Label Placement in Road Maps

Andreas Gemsa [*] Benjamin Niedermann [*] Martin Nöllenburg [*]

Abstract

A road map can be interpreted as a graph embedded in the plane, in which each vertex corresponds to a road junction and each edge to a particular road segment. We consider the cartographic problem to place non-overlapping road labels along the edges so that as many road segments as possible are covered by labels. We show that this is NP-hard in general, but can be solved in polynomial time if the road map is an embedded tree. Moreover, we give an efficient approximation algorithm for general road maps, assuming that each individual road can be labeled optimally in polynomial time.

1 Introduction

Map labeling is a well-known cartographic problem in computational geometry [7]. Depending on the type of map features, one can distinguish labeling of *points*, *lines*, and *areas*. Common cartographic quality criteria are that labels must be disjoint and clearly identify their respective map features. Most of the previous work concerns point labeling, while labeling line and area features received considerably less attention. In this paper we address labeling linear features, namely roads in a (static) road map.

Geometrically, a *road map* is the representation of a *road graph* G as an arrangement of fat curves in the plane $\mathbb{R}^2$. Each *road* is a connected subgraph of G (typically a simple path) and each edge belongs to exactly one road. Roads may intersect each other in *junctions*, the vertices of G, and we denote an edge connecting two junctions as a *road segment*. In road labeling the task is to place the road names optimally inside the fat curves representing the various road segments, see Fig. 1.

Chirié [1] presented a set of rules and quality criteria for label placement in road maps based on interviews with cartographers. This includes that (i) labels are placed inside and parallel to the road shapes, (ii) every road section between two junctions should be clearly identified, and (iii) no two road labels may intersect. Further, he gave a mathematical description for labeling a single road and introduced a heuristic for sequentially labeling all roads. Wolff et al. [6] took a more algorithmic perspective for a single linear feature to be labeled. Seibert and Unger [5] considered road networks that are grid-shaped. They showed that in those networks it is NP-complete to decide whether for every road at least one label can be placed. Yet, Neyer and Wagner [4] introduced an empirically efficient algorithm that finds such a labeling if possible.

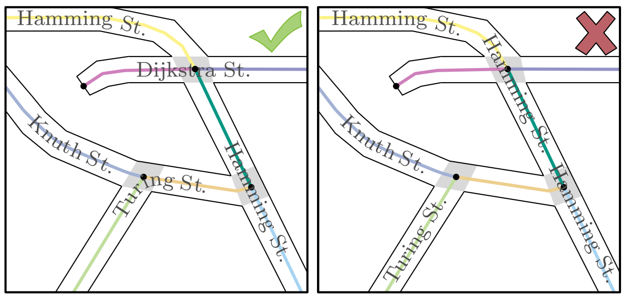

Figure 1: Two ways to label the same road network. Each road segment has its own color. Junctions are marked gray. The left image identifies all road segments.

While in grid-shaped road networks it is sufficient to place a single label per road to clearly identify all its road segments, this is not the case in general road networks. Consider the example in Fig. 1. Here, it is not obvious whether the orange road segment in the center belongs to *Knuth St.* or to *Turing St.* Simply maximizing the number of placed labels, as often done for labeling point features, can cause undesired effects like unnamed roads or clumsy label placements (e.g., around *Dijkstra St.* and *Hamming St.* in the right example of Fig. 1). Therefore, in contrast to Seibert and Unger [5], we aim for maximizing the number of *identified* road segments, i.e., road segments that can be clearly assigned to labels. Based on the criteria described by Chirié [1] we introduce a new and versatile model for road labeling. We show that the problem of maximizing the number of identified road segments is NP-hard. For the special case that the road graph is a tree, we present a polynomial-time algorithm and for the general case we give an efficient approximation algorithm, assuming that each individual road of the network can be labeled optimally in polynomial-time.

2 Preliminaries

As argued above, a road map is a collection of fat curves in the plane, each representing a particular piece of a named road. If two (or more) such curves

[*]Institute of Theoretical Informatics, Karlsruhe Institute of Technology (KIT), Germany. Email: {lastname}@kit.edu

intersect, they form junctions. A *road label* is again a fat curve (the bounding shape of the road name) that is contained in and parallel to the fat curve representing its road. We observe that labels of different roads can intersect only within junctions and that the actual width of the curves is irrelevant, except for defining the shape and size of the junctions. These observations allow us to define the following more abstract but basically equivalent road map model.

A *road map* is a planar *road graph* $G = (V, E)$ together with a planar embedding $\mathcal{E}(G)$, which can be thought of as the geometric representation of the road axes as thin curves. We denote the number of vertices of G by n, and the number of edges by m. Observe that since G is planar $m = O(n)$. Each edge $e \in E$ is either a *free edge*, which is not part of a junction, or a *junction edge*, which is part of a junction; see Fig. 2a. Each vertex $v \in V$ is either a *junction vertex* incident only to junction edges, or a *regular vertex* incident to one free and at most one junction edge. A junction vertex v and its incident edges are denoted as a *junction*. The edge set E decomposes into a set $\mathcal{R}$ of edge-disjoint *roads*, where each road $R \in \mathcal{R}$ induces a connected subgraph of G with the property that the end points of each free edge are regular vertices, and each junction edge is incident to one regular vertex and one junction vertex. We denote each free edge of a road including its two incident vertices (and their embedding) as a *road segment*. Thus a road decomposes into road segments, separated by junction vertices and their incident junction edges. Further, each road $R \in \mathcal{R}$ has a road name of length $\lambda(R)$.

For simplicity, we identify the embedding $\mathcal{E}(G)$ and the points in the plane covered by $\mathcal{E}(G)$. We also use $\mathcal{E}(v)$, $\mathcal{E}(e)$, and $\mathcal{E}(R)$ to denote the embeddings of a vertex v, an edge e, and a road R.

A *valid label* ℓ for a road R is a simple open curve $\ell \subseteq \mathcal{E}(R)$ of length $\lambda(R)$ whose end points must lie on road segments and not on junction edges or junction vertices. The start point of a valid label is denoted as the *head* $h(\ell)$ and the end point as the tail $t(\ell)$. A *labeling* $\mathcal{L}$ for a road map with road set $\mathcal{R}$ is a set of valid labels that are mutually non-overlapping, where we say that two labels ℓ and ℓ' *overlap* if they intersect in a point that is not their respective head or tail. Finally, we say that a label ℓ *identifies* a road segment r with free edge e if $\ell \cap \mathcal{E}(e) \neq \emptyset$.

Following the above mentioned cartographic quality criteria for labeled road maps [1], our goal is to find a labeling $\mathcal{L}$ that maximizes the number of identified road segments. We call this problem MaxIdentifiedRoads.

3 Computational Complexity

We can prove that MaxIdentifiedRoads for a general road map is NP-hard and that the corresponding decision problem is NP-complete. The hardness proof is by reduction from planar 3-SAT, which is well-known to be NP-hard [3]. The corresponding decision problem takes as input a road map together with an integer k and asks whether a labeling exists such that at least k road segments are identified.

In order to prove that this decision problem is in NP, we make the observation that a label ℓ can be described by its head $h(\ell)$ and a sequence σ of road segments and junctions that ℓ identifies and crosses, respectively. We create an oracle that first guesses a number $z \leq k$ of labels, and for each of the z labels a sequence σ of road segments and junctions. Thus, we obtain a set of labels for which the heads are not fixed yet. We now need to determine whether a labeling exists with the prescribed sequence of road segments and junctions for each label. Checking whether labels of the induced labeling overlap at junctions is straightforward. Determining whether all labels that identify the same road can be placed without overlaps can be done with a linear program in polynomial time. Due to space constraints we omit further details.

Theorem 1 MaxIdentifiedRoads *is NP-hard and its corresponding decision problem is NP-complete.*

4 Efficient Algorithm for Tree-Shaped Road Maps

In this section we assume that the underlying road graph of the road map is a tree $T = (V, E)$. We present a polynomial-time algorithm to solve MaxIdentifiedRoads optimally for trees.

First, we require some additional notation. We assume that T is rooted at an arbitrary leaf ρ. For two points $p, q \in \mathcal{E}(T)$ we define $\mathrm{d}(p, q)$ as the length of the shortest curve in $\mathcal{E}(T)$ that connects p and q. We call a point $p \in \mathcal{E}(T)$ *free*, if $p \in \mathcal{E}(e)$ for a free edge $e \in E$. For a valid label ℓ, we call $p \in \ell$ the *lowest point* p of ℓ if $\mathrm{d}(p, \mathcal{E}(\rho)) \leq \mathrm{d}(q, \mathcal{E}(\rho))$ for any $q \in \ell$. As T is a tree, the point p is uniquely determined. We distinguish two types of valid labels. A valid label ℓ is *vertical* if $h(\ell)$ or $t(\ell)$ is the lowest point of ℓ; otherwise we call ℓ *horizontal* (see Figure Fig. 2a). Without loss of generality we can assume that the lowest point of each vertical label ℓ is its head $h(\ell)$.

A basic, but crucial, observation for our algorithm is that for any road map there exists an optimal solution, in which all labels are pushed as far as possible towards the leaves in the tree. More specifically, we can transform any optimal labeling into a *canonical labeling* by moving each label away from the root and towards the leaves as far as possible while its head and tail must remain on their respective road segments. For a vertical label this direction is unique, while for horizontal labels we can choose any of the

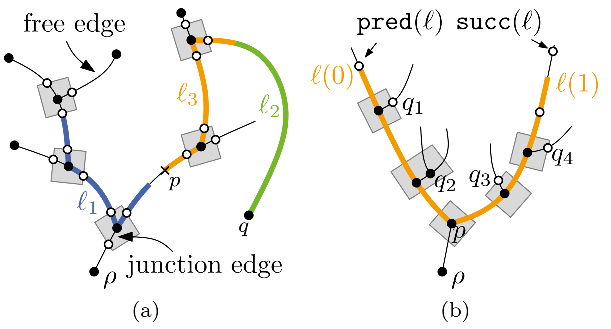

Figure 2: (a) A tree-shaped road map with canonical labeling. While label ℓ_1 is horizontal, ℓ_2 and ℓ_3 are vertical. The white circles represent regular vertices, the black circles junction vertices. (b) Illustration of the set $N(\ell)$ for a horizontal label. In that case $N(\ell) = \{q_1, q_2, q_3, q_3, \texttt{pred}(\ell), \texttt{succ}(\ell)\}$.

two. Then, for each label its head or tail either coincides with a leaf of T, with some internal regular vertex, or with the head of another label. For an illustration see Fig. 2a, where the tail of the horizontal label ℓ_1 coincides with a regular vertex, the tail of the vertical label ℓ_2 coincides with the leaf q, and the tail of ℓ_3 coincides with the head of ℓ_2. Among all optimal canonical labelings we can further restrict ourselves to a labeling with smallest number of labels. Hence there is no label that identifies only a single road segment and at the same time touches another label.

Using this observation, we determine a finite set of candidate positions for the heads or tails of labels in an optimal canonical labeling, and transform T into a tree $T' = (V', E')$ by subdividing some of T's edges so that it contains a vertex for every candidate position. To that end we construct for each vertex $v \in V$ a chain of tightly packed vertical labels, which starts at $\mathcal{E}(v)$, is directed to ρ, and ends when either the road ends, or adding the next label does not increase the number of identified road segments. We construct such a chain as follows. We place a valid, vertical label ℓ_1 such that $t(\ell_1) = \mathcal{E}(v)$; recall that we assume that the lowest point of each vertical label is its head. For $i = 2, 3, \dots$ we add a new valid vertical label ℓ_i with $t(\ell_i) = h(\ell_{i-1})$, if ℓ_i lies on the same road as ℓ_{i-1}, and ℓ_i's head and tail lie on different road segments. We use the heads of all those labels to subdivide the tree T obtaining T', i.e., for each head $h(\ell)$ of a label ℓ we add a vertex v' to V' embedded at $h(\ell)$. Since each chain consists of $O(n)$ labels the cardinality of V' is $O(n^2)$.

Observation 1 *For any canonical labeling $\mathcal{L}$ of the given road map and any label $\ell \in \mathcal{L}$ there exists a vertex v in the subdivision T' of T with $\mathcal{E}(v) = h(\ell)$ or $\mathcal{E}(v) = t(\ell)$.*

For our dynamic programming algorithm we introduce for each vertex p in T' a set $C(p)$ of *candidate labels*. This set contains all labels of any canonical labeling $\mathcal{L}$ whose lowest point is $\mathcal{E}(p)$. We obtain $C(p)$ for each vertex p as follows. Let T'_p be the subtree of T' that is rooted at p. If $\mathcal{E}(p)$ is free, the set $C(p)$ contains all valid vertical labels ℓ that start at $\mathcal{E}(p)$ and that completely lie in $\mathcal{E}(T'_p)$, i.e., $C(p)$ is the set of vertical labels whose lowest point is $\mathcal{E}(p)$. If p is a junction vertex, consider for each free vertex q of T'_p all valid labels that start at $\mathcal{E}(q)$, contain $\mathcal{E}(p)$ and lie completely in $\mathcal{E}(T'_p)$. In that case $C(p)$ denotes that set of labels, i.e., $C(p)$ is the set of horizontal labels whose lowest point is $\mathcal{E}(p)$.

We are now ready to describe our dynamic programming algorithm. We create a one-dimensional table DP, in which we store for each vertex $p \in V'$ the *weight* of the optimal solution for the subtree T'_p. We define the weight as the number of identified road segments in the subtree. In order to avoid counting road segments twice that are identified by multiple labels we use an auxiliary table CR, in which we store for each vertex $p \in V'$ whether there is a label with lowest point $\mathcal{E}(p)$ that identifies the road segment containing $\mathcal{E}(p)$. If so, we set the value of $\mathrm{CR}[p] = 1$. All entries of CR are initially set to 0.

The algorithm initializes the table entries for all leaves of T' with 0. We then determine the values of the remaining entries in a bottom-up fashion. For each vertex $p \in V'$ for which the entries in DP for all vertices in T'_p have already been determined, we compute the value of $\mathrm{DP}[p]$ as follows. We iterate over all labels in the set $C(p)$ of candidate labels. When considering a candidate ℓ we observe that choosing it separates the tree T'_p into several independent subtrees. For each of these we have already computed an optimal solution. Hence, we determine for each candidate ℓ the set of vertices in V' that are embedded on ℓ, which we call *terminals*. The independent subtrees are themselves rooted at the neighbors of the terminals that are not covered by ℓ. We denote the set of these roots by $N(\ell)$. Note that in the case that $h(\ell)$ or $t(\ell)$ are embedded on a vertex of T', this vertex is also the root of one of these independent subtrees.

In principle we are now ready to compute the value of $\mathrm{DP}[p]$, but we need to take special care not to count already identified road segments again. For this we introduce the following notation. For a valid label ℓ we denote the vertex in T' that is embedded on the same edge as $t(\ell)$ and *not* covered by ℓ by $\texttt{pred}(\ell)$. Should $t(\ell)$ be embedded on the same point as a vertex $v \in V'$ we instead set $\texttt{pred}(\ell) = v$. For horizontal labels we also denote the vertex in T' that is embedded on the same edge as $h(\ell)$ and *not* covered by ℓ by $\texttt{succ}(\ell)$. Should $h(\ell)$ be embedded on the same point as a vertex $v \in V'$ we instead set $\texttt{succ}(\ell) = v$. Now, let $c(\ell)$ denote the number of road segments identified by ℓ. Then the *value* of ℓ is $w(\ell) = c(\ell) +$

$\sum_{q \in N(\ell)} \mathrm{DP}[q] - \mathrm{CR}[\mathtt{succ}(\ell)] - \mathrm{CR}[\mathtt{pred}(\ell)]$ if ℓ is horizontal and $w(\ell) = c(\ell) + \sum_{q \in N(\ell)} \mathrm{DP}[q] - \mathrm{CR}[\mathtt{pred}(\ell)]]$ if ℓ is vertical. Now for each vertex p of T', we define

$$\mathrm{DP}[p] := \max \left[\{w(\ell) \mid \ell \in C(p)\} \cup \{ \sum_{q \text{ child of } p} \mathrm{DP}[q]\} \right].$$

So far we have not mentioned, how we compute $\mathrm{CR}[p]$. Since we know which label was chosen for $\mathrm{DP}[p]$ we can easily determine whether it covers the road segment p lies on and set the value of $\mathrm{CR}[p]$ accordingly. Note that this applies only to free vertices; for junction vertices p we always set $\mathrm{CR}[p] = 0$.

A naive implementation of our algorithm has a time complexity of $O(n^6)$ and requires $O(n^2)$ space. However, we can improve these bounds by storing only parts of T' when computing a single entry of DP, and by computing $\max\{w(\ell) \mid \ell \in C(p)\}$ without considering $C(p)$ explicitly (details omitted). These improvements yield the following theorem.

Theorem 2 *For a road map with a tree as underlying road graph,* MaxIdentifiedRoads *can be solved in $O(n^3 \log n)$ time using $O(n)$ space.*

Besides the *primary objective* to identify a maximum number of road segments, Chirié [1] also suggested several additional *secondary objectives*, e.g., labels should overlap as few as possible crossroads or labels should bend as little as possible. Note that our approach allows us to easily incorporate those secondary objectives. We just need to adapt $w(\ell)$ for $\ell \in L$ by some penalty. In this case, however, not all improvements of our algorithm can still be applied.

5 Approximation for General Road Maps

In this section we sketch an approximation algorithm for MaxIdentifiedRoads on an arbitrary road map, assuming that we can compute an optimal labeling for each individual road R separately in polynomial time. We denote the number of identified road segments of the optimal labeling of road R by $w(R)$.

We first construct a *conflict graph* of the roads, i. e., we construct a graph $G' = (\mathcal{R}, F)$ where we have for each road $R \in \mathcal{R}$ a vertex, and we have an edge between two vertices if the roads share a common junction. We then approximate a solution for the entire road map by computing a weighted maximum independent set in the conflict graph G'. We use a result by Kako et al. [2] who gave an approximation algorithm for computing maximum weighted independent sets in sparse graphs. They introduced the concept of the *weighted degree*. In our case this means that for each vertex $R \in \mathcal{R}$ its weighted degree is $\bar{d}(R) = \sum_{R' \in N(R)} w(R')/w(R)$, where $N(R)$ denotes the neighbors of R in G'.

Theorem 3 *There is an approximation algorithm for* MaxIdentifiedRoads *with approximation ratio $1/(\bar{d}_w + 1)$, where $\bar{d}_w$ is the average weighted degree of the graph. This algorithm has a time complexity of $O(\Delta^3 \cdot |\mathcal{R}| \log |\mathcal{R}| + A(\mathcal{R}))$, where Δ is the maximum degree of the graph and $A(\mathcal{R})$ is the time required to obtain individual optimal solutions for all roads in $\mathcal{R}$.*

Heuristically, we can improve the algorithm by iteratively applying it to the remaining unlabeled roads, ensuring that we do not produce conflicts with already placed labels.

The results of Theorems 2 and 3 imply that we can approximate MaxIdentifiedRoads in $O(\Delta^3 \cdot |\mathcal{R}| \log |\mathcal{R}| + |\mathcal{R}| \cdot n^3 \log n) = O(|\mathcal{R}| \cdot n^3 \log n)$ time if each road in $\mathcal{R}$ is a tree.

Initial experiments seem to suggest that for real-world instances the weighted degree $\bar{d}_w$ is small and that most roads are paths or trees of small size.

6 Conclusions

We investigated the problem of labeling road maps and showed that it is NP-hard in general, but can be solved in polynomial time for trees, and there exists an approximation algorithm for the general case.

As next step, we aim to develop further exact and approximation algorithms for other kinds of road networks. Additionally, we are planning to implement our algorithms and evaluate their performance on real world networks.

References

[1] F. Chirié. Automated Name Placement With High Cartographic Quality: City Street Maps. *Cart. and Geo. Inf. Science*, 27(2):101–110, 2000.

[2] A. Kako, T. Ono, T. Hirata, and M. M. Halldórsson. Approximation algorithms for the weighted independent set problem in sparse graphs. *Discr. Appl. Math.*, 157(4):617–626, 2009.

[3] D. Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.

[4] G. Neyer and F. Wagner. Labeling Downtown. In *Algorithms and Complexity*, volume 1767 of *LNCS*, pages 113–124. Springer, 2000.

[5] S. Seibert and W. Unger. The Hardness of Placing Street Names in a Manhattan Type Map. In *Algorithms and Complexity*, volume 1767 of *LNCS*, pages 102–112. Springer, 2000.

[6] A. Wolff, L. Knipping, M. van Kreveld, T. Strijk, and P. K. Agarwal. A simple and efficient algorithm for high-quality line labeling. In *Innovations in GIS VII: GeoComputation*, chapter 11, pages 147–159. Taylor & Francis, 2000.

[7] A. Wolff and T. Strijk. The Map Labeling Bibliography. `http://liinwww.ira.uka.de/bibliography/Theory/map.labeling.html`, 2009.