Map Simplification with Topology Constraints: Exactly and in Practice

Stefan Funke Thomas Mendel Alexander Miller Sabine Storandt Maria Wiebe

Abstract

We consider the classical line simplification problem subject to a given error bound ϵ but with additional topology constraints as they arise for example in the map rendering domain. While theoretically inapproximability has been proven for these problem variants, we show that in practice one can solve medium sized instances optimally using an integer linear programming approach and larger instances using an heuristic approach which for medium-sized real-world instances yields close-to-optimal results. Our approaches are evaluated on data sets which are synthetically generated, stem from the OpenStreetMap project, and the 2014 GIS Cup competition.

1 Introduction

In the classical line simplification problem (**CLSP**) we are given a polygonal chain $C = p_0p_1p_2 \dots p_n$ with $p_i \in \mathbb{R}^2$, an error parameter $\epsilon > 0$ and ask for a simplification of C, that is, indices $i_1 < i_2 < \dots < i_k$ with $0 < i_j < n$ such that the polygonal chain $\widetilde{C} = p_0p_{i_1}p_{i_2} \dots p_{i_k}p_n$ is a faithful approximation of C. Here 'faithful' means that for every 'shortcut' segment $s_j = p_{i_j}p_{i_{j+1}}$ of the simplification the furthest distance of a point in $\{p_{i_j+1}, \dots p_{i_{j+1}-1}\}$ to the shortcut segment s_j is at most ϵ. A natural optimization goal is to compute a faithful approximation with as few vertices as possible, that is, minimizing k.

Solving CLSP is of great interest in particular in the map rendering context. One of the main challenges for rendering map data on the screen arises from the abundance of data. Assume we want to render the road network of Germany on a mobile device like a tablet. A cross-country Autobahn like the A7 consists of several thousands of individual road segments. Rendering all of them is certainly a waste of time when dealing with the screen of a mobile device. So typically one would *simplify* the chain of segments by replacing subsequences of degree-2 nodes along the A7 by single road segments. Depending on the screen size and resolution, this can be done without really affecting the visual quality of the result. Naturally, this simplification should not introduce self-intersections, so a sensible generalization of CLSP to the map rendering context (originally when simplifying country boundaries) is the map simplification problem (**MSP**), where we are given a planar subdivision in form of a planar embedding of a straight-line graph $G(V, E)$, a parameter $\epsilon > 0$ and the goal is to solve CLSP for each degree-2 chain of the graph such that the total number of surviving vertices is minimized without introducing intersections (within a single degree-2 chain as well as between different degree-2 chains).

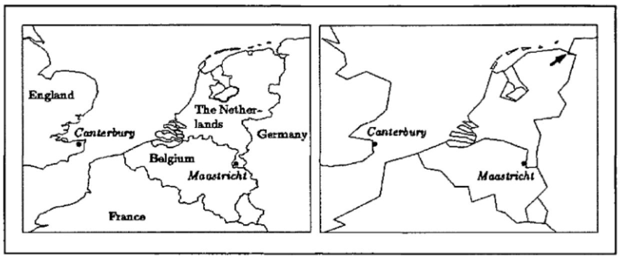


Figure 1: A map of Western Europe with inconsistencies after line simplification of country boundaries (from [1]), courtesy of de Berg et al.

Unfortunately, just solving MSP without additional care might lead to undesired effects, see Figure 1. In this simplification (right) of a map excerpt of Europe (left), some cities switched countries or ended up in the sea. This gives rise to a more general map simplification problem with topology constraints (**MSTOPOP**): Given a planar subdivision as a planar embedding of a straight-line graph $G(V, E)$, a parameter $\epsilon > 0$ and a set of points $P \subset \mathbb{R}^2$, the goal is to solve CLSP for each maximal degree-2 chain of the graph such that the total number of surviving vertices is minimized, no intersections are introduced, and every point $p \in P$ remains in the same face as before.

For our map rendering application, MSTOPOP is the most natural formulation of the respective optimization problem. Nevertheless, to allow for simpler solution strategies and efficient solution we will define a more local variant of this problem called **MSLOCTOPOP** in Section 2 (which still turns out to be theoretically hard to solve and even approximate).

1.1 Related Work

For CLSP there are several known algorithms, the most popular being the algorithm by Douglas/Peucker [2], which unfortunately does not guarantee absence of self-intersections nor optimality (i.e.

minimum number of surviving points) of the result. Its worst-case running time is $\Theta(n^2)$, though better running times are experienced in practice. The algorithm by Imai/Iri [5] guarantees as a result a minimum number of surviving points, but not absence of self-intersections. Its running time is $O(n^3)$ in its original version, but faster variants with $O(n^2)$ running time exist. Estkowski and Mitchell [4] have shown, that from a theoretical point of view, solving MSP or MSTOPOP optimally is a hopeless enterprise. They prove that for MSTOPOP it is NP-hard to obtain an approximate solution better than within a factor of $n^{1/5-\delta}$ for any $\delta > 0$. Their result carries over to MSP since topology constraint points do not play a role in their proof of approximation-hardness. In [1] de Berg et al. consider heuristic solutions to MSTOPOP and MSP, yet without a comparison with the respective optimum solutions. For MSP, an implementation is available in the CGAL library [6] following [3]. To our knowledge, no study has been conducted investigating how close to the optimum heuristic solutions are for MSTOPOP (due to lack of an exact solution).

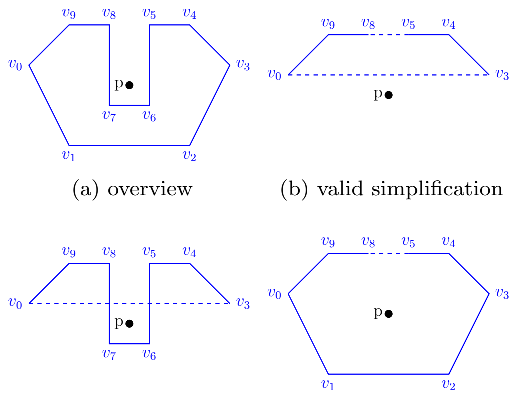
As a side note, during the GISCup'14 – a competition held during the ACM SIGSPATIAL conference 2014 – a variant of the problem (without a precision constraint – i.e., $\epsilon = \infty$) was tackled by several teams.

1.2 Our Contribution

We define a *local* variant of the map simplification with topology constraints problem (which theoretically is still hard to approximate) and derive a respective ILP formulation which can solve instances of moderate size optimally. We then develop a heuristic algorithm based on constrained triangulations and local simplification steps which empirically can be shown to produce close-to-optimal results for moderately sized instances (via comparison to the ILP solution). In contrast to the ILP solution this heuristic can also be used to solve large instances as they naturally occur in the map rendering domain.

2 Local Topology-Consistency

At first sight one might think that solving MSTOPOP is exactly what we want for our map rendering application. Consider the example in Figure 2(a), where we have a planar subdivision with two faces – one U-shaped face bounded by $v_0v_1 \dots v_9v_0$ and an outer face which also contains a topology constraint point p. For sufficiently large value of ϵ, the simplification shown in 2(b) is indeed a valid simplification according to MSTOPOP since p still lies in the outer face. This might be somewhat counterintuitive since p somehow 'switched sides' (even though topologically it is, of course, still on the right side). In particular, if we locally inspect the shortcut v_0v_3 which replaces the chain $v_0v_1v_2v_3$ there is indeed a switch of sides (c) – which is only healed topologically by shortcutting $v_5v_6v_7v_8$ by v_5v_8, which also is invalid on its own (d).



(a) overview (b) valid simplification

(c) self-intersection (d) topology constraint violated

Figure 2: Example of two simplifications, that are not valid alone but only in conjunction with each other.

We believe that it is not unnatural to demand that shortcuts *locally* don't make points switch sides. To that end we define the following local criterion to decide whether a shortcut is considered topology preserving.

Definition 1 *For given $\epsilon > 0$ and constraint point set P, a shortcut u_1u_k is considered a* valid shortcut *for the polygonal chain $C = u_1u_2 \dots u_k$ if*

- *the distance of u_i, $1 < i < k$ to the segment u_1u_k is at most ϵ.*
- *the polygon (possibly with self-intersections) defined by the polygonal chain $C' = u_1u_2 \dots u_ku_1$ does not contain[1] a constraint point.*

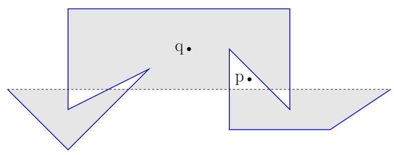


Figure 3: Example of a possible simplfciation. Point q would make this simplification locally inconsistent, whereas point p would allow this simplification. The interior of the polygon is given by the shaded area.

See Figure 3 for an illustration of this definition. Armed with this notion of a valid shortcut we can formally define the map simplification variant that we will be dealing with in the following.

Definition 2 (MSLOCTOPOP) *For a planar subdivision given as a straight-line embedding of a graph*

[1]With the interior of a (possibly complex) polygon defined by the *even-odd rule*. See Figure 3 for an example.

$G(V, E)$, *a set of constraint points* $P \subset \mathbb{R}^2$, *and an* $\epsilon > 0$, *the goal of the* Map Simplification with LOCal TOPOlogy constraints Problem (MSLOCTOPOP) *is to simplify degree-2 chains of G using non-intersecting valid shortcuts such that the total number of remaining vertices is minimized.*

As MSLOCTOPOP comprises MSP as a special case (no topology constraints), the hardness of approximation result in [4] carries over, hence there is little hope to find a polynomial-time approximation algorithm which solves MSLOCTOPOP with an approximation ratio substantially better than $n^{1/5}$.

3 An Integer Linear Programming Formulation for MSLOCTOPOP

As we have seen, even our specialization MSLOCTOPOP of MSTOPOP is hard to approximate, yet using an integer linear programming (ILP) formulation one might be able to obtain optimal solutions for many instances that occur in practice. We will develop a respective ILP in the following step by step.

Let us first concentrate on a single polygonal chain $C^l = p_1 p_2 \dots p_{n_l}$ of the planar subdivision where each p_i with $1 < i < n_l$ is a degree-2 node of the subdivision, p_1 and p_{n_l} are nodes with degree $\neq 2$. We first construct the set $S^l := \{s_1^l, s_2^l, \dots s_{k_l}^l\}$ of *valid shortcuts* for C^l. Note that the original edges are also valid shortcuts and $k_l \in O(n_l^2)$. Essentially we want to construct a path from p_1 to p_{n_l} using as few valid shortcuts as possible, so we introduce 0-1 variables $x_1^l, x_2^l, \dots x_{k_l}^l$ where $x_i^l = 1$ denotes that the shortcut s_i^l should be realized, $x_i^l = 0$ that it should not be used. As constraints we demand:

$$\sum_{s_j^l=(p_1,.)} x_j^l = 1 \tag{1}$$

$$\sum_{s_j^l=(,.p_{n_l})} x_j^l = 1 \tag{2}$$

that is, we select exactly one shortcut from S^l that is adjacent to p_1 and likewise for p_{n_l}. For every other vertex p_i, $1 < i < n_l$ of C^l we want that the number of incoming shortcuts equals the number of outgoing shortcuts (in fact both equal to 0 or to 1, but in our case there is no need to explicitly enforce that):

$$\forall 1 < i < n_l : \sum_{s_j^l=(.,p_i)} x_j^l - \sum_{s_j^l=(p_i,.)} x_j^l = 0 \tag{3}$$

We construct variables and respective constraints for each polygonal degree-2 chain in the planar subdivision. Then for every *intersecting* pair of shortcuts s_i^l, s_j^g (of the same or different polygonal degree-2 chains) we add a constraint

$$x_i^l + x_j^g \leq 1 \tag{4}$$

preventing the usage of both shortcuts simultaneously. The objective function is simply a minimization of the sum of all variables

$$\min \sum x_i^l \tag{5}$$

MSLOCTOPOP being NP-hard to approximate, we cannot expect our ILP formulation to be solvable efficiently for every input instance. Yet, instances occurring in real-world scenarios might well be solvable with a good ILP solver.

4 A Local Simplification Heuristic

In this section we present a heuristic to solve the MSLOCTOPOP problem. We iteratively remove single points of the subdivision by only inspecting local neighborhoods, yet preserving validity of the overall simplification. The basic idea is similar to [3] but also incorporates topology constraints.

We employ a *Constrained Triangulation (CT)* with the points being all vertices of the original subdivision as well as the constraint points, and the constraining edges being the edges of the subdivision. Let $\deg_G(v)$ denote the number the constraining edges adjacent to a node in the current CT.

Now for a given node v with $\deg_G(v) = 2$ one can quickly decide whether it can be removed and replaced by a shortcut without violating any topology constraint or creating intersections. Let v_1 and v_2 be the two neighbours of v in the current subdivision. There are two cases for which we can easily see, that we may *not* discard v:

- The nodes v, v_1 and v_2 form a triangle in the current planar subdivision. Removing v leads to a collapse of the 2-dimensional face spanned by the 3 nodes.
- The distance of v to the segment $\overline{v_1 v_2}$ is greater than ϵ. Then the shortcut $v_1 v_2$ is not valid.

If none of these cases applies we also demand that there exists no point $u \neq v_1, v_2$ adjacent (in the triangulation) to v which lies in the triangle $\triangle v_1 v v_2$, otherwise:

- If u is part of the constraint points in the subdivision, the simplification is invalid according to our *local topology-consistency*.
- If u is part of the subdivision boundary, the removal of v either introduces an intersection or changes the orientation of a face.

Given these criteria the complete algorithm is quite simple. For every point we compute whether it is removable with respect to these criteria and remove it if this is the case. This procedure is repeated until no more points can be removed.

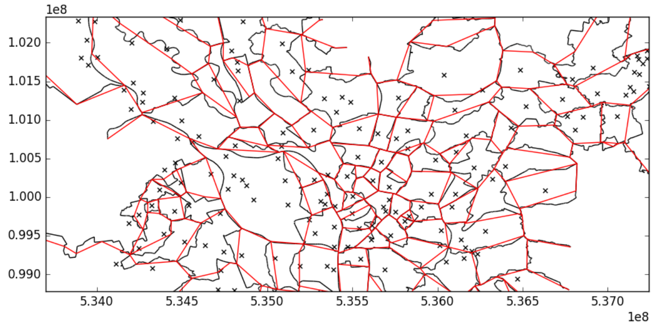


Figure 4: Hamburg data set, $\epsilon = 10^5$.

Note that we have to make sure, that the distance check is performed with respect to *all* the points that have possibly been replaced by a shortcut.

5 Experimental Results

We have compared both solution approaches on synthetic and real-world data, yet due to space restrictions we only report on some selected instances. The experiments were run on a standard laptop with an Intel Core i5/1.9GHz/12GB RAM. The local simplification heuristic uses the CGAL library [6], the ILPs were solved using the Gurobi solver. Clang 3.7 with the -O3 flag was used for compilation. From the OpenStreetMap project we extracted the datasets **HAM** – the administrative subdivision of the city of Hamburg with some POIs as topology constraint points, see Figure 4 – and **GMNY** – the country borders for Germany with all cities and towns as constraint points. From the GISCup'14 (`http://mypages.iit.edu/~xzhang22/GISCUP2014/`), the planar subdivisions **GIS4** and **GIS5** including topology constraint points were used.

Table 1 lists the results. For example, for a tolerance of $\epsilon = 10^4$ and the dataset HAM, the ILP approach reduces the number of surviving degree-2-nodes to 992 within 19 seconds. The heuristic approach, on the other hand takes only 0.4 seconds to obtain a result with 1219 surviving degree-2-nodes. So while not as small as the optimum ILP result, the heuristic result is reasonably close. For the large GMNY instance, the ILP approach could not determine a solution within one hour whereas the heuristic produced a solution within 14 seconds. For GIS4 and GIS5, the heuristic also produces solutions pretty close to the ILP optimum. For the latter two data sets we also had running times and result sizes of the runner-up algorithm at the 2014 GISCup – here called CROSS (we could not get hands on the winning algorithm). Note though, that the objective of the GISCup was not just minimization of the remaining subdivision but rather the ratio of removed points per unit of time. So while being blazingly fast, CROSS retains a lot more points even compared to our heuristic approach.

	GIS4	GIS5	HAM	GMNY
# nodes	26198	25203	10233	217863
# constraints	356	1607	194	97639
$\epsilon = 10000$				
ILP time (s)	296	105	19	-
Heur. time(s)	1.3	1.1	0.4	14
ILP output	348	476	992	-
Heur. output	433	566	1219	≈11k
$\epsilon = 100000$				
ILP time (s)	419	121	56	-
Heur. time(s)	1.3	1.2	0.4	17
ILP output	88	238	57	-
Heur. output	101	275	67	1864
$\epsilon = \infty$				
ILP time (s)	439	130		
Heur. time(s)	1.4	1.2		
CROSS time(s)	0.01	0.01		
ILP output	88	237		
Heur. output	100	274		
CROSS output	1759	2826		

Table 1: Pruning results for our algorithms: running times and size of the output.

References

[1] Marc de Berg, Marc van Kreveld, and Stefan Schirra. A new approach to subdivision simplification. In *ACSM/ASPRS Annual Convention & Exposition Technical Papers*, pages 79–88, Charlotte, North Carolina, USA, 1995. ACSM.

[2] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973. doi:10.3138/FM57-6770-U75U-7727.

[3] Christopher Dyken, Morten Dæhlen, and Thomas Sevaldrud. Simultaneous curve simplification. *J. of Geographical Systems*, 11(3):273–289, 2009.

[4] Regina Estkowski and Joseph S. B. Mitchell. Simplifying a polygonal subdivision while keeping it simple. In *Proc. 17th Ann. Symp. on Comp. Geom.*, SCG '01, pages 40–49. ACM, 2001.

[5] Hiroshi Imai and Masao Iri. Polygonal approximations of a curve – formulations and algorithms. In G. T. Toussaint, editor, *Computational Morphology*, pages 71–86. Elsevier Science, 1988.

[6] The CGAL Project. *CGAL User and Reference Manual*. CGAL Editorial Board, 4.7 edition, 2015.