Using man-machine dialogue to design objective functions

71[1]

Abstract. Computers are very effective to solve complex problems especially because they can test a great number of solutions in a short amount of time. However, in order to find a satisfying solution, computers have to evaluate tested solutions. Defining an objective function for the automatic evaluation of a solution is a hard task. In this paper, we propose an approach to facilitate the design of objective functions. This approach allows to revise an objective function through a man-machine dialogue: the machine presents solution samples to an expert that can express his preferences. Exploration methods are then used to improve the objective function by searching in an appropriate objective function space the one that is the most consistent with the expressed preferences. A case study in the real industrial context of cartographic generalisation is presented. This case study shows the difficulty for experts to define good objective function as well as the effectiveness of our objective function revision approach.

1 INTRODUCTION

Human beings frequently have to make a choice among different solutions for a same problem. This choice can be particularly complex when the number of solutions is very high. Computer science had brought new tools to help humans to make a choice. Actually, the computational capacity of the computers allows to test numerous alternatives in a short amount of time. Unfortunately, while human experts can easily give a qualitative evaluation of a solution quality, it is often far more difficult for them to express their expectations in a formal way that can be used by artificial systems.

This paper deals with the problem of the design of an objective function that translates the user outcome expectation to the system. Our approach is based on a dialogue between the user and the system. This dialogue consists in the presentation of solution samples to the user.

In Section 2, the context of this work is introduced. Section 3 presents our approach. Section 4 describes an application of our approach in the real industrial context of cartographic generalisation. At last, Section 5 concludes and presents the perspectives of this work.

2 CONTEXT

2.1 Optimisation problem and objective function

Optimisation problems are everywhere in the real world. Solving such a problem consists in finding, in the solution space, a solution that maximises an objective function. This objective function characterises the quality of a solution. The objective function design is a key point of the resolution of optimisation problems [14, 15]. Indeed, if the objective function does not pertinently translate the "real" quality of a solution, the solutions that will be found by the system will not be satisfying.

2.2 Related Works

The problem of objective function definition is a complex problem. Many pieces of work focus on the definition of such function for specific problems [10, 18] but few propose a generic approach for helping the user of an optimisation system to define it.

A first approach to solve this problem consists in using supervised machine learning techniques. These techniques allow to induce a general model from examples labelled by a user/expert. It is possible to use these techniques to learn an objective function from examples evaluated by a user. This approach was used in several works. For example, [18] used this approach in the domain of computer vision, and [6] for cognitive radio learning.

A second approach consists in establishing a man-machine dialogue to converge toward a formalisation of the user needs. [16] proposes to present pairs of solutions to the user. For each pair of solutions, the user selects the solution he prefers. Machine learning techniques are then use to learn an objective function from the user preferences. Our work is taking place in the continuity of this work. We propose to use the same approach based on a dialogue between the user and the system established through the presentation of samples. However, we propose to generalise this approach from samples composed of pairs of solutions to samples composed of n solutions. Moreover, we propose to let the user expressed his preference in a richer way than just selecting the best solution. At last, we propose to pass from the acquisition of an objective function to a revision task. Indeed, we assume that experts can often define good objective function and that this function can contain interesting information that can help to design an even better objective function.

2.3 Formalisation of the objective function design

Our work aims at providing a method to help users to design an objective function. We assume that a set of criteria is defined to characterise the solution quality. The objective function design consists then in defining an aggregating function that allows the solution quality to be assessed by a single value. We propose to formulate this aggregating function as a weighted means balanced by a power.

Let C be the set of criteria considered, w_i the weight associated to a criterion i, $Val_i(sol)$, the value of the criterion i for the solution sol, and p, an integer higher or equal to 1. The objective function is then defined by Equation 1.

[1]

$$quality(sol) = \left(\frac{1}{\sum_{i \in C} w_i^p} \sum_{i \in C} (w_i \cdot Val_i(sol))^p \right)^{\frac{1}{p}} \quad (1)$$

When p is equal to 1, the aggregation is a simple weighted average of the criteria value. The role of the p parameter is to control the relative weight of the highest criteria values over the lowest ones: the higher p, the more the highest criteria values are taken into account in the overall solution quality. Indeed, we can prove that this quality function tends to the max when p tends to ∞ (see Equation 2).

$$\lim_{p \to \infty} \left(\frac{1}{\sum_{i \in C} w_i^p} \sum_{i \in C} (w_i \cdot Val_i(sol))^p \right)^{\frac{1}{p}} = \frac{\max\limits_{i \in C} (w_i \cdot Val_i)}{\max\limits_{i \in C} w_i} \quad (2)$$

An interest to use such a representation for the objective function is to remain easily interpretable by domain experts - it is so possible for an expert to validate the result of an automatic learning of such a function.

3 PROPOSED APPROACH

3.1 General approach

It is often difficult for experts to express their outcome expectation toward a system in a formal way. In this context, we state the hypothesis that it is easier to comment solutions in a qualitative way. Thus, we propose to base our objective function design approach on the collection of user comments toward samples of solutions.

Our approach is composed of 3 steps (Figure 1): the first one consists in generating a set of solution samples. Each sample is composed of different solutions for a same problem instance. The second step consists in capturing the comments made by the user on each of these samples. The last step consists in using these comments to automatically revise the objective function. The following sections describe these three steps.

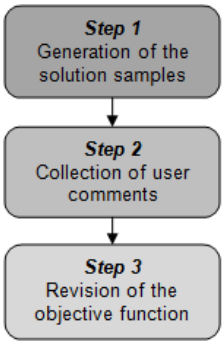

Figure 1. General approach

3.2 Building of solution samples

The first step concerns the building of the solution samples that will be shown to the user to capture his needs. Each sample is composed of n solutions of the same problem instance.

In order to build the solution samples, different problem instances have to be solved. The number of solved problem instances depends on the application context of our approach. In some contexts, it will only be possible to solve few problem instances due to the availability of the problem instances or to time constraint. Thus, for some application context, a sampling method will have to be used in order to select a representative subset of problem instances.

Once a set of problem instances solved, a set of solution samples is extracted from these instances. The solutions composing the solution samples can be chosen randomly and by a more complex strategy depending of the application context.

3.3 Capture of the user preferences

The second step concerns the collection of user comments. Each solution sample is presented to the user, who can then comment each of them.

We propose two kinds of comments:

- Comments linked to the quality of a solution.
- Comments linked to the preference of a solution over another one.

Concerning the first kind of comments, we defined two levels of quality. Let A be a solution. The two possible quality levels are:

- Solution A is $good$
- Solution A is bad

Concerning the second type of comments, we defined four possible preference relations. Let A and B be two different solutions. The four possible preference relations are:

- Solution A is far better than solution B.
- Solution A is better than solution B.
- Solution A is slightly better than solution B.
- Solutions A and B are equivalent.

Once this step carried out, we dispose of a set solution samples commented by the user.

3.4 Revision of the objective function

This last step consists in revising the existing objective function from the user comments collected in the previous step. Revising the evaluation function means finding, from the existing objective function, the parameter values (i.e. the criteria weights w_i and the power p) that best fits the comments given by the user.

We propose to formulate this problem as a minimisation problem. We define a global error function that represents the inadequacy between an objective function (and thus the parameter value assignment) and the user comments. Our goal is to find the parameter values that minimise the global error function.

Let $f_{obj}(sol)$ be the current objective function used to evaluate the quality of a solution sol. Let $s_{\{sol_i\}_{i \in [1,N]}}$ be a solution sample composed of N solutions for a same problem instance. Let c_s be a comment formulated by the user concerning the solution sample s.

We define the function $cst(S, f_{obj}, c_S)$ that determines for a solution sample s if the user comment c_s is consistent with the objective function f_{obj}. If the user comment c_sc is consistent, $cst(s, f_{obj}, c_s)$ is equal to 1, otherwise it is equal to 0. $cst(s, f_{obj}, c_s)$ is computed by Formulae 3:

$$cst(s, f_{obj}, c_s) = \begin{cases} 1 \; if \begin{cases} \begin{cases} c_s = \text{"}sol_A \; is \; good\text{"} \; and \\ f_{obj}(sol_A) \geq val^{good}_{min} \end{cases} \\ or \begin{cases} c_s = \text{"}sol_A \; is \; bad\text{"} \; and \\ f_{obj}(sol_A) \leq val^{bad}_{max} \end{cases} \\ or \begin{cases} c_s = \text{"}sol_A \; is \; far \; better \; than \; sol_B\text{"} \; and \\ val^{FB}_{min} \leq f_{obj}(sol_A) - f_{obj}(sol_B) \end{cases} \\ or \begin{cases} c_s = \text{"}sol_A \; is \; better \; than \; sol_B\text{"} \; and \\ val^{B}_{min} \leq f_{obj}(sol_A) - f_{obj}(sol_B) \leq val^{B}_{max} \end{cases} \\ or \begin{cases} c_s = \text{"}sol_A \; is \; slightly \; better \; than \; sol_B\text{"} \\ and \\ val^{SB}_{min} \leq f_{obj}(sol_A) - f_{obj}(sol_B) \leq val^{SB}_{max} \end{cases} \\ or \begin{cases} c_s = \text{"}sol_A \; and \; sol_B \; are \; equivalent\text{"} \; and \\ |f_{obj}(sol_A) - f_{obj}(sol_B)| \leq val^{Eq} \end{cases} \end{cases} \\ 0 \quad otherwise \end{cases} \tag{3}$$

This formula introduces some new parameters:

- Val^{good}_{min}: minimal quality value from which a solution can be considered as good.
- Val^{bad}_{min}: maximal quality value for a solution to be considered as bad.
- Val^{FB}_{min}: minimal difference quality value from which a solution is far better than another.
- Val^{B}_{min}: minimal difference quality value from which a solution is better than another.
- Val^{B}_{max}: maximal difference quality value for a solution to be better than another.
- Val^{SB}_{min}: minimal difference quality value from which a solution is slightly better than another.
- Val^{SB}_{max}: maximal difference quality value for a solution to be slightly better than another.
- Val^{Eq}: maximal difference quality value between two solution to be equivalent

These parameters confer a fuzzy aspect to the notion of compatibility and allow to make a link between the qualitative comments of the user and the numeric quality values. They have to be specifically defined for each application.

This global error function corresponds to the percentage of comments of the solution samples S that are inconsistent with the objective function f_{obj}. Let C_S be the set of comments contained in S. The global error is computed by the Formulae 4.

$$Error(f_{obj}, S) = \frac{100}{|C_S|} \cdot \sum_{c_s \in C_S} 1 - cst(s, f_{obj}, c_s) \tag{4}$$

The lower the $Error(f_{obj}, S)$ value, the better the objective function is. The goal is then to search the objective function parameter values that minimise this function. Because of the size of the search space that will usually be high, it is not possible to carry out a complete search. Thus, we propose to use a metaheuristic to find the best parameter values. In the literature, numerous metaheuristics were introduced [9, 7, 11, 8].

In order to facilitate the search process, we propose to take into account an initial objective function. Indeed, we make the hypothesis that, most of the time, experts can design a good objective function that can be a good start for the search process. In consequence, we propose to use a local search algorithm. The principle of this kind of algorithm is to start with an initial solution and to attempt to improve it by exploring its neighbourhood. These algorithms are usually very effective for this kind of search problem. There are numerous local search algorithms such as hill climbing, tabu search [8] or simulated annealing [11]. In the context of our objective function design process, the time spent by the search method is not a major issue. Hence, it is preferable to use methods which allow to avoid getting stuck in a local optimum, like the tabu search or the simulated annealing does, rather than a simple hill-climbing. Concerning the choice between these two methods, the experiments, we carried out, showed similar results.

Local search algorithms require the definition of the notions of 'solution neighbourhood'. We define the neighbourhood of a solution as the set of solutions for which only one parameter (the weight of a criterion w_i or the power p) has its value changed.

4 APPLICATION TO CARTOGRAPHIC GENERALISATION

4.1 Automatic cartographic generalisation

We propose to test our objective function design approach in the domain of cartographic generalisation. Cartographic generalisation is the process that aims at simplifying geographic data to suit the scale and purpose of a map. The objective of this process is to ensure the readability of the map while keeping the essential information of the initial data. As shown in Figure 2, the generalisation process is not a simple image reduction process but it requires the application of numerous operations such as geographic object displacement, scaling, removing, etc. to make them satisfy some legibility constraints.

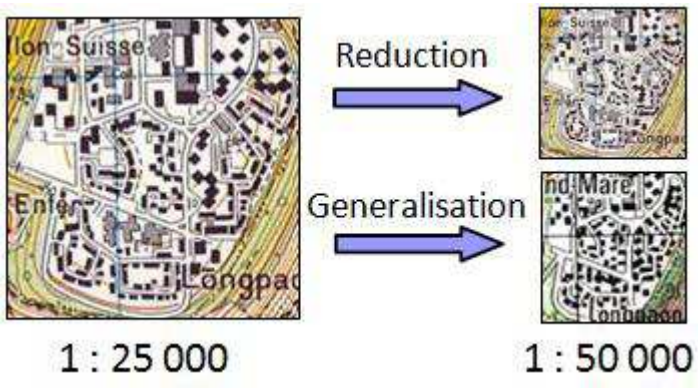

Figure 2. Example of cartographic generalisation

The automation of the generalisation process is an interesting industrial application context which is far from being solved. Moreover, it directly interests the mapping agencies that wish to improve their map production lines. At last, the multiplication of web sites allowing creating one's own map increases the needs of reliable and effective automatic generalisation processes.

One classic approach to automate the generalisation process is to use a local and step-by-step method [5, 17, 12]: each vector object of the database (representing a building, a road segment, etc.) is transformed by application of a sequence of generalisation algorithms realising atomic transformations. The choice of the applied sequence algorithms is not predetermined but built on the fly for each object according to heuristics and to its characteristics.

4.2 The generalisation system

The generalisation system that we used for our experiment is based on the well-establish AGENT model [3, 13]. In this model, geographic objects (roads, buildings, etc) are modelled as agents. Geographic agents manage their own generalisation, choosing and applying generalisation operations to themselves. Each state of the agent represents the geometric state of the considered geographic objects.

During its generalisation process, each agent is guided by a set of constraints that represents the specifications of the desired cartographic product. An example of constraint is, for a building agent, to be big enough to be readable. Each constraint has a satisfaction level between 0 (constraint not satisfy at all) and 100 (constraint perfectly satisfy). For each state, the agent computes its own satisfaction according to the values of its constraint satisfaction. To satisfy its constraints as well as possible, a geographical agent carries out a cycle of actions during which it tests different generalisation operations in order to reach a perfect state (where all of its constraints are perfectly satisfied) or at least the best possible state. The action cycle results in an informed exploration of a state tree. Each state represents the geometric state of the considered geographic objects. Figure 3 gives an example of a state tree obtained with the generalisation system.

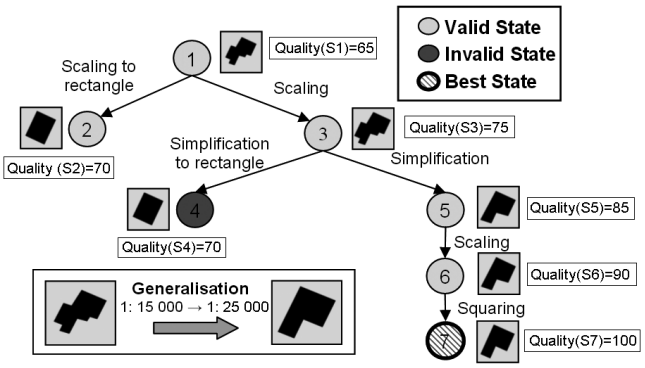

Figure 3. Example of a state tree for the generalisation of a building

4.3 Difficulties of the agent satisfaction function definition

The AGENT model has been the core of numerous research works and is used for map production in several mapping agencies. However, the question of the evaluation of the state of an agent is still an open issue. Indeed, designing such objective function when more than four constraints are in stake is often complex and fastidious [2]. Indeed, it requires finding a good balance between constraints that can be in opposition. For example, concerning building generalisation, the granularity constraint and the shape preservation constraints are in opposition: when the granularity constraints becomes more satisfied (less small details in the shape), the shape preservation constraints become less satisfied. Thus, having an approach like ours allowing to design the agent satisfaction function is particularly interesting.

4.4 Implementation of our approach for the AGENT model

We experiment our approach on an implementation of our user need definition module in Java, using the GeOxygene library for geographical data management [1]. Figure 4 presents our implemented interface. On the top panel, the initial state for a building is presented to the user, with, under, a sample of possible solutions. The expert can formulate, from the lower panel, comments for this sample.

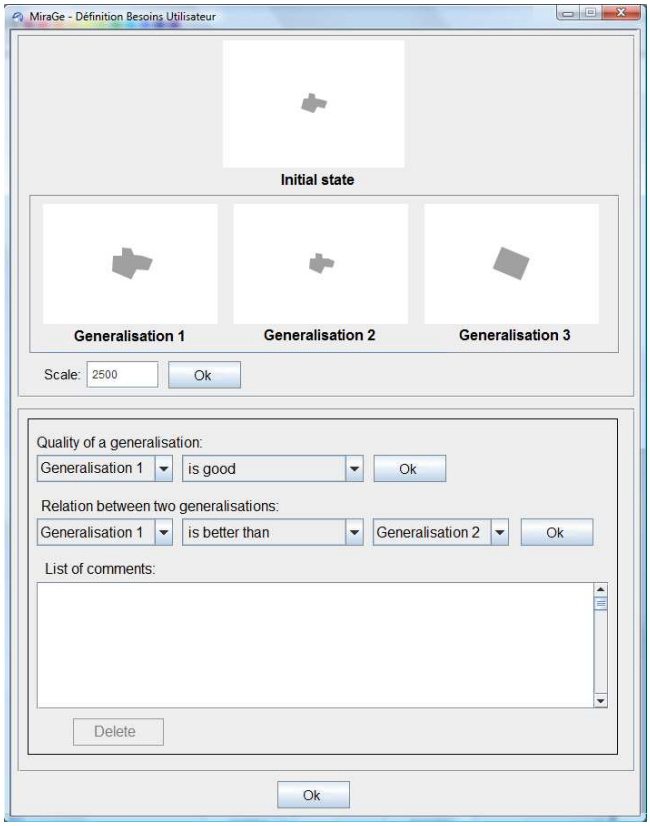

Figure 4. User comment acquisition interface

4.5 Case study

4.5.1 Setting of the case study

We propose to experiment our method for building generalisation for a traditional 1:25000 scale topographic map. The input data we used for the experiments are taken from the BDTopo, the one meter resolution topographic database produce by IGN, the French national mapping agency. We used five constraints for the building agents:

- The building size constraint (SC): this constraint incites a building to be big enough in order to be legible at the target scale.

- The building granularity constraint (GC): this constraint incites a building to have a simple shape: the building is transformed in order to delete its too short edges.
- The building squareness constraint QC): this constraint incites a building whose angles are almost orthogonal to have perfectly orthogonal angles.
- The building convexity constraint (CC): this constraint incites a building to preserve its convexity. Convexity is measured by the ratio of the building area and its convex hull area.
- The building elongation constraint (EC): this constraint incites the building to preserve its elongation. Elongation is, like convexity, a shape characteristic, which has to be preserved for the best.

4.5.2 Experimental protocol

In order to evaluate our approach, we build two sets of solution samples, both composed of 20 solution samples. Each sample consists in the comparison of 3 generalisations for a same building that were selected randomly among the available ones. These two solution sample sets were generated from different buildings, which were taken from the French cities of Bourg d'Oisans and L'Alpe d'Huez. For each of them, a generalisation expert formulated 100 comments. The first solution sample set, the $learning$ set, was used to revise the objective function. The second one, the $test$ set was used to evaluate the revised objective function.

In order to analyse the impact of the initial objective function quality, we tested our approach from two initial objective functions:

- *Basic objective function*: this function corresponds to the scenario where no knowledge is available concerning the evaluation of building generalisation. The objective function is a simple average of the constraint satisfaction values (p , and all constraint weights are equal to 1):

$$Sat(s) = \tfrac{1}{5}(Val_{SC}(sol) + Val_{GC}(sol) + Val_{QC}(sol) + Val_{CC}(sol) + Val_{EC}(sol)) \quad (5)$$

- *Expert evaluation function*: this function has been defined by a generalisation expert. The tuning of this function required a long process and demands the expert to know the generalisation system. For the need of 1:25000 topographic maps, the requirements concerning the building size and granularity is higher than the squaring, and other shape preservation constraints. The p parameter is used to give more importance to the constraint which is more satisfied. So, the following function is proposed:

$$Sat(s) = [\tfrac{1}{203}[(10 \cdot Val_{SC}(sol))^2 + (7 \cdot Val_{GC}(sol))^2 + (2 \cdot Val_{QC}(sol))^2 + (5 \cdot Val_{CC}(sol))^2 + (5 \cdot Val_{EC}(sol))^2]]^{\frac{1}{2}} \quad (6)$$

The parameter values used for the consistency computation were the following:

- Val_{min}^{good} = 80
- Val_{min}^{bad} = 65
- Val_{min}^{FB} = 15
- Val_{min}^{B} = 3
- Val_{max}^{B} = 30
- Val_{min}^{SB} = 1
- Val_{max}^{SB} = 20
- Val^{Eq} = 1

We used the simulated annealing [11] method to search the best parameter values.

4.5.3 Results and discussion

The objective function obtained after revising the $Basic$ objective function is presented Equation 7.The one obtained after revising the $Expert$ objective function is presented Equation 8.

$$Sat(s) = [\tfrac{1}{40819}[(8 \cdot Val_{SC}(sol))^5 + (2 \cdot Val_{GC}(sol))^5 + (6 \cdot Val_{QC}(sol))^5 + (0 \cdot Val_{CC}(sol))^5 + (3 \cdot Val_{EC}(sol))^5]]^{\frac{1}{5}} \quad (7)$$

$$Sat(s) = [\tfrac{1}{41570}[(8 \cdot Val_{SC}(sol))^5 + (4 \cdot Val_{GC}(sol))^5 + (6 \cdot Val_{QC}(sol))^5 + (1 \cdot Val_{CC}(sol))^5 + (1 \cdot Val_{EC}(sol))^5]]^{\frac{1}{5}} \quad (8)$$

Table 1 presents the results obtained on the two sample sets. The percentage value is given by the global error function defined Equation 3.4. It represents the part of comments formulated by the expert, which are not consistent with the assessment of the objective function. These values have to be as small as possible.

Table 1. Global error rate on the *learning* and *test* sets

	Basic	*Expert*	*Basic revised*	*Expert revised*
learning set	0.46	0.32	0.25	**0.24**
test set	0.45	0.32	0.26	**0.25**

First, we can notice the difficulties of defining a good objective function for an expert. Indeed, even with a good command of the AGENT model, the generalisation expert did not succeed in designing an objective function that perfectly translates his expectations toward the generalisation results.

Concerning the result obtained after revision, we can observe that the two revised objective functions are significantly better than the initial ones, both on the $learning$ and $test$ sets. We can also observe that the objective function obtained after revising the $Expert$ objective function is slightly better than the one obtained after revising the $Basic$ objective function. Indeed, the revised $Expert$ objective function keeps some of the specificity of the initial function, in particular concerning the high weight of the granularity constraint. This last observation confirms the interest of taking into account an initial objective function.

If the result obtained with the revised objective functions are better than ones obtained with the initial objective functions, the results are not perfect. Actually the global error rate is still high (0.24-0.25). This high error rate can be explained by two main reasons.

The first one concerns the lack of pertinent measures to describe the generalisation results. For example, when a comparison composed of two building generalisations, which differ only in term of orientation is shown, the user always prefers the one whose orientation is close to the building initial orientation. Because there is no orientation constraint taken into account into the evaluation function, the difference of the two generalisations can not be measured by the system, and the reason of the different assessment by the user remains ignored. Thus, adding pertinent constraint such as one assessing the building orientation could help to reduce the error rate.

The second reason comes from the formalism used to represent the objective function: a weighted means balanced by a power. This function has for advantage to be very simple and easily readable. However, it does not allow to express the discontinuity of some criteria concerning their contribution to the global solution quality. For example, light squaring problems are insignificant for the global generalisation quality, but if these problems are more serious, the generalisation quality can be very poor. Thus, extending the formalism we used to represent the objective functions in order to take into account these discontinuities can help to design better objective functions.

5 CONCLUSION

In this paper, we presented an approach dedicated to the revision of objective functions. This approach is based on a man-machine dialogue through the presentation of solution samples to an expert and the collection of his comments. An experiment, carried out in the domain of cartographic generalisation, showed that our approach can allow to significantly improve existing objective functions.

Our approach is based on the presentation of solution samples to an expert. The choice of the sample presented to the expert at each iteration can have a deep impact on the data collected and thus on the revision results. In our experiments, the presented samples were chosen randomly. More complex strategies could be defined to present more interesting samples to the user. These strategies could take into account the comments already formulated by the expert.

As mentioned in Section 4.5.3, another interesting perspective could consist in extending the formalism used to define the objective function in order to take into account the discontinuity of the criterion contributions. In this context, the works carried out in the domain of multi-criteria decision making (e.g. [4]) could be used as a base.

REFERENCES

[1] T. Badard and A. Braun, 'Oxygene: An open framework for the deployment of geographic web services', in *21st International Cartographic Conference (ACI/ICA)*, (2003).

[2] S. Bard, 'Quality assessment of cartographic generalisation', *Transaction in GIS*, **8**, 63–81, (2004).

[3] M. Barrault, N. Regnauld, C. Duchêne, K. Haire, C. Baeijs, Y. Demazeau, P. Hardy, W. Mackaness, A. Ruas, and R. Weibel, 'Integrating multi-agent, object-oriented, and algorithmic techniques for improved automated map generalization', in *ICC*, volume 3, pp. 2110–2116, (2001).

[4] J.P. Brans and B. Mareschal, 'Promethee methods', in *Multiple Criteria Decision Analysis: State of the Art Surveys*, Springer, (2005).

[5] K. Brassel and R. Weibel, 'A review and conceptual framework of automated map generalisation', *IJGIS*, **2**(3), (1988).

[6] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, 'Applications of machine learning to cognitive radio networks', *Wireless Communications*, **4**, 47–52, (2007).

[7] M. Dorigo and T. Sttzle, *Ant Colony Optimization*, MIT Press, 2004.

[8] F. Glover, 'Tabu search', *Journal on Computing*, (1989).

[9] J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[10] S. Kakade, Y.W. Teh, and S. Roweis, 'An alternative objective function for markovian fields', in *19th ICML*, (2002).

[11] S. Kirkpatrick, C.D. Gellatt, and Vecchi M.P., 'Optimization by simulated annealing', *Science*, **220**, 671–680, (1983).

[12] M. Neun, D. Burghardt, and R. Weibel, 'Automated processing for map generalization using web services', *GeoInformatica*, **13(4)**, 425–452, (2009).

[13] A. Ruas and C. Duchêne, 'A prototype generalisation system based on the multi-agent system paradigm', in *Generalisation of Geographic information: cartographic modelling and applications*, eds., W.A. Mackaness, A. Ruas, and L.T. Sarjakoski, chapter 14, 269–284, Elsevier Ltd, (2007).

[14] S. Russell and P. Norvig, *Artificial intelligence: a modern approach*, Prentice-Hall, 1995.

[15] P. Taillandier, C. Duchêne, and A. Drogoul, 'Knowledge revision in systems based on an informed tree search strategy: application to cartographic generalisation', in *International Conference on Soft Computing as Transdisciplinary Science and Technology*, (2008).

[16] P. Taillandier and J. Gaffuri, 'Objective function designing led by user preferences acquisition', in *International Conference on Information Technology and Applications (ICITA)*, (2009).

[17] J. M. Ware and C. B. Jones, 'Conflict reduction in map using iterative improvement', *GeoInformatica*, **2**(4), 383–407, (1998).

[18] M. Wimmer, F. Stulp, S. Pietzsch, and B. Radig, 'Learning local objective functions for robust face model fitting', *IEEE Transactions on Pattern Analysis andMachine Intelligence*, **30(8)**, (2008).