Preserving Interpretability in the Optimization of Fuzzy Systems: a Generic Algorithm in a Topological Framework

R. de Aldama[1] * M. Aupetit[1]
[1] CEA-LIST

CEA, LIST, Laboratoire Analyse de Données et Intelligence des Systèmes,
Gif-sur-Yvette F-91191, France ;
ricardo.de.aldama@iscpif.fr, michael.aupetit@cea.fr

Résumé :
Nous proposons un formalisme mathématique pour analyser l'interprétabilité d'une partition floue, ainsi qu'un algorithme générique pour la préserver pendant le processus d'optimisation du système flou. L'approche est assez souple et il aide à automatiser le processus d'optimisation. Certains outils sont empruntés au domaine de la topologie algébrique.

Mots-clés :
système flou, partition floue, interprétabilité, optimisation

Abstract:
We present a mathematical framework to analyze the interpretability of a fuzzy partition and a generic algorithm to preserve it during the optimization of the fuzzy system. This approach is rather flexible and it helps to highly automatize the optimization process. Some tools come from the field of algebraic topology.

Keywords:
fuzzy system, fuzzy partition, interpretability, optimization, tuning

1 Introduction

Fuzzy ruled based systems have found many real-world applications. One of their appealing features is that in most cases they are easily interpretable by humans. However, when used to tackle complex problems, there is often need to make use of automatic optimization methods that improve the original system (cf. [2]). These automatic methods have a drawback: It may entail important losses in the interpretability of the system, in particular in the fuzzy partitions. The goal of this paper is to deal with this loss of interpretability.

Let us say that the fuzzy system under study is composed of rules of the form "If x_1 is A_1 and ... x_n is A_n, then y is B", where x_i and y are linguistic variables and A_i and B are predicates. These predicates have their numeric counterparts: The fuzzy sets which formalize their meaning. If these rules are fixed and we adjust the parameters determining the fuzzy sets, the process is usually called *tuning* or *parametric optimization.* If we adjust the number of rules, the space of functions to which the fuzzy sets belong, or some other high-level components of the fuzzy system, the process is usually called *structural optimization* or *learning.* The work presented in this paper concerns the case of parametric optimization.

Although there is no standard definition for the notion of interpretability of a fuzzy system, we can distinguish, following [1, 3], two levels of interpretability: That of fuzzy partitions and that of rule analysis. In this paper we deal with the problem of preserving the interpretability of the fuzzy partitions during the process of parametric optimization. We can divide this work in two parts: Firstly we provide a mathematical framework in which the concept of interpretability may be formalized, and secondly we provide a generic algorithm that takes as input a fuzzy system and a function to optimize (that measures the quality of a fuzzy system) and gives as output an optimized fuzzy system that pre-

*Current address: Télécom ParisTech - TSI, CNRS LTCI, 46 rue Barrault, F-75634 Paris Cedex 13.

serves interpretability.

Thanks to this formalization the process of optimizing will be, in our view, much more painless for the user than in previous approaches. In particular it may be carried out not only by experts in optimization of fuzzy systems as usual, but also by users that are just experts in the problem-specific domain and whose knowledge in fuzzy theory may be limited.

In our approach we do not fix a priori the notion of interpretability. The mathematical framework that we propose is problem-independent and sufficiently generic to let the user establish which configuration he wants to preserve during the optimization process. The essential point is the formalization of the notion of interpretability in topological and geometrical terms. Its preservation implies some particular constraints on the acceptable solutions for the optimization problem. In the generic algorithm that we propose, the codification and verification of these constraints is automatically done.

The geometric and topological analysis begins with a fuzzy system that the user considers interpretable. The domain of each variable is partitioned in such a way that the relative order of the different membership functions is constant on each region. These regions, and the order relations associated to them, will determine the geometric and topological constraints that will be taken into account during the optimization. In order to codify this information, a key role is played by homology groups. We make use of these well-known algebraic objects, which are able to capture a very significant part of the topology of a space and are well-suited for computer calculations. There exist several implementations to compute different homology groups. The reader interested in more details may consult for instance [5, 6, 8].

2 A topological framework for the analysis of interpretability

2.1 The main idea

What we propose in this paper is not an absolute definition of interpretability, but rather a framework in which the actual definition, which will strongly depend on the user, can be expressed. We may talk then, given a user U, of *interpretability relative to* U. Our approach is strongly focused on topology: Our viewpoint is that the properties of the fuzzy partition that the user requires to be preserved are essentially of a topological nature.

Let us say a user defines a fuzzy partition such as the one on Figure 1. It seems reasonable to consider that the user requires the optimization process to preserve, at least, the order of the terms. This order, though not explicitly formalized, underlies the solution we usually find in the literature: To strongly constrain the possible variations of the membership functions, in order to obtain very similar configurations as the original one (as in Figure 1).

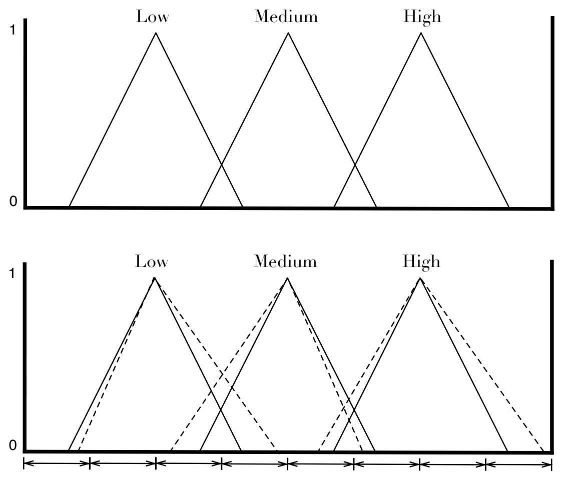

Figure 1: Example of a fuzzy partition and some possible constraints on it.

Some difficulties may arise if we try to define an order in a case such as that of Figure 2. In more general cases, such as those of 2-dimensional variables, the concept of order may not even make any sense. However, there are always some basic properties that the user wants to preserve to be able to attach some meaning to the system. In our approach, these properties have a topological nature and are locally determined by the values of the different membership functions. In particular, we think that the relative order of these values is crucial.

The main idea is to partition the numeric domain of the variable into regions in which the relative order of the membership functions is constant, such as in Figure 2.

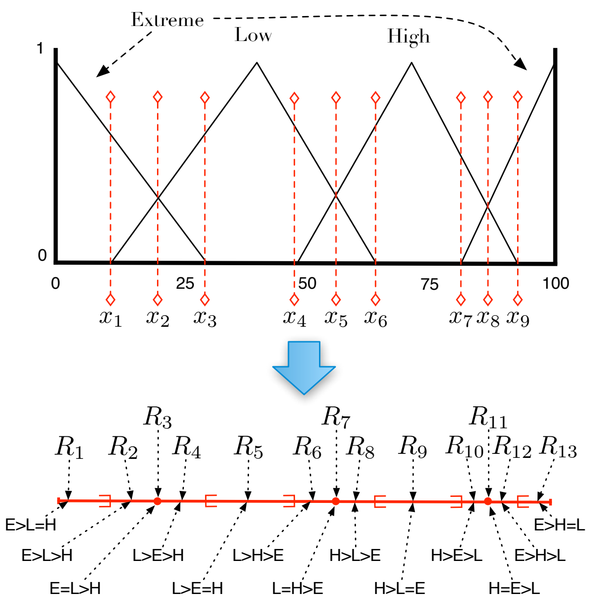
Extreme
Low
High
1
0
0
25
50
75
100
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9
R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9 R_{10} R_{11} R_{12} R_{13}
E>L=H
E>L>H
E=L>H
L>E>H
L>E=H
L>H>E
L=H>E
H>L>E
H>L=E
H>E>L
H=E>L
E>H>L
E>H=L

Figure 2: Decomposition of the domain in regions R_i in which the relative order of the membership functions is constant. We suppose that the domain of the variable is the interval [0, 100].

Some properties of this partition will be required to be preserved during the optimization process. Examples of such properties could be:

- There is a region R_2 in which the relation Extreme > Low > High holds, with neighbors R_1 and R_3, such that in R_1 we have Low = High < Extreme, and in R_3 we have Extreme = Low > High.
- The value 50 belongs to the region R_6 that verifies Low > High > Extreme.

The rest of the section will be devoted to make this main idea more precise. In particular, we will present two key notions: The *geometric* and *topological signatures*.

2.2 Notation and definitions

The definitions concerning fuzzy systems, such as linguistic variable, membership function, etc. are standard (see for instance [7]). We consider that the numeric domains associated to each linguistic variable are equipped with a natural topology (as it is the case with $\mathbb{R}^n$).

- Let Ω be the set of possible fuzzy systems under consideration, and let $A = A_1 \times \ldots \times A_n$ (typically $A \subseteq \mathbb{R}^n$) be the domain of the parameter vector that we consider as determining a fuzzy system. A solution to our optimization problem will be then an element $\bar{a} \in A$.
- We denote by $\omega : A \to \Omega$ the map that determines a fuzzy system $\omega(\bar{a})$ from the parameter vector $\bar{a}$. In particular ω determines every membership function of the system.
- We denote by V the set of all linguistic variables and we suppose it is the same for every $\omega \in \Omega$. We denote by Dom_v the domain of a linguistic variable $v \in V$.

2.3 Geometric signature

Let $\omega \in \Omega$ be a fuzzy system and $v \in V$ a linguistic variable. The *geometric signature of ω relative to v*, that we denote by $\mathscr{G}_\omega(v)$, is a mathematical object that captures all the potentially interesting properties of the partition induced by ω on Dom_v. It provides the regions in which the relative order of the different membership functions is constant, and together with each region, its corresponding order.

As an illustration, consider that for a certain $\omega \in \Omega$ and $v \in V$, Figure 2 represents the partition induced by ω on Dom_v. In this case $\mathscr{G}_\omega(v)$ is the map that associates to $i \in \{1, \ldots, 13\}$ the region R_i, together with the corresponding order relation on terms. For instance:

- $\mathscr{G}_\omega(v)(1)$ is the region R_1, i.e. the interval $[0, x_1]$, together with the order Extreme $>$ Low $=$ High.
- $\mathscr{G}_\omega(v)(3)$ is the region R_3, i.e. the point $\{x_2\}$, together with the order Extreme $=$ Low $>$ High. In practice, regions of low dimension (0 in this case) may be ignored.

In some cases the user might consider certain "dummy" functions $Dom_v \to [0, 1]$ to code particular constraints, such as interactions between membership functions. For instance, to deal with strong partitions we might consider the constant function 1 and the function $\sum_i \mu_i(x)$ (where μ_i represents the i-th membership function).

The *geometric signature of* ω, denoted by $\mathscr{G}_\omega$, is the map that associates $\mathscr{G}_\omega(v)$ to $v \in V$.

2.4 Topological signature

The *topological signature of ω relative to v*, that we denote by $\mathscr{T}_\omega(v)$, is a a weaker concept than that of the geometric signature, i.e. for $\omega, \eta \in \Omega$, if $\mathscr{G}_\omega(v) = \mathscr{G}_\eta(v)$ then $\mathscr{T}_\omega(v) = \mathscr{T}_\eta(v)$. It codes the topological information contained in $\mathscr{G}_\omega(v)$. The *topological signature of* ω is the map that associates $\mathscr{G}_\omega(v)$ to $v \in V$. We denote by $\mathscr{T}_\omega$.

In the field of computational topology, the use of homology groups is widely spread to deal with the topology of a space. We will not provide here any definition concerning homology theory, since it is out of the scope of this paper; nevertheless we should say that these groups are topological invariants of algebraic nature, that capture an important part of the topological information of a space and are well-suited from an algorithmic viewpoint. The reader interested may consult for instance [5], a standard reference in algebraic topology, or [6, 8] for an approach more focused on computational aspects.

We can propose then to code the topological signature in terms of these homology groups, that we denote by H_N for $N \in \mathbb{N}$. Let $v \in V$ and consider $\omega, \eta \in \Omega$ such that ω induces a partition on Dom_v composed of regions $R_1, \ldots, R_n$ and η induces a partition on Dom_v composed of regions $S_1, \ldots, S_n$. Then we say that $\mathscr{T}_\omega(v)$ and $\mathscr{T}_\eta(v)$ are equal if there is a n-permutation σ such that:

1. the order on terms corresponding to R_i is the same as that of $S_{\sigma(i)}$ for $i = 1, \ldots, n$, and moreover
2. $H_n(\bigcup_{k \in K} S_{h(k)}) \approx H_n(\bigcup_{k \in K} R_k)$ for each $K \subseteq I$ and $n \in \mathbb{N}$.

The homology groups are characterized by some integers, namely the *Betti numbers* and the *torsion coefficients*; they will be stored and used as topological signature. However, we should say that this is a general-purpose coding; in practice there may be different ways to implement the notion topological signature, depending mostly on the nature of Dom_v. In some cases the computation of these homology groups may not be necessary and a much more efficient coding can be devised.

To illustrate the notion of topological signature, consider that for a certain $\omega \in \Omega$ and

$v \in V$, Figure 2 represents the partition induced by ω on Dom_v. In this case, $\mathscr{T}_\omega(v)$ provides for each $i \in \{1, \dots, 13\}$ the the order on terms corresponding to the region R_i, and for each $K \subseteq \{1, \dots, 13\}$ the topological information of $\bigcup_{i \in K} R_i$. For instance, if we consider $K = \{4, 5\}$, $\mathscr{T}_\omega(v)$ codes the fact that $R_4 \cup R_5$ is connected, and if we consider $K = \{1, 6, 9\}$ the fact that $R_1 \cup R_6 \cup R_9$ is composed of three connected components. Essentially, $\mathscr{T}_\omega(v)$ codes the following information:

1. There are 13 regions R_i (each one being a connected set),
2. the order on terms corresponding to R_1 is Extreme $>$ Low $=$ High, that of R_2 is Extreme $>$ Low $>$ High, etc.
3. R_1 is neighbor of R_2, R_2 is neighbor of R_1 and R_3, etc.

Hence if we consider another $\eta \in \Omega$ whose decomposition of Dom_v is given by regions $S_1, \dots, S_M$, then $\mathscr{T}_\eta(v) = \mathscr{T}_\omega(v)$ iff $M = 13$, and for some permutation σ we have:

1. The order on terms corresponding to $S_{\sigma(1)}$ is Extreme $>$ Low $=$ High, that of $S_{\sigma(2)}$ is Extreme $>$ Low $>$ High, etc.
2. $S_{\sigma(1)}$ is neighbor of $S_{\sigma(2)}$, $S_{\sigma(2)}$ is neighbor of $S_{\sigma(1)}$and $S_{\sigma(3)}$, etc.

3 User interactions: An operational definition of interpretability

As we have already mentioned, we do not provide an absolute definition of interpretability, but rather, given a user U, a conceptual and operational framework to deal with interpretability relative to U. The goal of this section is to show how we can define and manipulate this interpretability relative to U, relaying on the notions presented in Section 2 and, importantly, on some interactions with U. We should mention that the interactions we present here seem to us flexible enough to cover most part of needs; however, other interactions could be consider.

Our base hypothesis is that the notion of interpretability has essentially a topological flavor. An oversimplified version of this hypothesis would be :

Assumption 1. *If a user U considers $\omega \in \Omega$ to be interpretable, then there is no $\eta \in \Omega$ considered as interpretable by U and such that $\mathscr{T}_\eta \neq \mathscr{T}_\omega$.*

Assumption 1 is slightly stronger than the actual assumption we make, however it synthesizes quite clearly the main idea of our approach. We want to provide an operational definition of interpretability relative to U. For this we need, of course, some interaction with U. Since we are talking about interpretability in the context of the optimization of a fuzzy system, we suppose that there exists at least one $\omega_0 \in \Omega$ that is interpretable relative to U and that U is capable of describing it, i.e. providing a parameter vector $\bar{a} \in A$ such that $\omega(\bar{a}) = \omega_0$.

This is the slightest interaction with U that our method needs. However, if we want to make our method more flexible, we can allow U to provide more information. Next we present the two other kind of interactions we may consider.

Relaxation of the topological conditions This is basically a relaxation of Assumption 1. Once U has provided a $\omega_0 \in \Omega$ that he considers to be interpretable, one could consider that for a solution $\bar{a} \in A$ to be acceptable, i.e. such that $\omega(\bar{a})$ is interpretable relatively to U, $\bar{a}$ must satisfy $\mathscr{T}_{\omega(\bar{a})} = \mathscr{T}_{\omega_0}$. Instead, we may let the user relax this condition: He could omit, if he wishes, some of the topological conditions imposed by $\mathscr{T}_{\omega_0}$. Typically it may consist in merging different regions and requiring a relaxed order on terms; in this case the relaxed order should be compatible with the order of the merged regions (see example in Figure 5). This notion of compatibility could be easily formalized in terms of the lattice of partial orders on terms.

Addition of geometric conditions Conversely U may strengthen the conditions for a solution to be considered interpretable. This extra conditions are of a geometric rather than topological nature. This will allow U to specify the regions to which certain points should belong. If we consider again Figure 2, U may want to include the condition "$0 \in R_1$", that is "0 should belong to the region indexed by 1", or more precisely "0 should belong to the region whose corresponding order on terms is Extreme $>$ Low $=$ High, that is neighbor of other region (namely R_2) whose corresponding order is Extreme $>$ Low $>$ High, that is neighbor of etc. ". It is clear that we can codify these kind of conditions in terms of the point 0 and the signature $\mathscr{T}_{\omega_0}$.

4 Algorithm

We present here the different parts of a generic algorithm that fulfills our purpose: To optimize a given fuzzy system while preserving its interpretability. In Figure 3 we can see a scheme of this algorithm, but rather than explaining it in its more abstract form, we prefer to focus in the explanation of a particular example. The generic case will easily be induced from this description.

Let us consider a certain fuzzy system ω_0 modeling a 2-dimensional problem and in which only one linguistic variable v is involved. For instance there may be some rules involving the terms East, West and Center that are used to activate some procedures: We could imagine a fuzzy controller that produces policy decisions (e.g. public transports, taxes, etc.) for towns in a certain area, following rules of the type "If town T is in region East then apply policy P to T". An example of the membership functions associated to East, West and Center can be found in Figure 4.

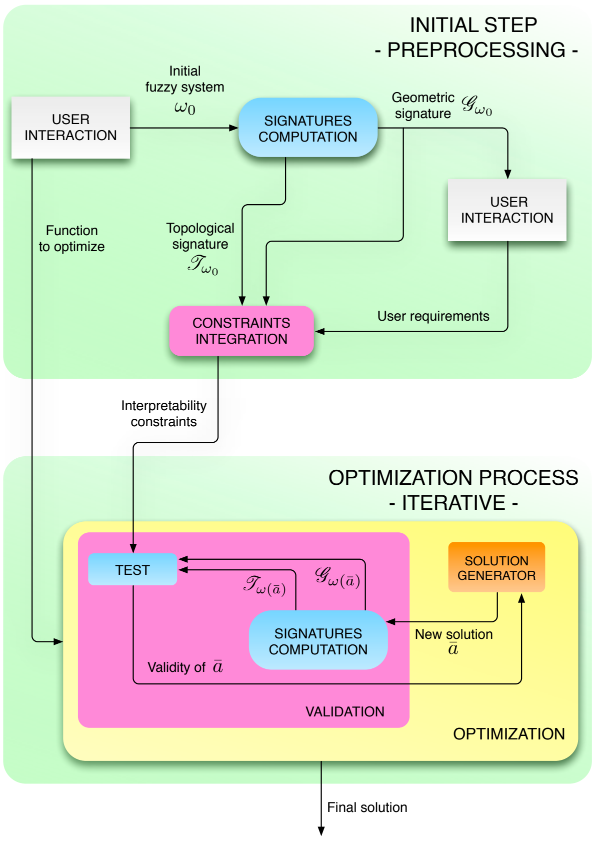

Figure 3: Scheme of the algorithm.

Let us say a user U considers ω_0 as interpretable and wants to optimize it using a performance function f.

Preprocessing

Step 0. The user gives ω_0 and f as input.

Step 1. The first part of the algorithm consists in computing the geometric signature, that is the regions in which the order of terms is constant. Let $\mu_{\text{West}}, \mu_{\text{Center}}, \mu_{\text{East}} : Dom_v \to [0,1]$ be the membership functions corresponding to the terms West, Center and East. The domain is discretize and each

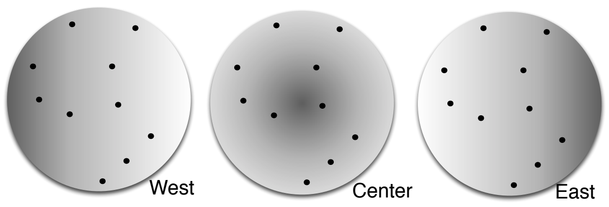

Figure 4: Example of three membership functions associated to a 2-dimensional variable. Darker colors represent values closer to 1. The black dots represent towns.

function is evaluated on each point of the grid. This evaluation induces a label for each point, e.g. a point x gets the label West < East < Center if $\mu_{\mathsf{West}}(x) < \mu_{\mathsf{East}}(x) < \mu_{\mathsf{Center}}(x)$. Then we can explicitly compute the regions (maximal connected components with the same label) by using, for instance, the method described in [4].

Step 2. At this point comes the second interaction with U (apart from Step 0): The regions are presented to him (we can omit regions of dimension 0 and 1) and then he can, first relax the topological conditions that will be imposed to the acceptable (interpretable) solutions, and afterwards impose some geometric conditions. In Figure 5 we can see an example in which U, only interested in the function with highest value, decides to relax the topological conditions by merging the regions that share the same highest-valued function; he also imposes the geometric conditions "town X must belong to the region in which the value of West is the biggest" and "town Y must belong to the region in which the value of Center is the biggest".

Step 3. No other interaction with U is needed, since he has just operationally defined what he considers as interpretable: This definition is essentially contained in the right side of Figure 5 (in Figure 6 we can find examples of interpretable and not-interpretable solutions). This information is then coded in terms of homology groups, following the explanations of Section 2 and using for instance the algorithms presented in [6].

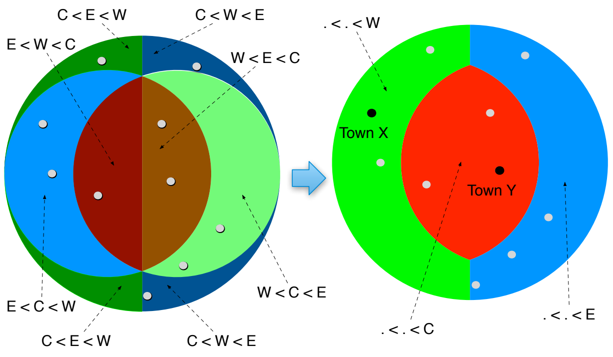

Figure 5: Regions induced by the functions in Figure 4. On the right side, a relaxation of the topological conditions and the addition of two geometric conditions: Since only the highest-valued functions are relevant, some labels are merged; moreover town X must belong to the region in which μ_{West} is the highest-valued function and town Y to the region in which μ_{Center} is the highest-valued function.

Optimization process

Step 4. This well-coded information, as well as the function f and ω_0, is given as an input to an optimization algorithm, and is interpreted as a constraint C on the (signatures of the) solutions. This optimization algorithm may be of different types (e.g. metaheuristic or exact) depending on the nature of f. As it is the case for any iterative optimization algorithm, it should contain a "solution generator" module. This module may have different ways of dealing with constraints. The most basic option would be to test C for each solution that it generates and to use the result of the test to generate a new solution. Another option would be to do some kind of preprocessing, in which the acceptable domain is approximated, and then to only generate valid solutions. In any case we will need to iterate a process similar to Step 1 and Step 3:

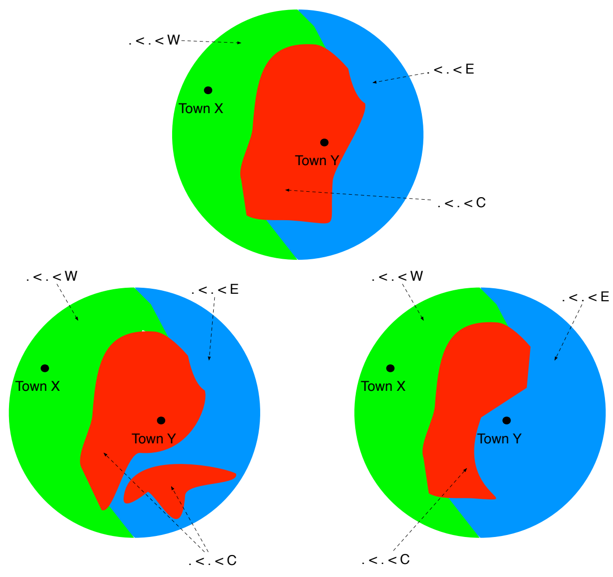

Figure 6: The top figure corresponds to a possible configuration of a solution that is acceptable. The bottom figures correspond to unacceptable configurations: The left one does not satisfy the topological conditions since the region whose highest-valued function is μ_{Center} is disconnected; the right one does not satisfy the geometric conditions since town Y does not belong to the region whose highest-valued function is μ_{Center}.

Given $\bar{a} \in A$, compute $\mathscr{G}_{\omega(\bar{a})}$ and $\mathscr{T}_{\omega(\bar{a})}$, and use them to test if $\bar{a}$ satisfies C. This will ensure that the final solution is interpretable relative to U.

5 Conclusion

We have presented a generic method to deal with the loss of interpretability in fuzzy partitions during the optimization of a fuzzy system. It relies essentially on topological concepts and tools, which confers a solid mathematical foundation, and makes use of different well-known algorithms in computational topology. Our definition of interpretability is not absolute, but rather relative to each user, who implicitly defines the notion by means of some specific interactions. That makes this approach quite flexible. Moreover the method can be uniformly applied in many situations without the need of an expert in optimization of fuzzy systems. The next step is to fully implement the algorithm and to validate it by user-testing.

References

[1] José M. Alonso, Serge Guillaume, and Luis Magdalena. Un indice d'interprétabilité des bases de règles floues. In *Rencontres LFA francophones sur la Logique Floue et ses Applications (LFA), Toulouse*, 2006.

[2] Oscar Cordón, Francisco Herrera, Frank Hoffmann, and Luis Magdalena. *Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases*. World Scientific Publishing Company, 2002.

[3] J.V. de Oliveira. Semantic constraints for membership function optimization. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 29(1):128–138, 1999.

[4] MB Dillencourt, H Samet, and M Tamminen. A General Approach to Connected-Component Representations Labeling for Arbitrary Image Representation. *Journal of the ACM (JACM)*, 39(2):253–280, 1992.

[5] Allen Hatcher. *Algebraic topology*. Cambridge University Press, 2002.

[6] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. *Computational homology*, volume 157 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2004.

[7] G J Klir and B Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice Hall New Jersey, 1995.

[8] Afra Zomorodian. *Topology for computing*, volume 16. Cambridge University Press, 2005.