Formalization and Automatic Interpretation of Map Requirements

Xiang Zhang[1,2], Jantien Stoter[1] & Tinghua Ai[2]
[1]International Institute for Geo-Information Science and Earth Observation
P.O. Box 6, 7500 AA Enschede, the Netherlands
{xzhang,stoter}@itc.nl
[2]School of Resource and Environment Sciences, Wuhan University, 430072, China
tinghuaai@gmail.com

ABSTRACT

The map requirements (constraints) can be interpreted by computer programs using their basic embedded functionalities. There are a huge number of constraints available to define the objective of various generalization outputs. Some of the constraints contain high-level knowledge which is not easy to interpret. This needs a huge amount of efforts to implement those constraints. The fact that many constraints have something in common makes the implementation per constraint a waste of resource. The paper proposes to decompose the constraints into more basic units, so as to interpret those constraints more flexible and reuse the already developed functionality as much as possible.

1. INTRODUCTION

This work is motivated by the effort of enabling automated evaluation of generalization output, which aims to give insight into the overall quality of generalized data and comparing different generalization solutions regarding specific map requirements and map tasks (Ruas, 2001; Mackaness & Ruas, 2007). The quality of generalization output is specified in terms of map requirements. To automate the process of evaluation, map requirements should be defined and formalized in a machine-interpretable manner. However, the formalization is not always an easy task, since map requirements might involve high-level knowledge which is difficult to formalize and interpret. This paper presents an approach to formalize map requirements by decomposing them into different types of low-level knowledge. Then a framework for automatic interpretation of the formalized map requirements is presented. The basic assumption is that, in order to cover all possible output maps, there are infinite map requirements which can be described by countable sorts of low-level knowledge. The formalization and interpretation at knowledge level rather than at requirement level will provide a more flexible framework for automated evaluation.

The work will only focus on the formalization and interpretation of map requirements without addressing how the requirements will be evaluated. Section 2 reviews previous effort on defining map requirements for generalization outputs (section 2.1). And data enrichment techniques for automatic interpretation are described in section 2.2. Section 3 proposes a conceptual framework for formalization and interpretation of map requirements in the context of automated evaluation. In section 4, we illustrate how the framework works by addressing a special interpretation issue. The paper ends up with conclusions in section 5.

2.   RELATED WORKS

2.1.   Defining Map Requirements based on Constraints

Beard (1991) introduced the first time a constraint-based approach which aims to provide a flexible framework for automated generalization. The constraints provide a natural means to define explicitly the requirements (objectives) of generalization output, without addressing how the result needs to be achieved (Weibel, 1996; Weibel and Dutton, 1998). The definition of map requirements with constraints offers also the possibility to evaluate generalization output (Burghardt et al., 2007). Galanda (2003) utilized a set of formalized constraints (e.g. metric, topological) for automated evaluation before and after a step of generalization iteration however, the formalization of constraints is far from being fully machine-interpretable. Burghart et al. (2007) found that the degree of formalization of constraints varies strongly and some of them can not be interpreted by computers. For instance, most metric constraints in their case can easily be formalized while shape constraints are insufficiently formalized. They also argued that all information that is added to generalization process by humans need to be formalized.

2.2.   Data Enrichment Techniques for Automatic Interpretation in Map Generalization

Huge gap still lies between data representation and data interpretation for automated generalization. Well-structured data can be interpreted automatically to some extent but still has a lot of implicit information which could only be interpreted visually (Ruas, 1998; Sester, 2000). The understanding of spatial data depends largely on contextual relationships in a scene (e.g. the adjacency and neighborhood information) which need to be explicitly modeled. However, it is arguable whether this kind of information is best stored or calculated on demand (Ruas & Lagrange, 1995). Due to the fact that the context or salient feature is always temporary and changing with the purpose and scale of maps, Dutton and Edwardes (2006) pointed out it is more useful to compute contextual information on demand. Meanwhile, they agreed also that the design of application-specific spatial databases should be supported by semantics modeling (e.g. *ontologies*), since semantic properties cannot be deduced from geometry. As for contextual especially structural aspect, considerable progress has been made. The recognition of spatial concepts (e.g. shape, bend structure, proximity, alignment, and cluster) is made possible by adopting auxiliary data structures, such as *Delaunay Triangulation*, *Voronoi Diagram* and *Minimum Spanning Tree* (Gold, 1994; Jones et al., 1995; Regnauld, 1996; Ai et al., 2000; Ai, 2007). Besides the enrichment mechanism in spatial aspect, enrichment in semantic aspect has already been carried out. For example, Stoter et al. (2007) discuss the design of a semantically-rich multi-scale data model for facilitating (semi-)automated generalization and producing coherent topographic databases across designated scales. The UML-based data model shows the possibilities to formalize constraints with Object Constraint Language (OCL) at schema level, while the interpretation of these formal constraints still depends on the functionalities of GIS systems. This paper attempts to build a link between the description and interpretation of cartographic constraints.

3. A FRAMEWORK FOR THE FORMALIZATION AND AUTOMATIC INTERPRETATION OF MAP REQUIREMENTS

3.1. Knowledge Decomposition for Constraint Formalization

Motivation of the Decomposition

In algorithm-driven generalization, different types of knowledge in constraints were mixed up and hardcoded into algorithms. The knowledge in one algorithm cannot be reused for the situations with even slight difference from the designed one (Steiniger, 2007) or for other parts of the generalization system. For instance, the description of bend structure can be used both for simplifying sinuous river and for generalizing grouping contour lines on the surface. Normally, different algorithms were developed separately and named *'river simplification'* and *'contour line generalization'* respectively. But this is not a modular approach. The structure knowledge in this case should be detached from semantic knowledge and be used independently for multiple purposes.

For the purpose of automated evaluation, one constraint is evaluated by one measure and treated as a whole (Galanda, 2003). One main disadvantage of this approach is that, when new constraints are added new measures have to be redesigned. The constraint might be very complex, and it is also true for the design of corresponding measure, which always involves repeated implementations for at least part of the existing algorithms. If different types of knowledge in algorithms can be decomposed and implemented independently, the previous algorithms in a system can be reused as much as possible.

Knowledge-based Decomposition of Constraints

The decomposition of generalization knowledge into different types of knowledge was discussed by previous work (e.g. Ruas and Lagrange, 1995). The authors decompose constraints into five type of knowledge: 1) *Geometric knowledge* contains basic knowledge of geometry type, length, area, distance, orientation, etc. 2) *Topological knowledge* contains well-defined elements like contain, meet, intersection, disjoint, and other topological relationships. 3) *Semantic knowledge* involves information about meaning of geographic objects and their interactions, such as Building is accessed by Road (topologic and semantic). 4) *Procedural knowledge* implies what actions (operators) to take under specific circumstances. 5) At last, *structural knowledge* describes characteristics of spatial organization.

Principles of knowledge-based decomposition are:
- Decomposed knowledge should be formalized for further recognition, as well as its context and hierarchical relations (Lüscher et al, 2007)
- High-level concepts (e.g. peninsula) in constraints should be decoupled into different low-level knowledge and their interrelation
- Low-level knowledge means it can be interpreted/implemented by automated generalization or evaluation systems (operational requirement)

Structural knowledge was frequently discussed in the past and it has different implications. For instance, structural knowledge was referred to as describing aesthetic and visual balance

(Weibel, 1996), which is difficult to define and measure. Some may define structure knowledge as describing spatial and semantic order as well as their interdependencies (AGENT, 1998).

For the sake of clarity, we define *structural knowledge* in our research as: *relations concerned with spatial arrangement/ordering of the sub-objects that form the relations*. For example, ring/star-shape of road network is composed of connected segments with different length and orientation; grid alignment of building clusters is form by the certain configuration of individual buildings; shape of geometric object is determined by its vertices and their spatial distribution. According to the number of features involved, *structure knowledge* can be divided into *intra-structure* (e.g. shape, bend of individual geometries) and *inter-structure* (e.g. alignment, pattern, distribution, density of a group of objects). Again, *intra-structure* can be divided further into *linear structure* and *non-linear structure*. The latter one considers not linear sequence but the context of the structure across 2-dimension space (e.g. hierarchical bend structure as will be discussed in section 4.3).

Multiple uses of structural knowledge (roles) can be distinguished:
- It acts as objective of generalization and evaluation (e.g. detect small bends; preserve the balance of building density in different area)
- It acts as the identification of specific situations (context), which helps the selection of operators, parameters, etc. (e.g. different constraint value in different context)
- It acts as basic analysis tools for interpreting geographic meaning from spatial data.

One problem is how to aggregate the decomposed parts of a constraint together for the interpretation of this constraint.

3.2. Automatic Interpretation Framework
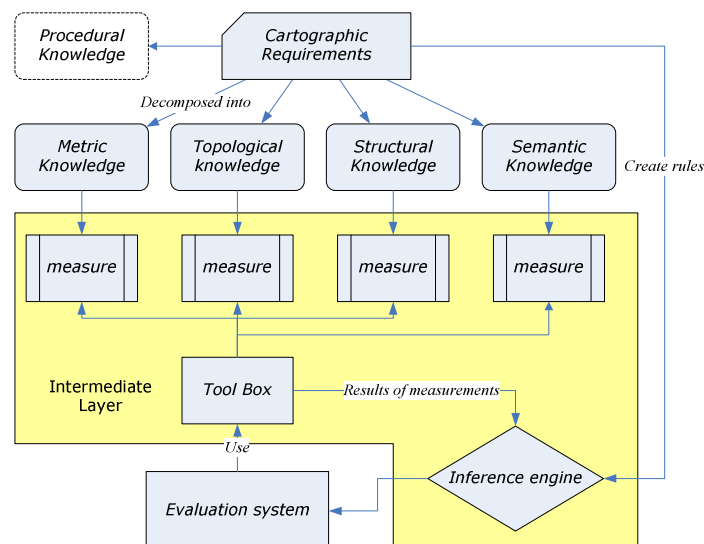
Overview of the Framework

Figure 1: Conceptual framework of machine-based formalization and interpretation of cartographic requirements for automated evaluation.

Figure 1 shows basic framework for automatic interpretation and implementation of constraints (i.e. cartographic requirements). The framework comprises three major layers. The first layer is cartographic requirements, which are specified by application domains and decomposed into low-level knowledge. The second layer is intermediate layer, which is used for interpreting the constraints based on the low-level knowledge. The evaluation layer makes the third layer.

In the motivation part of section 3.1, we mentioned that a constraint is used to be treated as a whole when it is evaluated. In that case, evaluation system can be designed as a two-layered architecture, where the intermediate layer in Figure 1 is dismissed and constraints are validated directly by evaluation system. While in this proposed framework, the intermediate layer is introduced for automatic interpretation of constraints and the evaluation of a constraint can be performed by the reasoning of the low-level knowledge. Within this layer, several elements should be paid attention to: a) measures in a *toolbox* are designed against the low-level knowledge but not on constraints; b) an *inference engine* should be developed to perform the reasoning process; c) the rule of the inference process is created by analyzing the syntactic structure of cartographic constraints. Note that *geometric*, *topological, structural* and *semantic knowledge* are first measured by the *toolbox*. The results of measures then enter into the *inference engine* which at last interprets the constraint.

Predicate Logic and Terminological Reasoning for Automatic Interpretation

We analyze the roles that different types of knowledge play in constraints, and then discuss two levels of inference faced in the process of machine-based interpretation. Firstly, let us observe some concrete constraints with semantics information at scale 1:50k:

	Feature type	Geometry	Attribute	Attribute value (partial)
Data model (partial)	Road	Line	Type	'inter-settlement', 'intra-settlement'
	Building	Polygon	Type	'individual', 'building block', 'settlement', 'industrial'
Constraint 1	Constraint: inter-settlement road should be contained by settlement and meet at the boundary of settlement building.			
Constraint 2	Constraint: building size should be at least 0.04 map mm^2 Preferred action: individual buildings in urban district with size < 0.04 map mm^2 should be eliminated; individual building in rural district with size from 0.01 to 0.04 map mm^2 should be enlarged to 0.04 map mm^2.			

Stoter et al. (2008) pointed out that preferred actions (procedural knowledge) suggested in map requirements should be added as part of the constraints, which will improve the description of generalization output. In Constraint 2, preferred actions are added. As for interpretation, Constraint 1 is simpler since all elements are machine-interpretable and semantically adequate, where the *topological knowledge* can be formulated by Line-Region relations (e.g. Egenhofer & Mark, 1995). While constraint 2 is not machine-interpretable since high-level concepts such as 'urban district' and 'rural district' cannot be inferred from the data model.

Analyzing the syntactic structure of cartographic constraints gives insights into how the low-level knowledge is organized logically in the constraints. This step is good for translating the constraints into **predicate logic** which can be used by the *inference engine* in the last section. In this paper, we propose to use *subject*, *subject modifier* (*s_modifier*), *predicate*, *object* and *object modifier* (*o_modifier*) to describe syntactic structure of constraints. For example, constraint: "intra-settlement road should get access to individual building" can be expressed as follows (pseudo formal language):

Access (Building | **Type** (Building) = 'individual', Road | Type (Road) = 'intra-settlement') = **True**

[*subject*]	→ Building
[*s_modifier*]	→ **Type** (Building) = 'individual'
[*object*]	→ Road
[*o_modifier*]	→ **Type** (Road) = 'intra-settlement'
[*predicate*]	→ **Access** (…) = **True**

Subject and *object* refer to map objects expressed in constraints; *predicate* describe the conditions to which the properties of objects or relationships between objects should adhere (e.g. **Size** (building1) > 0.04 map mm^2); *modifier* has the functionality of filtering the *subjects* or *objects* by setting conditions to them. For example, *semantic knowledge* (**Type** (Building) = 'individual') or *structural knowledge* (**Context** (Building) = 'rural district') can be used as different *modifiers*. The syntactic structure of constraints is illustrated in Figure 2.

Another kind of reasoning is also important for the automatic interpretation process. That is **terminological reasoning** as defined in Haarslev et al. (1994). If we observe constraint 2, it is easy to find that the *modifier* 'urban district' is an ill-defined concept, which cannot be interpreted. The concept can be formalized by decoupling it into lower-level knowledge such as size, shape, proximity distance, density of buildings in their local context. Similar formalization was discussed on topics of urban building structure recognition by (Lüscher et al, 2007). *Terminological reasoning* aims to identify and classify the instances in data based on the knowledge used to describe certain concepts. This kind of reasoning is implemented by incorporating a set of rules with spatial analysis techniques (e.g. spatial reasoning/structural recognition techniques). In case of constraint 2, this reasoning process is needed for identify the concept of 'urban district'.
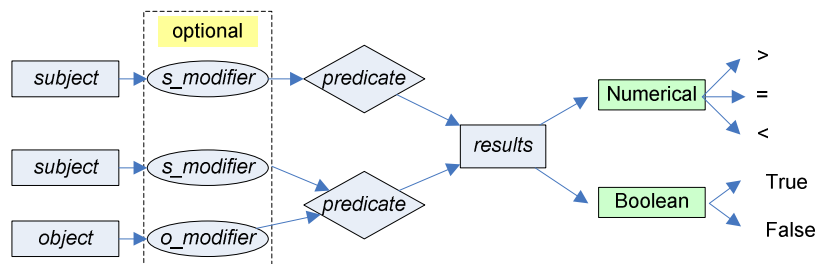

Figure 2: Syntactic structure of constraints

Within the proposed framework (i.e. *inference engine*), *predicate logic reasoning* is employed to see if certain condition is satisfied (syntactic level of constraints). *Terminological reasoning* occurs at lexical level of constraints, i.e. *subjects*, *objects* and concepts in *modifiers*, in case that they are not well-defined. Note that most *predicates* mentioned here are spatial predicate, and

the type of the results are either numerical or Boolean. Besides, *predicates* in constraints can be *1-nary*, *binary* and *n-nary* respectively. *1-nary predicate* evaluates the constraints on one object; *binary predicate* describes the constraint between 2 objects; and *n-nary predicate* describes the constraints on groups of objects. Several examples are given to demonstrate this *predicate logic* (pseudo formal language):

Size (Building) > X map mm^2
Access (Building, Road) = **True**
Density (target building group) = D**ensity** (initial building group) * X %
Exist (Building | size (Building) ∈ [a, b] *and* **Context** (Building) = 'rural district') = **True**

Where, **size** and **exist** are two *1-nary predicates*; access is a *binary predicate* concerning topological constraint; **density** is an *n-nary predicate*. The last *predicate logic* means if buildings in rural district with size in [a, b], then they should be preserved. But the *modifier*: **context** (Building) = 'rural district' should be interpreted through *terminological reasoning*.

4.   A SPECIAL INTERPRETATION ISSUE: BUILDINGS AT THE END OF PENINSULA

In this section, the framework is applied to a special interpretation issue to show the flexibility of the framework.

4.1. Problem Statement

In coastal area cartography at small scale, there are cartographic requirements like: *"roads leading to a building at the end of peninsulas must not be omitted"* and *"buildings at the end of peninsula should be preserved"*. These requirements can be expressed using the above-mentioned approach (pseudo formal language):

C1: **Exist** (Building | **Context** (Building) = *'peninsula'*) = **True**, where

[*subject*]   → Building;
[*s_modifier*]   → **Context** (Building) = *'peninsula'*;
[*predicate*]   → **Exist (…) = True**;

C2: **access** (Building, Road) = **true**, where

[*subject*]   → Building;
[*object*]   → Road;
[*predicate*]   → **Access (…) = True**;

All the elements in these two constraints can be tagged with syntactic symbols described in the second part of section 3.2. The form of the constraints is likely to be interpreted by computers. Take *C2* for example, it is evaluated with high level of automation based on topological relation *predicate logic*. However, it is not the case for *C1* since the *s_modifier*: **context** (Building) = *'peninsula'* is not well-defined, especially the concept of *'peninsula'*. Since 'peninsula' is not stored as attribute of polygons in databases. Even the meaning of *'at the end of'* in natural language is not clear, which is interpreted differently according to different scales and culture. But this is out of the scope of this paper.

4.2. Formalization of 'Peninsula' with Low-level Knowledge

Geographical phenomenon *'peninsula'* is defined in natural language as "a piece of land that is bordered on three sides by water. It can also be a headland, cape, island promontory, bill, point, or spit". The expression contains various synonymous and ambiguous terms. In order to be machine-readable, *'peninsula'* could be defined as a geometric bend structure (of lines or polygons) of coastal line that is adjacent to both land and sea feature. This definition makes use of *structural* and *semantic knowledge* explicit. We use a semantic model to demonstrate this formalization (see Figure 3).
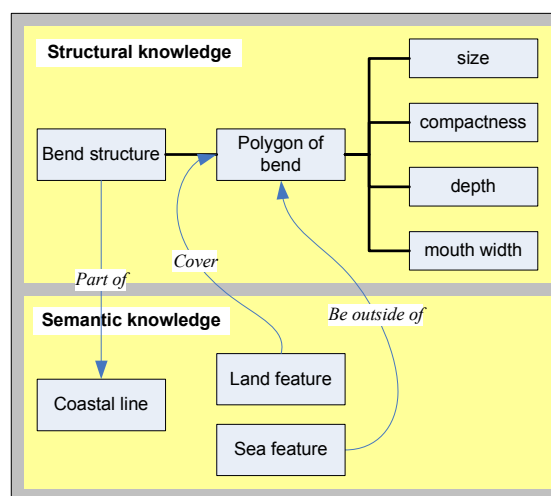

Figure 3: Semantic model of *'peninsula'*: its low-level knowledge and interconnections.

The figure shows what a 'peninsula' is, by different types of low-level knowledge and their interconnections. *'Peninsula'* is a *bend structure* that is a **part of** a *coastal line*; it is represented by the *polygon of that bend structure*, with *land feature* **inside** and *sea feature* **outside**; the bend polygon of *'peninsula'* is specified by a set of descriptors according to applications (only few of them are listed here). In this model, we use mainly *structural* and *semantic knowledge* and spatial relations between them. In practice, *geometric knowledge* should be also involved.

Terminological reasoning is suggested for triggering structure recognition techniques to tag polygons of bend and then identifying the geo-facts of *'peninsula'*. The adopted techniques are in the following section and will be implemented by spatial analysis *toolbox* at evaluation end as proposed in section 3.2.

4.3. Bend Structure Recognition Technique

Detection of bend structure: The description of bend structure has long been discussed and investigated, since bend structure is frequently found on linear and polygonal in the context of map generalization. There are a lot of techniques for detecting bend structure. We use a technique that was proposed by Ai et al. (2000) for its capability of detecting bend structure at different levels of detail by *Delaunay Triangulation*. The technique is illustrated in Figure 4. Detailed discussion will not be covered by this paper.
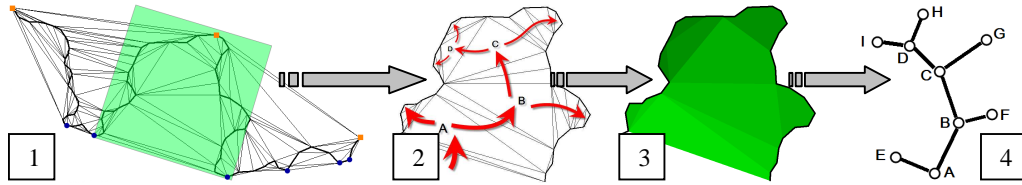
Figure 4: Detection of bend structure. 1: triangulation and characteristic point detection; 2: bend detection; 3: hierarchical bend structure; 4: binary tree representation of all embedded bends.

Characterization of bend structure: In the case study, we adopt size, compactness index, mouth width and depth of bend for characterizing a bend structure. The computation of mouth width and depth is based on triangulation and skeleton analysis (see Ai et al., 2000). With the advantages of the hierarchical representation of bend structure, all these characteristics can be derived at different levels of detail according to different applications. Table 1 outlines the chosen characteristics and their corresponding measures.

Characteristics	Measures	
Size	size =area (polygon); polygon is enclosed by bend segment and *base line*	
Compactness index	" the ratio of the area of the polygon over the circle whose circumference length is the same as the length of the circumference of the polygon" (Wang and Müller, 1998)	
Mouth width	width = length (*base line*)	
Depth	depth = length (*trend line*) (Ai, 2007)	

Table 1: Measures (terrain unit) for characterizing bend structure.

4.4. Identification of 'Peninsula'

Figure 5 illustrates the whole process of machine-based identification of *'peninsula'* by linking the low-level knowledge introduced in section 4.2 with the technique in section 4.3. One can hardly identify the polygon of *'peninsula'* without semantic knowledge like land and sea feature, since it is ambiguous to infer on which side the 'peninsulas' reside based on structural knowledge only (lower right picture). While this decision could be made by the following reasoning on semantic and topologic relations:

> **meet** (bend polygon, sea feature) **and** **coverby** (bend polygon, land feature) = **true**

A similar expression with slight change in *semantic knowledge* will help machines to identify *'bay'* (e.g. cove and harbor):

> **meet** (bend polygon, land feature) **and** **coverby** (bend polygon, sea feature) = **true**

9

The resulting area presented here are still very rough, and the candidates can be refined with different criteria (e.g. size, mouth width, depth, etc.) specified by users according to different applications.
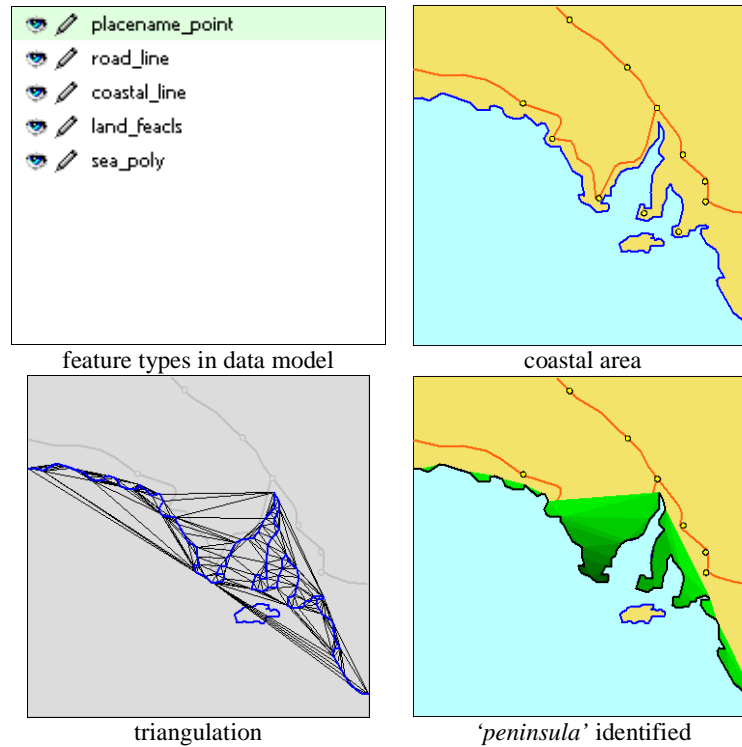

feature types in data model   coastal area

triangulation   *'peninsula'* identified

Figure 5: *'Peninsula'* identification

A further analysis based on the hierarchical bend structure description will give the possibility of interpreting the whole sentence: *'building at the end of peninsula'*. And this will not be discussed in detail in this paper.

5. CONCLUSIONS

The paper proposed a framework for the formalization and automatic interpretation of map requirements for automated evaluation. Following the framework, high-level knowledge in map requirements can be formalized into machine-readable low-level knowledge, so that they can be interpreted by evaluation system. The proposed framework was demonstrated by a special interpretation issue, which showed that further implementations for this topic is needed. Future work includes: 1) testing and using a formal language to represent map requirements under the proposed framework (the use of an ontology language seems to be promising); 2) research on how the proposed framework can be implemented and how decomposed map requirements can be used for evaluation.

REFERENCES

AGENT, 1998, Constraint Analysis. Deliverable A2, http://agent.ign.fr/deliverable/DA2.html

Ai, T., Guo, R., Liu, Y., 2000. A binary tree representation of bend hierarchical structure based on Gestalt principles. In: *Proceedings of the 9th International Symposium on spatial Data Handling*, Beijing, pp. 230-243.

Ai, T., 2007. The drainage network extraction from contour lines for contour line generalization. *ISPRS Journal of Photogrammetry & Remote Sensing*, Vol. 62 (2), pp. 93-103.

AGENT, 1997. Map generalisation by multi-agent technology, ESPRIT project 1997-2000, http://agent.ign.fr/

Beard, M. K., 1991. Constraints on Rule Formation. In Buttenfield, B. P., and R. B. McMaster (eds), *Map Generalization: Making Rules for Knowledge Representation*, Longman Group, pp. 121-135.

Burghardt, D., S. Schmidt and J.E. Stoter, 2007. Investigations on cartographic constraint formalization. *The 10th ICA Workshop on Generalisation and Multiple Representation*.

Dutton G. and A. Edwardes, 2006. Ontological Modeling of Geographical Relationships for Map Generalization. *The 9th ICA Workshop on Generalisation and Multiple Representation*.

Egenhofer, M. and D. Mark, 1995. Modeling Conceptual Neighborhood of Topological Line-Region Relations. *Int. J. Geographical Information Science*, Vol. 9, No. 5, pp. 555-565.

Galanda, M., 2003. Modelling constraints for polygon generalization. *The 5th ICA Workshop on Progress in Automated Map Generalization*, Paris, France.

Gold, C. (1994). Three Approaches to Automated Topology, and How Computational Geometry Helps. In: *Proc. 6th International Symposium on Spatial Data Handling*, Edinburgh, pp. 145-158.

Haarslev, V., R. Möller and C. Schröder, 1994. Combining Spatial and Terminological Reasoning. In: *Proceedings of 18th German Annual Conference on Artificial Intelligence*, Vol. 861 of Lecture Notes in Artificial Intelligence, pp. 142-153.

Jones, C.B., G.L. Bundy, and J.M. Ware, (1995). Map Generalization with a Triangulated Data Structure. *Cartography and Geographic Information Systems*, 22(4), pp. 317-331.

Lüscher, P., D. Burghardt, and R. Weibel, 2007. Ontology-driven Enrichment of Spatial Databases. *The 10th ICA Workshop on Generalisation and Multiple Representation*, Moscow.

Mackaness, W.A. and A. Ruas, 2007. Evaluation in the Map Generalisation Process. In: *Generalisation of Geographic Information: Cartographic Modelling and Applications*, chapter 5, pp. 89-111.

Regnauld, N., 1996. Recognition of building clusters for generalization. In: M. Kraak M. Molenaar, Advances in: GIS Research, *Proceedings of 7th Int. Symposium on Spatial Data Handling*, vol. 2, Faculty of Geod. Engineering, Delft, The Netherlands, P. Session 4B.

Ruas, A. and J.P. Lagrange, 1995. Data and Knowledge Modeling for Generalization. In: Muller, J-C., Lagrange, J.-P., and Weibel, R. (eds.): *GIS and Generalization: Methodological and Practice*, pp. 73-90. London: Taylor & Francis.

Ruas, A., 1998. O-O constraints modeling to automate urban generalisation process. In: *Proceedings of the the 8th Spatial Data Handling Symposium*, Vancouver, pp. 225-235.

Ruas, A., 2001. Automatic generalisation project: learning process from interactive generalisation. (OEEPE publication), n°39.

Sester, M., 2000. Knowledge acquisition for automatic interpretation of spatial data. *Int. J. Geographical Information Science*, Vol. 14, No. 1, pp. 1-24.

Steiniger, S., 2007. Enabling pattern-aware automated map Generalization. Ph. D. thesis, Department of Geography, University of Zürich.

Stoter, J.E., Quak, C.W., Oosterom, P.J.M. van, Meijers, B.M., Lemmens, R.L.G, and Uitermark, H.T., 2007, Considerations for the design of a semantic data model for a multi-representation topographical database. In H Kremers (Ed.), Lecture notes in information sciences. Berlin: CODATA, pp. 53-71.

Stoter, J.E. et al., 2008. A study on the state-of-the-art in automated map generalisation. *The 11th ICA Workshop on Generalisation and Multiple Representation.*

Wang, Z. and J.-C. Müller, 1998. Line Generalization Based on Analysis of Shape Characteristics. *Cartography and Geographic Information Systems*, Vol. 25, No. 1, pp. 3-15.

Weibel R. (1996) "A Typology of Constraints of Line Simplification", In: *Proceedings 7th Int. Symposium on Spatial Data Handling* (Advances in GIS Research II), Delft, The Netherlands: Taylor & Francis, 9A.1–9A.14.

Weibel, R., and G. Dutton, 1998. Constraint-Based Automated Map Generalization. In: *Proceedings of the 8th Spatial Data Handling Symposium*, Vancouver, pp. 214-224.