EFFICIENT AREA AGGREGATION BY COMBINATION OF DIFFERENT TECHNIQUES

Jan-Henrik Haunert
Institute of Cartography and Geoinformatics
Leibniz Universität Hannover
Appelstraße 9a, 30167 Hannover, Germany
jan.haunert@ikg.uni-hannover.de

Abstract

When reducing the scale of a topographic database, some areas of the data set become too small for representation and need to be aggregated with others, unintentionally but unavoidably leading to changes of some areas' land cover classes. In this paper, we approach this problem by optimization: Given a planar subdivision containing areas of different land cover classes, the problem is to aggregate areas into contiguous regions and to define the class for each region, such that each region satisfies a size threshold and the overall class change is minimal. In an earlier paper we proved the NP-hardness of this problem, presented a method by mixed-integer programming and introduced several heuristics. Our tests revealed that, even with the defined heuristics, our method does generally not allow to solve problem instances of more than 400 areas. Defining that compact shapes are preferred can enhance the problem statement. However, this does not change the complexity of the problem.
In this paper we present a new efficient heuristic for the problem. Our approach is to locally introduce intermediate levels of details. Steps between these scales can be processed using our previously presented method. This approach allows processing large data sets – a complete map sheet of the German topographic map at scale 1:50.000 was processed to meet the requirements for the scale 1:250.000. We show that our method generalizes an existing iterative algorithm for the same problem and compare the results being obtained with different settings of our method. Compared with the existing iterative algorithm, our method resulted in 27,4% less change of land cover classes.

KEY WORDS: AREA AGGREGATION, LAND COVER DATA, GENERALIZATION

1 Introduction

Topographic databases often contain areas of different land cover, which define a subdivision of the plane (Figure 1). In order to satisfy defined minimal dimensions for a smaller scale, areas need to be aggregated. In an earlier paper we proposed a method for this generalization problem based on mixed-integer programming, a combinatorial optimization technique (Haunert and Wolff, 2006). The method guarantees contiguous aggregates of sufficient size (referred to as regions) and minimizes a combined cost for class change and non-compactness of resulting shapes. A result with this setting is shown in Figure 4. Though the obtained results were very promising, the required running time turned out to be prohibitive for cartographic production.
This paper presents a new improvement of this method, which enables the processing of large data sets. The proposed algorithm locally introduces intermediate levels of details. To go from one level to the next, our existing optimization method is applied. We show how to define the intermediate scales, such that, in each iteration, no more than a user specified number of areas k needs to be processed at once.
In contrast to our global approach, which reaches the required target scale in a single step, our new approach is hierarchical, i.e. the aggregates in the final scale are composed of smaller aggregates of areas. Unfortunately, this approach does not give a theoretical guarantee of performance. However, we legitimate it by showing that it generalizes a well-known and accepted iterative method for the same problem. To be precise, by setting $k = 1$ the algorithm

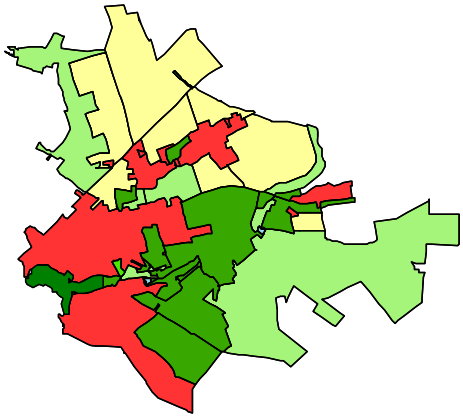

Figure 1: An example from the German topographic data set ATKIS at scale 1:50.000.

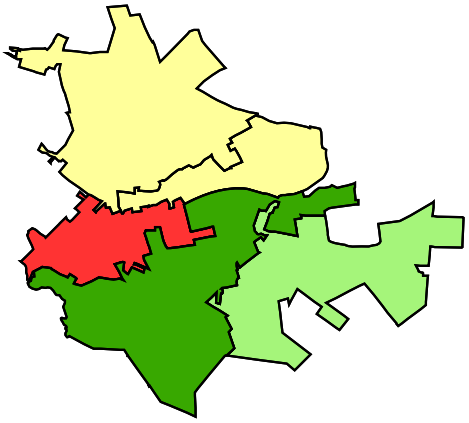

Figure 2: A result of the iterative algorithm at scale 1:250.000.

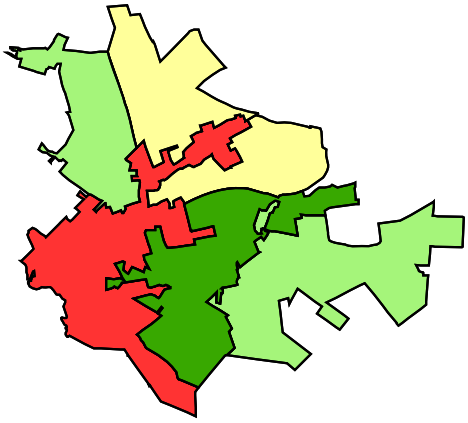

Figure 3: A result at scale 1:250.000 with minimum class change.

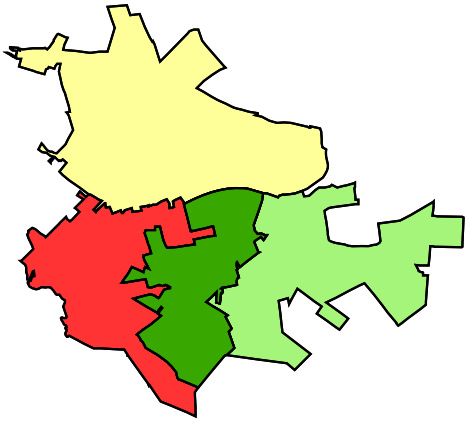

Figure 4: A result at scale 1:250.000 minimizing a combined cost for class change and non-compactness.

performs exactly as the iterative method proposed by van Oosterom (1995) for setting up the so-called GAP-tree. It is assumed that by increasing the number of areas that are processed at once, the result becomes more similar to the global optimum. An additional benefit is the tree structure of the result, which presumably can be exploited for adaptive zooming and incremental updating.

The method was tested for a data set corresponding to a complete map sheet of the German topographic map at scale 1:50.000. This was generalized to scale 1:250.000. Different settings were applied, clearly confirming the assumption that the quality increases for higher values of *k*. The paper is structured as follows: Section 2 reviews related work by other researchers. In Section 3 our previously presented global optimization approach is briefly described. Section 4 presents the new improvement of this method. Finally, Section 5 gives a conclusion.

2 Related work

The generalization of polygon maps requires solutions for multiple subproblems. Beside aggregation, the most important problems are class abstraction (van Smaalen, 2003), area collapse (Haunert & Sester, 2007) and line simplification (de Berg et al., 1995). In recent years, the integration of multiple operators for polygon generalization has been approached by

application of multi-agent systems (Galanda, 2003). However, due to the resulting computational complexity, these have been controlled by application of a hill-climbing approach. A problem of this is that it typically gets stuck in local optima (Michalewicz & Fogel, 2004). Deciding between a more integrated approach and more complex optimization techniques is clearly a trade-off. According to our experiences, iterative greedy methods do not produce aggregation results of high quality. We therefore propose to solve the aggregation problem independently by application of a more sophisticated optimization technique. To explain the observed shortcomings of an iterative method, we describe the algorithm of van Oosterom (1995) in Section 2.1.

2.1 An iterative approach to area aggregation

Van Oosterom (1995) defines an algorithm for the aggregation of areas as follows:

- In each iteration, the feature with lowest importance is selected.
- One of its neighbours is chosen according to a collapse function.
- The selected feature is merged with this neighbour and the next iteration is processed.

In order to satisfy minimal dimensions for a target scale, the importance of a feature can be defined by its area. In this case, the algorithm can be terminated if all areas satisfy the required size threshold.
Many proposed algorithms are specializations of this general method. Jaakkola (1997) uses the method within a more comprehensive generalization framework for raster based land cover maps. Podrenek (2002) discusses preferences for merges, which reflects the collapse function. Bobzien (2001) presents results with different definitions of the processing order. Generally, semantic similarities of classes, boundary lengths and area sizes are considered as criteria that need to be incorporated into the collapse function.
For a previous study we implemented the algorithm of van Oosterom and presented the possibility for incremental updating (Haunert and Sester, 2005). Though the results were mainly satisfactory we observed certain shortcomings. These can be seen in Figure 2, which shows a result for the example from Figure 1.
The red settlement area in the bottom left corner of Figure 1 is slightly too small for the target scale 1:250.000. A cartographer would try to save this feature by changing small adjacent forest areas (dark green) into settlement. The iterative algorithm however is greedy and not able of taking consequences for future actions into consideration. During the processing, the small green areas are merged with the bigger green areas on the right side. By this, the loss of the settlement area becomes definite. Examples like this motivate to apply global optimization techniques. We explain our approach in Section 3.

2.2 Optimization approaches to aggregation problems in other domains

Though we are not aware of any other global optimization approach to area aggregation in map generalization, there exists a multiplicity of related problems that have been investigated by researchers. Especially, in the field of operations research optimization methods for districting and aggregation problems have been developed. A typical application is the definition of sales districts presented by Zoltners and Sinha (1983). Their solution to find optimal districts is based on mathematical programming. A good theoretical introduction to this method from combinatorial optimization is given by Papadimitriou and Steiglitz (1998). Other researchers have applied meta-heuristics such as simulated annealing (Bergey et al., 2003). In (Haunert, 2007) we compare both approaches applied to area aggregation in map generalization.

3 An approach based on global optimization

In this section we explain our approach to area aggregation by optimization. In Section 3.1 we describe the optimization problem, i.e. the defined constraints and optimization objectives. In section 3.2, we review two methods, which we have developed for its solution: The first approach is based on mixed-integer programming, i.e. it is deterministic and exact (Haunert & Wolff, 2006). The second approach is based on simulated annealing, i.e. a randomized meta-heuristic (Haunert, 2007).

3.1 Aggregation as global optimization problem

In the original presentation of our approach we gave a formal definition based on graph theory (Haunert & Wolff, 2006). Here, only the basic ideas behind this are reviewed. We assume, that input and output map contain the same classes. In the case that generalized classes are defined for the target scale, reclassification and aggregation can simply be done successively.
Specifications of data sets often define size thresholds, which must be satisfied. These thresholds can be different for different classes. In the following, we assume that the same size threshold θ is defined for all land cover classes. This simplifies the notation of our algorithm. However, extending our method to the more general case can easily be done.
The aggregation problem is to group all areas into contiguous regions of size at least θ and to define the class of each region. To avoid that new classes pop up in the generalized map, it is defined that each region must contain at least one area whose class does not change.
Usually, it is not possible to satisfy the defined requirements without changing the classes of some areas. An obvious objective is to minimize the total amount of class changes, since generalization aims to keep the map similar to the input. For this we define a distance between pairs of classes representing costs that are charged for changing an area of unit size from one class into another. This distance models semantic similarities of classes: Changing an area from grassland to farmland is cheap, since both classes represent similar real world phenomena. In contrast, changing an area from grassland to water is relatively expensive. Generally, these distances are not symmetric, since certain classes are considered of higher importance than others. Thus, changing areas of a valuable class (e.g. settlement) into a very frequent class (e.g. grassland) can be defined to be more expensive than vice versa.
Our first experiments showed that it does not suffice to minimize the costs for class changes. The optimal results often contained small isthmuses that were created to connect areas of the same class while expending a minimum cost (Figure 3).
Because of this, we have defined measures of compactness. Both objectives, compactness and similarity of classes, are combined as weighted sum in a cost function, which is minimized. We assume that cartographers, expressing their subjective preferences, can define the parameters of our model. Figure 4 (right) shows the result of a setting we consider suitable.
The described problem was proven to be NP-hard, which means that it is very unlikely to find an efficient algorithm that reaches the optimal solution.

3.2 Review of developed solutions

A common method of combinatorial optimization is mathematical programming. Following this approach, we have defined integer and fractional variables that model the aggregation problem. The optimization objective and the constraints can be expressed by a linear term and linear inequalities, respectively. Such an optimization problem is called a mixed-integer linear program. Often, also the term mixed-integer program (MIP) is used with assumption that all expressions are linear. A MIP can be solved with branch-and-cut techniques, which are implemented in commercial programming libraries. For the solution of the aggregation

problem we used the ILOG™ CPLEX™ Callable Library, version 9.1. Tests of our method showed that optimal solutions can be found only for small instances of the problem. The processing time exponentially explodes: Instances with 30 and 40 areas were solved in 90 seconds and 13 hours, respectively. For an instance with 50 areas a solution was found, but after 20 hours the algorithm did still not prove its optimality. Because of this, specialized heuristics have been introduced resulting in an alternative MIP formulation and the elimination of variables. With this, instances with 400 areas were solved in 17 minutes. For small instances the cost for the obtained solution was about 10% higher than the optimum obtained without heuristics.
The implemented simulated annealing approach results in solutions of slightly less quality (again +10% compared to MIP with heuristics). However, the time needed to process large data sets is still enormous. The data set with 400 areas was processed in 8 minutes. The major disadvantage of this method compared to mixed-integer programming is the requirement for the definition of several tuning parameters, which are not inherent to the aggregation problem. An expert in cartography is probably able to define parameters like distances between land cover classes, but e.g. the definition of an annealing schedule either requires deep insight into the algorithmic theory or extensive experiments. Because of this, the mathematical programming approach should be preferred.
Nevertheless, our new approach does not require a combination with any specific optimization technique. It is simply assumed that instances up to a certain size can be solved near-optimally in reasonable time. The presented methods have been applied only for small problem sizes and it has been doubtful whether the approach is useful for cartographic production. The next section presents our new approach for handling large data sets.

4 A combined approach for efficient area aggregation

In this section we present a new algorithm for processing large data sets. Section 4.1 reviews a heuristic, which has been introduced but not fully exploited in our previous paper. We explain that, with this heuristic, the problem can be decomposed into smaller instances. The algorithm in Section 4.2 uses this fact. Results are presented in Section 4.3.

4.1 Decomposing the aggregation problem based on predefined centres

A heuristic, which was introduced in our previous paper, is to predefine some areas as centres of regions (Haunert & Wolff, 2006). Such centres have different meanings:

- It is not possible to change the class of a centre, i.e. a centre defines the class of the containing region.
- The centroid of such an area defines the geometrical centre of a region. This is used to measure the compactness of the region's shape, i.e., for each area contained in the region, a cost relative to its distance from the geometrical centre is charged.
- It is not possible to merge multiple centres in one region.

Clearly, this heuristic is reasonable, if large, dominant areas are selected as centres. A rather careful approach is to define those areas in the original map as centre, which already satisfy the size threshold θ. The effect of this definition is illustrated in Figure 5, showing the complete data set at scale 1:50.000. Each area being predefined as centre is filled black. It can be observed that such areas cover a rather big part of the data set.
Until now, this heuristic has been used to eliminate variables in our MIP. However, it is also possible to use the same heuristic for decomposing the aggregation problem into several

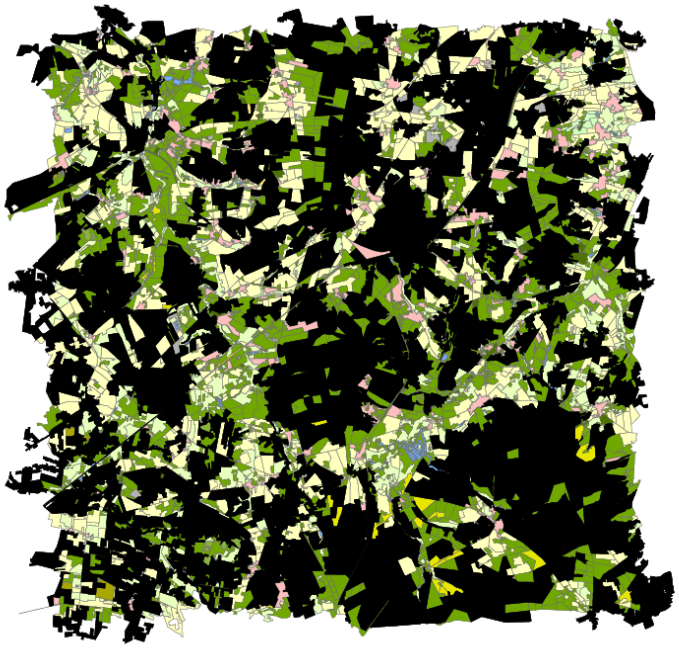

Figure 6: A data set at scale 1:50.000, all areas having sufficient size for the target scale 1:250.000 filled black.

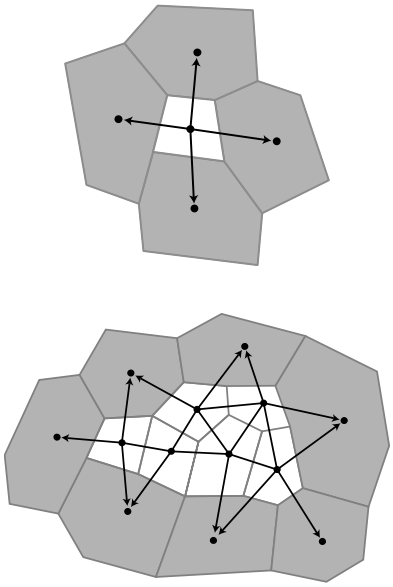

Figure 5: Instances of the aggregation problem appearing during our algorithm. Areas with size $< \theta$ white; areas satisfying threshold grey.

smaller problems. In fact, the aggregation problem can be solved independently for each set of areas that is completely surrounded by centres. This is simply because those features of a centre that potentially influence the result of the aggregation, i.e. its class and geometrical centre, are fixed. Unfortunately, this fact does not allow partitioning the data set into manageable problem instances. There is no guarantee that there is more than one connected component of areas not being predefined as centres and, in our example, the resulting 'islands' still contain too many areas. However, when defining an intermediate scale, the required size of regions decreases and, consequently, the number of centres increases. In the next section we discuss how to locally define intermediate scales and corresponding thresholds, such that the resulting problem instances have a limited size.

4.2 A generalization of the iterative algorithm

In each iteration of the algorithm from Section 2.1, only results of single, potential merge actions are analyzed. Such an iteration can be described as the solution of a very simple instance of the optimization problem, which is displayed in Figure 5 (top). The area, which was selected due to its minimal size, is displayed in white. All neighbours that are potential candidates for a merge are displayed in grey. Among these, the smallest area defines the threshold for the aggregation problem. Arrows display the direction of potential merges. By solving this instance optimally, the best neighbour is chosen and the merge is performed. The original iterative algorithm would have done the same.

A step like this can be regarded as a small decrease of scale, having the consequence that only the smallest area in the data set becomes too small for representation. With the heuristic from Section 4.1, this is the only area not being predefined as centre. The basic idea behind our combined approach is to move in bigger steps to the target scale, i.e., to iteratively solve instances similar to Figure 5 (top), but larger. By this, more combinations can be analyzed in one step, which probably results in solutions of higher quality.

Such an instance of the optimization problem is illustrated in Figure 5 (bottom). It includes multiple areas that are smaller than the area threshold (white), but again these areas are surrounded by predefined centres of sufficient size (grey). We refer to the set of white areas as being contained in the problem instance. Also here it is allowed to merge white areas with grey areas. Additionally, it is possible to define a new region by a set of white areas. Black lines without arrows are drawn for these potential merges that are not restricted by a direction. Instances like this can be solved with the developed optimization method, including several hundreds of areas. To define instances like this, we also iteratively select the smallest area, but instead of merging it with a neighbour it is assigned to an open problem instance. Throughout the algorithm these instances grow until a critical size is reached. In such a case, the problem is solved.

It remains to define which problem instances are too large to be handled and which instances can be solved with the developed optimization technique. It is clear that this decision depends on different criteria. By use of a multiprocessor computer and a state-of-the-art software for the solution of mixed-integer programs it is certainly possible to process larger instances than by application of a standard PC and a license-free optimizer. Also, different cartographers might be willing to spend different amounts of time for data processing. Because of this, it is reasonable to simply define an integer k. Instances with more areas smaller than θ (white areas in Figure 5) are not solved. Before defining k one should test whether the applied combination of hardware and software can cope with instances of this size.

Algorithm 1 specifies our approach:

```
P = a set of open problem instances, initially empty
a = smallest area in data set
while a is smaller than required for the target scale do
        P' = the set of problem instances in P containing a neighbor of a
        if total number of areas contained in instances P' < k then
                Remove all instances in P' from P.
                Create a new instance p comprising all areas in instances P' plus a.
                if p contains k areas then
                        //The new instance has critical size. Introduce intermediate scale:
                        Solve p (threshold = size of smallest centre in neighborhood).
                else
                        Insert p to P.
                end if
        else
                //The instance will exceed the critical size. Introduce intermediate scale:
                Solve the instance in P' containing most areas (threshold = size of a).
                Remove this instance from P.
        end if
        a = smallest area in data set, not contained in instances P
end while
Solve all remaining problem instances in P, applying the threshold for the target scale.
```

Algorithm 1: Defining and solving problem instances of critical size.

Let us consider the case $k = 1$. In this case, the set P is empty throughout the processing, as problem instances defined by the smallest area are immediately solved. Therefore, the algorithm generalizes the existing iterative algorithm.

In Section 3.1 we mentioned the disadvantage of tuning parameters, which are not inherent to the problem. The parameter k has such disliked characteristic. However, it can be argued that setting $k = 1$ is not wrong; it is simply more likely that solutions of higher quality can be obtained when foreseeing more merges at once. We can consider a chess player as allegory: It

is not possible to foresee a complete match. Because of this one will try to think as many moves ahead as possible.

4.3 Results

The proposed method has been tested for a complete data set of the ATKIS DLM 50, corresponding to a topographic map at scale 1:50.000. Before applying the aggregation, a pre-processing based on Straight Skeletons was applied, comprising area collapse, partial area collapse, and polygon decomposition (Haunert & Sester, 2007). The resulting data set includes 5537 areas. This is referred to as input (Figure 7). We tested our presented method for this, using different values for *k* and two different settings for the cost function, i.e. we tested our method for minimizing class change and for a combined cost for class change and non-compactness. We applied the thresholds defined in the ATKIS specifications for the scale 1.250.000.
When minimizing class change, setting $k = 200$ resulted in 27,4% less costs compared to the purely iterative method ($k = 1$). When minimizing a combined cost for class change and non-compactness, which we consider more adequate, the total improvement was only 7,8%. Table 1 summarizes our results for this bi-criteria objective. It can be seen that the costs for non-compactness are very similar for different values of *k*. It seems that, concerning this objective, the iterative algorithm does quite well by greedily choosing a neighbour. However, the decrease of costs for class change is still considerable (19,8%). Note that, for the special case $k = 1$, our implementation is certainly not optimal in matters of efficiency: In each iteration a MIP is defined and solved. Of course, explicitly testing all neighbours could be done faster.

k = maximal number of areas per iteration	1	50	100	150	200
processing time (minutes)	3,86	2,05	8,66	26,06	82,27
cost for class change (normalized)	100	90,0	86,8	85,8	80,2
cost for non-compactness (normalized)	100	99,4	98,0	98,8	98,2
total cost (normalized)	100	96,2	94,3	94,5	92,2

Table 1: Experimental results for a combined cost for class change and non-compactness

The resulting map with the combined objective and $k = 200$ is displayed in Figure 8. Figure 9 shows an example from this data set. The settlement (red) in the input data set (Figure 9, left) is too small for the target scale. Therefore, it is merged with a neighbour by the iterative algorithm (centre). When taking more combinations into account, i.e., for $k = 200$, the settlement can be saved by changing two small areas from forest to settlement (right). By this, the settlement becomes sufficiently large. This is preferred due to two reasons: Firstly, the change of the relatively large settlement would be more expensive. Secondly, the resulting shape is relatively compact. Examples like this motivated the development of the proposed method. The result confirms, that high quality solutions can be obtained.
In addition to this empirical analysis of the results, the next section gives a more abstract explanation about possible benefits of the algorithm.

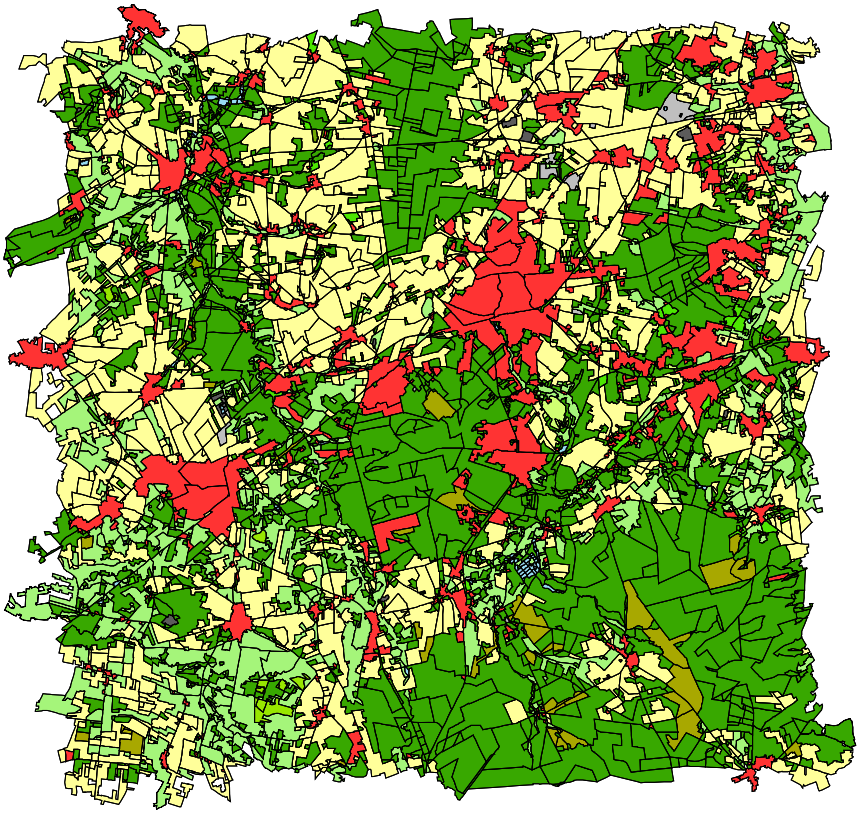

Figure 7: Input data set ATKIS DLM 50 (1:50.000) after pre-processing.

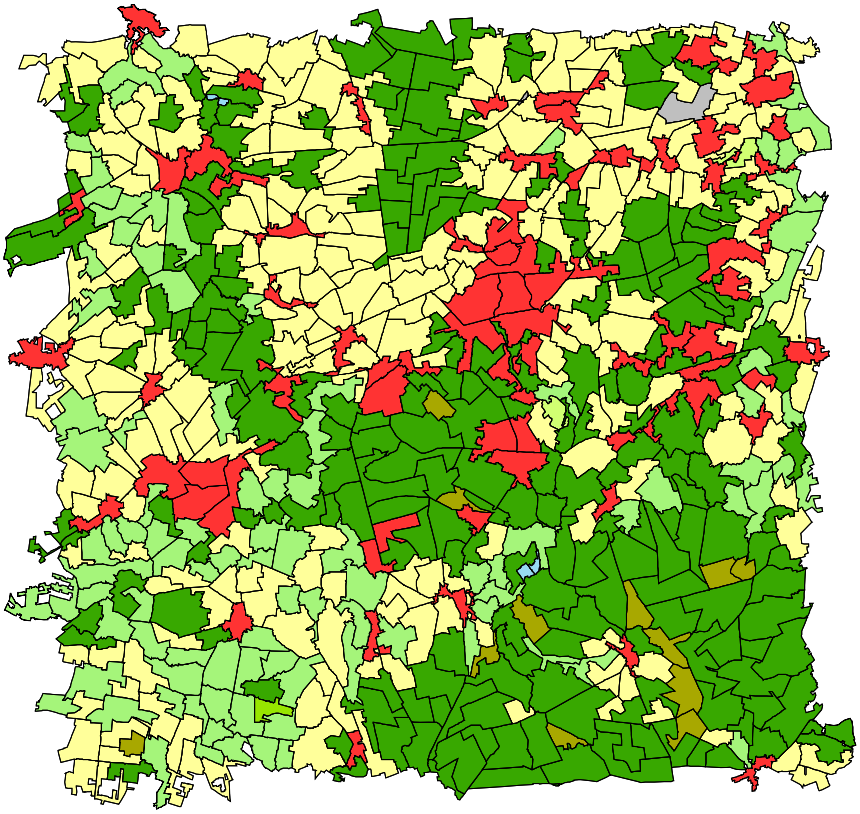

Figure 8: Result obtained with proposed method at scale 1:250.000. Problem instances with 200 areas or less solved by application of optimization method.

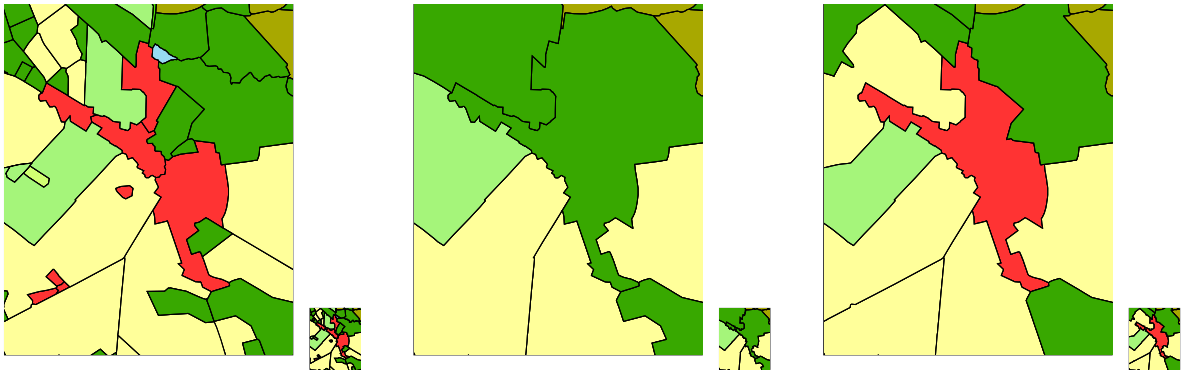

Figure 9: Input (left), result of purely iterative algorithm (centre), and result of combined approach (right). The latter can save the red settlement by 'stealing' two small green areas. The small boxes display the same samples at scale 1:250.000. Note that lines need to be simplified and colours were chosen to emphasize differences of classes.

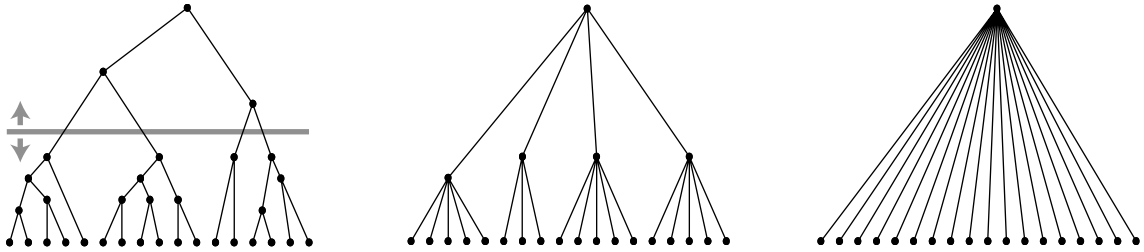

Figure 10: Tree structure resulting from original algorithm (left), combined method (centre) and global approach (right) for one region in the output data set. The grey line in the left subfigure depicts the possibility to define intermediate scale levels.

4.4 Future work

The proposed algorithm does not offer a theoretical guarantee of performance. However, in addition to the observed improvement of quality compared to the original algorithm, there are good reasons to apply the proposed procedure.

Figure 10 (left) shows a possible hierarchy of areas resulting from the purely iterative algorithm from Section 2.2. The displayed nodes represent areas in the original map (bottom), the resulting area in the output (top) and aggregates that were produced by merging pairs of areas. Assuming that the displayed distance of a node from the bottom line corresponds to the area's size, a map with no area smaller than a variable threshold can be defined by moving the grey line up or down. The map comprises the areas that correspond to the leaf nodes of the tree after cutting it off below the line. This hierarchical characteristic has been shown to be useful for adaptive zooming (van Oosterom, 1995) and incremental updating (Haunert and Sester, 2005).

Figure 10 (right) shows that the global approach does not have such characteristic. It offers a globally optimal solution for a target scale, which is obtained in a single step. The combined approach (centre) tries to reach the target scale via big steps. If needed to overcome the intractability of the problem, intermediate levels of detail are introduced.

Presumably, such stepping-stones can be useful to display maps at an intermediate scale, or to restrict the area of an update's influence, which is needed for incremental generalization. Recall that, applying the centre heuristic from Section 4.1, the problem can be decomposed into smaller instances. This is particularly interesting for incremental updating, as a local change in the original map does not require a complete new processing of the generalized

map. However, it is supposable that the resulting area of influence of an update becomes too large. Additional heuristics might be useful to restrict this problem.

5 Conclusion

A new method for the automated aggregation of areas in a planar subdivision has been developed, which has been shown to be applicable for large problem instances. The method combines an existing iterative procedure with techniques from combinatorial optimization.
The assumption that the method results in maps of higher quality than the purely iterative algorithm was confirmed by discussion of examples and a comparison of results being obtained with different settings. We have shown that our algorithm generalizes an existing iterative method and produces results with considerable less change of land cover classes compared to the purely iterative algorithm.
When minimizing class change, our method resulted in 27,4% less costs. However, our method only marginally improves the compactness of shapes. When considering both criteria, our method resulted in 19,8% less class change, 1,8% less costs for non-compact shapes and, as a result of a weighted sum, 7,8% less total costs.
We conclude that, in contrast to the purely iterative method, our method is able to preserve important features by sacrificing less important ones.
Future work will concern with possibilities for adaptive zooming and updating.

Acknowledgement

This work shows results from the project entitled Updating of Geographic Data in a Multiple Representation Database. The project is funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG). It is part of the bundle-project entitled Abstraction of Geographic Information within the Multi-Scale Acquisition, Administration, Analysis and Visualisation.

References

de Berg, M., van Kreveld, M., & Schirra S., 1995: A new approach to subdivision simplification. In Twelfth International Symposium on Computer-Assisted Cartography, volume 4, pages 79–88, Charlotte, North Carolina, USA.

Bergey, P., Ragsdale, C. & Hoskote, M., 2003: A simulated annealing genetic algorithm for the electrical power districting problem. Annals of Operations Research, 121:33–55.

Bobzien M., 2001: Flächenzusammenfassungen in der Modellgeneralisierung. In: Mitteilungen des Bundesamtes für Kartographie und Geodäsie, Band 20: Arbeitsgruppe Automation in der Kartographie - Tagung 2000, pp. 19–29. Verlag des Bundesamtes für Kartographie und Geodäsie, Frankfurt am Main, Germany.

Galanda, M., 2003: Automated Polygon Generalization in a Multi Agent System. PhD thesis, Department of Geography, University of Zurich, Switzerland.

Haunert, J.-H., 2007: Optimization methods for area aggregation in land cover maps. To appear in: Proc. of XXIII International Cartographic Conference, 4–10 July 2007, Moscow, Russia.

Haunert, J.-H. & Sester, M., 2005: Propagating updates between linked data sets of different scale. In Proc. of XXII International Cartographic Conference, 11–16 July 2005, A Coruna, Spain.

Haunert, J.-H. & Sester, M., 2007: Area Collapse and Road Centerlines based on Straight Skeletons. Accepted for publication in Geoinformatica.

Haunert, J.-H. & Wolff, A., 2006: Generalization of land cover maps by mixed integer programming. In GIS '06: Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems, pages 75–82, Arlington, Virginia, USA.

Jaakkola, O., 1997: Quality and Automatic Generalization of Land Cover Data. PhD thesis, Department of Geography, University of Helsinki.

Michalewicz, Z. & Fogel, D. B., 2004: How to Solve It, Modern Heuristics. Springer-Verlag GmbH, Berlin, Germany.

van Oosterom, P., 1995: The GAP-tree, an approach to 'on the fly' map generalization of an area partitioning. In: Muller, J. C., Lagrange, J. P. & Weibel, R. (Editors) GIS and Generalization: Methodology and Practise. Taylor & Francis, London, pp. 120–132.

Papadimitriou, C. & Steiglitz, K., 1998: Combinatorial Optimization. Dover Publications, Inc., Mineola, NY, USA.

Podrenek, M., 2002: Aufbau des DLM50 aus dem Basis-DLM und Ableitung der DTK50 – Lösungsansatz in Niedersachsen. In: KS, Band 6, Kartographie als Baustein moderner Kommunikation, pp.126–130.

van Smaalen, J, 2003: Automated Aggregation of Geographic Objects. PhD thesis, Wageningen University, The Netherlands.

Zoltners, A. & Sinha, P., 1983: Sales territory alignment: A review and model. Management Science, 29(11):1237–1256.