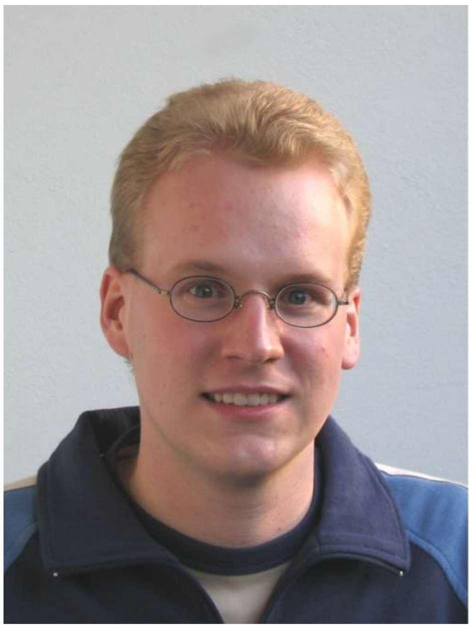

Jan-Henrik Haunert was born in 1978 and studied Surveying at the Leibniz Universität Hannover. He graduated in 2003 and obtained the Master's degree (Dipl.-Ing.). Since January 2004 he is a scientific assistant of the Institute of Cartography and Geoinformatics at the Leibniz Universität Hannover. The focus of his research lies on automated map generalization. In a project funded by the German Research Foundation he develops methods for the automatic updating process of a Multiple Representation Database.

OPTIMIZATION METHODS FOR AREA AGGREGATION IN LAND COVER MAPS

Jan-Henrik Haunert

Institute of Cartography and Geoinformatics, Leibniz Universität Hannover

Appelstraße 9a, 30167 Hannover, Germany

jan.haunert@ikg.uni-hannover.de

Abstract

The aggregation of areas is an important subproblem of the map generalization task. Especially, it is relevant for the generalization of topographic maps which contain areas of different land cover, such as settlement, water, or different kinds of vegetation. An existing approach is to apply algorithms that iteratively merge adjacent areas, taking only local measures into consideration. In contrast, global optimization methods are proposed in this paper to derive maps of higher quality. Given a planar subdivision in which each area is assigned to a land cover class, we consider the problem of aggregating areas such that defined thresholds are satisfied. The aggregation is directed at two objectives: Classes of areas shall change as little as possible and compact shapes are preferred. In this paper, the problem is formalized and two different approaches are compared, namely mixed-integer programming and simulated annealing.

1 Introduction

Generally, the aim of map generalization is to create a map that satisfies requirements of a reduced target scale, while preserving characteristic features of an original map. While formalized requirements like minimal dimensions are often defined in the specifications of data sets, the formal description of the statement's second part is a rather difficult task. However, if the changes applied to the source data set can be expressed by quantitative measures, the generalization task can be formalized as a constrained optimization problem.

In a previous paper we presented a method based on this approach for the aggregation of areas in a planar subdivision (Haunert & Wolff, 2006). In topographic data bases such a representation is commonly used for areas of different land cover classes. The aggregation problem is due to area thresholds that are defined differently for the source and the target data set. Simply omitting features from the source data set that are too small for the target scale would violate the prohibition of gaps in a planar subdivision. Therefore, features need to be merged with neighbors, which results in changes of their classes. In our earlier paper, we proved that solving the aggregation problem with minimum change of land cover classes is NP-hard, meaning that it is unlikely to find an efficient algorithm. We therefore introduced mixed-integer programs for the problem and applied heuristics to eliminate variables. In this paper we compare this method with another heuristic approach, namely simulated annealing. After discussing related work (Section 1.1), we explain both methods in general (Section 2), define the area aggregation problem (Section 3), and present our solutions for the problem (Sections 4 and 5). We present and compare the obtained results in Section 6 and conclude the paper (Section 7).

1.1 Related work

Different researchers have proposed iterative methods for the area aggregation problem. The following algorithm is described by van Oosterom (1995):

In each iteration the feature with lowest importance is selected. The selected feature is merged with a neighbor, which is chosen according to a collapse function, and the next iteration is processed. The iteration can be terminated, if all areas satisfy the minimal dimension that is required for the target scale.

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. Generally, semantic similarity of classes, boundary lengths, and area sizes are considered as criteria that need to be incorporated into the collapse function.

The main problem with these iterative approaches is that consequences for future actions are not taken into account, when greedily selecting a neighbor. Therefore, a global approach will be presented in this paper.

Though there has not been any 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 Hess & Samuels (1971). Their solution to find optimal districts is based on mathematical programming. Other researchers have applied meta-heuristics such as simulated annealing (Bergey et al., 2003). We briefly explain the general principles of these two optimization techniques in the next section.

2 Applied techniques of combinatorial optimization

In this section, we briefly explain mathematical programming (Section 3.1) and simulated annealing (Section 3.2). For a detailed introduction and further references we refer to Papadimitriou & Steiglitz (1998) and Reeves (1993).

2.1 Mathematical programming

Let us first define a *linear program* (LP): Given an $m \times n$ Matrix A, an m-vector b, and an n-vector c, minimize $c^T \cdot x$ subject to $A \cdot x \geq b$, $x \geq 0$, with $x \in R^n$. Generally, an LP can be solved in polynomial time. Most commonly the *simplex algorithm* is applied. Although this theoretically may require exponential time, it solves LPs with hundreds of thousands of variables in practice.

By replacing the continuous variables $x \in R^n$ in this definition by integer variables $x \in Z^n$, we define an *integer linear program* (ILP) or simply *integer program* (IP). Many combinatorial optimization problems can be formulated as IP. Though the definitions of IP and LP are very similar, the computational complexity of solving an IP is much higher. In fact the problem is NP-hard. However, several algorithms have been developed for the solution of IPs, which have been found out to be useful for applications. A *mixed-integer* *program* (MIP) is a combination of an LP and an IP, i.e., it may contain continuous as well as integer variables. Basically, a MIP can be solved with the same techniques as an IP.

A method that is implemented in several commercial software packages is called *branch-and-cut.* The software we used for our experiments is the ILOG CPLEX Callable Library 9.100. This allows to integrate branch-and-cut techniques with Java applications. Our tests were performed on a Linux server with 4 GB RAM and a 2.2 GHz AMD-CPU.

2.2 Simulated annealing

The techniques described in Section 2.1 restrict to objectives and constraints that can be expressed by linear combinations of variables. Even in case that such a formulation of a problem is found, the branch-and-cut technique can turn out to be inefficient and therefore inappropriate for application. However, it is often not necessary to insist on finding the globally optimal result. Therefore, *heuristic* techniques have been developed. Generally, these attempt to find relatively good solutions in reasonable time. Two different types of heuristics need to be distinguished: Heuristics that are designed for a specific problem and those that offer solutions for a very general class of problems (*meta-heuristics*). We will introduce heuristics of the first type in Section 4 to eliminate some variables in our mixed-integer programs. A prominent meta-heuristic is *simulated annealing*, which goes back to Kirkpatrick (1983) and has been applied to map generalization by Ware et al. (2003). We explain its basic principles in this section and present the application to the area aggregation problem in Section 5.

To explain simulated annealing, let us first consider a *hill-climbing* method: Starting from a feasible solution, hill climbing iteratively moves to a solution which is cheaper according to a cost function *c*, e.g. it selects the best solution in a defined neighborhood of the current solution. The problem with the hill-climbing approach is that it usually gets stuck in local optima. The simulated annealing approach is to occasionally accept moves to worse solutions, in order to escape these local optima. For this, a *temperature* T is introduced, which controls the probability of accepting worse solutions. Initially, T is high, meaning that it is likely that worse solutions are accepted. During the simulation T is decreased according to a defined *annealing schedule*. Commonly, a multiplier $\alpha \in [0,1]$ is introduced for this. The following algorithm defines the common simulated annealing approach:

1. Find an initial feasible solution s and define the temperature by $T \leftarrow T_0$.

2. Randomly select a solution s' in the neighborhood of s.
3. If $c(s') \leq c(s)$, set $s' \leftarrow s$, else, set $s' \leftarrow s$ with probability $\exp\left(-\frac{c(s')-c(s)}{T}\right)$.
4. Reduce the temperature, i.e., set $T \leftarrow \alpha \cdot T$.
5. Proceed with 2 until the temperature falls below a threshold T_E.

We specify this approach for the area aggregation problem in Section 5.

3 Problem definition

The definition of the aggregation problem is based on the adjacency graph $G(V,E)$ of the planar subdivision. This contains a node $v \in V$ for each area and an edge $\{u,v\} \in E$, if the areas corresponding to u and v share a common boundary. Each node v has an initial color $\gamma(v)$ and a weight $w(v)$, corresponding to the area's class and size, respectively. A feasible solution of the area aggregation problem is defined by a partition $P = \{V_1, V_2, \ldots, V_n\}$ of V and a new color $\gamma'(v)$ for each node v, such that each region $V' \in P$ is contiguous, contains only nodes of the same new color (referred to as γ'), contains at least one node with unchanged color, and satisfies a color dependent weight threshold $\theta(\gamma')$. Subject to these constraints, the problem is to find the solution which minimizes a sum of costs c_1 for color change and costs c_2, which are charged to penalize non-compact shapes. The total costs for color change are defined by

$$c_1 = s_1 \cdot \sum_{v \in V} w(v) \cdot d(\gamma(v), \gamma'(v)),$$

with $d(\gamma(v), \gamma'(v))$ expressing the costs that are charged to change an area of unit size from the old color into the new one. The idea behind this approach is to charge relatively low costs for semantically similar classes, such as deciduous forest and coniferous forest, and high costs for dissimilar classes, such as water and settlement. We also allow for asymmetric distances, as it might be favored to keep important classes unchanged. In this case one would define a relatively high distance from this class to others. The parameter $s_1 \in [0,1]$ needs to be set to define the weight of this objective.

The cost c_2 for non-compactness is a combination of two measures, which are based on the perimeters of regions and the average distance to a center of the region. The latter measure has been applied by Zoltners & Sinha (1983). For this we define $\delta(v,u)$ to be the Euclidean distance between the centroids of two areas. Similar to s_1, $s_2 \in [0,1]$ defines the weight of this objective.

4 Area aggregation by mathematical programming

In our earlier paper we tested different MIP formulations for solving the area aggregation problem (Haunert & Wolff, 2006). The processing time turned out to be very high. Even with the best performing MIP, at most 40 nodes were processed with proof of optimality, i.e., with lower bound equal to the objective value of the integer solution. An instance of this size was solved in 12.7 hours.

Due to this performance, we applied a heuristic resulting in a MIP formulation similar to the one of Zoltners & Sinha (1983). The approach of this is to define a strong contiguity requirement based on a precedence relationship. According to this, each region contains a node called centre, and, for each other node in the region, there must exist a neighbor in the same region which is closer to the centre. Note that each node is a potential centre, i.e., it is generally not required to predefine the set of centers. Obviously, certain regions become infeasible with this stricter requirement for contiguity. However, it is likely that only non-compact regions are excluded, which anyway are not optimal. With this approach, the same instance with 40 nodes was solved in 62 seconds. For the processed instances, the cost of the solution increased maximally by 5%. We present a version of our MIP based on precedence relationship, which neglects the objective for small perimeters. In fact, with this simplification, the MIP becomes a *binary program* containing only binary variables $x_{uv} \in \{0,1\}$. Setting $x_{uv} = 1$ means to assign node $v \in V$ to the region with centre $u \in V$.

$$\text{Minimize} \quad \sum_{u\in V}\sum_{v\in V} w(v)\cdot\left[s_1 \cdot d(\gamma(v),\gamma(u)) + s_2 \cdot \delta(v,u)\right]\cdot x_{uv} \quad \text{subject to}$$

$$\sum_{u\in V} x_{uv} = 1, \quad \forall v \in V, \qquad \text{(each node is assigned to one centre)}$$

$$\sum_{v \in V} w(v) \cdot x_{uv} \geq \theta(\gamma(u)) \cdot \mathrm{x}_{\mathrm{uu}}\ , \quad \forall u \in V, \qquad \text{(for each region the area threshold is satisfied)}$$

$$\sum_{w \in \mathrm{Pred}_u(v)} x_{uw} \geq x_{uv}\ , \quad \forall u, v \in V : u \neq v. \qquad \text{(each region is contiguous in the defined sense)}$$

The last constraint means that for each node *v* being assigned to centre *u*, there exists at least one node in the set of predecessors $\mathrm{Pred}_u(v)$, which is also assigned to *u*. This ensures the strong contiguity requirement.

Two additional heuristics were discussed and tested in our earlier paper (Haunert & Wolff, 2006). The first is to predefine large, dominant areas as centers and to exclude small areas from the set of predefined centers. The second is to assume that two nodes with a large distance in between do not become merged in the same region. Both heuristics allow to eliminate variables, which speeds up the processing. This allowed to process instances with 400 nodes in 17 minutes, leading to solutions of approximately 10% more costs compared to the MIP without heuristics. We will present results of this setting in Section 6.

5 Area aggregation by simulated annealing

Our method by simulated annealing is based on the algorithm in Section 2.2. The initial feasible solution can be found with the iterative algorithm of van Oosterom (1995) from Section 1.1. The most important remaining design issue is to define the neighborhood of a feasible solution. Given a feasible solution, we define its neighborhood as the set of solutions that can be obtained by application of a single node interchange operation, i.e., one node is removed from a region and assigned to another adjacent region as being shown in Figure 1. The normal case is shown in Figure 1a. Figure 1b shows a special case in which a region is separated into two contiguous regions when removing a contained node.

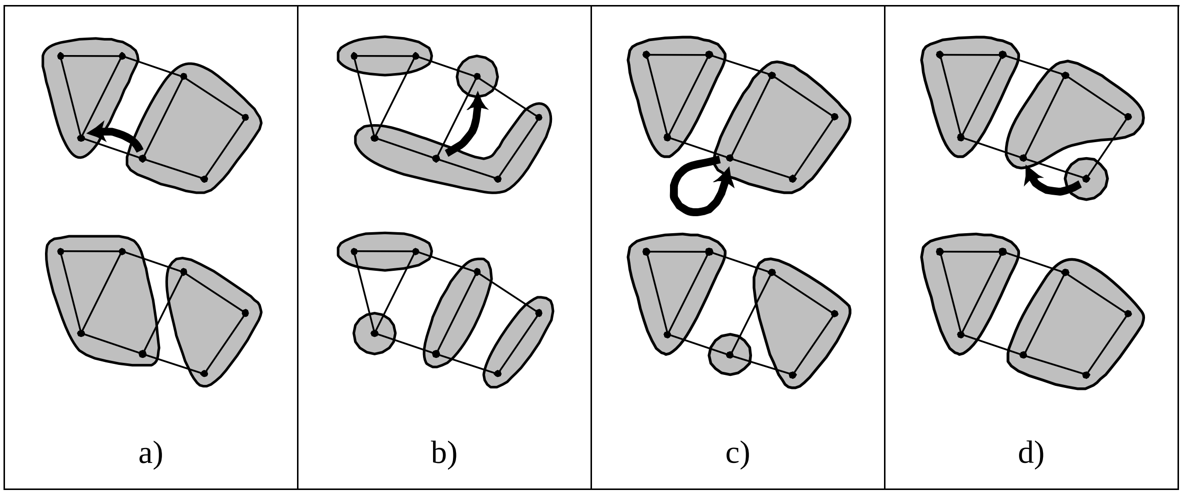

Figure 1: The node interchange operation before (top) and after application (bottom)

To allow for more variation, we define that a single node can also form a new region after being removed from an aggregate (Figure 1c). Contrarily, such a region with only one node can disappear, if the node is assigned to another region (Figure 1d).

Obviously, by application of a node interchange operation, a solution might become infeasible. For example, by removing a node from a region, the threshold of the region might be violated. However, it is critical to restrict the set of allowed node interchange operations to those that produce feasible solutions. Consider an initial solution containing only regions that exactly satisfy their weight thresholds: Removing any node from its region will create an infeasible solution. Thus, the initial solution represents an isolated point in the solution space. It is clear that under such conditions it is not possible to reach the global optimum. In order to ensure the connectivity of all solutions via the defined neighbor relationship, we relax the constraint for weight feasibility and charge for each region $V' \in P$ that is smaller than its threshold an additional cost equal to

$$s_3 \cdot \left(\theta(\gamma') - \sum_{v \in V'} w(v) \right).$$

Again, the parameter $s_3 \in [0,1]$ needs to be set to define the weight of this objective.

A problem of this approach is that the algorithm might terminate with regions that do not satisfy the threshold constraint. Therefore, we need to define an algorithm that repairs these infeasible solutions. Again we apply the iterative algorithm for this, i.e., we select the smallest infeasible region and merge it with the best neighbor until the result is feasible.

Probably, the most difficult problem that appears when applying simulated annealing is the definition of several tuning parameters that are not inherent to the problem itself. For our application this concerns the initial temperature T_0, the final temperature T_E, the weight of the penalty for too small regions s_3, and the number of iterations. With these parameters, the annealing parameter α is defined. Normally, the only way to find the best parameters for a certain problem is to perform experiments. The results discussed in the next section were found with this approach.

6 Results

We first show a result that was obtained with our method based on mixed-integer programming and then discuss the effects of the applied heuristics. Figure 2(left) shows an example from a topographic data base at scale 1:50.000. The aim is to aggregate the areas such that the specifications of the target scale 1:250.000 are satisfied. The red settlement area in the centre of the clipping is too small according to the defined threshold. By application of the existing iterative algorithms such features will usually be merged with a neighbor. Thus, the settlement, which is often considered to be an important map feature, will be lost. However, by application of global optimization techniques, it is possible to sacrifice smaller areas, in order to safe the valuable feature. Two results are shown, which were obtained by application of different objectives. Figure 2(centre) shows the result when minimizing color change. In this case, small forest areas are changed into settlement, producing a long bridge to a small settlement area. By this, the settlement becomes sufficiently large. However, the complex shapes might be disfavored by cartographers. Therefore, we propose to apply a combination of costs for color change and non-compact regions. With this setting we obtained the result in Figure 2(right). Again, the settlement was saved by sacrificing small forest areas. This time, however, a more compact shape was produced. We assume that the formalized objective function sufficiently models the aims of area aggregation, but other operations such as line simplification need to be applied to obtain a generalized map.

Table 1 summarizes the results that were obtained using different techniques. We increased the number of iterations for simulated annealing linearly in the number of nodes. The parameters T_0, T_E, and s_3 were not modified. The theoretically possible case of violated area thresholds in the result, which was discussed in Section 5, did never happen.

Simulated annealing produces solutions of less quality than our MIP with problem-specific heuristics. The cost for the solutions increased maximally by 26%. Still, the costs are much lower than with the iterative algorithm. For small instances it can be observed that the MIP outperforms simulated annealing in matters of time and quality. For large instances, simulated annealing is faster and the costs become more similar.

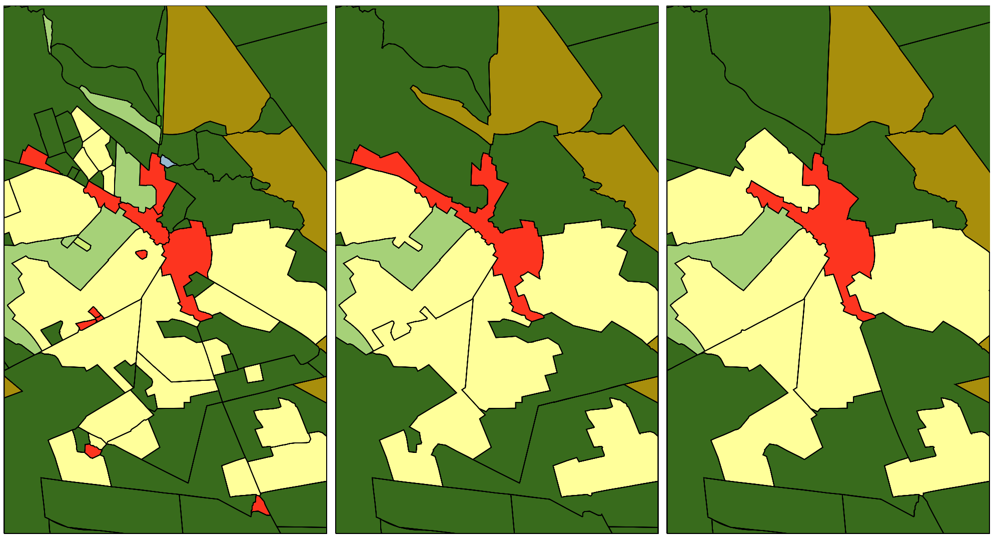
Figure 2: An example from the German data set ATKIS DLM 50 at scale 1:50.000 (left) and two solutions which are feasible according to the specifications of the ATKIS DLM 250 (scale 1:250.000). The solution in the centre minimizes color change; the right solution minimizes a combined cost for color change and non-compactness.

nodes	Iterative Algorithm	MIP without heuristics		MIP with heuristics		Simulated annealing		
	cost	time	cost	time	cost	iterations	time	cost
50	6,35	20h	2,15*	0,45s	2,34	25000	11,6s	2,75
200	13,37			100,4s	6,35	100000	117,5s	7,99
300	22,56			714,7s	14,68	150000	276,0s	16,76
400	29,04			1366,9s	19,15	200000	483,1s	20,72

Table 1: Experimental results. All MIPs were solved to optimality except *.

7 Conclusion

We have presented two very different approaches to the same area aggregation problem. The first method is deterministic and based on mixed-integer programs being solved with branch-and-cut methods. Problem-specific heuristics were applied to obtain an appropriate performance. The second method is simulated annealing, i.e., a randomized meta-heuristic. We conclude that, for small instances, our mixed-integer programs with heuristics outperform simulated annealing in matters of time and quality of the result. For large instances, however, simulated annealing is a competitive alternative.

Two disadvantages of simulated annealing need to be emphasized. Firstly, the results are not reproducible. Secondly, the tuning of parameters that are not inherent to the generalization problem is difficult, i.e., not possible without insight into the algorithmic theory. In contrast, the problem-specific heuristics for the MIPs can easily be explained to a cartographer who can argue whether he agrees on these restrictions or not. Due to these reasons we concentrate future research on the approach by mixed-integer programming.

8 References

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.

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.

Hess, W. & Samuels, S., 1971: Experiences with a sales districting model: Criteria and implementation. Management Science, 18(4, Part II):41–54.

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

Kirkpatrick, S., Gelatt Jr, C. & Vecchi, M., 1983: Optimization by simulated annealing. Science, 220(4598):671–680.

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, UK, 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, S.126-130, Bonn, Germany.

Reeves, C. R., 1993: Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publications, Oxford, UK.

Ware, J.M., Jones, C.B. & Thomas, N., 2003, Automated cartographic map generalisation with multiple operators: a simulated annealing approach. The International Journal of Geographical Information Science, 17(8):743–769.

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