'GENERALIZATION BY EXAMPLE': INTERACTIVE PARAMETER CONTROL IN LINE GENERALIZATION USING GENETIC ALGORITHMS

Stefan F. Keller

Department of Geography, University of Zurich
Winterthurerstrasse 190, CH-8057 Zurich (Switzerland)
Phone: +41-1-257 51 51 / Fax: +41-1-362 52 27
E-mail: keller@gis.geogr.unizh.ch

Abstract

The selection of generalization operators and the respective parameters is one of the most difficult parts automating the digital map making process. Existing generalization tools support direct manipulation and graphical user interfaces, but the users still have no control of the operators at a task oriented level. To select the appropriate parameters they have to communicate with the system in a rather indirect way while setting numerically the parameter values. It is argued, that because of the complexity of the domain automated generalization tools need to be designed as interactive, cooperative, learning and adaptive systems.

Investigation on the use of computational intelligence techniques embedded in interactive systems lead to the idea of 'generalization by example': this approach provides new options for interaction at the adequate conceptual level and at the same time it delivers semi-formal knowledge through interaction logging.

An implementation is laid out based on this approach using genetic algorithms (GA) for optimization tasks. The goal is to find the best parameters for line simplification algorithms (Douglas-Peucker and Lang). First, the user draws a 'sample line'. Then, the GA tries to find the optimal parameters for the best approximation of the generated output compared to this sample line. Structure measures evaluate the output. If the user accepts the found parameters they will be used to generalize the entire line work. It is shown that the results lie in a reasonable range but that there are more tests to carry out for proving the feasibility of this approach.

1 Introduction

One of the crucial success factors of cartographic or geographical information systems (GIS) is how economically the spatial database is being captured and updated. In traditional cartography the maintenance of a single spatial master database and derivation of smaller scale map series are well known concepts and constitute the core skills of cartographers.

With the maturing of GIS more and more projects are in transition from data capturing to maintenance phase and try to multiplicate the efforts. In the digital environment an effective way to achieve this is providing primary and value-added products to as many users as possible. These products are mostly communicated in graphical form, therefore functions for *cartographic generalization* are needed. As a complementary part, functions for *model generalization* would also become necessary to reduce and filter the huge amount of accumulated data. Now, the users are requesting more update and generalization functions in GIS. Consequently, for the automation of cartographic and model generalization several tools are under development and research initiatives have been established [15, 16]. Update techniques could become another area of interest in the future [7].

Generalization is the process to derive graphic target products from a source database at smaller scale while maintaining good readability and keeping the most important information according to the

intentions of the map author. This paper focuses on cartographic generalization (for short, generalization). It is a complex problem for many reasons explained below.

Past research in generalization concentrated on algorithmic methods but, as it turned out, they only offered partial solutions to narrowly defined deterministic problems [18]. Thus, holistic approaches were sought including more built-in knowledge. Therefore research interest shifted towards artificial intelligence techniques, more specifically to expert systems and knowledge-based systems. Because of the lack of experience in this field (regarding implementation and suitability of the new techniques) most of these systems could not fulfill the expectations. The main reasons for the failures have been recognized, namely, the lack of knowledge and the type of complexity of the domain:

First, knowledge acquisition is difficult because the cartographic knowledge is very implicit (dubbed 'the knowledge acquisition bottleneck'). It is analogically and graphically encoded, and it is hard to access because intuition is involved and there are no knowledge sources readily available, like human experts, text documents, maps and process log files [19]. In generalization we are looking especially for procedural knowledge (like the optimal sequence of operators in a specific situation) but it is known to be hard to elicit compared to static descriptive knowledge.

Second, it is a complex domain because it is difficult to define objective goals (and evaluation criteria). It is ill-structured, semi-formal and involves analytical, comparative, and constructive processes. Additionally, there exists an infinite diversity of real-world objects to be represented. Therefore, control that involves procedural knowledge, becomes an important issue.

Consequently, any progress in generalization needs to cope with the problems of knowledge acquisition, knowledge implementation (data modeling and representation) and program control (user interaction). These are forming the three building blocks of 'knowledge systems' which is a broader term, including expert systems, as described in Keller [8]. Current research in generalization can also be partitioned into similar groups. More specifically, the following core problems are in context with the generalization problem described here:

1. *Knowledge acquisition*: Investigation in knowledge acquisition is needed especially for procedural (control) and structural knowledge (evaluation). To overcome the scarcity even of raw information process tracing or re-engineering techniques are currently employed (see [17,13]). Additionally, the generalization process needs to be evaluated at some level and will constitute a key function in the whole design of future implementations [20,15].
2. *Knowledge implementation*: For supporting a comprehensive approach the underlying data models need to be reviewed and new data structures engineered [10]. Subsequently, this could lead to advanced algorithms based on existing ones like, for example, Voronoi diagrams which can be used for feature displacement. Knowledge representation plays also an important role [10,8].
3. *Control and reasoning*: Interaction and decision support became a major issue because most of the current implementations of generalization tools are benefiting from graphical user interfaces based on the concepts of 'amplified intelligence' [18]. Here, an extension to the artificial intelligence approach was proposed with systems assisting the user's decision making process and keeping him in the man-machine interaction loop.

In this paper the idea of using prototypical examples for guiding generalization - like in case-based reasoning discussed below - is pursued further, leading to an approach which we termed 'generalization by example' (or 'generalization by analogy').

In the following section an overview of the potential of 'computational intelligence' (CI) techniques is given and the basic reasons for using interactive systems are explained. Then, we describe a preliminary implementation of a tool based on the principle of 'generalization by example' to interactively set the parameters of line simplification algorithms. Finally, preliminary results are discussed.

2 Computational intelligence and interactive systems

This implementation of 'generalization by example' is designed on the basis of CI techniques embedded in interactive systems. We strongly believe that only the integration of classical programming techniques extended with CI models will succeed in building complex systems. Additionally, they have to be shaped with human-centered software development using incremental prototyping.

2.1 Computational intelligence in generalization

We want to emphasize that we look at artificial intelligence from an engineering point of view. Thus, we prefer the term 'computational intelligence' to include techniques like artificial neural nets (NN), evolutionary strategies and other computational methods that exhibit the most abstract sort of intelligence in order to support human tasks. The application of CI methods to generalization is a relatively new research topic, therefore not many references exist. The potentials and limitations of artificial intelligence techniques have been assessed by Keller [9] and Weibel et al. [19]. The field has been classified in many ways, one was the distinction between symbolic (e.g. knowledge-based systems) and subsymbolic systems (e.g. NN). Machine learning is a main component of artificial intelligence. Carbonell [2] divided machine learning into four different types:

1. *Inductive learning* was identified to have some potential for knowledge acquisition, as described in Weibel [20]. In learning through induction positive and negative examples are being used to induce more general concepts. After completion, the examples are not needed anymore.
2. *Analytical learning and reasoning by analogy* introduced recently a major extension to the expert system paradigm: it was termed 'case-based reasoning' (CBR). As CBR says, it is not only a machine learning paradigm but also a particular reasoning method. Learning occurs here as a natural 'by-product': when a problem is successfully solved, it is mapped to a case-template and saved to a case-base in order to solve a similar problem more efficiently next time. Therefore, in contrast to inductive and connectionist learning, in CBR the 'examples' (cases) are being stored. Keller [8] reviewed the use of CBR to solve generalization problems, and defined the prerequisites and a preliminary design. The idea of using CBR in generalization was induced by the fact that it fits well into the work and education style in traditional cartography. Texts such as in Imhof [6] are full of 'good' and 'poor' examples of generalization or other design processes. Cartographers then attempt to follow analogically and as closely as possible the prototypical cases.
3. Armstrong [1] used *genetic algorithms* (GA) to 'generate-and-test' generalization alternatives in order to empirically find near-optimal solutions.
4. The use of *connectionist learning* or neural nets was examined by Muller [14], while Werschlein and Weibel [21] are trying to look for correlation classes in line generalization.

2.2 Reasons for using interactive systems

Because generalization is an ill-defined domain it is argued that this process is too complex to be modeled as a fully automatic batch system from scratch. In building our system there are four basic design principles employed to cope with complexity: interactivity, cooperativity, incremental learning, and adaptivity.

First, the proposed strategy is to use interactivity to give control to the user for informal tasks and programming the computer to support him actively. Informal tasks require human interaction because it is assumed that the computer will never have access to all information needed. On the other hand, the computer can take over formal tasks - an activity where it is best suited.

Another argument for interactive systems is that generalization is a situative process which is part of a changing environment. Even the introduction of this system itself will change the work place of the user. Not being situated means that at any time there is still some on-line information lacking, so the system can never be updated on time according to the changing environment. Therefore, it is only the user who has the needed information available. He is more 'situated' in the complex and changing process.

Second, 'cooperative' systems provide interfaces which suggest partial solutions to the less skilled user (like 'agents'). This means that the computer will sometimes pick up control over the process, but the user can still follow the decisions and is keeping the top-level control.

Third, incremental learning makes sure that unforeseen, previously implicit knowledge is being introduced in the system. This is part of the overall strategy to build better generalization systems because the knowledge in this domain is scarce.

Fourth and last, adaptivity means that the system automatically adjusts to new environments through a learning component. For example, in a different culture to map or over time - given the same input- the parameters could be different. Therefore, functions are needed which incrementally acquire procedural and static knowledge using direct processing of transaction logging.

In the system described here, only the first two design elements have been implemented yet.

2.3 Genetic algorithms for optimization

Genetic algorithms (GA) are a model of machine learning which is inspired by the metaphor of biological evolution in nature. GA have their origins in robotics and optimization techniques and are viewed as a particular method from the field of 'evolutionary strategies'. Common to this family of algorithms is the simulation of evolution of individual structures through processes called selection, mutation and recombination (or crossover). The processes are evaluated by the performance (or fitness) of individual structures (see e.g. [4,5]).

GA apply an informed pseudo-parallel search in the parameter space. Search can be viewed both as a kind of problem-solving and an optimizing process. More precisely, GA use a general heuristic for exploration in a population of structures and come up relatively fast with a suboptimal solution. This near optimum is close to a global optimum with very high probability. GA are bad suited for problems that require explicit problem domain knowledge and where this knowledge is relatively easy to formalize. However, GA sometimes outperform other algorithms (e.g. linear programming) on problems which are NP-complete (= infinite number of possibilities) and are often used where other problem solving strategies failed. Although simplistic from a biologist's point of view, these algorithms provide robust and powerful adaptive search mechanisms.

As a result, GA can be used to optimize functions for which derivatives are difficult to find or even non-existent, like in the domain of graphic design. They do not require any knowledge of the gradient of the objective function. This is because in GA the search is not conducted from point-to-point - like in optimization and linear programming - rather it searches from a population of points to a new population of points in the parameter space by using probabilistic rules. These rules are the 'recombination' and 'mutation' operators, which are usually conducted using some type of random number generation to specify how the new population of points will result from the current one.

Figure 1 shows the data flow external to the GA.

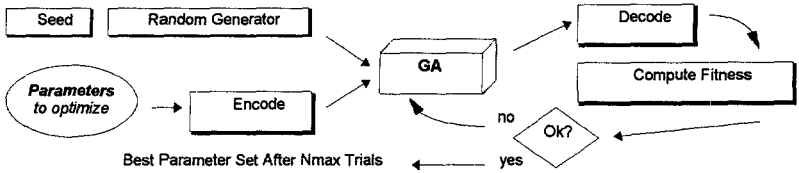


Figure 1. Starting with a random seed the parameters are encoded into bit strings, the GA (as a 'black box') is being called, which in turn needs an external fitness function.

In figure 2 the internal functions of a GA are explained. Initially the pool of parameter sets needs to be prepared. This can be done with a random generator, or, if knowledge from the application domain exists, look-up tables can be used. The pool is being encoded into binary strings. In the next step the pool ('population') is being evaluated by the externally defined fitness or performance function. Based on this evaluation the selection process picks the 'fittest' with some higher probability that the others, performs a recombination, possibly a mutation and then, the process starts over. This marks one cycle (iteration, trial) and represents the 'lifetime' of one 'generation'. It is repeated until a specific stop or convergence criterion is being reached (e.g. a fixed number of iterations, an error or a convergence threshold). Finally, when the optimization process stopped, the best parameters ranked by the fitness value become the output.

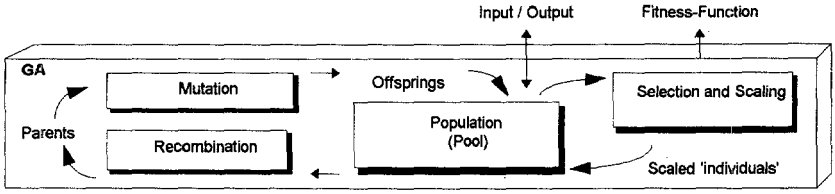


Figure 2. Internally in the GA - after being called - the 'individuals' (the parameters) are selected with a given probability (evaluated by the external fitness function), recombined with others, and mutated until a stopping criterion is reached.

3 'Generalization by example' for parameter control

The selection of generalization operators is one of the most difficult parts automating the digital map making process. In existing generalization tools, often there are various algorithms and parameters at disposal. We found that it is difficult to choose the right algorithms and parameter settings to solve a given generalization problem at hand. The systems do not indicate which operator sequence to select and often they offer little help to start with reasonable defaults. Additionally, it is hard to get control over the impact of the different numerical parameters to the result, because the level of decision making is on the graphical output at target scale. There is no intuitive relation between the values of the parameters and the final graphical effect to the output. This is because the user has to provide numbers for the input instead of describing analogically the properties of the objects at target scale. Thus, it can be stated that the systems - though built as interactive learning and assisting environments - still do not to give the user full control over the map making process at a task or process oriented level.

In some cases, once having picked an algorithm for a task, users are not reconsidering this decision afterwards. This could be explained that it is time consuming or tedious to evaluate visually the algorithms. In other cases experienced users realized that the impact of a particular algorithm has to be compensated because it produced unwanted side-effects like over-emphasizing parts of the line structure. In this case, one could conclude that the algorithm is probably not suited to the data input (because of his underlying data structures) or it is insensitive to the essential information (e.g. topology).

3.1 'Generalization by example'

In 'generalization by example' the user describes analogically the properties of the objects at target scale: he provides a solution by drawing a typical example. Then, the system analyses the input and tries to replicate this part using the same input data. This means computationally to adjust the operators and parameters appropriately. Now, finding a sequence and the setting of the parameters is being viewn as an optimization task: they all are competing for the best approximation to the example, ranked by an evaluation function. After having found the best sequence and parameters, they are presented to the user. If he or she accepts the proposition of the system, the choice is being applied to the entire active working area using the original data.

Compared to other machine learning methods, examples are being treated here at the time the user provides them. This approach combines CI techniques and interactive systems: like in amplified intelligence key decisions default explicitly to the user, who initiates and controls the operators which finally carry out the generalization tasks.

3.2 Implementation

In this implementation the objective is to compute reasonable default settings for the parameters of two line simplification algorithms, Douglas-Peucker [3] and Lang [11]. For optimization, we have chosen GA because they are good documented and especially well suited for parameter optimization. In this test the source and target scales are 1:100,000 and 1:250,000 over several line works. It is a road database coming from the French National Mapping Agency (Institut Géographique National).

The workflow of 'generalization by example' applied to parameter setting starts with the user who interactively draws, on a representative part of the working area, a generalized line at target scale, called the 'sample line' (figures 3. and 4.). This manual generalization is being used to derive the parameters for a previously selected algorithm that would match best the sample line. Internally, the system generates iteratively some parameter settings and evaluates them. Finally the system presents to the user one or several propositions which match best the sample line. Once the user has accepted the control parameters for the generalization algorithm, they are expanded to the entire window or layer. Additionally, process tracing functions could be hooked into this process to log data for subsequent use. Thus, as a side-effect, this approach has built-in knowledge acquisition capabilities.

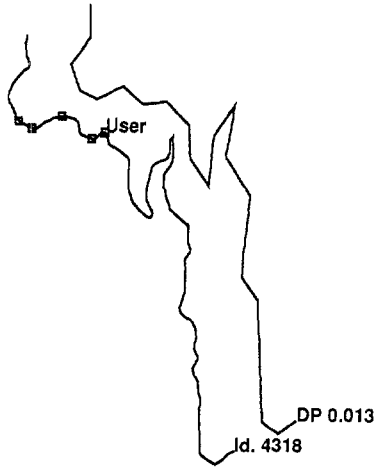
The interactive setting of parameters is implemented in five steps. User interaction is involved in the first step (drawing the sample line) and in the last step (accepting or rejecting the parameters):

1. A sample line, which should be a small but representative part of the original polyline, is being drawn interactively by the user to indicate the implicit geometric properties as, for example, the sinuosity of the intended generalization output.
2. From the sample line the corresponding part of the original polyline is being taken for input to compute the output with a different set of parameters.

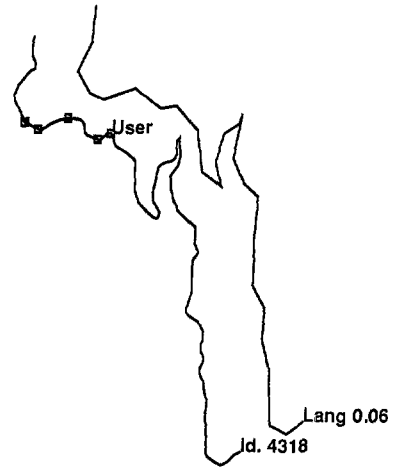
3. The output is evaluated by the system and those settings which match best the given sample line are selected.
4. The final output of the whole polyline is generated using the found parameters.
5. The output is being shown to the user who can decide either to accept, reject, or modify the parameters. If he accepts, they are applied to the entire working area.

For the user drawing the sample line there are two geometrical 'degrees of freedom' possible a) being restricted to existing points on the line (resulting in a strict simplification) and b) drawing 'free hand' which complicates evaluation and tests for plausibility.

In this test the evaluation function is calculating the mean difference between the values from area and vector displacement as defined by McMaster [12]. A weighting formula was being used to integrate the different evaluation measures. It is clear, that they only form a starting point because the line structure is only being described as basic geometric representations. Tuning the GA was a time consuming part of the experiment. In GA there are many parameters to be set for itself. So, empirical testing was needed to get satisfying convergence behavior. The convergence criterion was set to a fixed number of cycles. There are other criteria remaining to be tested. The result using the two algorithms with the same input data is being shown in figures 3. and 4.



Douglas-Peucker p1=0.013



Lang p1=0.06 p2=5

Figures 3. and 4. 'Generalization by example': the squares indicate the trace of the user input, called the 'sample line'. A genetic algorithm finds those parameter values which generate the most similar output to the sample from the original line using the algorithms Douglas-Peucker (0.013 map units) and Lang (0.06 map units and 5 control points). The parameter settings are then used to simplify the whole selected original line (no. 4318).

3.3 Discussion

Visually evaluating the output of both algorithms gives a good impression of the feasibility of the approach. Compared with each other using the same input data, the Lang algorithm produced a better result. In figure 4, the shape around the 'half island' looks smoother in contrast to the output of Douglas-Peucker (figure 3), where the same area got a crisp peak which overemphasizes the information from the input data. In general, the results show that the found parameter settings lie in a reasonable range given the target scale and the intentions of the user.

It is assumed that the implemented evaluation criteria are not sufficient to cover more complex line data. Though, they seem to be robust enough for the type of data used. Additionally, it turned out that the weights for the simple weighting formula that integrates the different evaluation measures are difficult to find. Only further empirical tests will give more information about the priorities of each measurements contributing to the final evaluation score (the final 'fitness value').

The display of the output at target scale has some impact to the perception of the user: display at source scale would be easier to edit (for postprocessing) but at target scale, e.g. the minimal dimensions are easier to verify.

The computation of defaults of line simplification algorithms can easily be extended to find sequences and combinations of algorithms or even to include additional generalization operators, e.g. smoothing. One of the strengths of GA are their scalability. This means, that if more parameters and algorithms are added this has little influence to the basic functionality or response time of the system. It has to be mentioned however, that this implementation has no satisfying response time yet due to the big amount of iterations of the GA. But, GA are well suited for parallel optimization and could profit from parallel computer architectures.

4 Conclusions

This is an initial report of our research to evaluate CI techniques embedded in interactive systems. Because of the complexity of the domain it is concluded that automated generalization tools need to be shaped as interactive, cooperative, learning and adaptive principles.

The idea of 'generalization by example' is mainly pursued to investigate new methods which support the human decision making process. It is worthwhile to examine other typical generalization tasks with this approach, for example, the selection of clustered point objects (like distributed settlements) by showing the distribution of a sample area. The displacement of objects where neighbors are moved similar to the action initiated by the user would be another example.

It is shown that this approach provides new options for interaction at the adequate conceptual level and at the same time it could deliver more semi-formal knowledge through interaction logging. The next step is to complement the algorithmic operators by more support facilities (e.g., tools giving information on feature clustering, spatial conflicts and overlaps), also providing decision support for the user.

The future work agenda contains the following items:

- To integrate more algorithms for simplification and to perform further tests with different data sets.
- To extend the system with sequence optimization by adding, e.g. smoothing operators.
- Most urgently, more effective (absolute) differentiating measures and (comparative) matching functions for evaluating the quality of generalization results are sought.

More basic questions are: What is the behavior of the system, if the sample line is being drawn in a sinuous area (like a mountain road) which is followed by a quite separate characteristic (like a straight highway)? Can the optimization process be replaced by precomputed look-up tables? How good are these CI-based tools embedded in the human-computer environment? How can this approach be extended with 'generate-and-test' to automatically come up with generalized solutions?

We strongly believe that this research will lead to better theoretical and practical grounds for the automation of these complex processes. The distinction between interactive and batch systems is a very important design decision of a generalization system. The results give some hints that this type of interactivity could be a useful component for the construction of those tools.

The near future will show whether interactive systems are an intermediate step towards fully 'roboterized' systems, or, if the problem is too complex, and assisting, cooperative systems are a possible alternative.

Acknowledgments

The author is especially thankful for getting the data from the COGIT laboratory team at the Institut Géographique National, France, and for the discussions with Robert Weibel, Hansruedi Bär, Carly Williams, and the patience and cherish support from Sonja, Muriel and Dominik.

References

[1] Armstrong, M.P. (1993): A Coarse-Grained Asynchronous Parallel Approach to the Generation and Evaluation of Map Generalization Alternatives. NCGIA Specialist Meeting on Formalizing Cartographic Knowledge, Buffalo (USA), October 23-27, 1993, 37-43.

[2] Carbonell, J.G. (1990): Introduction: Paradigms for Machine Learning. In: Carbonell, J.G. (ed.): Machine Learning: Paradigms and Methods. Cambridge, MA: MIT Press.

[3] Douglas, D.H. and Peucker, T.K. (1973): Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature. The Canadian Cartographer, December 1973, 10(2): 112-122.

[4] Goldberg, D.E. (1989): Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading.

[5] Holland, J.H. (1975): Adaption in Natural and Artificial Systems. University of Michgan Press, Ann Harbor, MI.

[6] Imhof, E. (1937): Das Siedlungsbild in der Karte. Mitteilungen der Geographisch-Ethnographischen Gesellschaft, Zürich, Band 37, 17-85.

[7] Keller, S.F. (1994): Sustained Spatial Data Management in Real-World Projects - A Research Focus. In: Nievergelt et al. (eds.): IGIS'94 International Workshop on Advanced Research in GIS. Monte Verità, Ascona, Switzerland, February 28 - March 4, 1994, Lecture Notes on Computer Science (LNCS) No. 884, International Series, Springer, 155-167.

[8] Keller, S.F. (1994): On the Use of Case-Based Reasoning in Generalization. Proceedings of the Sixth International Symposium on Spatial Data Handling '94 (SDH '94). September 5-9, 1994, Edinburgh, Scotland, 2:1118-1132.

[9] Keller, S.F. (1995): Potentials and Limitations of Artificial Intelligence Techniques Applied to Generalization. In: Müller, J.-C., Weibel, R., Lagrange, J.P. and Salgé, F. (eds.): GIS and Generalization: Methodological and Practical Issues. London: Taylor & Francis.

[10] Lagrange, J.P. and Ruas, A. (1994): Geographic Information Modelling: GIS and Generalisation. Proceedings of the International Symposium on Spatial Data Handling '94. September 5-9, 1994, Edinburgh, Scotland, 2:1099-1117.

[11] Lang, T. (1969): Rules for the Robot Draftsmen. The Geographical Magazine, 42: 50-51.

[12] McMaster, R.B. (1987): The Geometric Properties of Numerical Generalization, Geographical Analysis, 19(4): 330-346.

[13] McMaster, R.B., and Mark, D.M. (1991): The Design of a Graphical User Interface for Knowledge Acquisition in Cartographic Generalization, GIS/LIS'91, 1: 311-320.

[14] Müller, J.-C. (1992): Parallel Distributed Processing: An Application to Geographic Feature Selection. Proceedings of the Fifth International Symposium on Spatial Data Handling, Charleston, SC, 1: 230-240.

[15] Müller, J.-C., Weibel, R., Lagrange, J.P., and Salgé, F. (1995): Generalization: State of the Art and Issues. In: Müller, J.-C., Weibel, R., Lagrange, J.P. and Salgé, F. (eds.): GIS and Generalization: Methodological and Practical Issues. London: Taylor & Francis.

[16] OEEPE (1993): Questionnaire for Evaluation of Generalization Systems. Version 0.1, OEEPE Working Group on Generalization.

[17] Weibel, R., and Buttenfield, B. (1988): Map Design for Geographic Information Systems. GIS/LIS'88, San Antonio, TX, 1: 350-359.

[18] Weibel, R. (1991): Amplified Intelligence and Rule-Based Systems. In: Buttenfield, B.P. and R.B. McMaster (eds.): Map Generalization - Making Rules for Knowledge Representation. Longman, Harlow, UK: 172-186.

[19] Weibel, R., Keller, S.F., and Reichenbacher, T. (submitted): Overcoming the Knowledge Acquisition Bottleneck in Map Generalization: The Role of Interactive Systems and Computational Intelligence. Conference on Spatial Information Theory (COSIT'95). September 21-23, 1995, Semmering near Vienna (Austria).

[20] Weibel, R. (1995): Three Essential Building Blocks for Automated Generalization. In: Müller, J.-C., Weibel, R., Lagrange, J.P. and Salgé, F. (eds.): GIS and Generalization: Methodological and Practical Issues. London: Taylor & Francis.

[21] Werschlein, T., and Weibel, R. (1994): Use of Neural Networks in Line Generalization. EGIS'94, Paris, March 30 - April 1, 1994, 76-85.