Using human-machine dialogue to refine generalisation evaluation function

Patrick Taillandier, Julien Gaffuri

Using human-machine dialogue to refine generalisation evaluation function

Patrick Taillandier[1,2] and Julien Gaffuri[3]

[1] IRD, UMI UMMISCO 209,
32 avenue Henri Varagnat, 93143 Bondy, France
[2] IFI, MSI, UMI 209,
ngo 42 Ta Quang Buu, Ha Noi, Viet Nam
[3] IGN – COGIT laboratory – Paris-Est University
73 avenue de Paris, 94165 Saint-Mandé cedex, France
patrick.taillandier@gmail.com
julien.gaffuri@ign.fr

1. Introduction

More and more geographical data producers use automated generalisation to produce their data. Indeed, the spreading of artificial intelligence techniques has allowed an important improvement of the generalisation process automation.

A classic approach consists in formalising generalisation as an optimisation problem: the goal is to find a state of the data that maximises a function. [2], [7] and [13] present adaptations to generalisation of the finite elements method, whose purpose is to minimise an energy function. [22] use simulated annealing to find a global optimum of a utility function. By using genetic algorithms, [9] and [23] propose to maximise a fitness function. All these methods use a function that is supposed to assess the generalisation state of the data, according to the user need. We propose to call this function "evaluation function" (the expressions "utility function", "fitness function", "energy" or even "satisfaction" are also often used). A key issue of this approach concerns the design of this evaluation function. Indeed, in order to get good results, such systems have to know what it is searching, i.e. what a good generalisation of the input data is. Unfortunately, designing such a function remains a difficult task. Indeed, while the final user of the generalised data can easily describe his need in natural language, it is often far more difficult for him to express his expectations in a formal language that can be used by generalisation systems. This problem is particularly complex when numerous measures are used to characterise the quality of a generalisation and when no simple links between these measures values and the solution quality can be found.

In this paper, we propose an approach dedicated to the design of generalisation evaluation functions. An evaluation function previously designed by a user is improved through a dialogue between this user and the generalisation system. The idea is to collect user preferences by letting the user compare different generalisation results for a same object (or group of objects).

In Section 2, the general context of our work is introduced. Section 3 is devoted to the presentation of our approach. Section 4 describes an experiment carried out for the building generalisation. Section 5 concludes and presents the perspectives of this work.

2. Context

2.1. Automated evaluation of generalisation results

If many works focus on the generalisation process automation, only a few deal with the problem of the automatic evaluation of generalisation outcomes.

A classic approach used by many generalisation systems consists in evaluating the generalisation quality by means of a set of constraints translating the expectation towards the generalisation [5]. The constraint assessment is often represented by a numeric satisfaction value. The overall generalisation is evaluated by aggregating all the constraint satisfaction values. If the computation of individual constraint satisfaction values is often well-managed, the definition of the aggregating function remains complex [19]. 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, the preservation constraints become less satisfied. In this work, we are interested in the definition of this aggregation function.

Among other works that deal with automatic generalisation evaluation, [3] proposes a generalisation evaluation method based on the definition of functions translating the expected evolution of geographic objects during generalisation. These functions are then aggregated in order to get a global quality value of the generalised data.

In the context of an EuroSDR project, that concerns the evaluation of generalisation systems, [18] propose to use both manual (expert) and automatic evaluation to assess the quality of generalised data. Concerning the automatic evaluation, the used approach is based on the evaluation of a set of constraints. In the context of this project, the goal was to evaluate a complete generalised dataset, while we rather propose to focus on the individual generalisation quality of objects or groups of objects.

2.2. Design of an evaluation function

The evaluation function design is a complex problem which was studied in various fields. Indeed, the definition of such function is a key point of the resolution of optimisation problems [17], [20]. Many works were interested in the definition of these functions for specific problems [13] [24] but few proposed general approaches for helping optimisation systems users to define it.

A classic approach to solve this problem consists in using supervised machine learning techniques. These techniques consist in inducing a general model from examples labelled by an expert. In this context, it is possible to learn an evaluation function from examples assessed by an expert. This approach was used in several works, like [24] in the domain of computer vision, and [8] for the learning of cognitive radio. To be effective, this approach requires that an expert is able to give a quantitative evaluation of the different examples, which can be difficult when many criteria can be used to evaluate the examples.

2.3. Formalisation of the evaluation function design

Our work aims at providing a method to help users to design an evaluation function. We assume that a set of constraints is defined that translate the expectation toward the generalised data. We assume as well that the constraints assessment is represented by a numeric satisfaction value. The higher this value, the more satisfied the constraint is, thus better the generalisation is. The evaluation function design consists then in defining an aggregating function that allows the generalisation quality to be assessed by a single value. A key point of our problem is to determine a satisfying model for the aggregating function. We propose to formulate this aggregating function by a weighted means balanced by a power.

Let C be the set of constraints considered, w_i the weight associated to a constraint i, $Val_i(gen)$, the satisfaction value of the constraint i for the generalisation gen, and p, an integer higher or equal to 1. An evaluation function is then defined as follows:

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

When p is equal to 1, the aggregation is a simple weighted average of the constraints satisfactions. The role of the p parameter is to control the relative weight of the most satisfied constraints over the less satisfied ones: the higher is p, the more satisfied constraints are taken into account in the overall quality of the object. An interest of such a representation 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. The level of interpretation of results is often considered in generalisation systems design [16].

3. Proposed approach

3.1. General approach

It is often difficult for experts to express their outcome expectation toward a generalisation system in a formal way. In this context, we state the hypothesis that it is easier to comment generalisation results in a qualitative way. Thus, we propose to base our objective function design approach on the collection of user comments toward generalisation results. This approach is close to the one proposed by [12] concerning the parameterisation of the generalisation process. However, a difference is that the user will not just select his/her preferred generalisation among a set, but he/she will compare these generalisations. Each comparison is composed of two generalisations of the same object (or group of objects). The user can give his/her preferences toward these two generalisations, i.e. if a generalisation is far better or slightly better than the other one, etc. The system then automatically builds the evaluation function from the collected preference data.

Our approach is composed of 3 steps (Figure 1): the first one consists in generating a set of generalisation samples. Each sample is composed of different generalisations for a same object (or group of objects). 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 refine the evaluation function. The following sections describe these three steps.

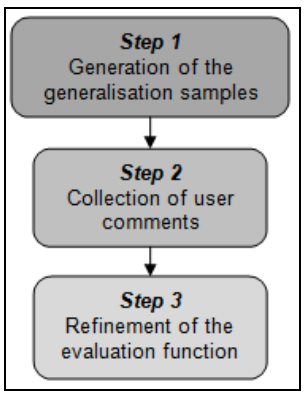

Figure 1. General approach

3.2. Building of generalisation samples

The first step concerns the building of the generalisation samples that will be shown to the user to capture his needs. Each sample is composed of *n* generalisations of the same object (or groups of objects).

In order to build the generalisation samples, different objects (or groups of objects) have to be generalised. The number of generalised objects depends on the application context of our approach. In some contexts, it will only be possible to generalise few objects due to the availability of the objects or to time constraint. Thus, for some application context, a sampling method such as the one presented in [21] will have to be used in order to select a representative subset of objects.

We assume that the generalisation of the objects (or a group of objects) will lead to the generation of at least *n* possible generalisations. These possible generalisations are used to build a generalisation sample.

3.3. Capture of the user preferences

The second step concerns the collection of user comments. A set of generalisation 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 generalisation.
- Comments linked to the preference of a generalisation over another one.

Concerning the first kind of comments, we defined two levels of quality. Let *A* be a generalisation. 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 generalisations. The four possible preference relations are:

- Generalisation *A* is far better than generalisation *B*.
- Generalisation *A* is better than generalisation *B*.
- Generalisation *A* is slightly better than generalisation *B*.
- Generalisation *A* and *B* are equivalent.

Figure 2 presents the comparison interface of the prototype we developed.

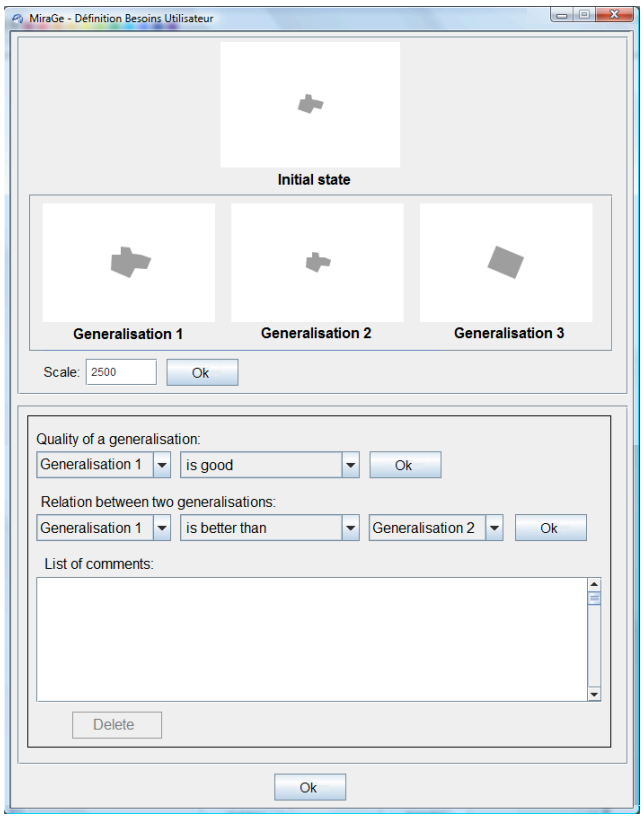

Figure 2. The user preference capture interface implemented

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

3.4. Refinement of the evaluation function

This last step consists in refining the evaluation function from the user comments collected in the previous step. Refining the evaluation function means finding, from the existing evaluation 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 evaluation 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 $f_{eval}(gen)$ be the current evaluation function used to evaluate the quality of a generalisation *gen*.

Let $s_{\{gen_i\}_{i\in[1,N]}}$ be a generalisation sample composed of N generalisations for a same object (or group of objects).

Let c_s be a comment formulated by the user concerning the generalisation sample s.

We define the function *comp(s, f_{eval}, c_s)* that determines for a generalisation sample *s* if the user comment c_s is compatible with the evaluation function f_{eval}. If the user comment c_s is compatible with f_{eval}, *comp(s, f_{eval}, c_s)* is equal to *1*, otherwise it is equal to *0*. *comp(s, f_{eval}, c_s)* is computed by the following formula:

$$comp(s, f_{eval}, c_s) = \begin{cases} 1 \; if \begin{cases} \text{or} \begin{cases} c_s = \text{"}gen_A \; is \; good\text{"} \; and \\ f_{eval}(gen_A) \geq val_{min}^{good} \end{cases} \\ \text{or} \begin{cases} c_s = \text{"}gen_A \; is \; bad\text{"} \; and \\ f_{eval}(gen_A) \leq val_{max}^{bad} \end{cases} \\ \text{or} \begin{cases} c_s = \text{"}gen_A \; is \; far \; better \; than \; gen_B\text{"} \; and \\ val_{min}^{FB} \leq f_{eval}(gen_A) - f_{eval}(gen_B) \end{cases} \\ \text{or} \begin{cases} c_s = \text{"}gen_A \; is \; better \; than \; gen_B\text{"} \; and \\ val_{min}^{B} \leq f_{eval}(gen_A) - f_{eval}(gen_B) \leq val_{max}^{B} \end{cases} \\ \text{or} \begin{cases} c_s = \text{"}gen_A \; is \; slightly \; better \; than \; gen_B\text{"} \\ and \\ val_{min}^{SB} \leq f_{eval}(gen_A) - f_{eval}(gen_B) \leq val_{max}^{SB} \end{cases} \\ \text{or} \begin{cases} c_s = \text{"}gen_A \; and \; gen_B \; are \; equivalent\text{"} \; and \\ |f_{eval}(gen_A) - f_{eval}(gen_B)| \leq val^{Eq} \end{cases} \end{cases} \\ 0 \quad otherwise \end{cases}$$

This formula introduces 8 new parameters:

- $val_{\min}^{good}$: minimal quality value from which a generalisation can be considered as good.
- $val_{\max}^{bad}$: maximal quality value for a generalisation to be considered as bad.
- $val_{\min}^{FB}$: minimal difference quality value from which a generalisation is far better than another one.
- $val_{\min}^{B}$: minimal difference quality value from which a generalisation is better than another one.
- $val_{\max}^{B}$: maximal difference quality value for a generalisation to be better than another one.
- $val_{\min}^{SB}$: minimal difference quality value from which a generalisation is slightly better than another one.
- $val_{\max}^{SB}$: maximal difference quality value for a generalisation to be slightly better than another one.
- val^{Eq}: maximal difference quality value between two generalisations 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.

The global error function corresponds to the percentage of comments of the set of generalisation samples *S* that are incompatible with the evaluation function f_{eval}. Let C_S be the

set of comments contained in *S*. The global error is computed by the following formula:

$$Error(f_{eval}, S) = \frac{100}{|C_S|} \times \sum_{c_s \in C_S} 1 - comp(s, f_{eval}, c_s)$$

The lower the $Error(f_{eval}, S)$ value, the better the evaluation function is. The goal of this step is then to search the evaluation function parameter values that minimise this error 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 [10][11][15].

In this work, we search to refine an existing evaluation function and to not learn one from scratch. Indeed, we make the hypothesis that, most of the time, experts can design a good evaluation 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 [10] or simulated annealing [15]. In the context of our evaluation 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. Case study: evaluation of building generalisation

4.1. Context

4.1.1. The generalisation system

Our experiment use a generalisation system based on the AGENT model [4][16] and the GéOxygene Java library [1]. In this model, geographic objects (roads, buildings, etc) are modelled as agents. These geographic agents manage their own generalisation by 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. Each constraint has a satisfaction level between 0 (constraint not satisfy at all) and 100 (constraint perfectly satisfy).

To satisfy its constraints as well as possible, a geographical agent carries out a cycle of actions during which it tests different actions 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. Figure 3 gives an example of a state tree obtained with the generalisation system for a building.

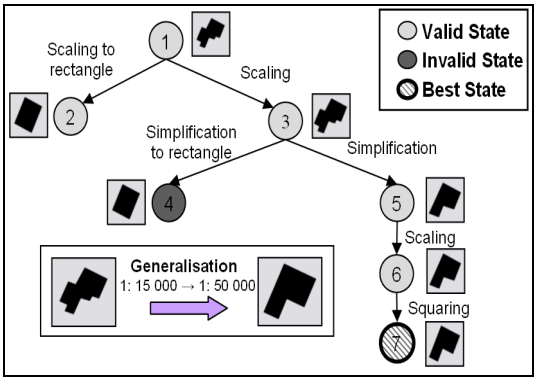

Figure 3. Example of a state tree build for a building generalisation

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 asked. The definition of the weight values is often complex and fastidious when more than five constraints are in stake [3]. Thus, having an approach like the one we proposed is particularly interesting.

4.1.2. Building generalisation

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 use five constraints for the building agents:

- *Building size constraint (SC):* this constraint incites a building to be big enough in order to be legible at the target scale.
- *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.
- *Building squareness constraint (QC):* this constraint incites a building whose angles are almost orthogonal to have perfectly orthogonal angles.
- *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.
- *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.

4.2. Experimental protocol

In order to evaluate our approach, we build two sets of generalisation samples, both composed of 20 generalisation samples. Each sample consists in the comparison of 3 generalisations for a same building that were selected randomly among the available ones. These two generalisation 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 generalisation sample set, the *learning* set, was used to revise the evaluation functions. The second one, the *test* set was used to evaluate the refined evaluation function.

In order to analyse the impact of the initial objective function quality, we tested our approach with two initial evaluation 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*):

$$Sat(gen) = \frac{1}{5}\left(Val_{SC}(gen) + Val_{GC}(gen) + Val_{QC}(gen) + Val_{CC}(gen) + Val_{EC}(gen)\right)$$

- *Expert evaluation function*: this function has been defined by a generalisation expert. The tuning of this function had required a long process and knowledge about 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 constraints that are more satisfied. So, the following function is proposed:

$$Sat(gen) = \sqrt{\frac{1}{203}\left((10 \cdot Val_{SC}(gen))^2 + (7 \cdot Val_{GC}(gen))^2 + (2 \cdot Val_{QC}(gen))^2 + (5 \cdot Val_{CC}(gen))^2 + (5 \cdot Val_{EC}(gen))^2\right)}$$

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

- $val_{\min}^{good} = 80$
- $val_{\max}^{bad} = 65$
- $val_{\min}^{FB} = 15$
- $val_{\min}^{B} = 3$
- $val_{\max}^{B} = 30$
- $val_{\min}^{SB} = 1$
- $val_{\max}^{SB} = 20$
- $val^{Eq} = 1$

We used the simulated annealing [15] method to search the best parameter values.

4.3. Results and discussion

The objective function obtained after refining the *Basic* evaluation function is the following:

$$Sat(gen) = \left(\frac{1}{40819}\left((8 \cdot Val_{SC}(gen))^5 + (2 \cdot Val_{GC}(gen))^5 + (6 \cdot Val_{QC}(gen))^5 + (0 \cdot Val_{CC}(gen))^5 + (3 \cdot Val_{EC}(gen))^5\right)\right)^{\frac{1}{5}}$$

The one obtained after revising the *Expert* evaluation function is the following:

$$Sat(gen) = \left(\frac{1}{41570}\left((8 \cdot Val_{SC}(gen))^5 + (4 \cdot Val_{GC}(gen))^5 + (6 \cdot Val_{QC}(gen))^5 + (1 \cdot Val_{CC}(gen))^5 + (1 \cdot Val_{EC}(gen))^5\right)\right)^{\frac{1}{5}}$$

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

	Global error			
	Basic evaluation function		*Expert evaluation function*	
	Initial function	*Refined function*	*Initial function*	*Refined function*
Learning set	0.46	0.25	0.32	**0.24**
Test set	0.45	0.26	0.32	**0.25**

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

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

Concerning the result obtained after revision, we can observe that the two refined evaluation functions are significantly better than the initial ones, both on the *learning* and *test* sets. We can also observe that the evaluation function obtained after refining the *Expert* evaluation function is slightly better than the one obtained after refining the *Basic* evaluation function. Indeed, the refined *Expert* evaluation 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 evaluation function.

If the result obtained with the refined evaluation functions are better than ones obtained with the initial evaluation 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 evaluation 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 evaluation functions in order to take into account these discontinuities can help to design better evaluation functions.

5. Conclusion

In this paper, we presented an approach dedicated to the refinement of generalisation evaluation functions. We proposed a method based on a human-machine dialogue and the capture of user preferences on samples of generalisation results. An experiment, carried out for building generalisation, showed that our approach can help users to improve their

evaluation functions.

Our approach is based on the presentation of generalisation 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 refinement 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 4.3, another interesting perspective could consist in extending the formalism used to define the evaluation 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. [6]) could be used as a base.

References

1. Badard, T., Braun, A.: OXYGENE: An open framework for the deployment of geographic web services. International Cartographic Conference, Durban, South Africa, pp 994-1004 (2003)
2. Bader, M.: Energy minimization methods for feature displacement in map generalisation. Ph.D. thesis, Zurich university, Department of geography (2001)
3. Bard S.: Quality Assessment of Cartographic Generalisation, Transactions in GIS, 8, pp. 63-81 (2004)
4. Barrault, M., Regnauld, N., Duchêne, C., Haire, K., Baeijs, C., Demazeau, Y., Hardy, P., Mackaness, W., Ruas, A., Weibel, R.: Integrating multi-agent, object-oriented, and algorithmic techniques for improved automated map generalization. International Cartographic Conference, pp. 2110--2116, (2001)
5. Beard, K.: Constraints on rule formation. In Buttenfield, B. & Mcmaster, R. (eds.) Map generalisation: making rules for knowledge representation, 121-135 (Longman Scientific and Technical, 1991).
6. Brans, J.P., Mareschal, B.: 'Promethee methods', in *Multiple Criteria Decision Analysis: State of the Art Surveys*, Springer (2005).
7. Burghardt, D., Meier, S.: Cartographic displacement using the snakes concept. In: W. Foerstner & L. Pluemer (eds.), Semantic modelling for the acquisition of topographic information from images and maps. Birkhaeuser verlag, Basel (1997)
8. Clancy, C., Hecker, J., Stuntebeck, E., O'Shea, T.: Applications of Machine Learning to Cognitive Radio Networks, Wireless Communications, IEEE, vol. 14, no. 4, pp. 47-52 (2007)
9. Deng, H., Fang, W., Yang, O., Li, Y.: A model of road network generalisation based on genetic algorithms. International Cartographic Conference, Durban, South Africa, (2003).
10. Glover, F.: Tabu search. Journal on Computing (1989)
11. Holland, J.H.: Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press (1975)
12. Hubert, F., Ruas, A.: A method based on samples to capture user needs for generalisation, Workshop on progress in automated map generalisation, Paris (2003).

13. Højholt, P.: Solving local and global space conflicts in map generalisation using a finite element method adapted from structural mechanics. SDH'98, pp. 679-689 (1998)

14. Kakade, S., Teh, Y.W., Roweis, S.: An alternative objective function for Markovian fields. In Proc. ICML (2002)

15. Kirkpatrick, S., Gellatt, C., Vecchi, M.P.: Optimization by Simulated Annealing, Science 220, pp.671—680 (1983)

16. Ruas, A., Duchêne, C.: A prototype generalisation system based on the multi-agent system paradigm. In: Mackaness, W.A., Ruas, A., Sarjakoski, L.T. (ed.), Generalisation of Geographic information: cartographic modelling and applications, Elsevier Ltd, pp. 269—284 (2007)

17. Russel, S., Norvig, P.: Artificial intelligence: a modern approach Prentice-Hall (1995)

18. Stoter, J., Burghardt, D., Duchêne, C., Baella, B., Bakker, N., Blok, C., Pla, M., Regnauld, N., Touya, G., Schmid, S.: Methodology for evaluating automated map generalization in commercial software, Computers, Environment and Urban Systems, vol.33, n°5, pp 311-324 (2009)

19. Taillandier, P.: Diagnosis in systems based on an informed tree search strategy: application to cartographic generalisation, CSTST Student workshop, Cergy-Pontoise, France (2008)

20. Taillandier, P., Duchêne, C., Drogoul, A.: Knowledge revision in systems based on an informed tree search strategy: application to cartographic generalisation. CSTST pp. 273--278, Paris (2008)

21. Taillandier, P., Gaffuri, J.: Automatic Sampling of Geographic objects. *GIScience*, Zurich, Switzerland (2010).

22. Ware, M. J., Jones, C. B.: Conflict Reduction in Map Generalization Using Iterative Improvement. GeoInformatica 2(4):383-407 (1998)

23. Wilson, I. D., Ware, J. M., Ware, J. A.: Reducing graphic conflict in scale reduced maps using a genetic algorithm. Workshop on progress in automated map generalisation, commission on map generalisation, Paris, France (2003)

24. Wimmer, M., Stulp, F., Pietzsch, S., Radig, B.: Learning local objective functions for robust face model fitting. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 30(8) (2008)