a global view of images containing fuzzy classes.

ACKNOWLEDGMENT

This research has been partially supported by the Government of Spain, grant BFM2002-0281.

REFERENCES

[1] A. Amo, D. Gómez, J. Montero and G. Biging: "Relevance and redundancy in fuzzy classification systems," Mathware and Soft Computing 8, 203-216 (2001).

[2] A. Amo, J. Montero and G. Biging: "Classifying pixels by means of fuzzy relations," International Journal of General Systems 29, 605–621 (2000).

[3] A. Amo, J. Montero, G. Biging and V. Cutello: "Fuzzy classification systems," European Journal of Operational Research 156, 495–507 (2004).

[4] A. Amo, J. Montero and V. Cutello: "On the principles of fuzzy classification," Proceedings of the annual North American Fuzzy Information Processing Society conference (NAFIPS); pp. 675–679 (1999).

[5] A. Amo, J. Montero, A. Fernández, M. López, J. Tordesillas and G. Biging: "Spectral fuzzy classification: an application," IEEE Transactions on Systems Man and Cybernetics (C) 32, 42–48 (2002).

[6] J.C. Bezdek: *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[7] J.C. Bezdek and J.D. Harris: "Fuzzy partitions and relations: an axiomatic basis for clustering," Fuzzy Sets and Systems 1, 111–127 (1978).

[8] G.M. Foody: "The continuum of classification fuzziness in thematics mapping," Photogrammetric Engineering and Remote Sensing 65, 443–451 (1999).

[9] D. Gómez, J. Montero, J. Yáñez and C. Poidomani: "A Coloring Algorithm for Image Classification," *Proceedings of Fuzzy Logic in Nuclear Science conference, F.L.I.N.S.* (Blankenberghe, Belgium, September 1-4, 2004).

[10] D. Gómez, J. Montero, J. Yáñez and C. Poidomani: "A fuzzy graph coloring algorithm for image classification," Technical Report, Department of Statistics and Operational Research, Faculty of Mathematics, Complutense University of Madrid, Spain (2004).

[11] J. R. Jensen: *Introductory Digital Image Processing. A Remote Sensing Perspective*, Prentice Hall, Neew York, 1996.

[12] E.E. Kerre and M. Nachtegael: *Fuzzy Techniques in Image Processing*, Physica-Verlag, Heidelberg, 2000.

[13] L. Kóczy: "Fuzzy graphs in the evaluation and optimization of netwoks," Fuzzy sets and systems 46:307–319 (1992).

[14] J. Montero, A. Pearman and J. Tejada: "Fuzzy multicriteria decision support for budget allocation in the transport sector," TOP 3:47-68 (1995).

[15] J.N. Mordeson and S. Nair: *Fuzzy graphs and Fuzzy Hypergraphs*, Physica-Verlag, Heidelberg, 2000.

[16] S.K. Pal, A. Ghosh and M.K. Kundu: *Soft Computing for Image Processing*, Physica-Verlag, Heidelberg, 2000.

[17] P.M. Pardalos, T. Mavridou and J. Xue: "The Graph Coloring Problem: A Bibliographic Survey". In: D.Z. Du and P.M. Pardalos (eds.): *Handbook of Combinatorial Optimization* (vol. 2), Kluwer Academic Publishers, Boston, 1998; 331–395.

[18] P.P. Perny and B. Roy: "The use of fuzzy outranking relations in preference modelling," Fuzzy Sets and Systems 49:33–53 (1992).

[19] A. Rosenfeld: "Fuzzy graphs," In: L.A. Zadeh, K.S. Fu and M. Shimura (Eds.): *Fuzzy sets and their applications to cognitive and decision processes*, Academic Press, New York, 1975; 77–95.

[20] E.H. Ruspini: "A new approach to clustering," Information and Control 15:22–32 (1969).

[21] J. Siskos, J. Lochard and J. Lombard: "A multicriteria decision making methodology under fuzziness: application to the evaluation of radiological protection in nuclear power plants," in H.J.Zimmermann, L.A. Zadeh and B.R. Gaines (eds.): *Fuzzy Sets and Decision Analysis*, North Holland, Amsterdam, 1984; 261–283.