# TOWARDS A SCALE DEPENDENT FRAMEWORK FOR CREATING VARIO-SCALE MAPS

Martijn Meijers[1,*], Peter van Oosterom[1], Radan Šuba[2], Dongliang Peng[1]

[1] Delft University of Technology, Faculty of Architecture, OTB Research for the Built Environment, GIS-technology – (b.m.meijers, p.j.m.vanoosterom, d.l.peng)@tudelft.nl
[2] WAG, Czech Republic – radan.suba@gmail.com

**Commission IV, WG IV/1**

**KEY WORDS:** cartographic generalization, vario-scale map, generalization process, map scale, scale range, scale dependent framework, feature class, generalization operation

**ABSTRACT:**

Traditionally, the content for vario-scale maps has been created using a ‘one fits all’ approach equal for all scales. Initially only the delete/merge operation was used to create the vario-scale data using the importance and the compatibility functions defined at class level (and evaluated at instance level) to create the tGAP structure with planar partition as basis. In order to improve the generalization quality other operators and techniques have been added during the past years; e. g. simplify, collapse (change area to line representation), split, attractiveness regions and the introduction of the concept of linear network topology. However, the decision which operation to apply has been hard coded in our software, making it not very flexible. Further, we want to include awareness of the current scale when deciding what generalization operation to apply. For this purpose we propose the scale dependent framework (SDF), which at its core contains the encoding of the generalization knowledge in the SDF conceptual model. This SDF model covers the representation of scale dependent class importance, scale dependent class compatibility values, scale dependent attractiveness regions and last but not least specification of generalization operations that are scale and class dependent. By changing the settings in the SDF configuration and re-running the vario-scale generalization process, we can easily experiment in order to find best settings (for specific map user needs). In this paper we design the SDF conceptual model and explicitly motivate and define the scope of its expressiveness. We further present the improved scale dependent tGAP creation software and present initial results in the form of better created vario-scale map content.

## 1. INTRODUCTION

For creating a *vario-scale map*, we start from a large scale vector base map with polygon objects that form a planar partition (every part of the mapped domain assigned to exactly one polygon, without gaps or overlaps). A vario-scale map changes with respect to the scale, where a delta change in the scale results in a corresponding small change on the map. In other words, the smaller the delta, the more slightly the change on the map. To implement a vario-scale map, we use automated generalization. The generalization process developed by us leads to a topological Generalized Area Partition (tGAP) data structure, from which we can produce maps that change in a gradual way. The changes were even made more gradual via the Space Scale Cube (SSC), which uses the scale as a dimension to enable smooth transitions (van Oosterom, 2005; Meijers and van Oosterom, 2011; van Oosterom and Meijers, 2014).

Originally, this generalization process consists of selecting the least-important area object in the map, then decides on the most compatible neighbor, and merges these two neighboring areas into one new area. For deciding the least-important object an importance function is used, which allows to put more emphasis on areas of certain feature classes (by increasing their weight and thus importance) and/or size of the feature. For making a decision on what is an attractive and compatible neighbor to merge with, we use a compatibility matrix. This matrix is based on the compatibility of feature classes. The compatibility of two neighboring objects is determined based on this matrix and the length of the common boundary. At the earlier stage of the research, only one generalization operation was available: merging of area objects.

The more recent developments of the tGAP creation process have focused on making more generalization operations available: A split operation, to divide the area of a polygonal geometry over its neighbors and a line simplification operation that allows to simplify boundaries between areas (and at the same time keep topology safe, i.e. not introducing unwanted intersections between boundaries). Each operation makes use of compatibilities and importance values in its own way. Further refinement of the generalization process included three main directions. The first one is to explicitly model networks such as roads, waterways, and railroads even if elements are represented by areas on large scale representation. The second one is to include features which are represented by lines (in addition to the areas, cf. Šuba et al., 2015, 2016). The third one is to use additional regions in which features are more attractive to each other when they reside in one region compared to features that reside in two different regions (Šuba, 2017). Note that these regions can originate from different sources, such as an external source of a smaller-scale map or the result of an optimization process. For example, Haunert and Wolff (2010) used an integer linear program to compute an optimization of area aggregation for a target scale, which does not generate maps at intermediate scales.

These latest developments lead to tailor made, but hard coded solutions for dealing with road/water elements in the map — from large scale (road/water elements represented as areas) to very

[*] Corresponding author

small scale (elements as lines). To make tuning vario-scale generalization easier, a more flexible approach is required. We like to orchestrate and steer which decisions to take at every step of the generalization process. Such an orchestration depends on the intended scale for which the generalized data is to be used.

In this paper, we present the new scale dependent framework (SDF) to produce content of a vario-scale map. The framework offers a collection of methods to steer and orchestrate the generalization process. The first problem we tackle is that for each step throughout the process we explain how to estimate what the approximate map scale will be. So far no intended map scale was known (just importance of objects – mainly related to their size and importance of the class). This estimation is useful for selecting a relevant generalization operation (i.e. merge or split). Secondly, the more important part of our contribution is that we allow the producer of a vario-scale map to specify what generalization decisions to take (per scale range and per feature class). This, by means of configuration, instead of hard coding the decisions. Vario-scale map producers can specify:

1. Per feature class and per scale range to have a different preference for which generalization operation to apply (merge or split);

2. Per feature class to have different importance values per scale-range (note, these are only applied when recomputing importance, after split/merge operation of new feature, not for all others as these are in priority queue, and would be heavy operation to perform);

3. Have compatibility matrix with different values per scale-range (having scale ranges available can change the rules for attractiveness between classes per range, e.g. make it less attractive to merge land and water for smaller scale maps).

4. Define regions that are scale dependent, to influence finding the most compatible neighbour (e. g. while merging).

Section 2 presents related work. Section 3 introduces the conceptual model of the SDF and illustrates how it is populated. Section 4 presents how the SDF is used in a generalization step of the tGAP creation process. Section 5 shows some initial results and Section 6 concludes the paper by a discussion and giving a description of future work.

## 2. RELATED WORK

As illustrated by Müller et al. (1995) and Weibel (1997), geographical information is scale dependent. We should provide users appropriate level of details when users are zooming in or out. Brewer and Buttenfield (2007) designed the *ScaleMaster* to guide symbol modification and geometry change in order to generate a specific-scale map from multiple databases. In addition to geometry change, they emphasized the use of symbol modification to better reduce overall workloads. Their guiding principle, when users zooming out, was to avoid visual clutter while making sure that the geographical information was easily perceived. As ScaleMaster is a guide to read, cartographers still have to produce maps manually. Touya and Girres (2013) developed ScaleMaster 2.0 to realize automatic generalization according to the guide.

Regarding how much content we should keep for a specific scale, Töpfer and Pillewizer (1966) proposed the *radical law*. By using some parameters, this law can take into account whether or not the symbols are exaggerated and whether the symbols are linear or areal. To support vario-scale maps, Peng (2017, chapter 4) continuously generalized buildings to built-up area by growing. Also they aggregated the buildings that became too close. In the experiment, their number of buildings decreased quite consistent with the radical law. However, the total area of buildings increase at the beginning, and then decreased almost linearly with respect to the denominator of the scale. This pattern is very different from the radical law. Jiang (2015) proposed to keep content according to the so-called *fractal nature*, which means there are far more small features than large ones. He argued that the fractal nature is the root of the radical law of Töpfer and Pillewizer (1966). In order to implement the fractal nature, he employed *head/tail* breaks, where head means those features that are larger than the average and tail means smaller. When zooming out, he suggested keeping only the head features for the next smaller scale map. Šuba et al. (2016) observed the quantity of features during their continuous generalization of road network. The analyzed the numbers of building faces, road faces, road edges, and terrain faces. Also they depicted the area sizes of buildings, roads, terrains, and waters. Recently, Karsznia and Weibel (2017) used machine learning for settlement selection. Their method learned selection rules from a pair of maps: one map at scale $1 : 250{,}000$ and a manually generalized version of the map at scale $1 : 500{,}000$. According to the learned rules, their method automatically generalized the original map to obtain a new map also at scale $1 : 500{,}000$. By comparing to the manual version, the automatic one has accuracy $75.6\%$ in the worst case, which means that machine learning is a promising tool for generalization. Chen et al. (2009) considered the density as an important aspect for maps of road networks. They managed to control the densities of different local regions by using a mesh-based method. Moreover, Touya and Reimer (2015) inferred the scales for the features on OpenStreetMap. The features can have different *geometric precision* depending on the volunteers. Therefore, it is necessary to know on what scales we should present the features.

## 3. CONFIGURATION OF THE GENERALIZATION PROCESS

To make tuning tGAP creation easier, a more flexible approach is required. We like to orchestrate and steer which decisions to take at every step of the generalization process. Such an orchestration depends, among others, on the intended scale for which the generalized data is to be used. The scale dependent framework (SDF) has at its kernel a conceptual model able to represent all information needed to describe what generalization operation is used in what situation (for which class at which scale range). Figure 1 depicts the UML diagram. The core class in the SDF model is the FeatureClass representing the identifiers, descriptions and the roles (in the codeList roleType with values: areal, linear or point) of the different types of classes used in the map. The instances of the classes may exist throughout all scales. The classes GeneralizationAction, ClassCompatibilityMatrix and AttractivenessRegion (all subclasses of the abstract ScaleRange class) specify the knowledge facts needed for generalization which are possibly scale dependent.

The SDF conceptual model can be converted into a technical model for a database implementation (SQL DDL schema), which is then next populated with the SDF records, before the actual

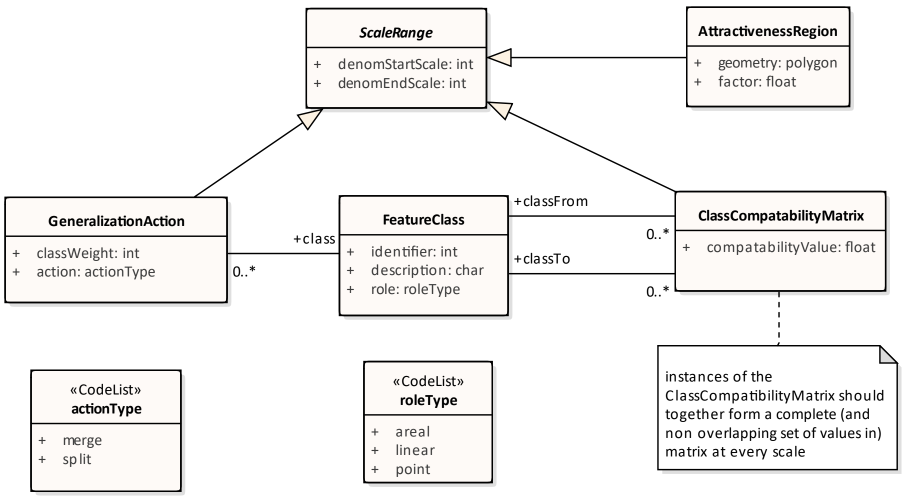


Figure 1. The conceptual model of the SDF.

| class | description | role |
|---|---|---|
| 10310 | main road | linear |
| 10410 | regional road | linear |
| 10510 | local road | linear |
| 10600 | street | linear |
| 10710 | road for parking | areal |
| 10780 | all other features | linear |
| 12400 | stream | linear |
| 12500 | lake | areal |
| 13000 | buildings | areal |
| 14010 | arable land | areal |
| 14040 | orchard | areal |
| 14060 | mixed forest | areal |
| 14080 | deciduous forest | areal |
| 14090 | coniferous forest | areal |
| 14130 | grassland | areal |
| ... | ... | ... |

Table 1. Example 'feature class' table. The classes are taken from Top10NL (Dutch topographic data set intended to be used at scale 1:10,000).

vario-scale map generalization process is started. This data model allows to store the essential information that we need while generalizing. Note that we could also convert the SDF conceptual model in an XML schema for the exchange format (another technical model) intended for the exchange of the essential generalization information.

Our scale dependent framework is stored by means of several database tables. Attributes and named associations become columns in the database tables. Inside the tables we store the configuration for the generalization process and the role features play on the map. The configuration is expressed based on *class* information, and what should happen with a certain class of objects at a certain scale. At instance level decisions will be taken during the generalization (e. g. looking at connectivity of linear features), but at the moment there is less direct influence possible for an end user (i.e. a new generalization operation should be implemented in system).

**Table: feature class** Within this table all feature classes are stored that are present in the source data set, together with a human readable description.

The table stores three fields:

1. *class*: Integer that uniquely identifies the class
2. *description*: Human description of the feature class
3. *role*: Role that the objects of this class play on the map. When an object belongs to a class that represents objects with nature of linear network features (for example roads or water ways), different kinds of generalization operation can be executed (executing a more specialized merge operation that also takes into account this network, trying not to break it apart)

Table 1 shows some example records.

**Table: generalization actions.** This is the core table for recording which class should undergo which generalization operation at what scale (see Table 2).

The fields that this table contains are:

1. *class*: Integer, pointing to feature class table
2. *weight*: Integer, relative importance value for this feature class class
3. *start scale*: Denominator of start of scale range
4. *end scale*: Denominator of end of scale range
5. *action*: Generalization action to execute [merge — mergeLinear — split]

Importance values are related to the usage type of the map (gives a relative importance between classes, to be able to express for example in the generalization process that water ways should be preserved longer).

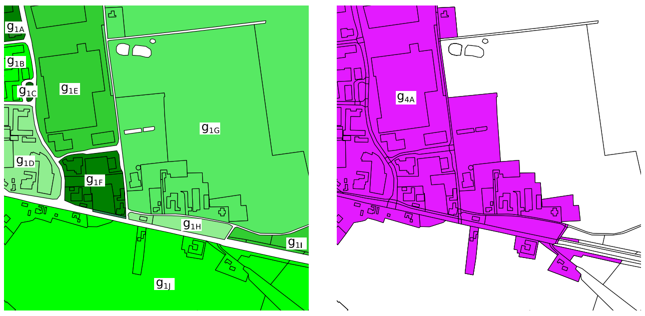
Per class a complete scale range should be defined, without gaps, and overlaps (and cover all scales: $1 : 1 — 1 : \infty$). This means that at least per feature class 1 entry is required in this table. For example, Table 2 shows that main road objects (class 10310) will be merged with a compatible neighbour in the range 1:1 to 1:30,000, and after scale 1:30,000 will be split over their neighbours. To describe this behaviour two entries are made in the generalization actions table.

| class | weight | start scale | end scale | action |
|---|---|---|---|---|
| 10310 | 100 | 1 | 30,000 | merge |
| 10310 | 100 | 30,000 | $\infty$ | split |
| 10410 | 100 | 1 | 20,000 | merge |
| 10410 | 100 | 20,000 | $\infty$ | split |
| 10510 | 10 | 1 | 20,000 | merge |
| 10510 | 10 | 20,000 | $\infty$ | split |
| 10600 | 10 | 1 | 15,000 | merge |
| 10600 | 10 | 15,000 | $\infty$ | split |
| 10710 | 1 | 1 | 20,000 | merge |
| 10710 | 1 | 20,000 | $\infty$ | split |
| 10780 | 1 | 1 | $\infty$ | merge |
| 12400 | 1000 | 1 | $\infty$ | split |
| 12500 | 1000 | 1 | $\infty$ | split |
| 13000 | 500 | 1 | $\infty$ | merge |
| 14010 | 1 | 1 | $\infty$ | merge |
| 14040 | 1 | 1 | $\infty$ | merge |
| 14050 | 1 | 1 | $\infty$ | merge |
| 14060 | 1 | 1 | $\infty$ | merge |
| 14100 | 1 | 1 | $\infty$ | merge |
| 14120 | 1 | 1 | $\infty$ | merge |
| 14130 | 1 | 1 | $\infty$ | merge |
| ... | ... | ... | ... | ... |

Table 2. Generalization actions. Parameters for the generalization process as can be specified. Note that the start scale and end scale columns contain the specification which generalization operation to carry out for which scale range. In case a polygon with the classification 'local road' (feature class 10510) is selected to be generalized, the operation that will be applied is merge (in the scale range 1:start scale – 1 : 20,000) or split (1 : 20,000 – 1 : $\infty$), dependent on the current estimated display scale.

| code-from | code-to | compatibility | start scale | end scale |
|---|---|---|---|---|
| 10310 | 10310 | 10000 | 1 | 20,000 |
| 10310 | 10310 | 1000 | 20,000 | $\infty$ |
| 10310 | 10410 | 1 | 1 | $\infty$ |
| 10410 | 10310 | 1 | 1 | $\infty$ |
| ... | ... | ... | ... | ... |
| 12500 | 14010 | 10 | 1 | 20,000 |
| 12500 | 14010 | 1 | 20,000 | $\infty$ |
| ... | ... | ... | ... | ... |
| 14080 | 14060 | 100 | 1 | $\infty$ |
| 14080 | 14010 | 10 | 1 | $\infty$ |
| 14080 | 13000 | 0.1 | 1 | $\infty$ |
| ... | ... | ... | ... | ... |

Table 3. Example of a scale dependent compatibility matrix.



(a) Attractiveness regions are formed by removing the road objects and uniting all connected components.

(b) The objects (in purple) belong to an urban region (obtained from an external source).

Figure 2. Examples of attractiveness regions

**Table: class compatibility matrix** The creation of the tGAP structure is driven by two main aspects. First, the global order of the features based on importance value, i.e. in every step the least-important feature is selected and processed (merge or split). Second, selection of the most compatible neighbour. To influence the decision, the compatibility matrix is used (see Table 3). Within the scale dependent framework, every compatibility value that can be specified, also gets an associated scale range. The result is that compatibility between classes can be changed (e. g. water and land are compatible for large-scale maps, but for smaller scales land and water are made quite incompatible, to prevent water taking over too much land).

The fields present in the class compatibility matrix are:

- *class-from*: Integer, pointing to feature class table
- *class-to*: Integer, pointing to feature class table
- *comp-value*: Float, compatibility (the higher, the more compatible)
- *start scale*: Denominator of start of scale range
- *end scale*: Denominator of end of scale range

In some cases map features, such as buildings, gardens and sheds forming built-up area, should stay visible longer on the map in aggregated form. Here we propose an additional tool for steering how these features will be aggregated. The idea is based on defining coarse regions, depicted by polygons. For every region a scale range is associated when the region is to be used. For every map object a relationship is determined with the relevant regions. If a map object overlaps with the polygon it is said to be a member of the region. The fact that objects are, or are not, a member of the same region will have an influence on the calculated compatibility value, by determining a scaling factor to be used for the resulting compatibility value. The regions can stem from external sources or can be generated from the source data set by geometric processing (e.g. leaving out the road network and making a union of the resulting polygons). Note that the regions can also be overlapping and one object can be member of multiple regions.

Figure 2 gives an illustration of objects being part of different regions, where the green regions are defined by leaving out the road objects and the purple region is a definition of urban area, obtained from an external source.

**Table: attractiveness regions** For storing the information about the regions, we use the following fields:

- *geometry*: Polygon that describes a region its extent
- *start scale*: Denominator of start of scale range when the region is to be used
- *end scale*: Denominator of end of scale range
- *factor*: Multiply with this factor the compatibility value when objects are in the same group

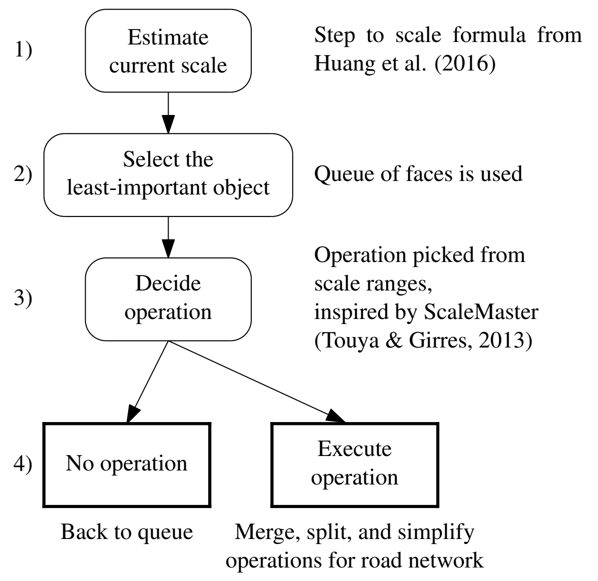

1) Estimate current scale
Step to scale formula from Huang et al. (2016)
2) Select the least-important object
Queue of faces is used
3) Decide operation
Operation picked from scale ranges, inspired by ScaleMaster (Touya & Girres, 2013)
4) No operation
Execute operation
Back to queue
Merge, split, and simplify operations for road network

Figure 3. Outline of actions for one generalization step.

During the phase for determining the compatibility value, a factor larger than 1 is used, when an object is in the same region as one of its neighbours (neighbour is more attractive). The regions make it possible to better steer what should happen for example in case of rural and urban areas, i.e. to prevent merges from rural to urban regions.

## 4. OUTLINE OF A GENERALIZATION STEP IN THE PROCESS

The whole generalization process is composed of a sequence of steps. Every step consists of multiple actions, where decision making is included. Repeating theses steps results in simpler and simpler maps. So far, making decisions in the vario-scale generalization process was based on feature class, geometry configuration, etc. The information about current scale, for which specific objects were processed, was never included, until now. We propose an algorithm to improve automatic generalization for every generalization step based on estimated target scale. Figure 3 shows the outline. One generalization step is composed of the following sequence of steps: Estimate target scale, select the least-important object, decide on if we generalize, and, if so, execute an generalization operation.

### 4.1 Estimate current scale

With respect to the relation between the tGAP structure and the scale, we employed map generalization with tGAP to produce a sequence of successively more generalized maps so that these maps go well together, similar to Chimani et al. (2014) and Peng et al. (2017). Instead of considering each level of generalization independently (current practice), we consider the sequence of maps as a whole. For example, a removed building should not appear again when users are zooming out. In our perspective, each intermediate map is as important as the final map. Our proposed strategy starts by estimating a scale for which the operation is performed. This estimation originates from recent development of a web client for vario-scale maps, see Appendix of Huang et al. (2016). When a user requests a map in the viewer, a scale has to be determined. We assume that at every tGAP step there is one fewer areal object on the map. The relationship between a tGAP

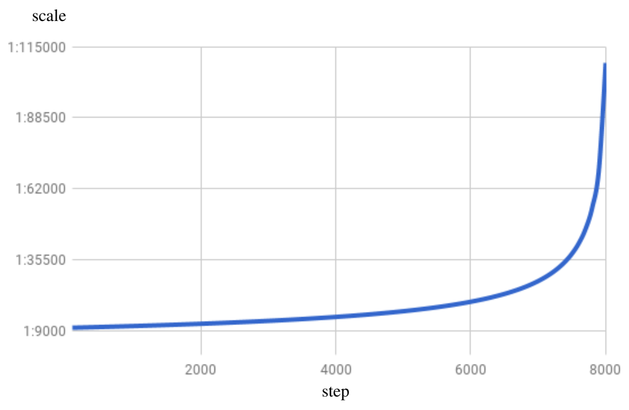


Figure 4. Graph of relationship between number of objects and scale for sample data, where $S_\mathrm{b} = 10{,}000$ and $N_\mathrm{b} = 8{,}068$.

step and the estimated denominator of scale can be described by the following formula:

$$S_\mathrm{t}(Q) = S_\mathrm{b}\sqrt{\frac{N_\mathrm{b}}{N_\mathrm{b} - Q}} \quad (1)$$

where $S_\mathrm{b}$ is the denominator of base/start map’s scale, e. g., $S_\mathrm{b} = 10{,}000$ for the instance shown in Figure 4. $N_\mathrm{b}$ is the total number of objects on the base map. Variable $Q$ is the number of steps. Note that for a given data set, only $Q$ is a variable while $N_\mathrm{b}$ and $S_\mathrm{b}$ are constants. For example, Figure 4 shows the relationship between steps and estimated scales for a real data set.

### 4.2 Select the least-important object

In our approach every generalization step picks the corresponding least-important object in tGAP to perform a generalization operation The least-important object is defined by the importance function: $\text{Importance}(a) = \text{Area}(a) * \text{WeightClass}(a)$. In other words, the importance of a feature is based on its size and the weight of the class to which it belongs. Before the generalization process starts, we compute importance values for all the objects and store the objects in a priority queue according to the importance values. Then we are able to process the objects from the least-important one to the most-important one. One option to deal with the resulting object of a generalization operation is to put the object back in the priority queue according to the sum of the two old importance values (in case of a merge operation). It would also be possible to change the importance values for all objects after each generalization step (as we have a new estimated map scale). This, however, is expensive due to the fact that we have a priority queue that orders all objects globally, which means we have to reorder after generalization step. Another less expensive, but still scale aware, operation is to recalculate the importance value of the new object and then insert the new object in the queue according to its new importance value. For this the class weight, that is now scale dependent, can be used.

### 4.3 Decide operation

At this phase of the process, the least-important object and the estimated scale are known. Depending on the relevant role and action, obtained via the feature class and estimated scale, we can pick the right generalization operation. Also the attractiveness regions for this map scale are used in deciding what the most compatible neighbour is. Moreover, a merge operation can be executed in some variants: 1. a merge based on both the length of

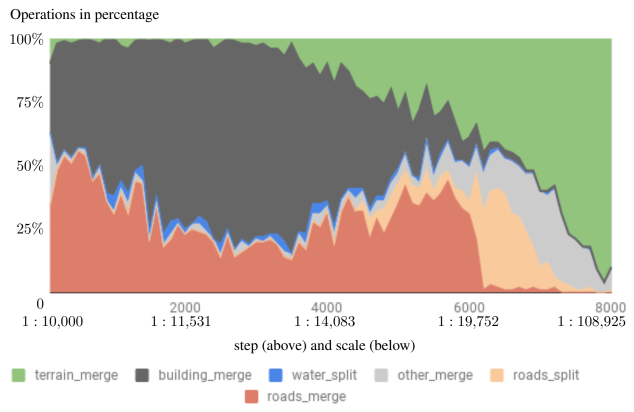


Figure 5. The percentages of operations executed per 100 steps in the tGAP creation process.

common boundary between the faces and the compatibility matrix, 2. a merge taking into account the linear network around area objects, cf. Šuba et al. (2015).

## 4.4 No operation

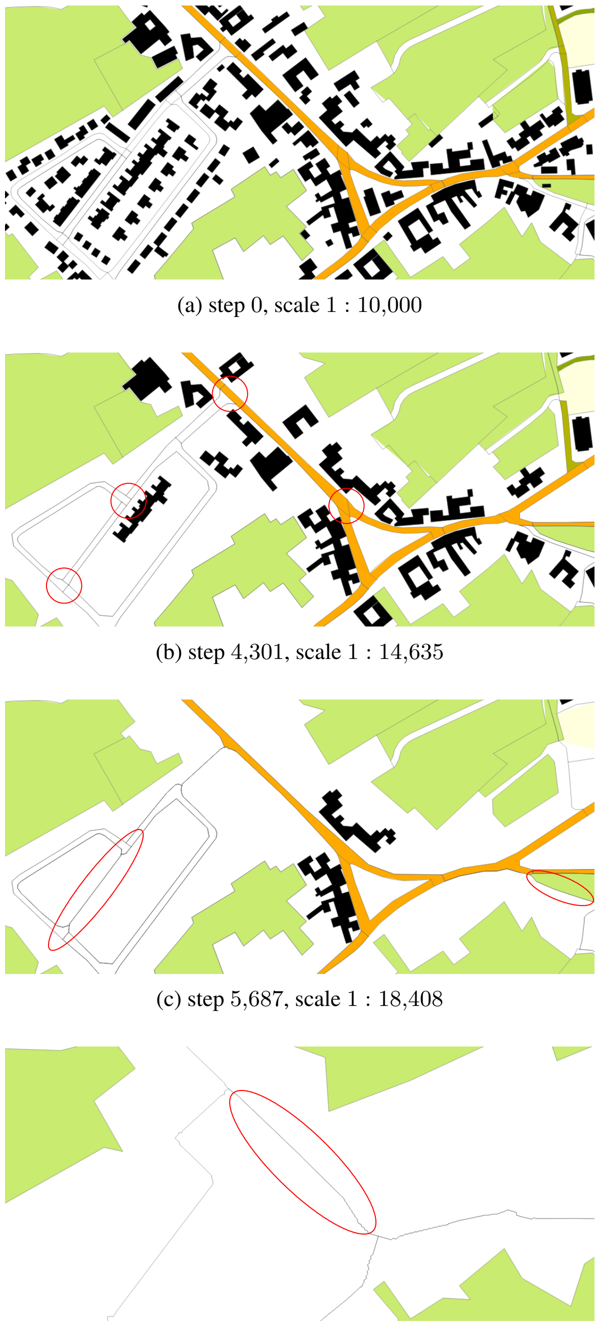
There are situations when a selected operation is better to postpone because a better operation may exist for a specific geometrical configuration. Those situations in principle contradict an original intended operation (as specified in Table 2). We deal with such a situation in this section. Currently there is only one option implemented. That is, if a face does not satisfy a certain condition, then we raise its importance and put it back in the priority queue. No operation like merge or split is performed for this tGAP step, and the face will be processed later.

## 4.5 Execute operation

The last step of the proposed process for one generalization step is to perform the selected operation and deal with all consequences (change of geometry or attributes). Only at this stage is the tGAP structure changed. Our current prototype provides merge, split, and simplification as boundary operations.

## 5. RESULTS

Currently we are implementing the new framework for scale dependent tuning of the generalization process. The preliminary results we have obtained so far show that the rules as specified are taken into account and are effective. From Figure 6a to Figure 6b, we can see some merges of roads (the red circles in Figure 6b mark some examples). The red area in Figure 5 represents the amount of roads' merging. That these merge operations should take place is specified in Table 2. The rules specify that all road objects should be merged from scale $1 : 1$ up to (at least) scale $1 : 15{,}000$. By comparing Figure 6b with Figure 6c, one can observe that some streets (class 10600) are split as of step $5{,}687$ (for example, the road marked by the red ellipse in Figure 6c). This phenomenon is also illustrated in Figure 5, where the orange area starts to emerge and then grows (thus roads are being split, starting at the specified scale). Also we can see that the main roads continue merging (red area in Figure 5). This again reflects the rules as specified in Table 2. At step $7{,}664$ also the main roads have been split (see for example the road marked by ellipse in Figure 6d)). Moreover, all the buildings have been removed. This is why the black area in Figure 5 degenerates to a line.



(a) step 0, scale $1 : 10{,}000$

(b) step $4{,}301$, scale $1 : 14{,}635$

(c) step $5{,}687$, scale $1 : 18{,}408$

(d) step $7{,}664$, scale $1 : 44{,}688$

Figure 6. Initial results. Merging and splitting of road objects follows the generalization actions, dependent on the estimated scale, specified in Table 2.

## 6. DISCUSSION AND FUTURE WORK

We have presented how we have made the tGAP creation process more scale aware, and that, dependent on the estimated map scale we can express better than before what generalization operation should happen. We have introduced the SDF conceptual model, which makes it possible to store the information of making these scale dependent decisions. Moreover, our initial experiments indicate that this information is used effectively within the resulting tGAP creation process. The concept of attractiveness regions (groups of objects) also fits well in the scale dependent framework for creating vario-scale maps: Different regions may be switched on and off depending on the scale (when computing compatibility). From the experiment we learned that the new framework gives more grip on the generalization process, and thus on the resulting vario-scale map content. Therefore, this framework is promising.

Having the scale dependent framework available makes it possible to tune the generalization process, though the approach should now be tested more rigorously. This will require iterative cycles of specifying, visualizing and evaluating the resulting maps, and possibly adjusting the rules as specified. The SDF allows much easier tuning of the generalization process at an higher level: Instead of changing hard-coded values in the source code it is now possible to just edit the configuration and re-run the generalization process. By testing and assessing the resulting vario-scale representation, it is easy to adapt certain aspects of the generalization process to get better vario-scale content. According to the configuration, this adaptation can be related to certain types of features or operations within a certain scale range. However, as this tuning may require quite some work and is likely to be similar for data sets with similar nature, it would be good to be able to share this knowledge/information. Therefore, we could make a predefined set of abstract superclasses and related operations available in a default configuration. We could set a number to each of the abstract superclasses, as this would fit in the configuration tables. By mapping from the concrete feature classes of a data set to the abstract superclasses, we can share the information on the generalization process. As a result, data sets from different providers could be generalized in similar way, e. g. topographic data sets from Germany and from The Netherlands could share the same set of generalization rules. Next to this, as map scale is now made explicit and known with each generalization step, we can simplify the boundaries between areas in a more granular way than before. For simplifying boundaries' geometry, this implies that we know how detailed an edge is and then we can determine the boundaries to be simplified. Furthermore, we should not over-simplify the boundaries. Finally, we now have only area features in the queue (no lines). In the scale dependent framework we would also like to have line features in the queue and to associate which generalization operation is needed when the least-important line feature is selected. We should investigated if we want to put line features and area features in the same queue, or we prefer to have two separate queues. Despite these new future work items, the current initial version of the scale dependent framework for the tGAP/SSC creation is a big step forward.

## ACKNOWLEDGEMENTS

This research has been carried out in the project ‘Vario-scale representation of Shenzhen maps’, which was financially supported by Shenzhen Research Center of Digital City Engineering. The authors gratefully acknowledge this support.

## REFERENCES

Brewer, C. A. and Buttenfield, B. P., 2007. Framing guidelines for multi-scale map design using databases at multiple resolutions. *Cartography and Geographic Information Science* 34(1), pp. 3–15.

Chen, J., Hu, Y., Li, Z., Zhao, R. and Meng, L., 2009. Selective omission of road features based on mesh density for automatic map generalization. *International Journal of Geographical Information Science* 23(8), pp. 1013–1032.

Chimani, M., van Dijk, T. C. and Haunert, J.-H., 2014. How to eat a graph: Computing selection sequences for the continuous generalization of road networks. In: *Proc. 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACMGIS'14)*, Dallas, TX, USA, pp. 243–252.

Haunert, J.-H. and Wolff, A., 2010. Area aggregation in map generalisation by mixed-integer programming. *International Journal of Geographical Information Science* 24(12), pp. 1871–1897.

Huang, L., Meijers, M., Šuba, R. and van Oosterom, P., 2016. Engineering web maps with gradual content zoom based on streaming vector data. *ISPRS Journal of Photogrammetry and Remote Sensing* 114, pp. 274–293.

Jiang, B., 2015. The fractal nature of maps and mapping. *International Journal of Geographical Information Science* 29(1), pp. 159–174.

Karsznia, I. and Weibel, R., 2017. Improving settlement selection for small-scale maps using data enrichment and machine learning. *Cartography and Geographic Information Science* 45(2), pp. 111–127.

Meijers, M. and van Oosterom, P., 2011. The space-scale cube: An integrated model for 2D polygonal areas and scale. In: E. Fendel, H. Ledoux, M. Rumor and S. Zlatanova (eds), *ISPRS Archives Volume XXXVIII-4/C21, 28th Urban Data Management Symposium*, Delft, pp. 95–101.

Müller, J.-C., Weibel, R., Lagrange, J.-P. and Salgé, F., 1995. Generalization: State of the art and issues. In: J.-C. Müller, J.-P. Lagrange and R. Weibel (eds), *GIS and Generalization: Methodology and Practice*, GISDATA, Taylor & Francis, London, UK, chapter 1, pp. 3–17.

Peng, D., 2017. An Optimization-Based Approach for Continuous Map Generalization. PhD thesis, University of Würzburg, Würzburg, Germany.

Peng, D., Wolff, A. and Haunert, J.-H., 2017. Using the A$^\star$ algorithm to find optimal sequences for area aggregation. In: M. P. Peterson (ed.), *Proc. 28th International Cartographic Conference (ICC'17), Advances in Cartography and GIScience*, Lecture Notes in Geoinformation and Cartography, Springer, Washington DC, USA, pp. 389–404.

Šuba, R., Meijers, M. and van Oosterom, P., 2015. Large scale road network generalization for vario-scale map. In: *Proceedings of the 18th ICA Workshop on Generalisation and Multiple Representation*, Rio de Janeiro, p. 10.

Šuba, R., Meijers, M. and van Oosterom, P., 2016. Continuous road network generalization throughout all scales. *ISPRS International Journal of Geo-Information* 5(8), pp. 21.

Töpfer, F. and Pillewizer, W., 1966. The principles of selection. *The Cartographic Journal* 3(1), pp. 10–16.

Touya, G. and Girres, J.-F., 2013. ScaleMaster 2.0: A ScaleMaster extension to monitor automatic multi-scales generalizations. *Cartography and Geographic Information Science* 40(3), pp. 192–200.

Touya, G. and Reimer, A., 2015. Inferring the scale of OpenStreetMap features. In: J. Jokar Arsanjani, A. Zipf, P. Mooney and M. Helbich (eds), *OpenStreetMap in GIScience: Experiences, Research, and Applications*, Lecture Notes in Geoinformation and Cartography, Springer, pp. 81–99.

van Oosterom, P., 2005. Variable-scale topological data structures suitable for progressive data transfer: The gap-face tree and gap-edge forest. *Cartography and Geographic Information Science* 32(4), pp. 331–346.

van Oosterom, P. and Meijers, M., 2014. Vario-scale data structures supporting smooth zoom and progressive transfer of 2D and 3D data. *International Journal of Geographical Information Science* 28(3), pp. 455–478.

Šuba, R., 2017. Design and development of a system for vario-scale maps. PhD thesis, Delft University of Technology.

Šuba, R., Meijers, M. and van Oosterom, P., 2015. Large scale road network generalization for vario-scale map. In: *Proc. 18th ICA Workshop on Generalisation and Multiple Representation*, Rio de Janeiro, Brazil. 21 pages.

Weibel, R., 1997. Generalization of spatial data: Principles and selected algorithms. In: M. van Kreveld, J. Nievergelt, T. Roos and P. Widmayer (eds), *Algorithmic Foundations of Geographic Information Systems*, Lecture Notes in Computer Science, Vol. 1340, Springer, chapter 5, pp. 99–152.