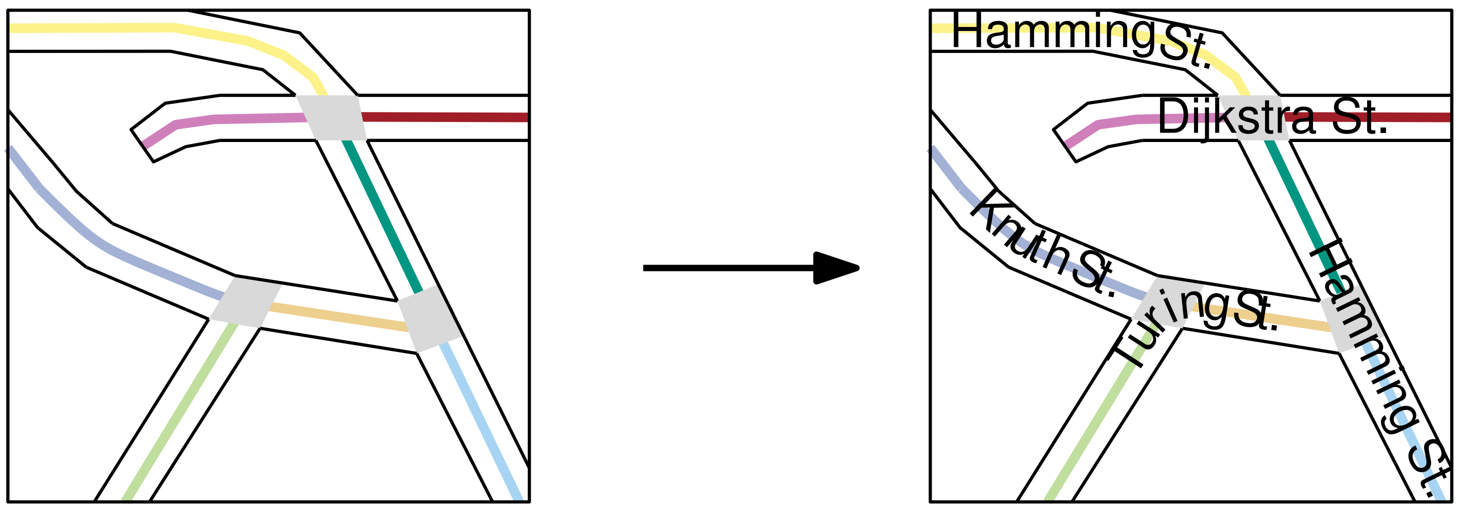
Karlsruhe Institute of Technology

Label Placement in Road Maps

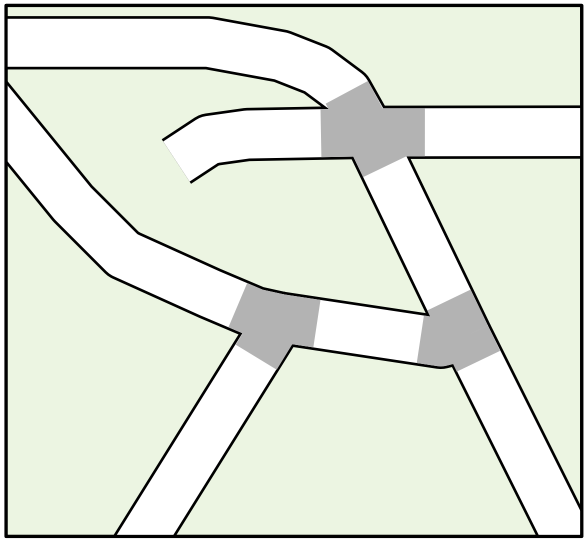
Andreas Gemsa, Benjamin Niedermann, Martin Nöllenburg

INSTITUTE OF THEORETICAL INFORMATICS · KARLSRUHE INSTITUTE OF TECHNOLOGY

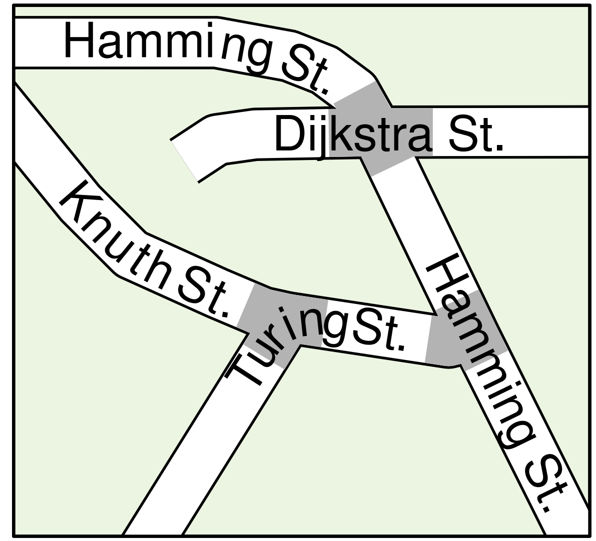


Introduction

Given: Road map.



Find: *Good* overlapping-free labeling



Questions discussed in this talk:

- Why should we consider road labeling?
- How to model the problem of road labeling?
- What is the computational complexity of labeling a road map?
- What algorithms can be found for labeling road maps?

Motivation

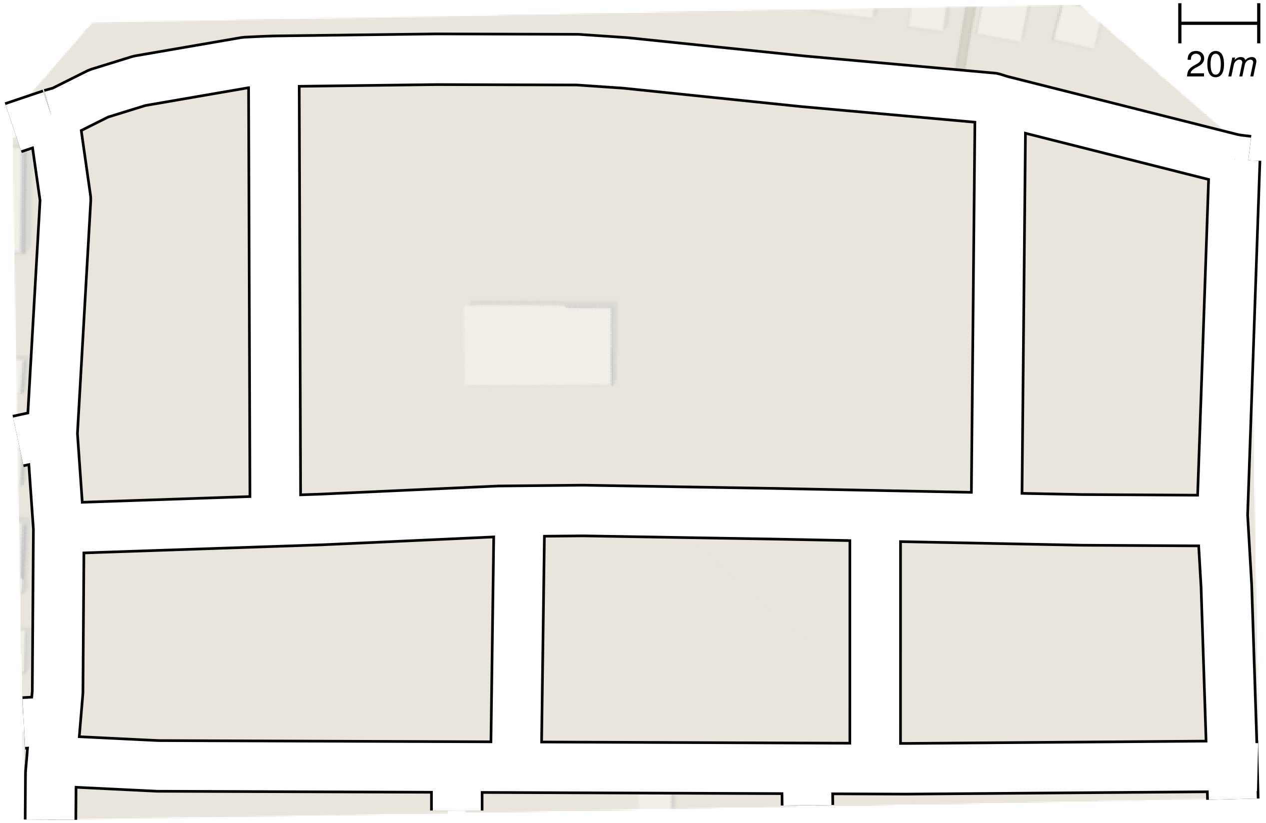
Why considering road labeling?

Example: Google Maps

Location: Karlsruhe Hermann-Vollmerstraße

Why considering road labeling?

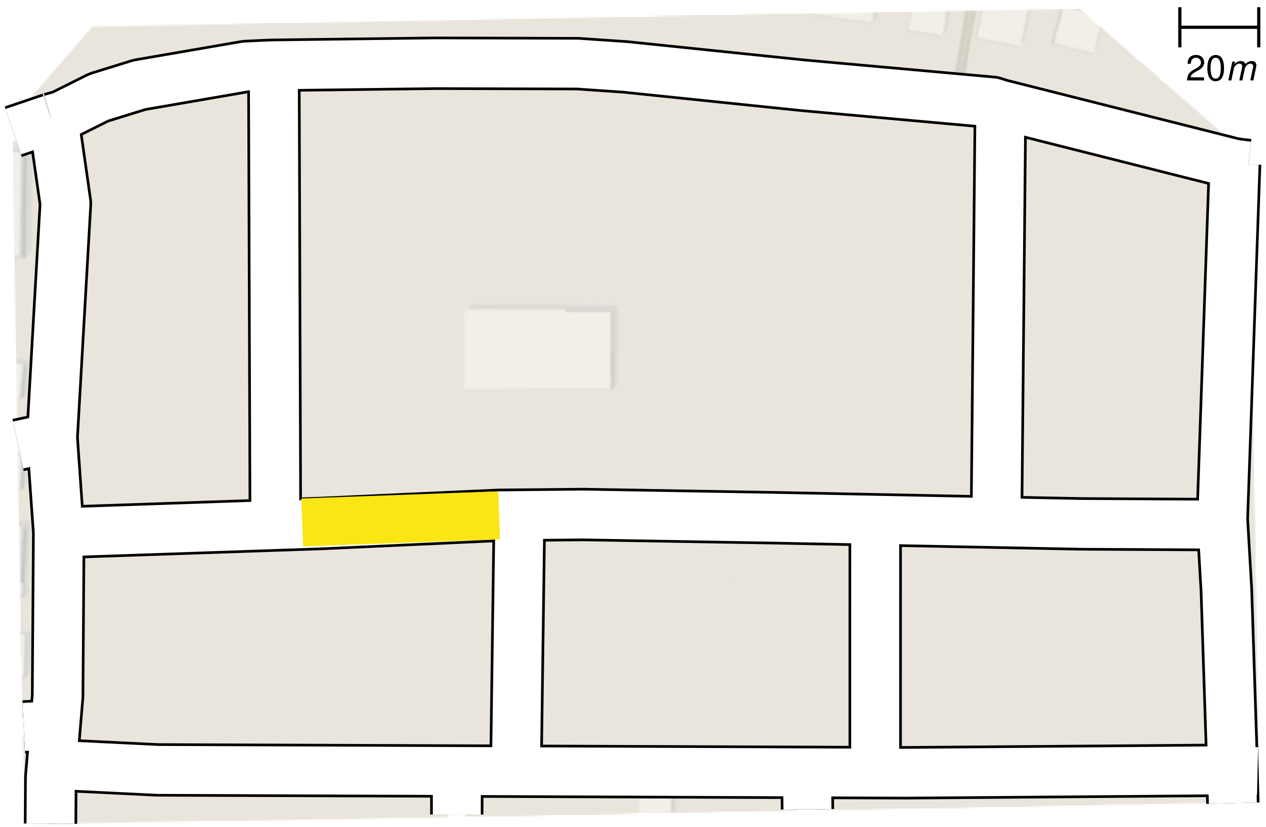
Example: Google Maps



Location: Karlsruhe Hermann-Vollmerstraße

Why considering road labeling?

Example: Google Maps



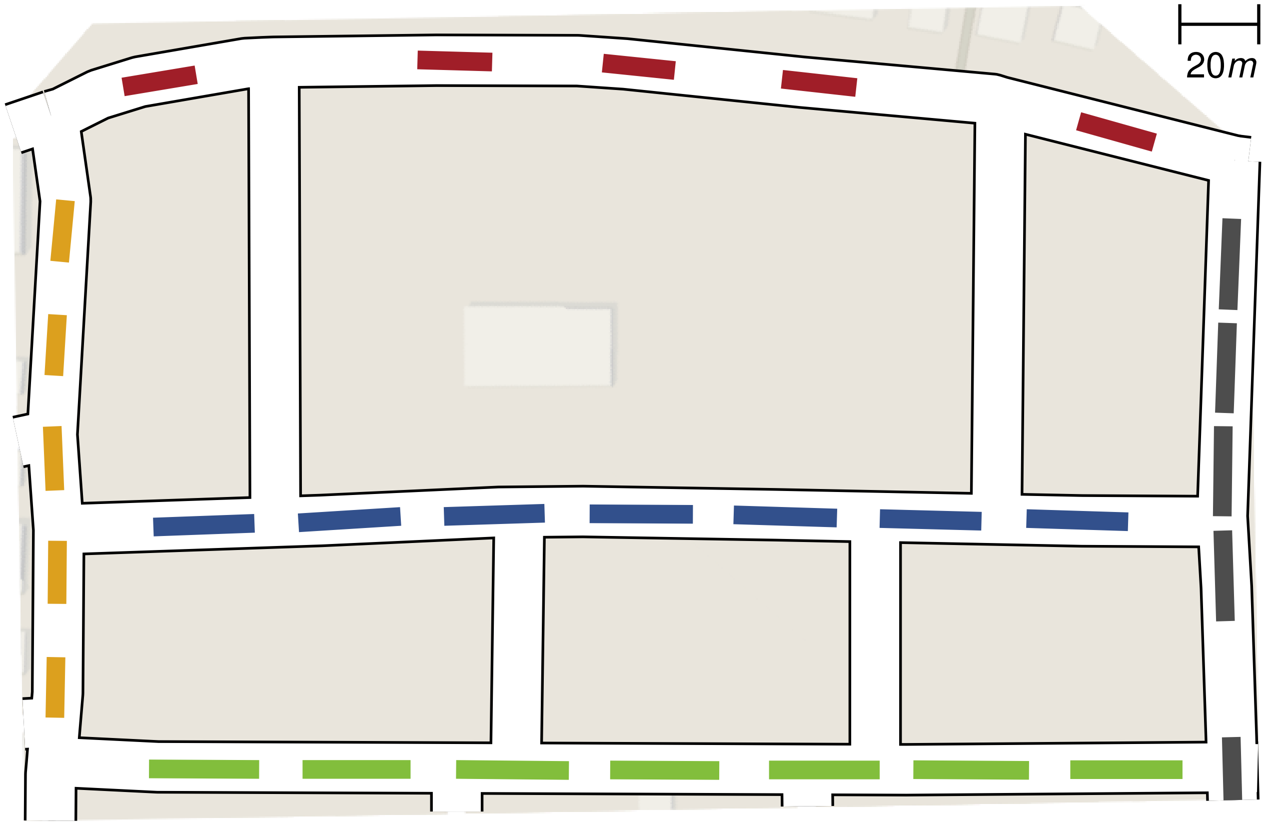
Map:
7 roads
19 road sections

Location: Karlsruhe Hermann-Vollmerstraße

road section = part of road between two junctions

Why considering road labeling?

Example: Google Maps



Map:
7 roads
19 road sections

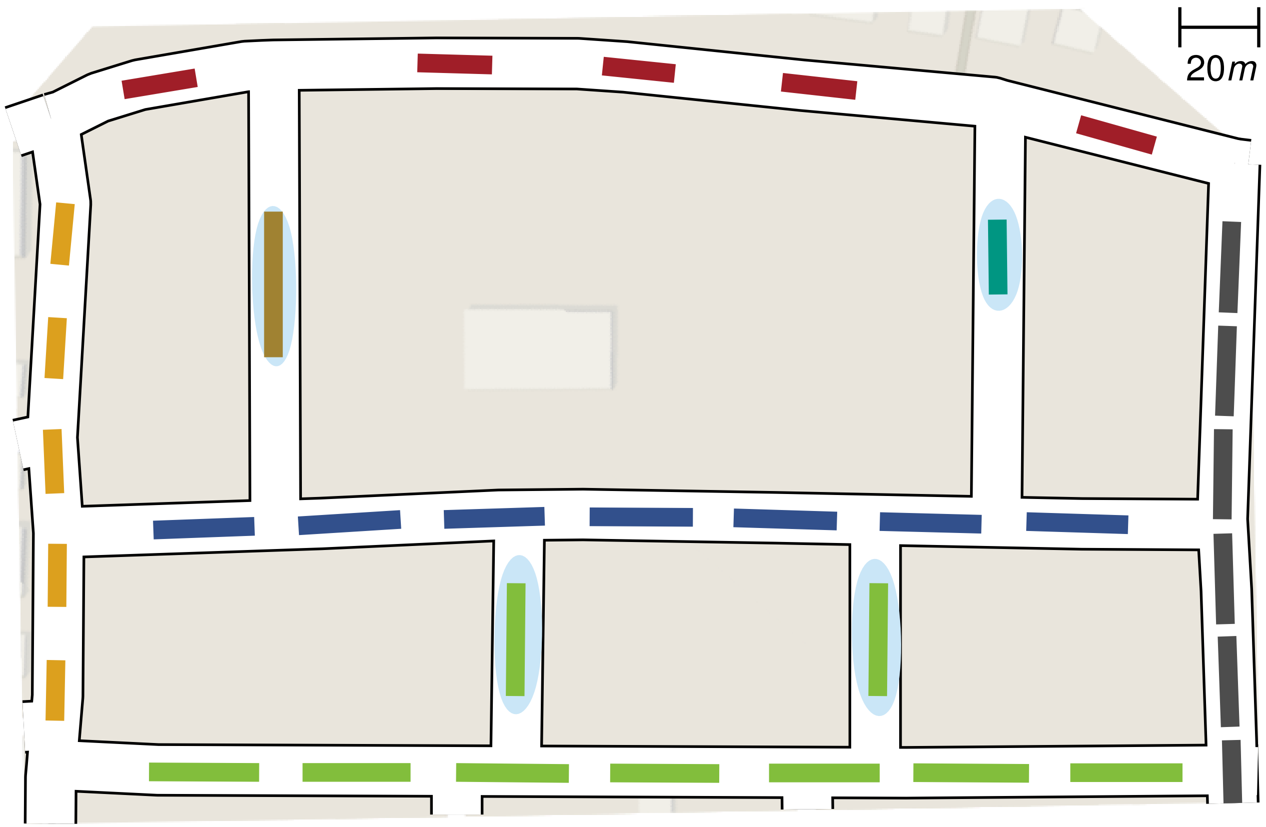
Labeling:
30 labels
5 roads are labeled
15 road sections are labeled.

Location: Karlsruhe Hermann-Vollmerstraße

road section = part of road between two junctions

Why considering road labeling?

Example: Google Maps



Location: Karlsruhe Hermann-Vollmerstraße

Map:
7 roads
19 road sections

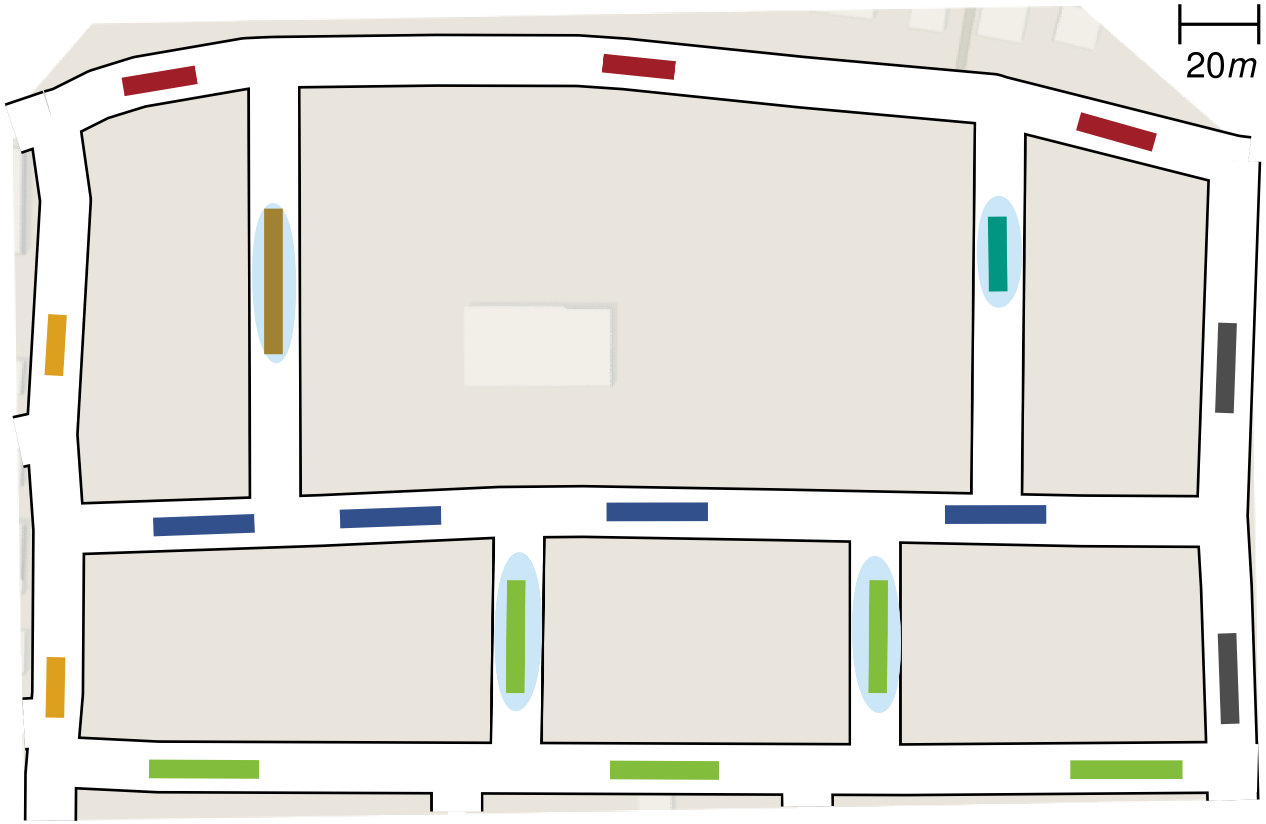
Labeling:
30 labels
5 roads are labeled
15 road sections are labeled.

Improvement:
4 labels added
7 roads are labeled
19 road sections are labeled.

road section = part of road between two junctions

Why considering road labeling?

Example: Google Maps



Location: Karlsruhe Hermann-Vollmerstraße

Map:
7 roads
19 road sections

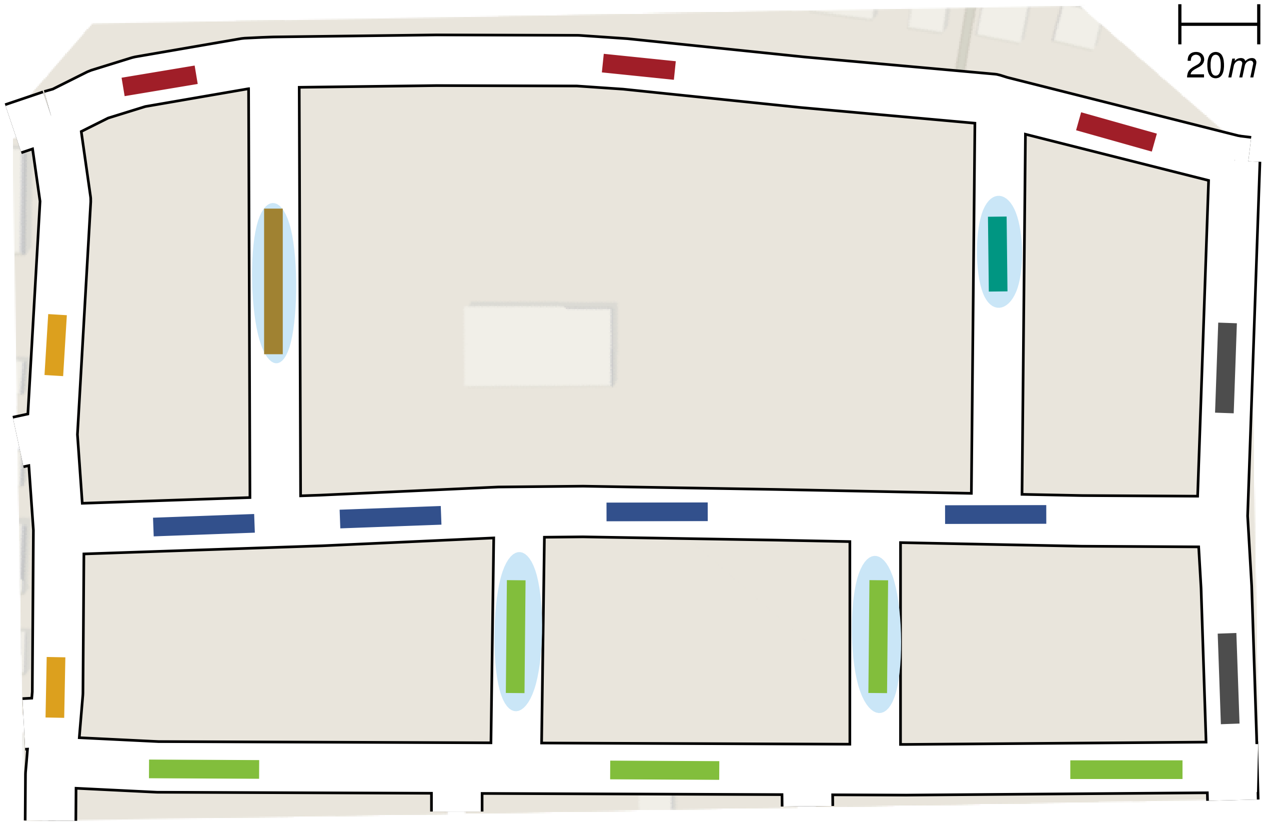
Labeling:
30 labels
5 roads are labeled
15 road sections are labeled.

Improvement:
4 labels added
7 roads are labeled
19 road sections are labeled.
16 labels less

road section = part of road between two junctions

Why considering road labeling?

Example: Google Maps



Location: Karlsruhe Hermann-Vollmerstraße

Map:
7 roads
19 road sections

Labeling:
30 labels
5 roads are labeled
15 road sections are labeled.

Improvement:
4 labels added

7 roads are labeled
19 road sections are labeled.
16 labels less

→ Conclusion: many labels, but labeling can be improved.

road section = part of road between two junctions

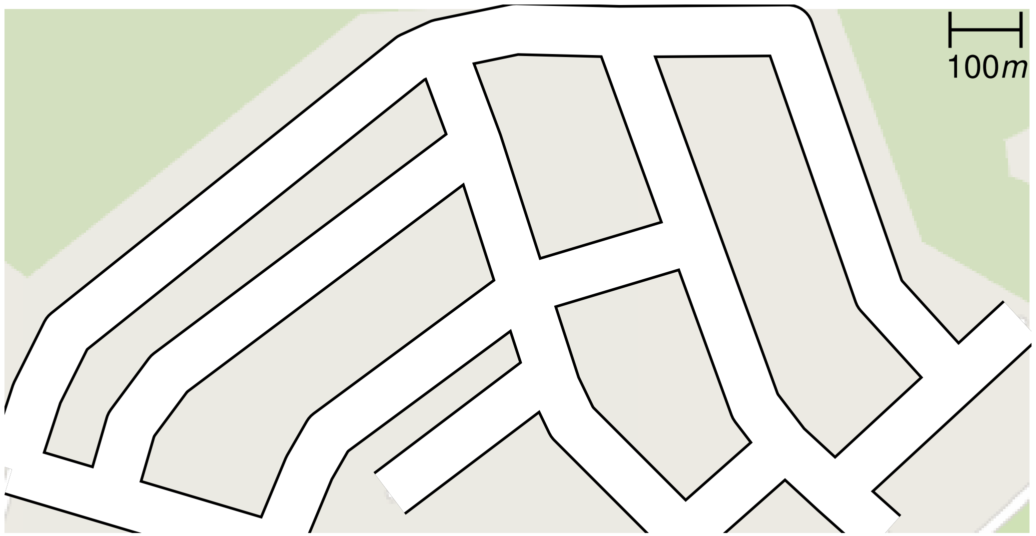
Why considering road labeling?

Example: Bing Maps

Location:
Karlsruhe Kirchbuel

Why considering road labeling?

Example: Bing Maps



Location:
Karlsruhe Kirchbuel

Map:
8 roads
18 road sections

Why considering road labeling?

Example: Bing Maps

Location:
Karlsruhe Kirchbuel

Map:
8 roads
18 road sections

Labeling:
8 labels
7 roads are labeled
12 road sections are labeled.

Why considering road labeling?

Example: Bing Maps

Location:
Karlsruhe Kirchbuel

Map:
8 roads
18 road sections

Labeling:
8 labels
7 roads are labeled
12 road sections are labeled.

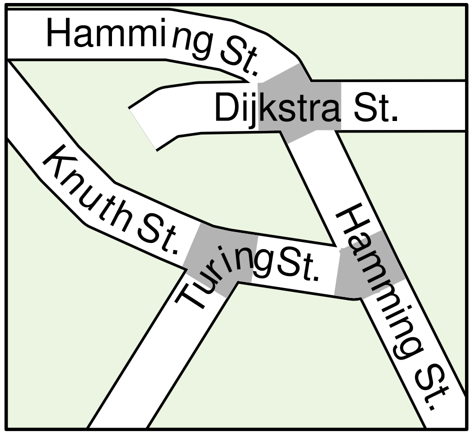
Improvement:
4 labels are moved
2 labels are added

7 roads are labeled
17 road sections are labeled.

Related Work

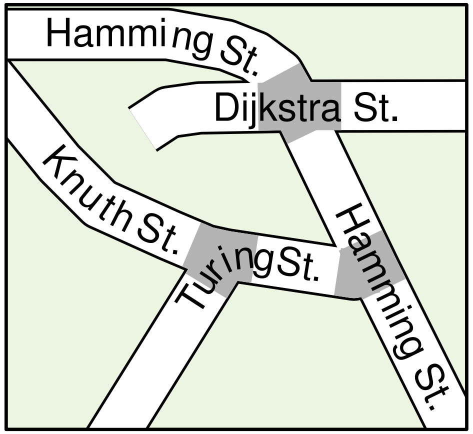
Criteria for road labeling [Chirié, 2000]

1) Labels placed inside and parallel to road shapes.
2) Every road section between two junctions should be clearly identified.
3) No two road labels may intersect.



Related Work

Criteria for road labeling [Chirié, 2000]



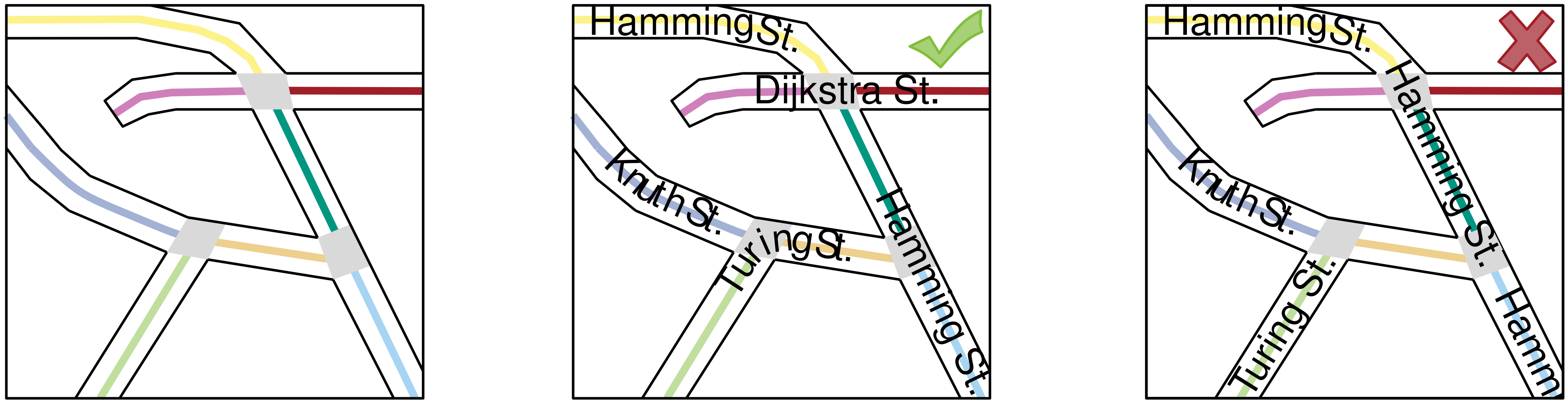
1) Labels placed inside and parallel to road shapes.
2) Every road section between two junctions should be clearly identified.
3) No two road labels may intersect.

Road labeling on grids

- *NP*-complete to decide whether for every road at least one label can be placed. [Seibert and Unger, 2000]
- Empirically efficient algorithm that finds such a labeling if possible [Neyer and Wagner, 2000]

Model for labeling road maps.

Basic Observations



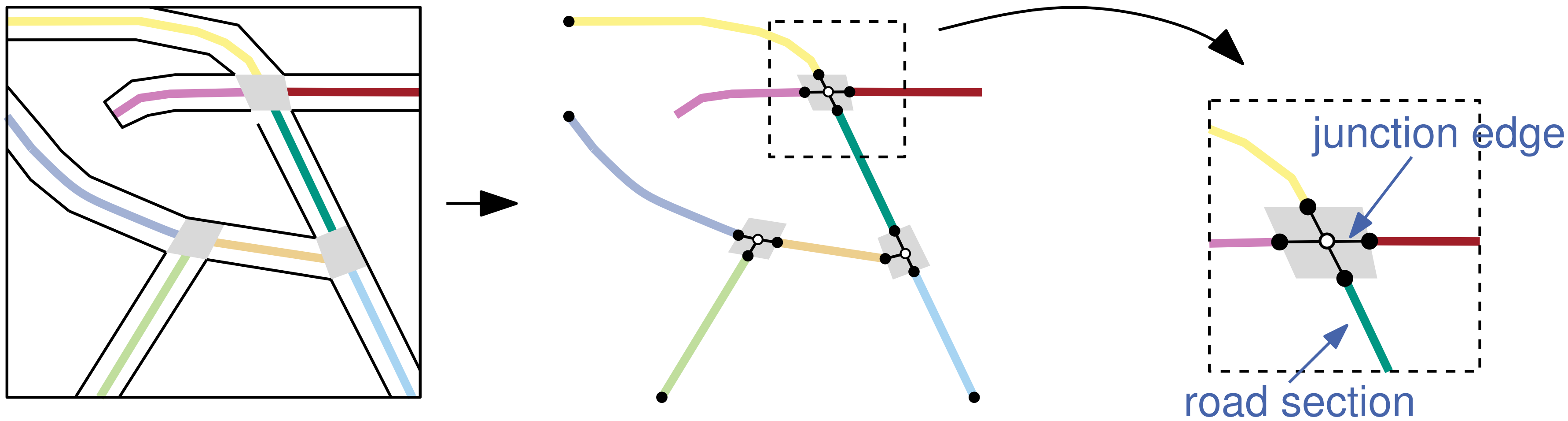
1) Roads can be decomposed into road sections.

→ Part of the road that lies in between two junctions.

2) Labels are part of the roads.
3) Labels of the same road have the same length.
4) Labels of different roads may only intersect on junction areas.
5) Maximizing the number of labels is not sufficient.

→ Maximize number of labeled road sections.

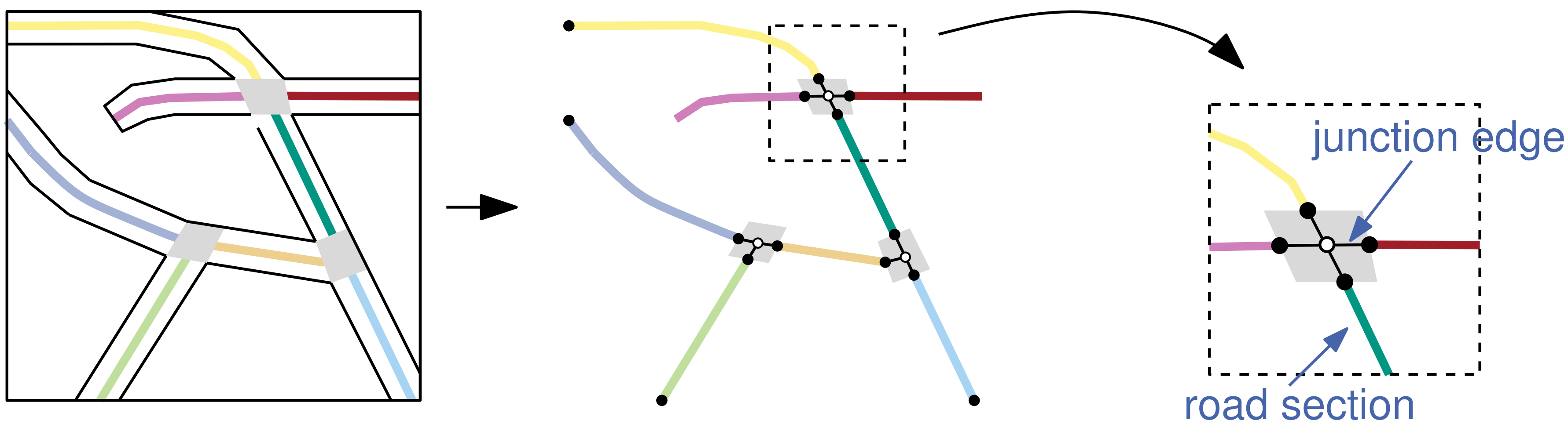
Model



Model road map as a planar embedded graph $G = (V, E)$.

- Each road section becomes an edge.
- Introduce *junction edges* to model junctions.
- Embedding of the edges is given by the skeleton of the road map.

Model

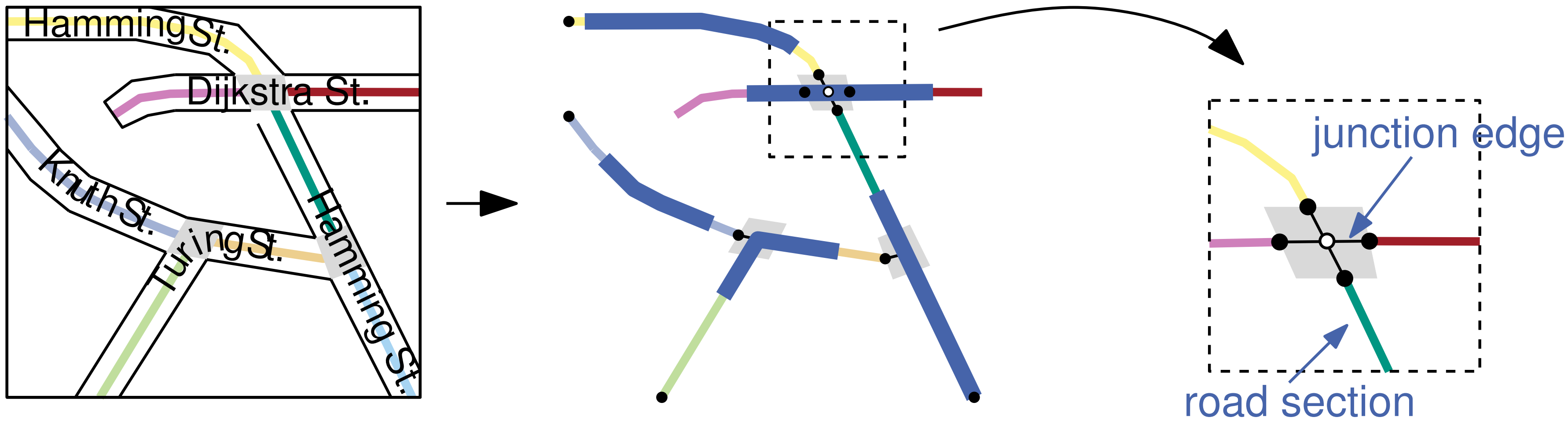


Model road map as a planar embedded graph $G = (V, E)$**.**

- Each road section becomes an edge.
- Introduce *junction edges* to model junctions.
- Embedding of the edges is given by the skeleton of the road map.

Road = connected subgraph of G such that all contained road sections have the same name.

Model

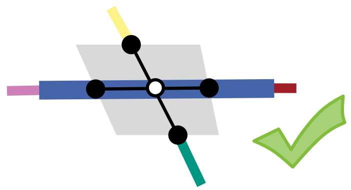


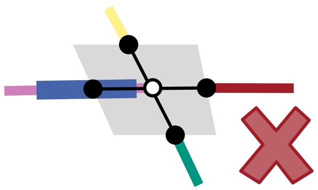
Model road map as a planar embedded graph $G = (V, E)$.

- Each road section becomes an edge.
- Introduce *junction edges* to model junctions.
- Embedding of the edges is given by the skeleton of the road map.

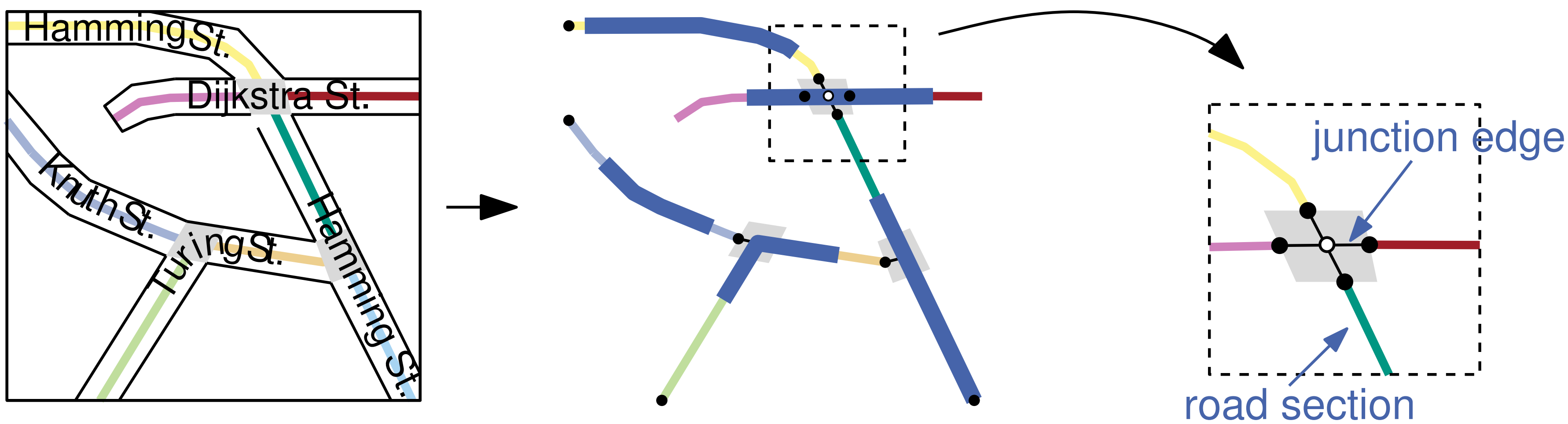
Labels are curves in the embedding.

- Labels must end on road sections, i.e., not on junction edges.
- Labels of same road have same length.





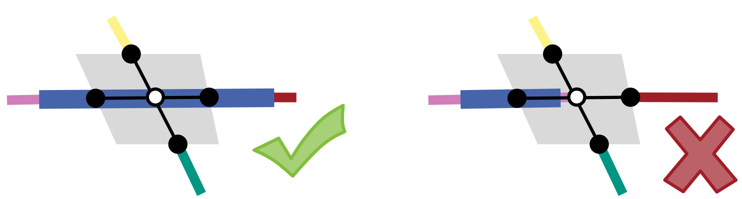
Model



Model road map as a planar embedded graph $G = (V, E)$.

- Each road section becomes an edge.
- Introduce *junction edges* to model junctions.
- Embedding of the edges is given by the skeleton of the road map.

Labels are curves in the embedding.



- Labels must end on road sections, i.e., not on junction edges.
- Labels of same road have same length.

Objective: Maximize number of labeled road sections.

Computational complexity of road labeling.

Complexity

Problem of maximizing number of labeled road sections is NP-hard.

Complexity

Problem of maximizing number of labeled road sections is NP-hard.

Reduction: *monotone planar 3-SAT*.

Given: 3SAT-Clauses $\mathcal{C}$ such that

- each clause either contains positive or negative literals, and
- variable-clause-graph G is planar.

Question: Is $\mathcal{C}$ satisfiable? (NP-hard)

Complexity

Problem of maximizing number of labeled road sections is NP-hard.

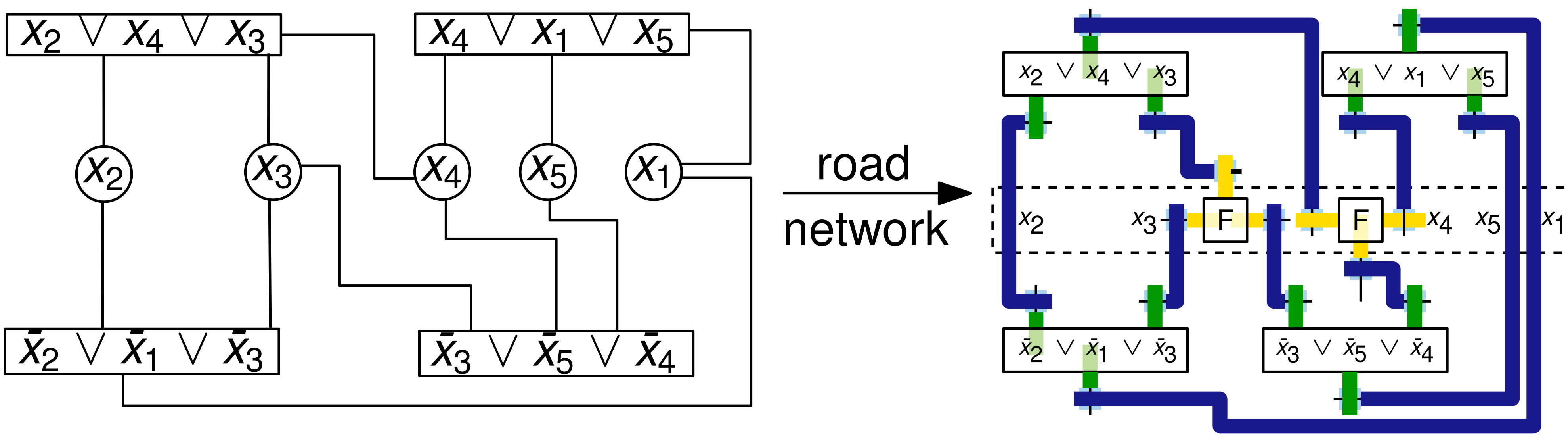
Reduction: *monotone planar 3-SAT*.

Given: 3SAT-Clauses $\mathcal{C}$ such that

- each clause either contains positive or negative literals, and
- variable-clause-graph G is planar.

Question: Is $\mathcal{C}$ satisfiable? (NP-hard)

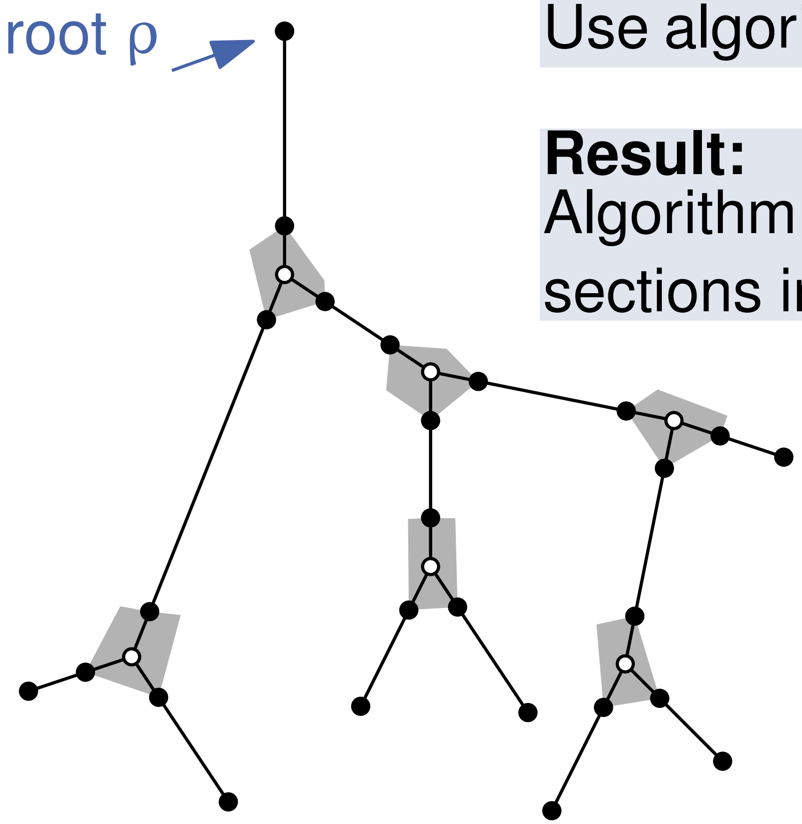
Example: $x_2 \vee x_4 \vee x_3 \quad x_4 \vee x_1 \vee x_5 \quad \bar{x}_2 \vee \bar{x}_1 \vee \bar{x}_3 \quad \bar{x}_3 \vee \bar{x}_5 \vee \bar{x}_4$



Algorithms for labeling road maps.

Trees

Assumption: Road map is tree $T = (V, E)$.



Motivation:
Use algorithms for trees as basis for heuristics.

Result:
Algorithm that labels maximum number of road sections in $O(n^3)$ time.

Motivation for Trees

Preprocessing Step:

Remove or cut any edge from road map that can be labeled trivially without loosing the optimal labeling.

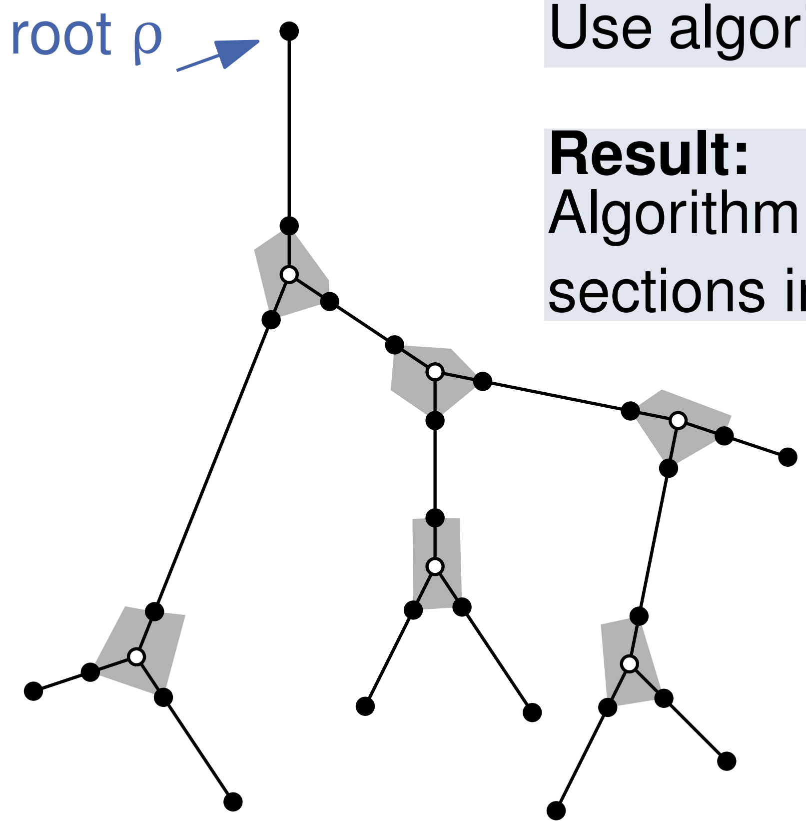
For example: Long road sections.

$\longrightarrow$ road map decomposes into subgraphs.

Number of	subgraphs in decomposition			
	trees	1 cycle	$\geq$ 2 cycles	$\sum$
Paris	20604 89.9%	1742 7.6%	583 2.5%	22929 100%
London	20538 94.1%	1012 4.6%	275 1.3%	21825 100%
Los Angeles	47131 97.7%	767 1.6%	350 0.7%	48248 100%

Trees

Assumption: Road map $G = (V, E)$ is tree.

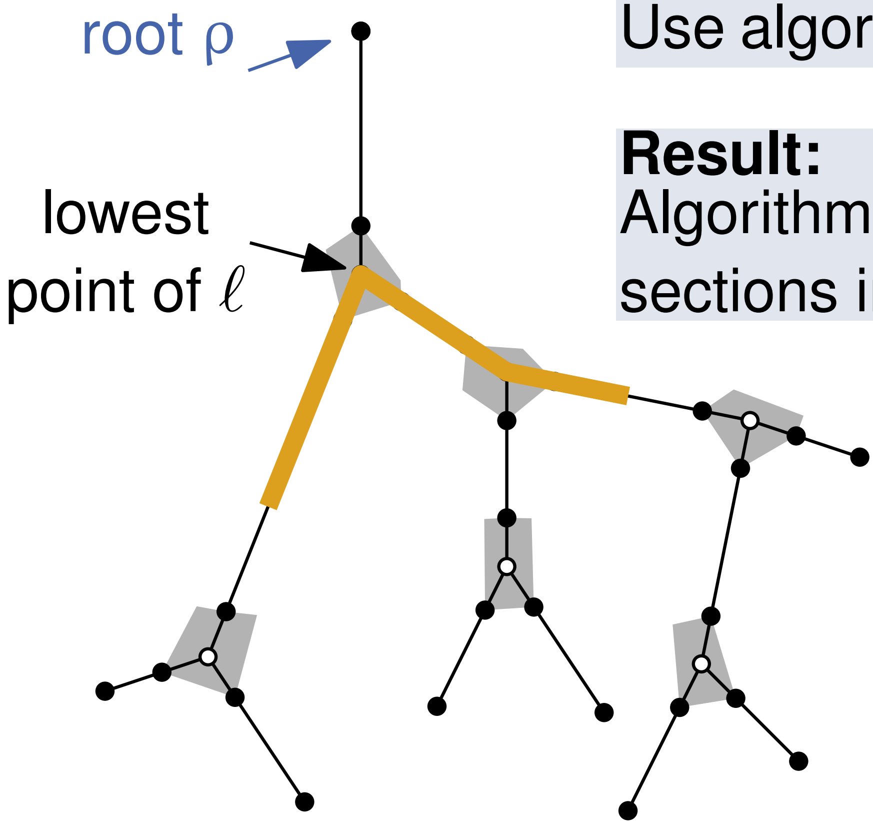


Motivation:
Use algorithms for trees as basis for heuristics.

Result:
Algorithm that labels maximum number of road sections in $O(n^3)$ time.

Trees

Assumption: Road map $G = (V, E)$ is tree.

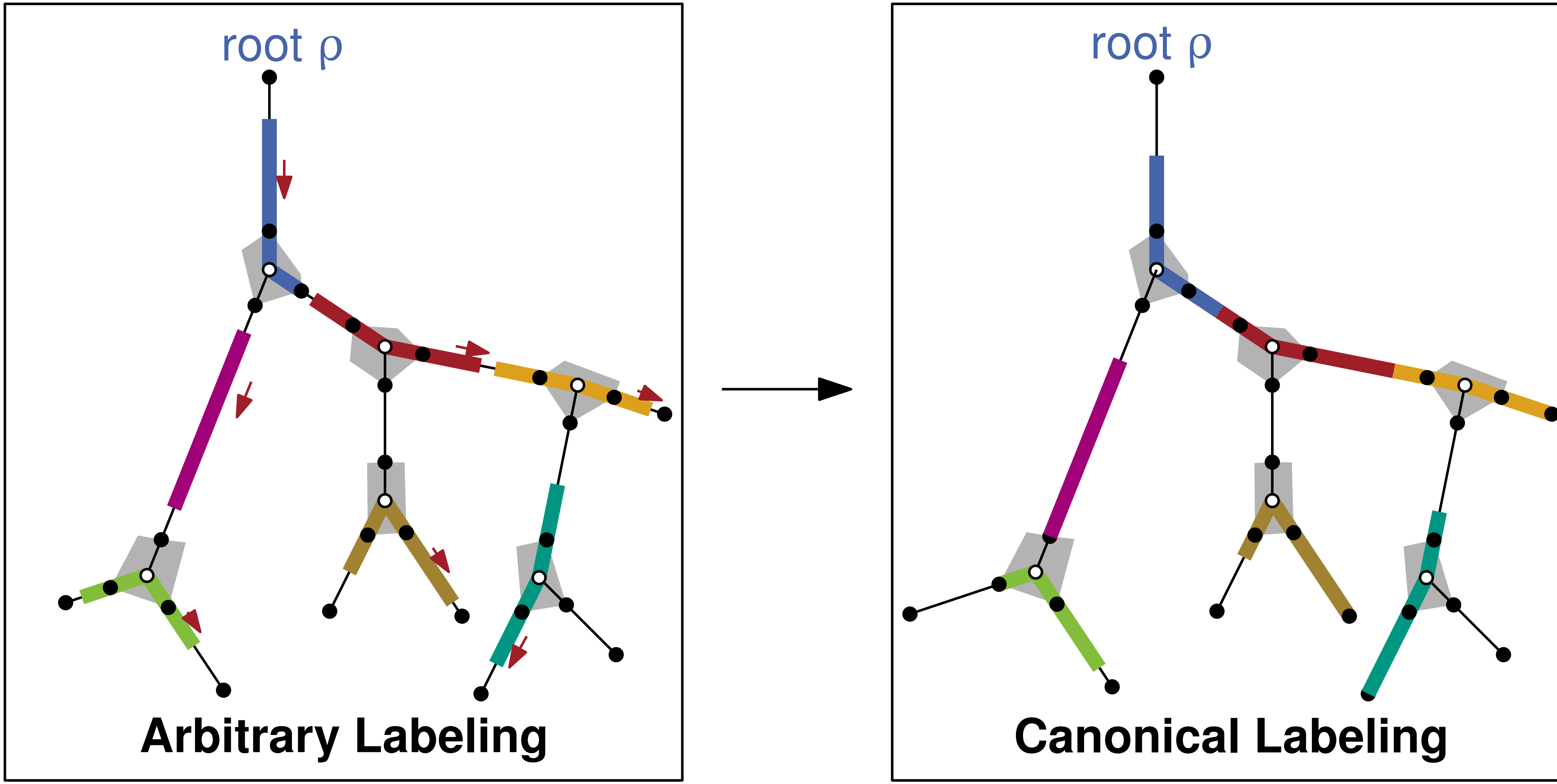


Motivation:
Use algorithms for trees as basis for heuristics.

Result:
Algorithm that labels maximum number of road sections in $O(n^3)$ time.

Notation:
Point of label ℓ with smallest distance to ρ in T is called lowest point of ℓ.

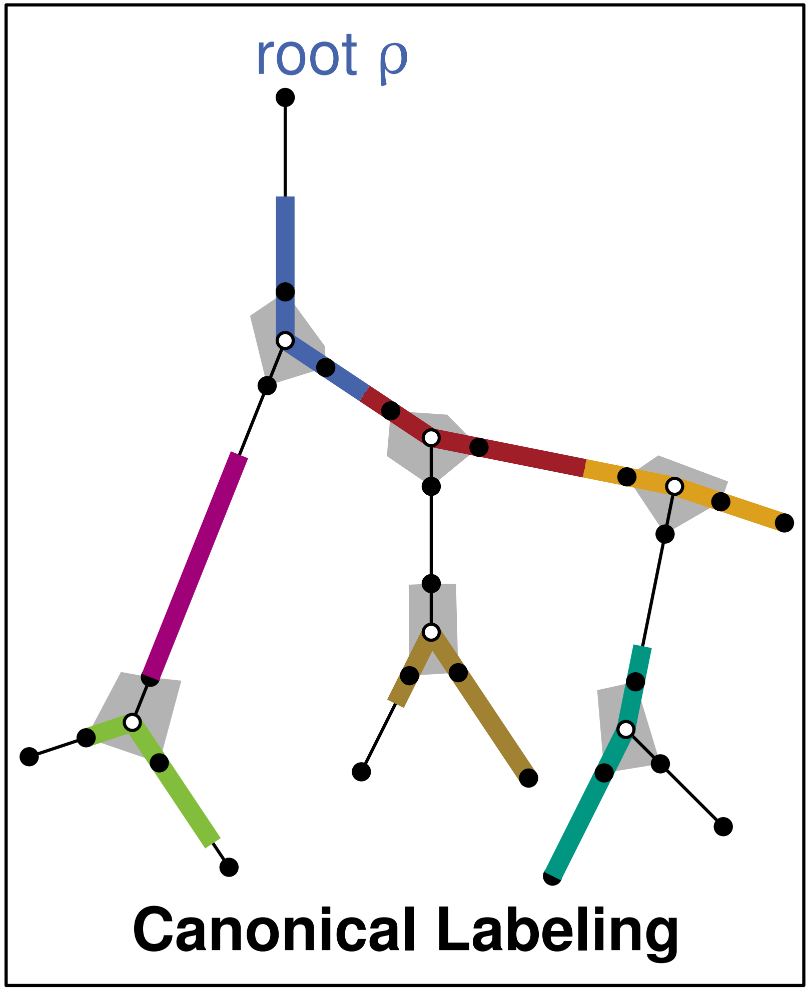
Canonical Labeling



Push labels towards leaves without changing the labeled road sections.

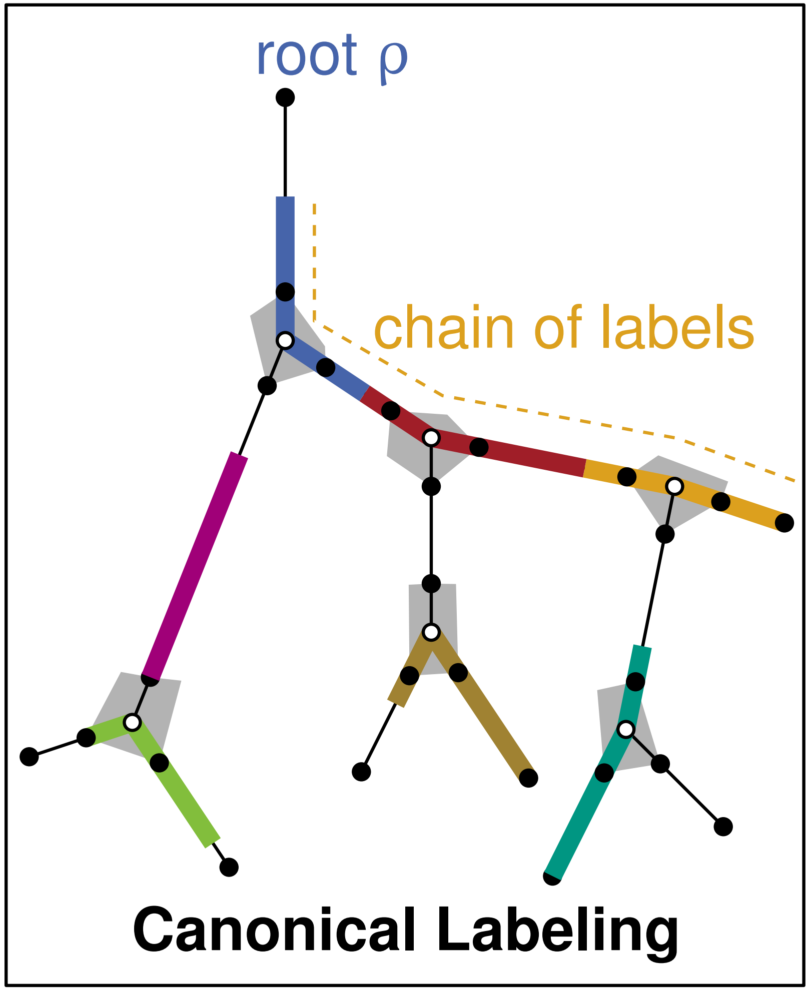
→ Every labeling can be transformed into canonical labeling.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

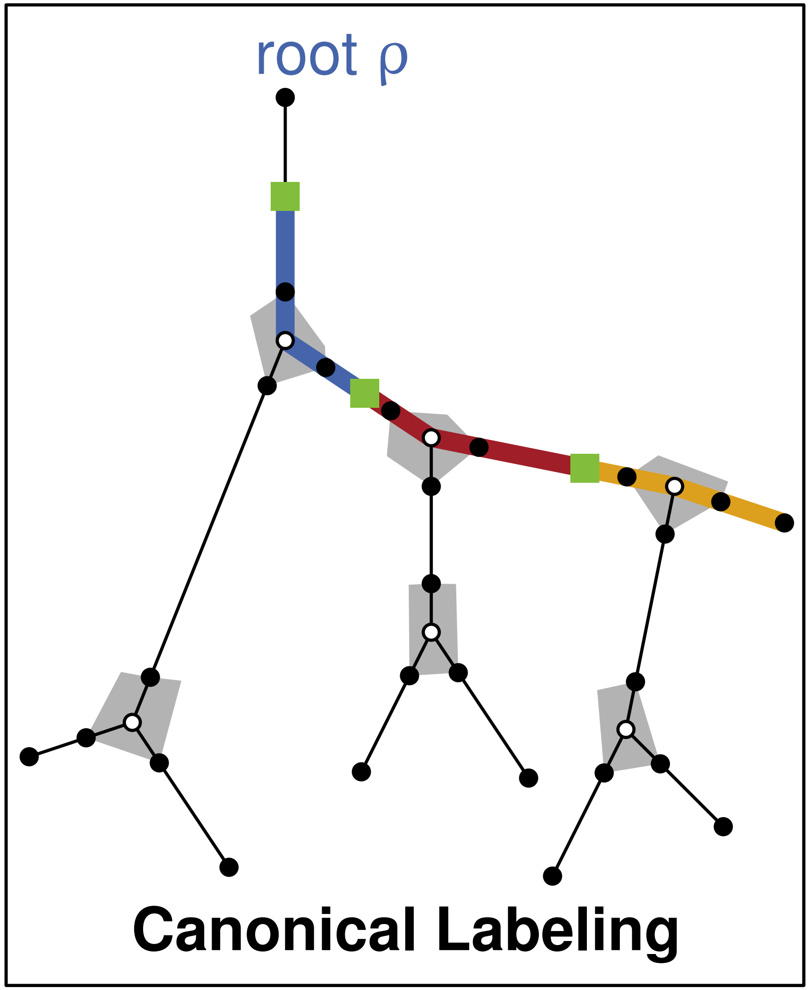
Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

(2) Labels form tightly packed chains.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

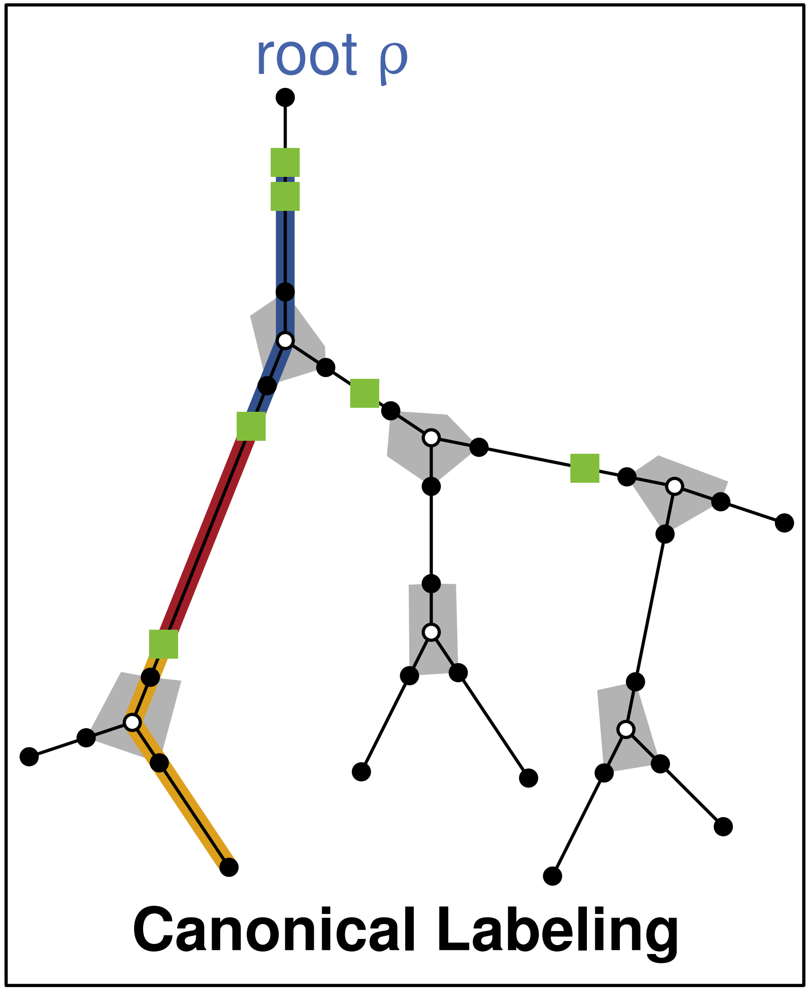
(2) Labels form tightly packed chains.

Idea: Construct all potential start points of labels in a canonical labeling.

→ Construct all possible chains.

→ Each chain induces subdivision nodes.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

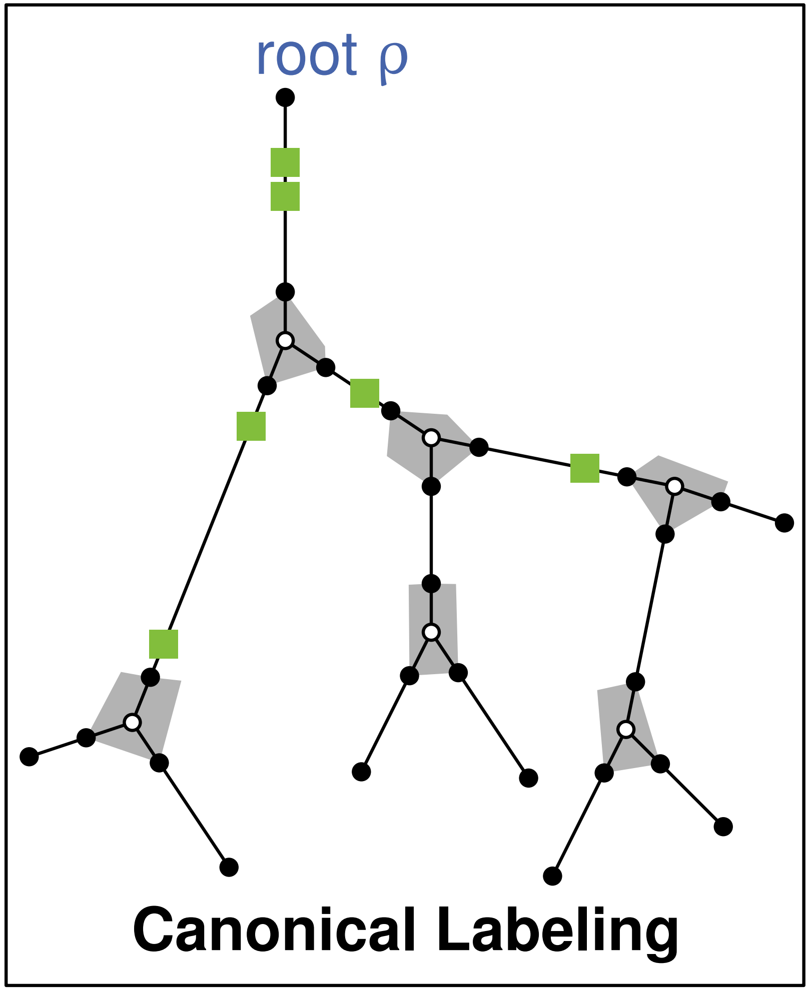
(2) Labels form tightly packed chains.

Idea: Construct all potential start points of labels in a canonical labeling.

→ Construct all possible chains.

→ Each chain induces subdivision nodes.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

(2) Labels form tightly packed chains.

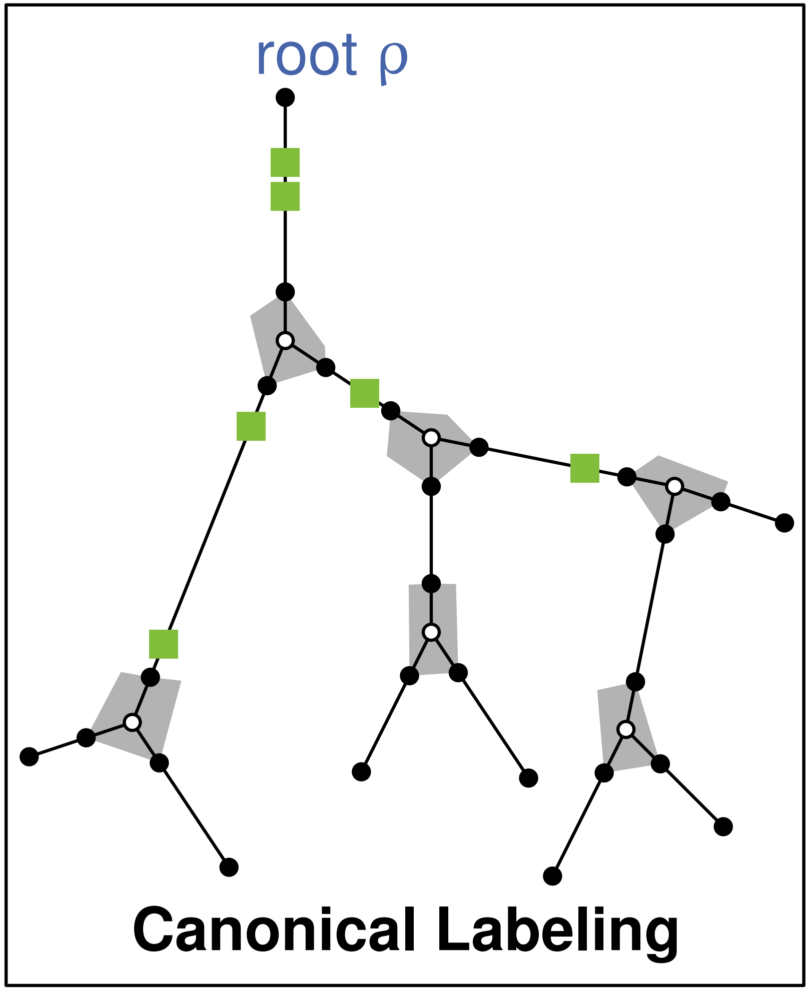
Idea: Construct all potential start points of labels in a canonical labeling.

→ Construct all possible chains.

→ Each chain induces subdivision nodes.

→ Subdivision tree T'=
Original tree T +
any possible start point of label in canonical labeling.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

(2) Labels form tightly packed chains.

Idea: Construct all potential start points of labels in a canonical labeling.

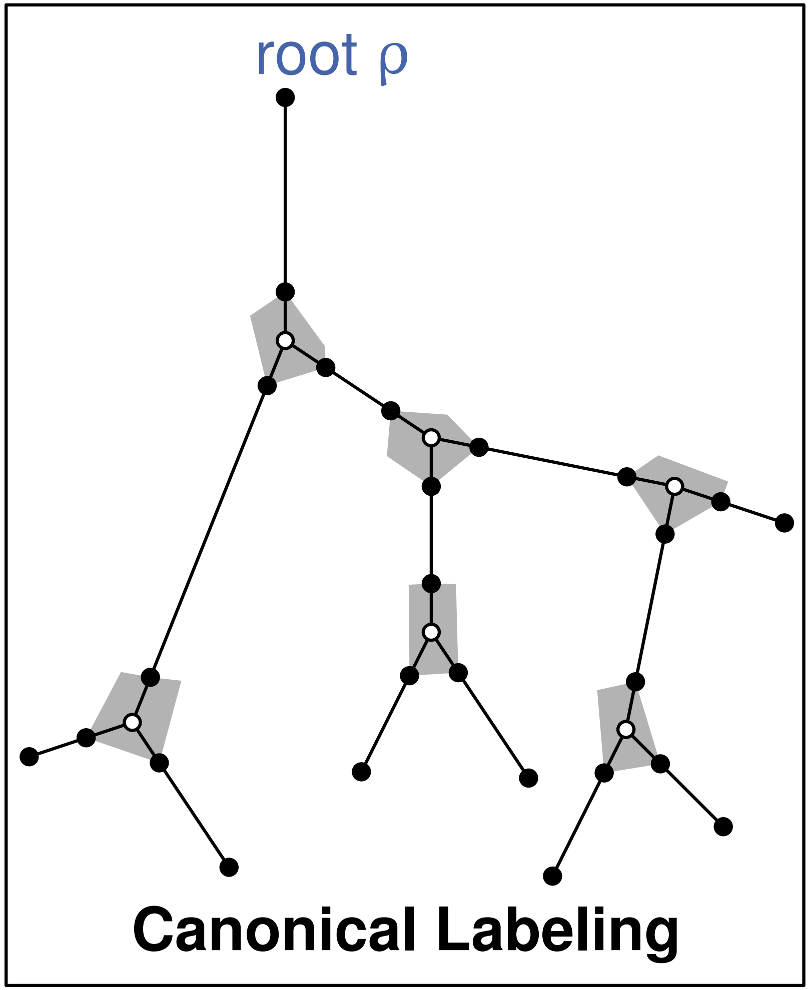
→ Construct all possible chains.

→ Each chain induces subdivision nodes.

→ Subdivision tree T'=
Original tree T +
any possible start point of label in canonical labeling.

T' has $O(n^2)$ vertices.

Properties of Canonical Labeling



(1) Each label starts either at vertex or at a previous label.

(2) Labels form tightly packed chains.

Idea: Construct all potential start points of labels in a canonical labeling.

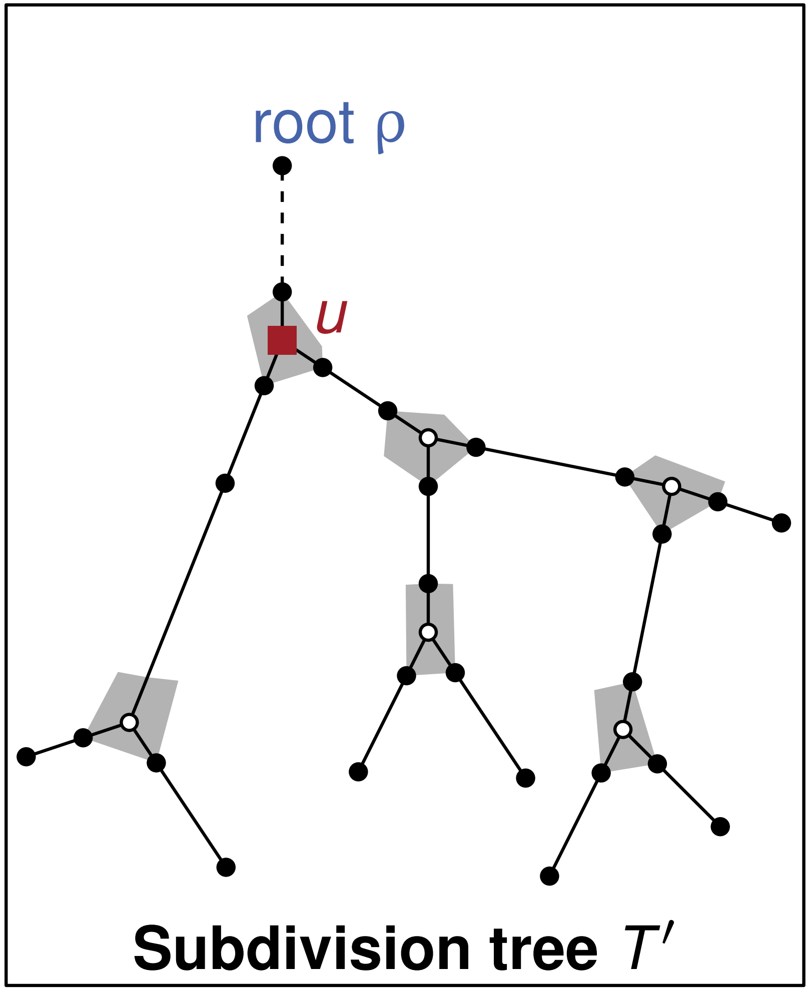
→ Construct all possible chains.

→ Each chain induces subdivision nodes.

→ Subdivision tree T'=
Original tree T +
any possible start point of label in canonical labeling.

T' has $O(n^2)$ vertices.

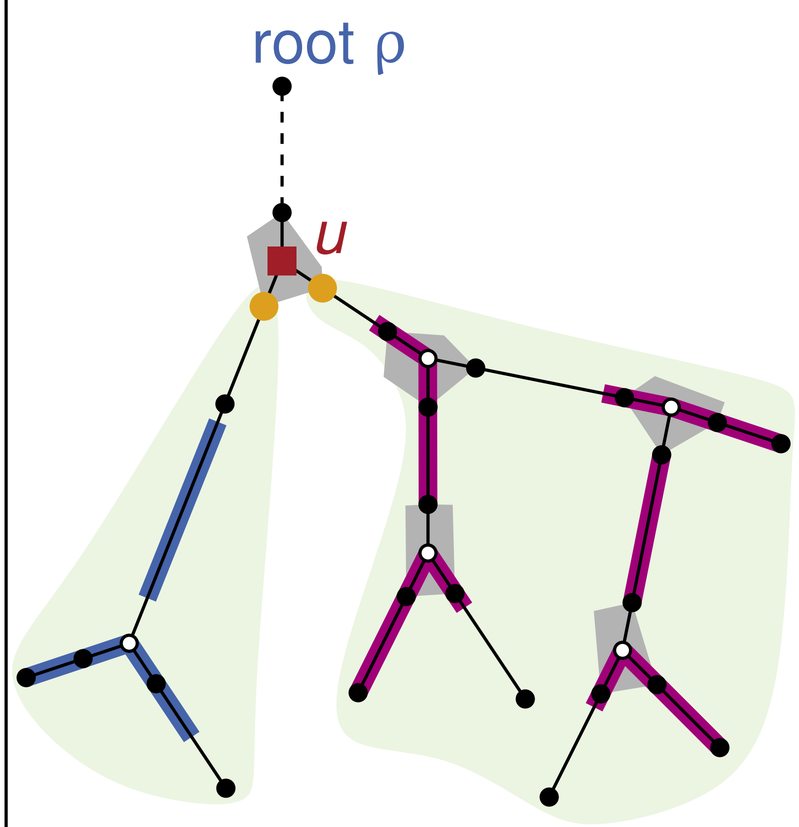
Optimal Labeling



Consider vertex u of subdiv. tree T'.

$\mathcal{L}_u$= optimal labeling of tree T'_u rooted at u.

Optimal Labeling



Subdivision tree T'

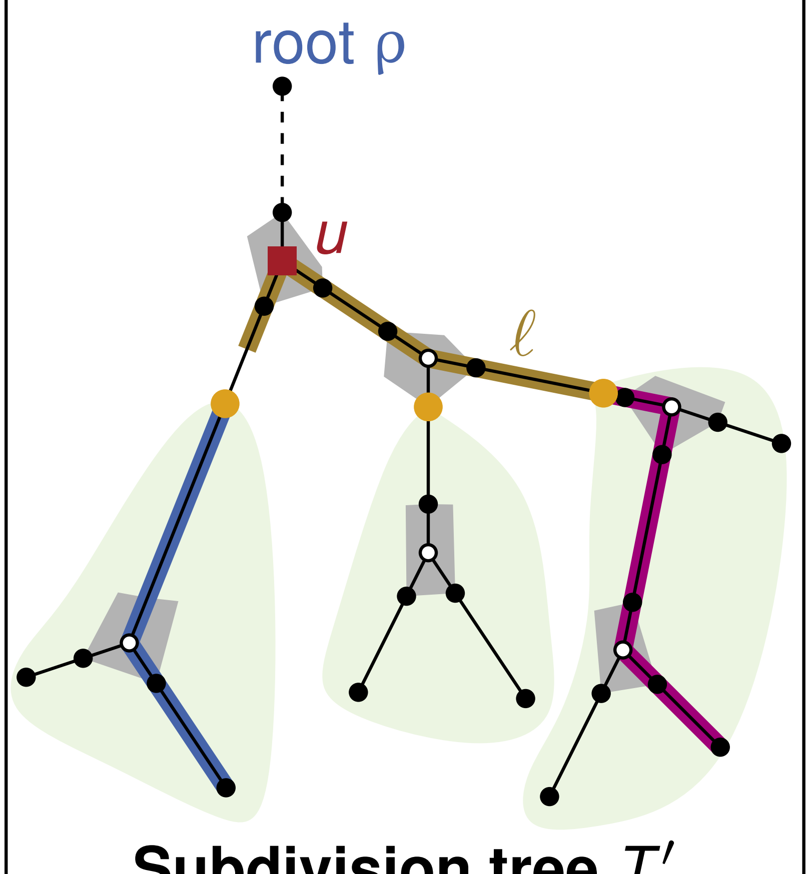
Consider vertex u of subdiv. tree T'.

$\mathcal{L}_u$= optimal labeling of tree T'_u rooted at u.

Case 1: $\nexists$ label $\ell \in \mathcal{L}_u$ with lowest point u.

$$\mathcal{L}_u = \bigcup \{\mathcal{L}_v \mid v \text{ is child of } u.\}$$

Optimal Labeling



Consider vertex u of subdiv. tree T'.

$\mathcal{L}_u$= optimal labeling of tree T'_u rooted at u.

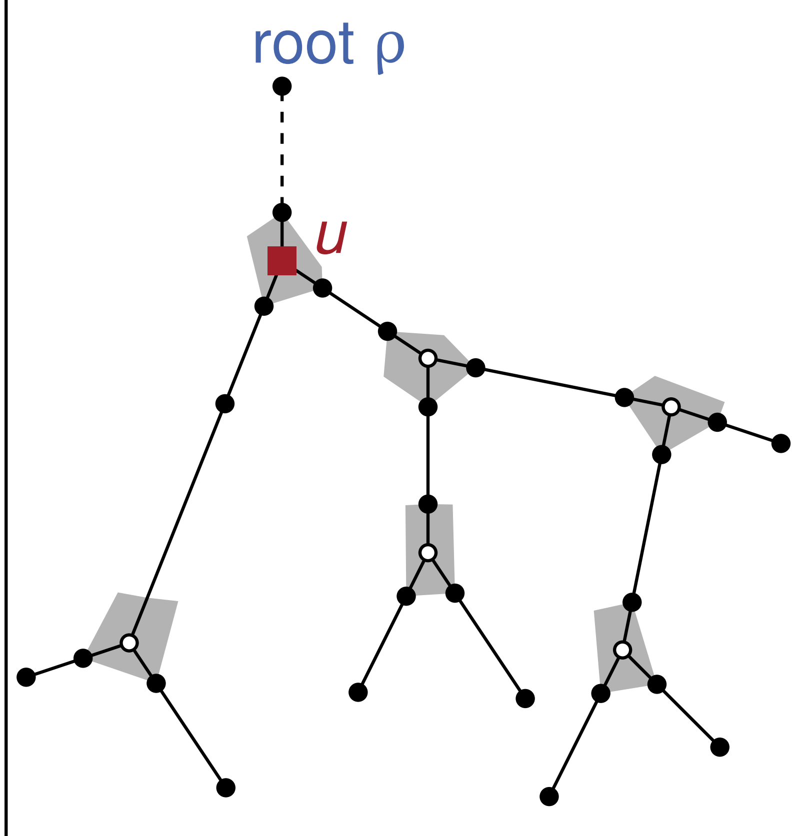
Case 1: $\nexists$ label $\ell \in \mathcal{L}_u$ with lowest point u.

$$\mathcal{L}_u = \bigcup \{\mathcal{L}_v \mid v \text{ is child of } u.\}$$

Case 2: $\exists$ label $\ell \in \mathcal{L}_u$ with lowest point u.

$$\mathcal{L}_u = \bigcup \{\mathcal{L}_v \mid v \text{ is child of } \ell.\} \cup \{\ell\}$$

Optimal Labeling



Consider vertex u of subdiv. tree T'.

$\mathcal{L}_u$= optimal labeling of tree T'_u rooted at u.

Case 1: $\nexists$ label $\ell \in \mathcal{L}_u$ with lowest point u.

$$\mathcal{L}_u = \bigcup \{\mathcal{L}_v \mid v \text{ is child of } u.\}$$

Case 2: $\exists$ label $\ell \in \mathcal{L}_u$ with lowest point u.

$$\mathcal{L}_u = \bigcup \{\mathcal{L}_v \mid v \text{ is child of } \ell.\} \cup \{\ell\}$$

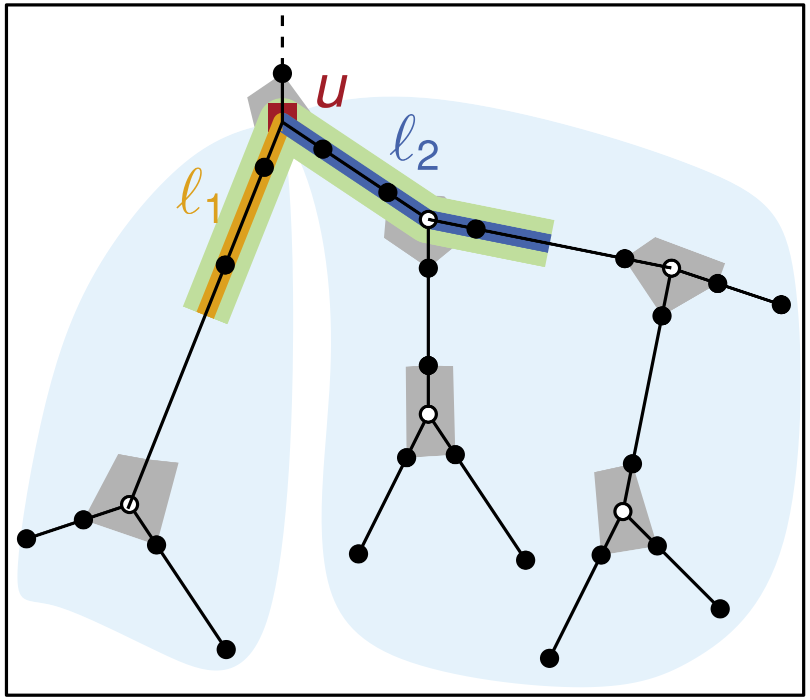
→ consider any label with lowest point u

→ dynamic programming

Bottom-up approach to obtain optimal labeling $\mathcal{L}_\rho$ of T

→ $O(n^5)$ time and $O(n^2)$ space.

Acceleration (Sketch)



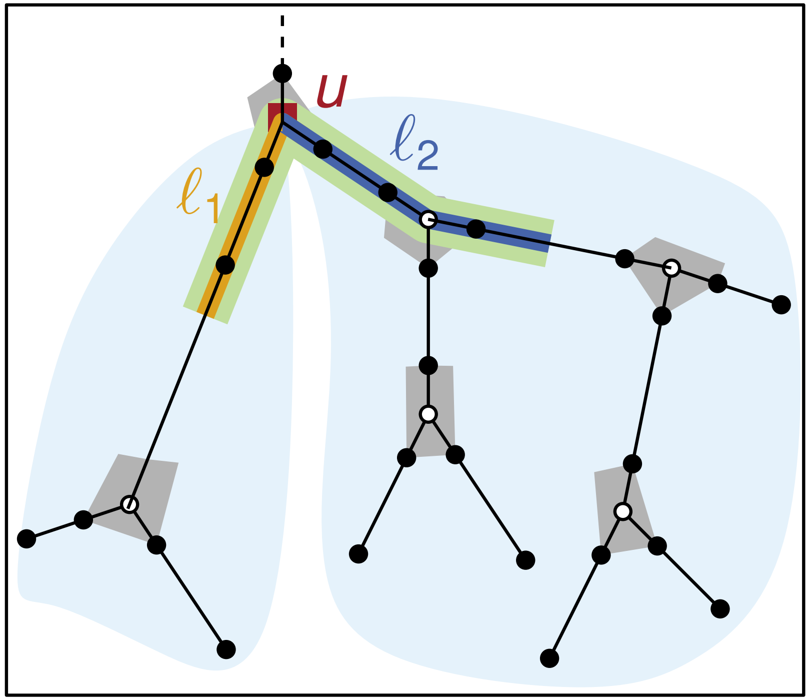
Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

$\longrightarrow$ extend into different sub-trees.

$\longrightarrow$ $\text{length}(\ell_1) + \text{length}(\ell_2) = \text{length}(\ell)$

Acceleration (Sketch)



Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

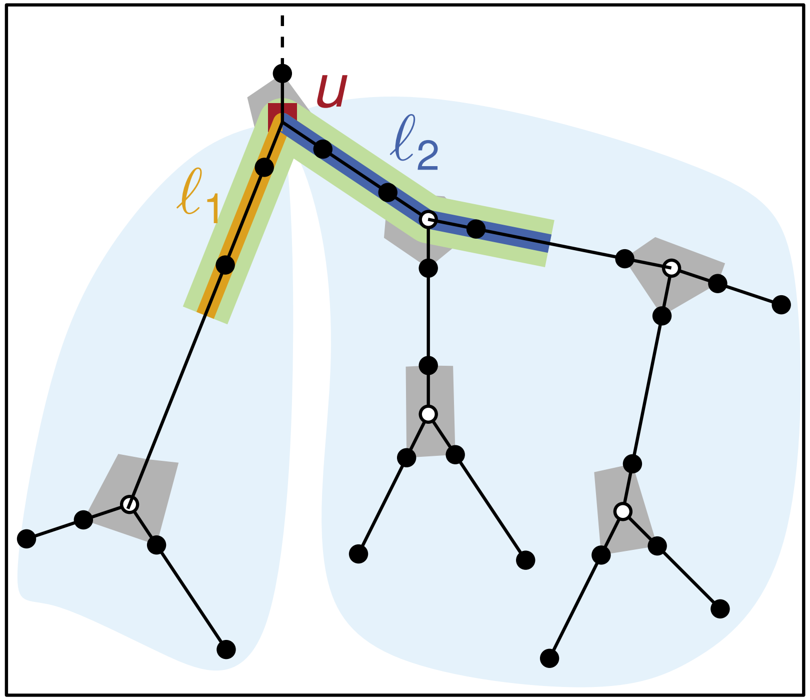
→ extend into different sub-trees.

→ $\text{length}(\ell_1) + \text{length}(\ell_2) = \text{length}(\ell)$

Let $v_1, \ldots, v_k$ denote the children of u.

B_i is tree $T'_{v_i} + \{u, v_i\}$

Acceleration (Sketch)



Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

$\longrightarrow$ extend into different sub-trees.

$\longrightarrow$ length(ℓ_1) + length(ℓ_2) = length(ℓ)

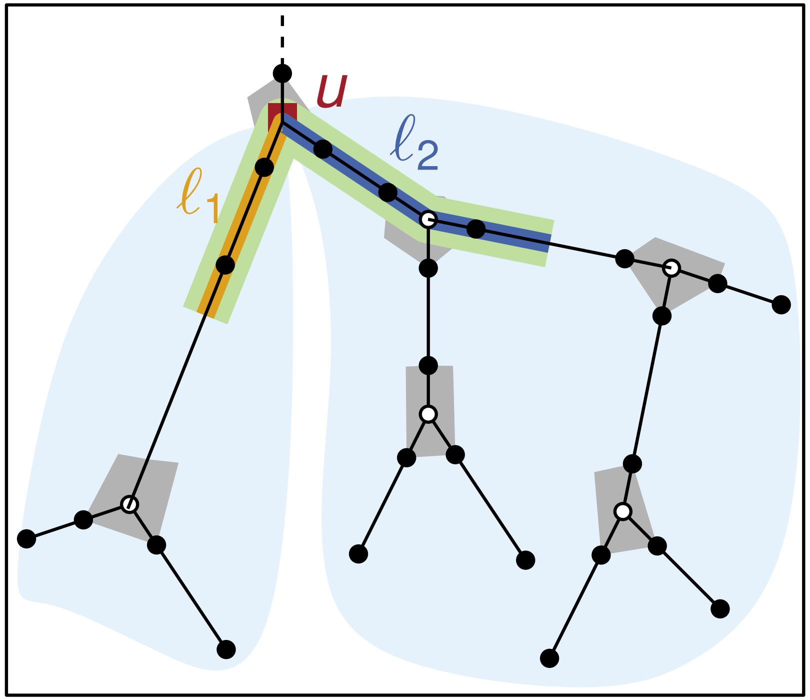
Let $v_1, \dots, v_k$ denote the children of u.

B_i is tree $T'_{v_i} + \{u, v_i\}$

Let $\mathcal{L}_{ij}^d$ be optimal labeling of T'_u such that

- $\mathcal{L}_{ij}^d$ contains label ℓ with lowest point u
- ℓ extends into B_i by length d
- ℓ extends into B_j

Acceleration (Sketch)



Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

$\longrightarrow$ extend into different sub-trees.

$\longrightarrow$ length(ℓ_1) + length(ℓ_2) = length(ℓ)

Let $v_1, \ldots, v_k$ denote the children of u.

B_i is tree $T'_{v_i} + \{u, v_i\}$

Let $\mathcal{L}^d_{ij}$ be optimal labeling of T'_u such that

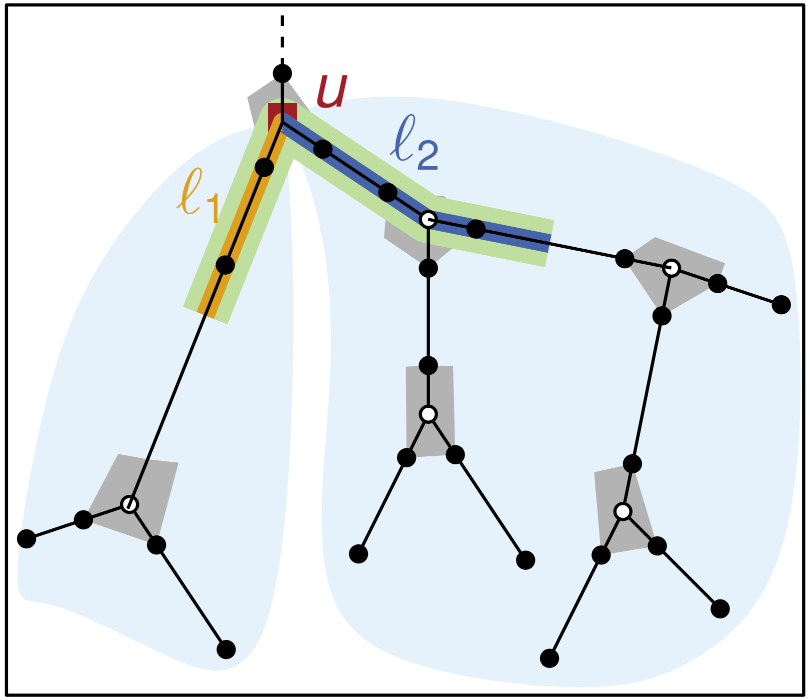
- $\mathcal{L}^d_{ij}$ contains label ℓ with lowest point u
- ℓ extends into B_i by length d
- ℓ extends into B_j

For each pair B_i, B_j build datastructure D_{ij}

Query: length $d \in \mathbb{R}^+$

Output: Value of $\mathcal{L}^d_{ij}$

Acceleration (Sketch)



Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

⟶ extend into different sub-trees.

⟶ length(ℓ_1) + length(ℓ_2) = length(ℓ)

Let $v_1, \ldots, v_k$ denote the children of u.

B_i is tree $T'_{v_i} + \{u, v_i\}$

Let $\mathcal{L}^d_{ij}$ be optimal labeling of T'_u such that

- $\mathcal{L}^d_{ij}$ contains label ℓ with lowest point u
- ℓ extends into B_i by length d
- ℓ extends into B_j

For each pair B_i, B_j build datastructure D_{ij}

Query: length $d \in \mathbb{R}^+$

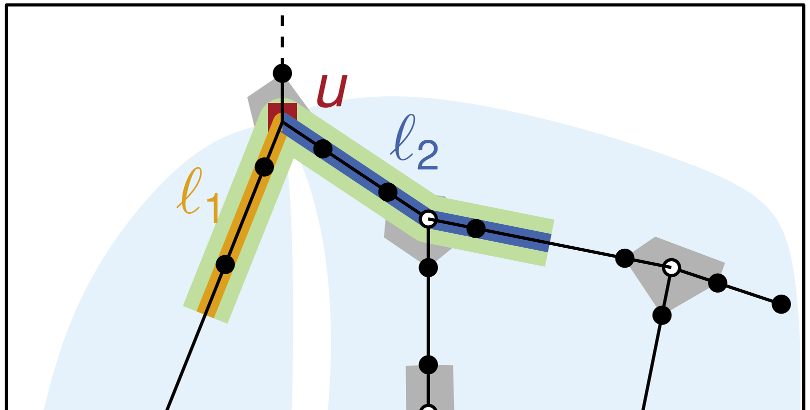
Output: Value of $\mathcal{L}^d_{ij}$

Each vertex in B_i and B_j induces a query.

⟶ Take the best result.

⟶ Paper: $O(n)$ queries in $O(n)$ time per vertex is sufficient.

Acceleration (Sketch)



Consider label ℓ with lowest vertex u.

Observation: Lowest point splits label ℓ into two sub-labels ℓ_1 and ℓ_2.

⟶ extend into different sub-trees.

⟶ length(ℓ_1) + length(ℓ_2) = length(ℓ)

Theorem:
If the road map is a tree, the maximum number of road sections can be labeled in $O(n^3)$ time and $O(n)$ space.

- D_{ij} contains label ℓ with lowest point u
- ℓ extends into B_i by length d
- ℓ extends into B_j

For each pair B_i, B_j build datastructure D_{ij}

Query: length $d \in \mathbb{R}^+$

Output: Value of $\mathcal{L}_{ij}^d$

Each vertex in B_i and B_j induces a query.

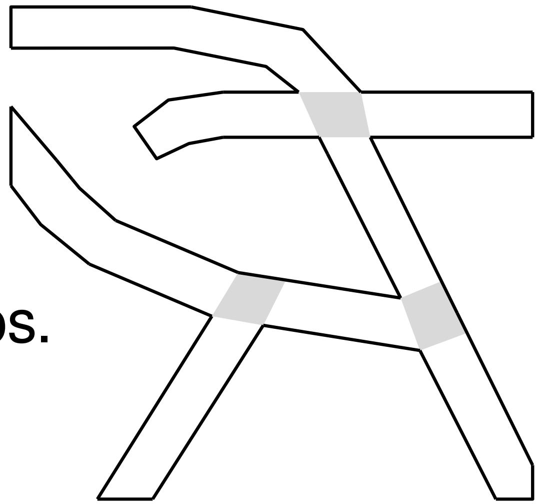
⟶ Take the best result.

⟶ Paper: $O(n)$ queries in $O(n)$ time per vertex is sufficient.

Conclusion

Results:

- General model for placing labels in road maps.
- Complexity result: NP-hardness.
- Polynomial time algorithm for tree-shaped maps.
- Initial experiments.



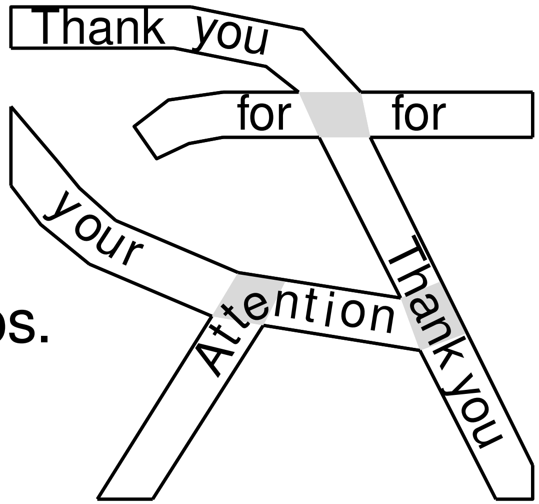
Future work:

- Heuristics and approximation algorithms.
- Implementation and evaluation of algorithms.
- Solving problem on super classes of trees.

Conclusion

Results:

- General model for placing labels in road maps.
- Complexity result: NP-hardness.
- Polynomial time algorithm for tree-shaped maps.
- Initial experiments.



Future work:

- Heuristics and approximation algorithms.
- Implementation and evaluation of algorithms.
- Solving problem on super classes of trees.