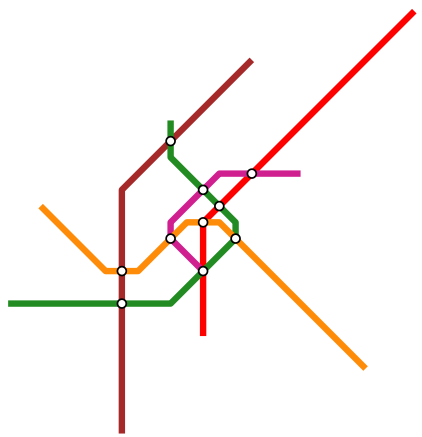
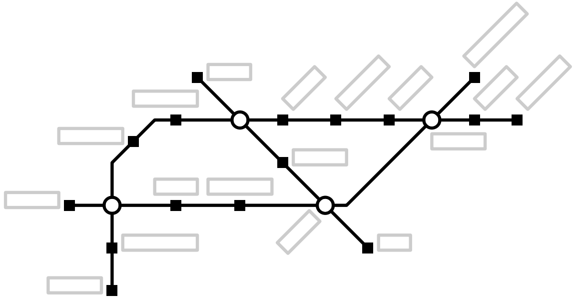
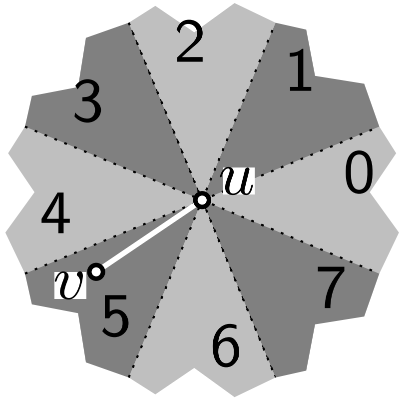
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Visualization of Graphs

Lecture 12:
Octilinear Graph Drawing
Metro Map Layout



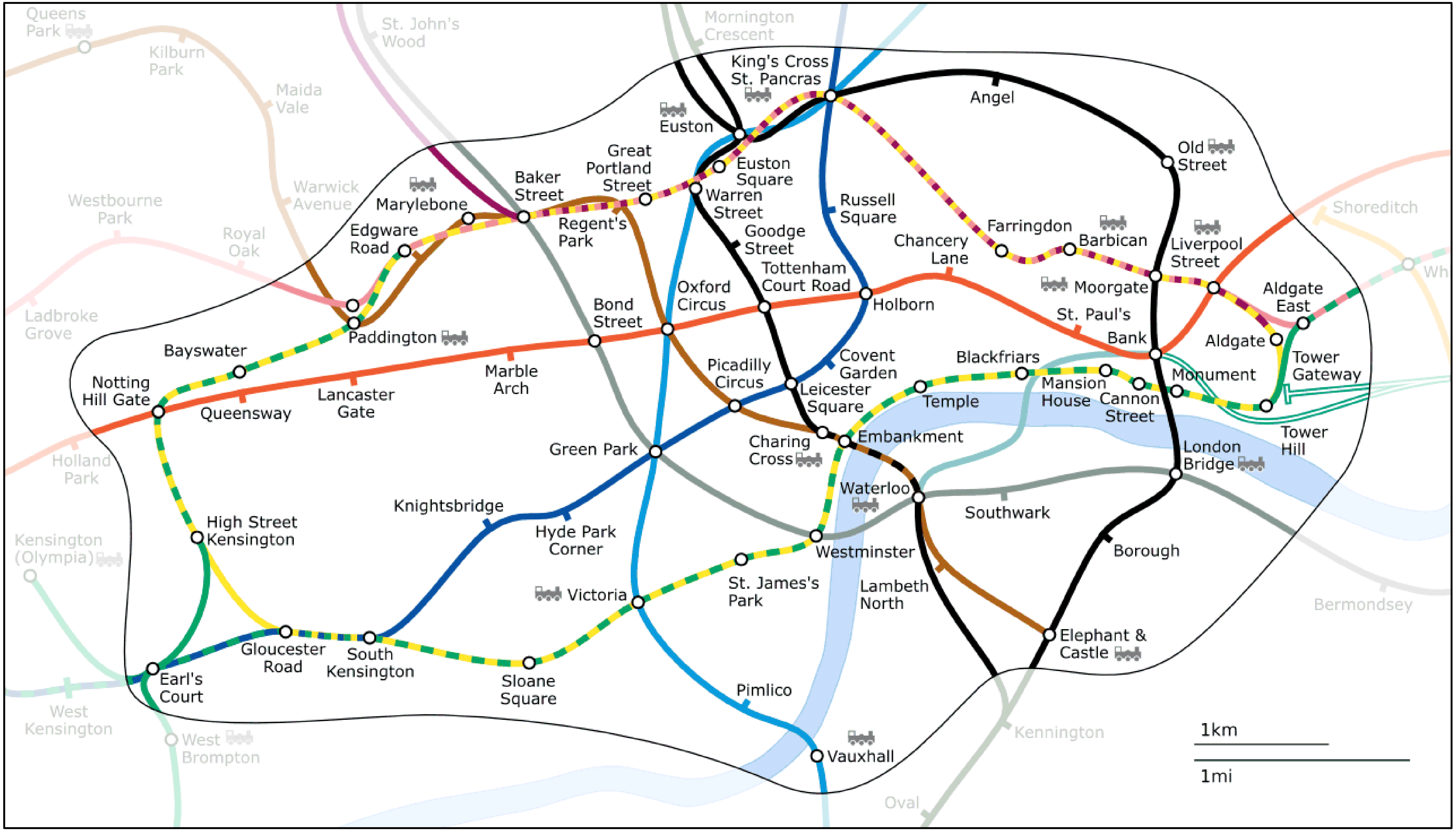




Part I:
Schematic Metro Maps

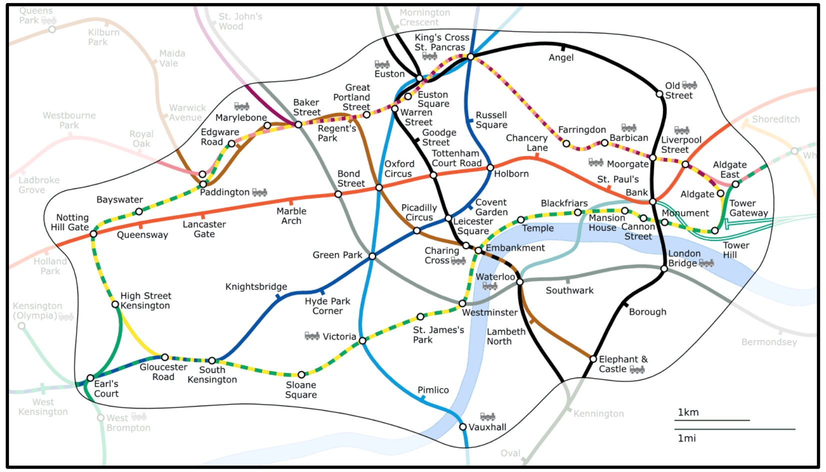
Jonathan Klawitter

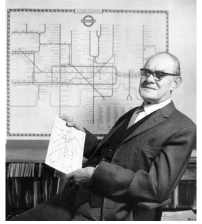
What is a Schematic Metro Map?



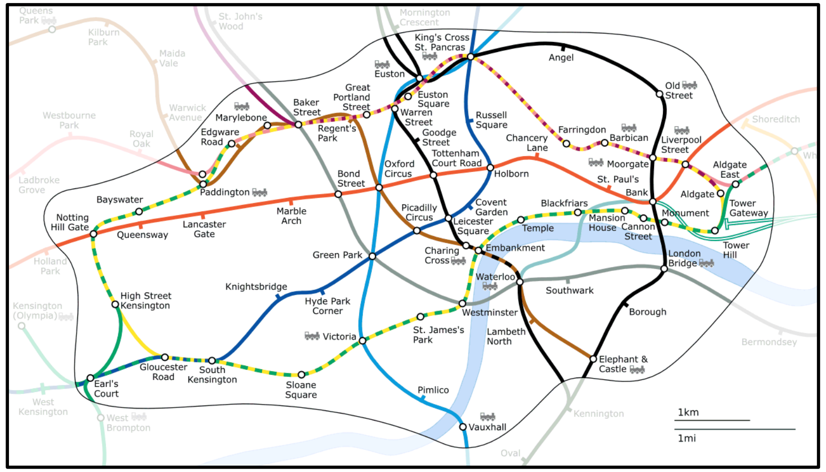
What is a Schematic Metro Map?

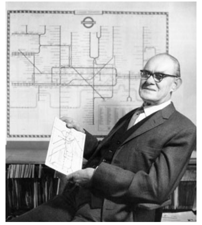
What is a Schematic Metro Map?





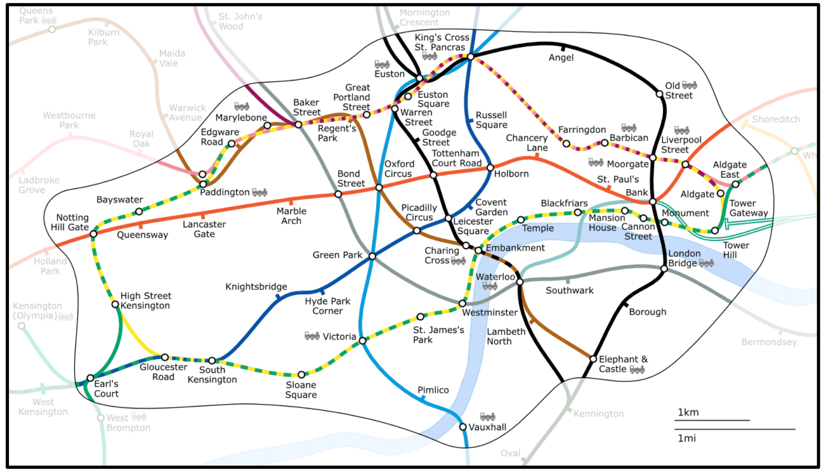
What is a Schematic Metro Map?

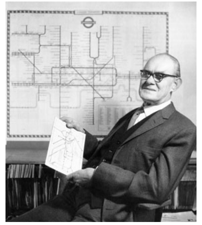




- map/diagram that shows stations connected by metro lines

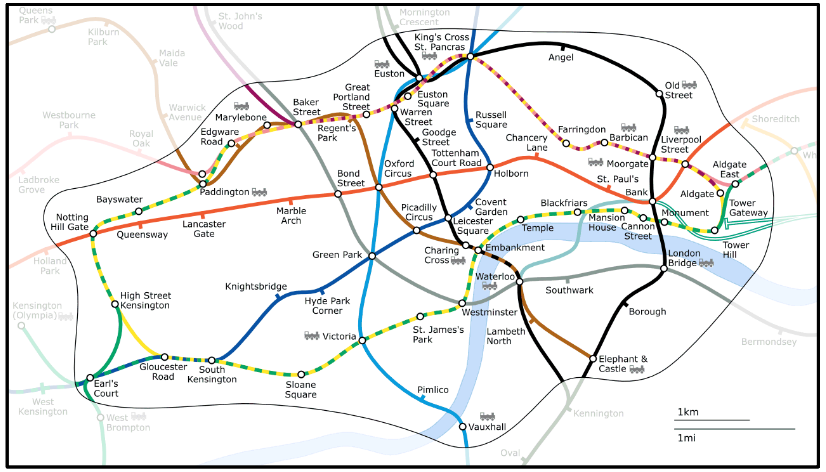
What is a Schematic Metro Map?

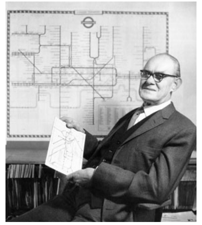




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography

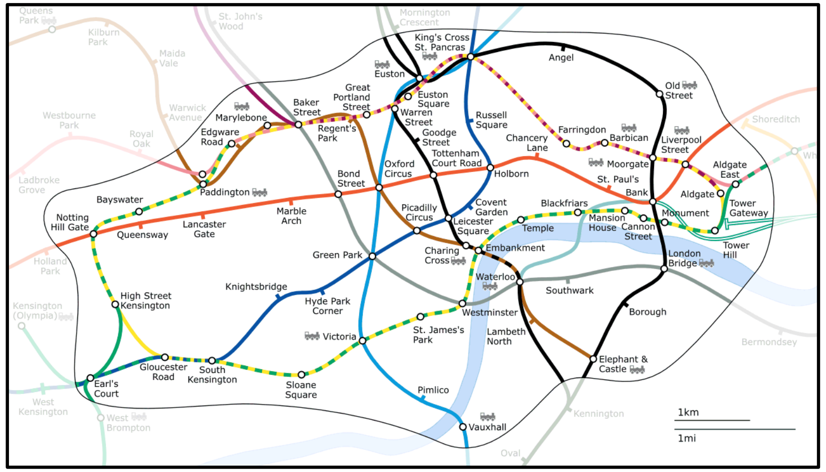
What is a Schematic Metro Map?

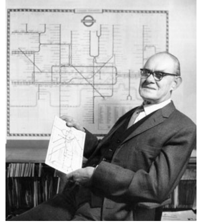




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers

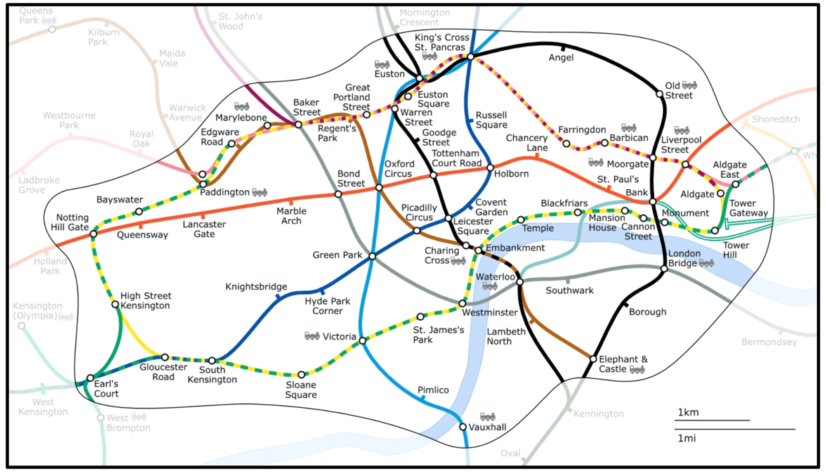
What is a Schematic Metro Map?

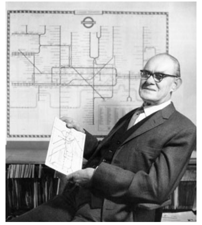




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - “*How do I quickly get from A to B?*”

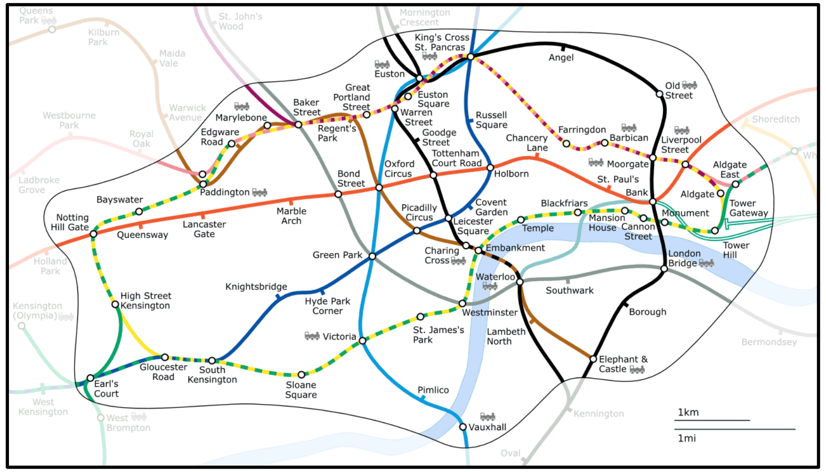
What is a Schematic Metro Map?

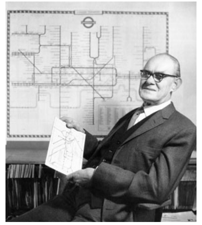




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - "*How do I quickly get from A to B?*"
  - "*Where do I need to change trains?*"

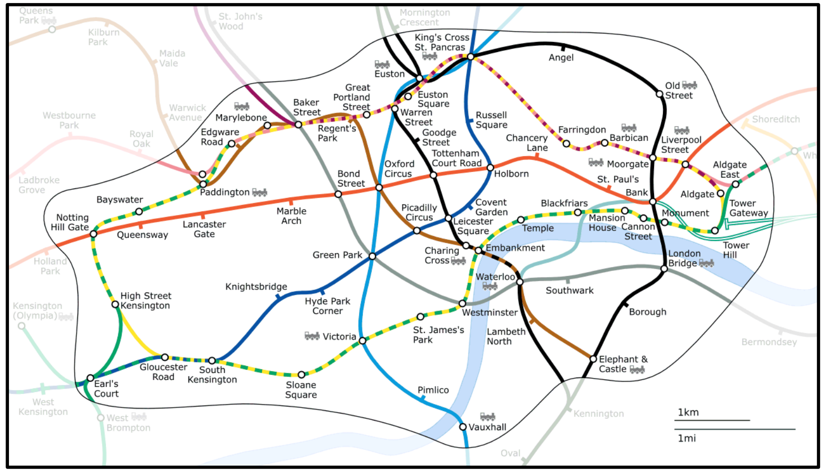
What is a Schematic Metro Map?

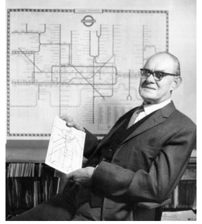




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - “*How do I quickly get from A to B?*”
  - “*Where do I need to change trains?*”
- distorts scale and geometry

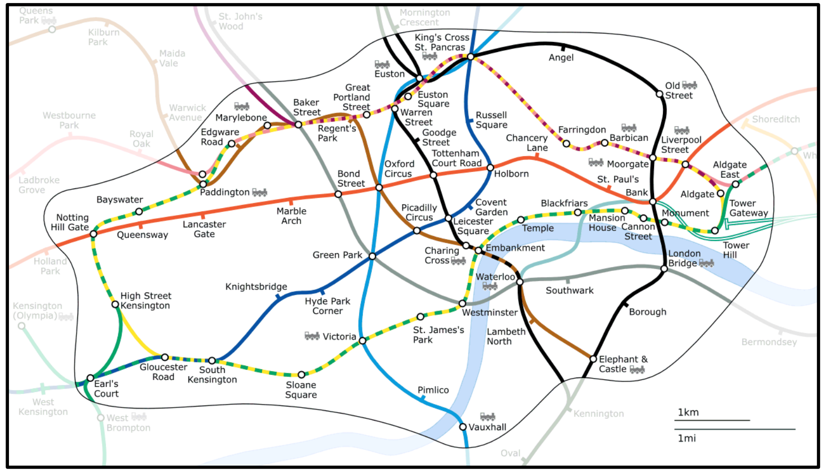
What is a Schematic Metro Map?

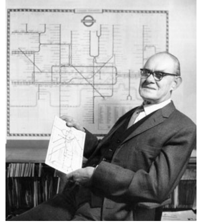




- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - “*How do I quickly get from A to B?*”
  - “*Where do I need to change trains?*”
- distorts scale and geometry
- metro map design still a largely manual process

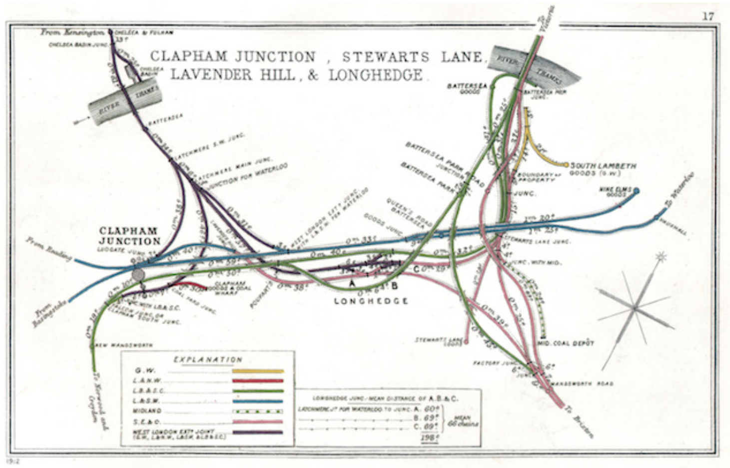
What is a Schematic Metro Map?





- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - “*How do I quickly get from A to B?*”
  - “*Where do I need to change trains?*”
- distorts scale and geometry
- metro map design still a largely manual process
- optimizing network layout computationally challenging

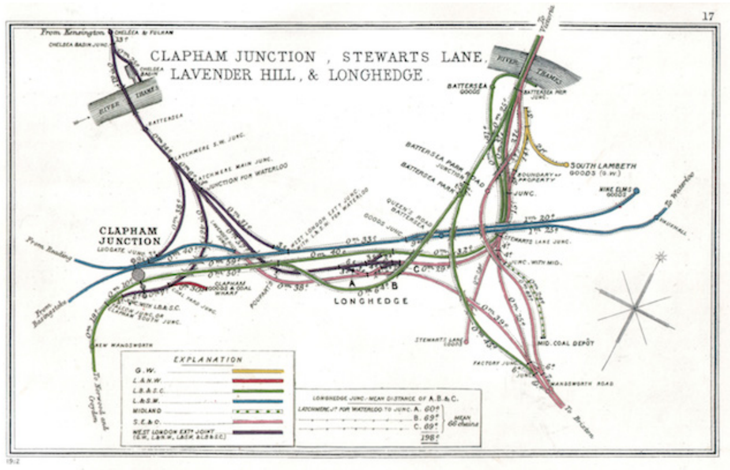
Subtasks in Metro Map Layout



Input.

- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Subtasks in Metro Map Layout



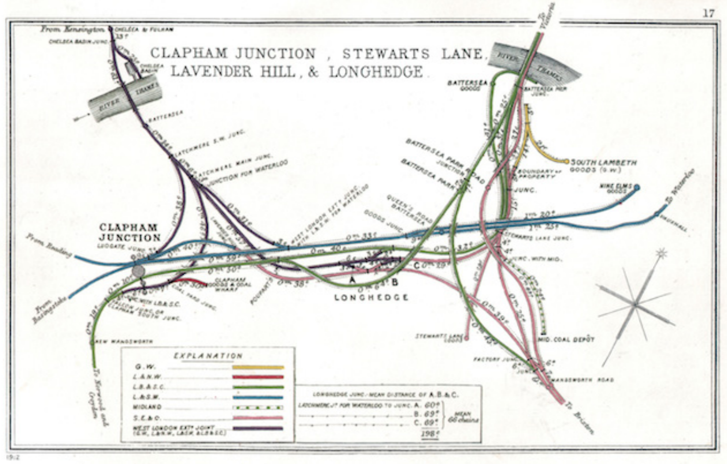
+

→

Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Subtasks in Metro Map Layout



+

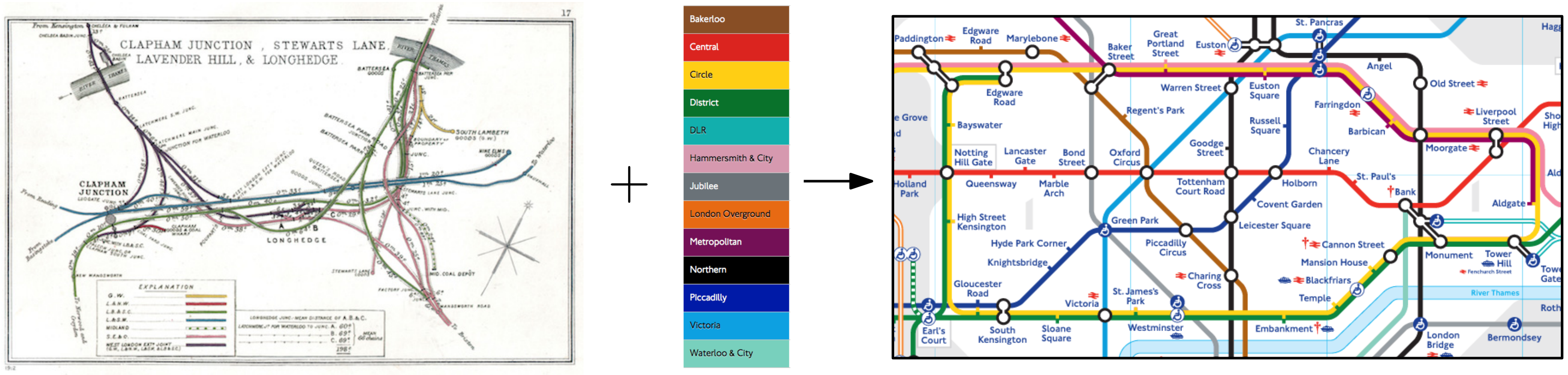
→

Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

Subtasks in Metro Map Layout



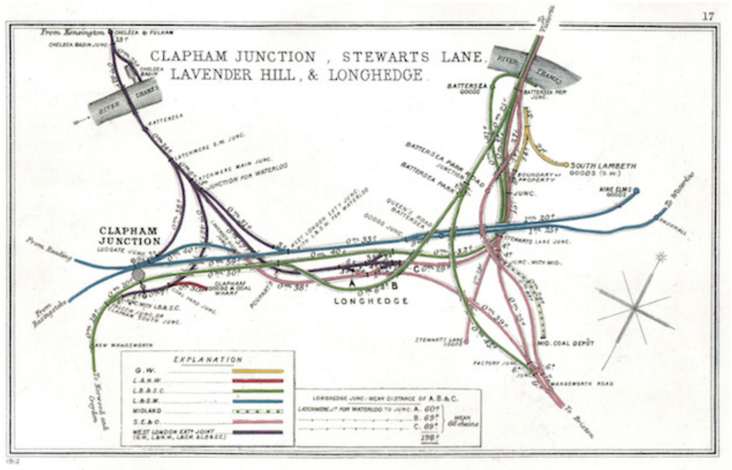
Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices

Subtasks in Metro Map Layout



+

→

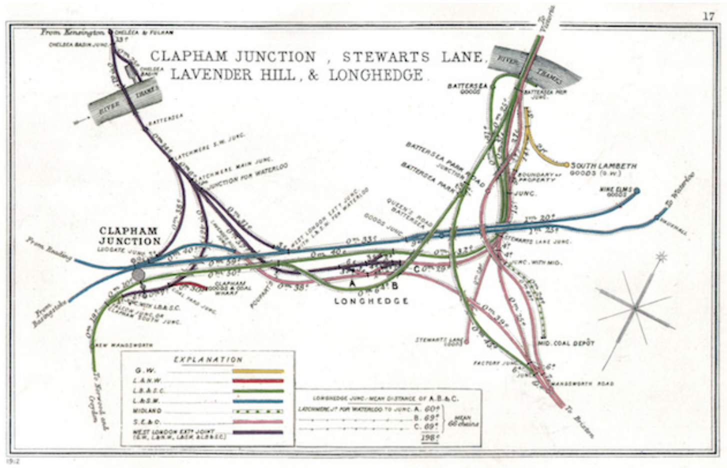
Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout

Subtasks in Metro Map Layout



+

→

Input. ■ geographically embedded railway network G

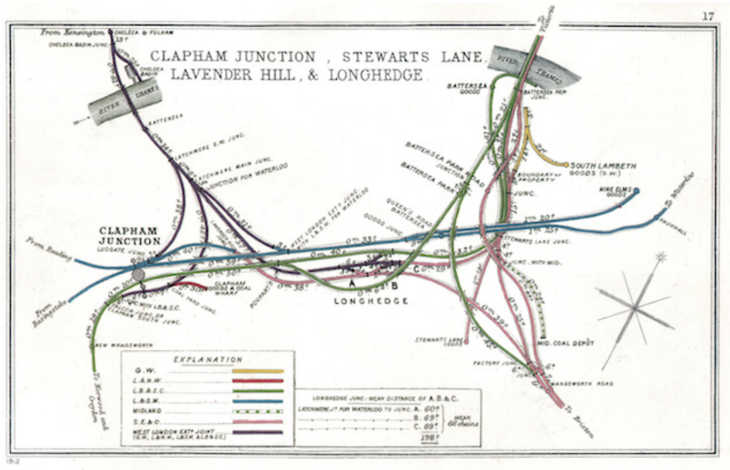
■ set of metro lines $\mathcal{L}$ serving G

Output. ■ **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling

Subtasks in Metro Map Layout



+

→

Input. ■ geographically embedded railway network G

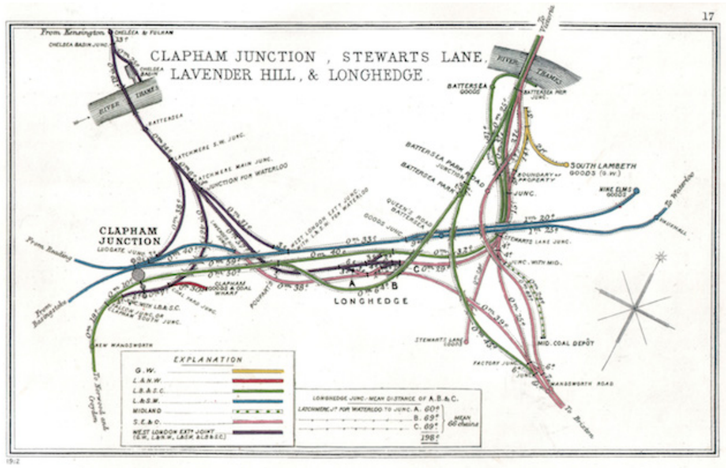
■ set of metro lines $\mathcal{L}$ serving G

Output. ■ **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

Subtasks in Metro Map Layout



$+$ $\longrightarrow$

Input. ■ geographically embedded railway network G

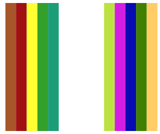
■ set of metro lines $\mathcal{L}$ serving G

Output. ■ **optimal** metro map layout (whatever it means)

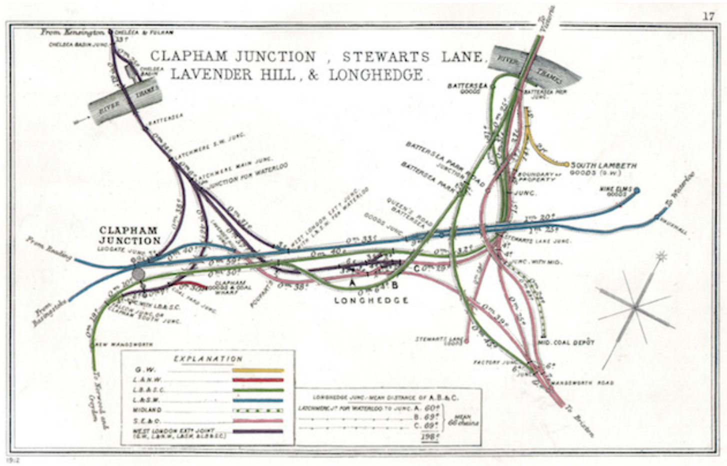
Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

colors



Subtasks in Metro Map Layout



+

→

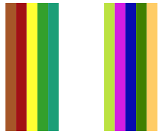
Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

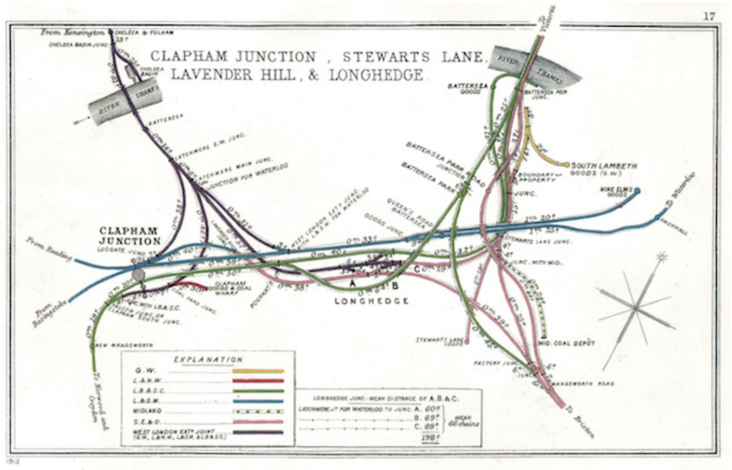
colors



fonts

Paddington
PADDINGTON
Paddington

Subtasks in Metro Map Layout



+

→

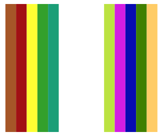
Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

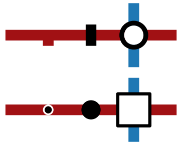
colors



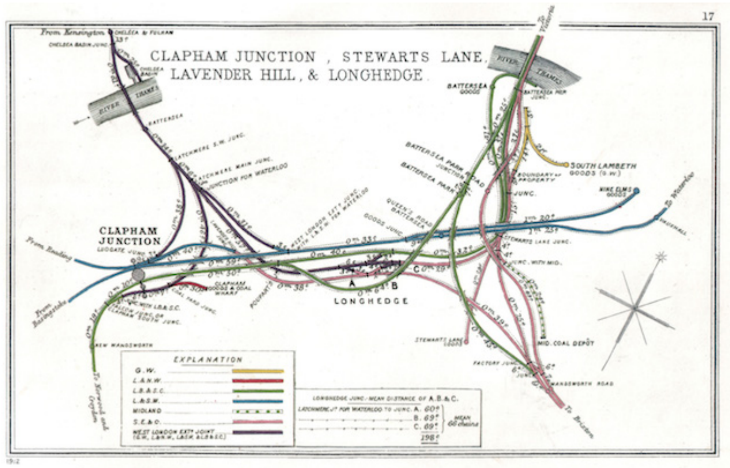
fonts

Paddington
PADDINGTON
Paddington

symbols



Subtasks in Metro Map Layout



+

→

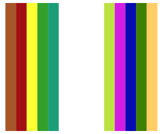
Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

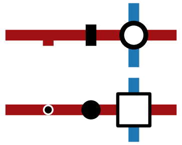
colors



fonts

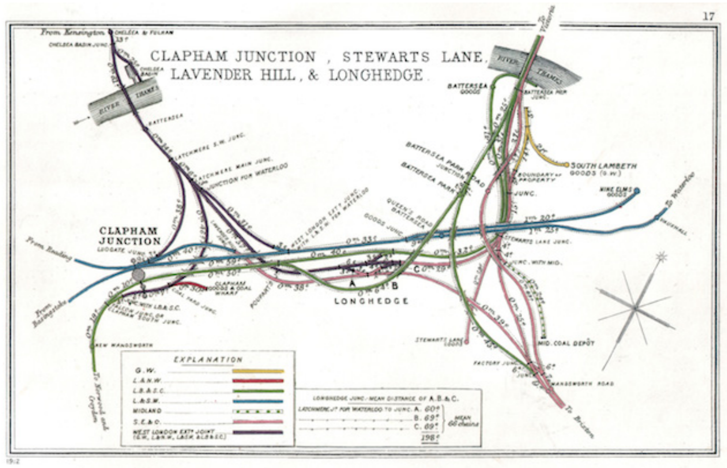
Paddington
PADDINGTON
Paddington

symbols



→ very salient,
but not a computational problem

Subtasks in Metro Map Layout



+

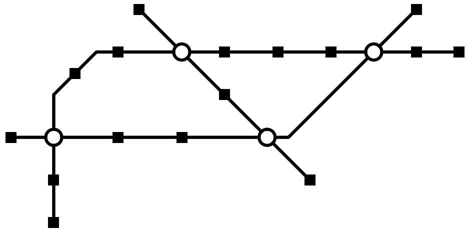
→

Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

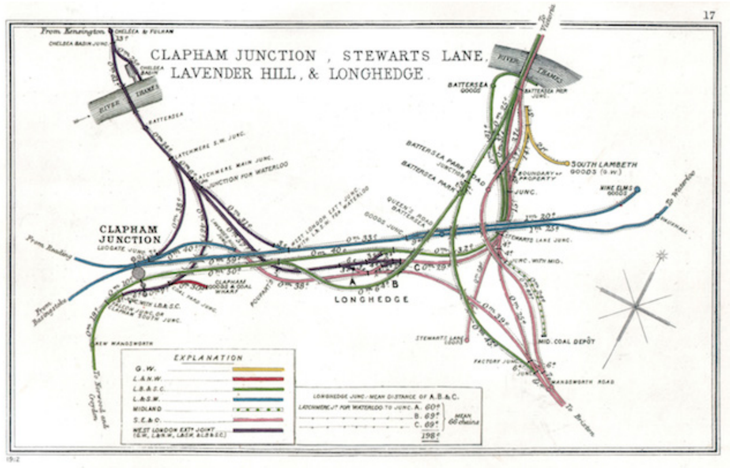
Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing



determine geometry of network layout

Subtasks in Metro Map Layout



+ →

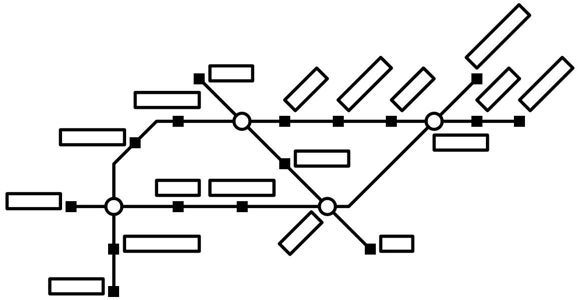
Input. ■ geographically embedded railway network G

■ set of metro lines $\mathcal{L}$ serving G

Output. ■ **optimal** metro map layout (whatever it means)

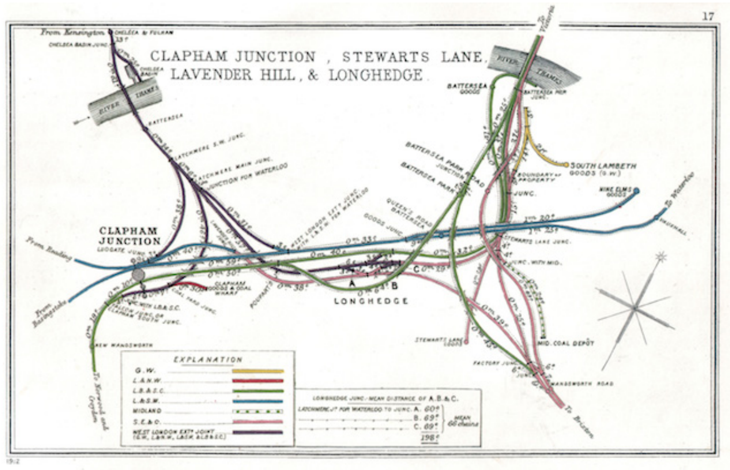
Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing



determine positions of station names

Subtasks in Metro Map Layout



+

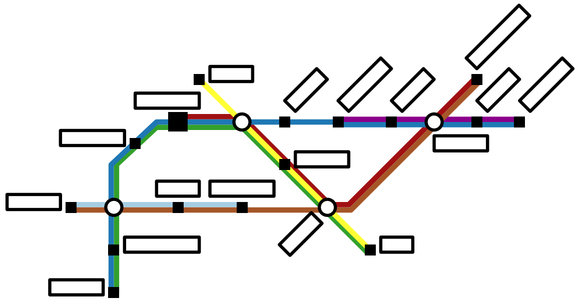
→

Input.
- geographically embedded railway network G
- set of metro lines $\mathcal{L}$ serving G

Output.
- **optimal** metro map layout (whatever it means)

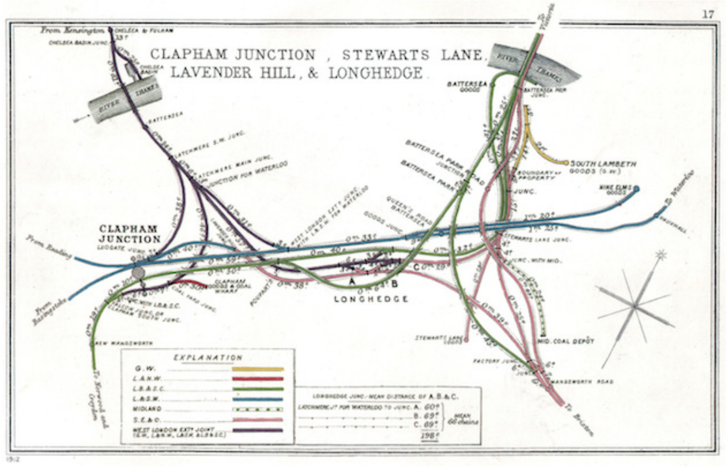
Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing



determine line routing and ordering of bundles

Subtasks in Metro Map Layout



+

→

Input. ■ geographically embedded railway network G

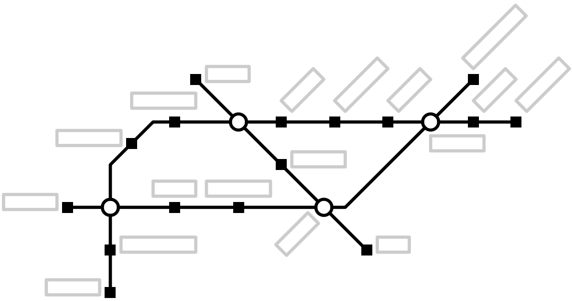
■ set of metro lines $\mathcal{L}$ serving G

Output. ■ **optimal** metro map layout (whatever it means)

Divide the task into several subtasks:

- rendering and design choices
- network layout
- station labeling
- metro line routing

focus today



Formalizing the Network Layout Problem

Given.
- graph $G = (V, E)$ geometrically embedded in $\mathbb{R}^2$
- vertex set V (stations)
- edge set E (rail links)
- set of paths $\mathcal{L}$ (metro lines in G)

Goal. **schematic** layout of $(G, \mathcal{L})$ that
- satisfies a set of layout constraints
- optimizes a set of quality criteria

Formalizing the Network Layout Problem

Given.
- graph $G = (V, E)$ geometrically embedded in $\mathbb{R}^2$
- vertex set V (stations)
- edge set E (rail links)
- set of paths $\mathcal{L}$ (metro lines in G)

Goal. **schematic** layout of $(G, \mathcal{L})$ that
- satisfies a set of layout constraints
- optimizes a set of quality criteria

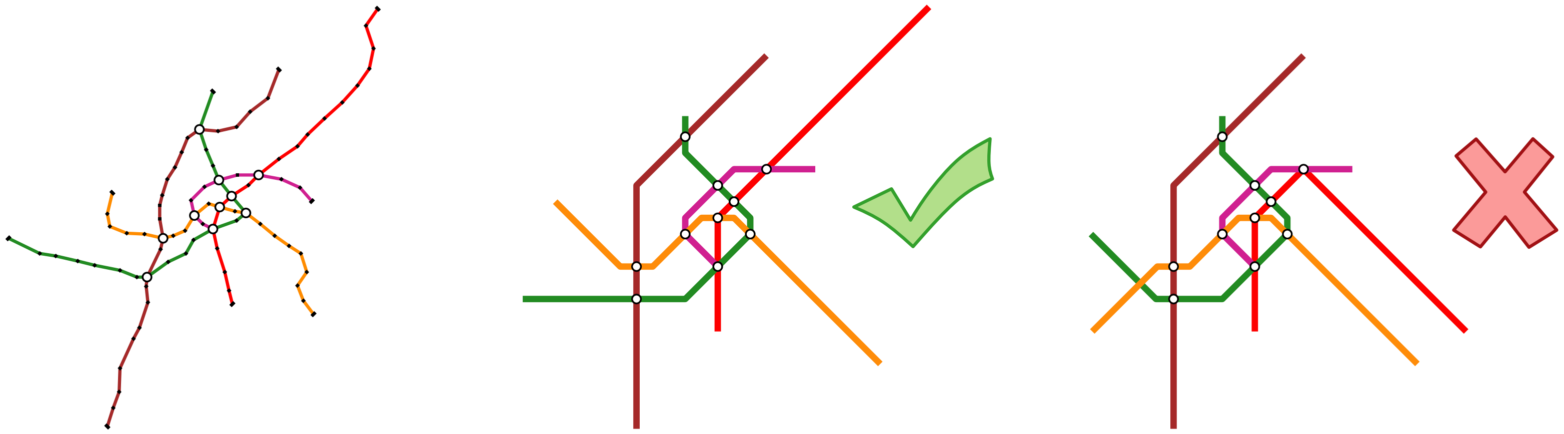
But what are the constraints and quality criteria?

$\rightarrow$ extract common principles of existing, manually designed metro maps

Design Rules

(R1) Do not change the network topology.

- no new crossings
- no changes in circular vertex orders

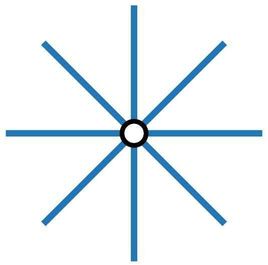


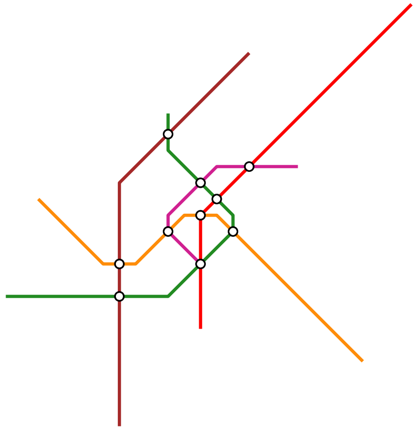
Design Rules

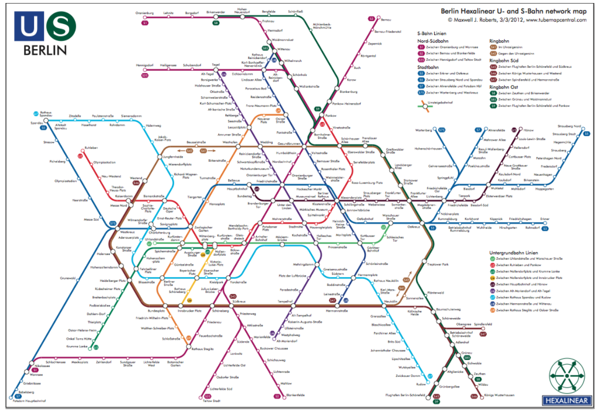
(R1) Do not change the network topology.

(R2) Restrict edge orientations.

- mostly octilinear (octolinear) orientation systems
- also curvilinear and other alternative orientation systems







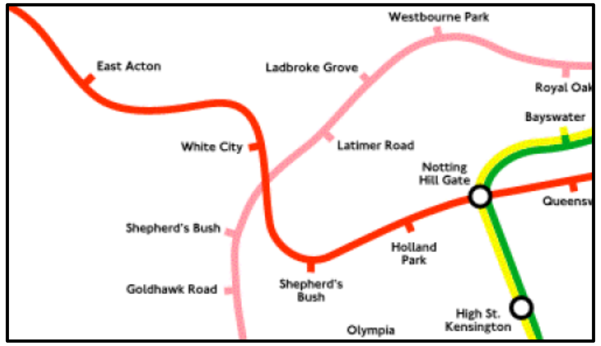
Design Rules

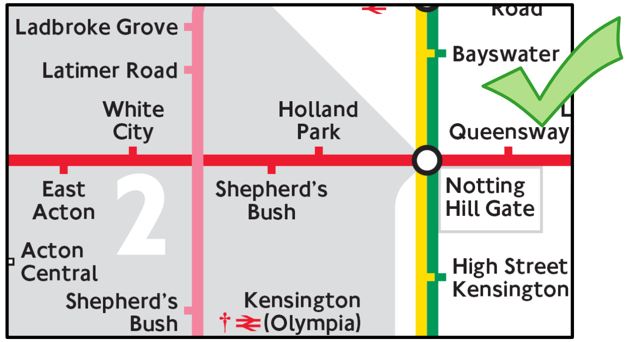
(R1) Do not change the network topology.

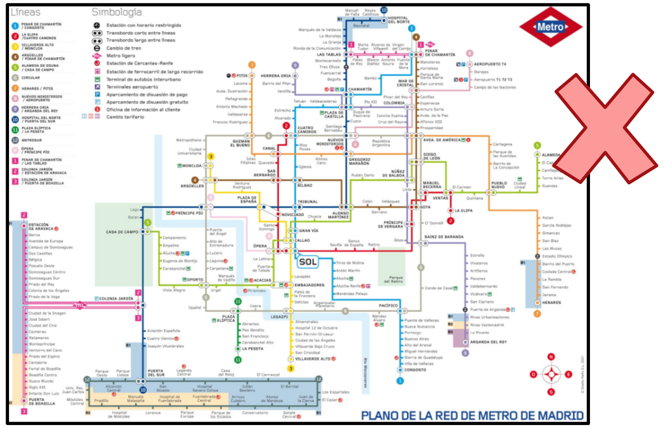
(R2) Restrict edge orientations.

(R3) Draw metro lines as simple and monotone as possible.

- avoid bends
- prefer obtuse bend angles
- for curves: prefer uniform curvature, few inflection points







Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

(R3) Draw metro lines as simple and monotone as possible.

(R4) Let lines pass straight through interchanges.

- avoids visual ambiguities in complex stations

Design Rules

(R1) Do not change the network topology.

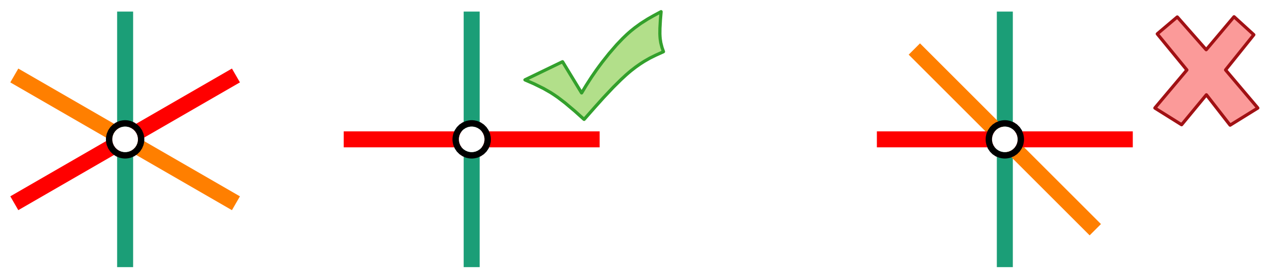
(R2) Restrict edge orientations.

(R3) Draw metro lines as simple and monotone as possible.

(R4) Let lines pass straight through interchanges.

(R5) Use large angular resolution in stations.

- distributes edges evenly for balanced appearance



Design Rules

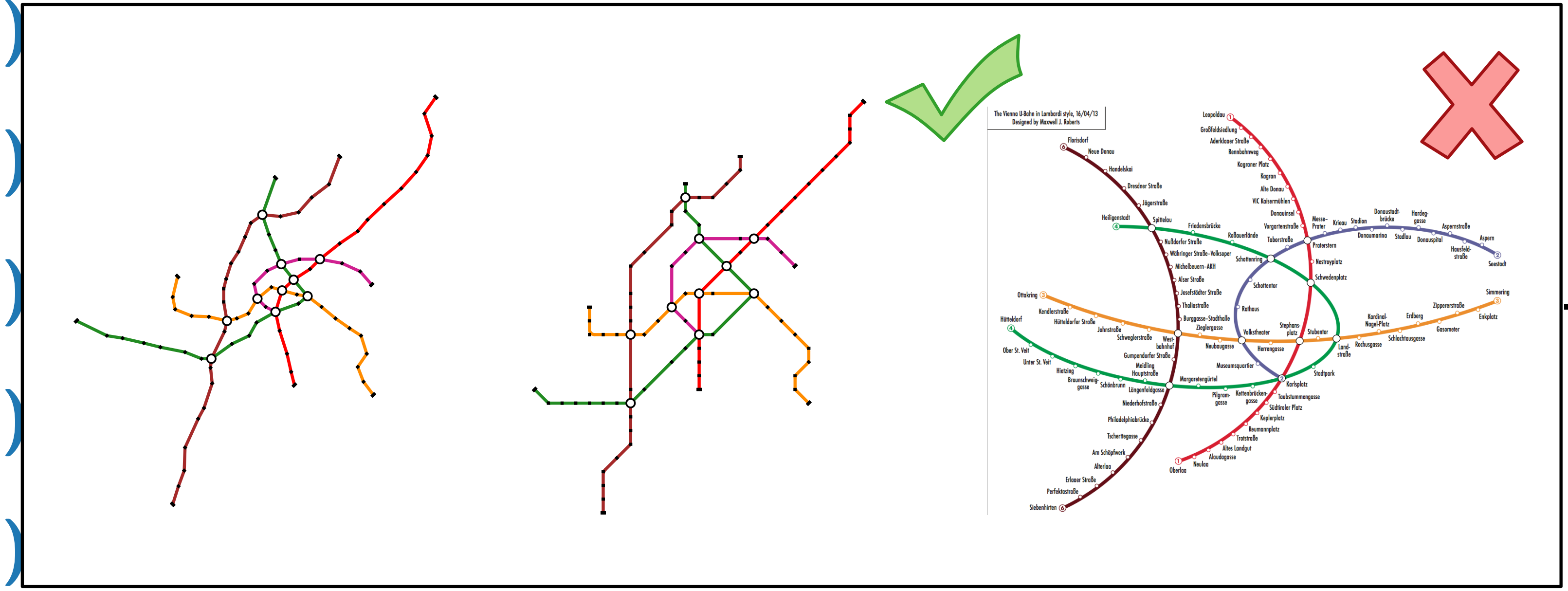
(R1)

(R2)

(R3)

(R4)

(R5)



(R6) Minimize geometric distortion and displacement.

- maintains resemblance to geography
- preserves user's mental map
- applicable locally or globally

topographicity

Design Rules

(R1) Do not change the network topology.

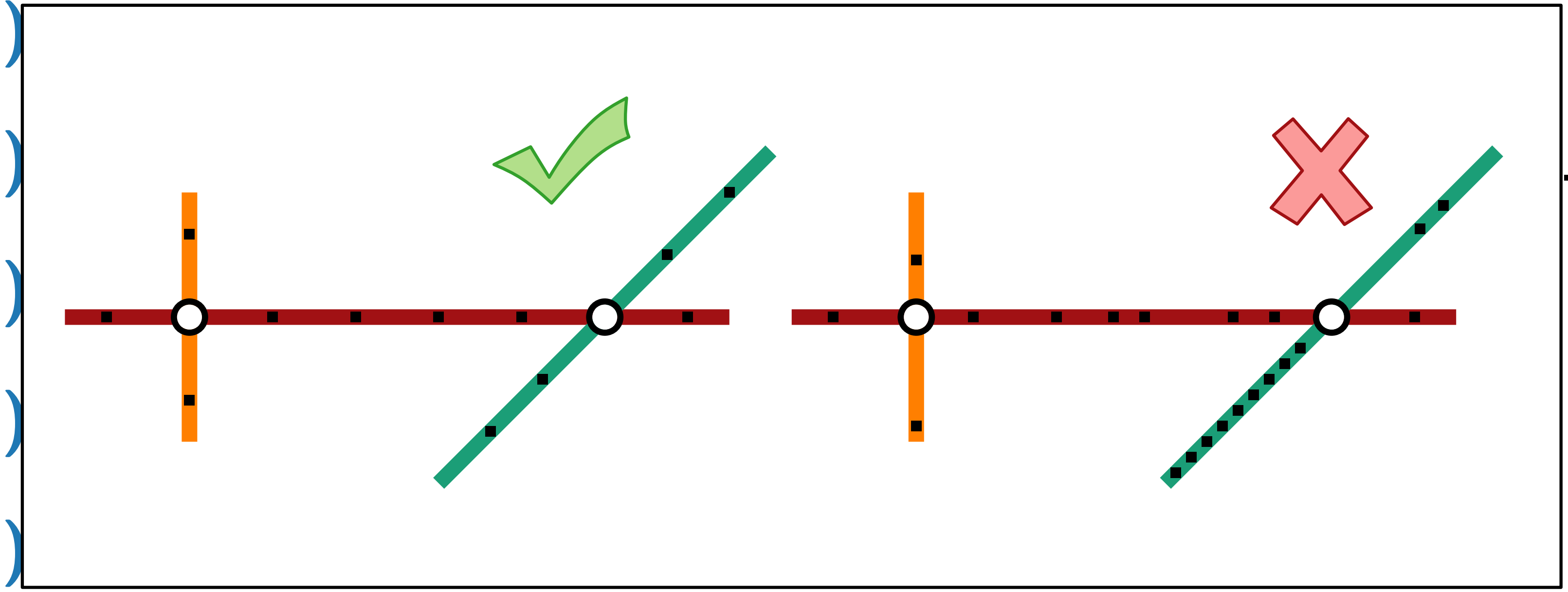
(R2)

(R3)

(R4)

(R5)

(R6)



(R7) Use uniform edge lengths.

- geographic distances less important
- network hop-distances more important
- balanced appearance

Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

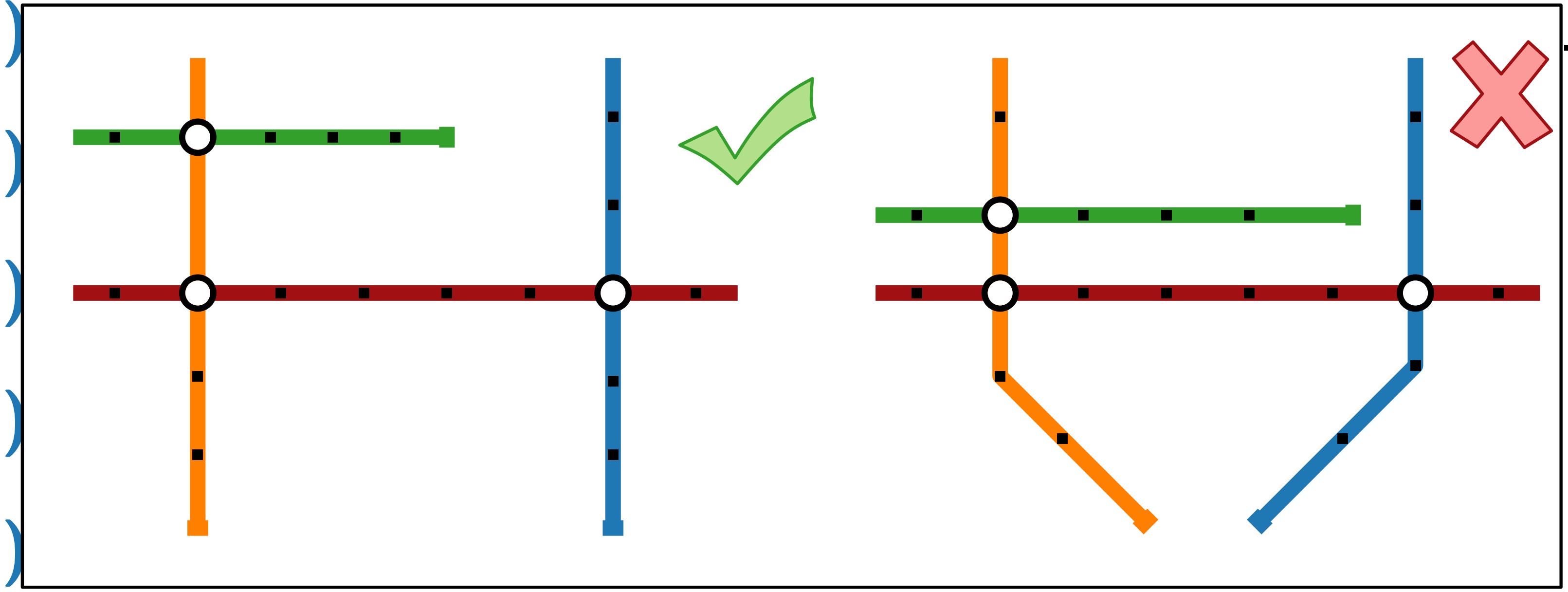
(R3)

(R4)

(R5)

(R6)

(R7)



(R8) Keep unrelated features apart.

- guarantees minimum clearance between features
- avoids ambiguities

Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

(R3)

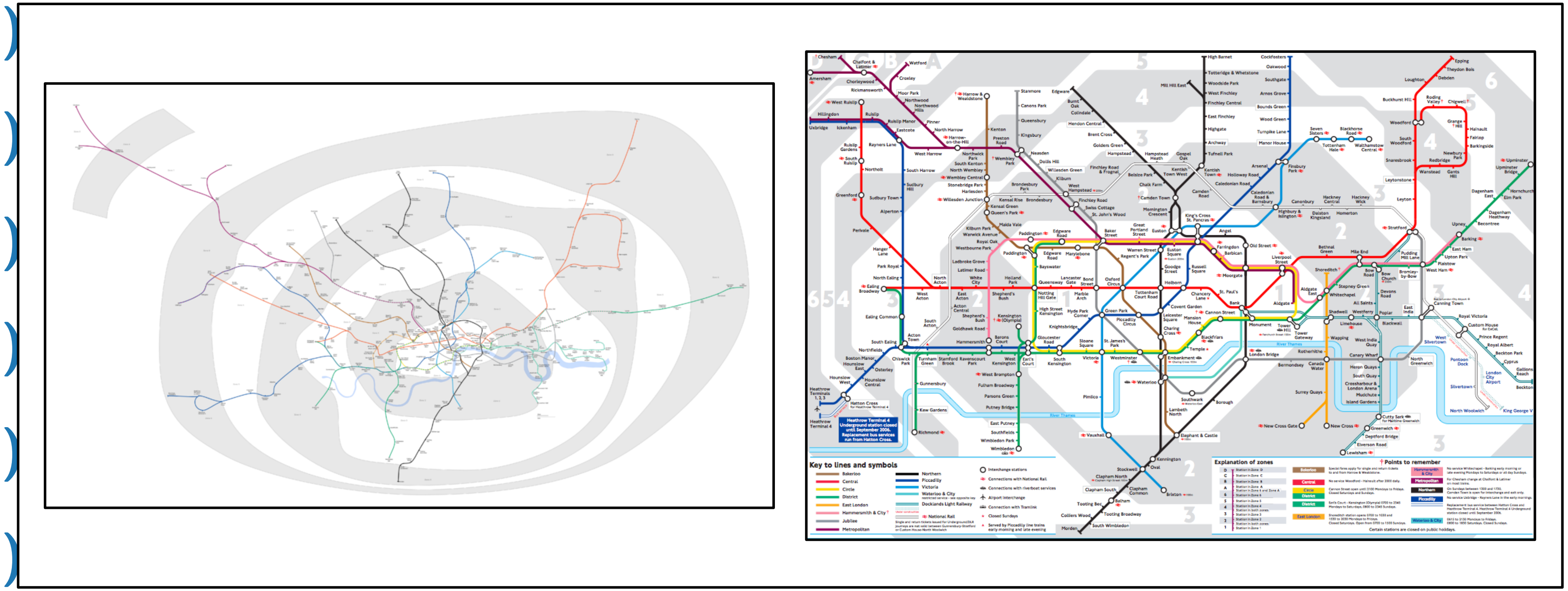
(R4)

(R5)

(R6)

(R7)

(R8)



(R9) Avoid large empty spaces.

- balances local feature density
- possibly fill gaps with legends

Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

(R3) Draw metro lines as simple and monotone as possible.

(R4) Let lines pass straight through interchanges.

(R5) Use large angular resolution in stations.

(R6) Minimize geometric distortion and displacement.

(R7) Use uniform edge lengths.

(R8) Keep unrelated features apart.

(R9) Avoid large empty spaces.

Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

(R3) Draw metro lines as simple and monotone as possible.

(R4) Let lines pass straight through interchanges.

(R5) Use large angular resolution in stations.

(R6) Minimize geometric distortion and displacement.

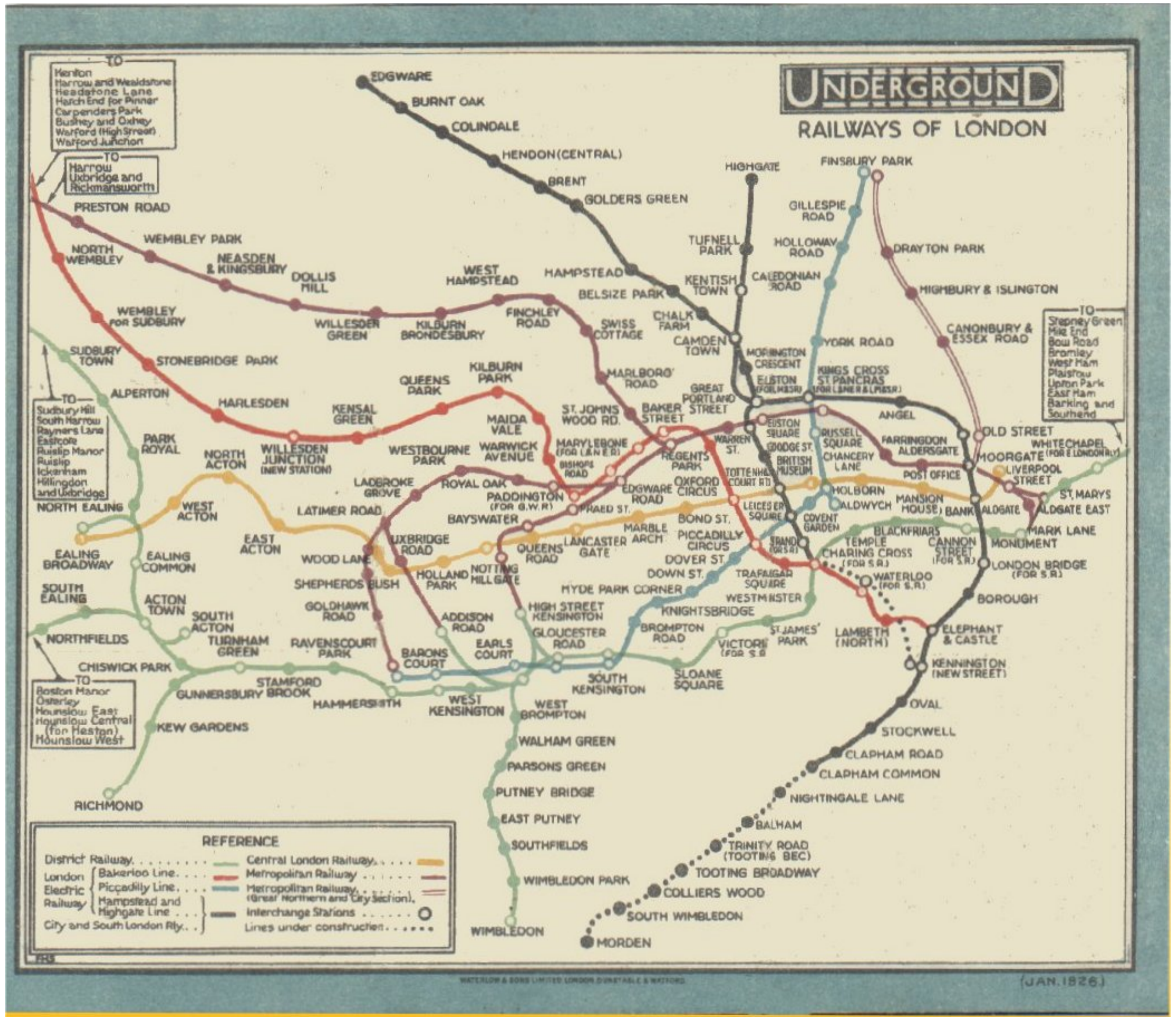
(R7) Use uniform edge lengths.

(R8) Keep unrelated features apart.

(R9) Avoid large empty spaces.

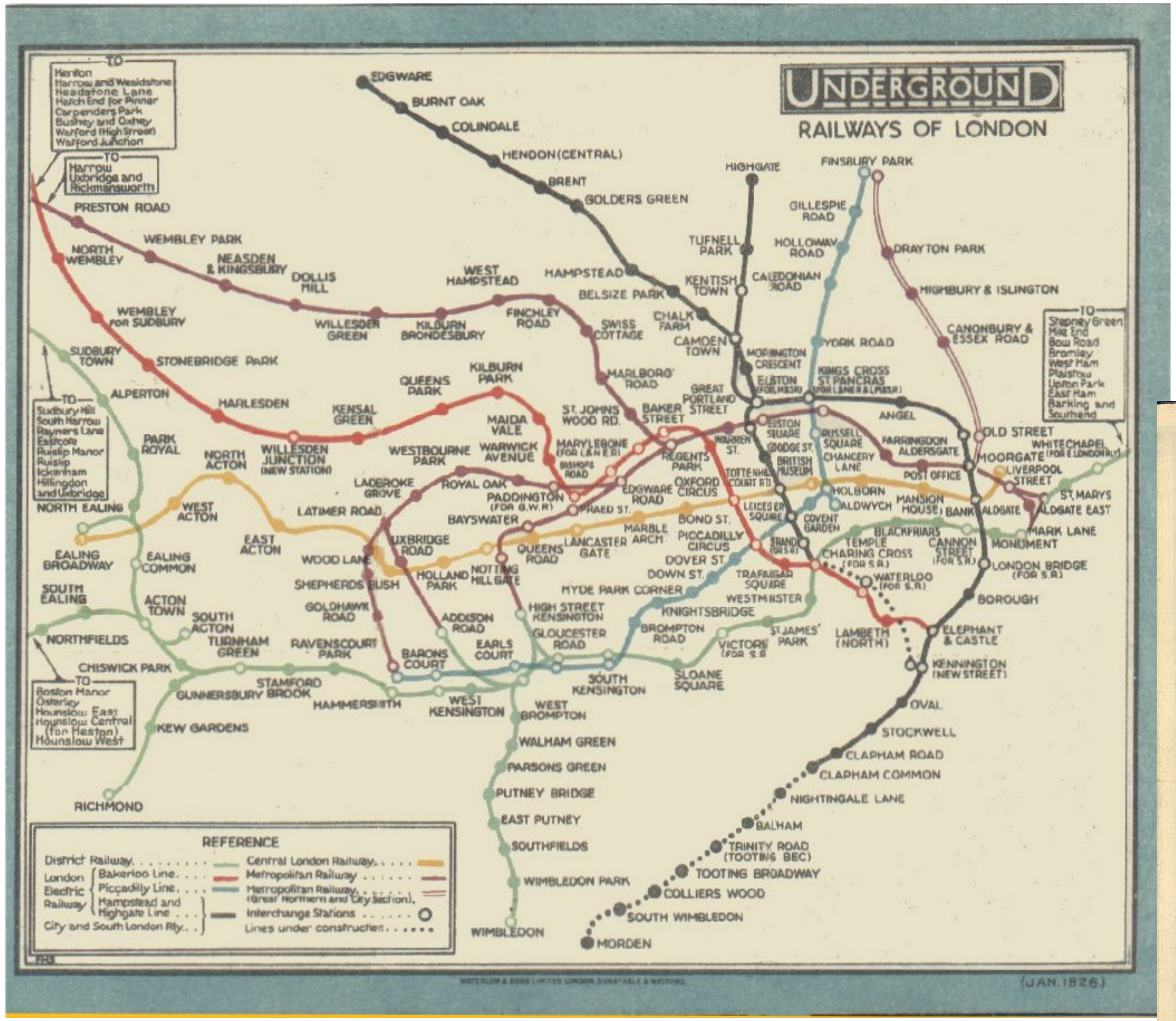
$\rightarrow$ **rules are potentially conflicting and need priorities**

A Bit of History



London 1927 (Fred H. Stingemore)

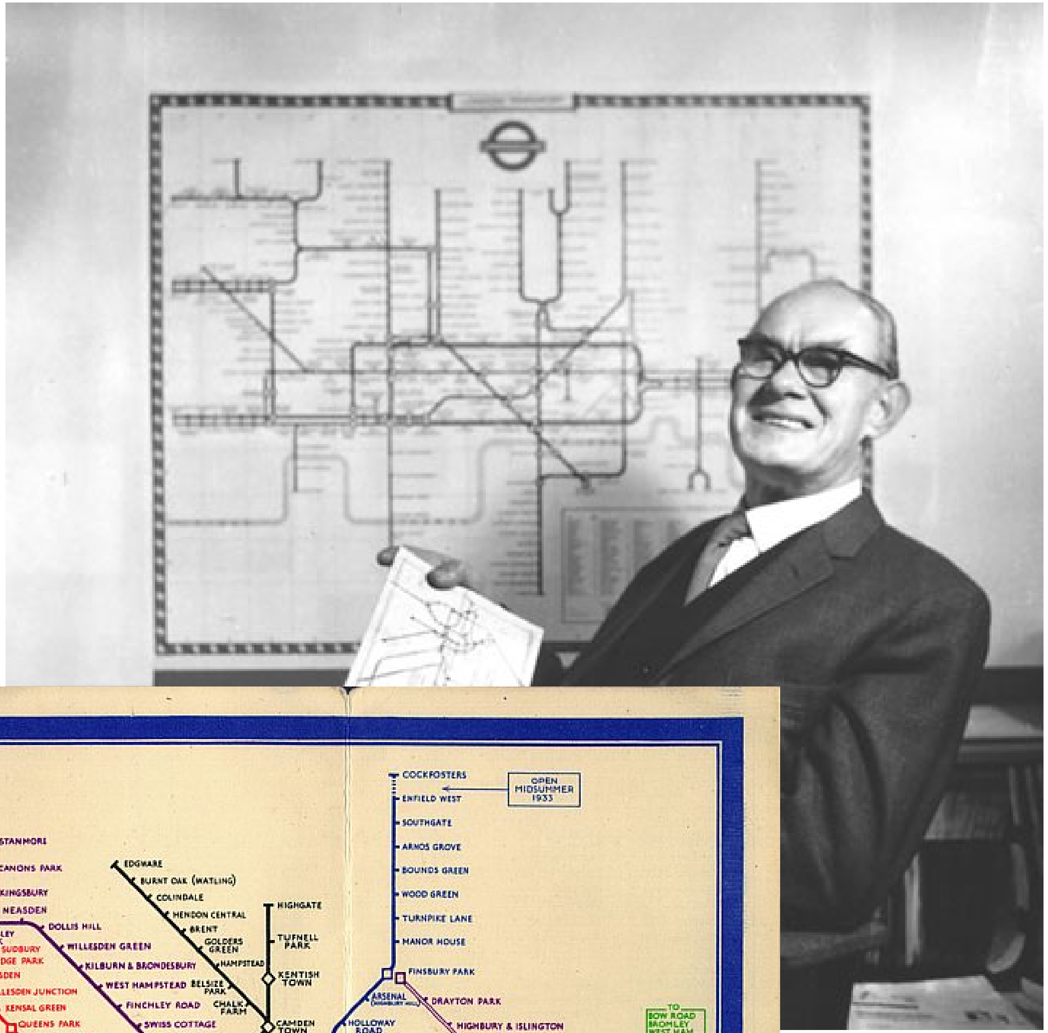
A Bit of History



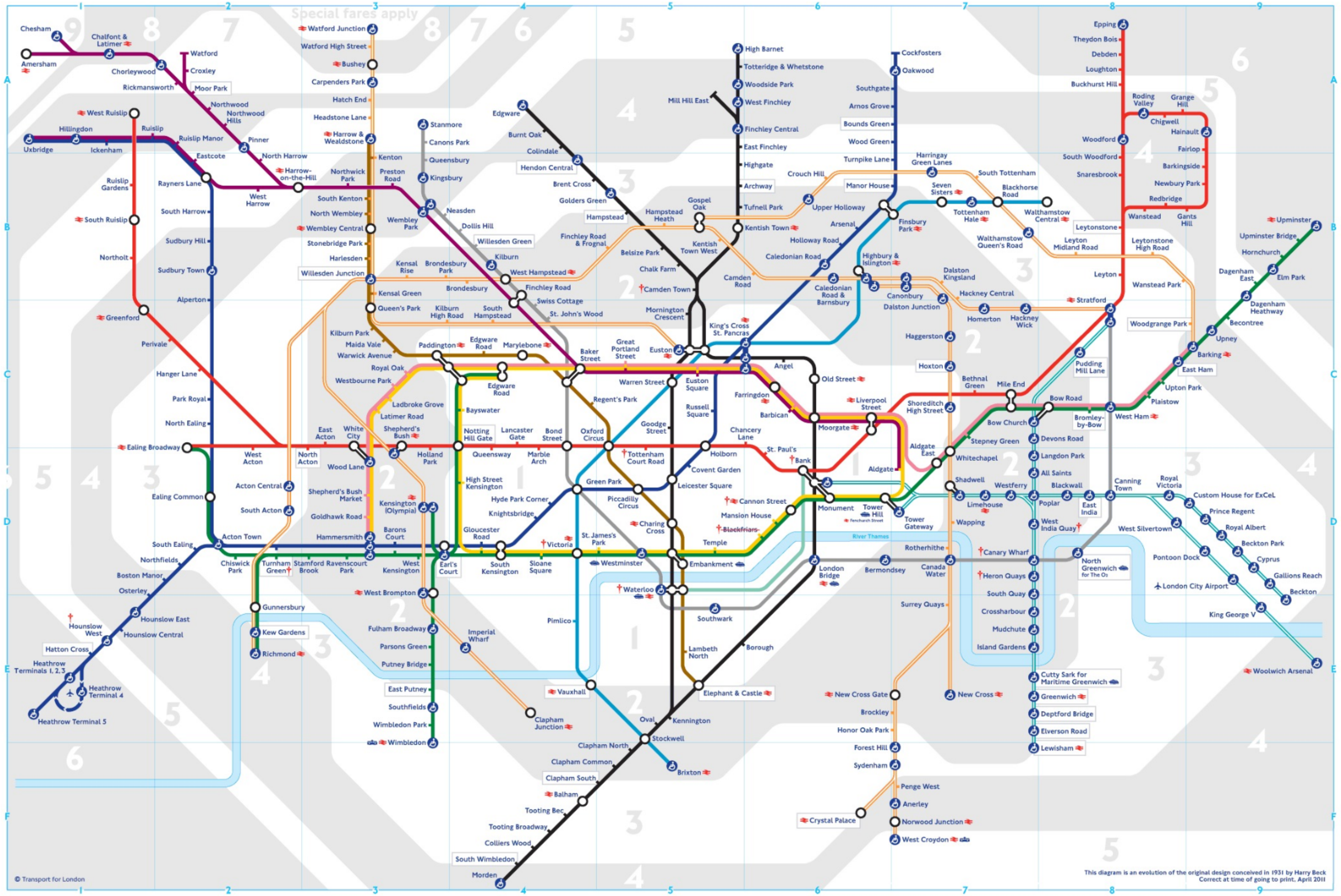
London 1927 (Fred H. Stingemore)

(c) Mike Yashworth

Henry Beck
London 1933

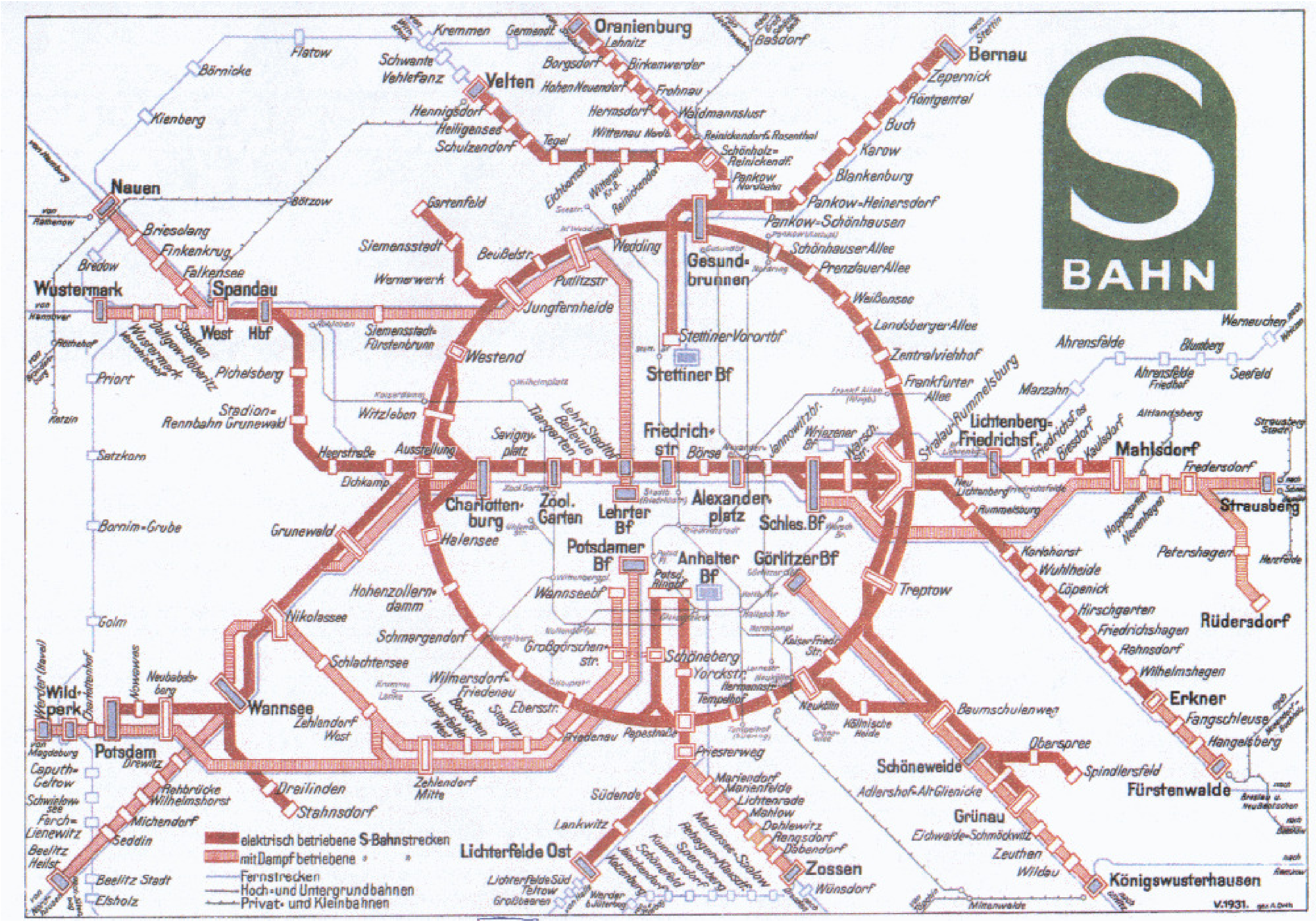


A Bit of History



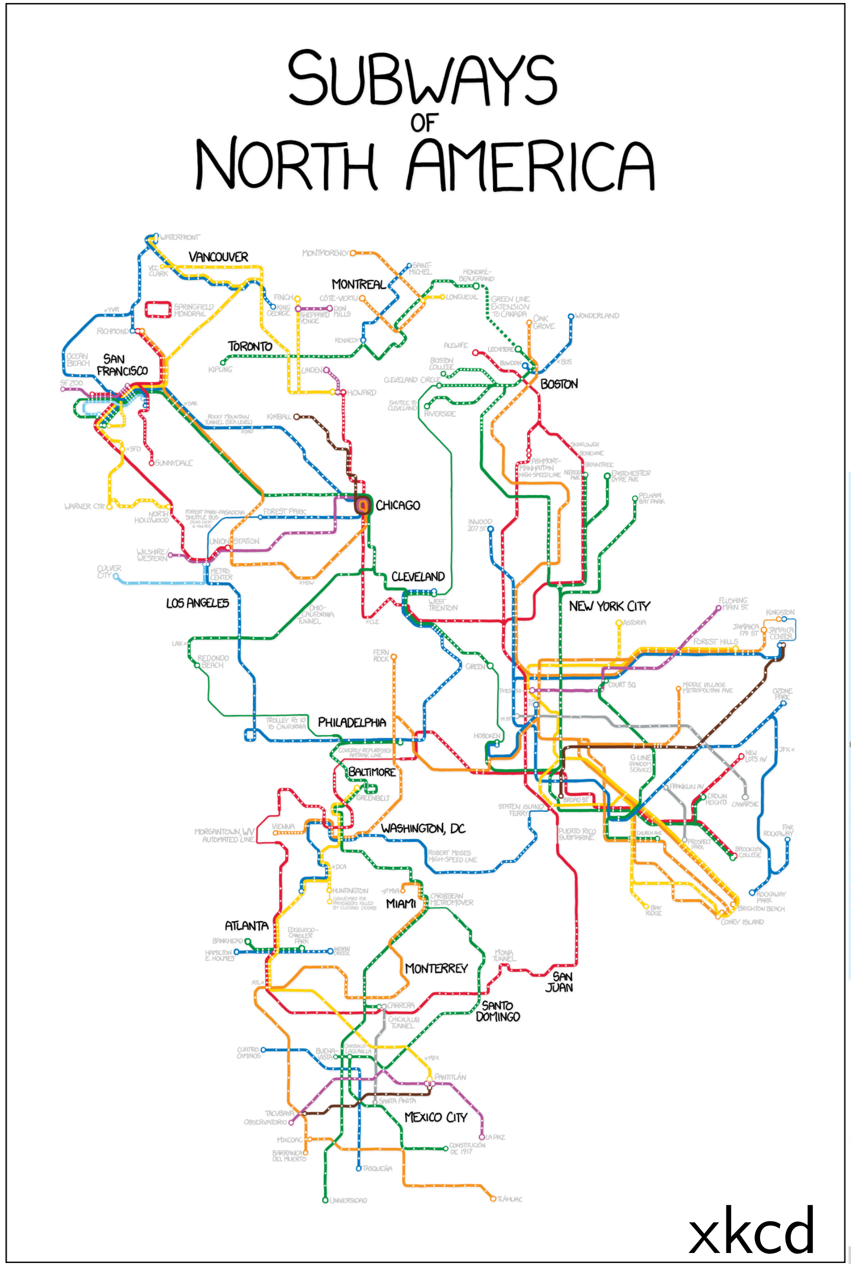
Tube Map
voted Design Icon 2006
(2nd after Concorde)

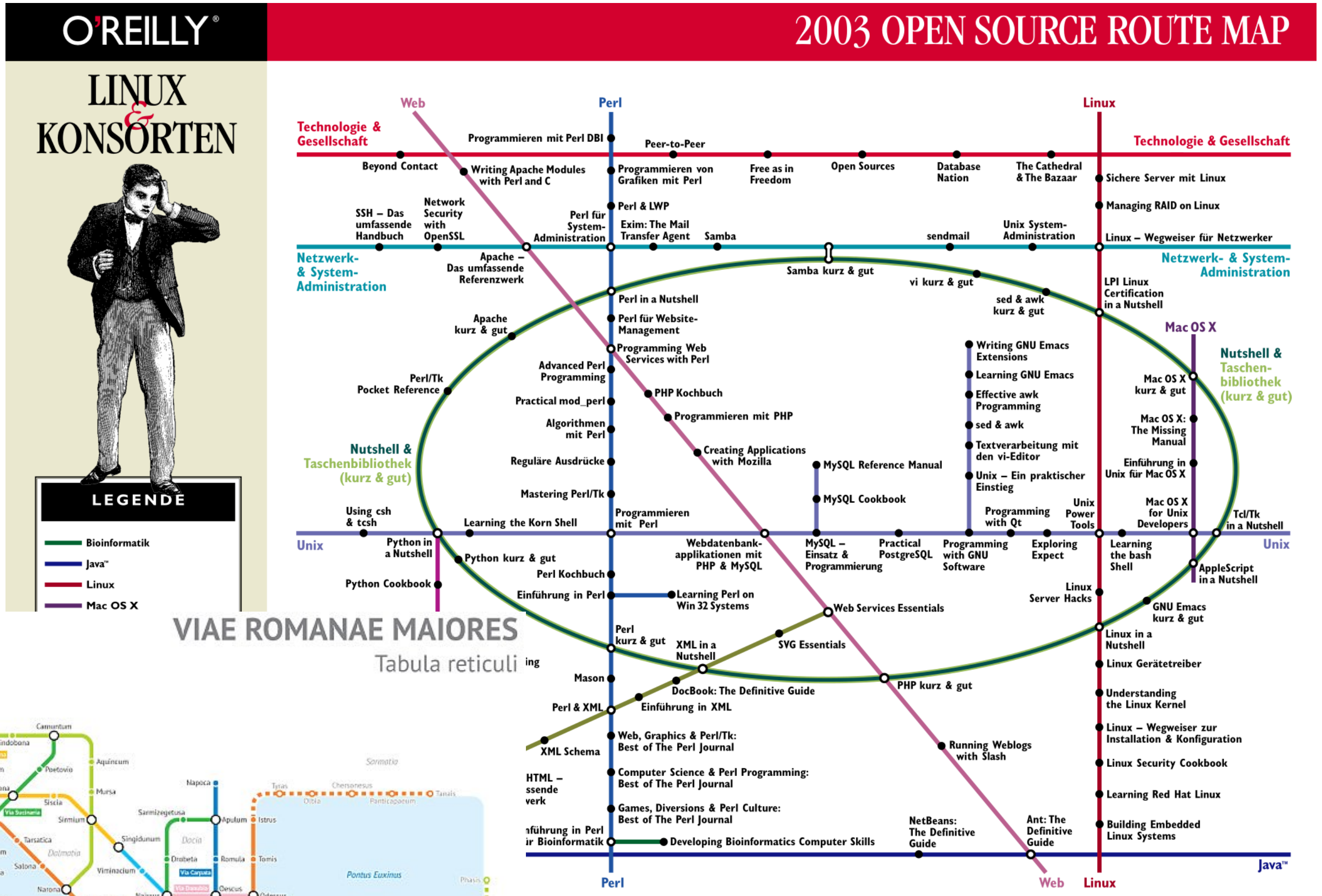
A Bit of History



Berlin 1931 (redrawn by Maxwell Roberts)

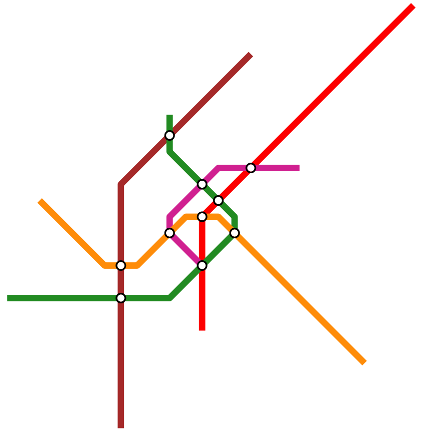
A Bit of History



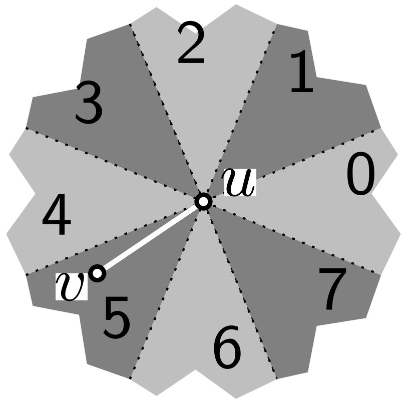


Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

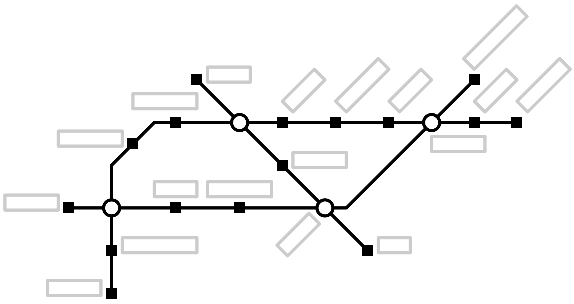
Visualization of Graphs



Lecture 12:
Octilinear Graph Drawing
Metro Map Layout



Part II:
Complexity and Path-Based Schematization



Jonathan Klawitter

Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

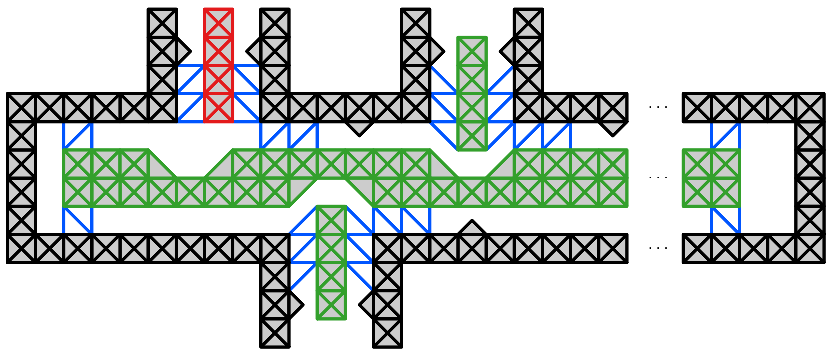
Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid "mechanical" gadgets

Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees $\leq$ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid "mechanical" gadgets

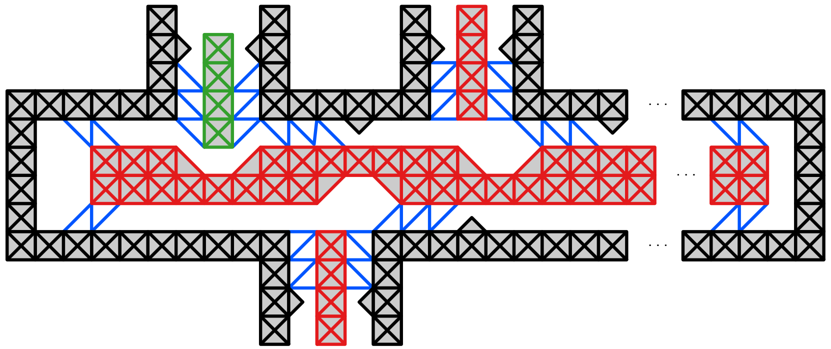


Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid "mechanical" gadgets

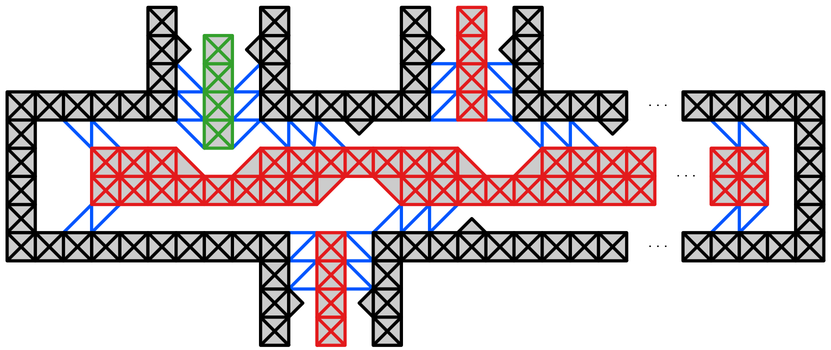


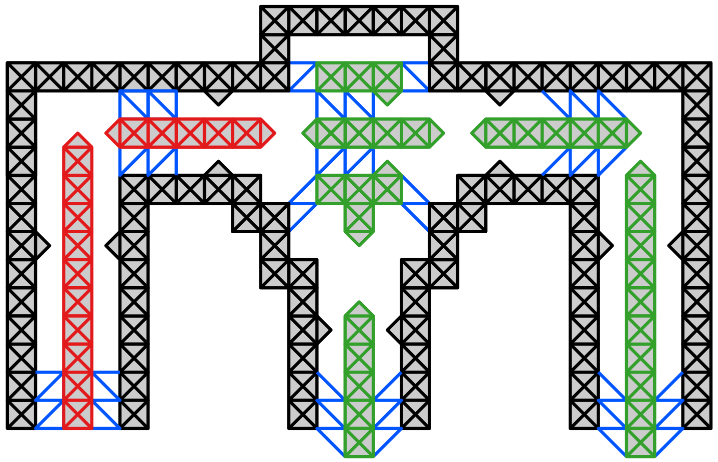
Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees $\leq$ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid "mechanical" gadgets



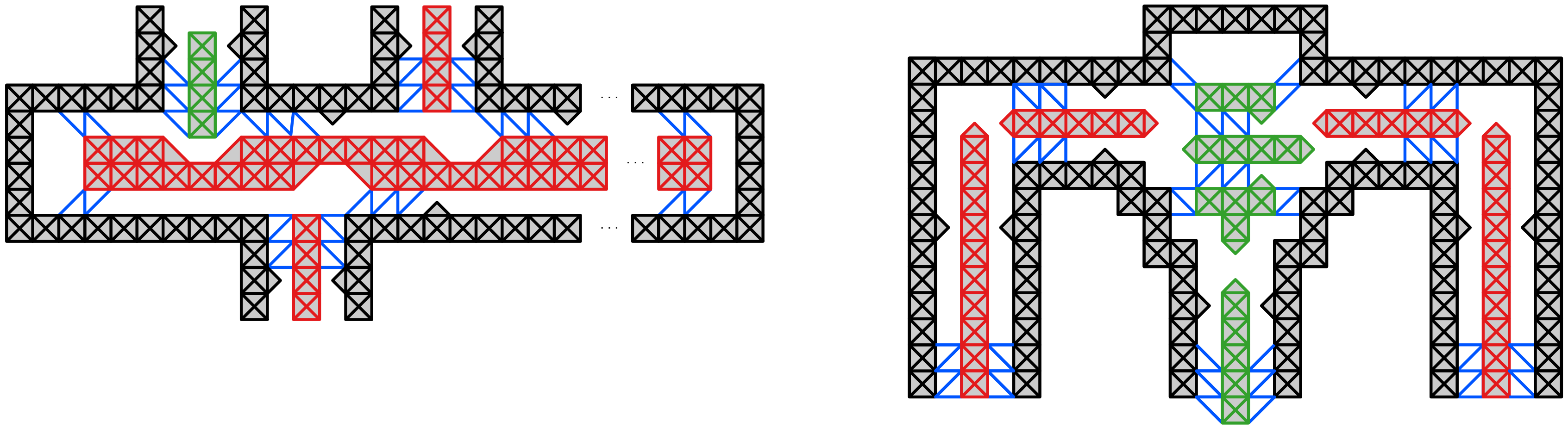


Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid “mechanical” gadgets

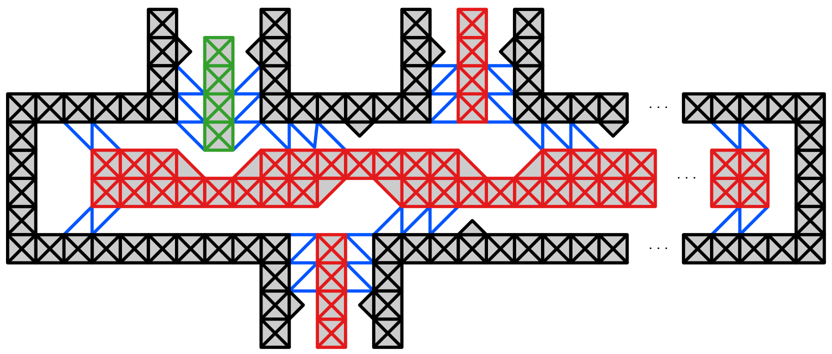


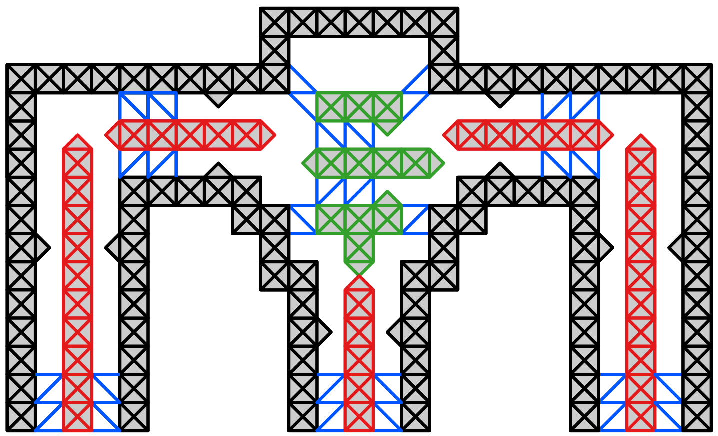
Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees $\leq$ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem PLANAR-3SAT using rigid "mechanical" gadgets



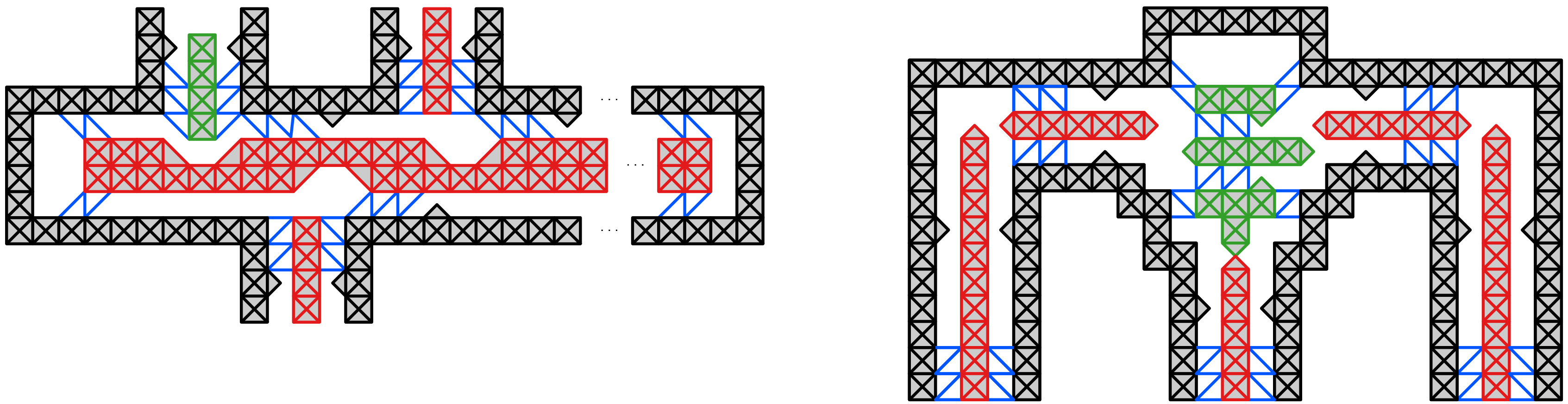


Complexity

Theorem 1. [Nöllenburg 2005]
For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

Sketch of proof.

Reduction from Boolean satisfiability problem $\textsc{planar-3Sat}$ using rigid "mechanical" gadgets



Remark.

- no efficient exact algorithms to expect
- same problem without diagonals (rectilinear) is efficiently solvable [Tamassia '87]

Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

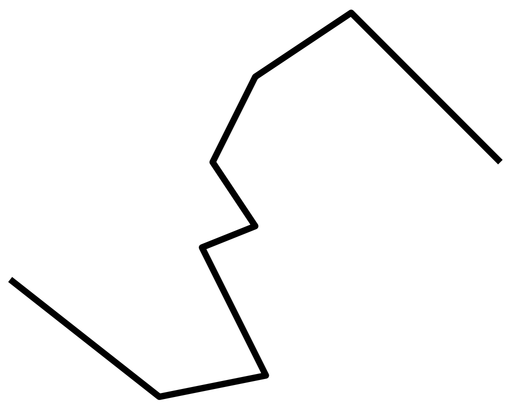
Criteria.
- minimize number of links **(R3)**

Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

Criteria.
- minimize number of links **(R3)**

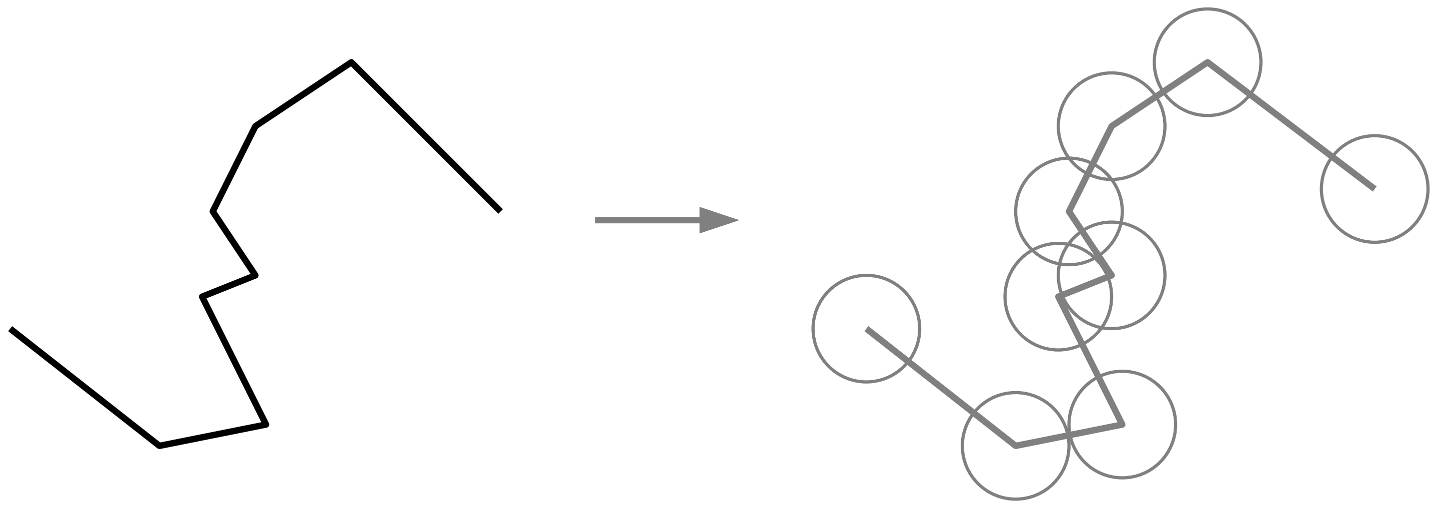


Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

Criteria.
- minimize number of links **(R3)**

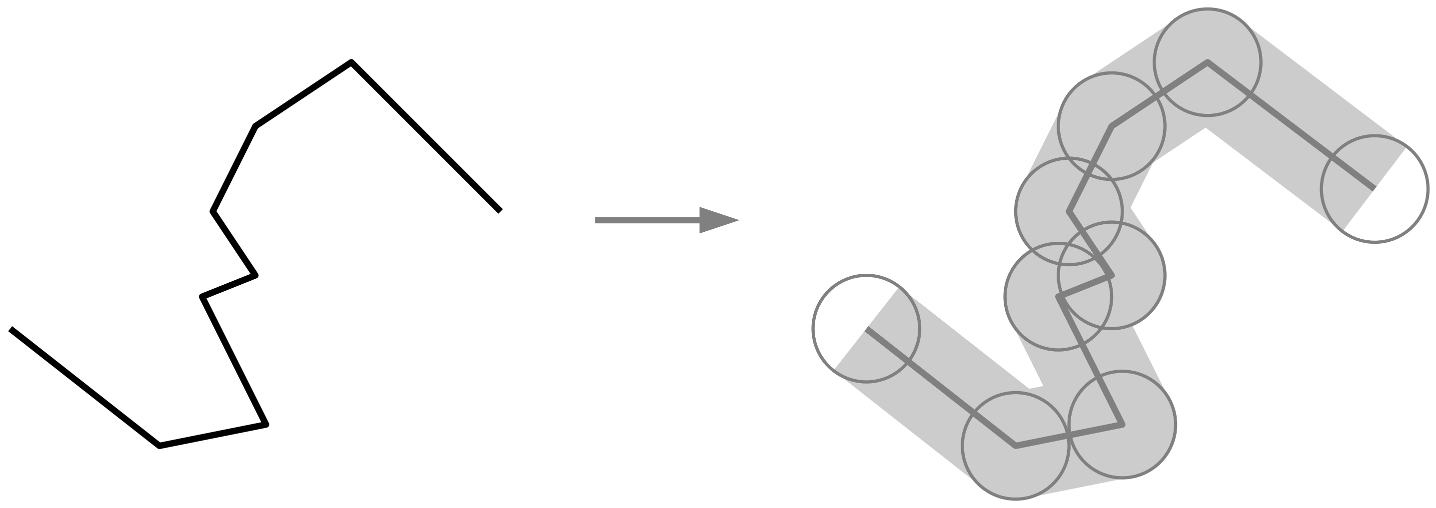


Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

Criteria.
- minimize number of links **(R3)**

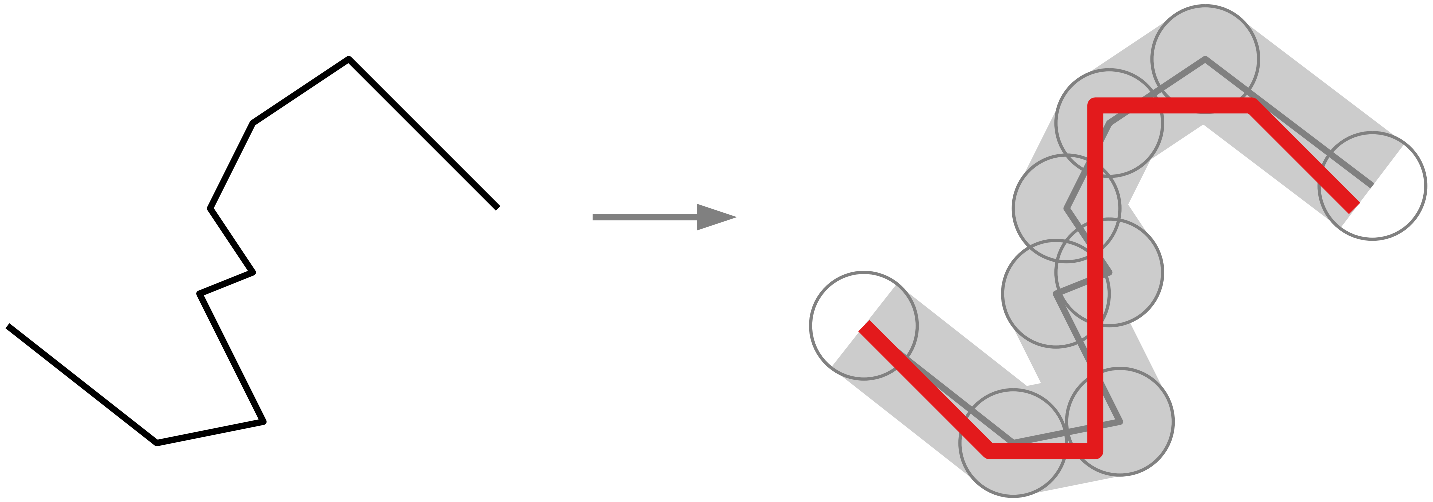


Path-Based Schematization

Goal. Solve restricted problem, where G is a path (or polyline)

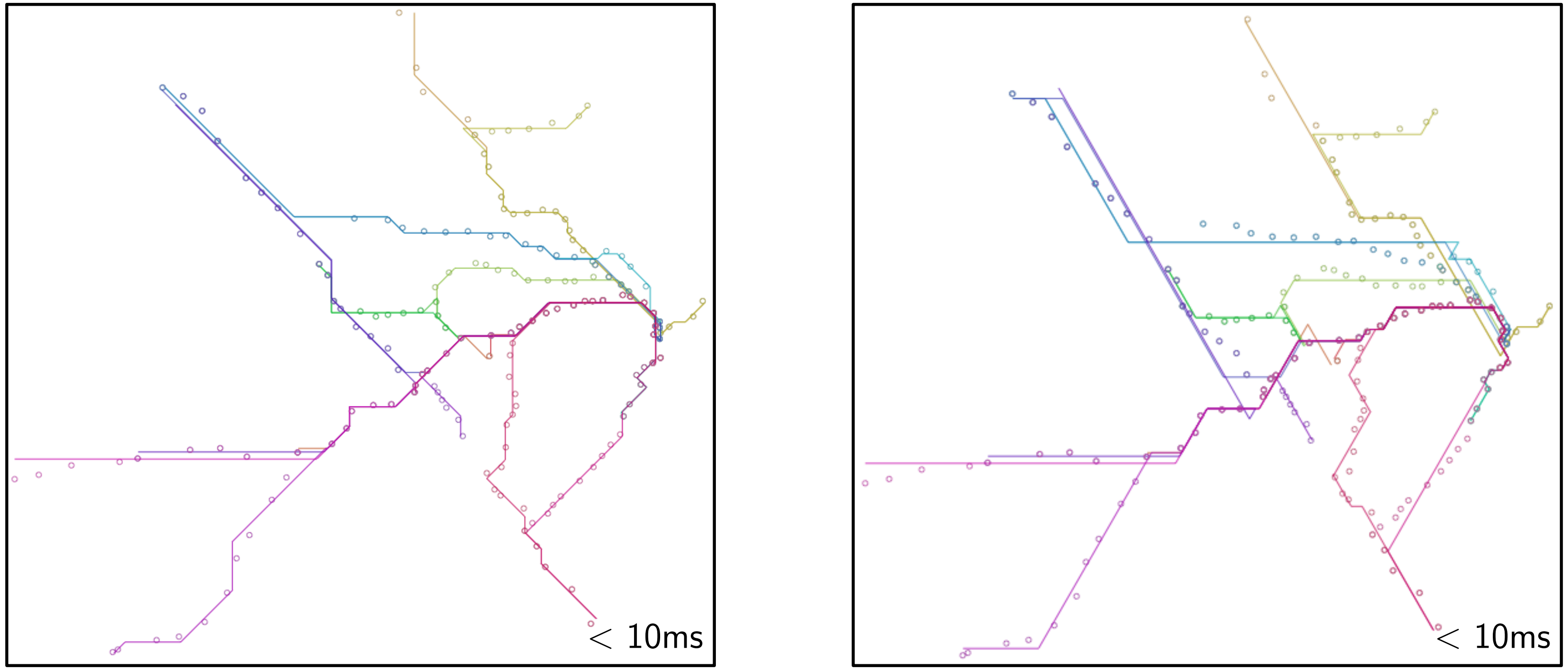
Constraints.
- $\mathcal{C}$-oriented edges (e.g. octilinear) **(R2)**
- bounded displacement **(R6)**

Criteria.
- minimize number of links **(R3)**



Path-based schematization – example

Theorem 2. [Dwyer, Hurst, Merrick '08]
For a path P of length n and orientation set $\mathcal{C}$ a $\mathcal{C}$-oriented schematized path can heuristically be fitted to the vertices in $O(|\mathcal{C}|n)$ time (or $O(|\mathcal{C}|n \log n)$) using least-squares regression.



$\mathcal{C}$-oriented Route Sketches

Theorem 3. [Delling et al. 2010]

Given a monotone path P and a set $\mathcal{C}$ of admissible edge slopes, we can compute, in $O(n^2)$+solve(LP) time, a $\mathcal{C}$-oriented schematization of P, which

- preserves the orthogonal order of P,
- has minimum slope deviation,
- has minimum total length.

$\mathcal{C}$-oriented Route Sketches

Theorem 3. [Delling et al. 2010]

Given a monotone path P and a set $\mathcal{C}$ of admissible edge slopes, we can compute, in $O(n^2)$+solve(LP) time, a $\mathcal{C}$-oriented schematization of P, which

- preserves the orthogonal order of P,
- has minimum slope deviation,
- has minimum total length.

relative north-south-east-west relationship of all vertices

$\mathcal{C}$-oriented Route Sketches

Theorem 3. [Delling et al. 2010]
Given a monotone path P and a set $\mathcal{C}$ of admissible edge slopes, we can compute, in $O(n^2)$+solve(LP) time, a $\mathcal{C}$-oriented schematization of P, which

- preserves the orthogonal order of P,
- has minimum slope deviation,
- has minimum total length.

relative north-south-east-west relationship of all vertices

Proof.
- dynamic programming for slope assignment
- LP for length assignment

$\mathcal{C}$-oriented Route Sketches

Theorem 3. [Delling et al. 2010]
Given a monotone path P and a set $\mathcal{C}$ of admissible edge slopes, we can compute, in $O(n^2)$+solve(LP) time, a $\mathcal{C}$-oriented schematization of P, which

- preserves the orthogonal order of P,
- has minimum slope deviation,
- has minimum total length.

relative north-south-east-west relationship of all vertices

Proof.
- dynamic programming for slope assignment
- LP for length assignment

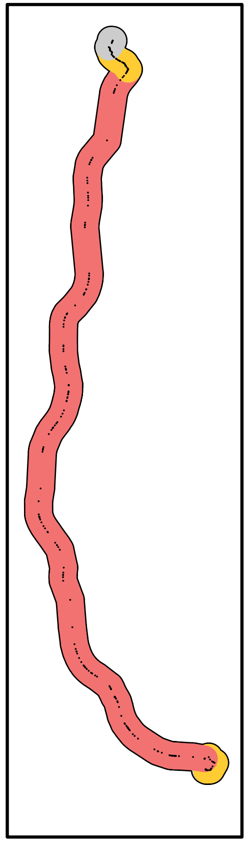
Theorem 4. [Brandes & Pampel 2009, Gemsa et al. 2011]
The d-regular (non-monotone) route sketch problem is NP-hard for any $d \geq 1$, where $\mathcal{C} = \{i \cdot 90°/d \mid i \in \mathbb{Z}\}$.

$\mathcal{C}$-oriented Route Sketches

Example. Bremen to Cuxhaven [Gemsa et al. 2011]

$\mathcal{C}$-oriented Route Sketches

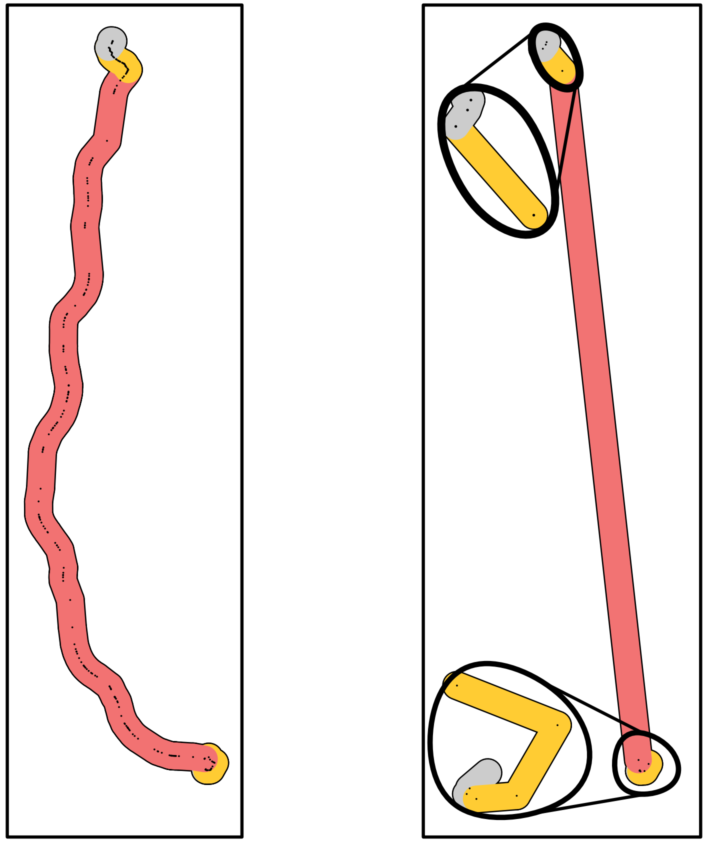
Example. Bremen to Cuxhaven [Gemsa et al. 2011]



input

$\mathcal{C}$-oriented Route Sketches

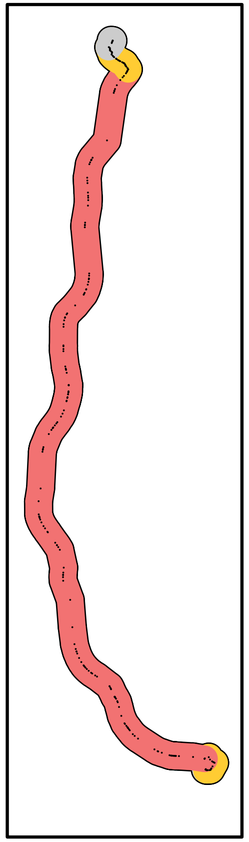
Example. Bremen to Cuxhaven [Gemsa et al. 2011]



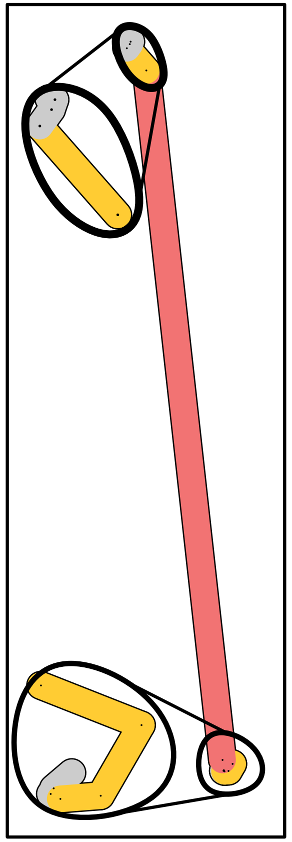
input simplified

$\mathcal{C}$-oriented Route Sketches

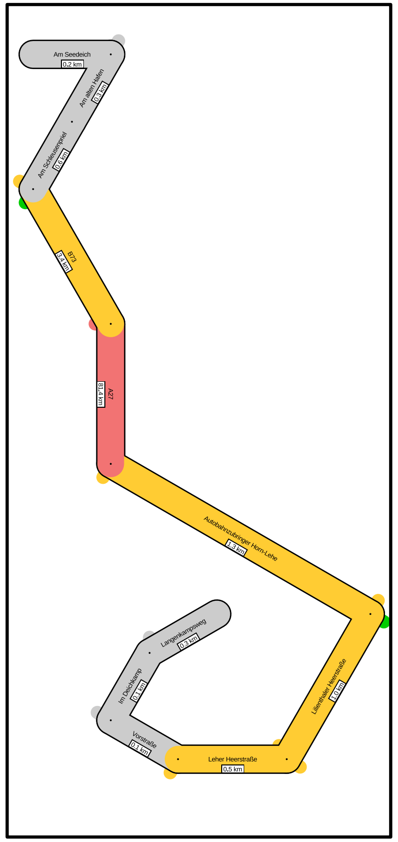
Example. Bremen to Cuxhaven [Gemsa et al. 2011]



input



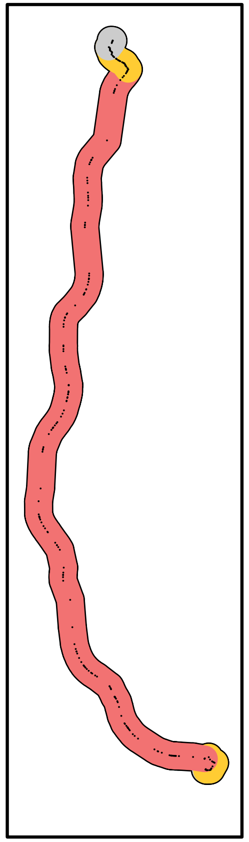
simplified



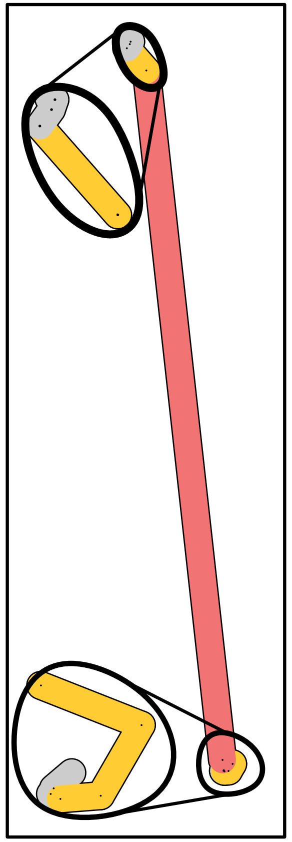
optimized by MIP

$\mathcal{C}$-oriented Route Sketches

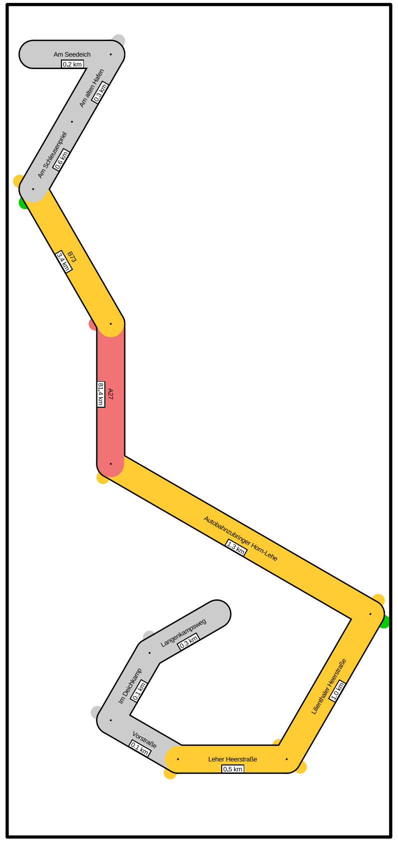
Example. Bremen to Cuxhaven [Gemsa et al. 2011]



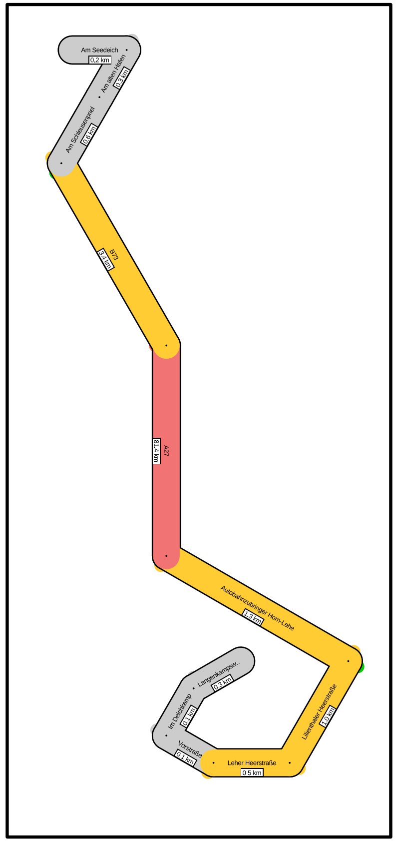
input



simplified



optimized by MIP



including length order constraint

Path-Based Schematization – Discussion

Pros.

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:
  - decompose metro network into paths

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:
  - decompose metro network into paths
  - schematize individual paths

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:
  - decompose metro network into paths
  - schematize individual paths
  - glue schematized paths together at interchanges

Cons.

Path-Based Schematization – Discussion

Pros.

- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:
  - decompose metro network into paths
  - schematize individual paths
  - glue schematized paths together at interchanges

Cons.

- no guarantee on network topology **(R1)**

Path-Based Schematization – Discussion

Pros.

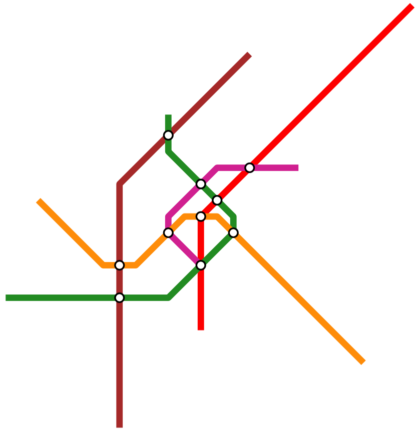
- polynomial running times
- $\mathcal{C}$-orientation and bounded displacement guaranteed
- bend minimization
- extends to metro networks:
  - decompose metro network into paths
  - schematize individual paths
  - glue schematized paths together at interchanges

Cons.

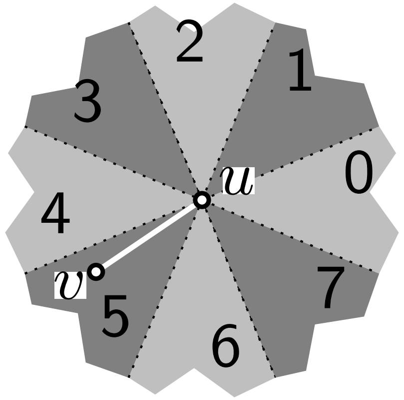
- no guarantee on network topology **(R1)**
- distortion/displacement too limited for metro maps

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

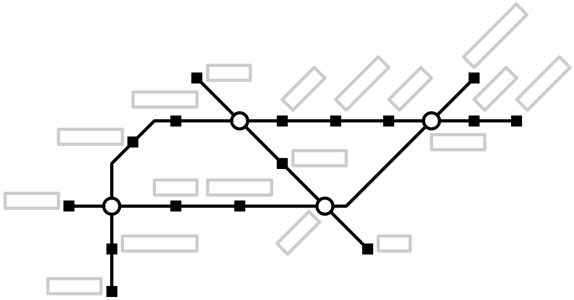
Visualization of Graphs



Lecture 12:
Octilinear Graph Drawing
Metro Map Layout



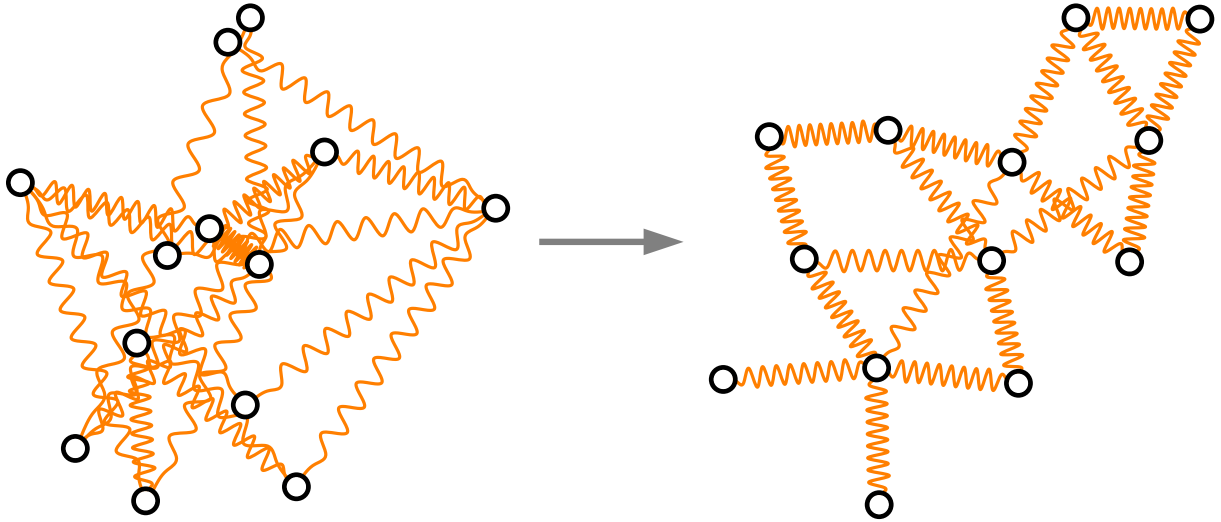
Part III:
Force-Based Schematization



Jonathan Klawitter

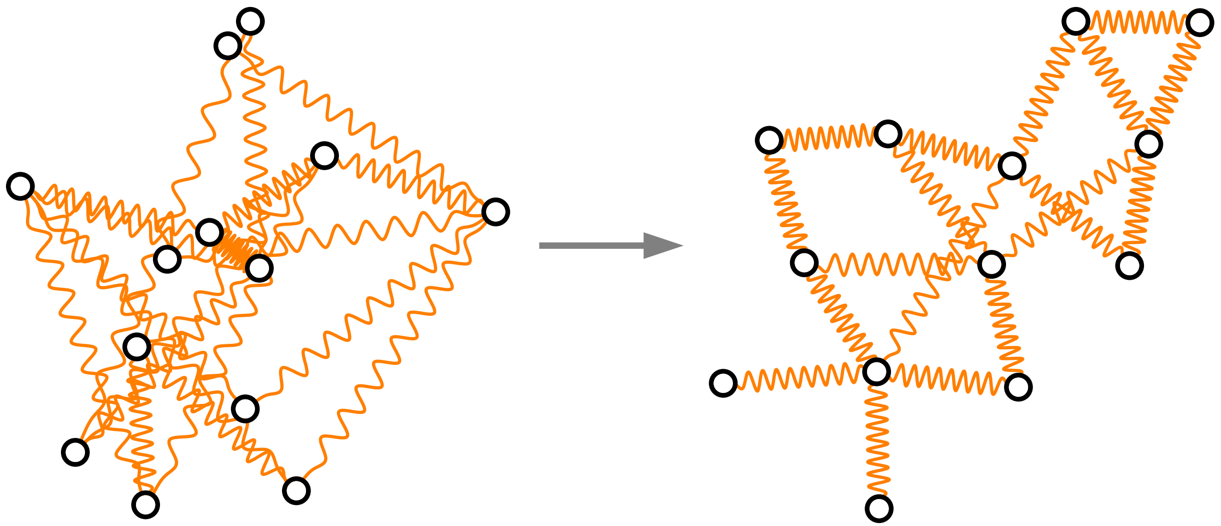
Force-Based Schematization

Idea. Apply well known force-based graph drawing approach



Force-Based Schematization

Idea. Apply well known force-based graph drawing approach

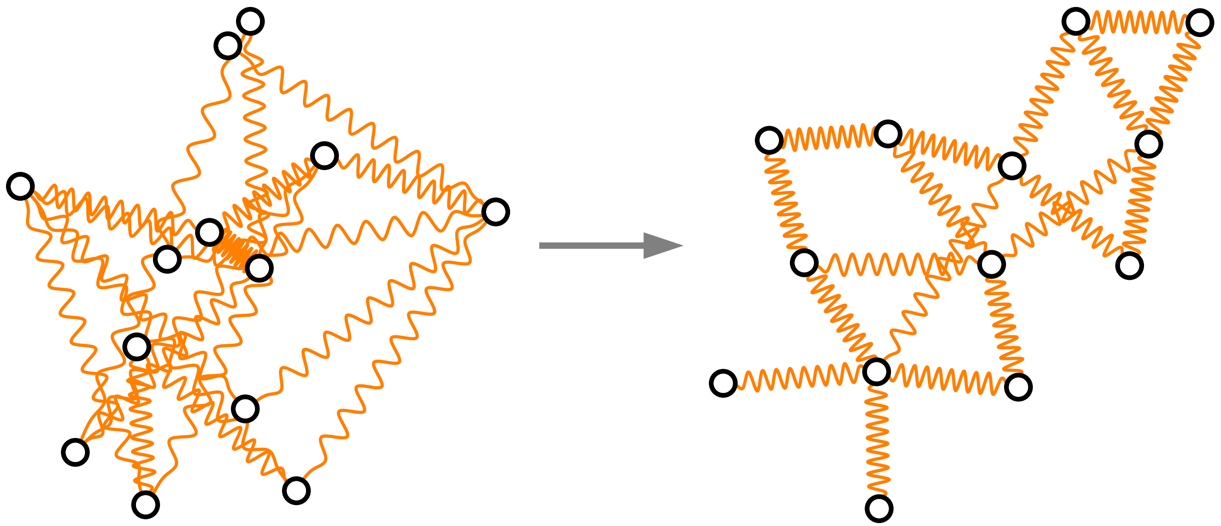


Recall.

- vertices are charged particles repelling each other

Force-Based Schematization

Idea. Apply well known force-based graph drawing approach

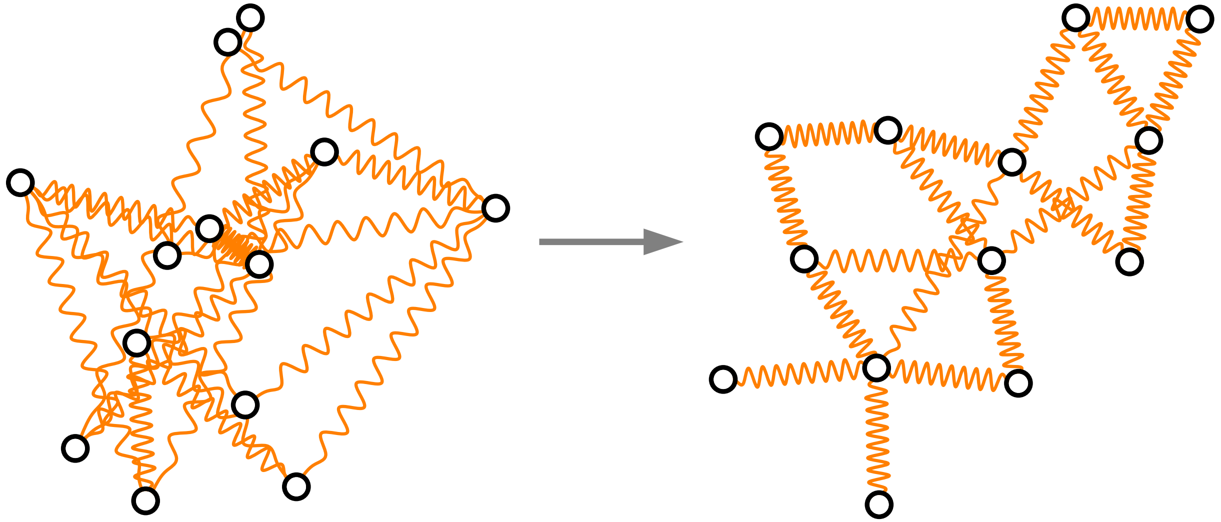


Recall.

- vertices are charged particles repelling each other
- edges are springs pulling edges into target length

Force-Based Schematization

Idea. Apply well known force-based graph drawing approach

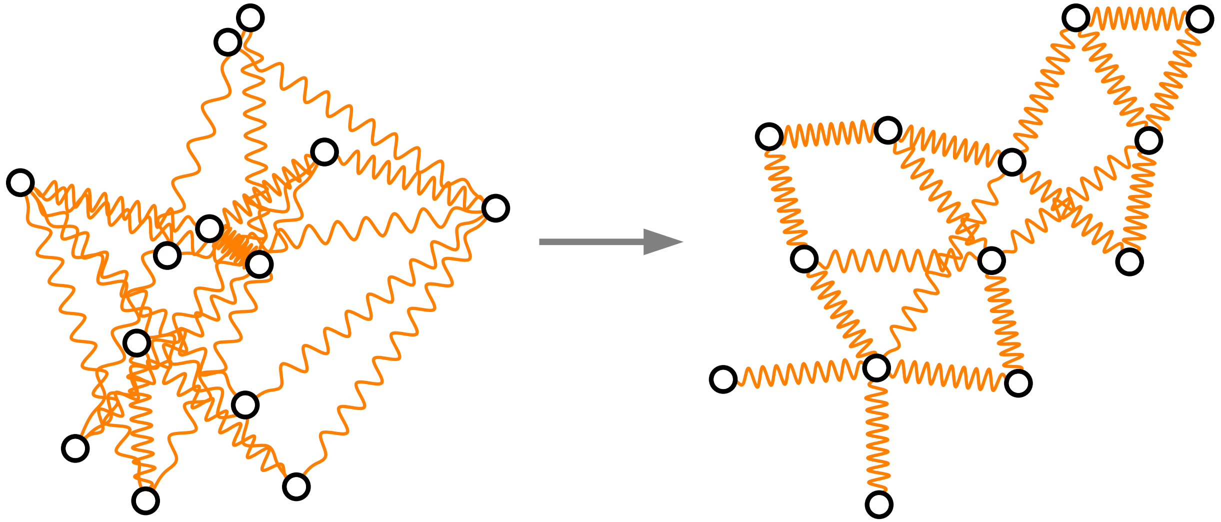


Recall.

- vertices are charged particles repelling each other
- edges are springs pulling edges into target length
- iteratively calculate and apply forces until system stabilizes

Force-Based Schematization

Idea. Apply well known force-based graph drawing approach



Recall.

- vertices are charged particles repelling each other
- edges are springs pulling edges into target length
- iteratively calculate and apply forces until system stabilizes

$\rightarrow$ define additional forces to model
subset of metro map design rules

Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**

Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**

Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**

Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**
- spring lengths model uniform edge lengths **(R7)**

Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**
- spring lengths model uniform edge lengths **(R7)**
- vertex repulsion models feature separation **(R8)**

Force-Based Schematization – Octilinear

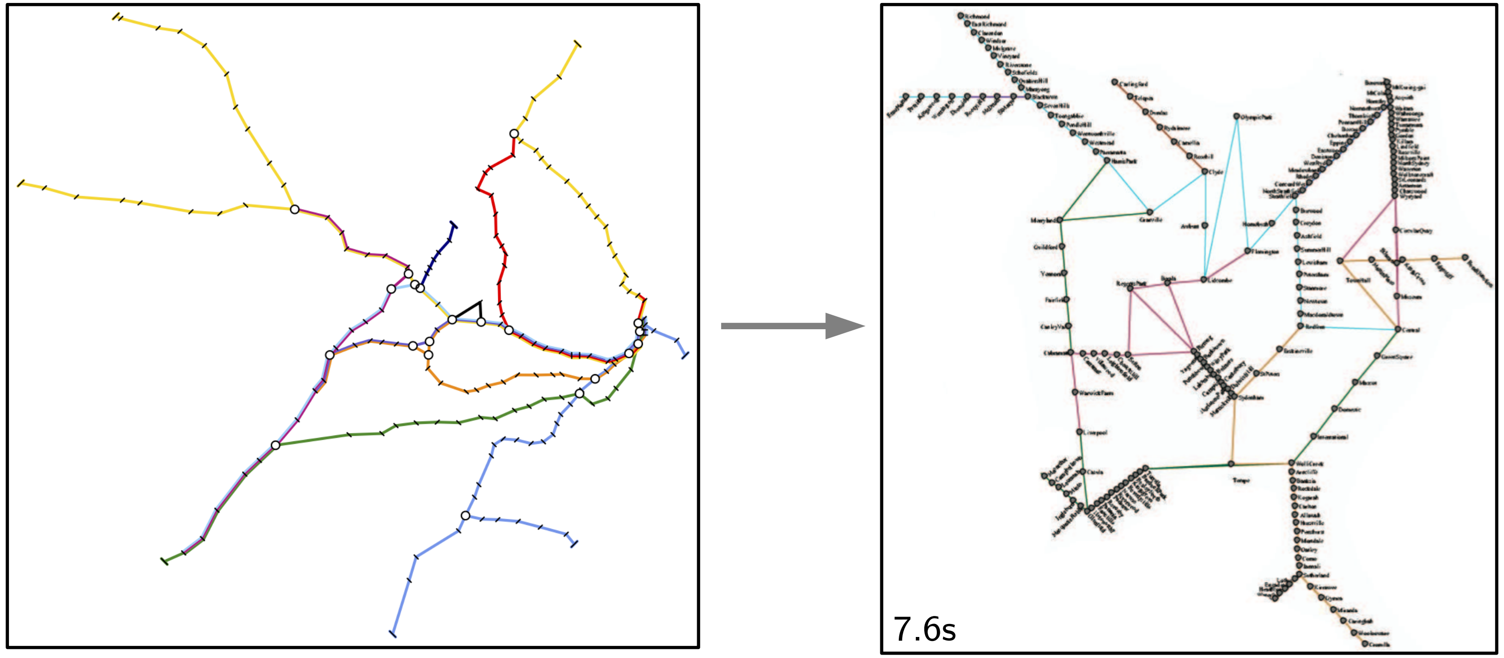
[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**
- spring lengths model uniform edge lengths **(R7)**
- vertex repulsion models feature separation **(R8)**
- station labels placed in independent 2nd step

Force-Based Schematization – Octilinear

[Hong et al. 2006]

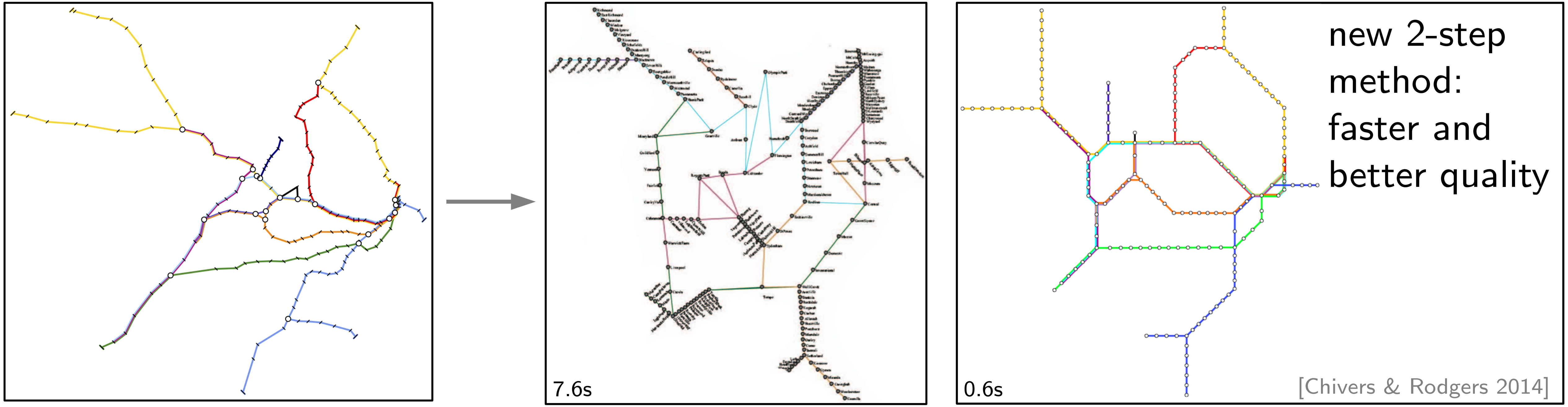
- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**
- spring lengths model uniform edge lengths **(R7)**
- vertex repulsion models feature separation **(R8)**
- station labels placed in independent 2nd step



Force-Based Schematization – Octilinear

[Hong et al. 2006]

- contract degree-2 vertices into weighted edges **(R3)**
- define octilinear magnetic field attracting edges **(R2)**
- only apply topology preserving vertex moves **(R1)**
- spring lengths model uniform edge lengths **(R7)**
- vertex repulsion models feature separation **(R8)**
- station labels placed in independent 2nd step



Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces improving curve shape (low curvature) **(R3)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces improving curve shape (low curvature) **(R3)**
- merge curves whenever possible **(R3)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces improving curve shape (low curvature) **(R3)**
- merge curves whenever possible **(R3)**
- forces improving angular resolution **(R5)**

Force-Based Schematization – Bézier

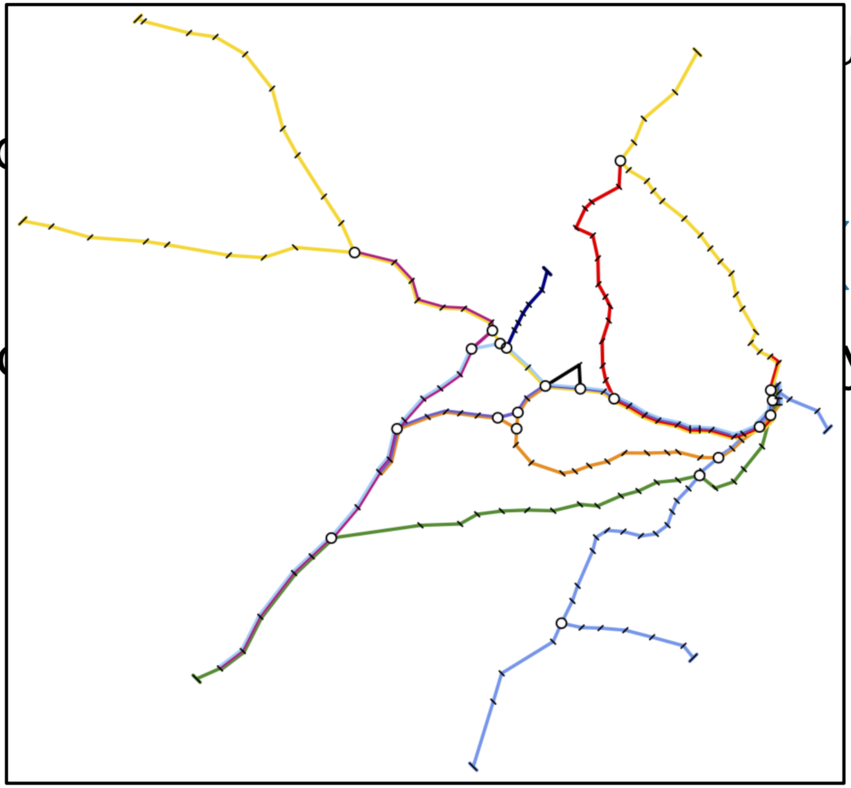
[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces improving curve shape (low curvature) **(R3)**
- merge curves whenever possible **(R3)**
- forces improving angular resolution **(R5)**
- apply topology-preserving moves only **(R1)**

Force-Based Schematization – Bézier

[Fink et al. 2013]

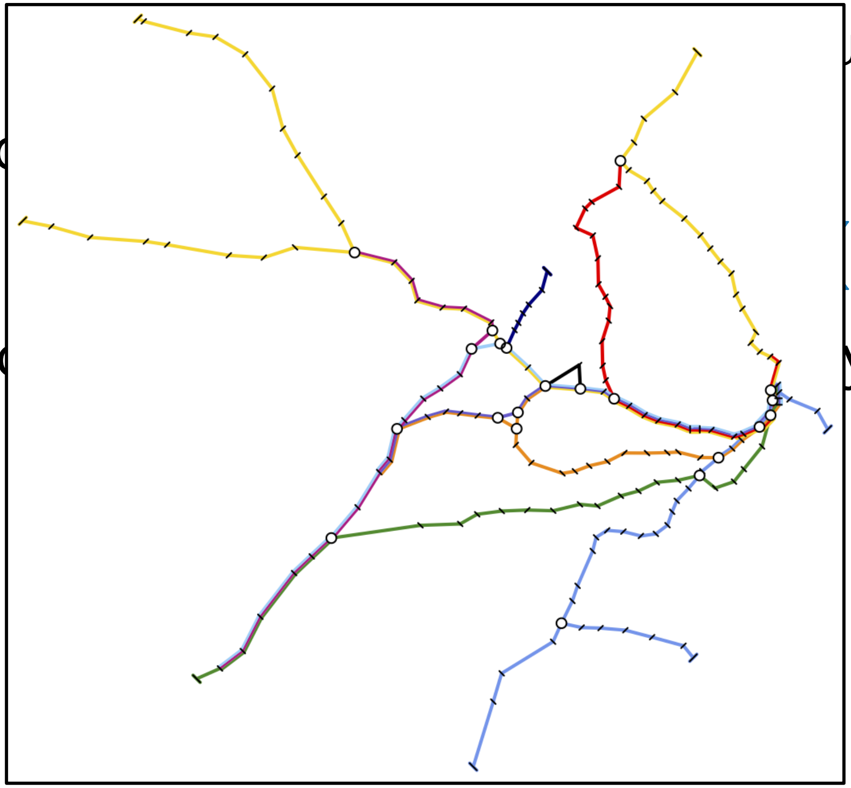
- convert (octilinear) input layout into Bézier curves
  $\rightarrow$ vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces i urvature) **(R3)**
- merge c)
- forces i **(R5)**
- apply t y **(R1)**

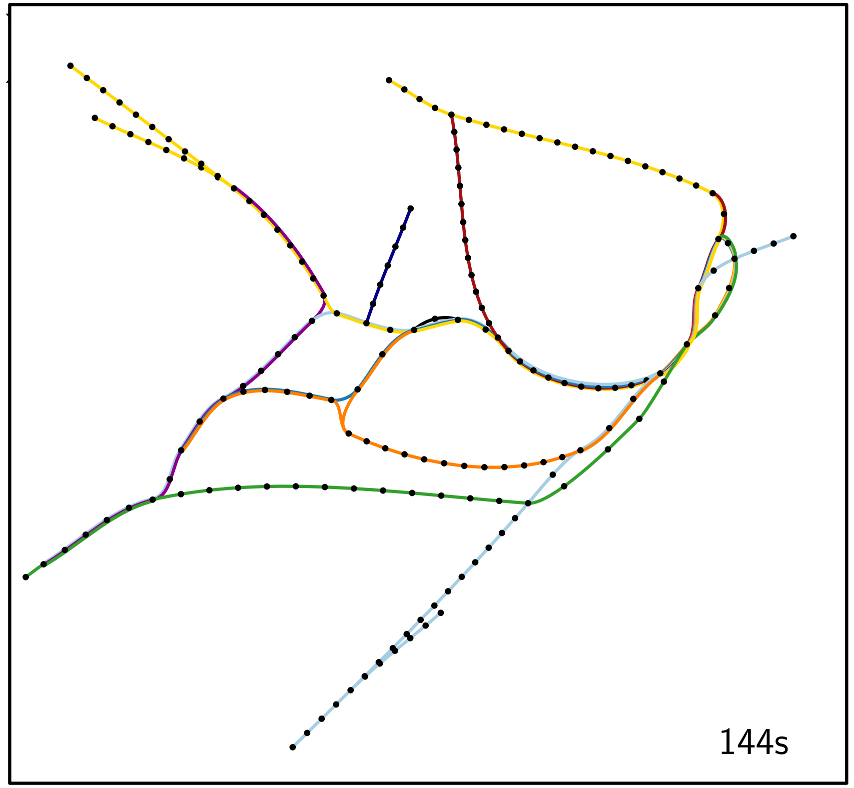


Force-Based Schematization – Bézier

[Fink et al. 2013]

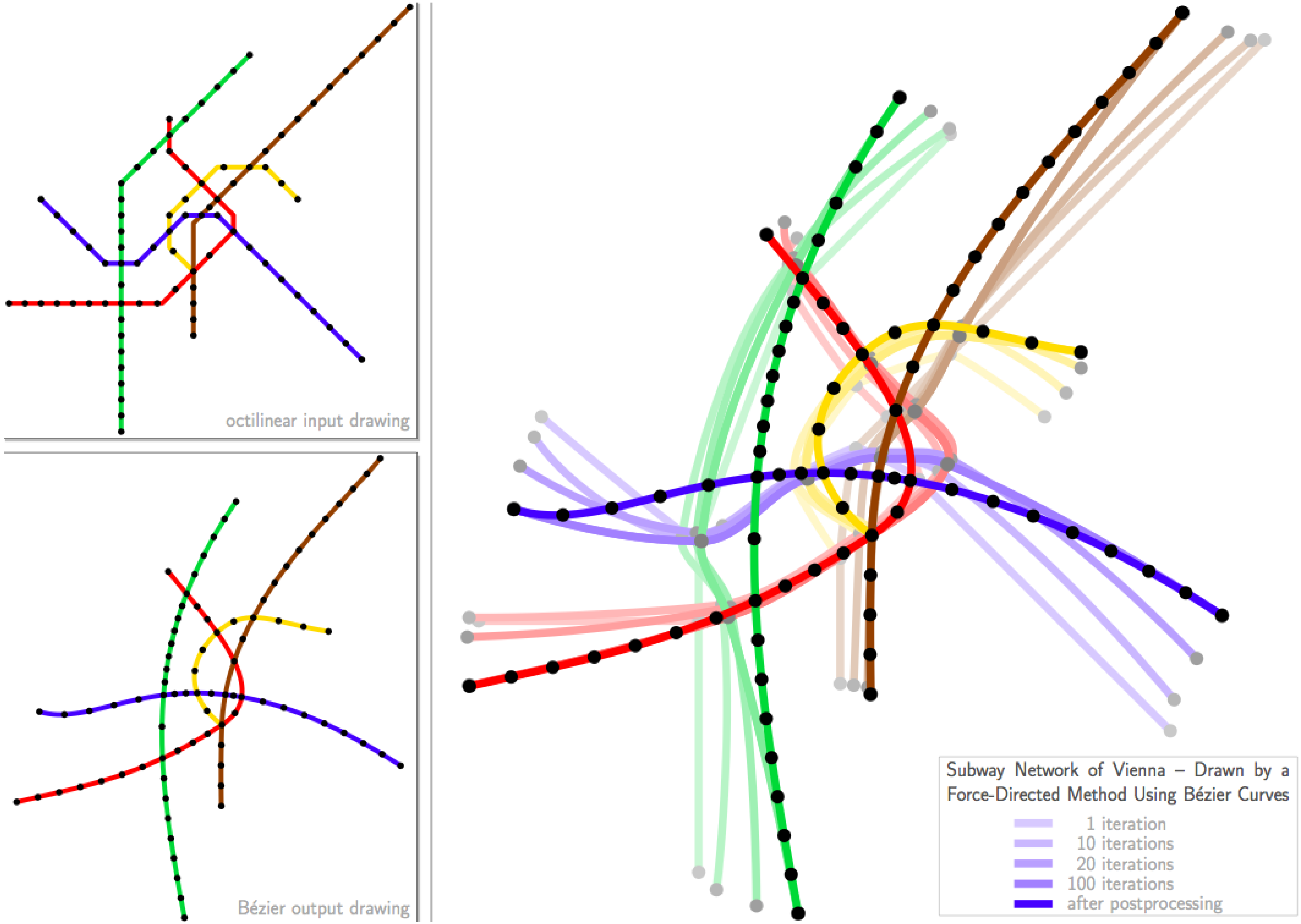
- convert (octilinear) input layout into Bézier curves
  → vertices and control points, both subject to forces
- metro line subcurves share tangents in common vertex **(R4)**
- (standard) attractive and repulsive forces **(R7)+(R8)**
- (weak) force towards initial position **(R6)**
- forces i urvature)
- merge c **)**
- forces i **(R5)**
- apply t y **(R1)**





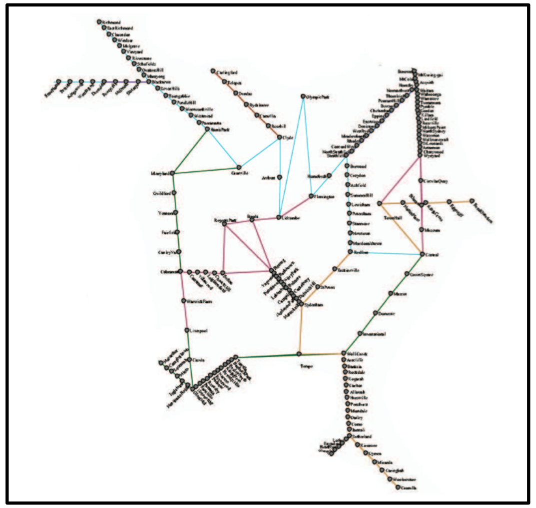
Force-Based Schematization – Bézier

[Fink et al. 2013]



Force-Based Schematization – Discussion

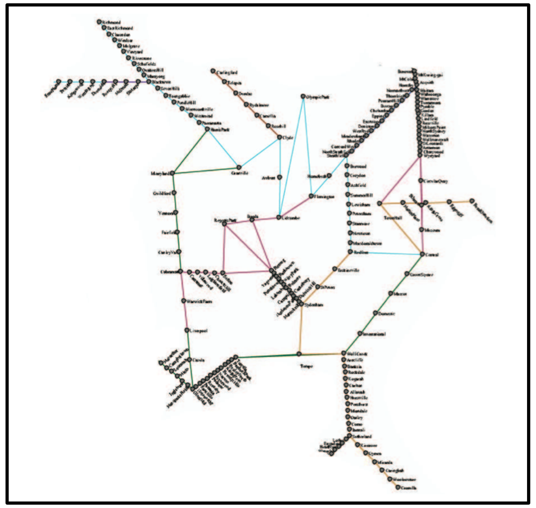
Octilinear.



Force-Based Schematization – Discussion

Octilinear.

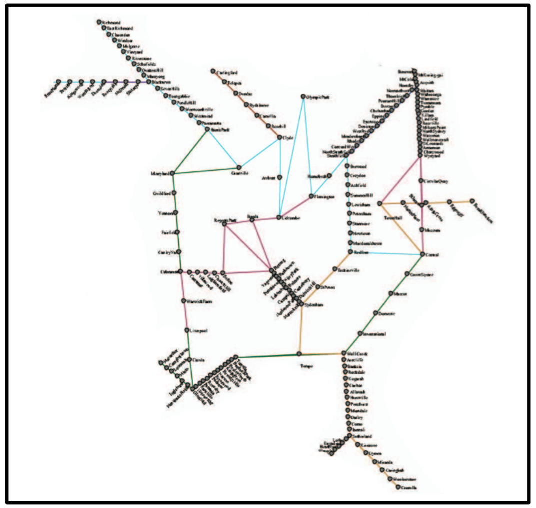
- guarantees topology **(R1)**



Force-Based Schematization – Discussion

Octilinear.

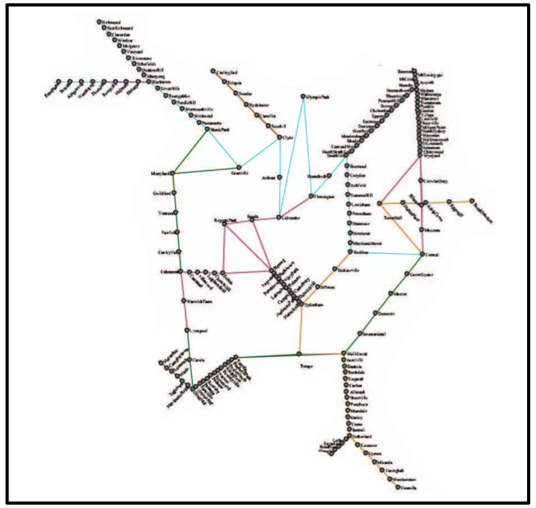
- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough



Force-Based Schematization – Discussion

Octilinear.

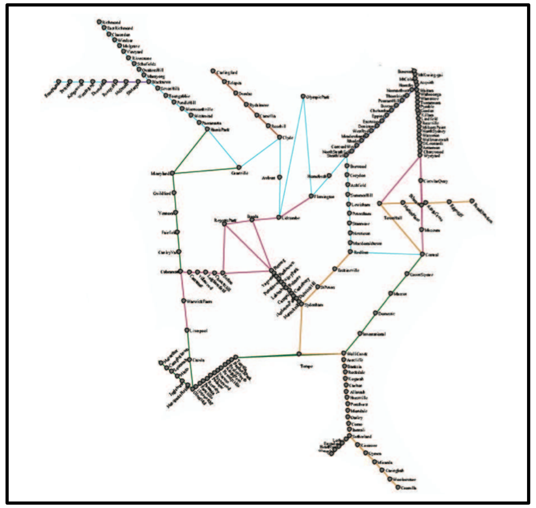
- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity



Force-Based Schematization – Discussion

Octilinear.

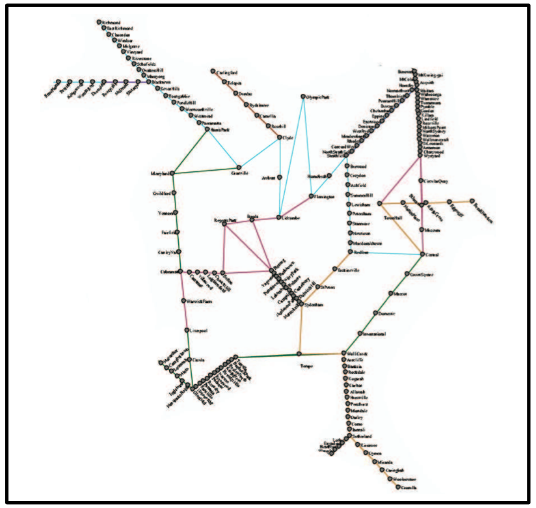
- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths



Force-Based Schematization – Discussion

Octilinear.

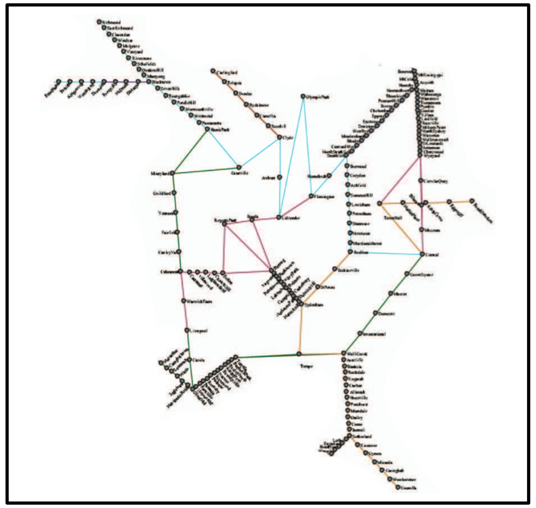
- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges



Force-Based Schematization – Discussion

Octilinear.

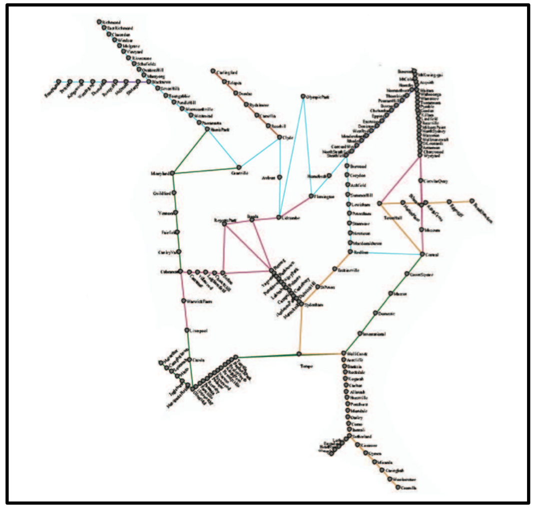
- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



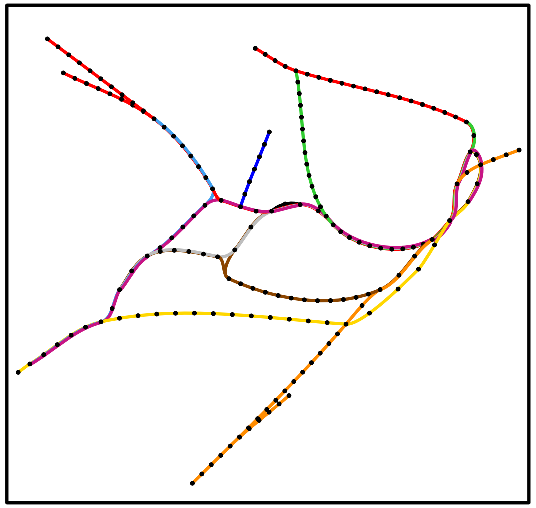
Force-Based Schematization – Discussion

Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



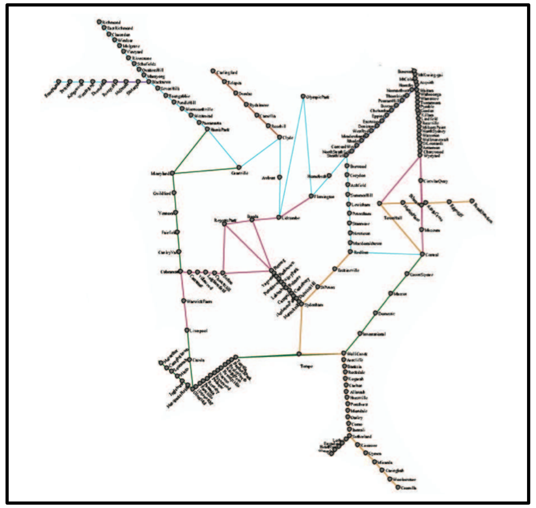
Bézier.



Force-Based Schematization – Discussion

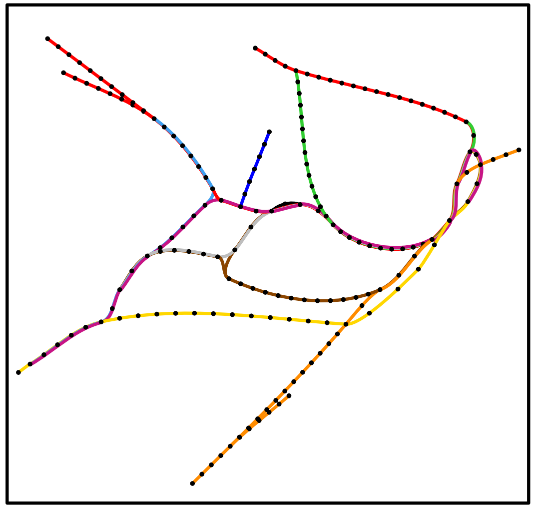
Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



Bézier.

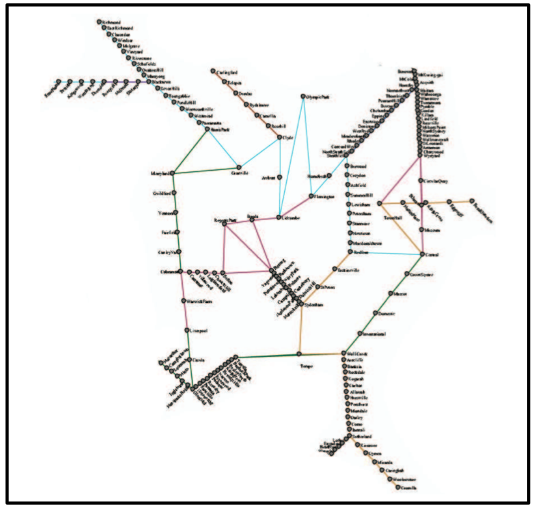
- guarantees topology **(R1)**



Force-Based Schematization – Discussion

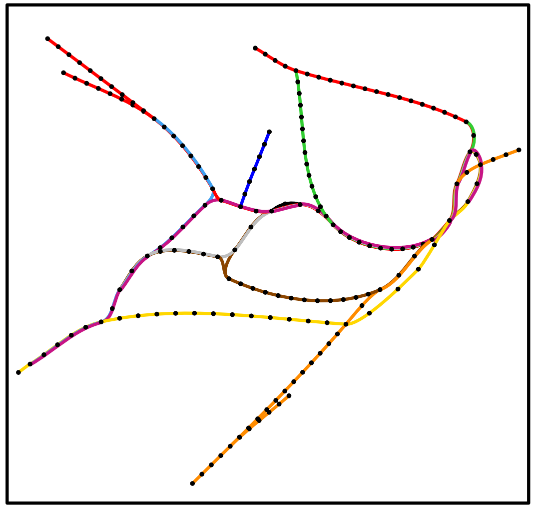
Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



Bézier.

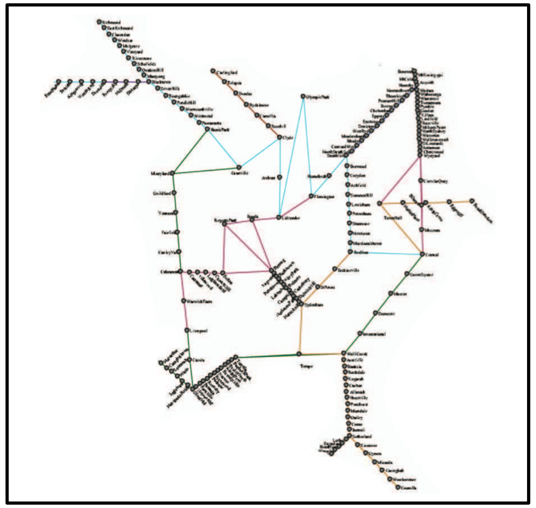
- guarantees topology **(R1)**
- takes almost all design rules into account



Force-Based Schematization – Discussion

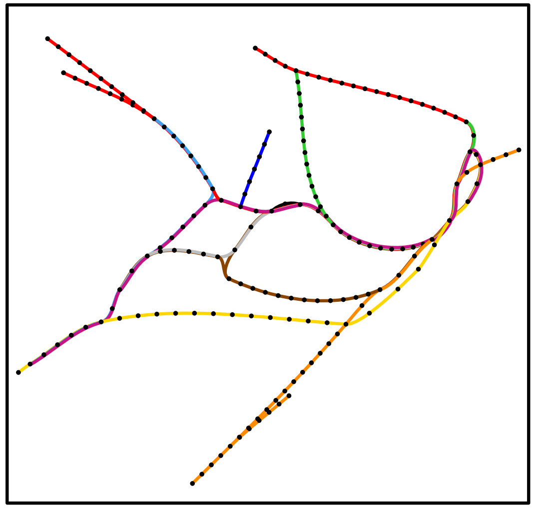
Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



Bézier.

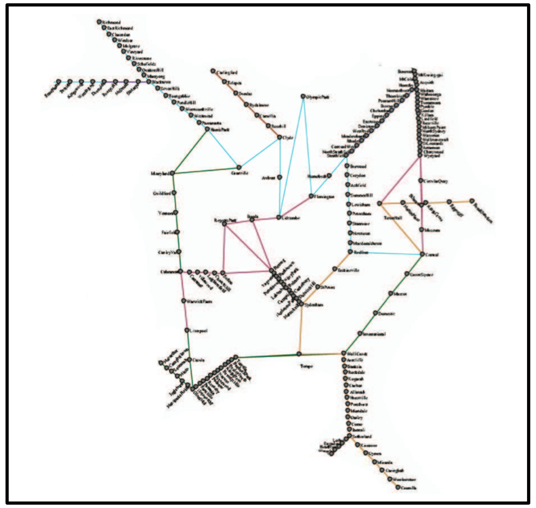
- guarantees topology **(R1)**
- takes almost all design rules into account
- first curvilinear metro map algorithm



Force-Based Schematization – Discussion

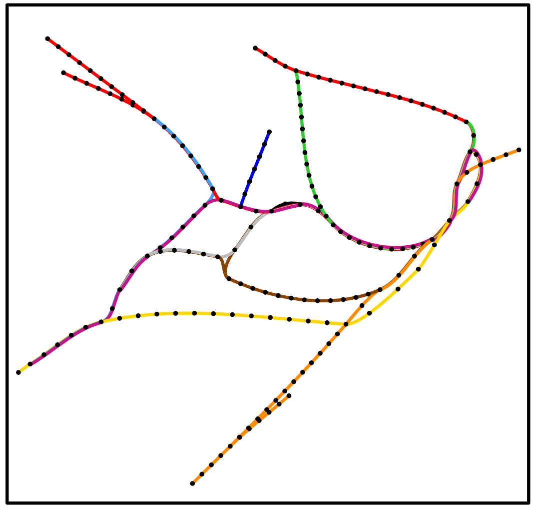
Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



Bézier.

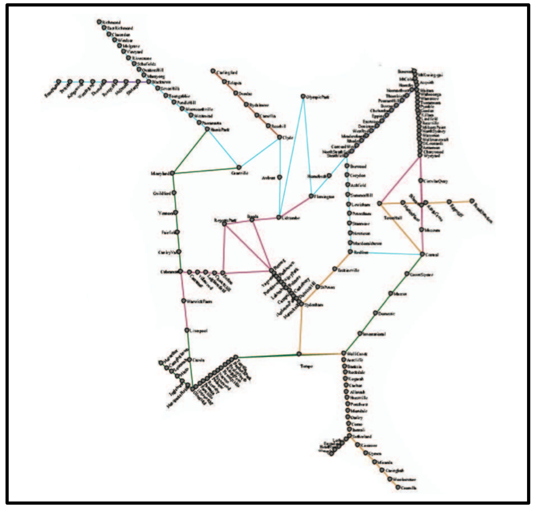
- guarantees topology **(R1)**
- takes almost all design rules into account
- first curvilinear metro map algorithm
- works well on small and medium instances



Force-Based Schematization – Discussion

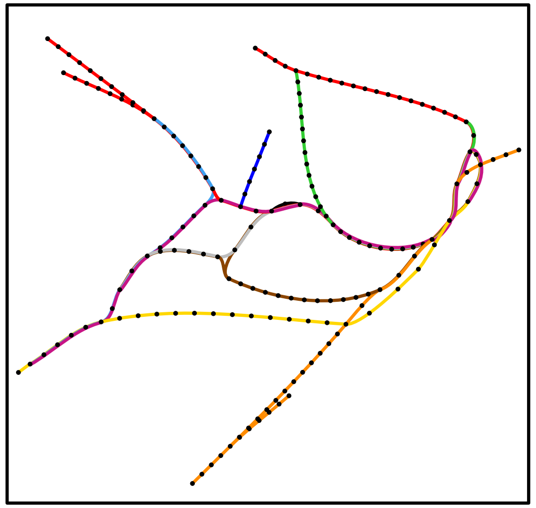
Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity
- quite unbalanced edge lengths
- bends in interchanges
- no distortion restriction



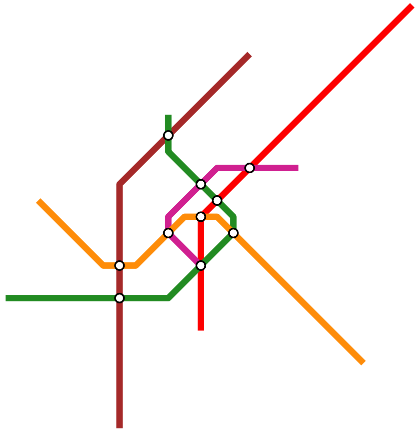
Bézier.

- guarantees topology **(R1)**
- takes almost all design rules into account
- first curvilinear metro map algorithm
- works well on small and medium instances
- difficulties with more complex networks

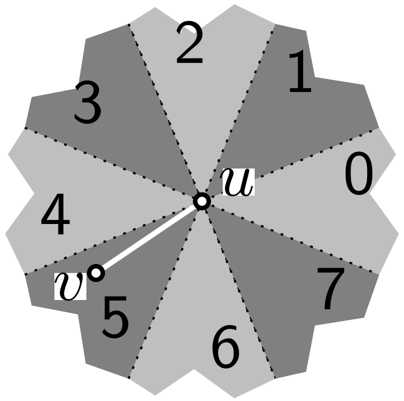


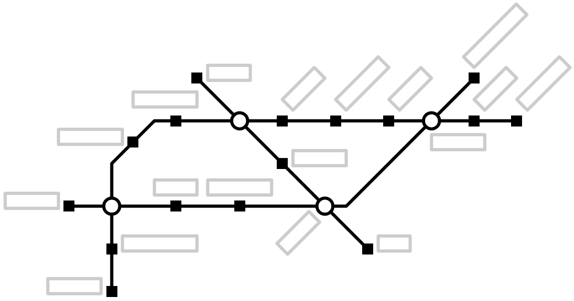
Visualization of Graphs

Lecture 12:
Octilinear Graph Drawing
Metro Map Layout



Part IV:
Local Schematization





Jonathan Klawitter

Local Schematization

Idea.

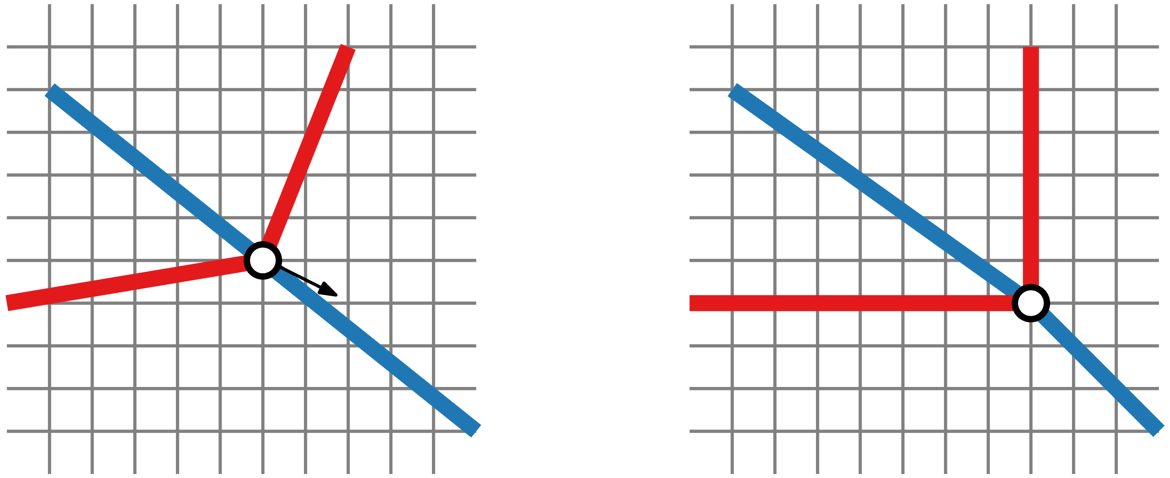
- define a (multi-criteria) layout quality function

es

Local Schematization

Idea.

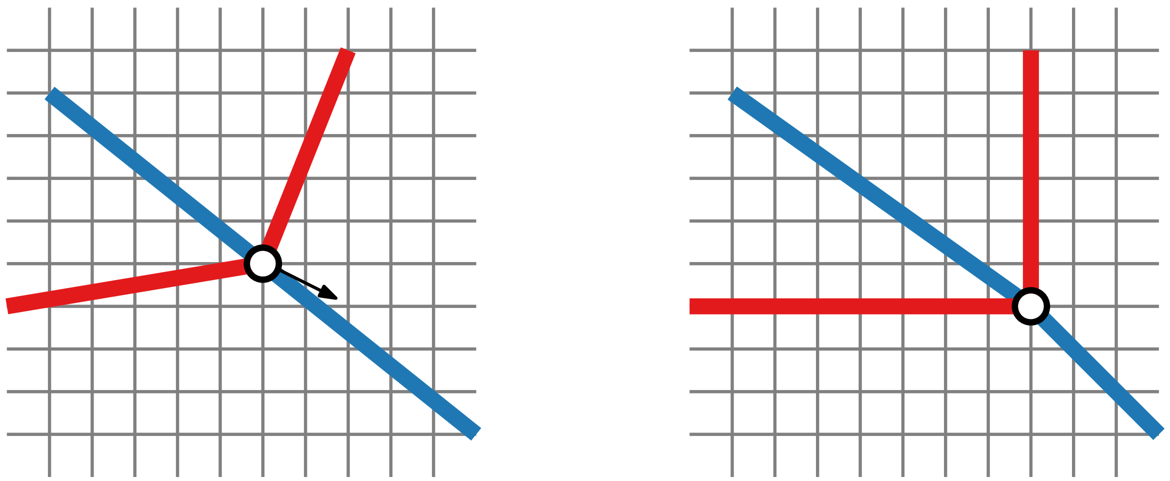
- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions



Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .



Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)

Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**

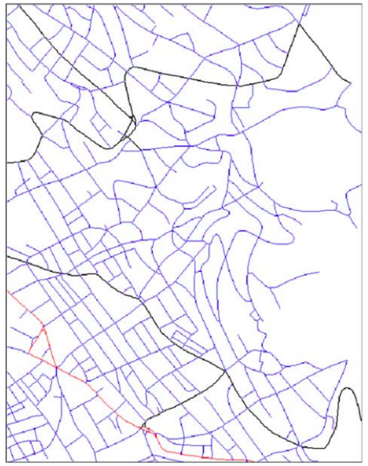
Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**



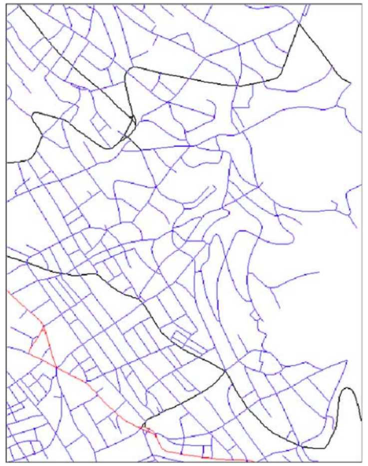
Local Schematization

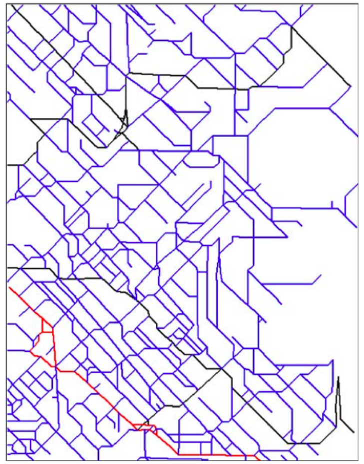
Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**





100

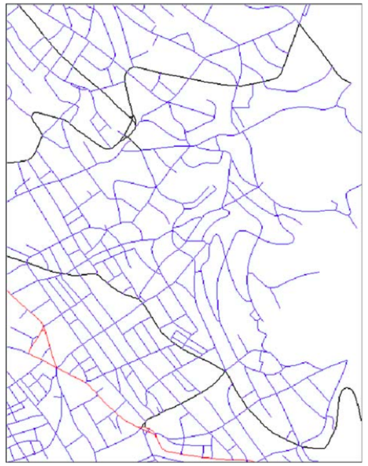
Local Schematization

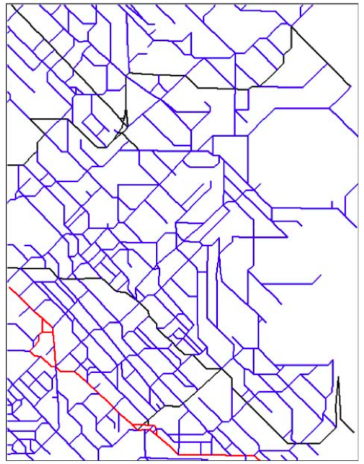
Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, ...

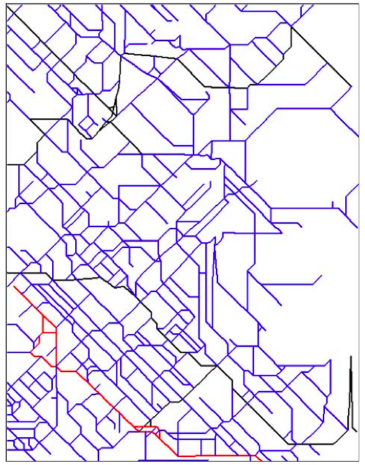
[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**





100



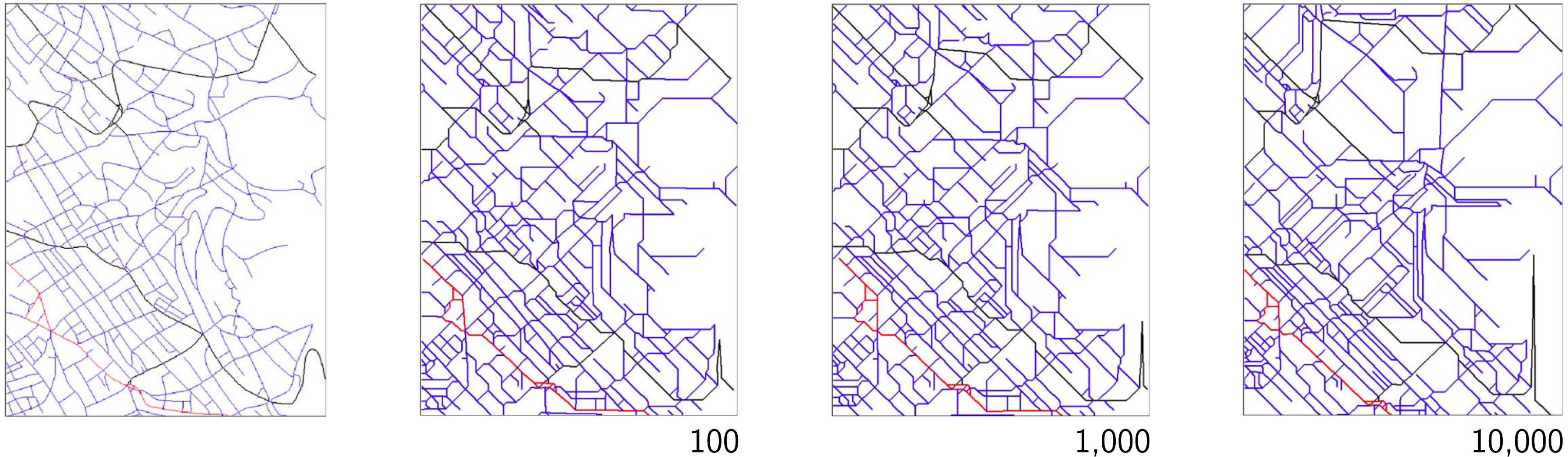
1,000

Local Schematization

Idea.
- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Avelar & Müller 2000]

- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**



Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, ...

[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function

Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function
- contract degree-2 vertices prior to optimization

Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, ...

[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function
- contract degree-2 vertices prior to optimization
- implemented more design rules (topology **(R1)**, octilinearity **(R2)**, displacement **(R6)**, edge lengths **(R7)**, separation **(R8)**)

Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .

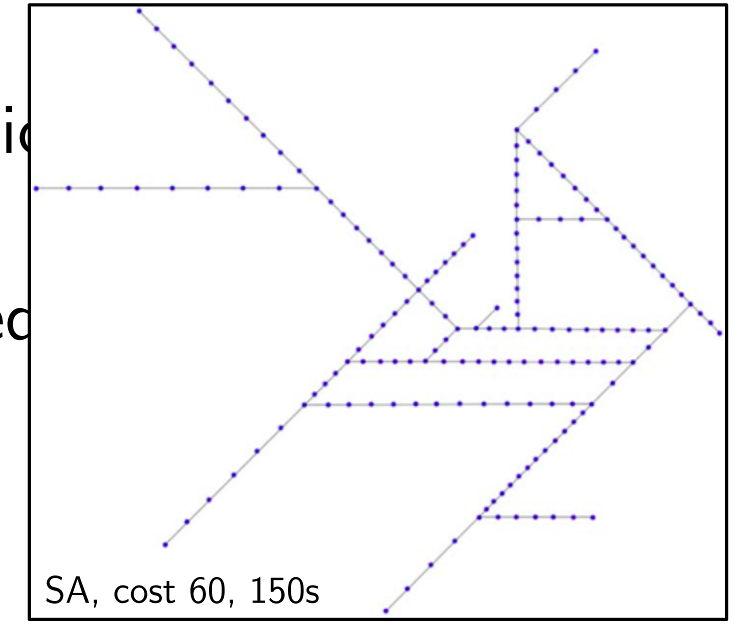
[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function
- contract degree-2 vertices prior to optimization
- implemented more design rules (topology **(R1)**, octilinearity **(R2)**, displacement **(R6)**, edge lengths **(R7)**, separation **(R8)**)
- simulated annealing (2006) and ant colony optimization (2013)

Local Schematization

Idea.

- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving verti[ces] to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, . . .


SA, cost 60, 150s

[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function
- contract degree-2 vertices prior to optimization
- implemented more design rules (topology **(R1)**, octilinearity **(R2)**, displacement **(R6)**, edge lengths **(R7)**, separation **(R8)**)
- simulated annealing (2006) and ant colony optimization (2013)

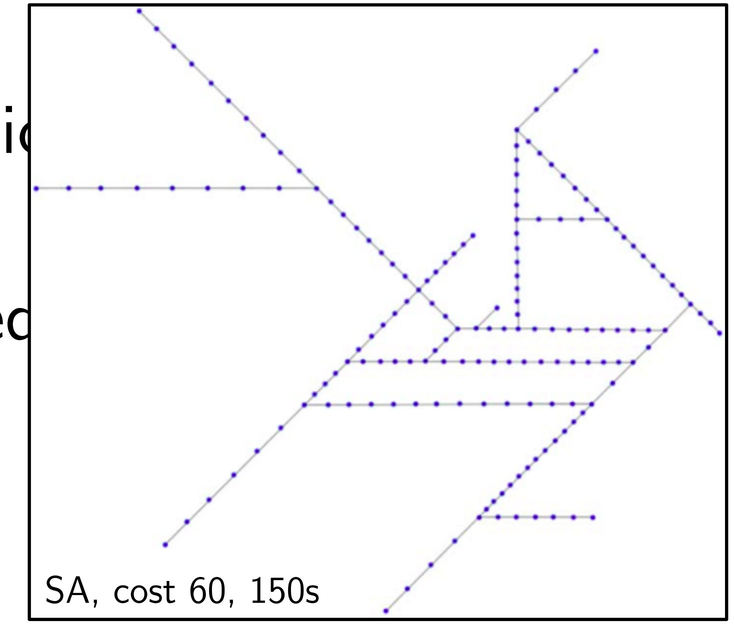
Local Schematization

Idea.

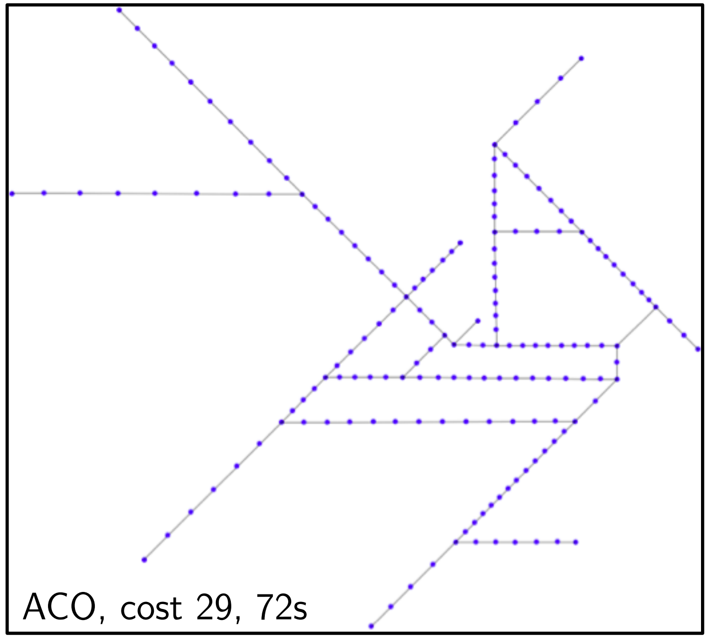
- define a (multi-criteria) layout quality function
- improve layout quality step-by-step by locally moving verti[ces] to better nearby (grid) positions
- applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, ...

[Ware et al. 2006, Ware & Richards 2013]

- weighted multicriteria function
- contract degree-2 vertices prior to optimization
- implemented more design rules (topology **(R1)**, octilinearity **(R2)**, displacement **(R6)**, edge lengths **(R7)**, separation **(R8)**)
- simulated annealing (2006) and ant colony optimization (2013)



SA, cost 60, 150s



ACO, cost 29, 72s

Local Schematization

[Stott et al. 2011]

Idea.

- design rules as before

Local Schematization

[Stott et al. 2011]

Idea.

- design rules as before
- additionally include metro map specific criteria (bend minimization **(R3)**, interchange straightness **(R4)**, angular resolution **(R5)**, relative positions **(R6)**)

Local Schematization

[Stott et al. 2011]

Idea.

- design rules as before
- additionally include metro map specific criteria (bend minimization **(R3)**, interchange straightness **(R4)**, angular resolution **(R5)**, relative positions **(R6)**)
- integrate alternating label placement rounds

Local Schematization

[Stott et al. 2011]

Idea.

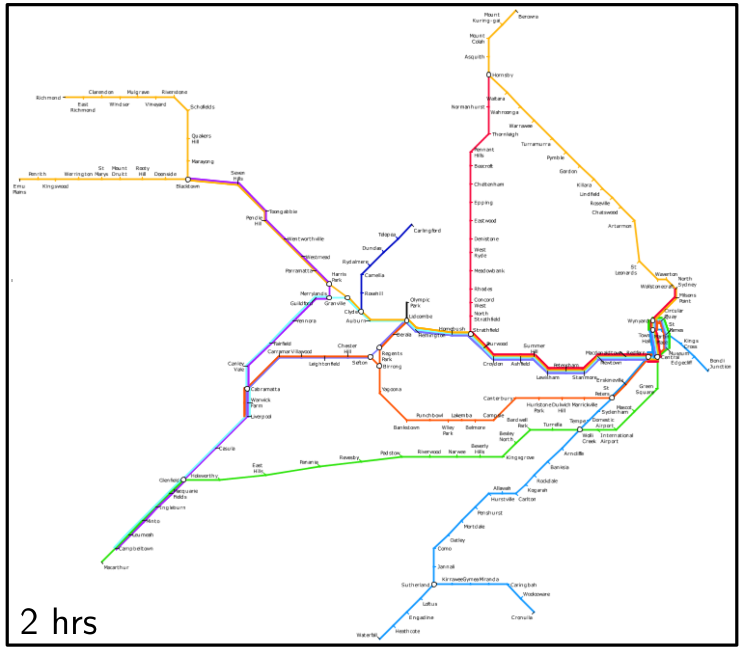
- design rules as before
- additionally include metro map specific criteria (bend minimization **(R3)**, interchange straightness **(R4)**, angular resolution **(R5)**, relative positions **(R6)**)
- integrate alternating label placement rounds
- some ad-hoc fixes for local minima situations

Local Schematization

[Stott et al. 2011]

Idea.

- design rules as before
- additionally include metro map specific criteria (bend minimization **(R3)**, interchange straightness **(R4)**, angular resolution **(R5)**, relative positions **(R6)**)
- integrate alternating label placement rounds
- some ad-hoc fixes for local minima situations

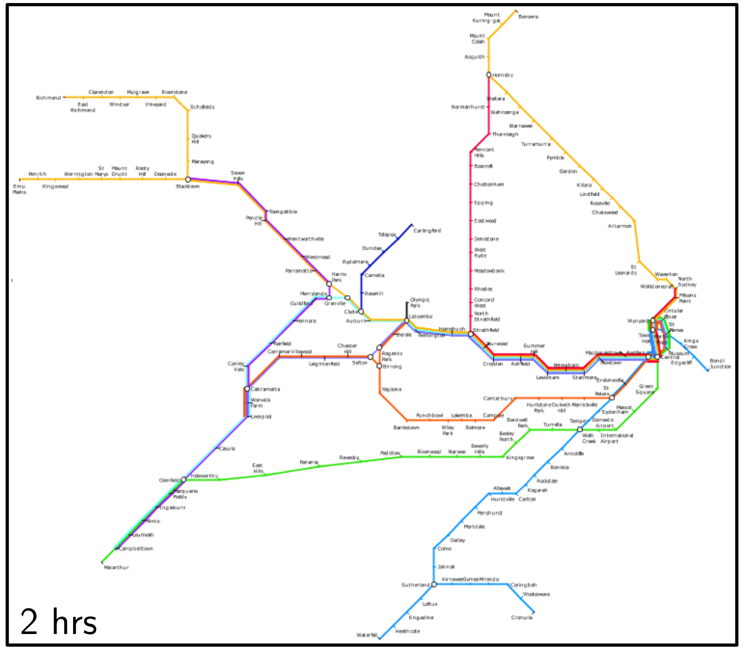


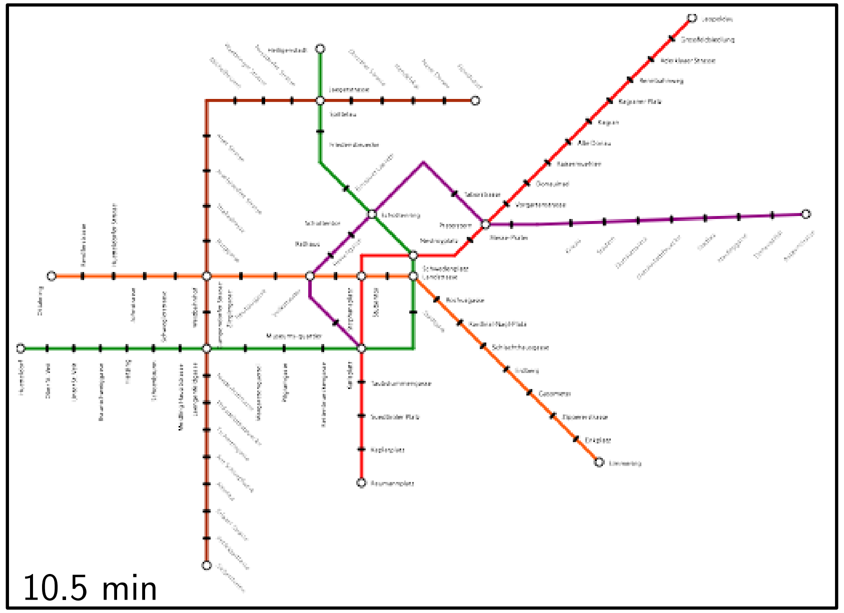
Local Schematization

[Stott et al. 2011]

Idea.

- design rules as before
- additionally include metro map specific criteria (bend minimization **(R3)**, interchange straightness **(R4)**, angular resolution **(R5)**, relative positions **(R6)**)
- integrate alternating label placement rounds
- some ad-hoc fixes for local minima situations





Local Schematization – Discussion

Pros.

Cons.

Local Schematization – Discussion

Pros.

- flexible framework, easy to integrate new criteria

Cons.

Local Schematization – Discussion

Pros.

- flexible framework, easy to integrate new criteria
- recent methods improved visual layout quality

Cons.

Local Schematization – Discussion

Pros.

- flexible framework, easy to integrate new criteria
- recent methods improved visual layout quality
- integration of layout and labeling

Cons.

Local Schematization – Discussion

Pros.

- flexible framework, easy to integrate new criteria
- recent methods improved visual layout quality
- integration of layout and labeling

Cons.

- optimization of criteria, but no guarantees (except topology)

Local Schematization – Discussion

Pros.

- flexible framework, easy to integrate new criteria
- recent methods improved visual layout quality
- integration of layout and labeling

Cons.

- optimization of criteria, but no guarantees (except topology)
- susceptible to local minima

Local Schematization – Discussion

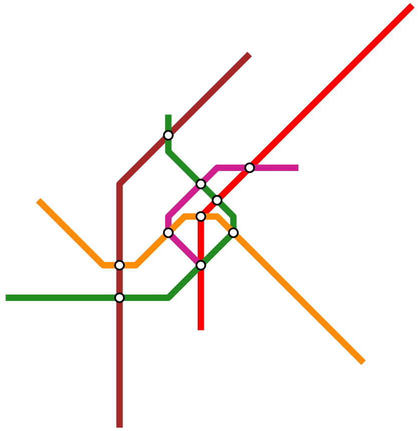
Pros.

- flexible framework, easy to integrate new criteria
- recent methods improved visual layout quality
- integration of layout and labeling

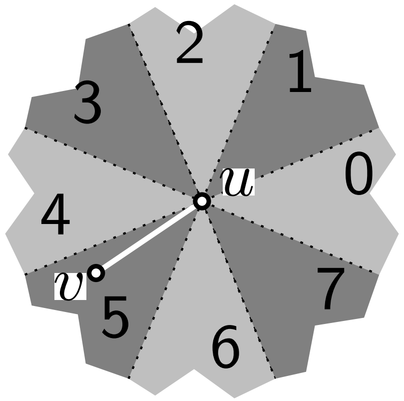
Cons.

- optimization of criteria, but no guarantees (except topology)
- susceptible to local minima
- long running times

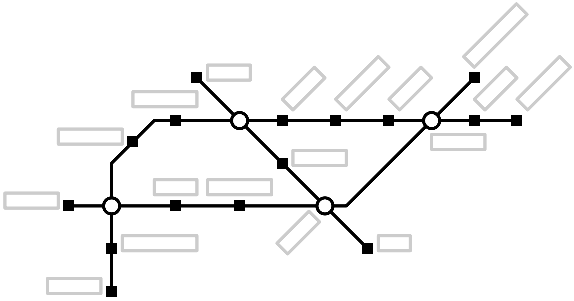
Visualization of Graphs



Lecture 12:
Octilinear Graph Drawing
Metro Map Layout



Part V:
Mixed-Integer Programming



Jonathan Klawitter

Mixed-Integer Programming

[Nöllenburg & Wolff 2011]

- find exact optimum solution using combinatorial optimization

Mixed-Integer Programming

[Nöllenburg & Wolff 2011]

- find exact optimum solution using combinatorial optimization
- split design rules into **hard** and **soft** constraints

Mixed-Integer Programming

[Nöllenburg & Wolff 2011]

- find exact optimum solution using combinatorial optimization
- split design rules into **hard** and **soft** constraints
- model constraints as linear (in)equalities and linear objective function $\rightarrow$ mixed-integer programming **MIP**

Mixed-Integer Programming

[Nöllenburg & Wolff 2011]

- find exact optimum solution using combinatorial optimization
- split design rules into **hard** and **soft** constraints
- model constraints as linear (in)equalities and linear objective function $\rightarrow$ mixed-integer programming **MIP**
- integrate overlap-free station labeling in same model

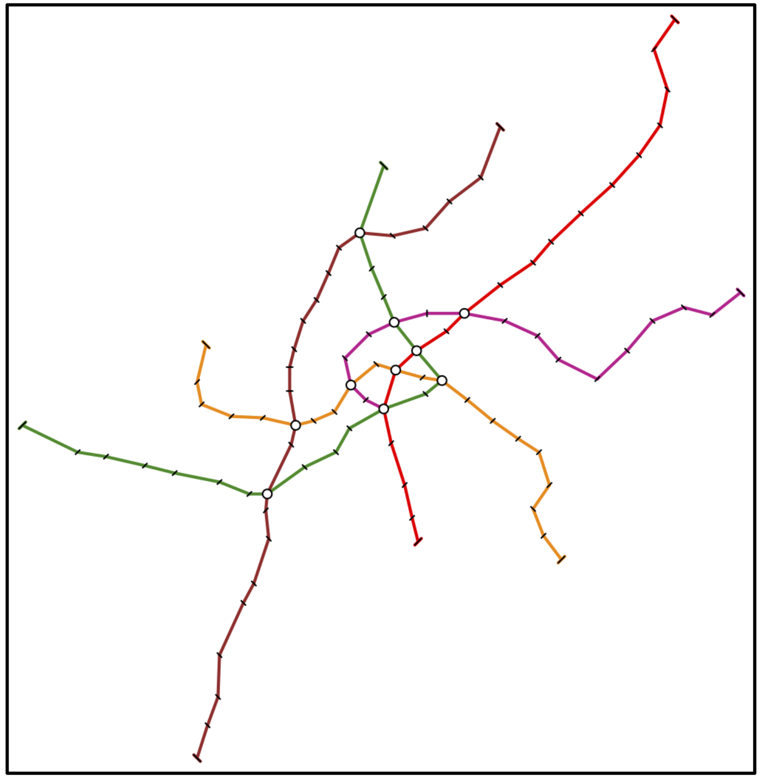
Mixed-Integer Programming

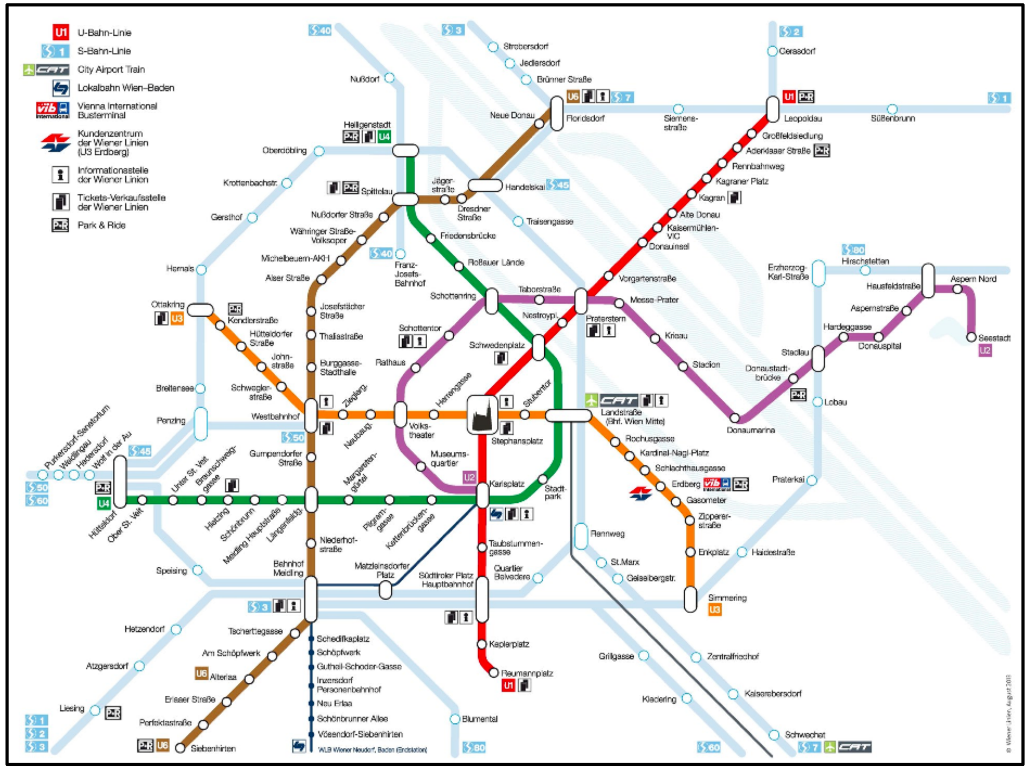
[Nöllenburg & Wolff 2011]

- find exact optimum solution using combinatorial optimization
- split design rules into **hard** and **soft** constraints
- model constraints as linear (in)equalities and linear objective function $\rightarrow$ mixed-integer programming **MIP**
- integrate overlap-free station labeling in same model
- can use sophisticated optimization tools as black box solvers (e.g., CPLEX, Gurobi)

Hard Constraints

(R1) preserve embedding/**topology** and **planarity**

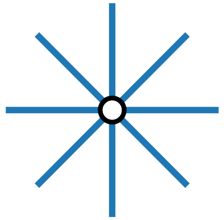


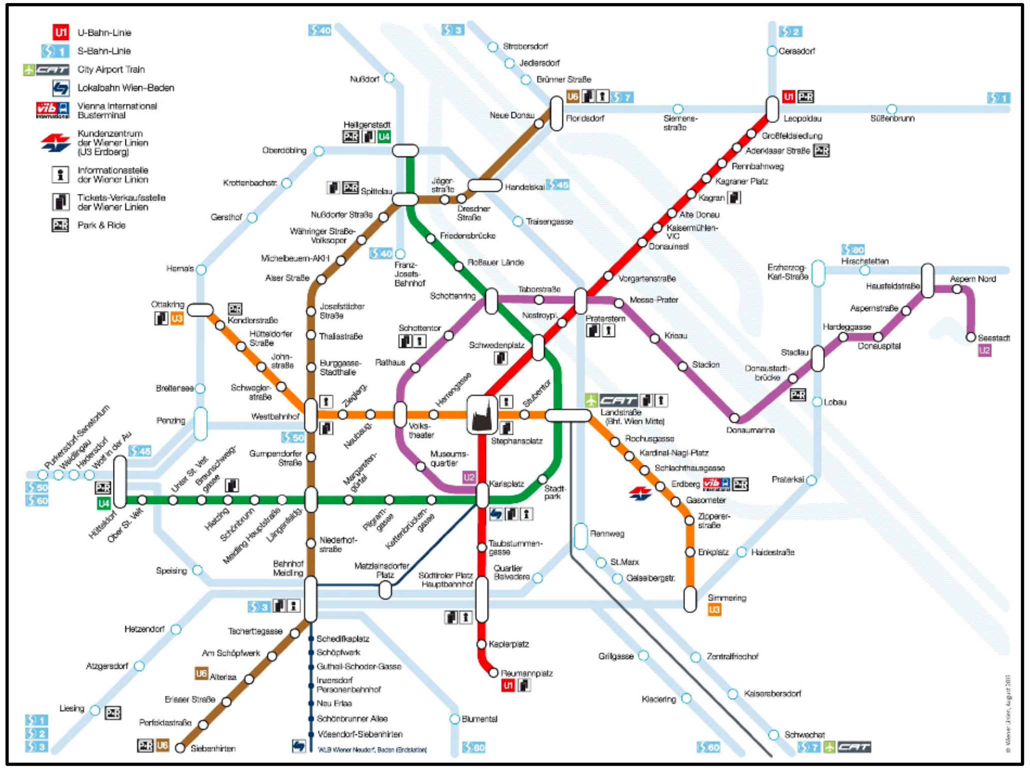


Hard Constraints

(R1) preserve embedding/**topology** and **planarity**

(R2) draw all edges **octilinearly**



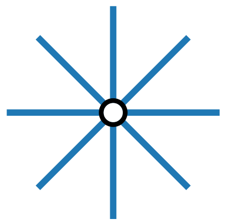


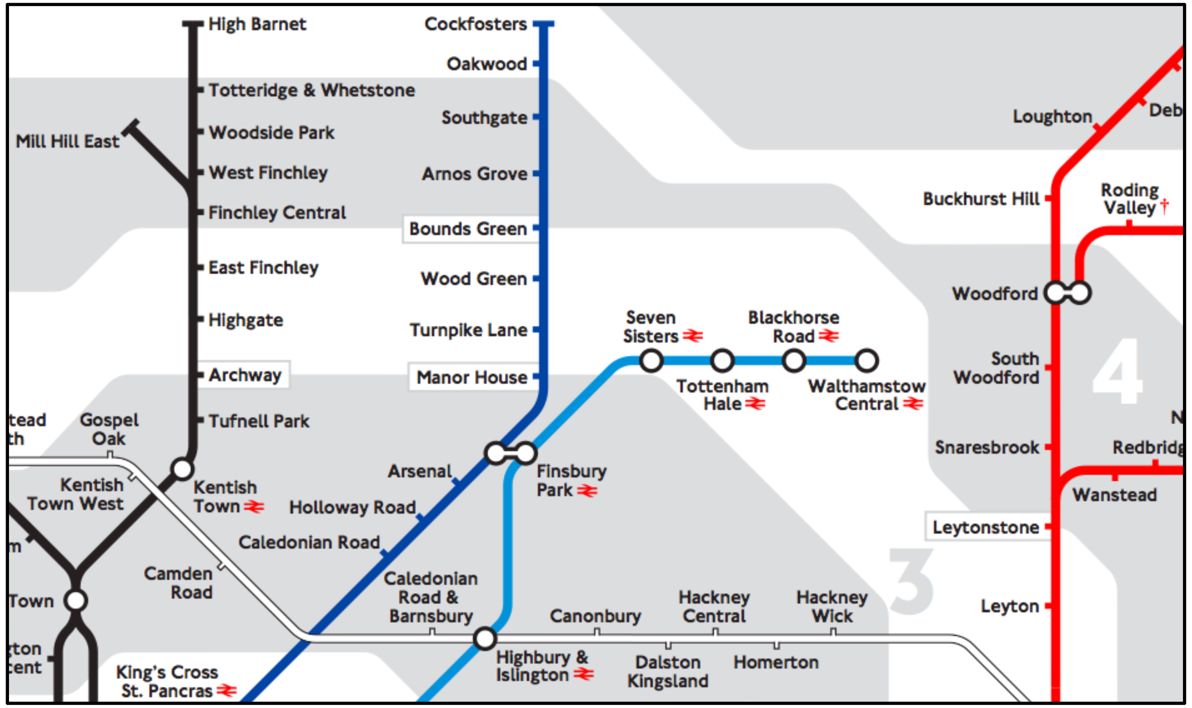
Hard Constraints

(R1) preserve embedding/**topology** and **planarity**

(R2) draw all edges **octilinearly**

(R7) enforce **minimum edge length** $\ell_{\min}$





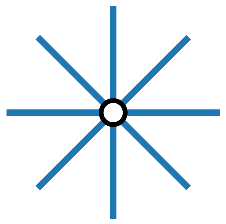
Hard Constraints

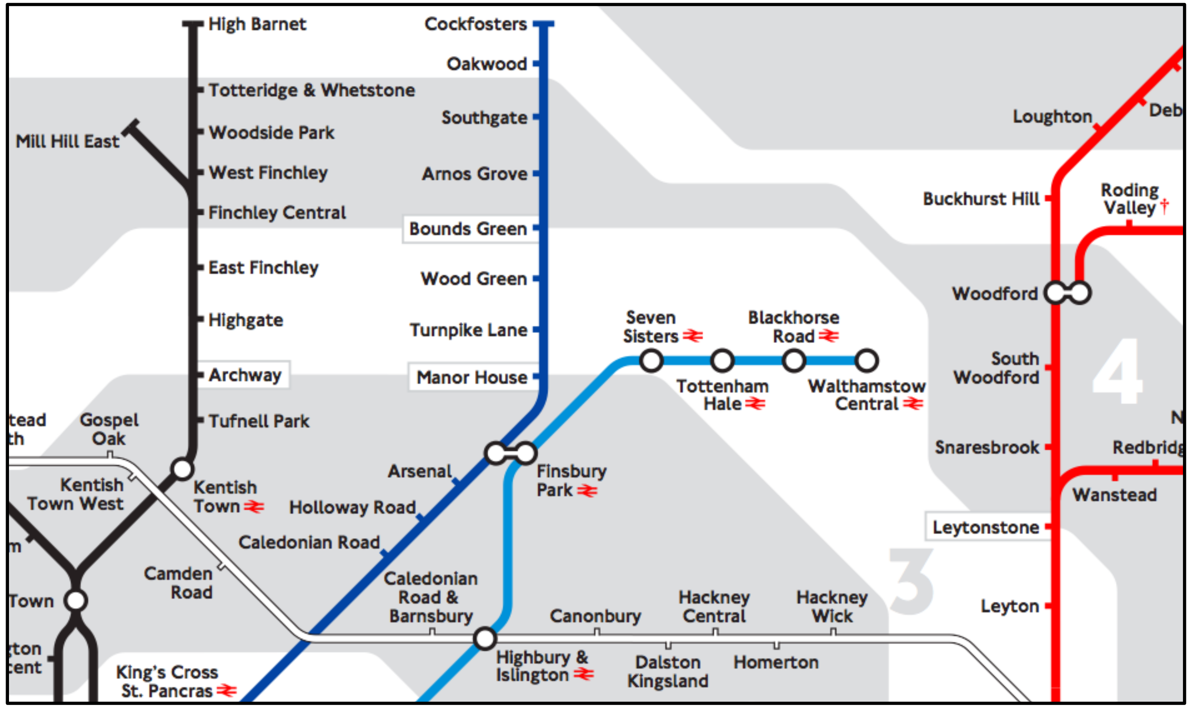
(R1) preserve embedding/**topology** and **planarity**

(R2) draw all edges **octilinearly**

(R7) enforce **minimum edge length** ℓ_{min}

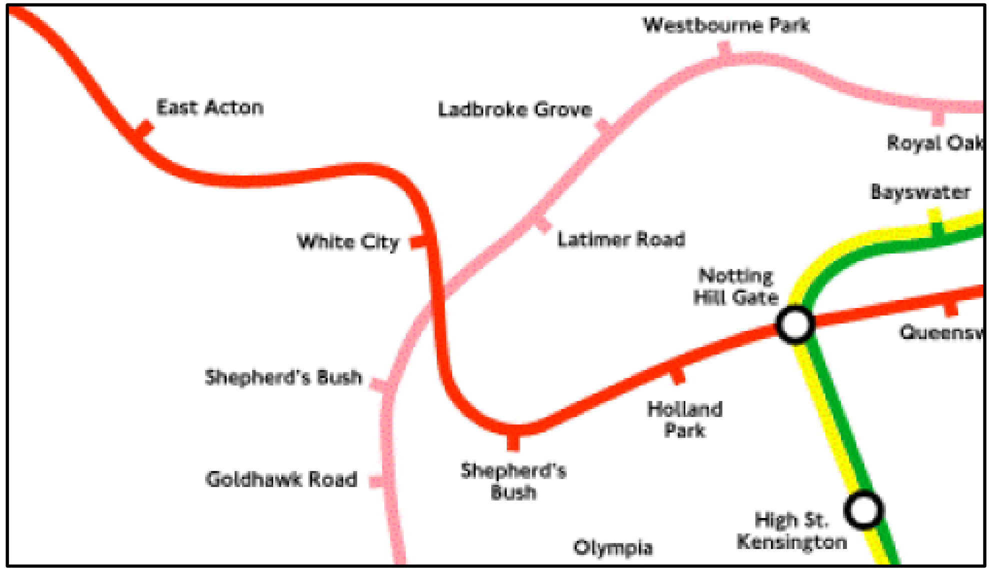
(R8) enforce **minimum feature separation** d_{min}

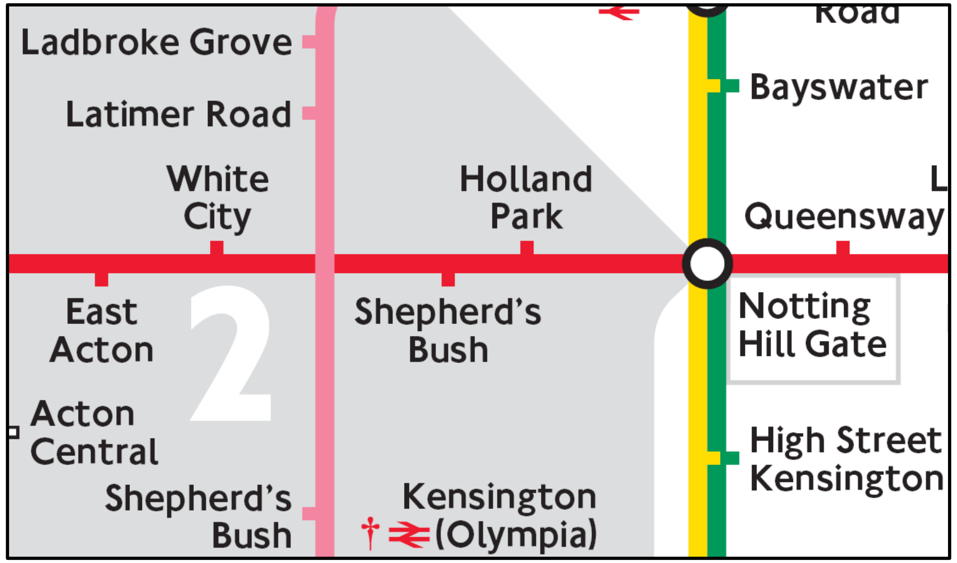




Soft Constraints

(R3+R4) draw lines in $\mathcal{L}$ with **few bends**

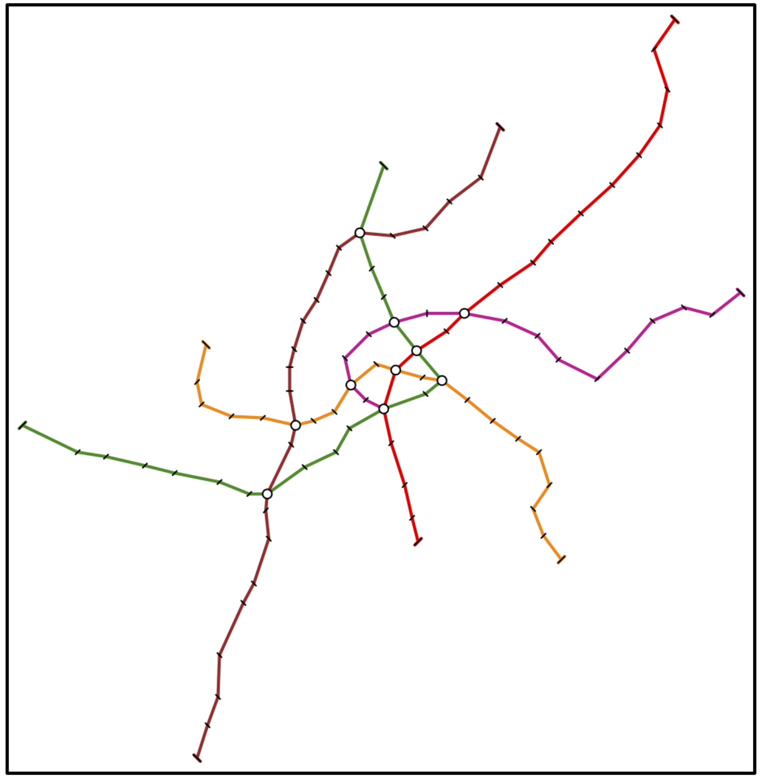


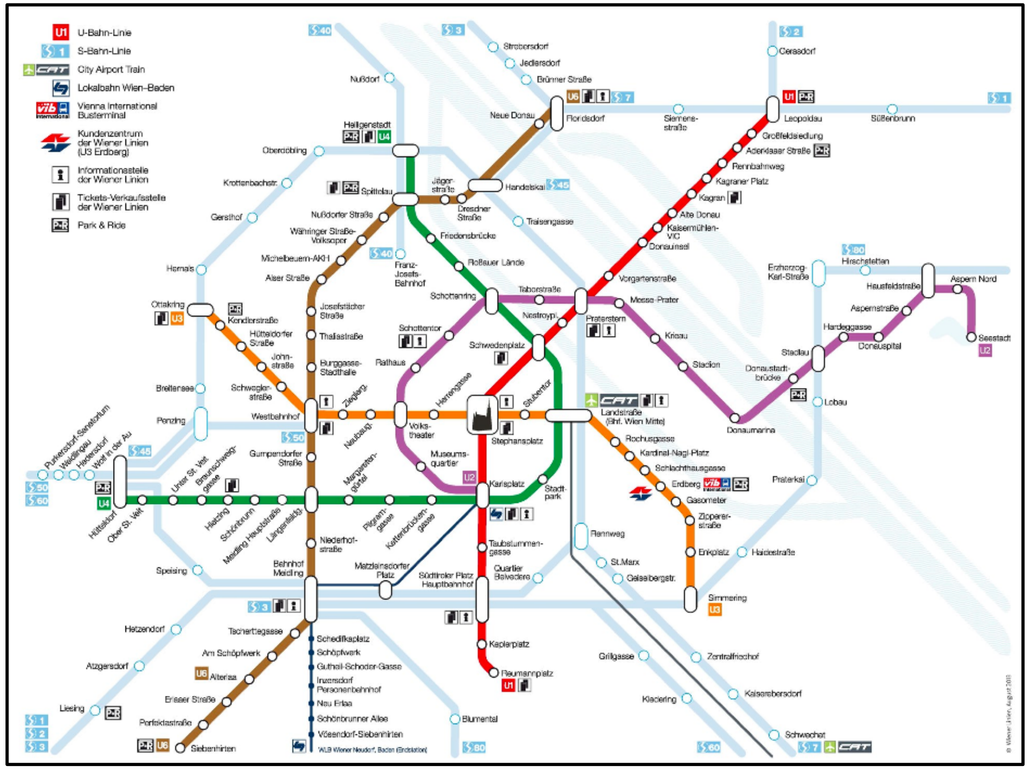


Soft Constraints

(R3+R4) draw lines in $\mathcal{L}$ with **few bends**

(R6) minimize geometric **distortion**



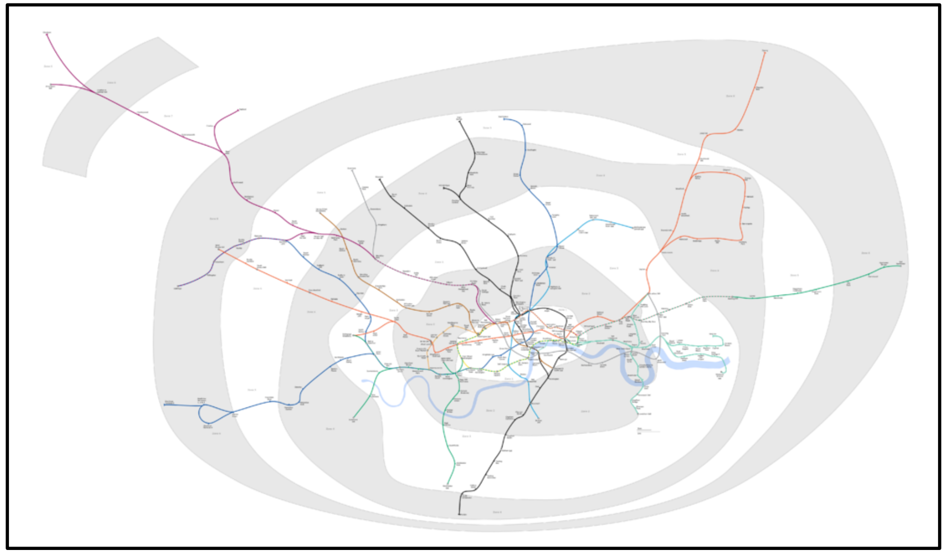


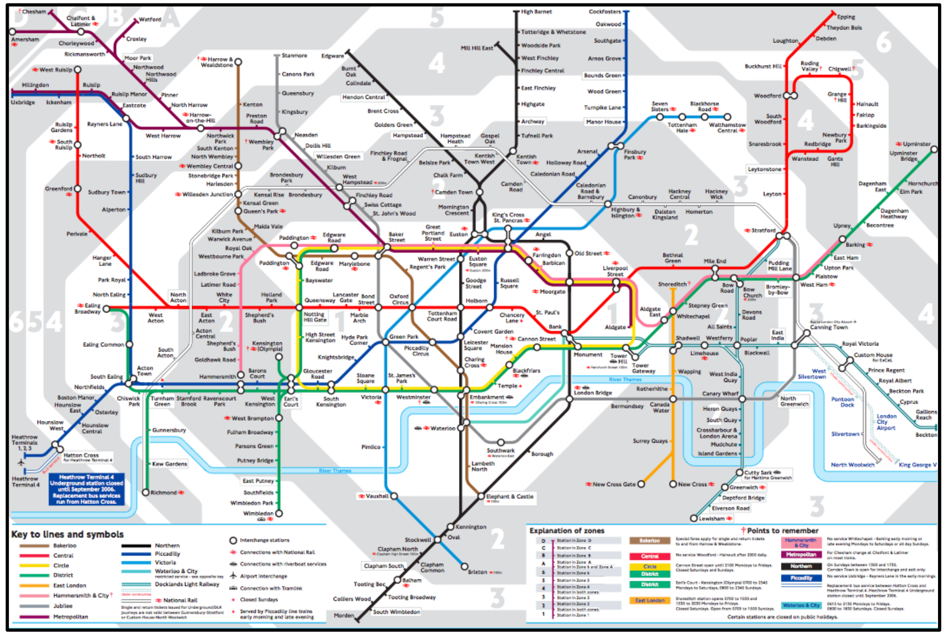
Soft Constraints

(R3+R4) draw lines in $\mathcal{L}$ with **few bends**

(R6) minimize geometric **distortion**

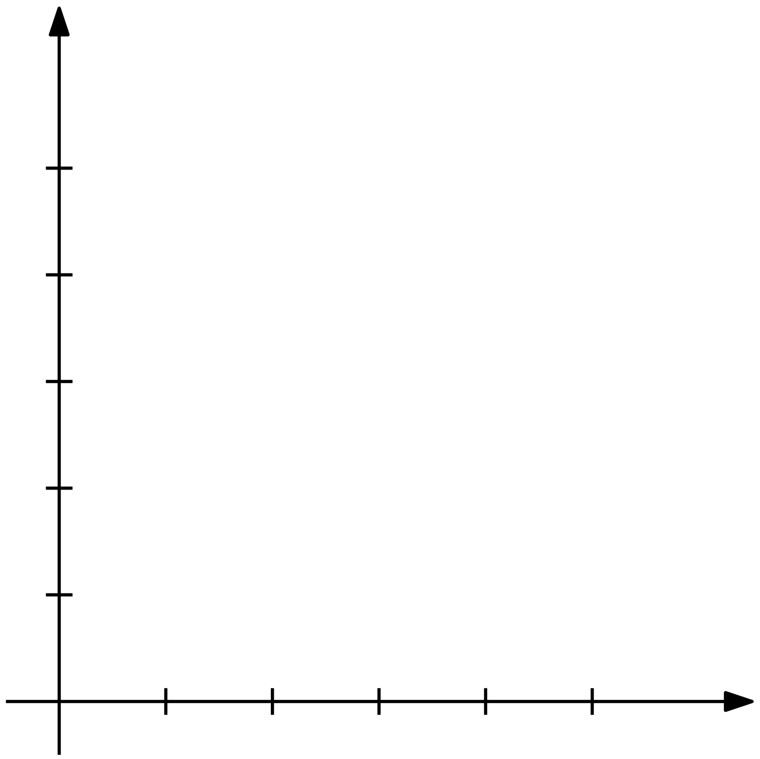
(R7) minimize total **edge length**





Linear Programming

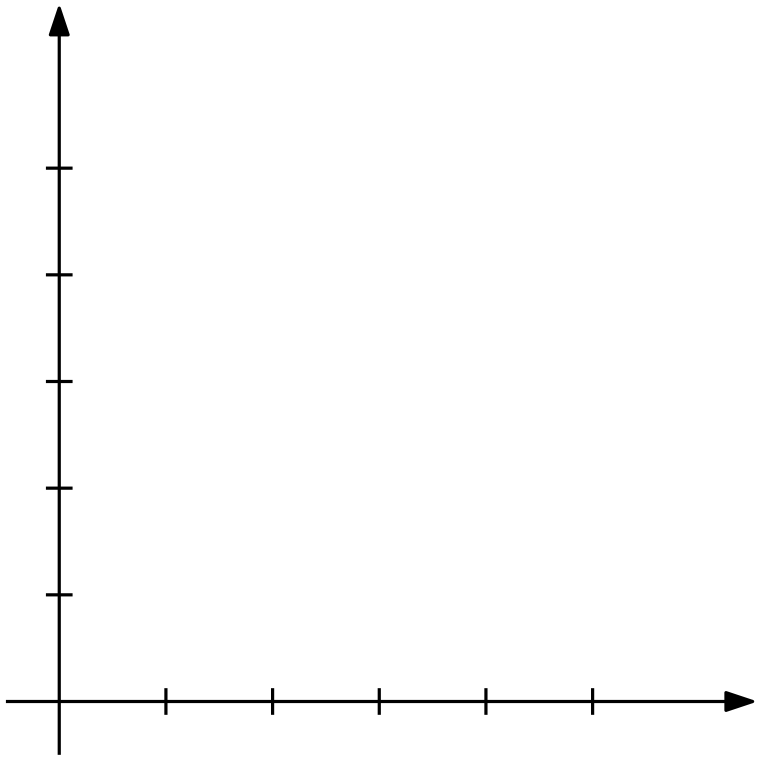
Linear Programming (LP) is an efficient optimization method for



Linear Programming

Linear Programming (LP) is an efficient optimization method for

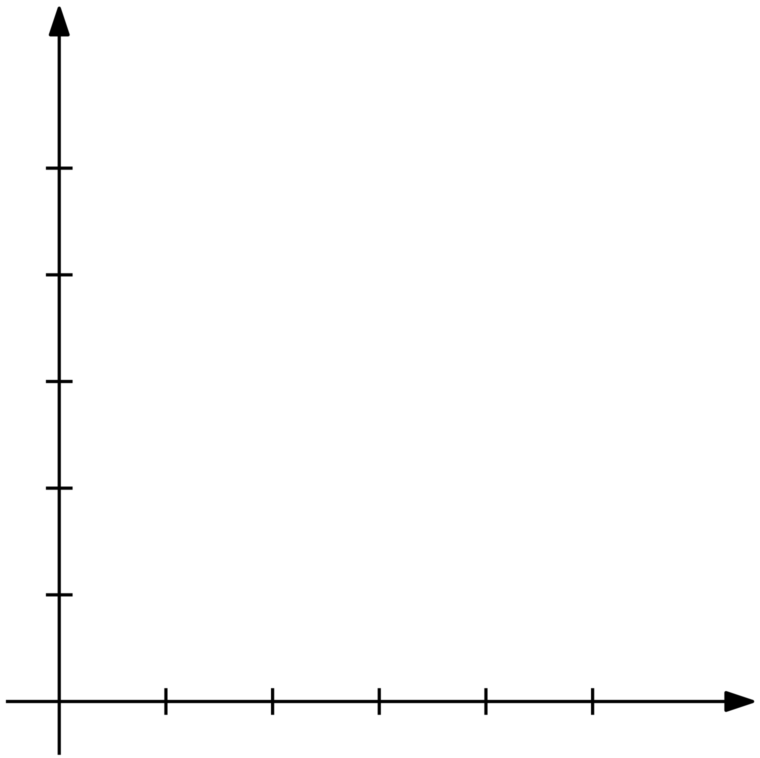
- linear constraints



Linear Programming

Linear Programming (LP) is an efficient optimization method for

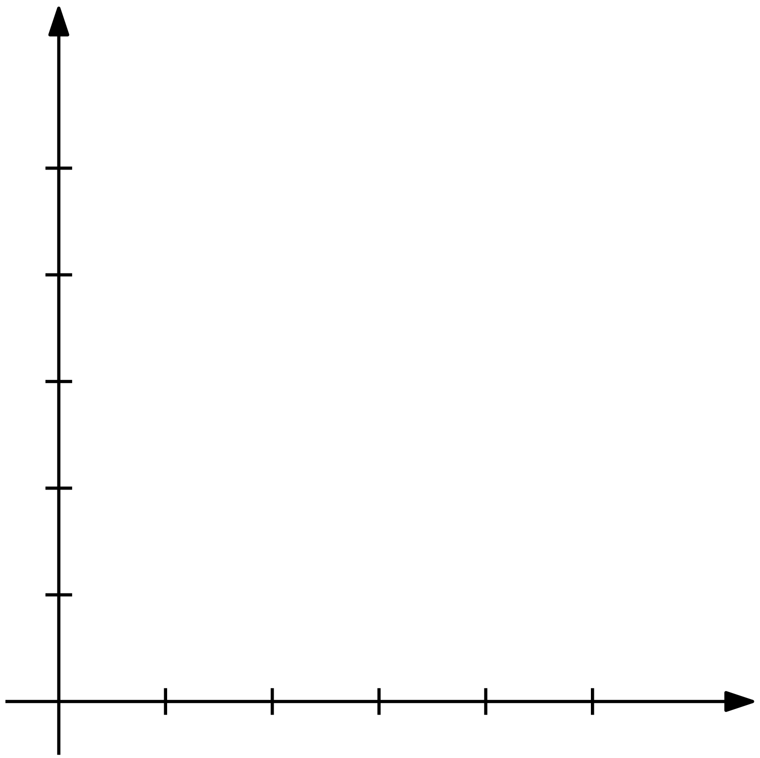
- linear constraints
- linear objective function



Linear Programming

Linear Programming (LP) is an efficient optimization method for

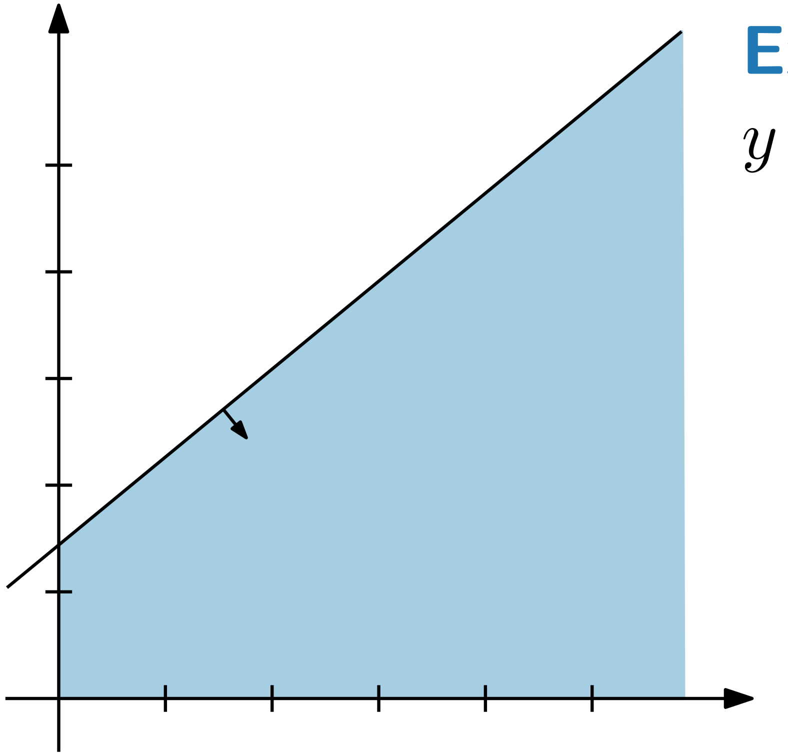
- linear constraints
- linear objective function
- real-valued variables



Linear Programming

Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables

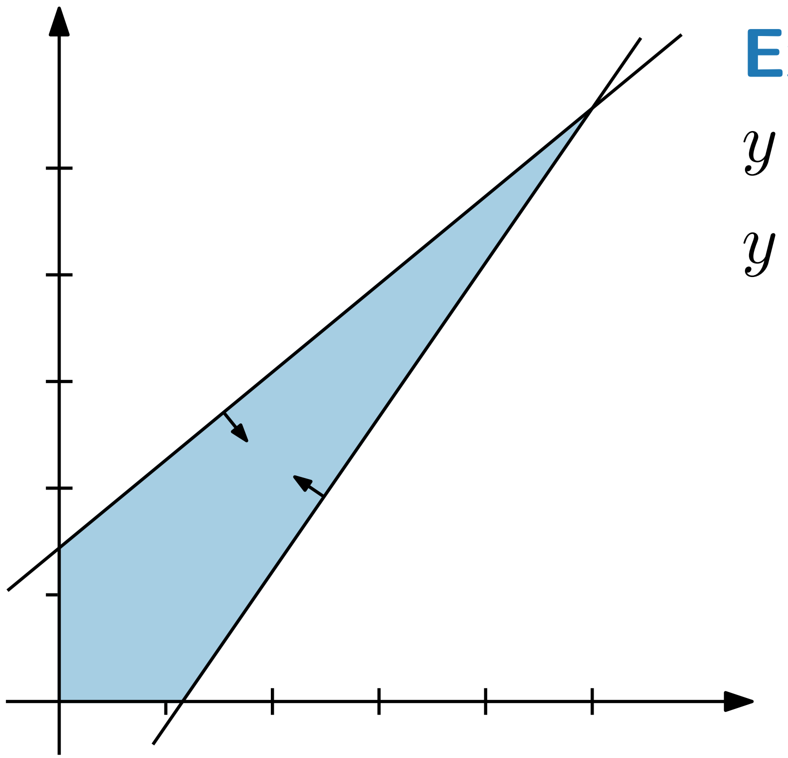


Example.
$y \leq 0.9x + 1.5$

Linear Programming

Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables



Example.

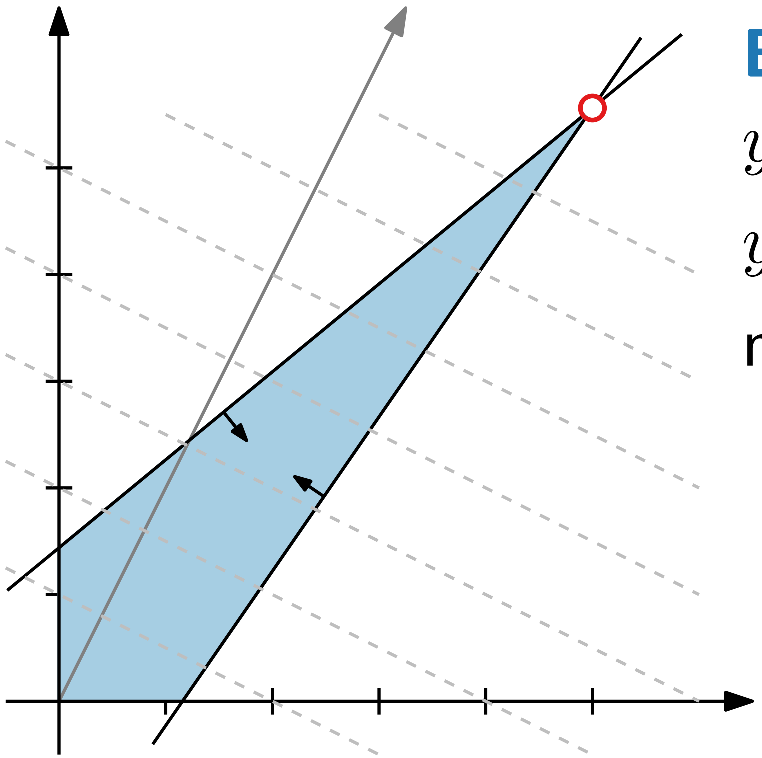
$y \leq 0.9x + 1.5$

$y \geq 1.4x - 1.3$

Linear Programming

Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables



Example.

$y \leq 0.9x + 1.5$

$y \geq 1.4x - 1.3$

maximize $x + 2y$

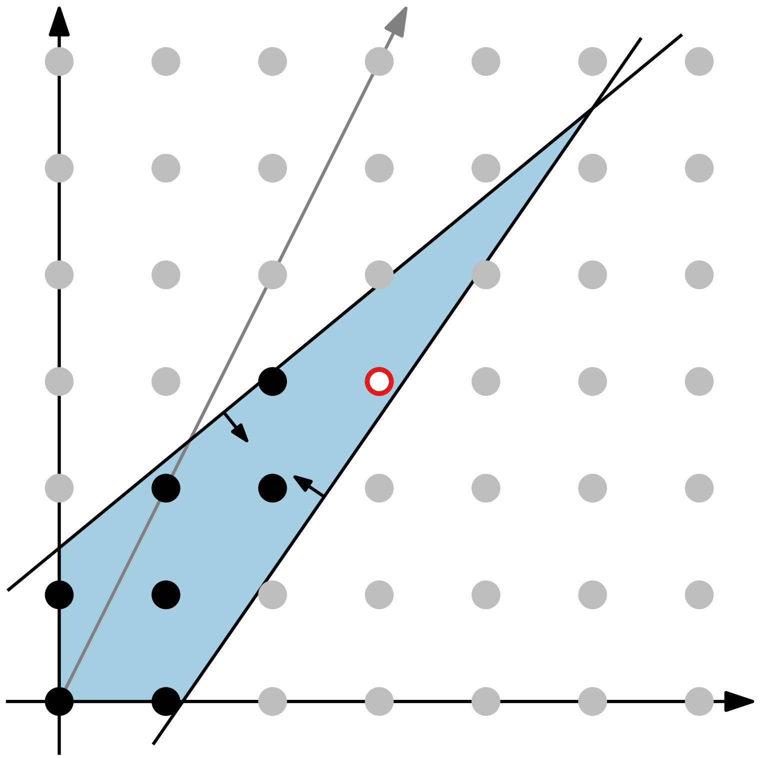
Linear Programming

Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables

Mixed Integer Programming (MIP)

- in addition binary and integer variables



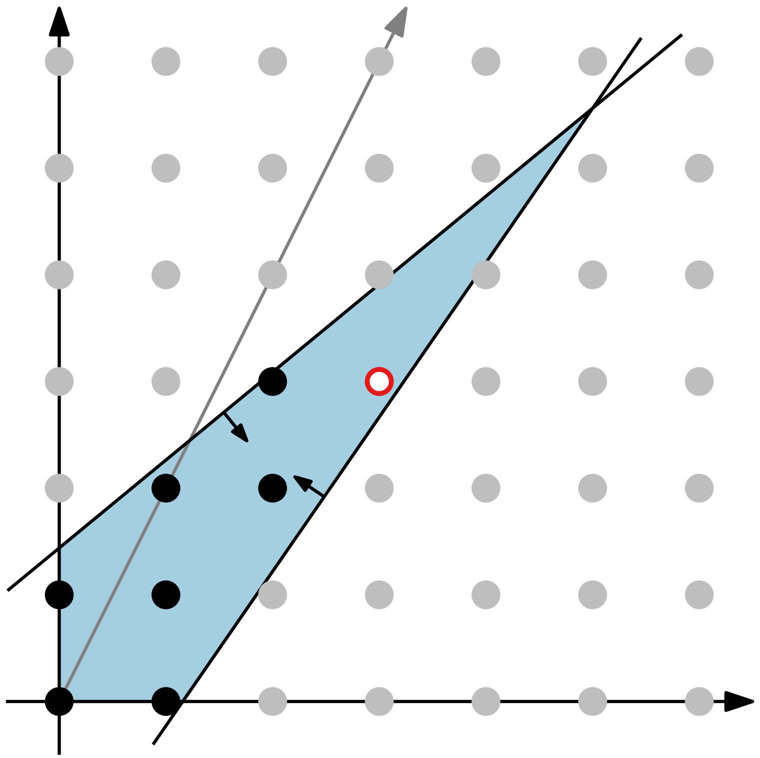
Linear Programming

Linear Programming (LP) is an efficient optimization method for

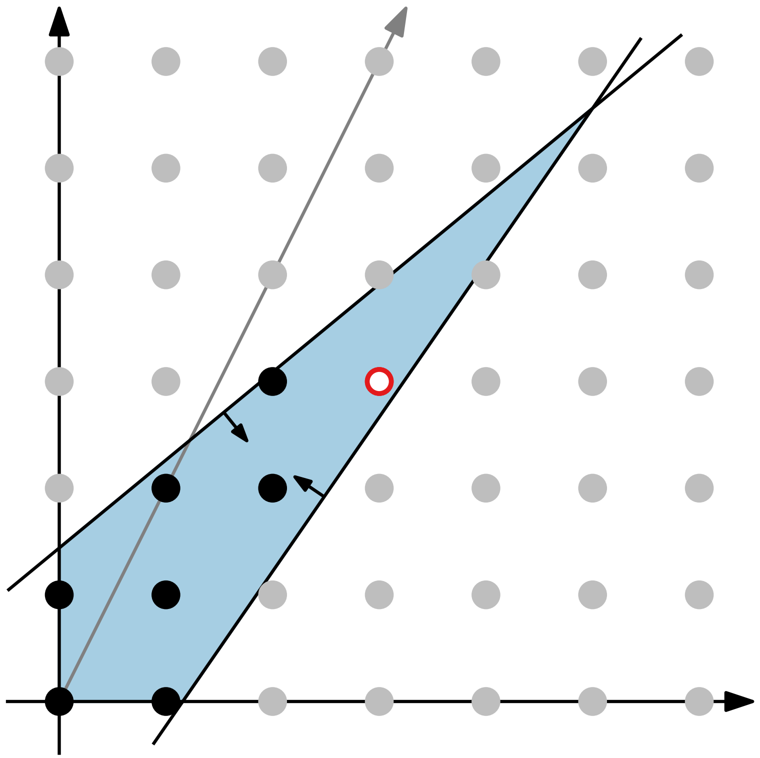
- linear constraints
- linear objective function
- real-valued variables

Mixed Integer Programming (MIP)

- in addition binary and integer variables
- NP-hard optimization problem



Linear Programming



Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables

Mixed Integer Programming (MIP)

- in addition binary and integer variables
- NP-hard optimization problem
- still method of choice for many practical optimization tasks

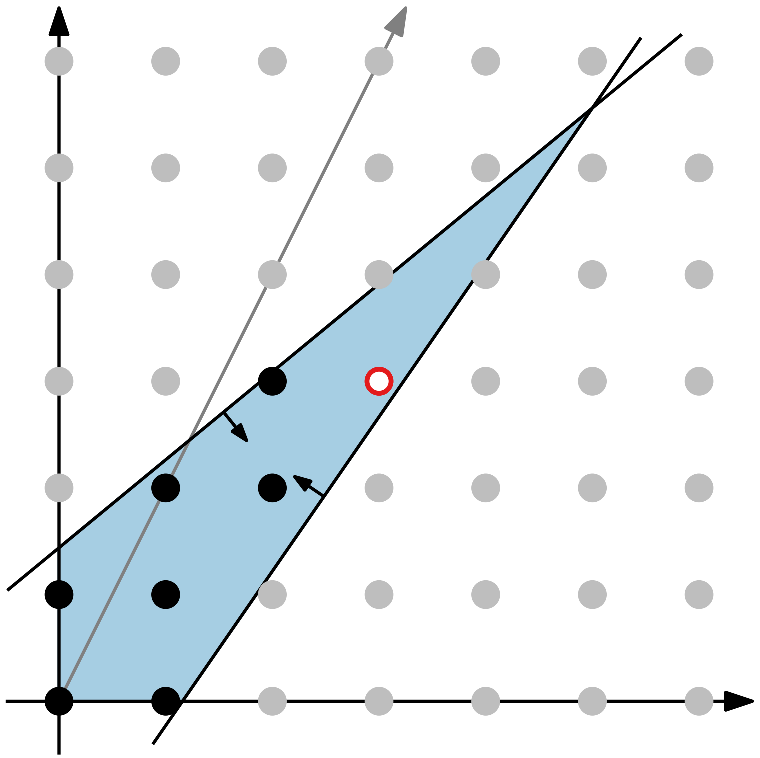
Linear Programming

Linear Programming (LP) is an efficient optimization method for

- linear constraints
- linear objective function
- real-valued variables

Mixed Integer Programming (MIP)

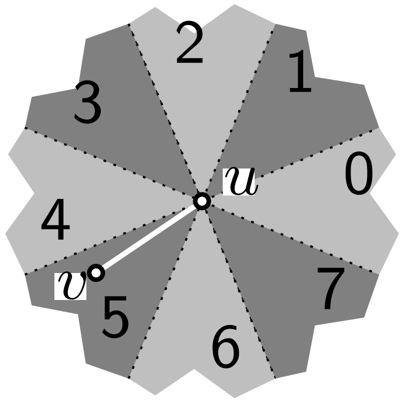
- in addition binary and integer variables
- NP-hard optimization problem
- still method of choice for many practical optimization tasks



Theorem 5.
Metro map layout can be modeled as MIP such that

- hard constraints $\rightarrow$ linear constraints
- soft constraints $\rightarrow$ linear objective function

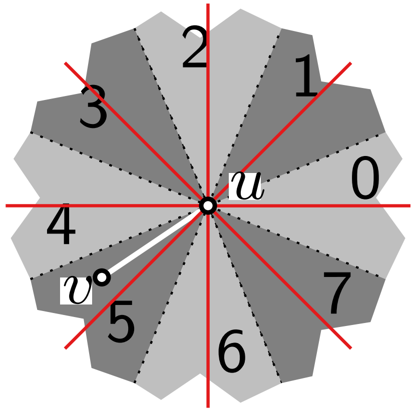
Sectors and Coordinates



Sectors.

- for each vertex u partition the plane into eight sectors numbered 0–7
  here: $\text{sec}(u, v) = 5$ in the input

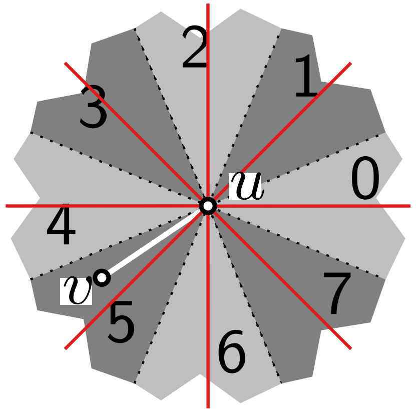
Sectors and Coordinates



Sectors.

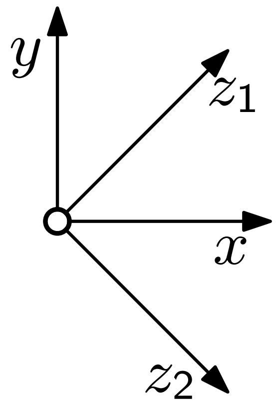
- for each vertex u partition the plane into eight sectors numbered 0–7
  here: $\text{sec}(u, v) = 5$ in the input
- number octilinear edge directions accordingly
  here, e.g., $\text{dir}(u, v) = 5$

Sectors and Coordinates



Sectors.

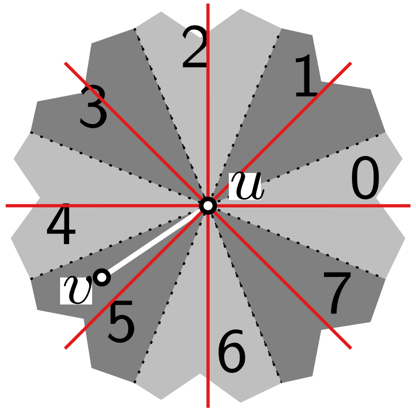
- for each vertex u partition the plane into eight sectors numbered 0–7
  here: $\text{sec}(u, v) = 5$ in the input
- number octilinear edge directions accordingly
  here, e.g., $\text{dir}(u, v) = 5$



Coordinates.

- assign (redundant) z_1- and z_2-coordinates to each vertex v
  - $z_1(v) = \frac{1}{2} \cdot (x(v) + y(v))$
  - $z_2(v) = \frac{1}{2} \cdot (x(v) - y(v))$

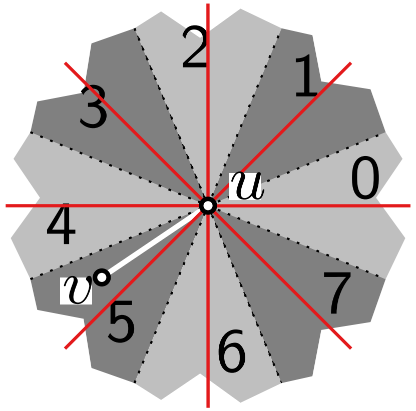
Octilinearity and Relative Position



Goal.

Draw the edge uv

Octilinearity and Relative Position

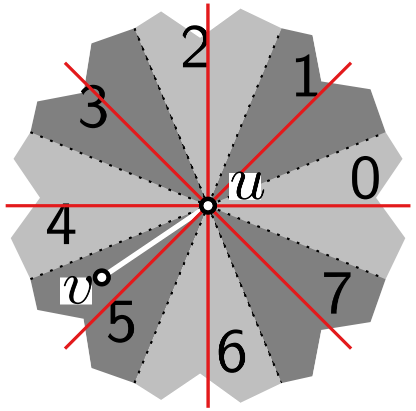


Goal.

Draw the edge uv

- octilinearly **(R2)**

Octilinearity and Relative Position

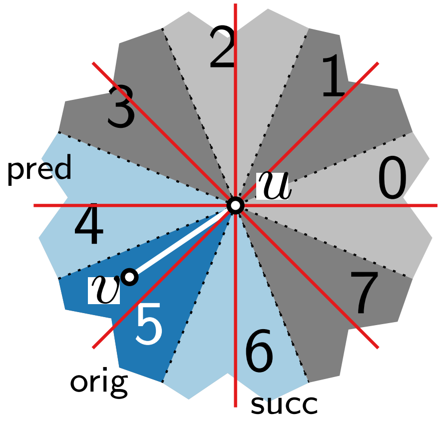


Goal.

Draw the edge uv

- octilinearly **(R2)**
- with minimum length $\ell = \ell_{uv}$ **(R7)**

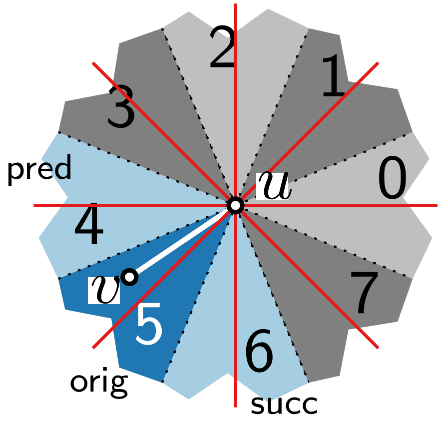
Octilinearity and Relative Position



Goal.
Draw the edge uv

- octilinearly **(R2)**
- with minimum length $\ell = \ell_{uv}$ **(R7)**
- restricted to the three best directions **(R6)**

Octilinearity and Relative Position



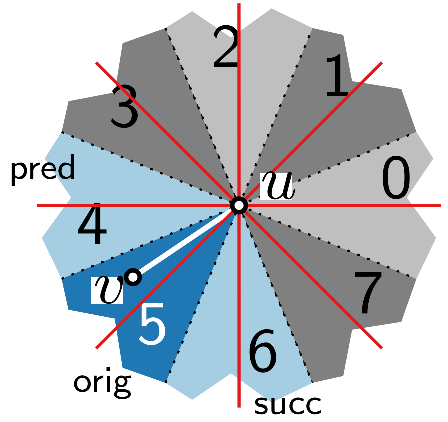
Goal.

Draw the edge uv

- octilinearly **(R2)**
- with minimum length $\ell = \ell_{uv}$ **(R7)**
- restricted to the three best directions **(R6)**

How to model this using linear constraints in a MIP?

Octilinearity and Relative Position



Goal.
Draw the edge uv

- octilinearly **(R2)**
- with minimum length $\ell = \ell_{uv}$ **(R7)**
- restricted to the three best directions **(R6)**

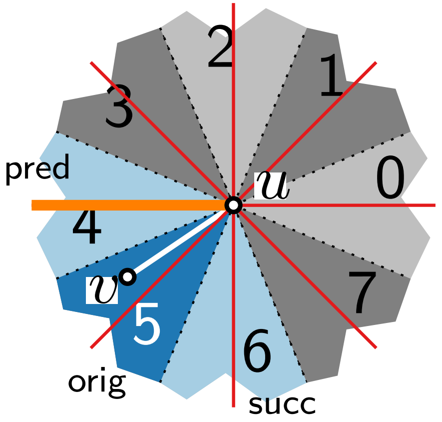
How to model this using linear constraints in a MIP?

Introduce **binary variables**

$$\alpha_{\mathsf{pred}}(u, v) + \alpha_{\mathsf{orig}}(u, v) + \alpha_{\mathsf{succ}}(u, v) = 1$$

to select exactly one of the three sectors.

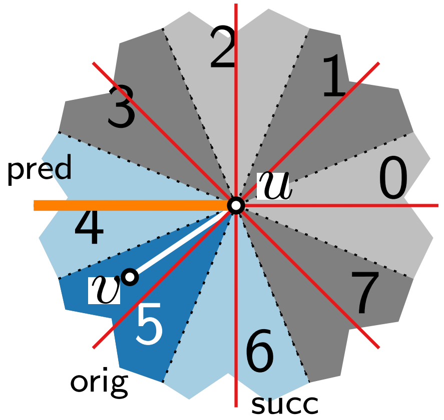
Octilinearity and Relative Position



Predecessor sector.

$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell \end{aligned}$$

Octilinearity and Relative Position

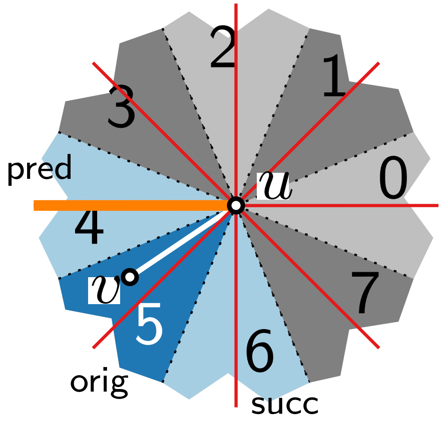


very large constant

Predecessor sector.

$$\begin{aligned} y(u) - y(v) &\le M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ -y(u) + y(v) &\le M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ x(u) - x(v) &\ge -M(1 - \alpha_{\mathsf{pred}}(u,v)) + \ell \end{aligned}$$

Octilinearity and Relative Position



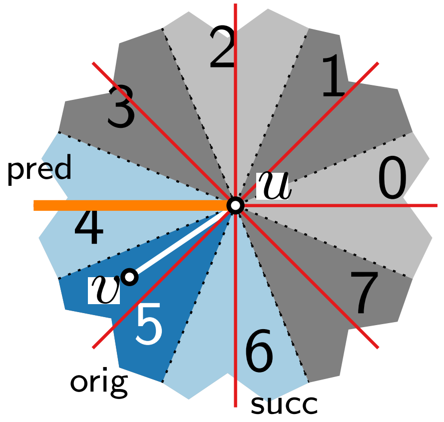
Predecessor sector.

very large constant

$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\mathsf{pred}}(u,v)) + \ell \end{aligned}$$

How does this work?

Octilinearity and Relative Position



very large constant

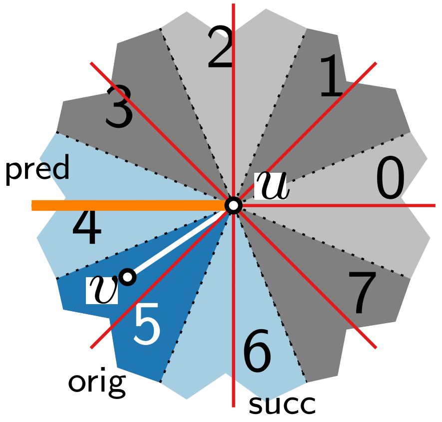
Predecessor sector.

$$
\begin{aligned}
y(u) - y(v) &\le M(1 - \alpha_{\mathsf{pred}}(u,v)) \\
-y(u) + y(v) &\le M(1 - \alpha_{\mathsf{pred}}(u,v)) \\
x(u) - x(v) &\ge -M(1 - \alpha_{\mathsf{pred}}(u,v)) + \ell
\end{aligned}
$$

How does this work?

Case 1: $\alpha_{\mathsf{pred}}(u,v) = 1$

Octilinearity and Relative Position



Predecessor sector.

very large constant

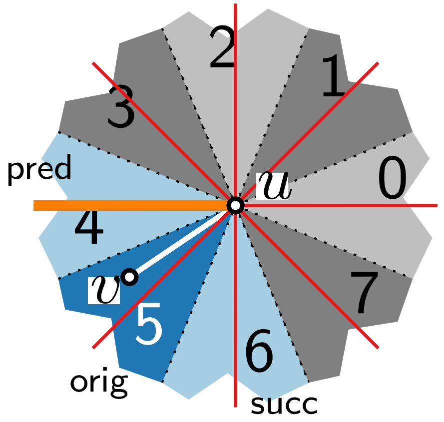
$$
\begin{aligned}
y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\
-y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\
x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell
\end{aligned}
$$

How does this work?

Case 1: $\alpha_{\text{pred}}(u,v) = 1$

$$
\begin{aligned}
y(u) - y(v) &\leq 0 \\
-y(u) + y(v) &\leq 0 \\
x(u) - x(v) &\geq \ell
\end{aligned}
$$

Octilinearity and Relative Position



Predecessor sector.

very large constant

$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\mathsf{pred}}(u,v)) + \ell \end{aligned}$$

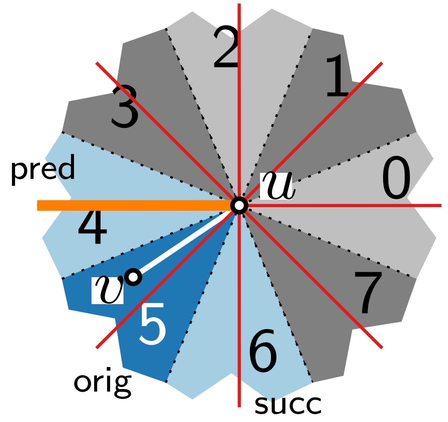
How does this work?

Case 1: $\alpha_{\mathsf{pred}}(u,v) = 1$

$$\begin{aligned} y(u) - y(v) &\leq 0 \\ -y(u) + y(v) &\leq 0 \\ x(u) - x(v) &\geq \ell \end{aligned}$$

Case 2: $\alpha_{\mathsf{pred}}(u,v) = 0$

Octilinearity and Relative Position



Predecessor sector.

very large constant

$$
\begin{aligned}
y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\
-y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\
x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell
\end{aligned}
$$

How does this work?

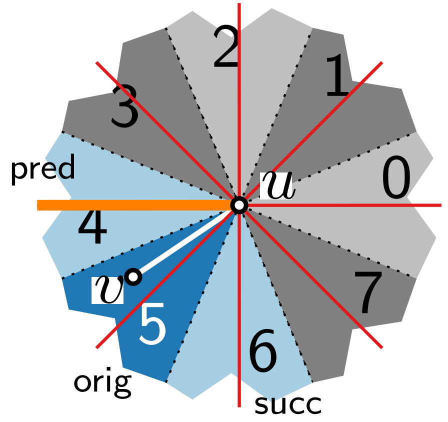
Case 1: $\alpha_{\text{pred}}(u,v) = 1$

$$
\begin{aligned}
y(u) - y(v) &\leq 0 \\
-y(u) + y(v) &\leq 0 \\
x(u) - x(v) &\geq \ell
\end{aligned}
$$

Case 2: $\alpha_{\text{pred}}(u,v) = 0$

$$
\begin{aligned}
y(u) - y(v) &\leq M \\
-y(u) + y(v) &\leq M \\
x(u) - x(v) &\geq -M + \ell
\end{aligned}
$$

Octilinearity and Relative Position



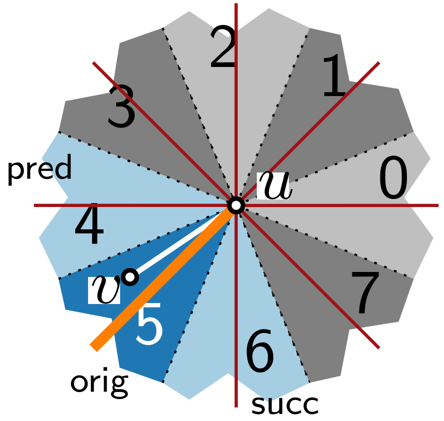
Predecessor sector.

very large constant

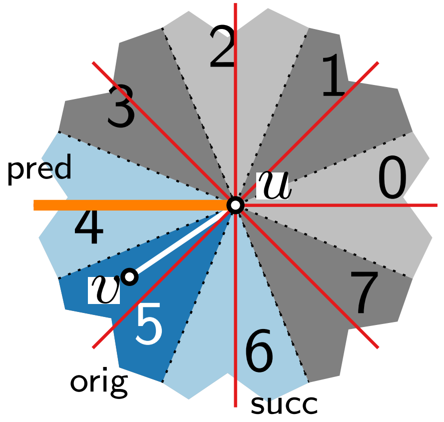
$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\mathsf{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\mathsf{pred}}(u,v)) + \ell \end{aligned}$$

Original sector.

$$\begin{aligned} z_2(u) - z_2(v) &\leq M(1 - \alpha_{\mathsf{orig}}(u,v)) \\ -z_2(u) + z_2(v) &\leq M(1 - \alpha_{\mathsf{orig}}(u,v)) \\ z_1(u) - z_1(v) &\geq -M(1 - \alpha_{\mathsf{orig}}(u,v)) + \ell \end{aligned}$$



Octilinearity and Relative Position



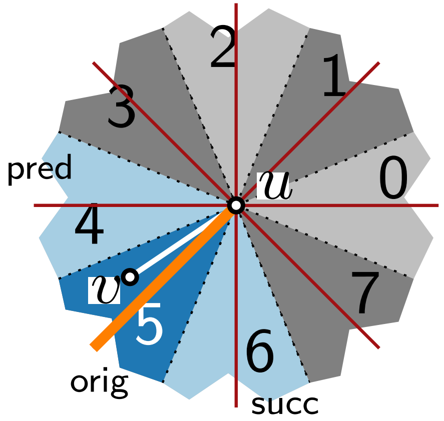
very large constant

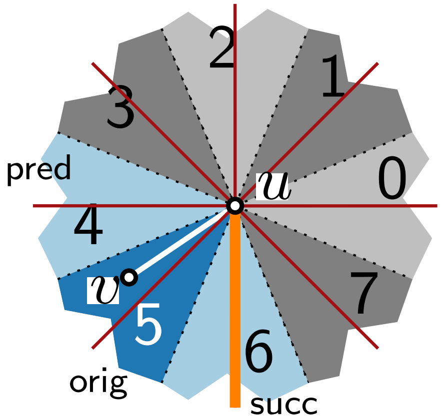
Predecessor sector.

$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell \end{aligned}$$

Original sector.

$$\begin{aligned} z_2(u) - z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\ -z_2(u) + z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\ z_1(u) - z_1(v) &\geq -M(1 - \alpha_{\text{orig}}(u,v)) + \ell \end{aligned}$$





Successor sector.

$$\begin{aligned} x(u) - x(v) &\leq M(1 - \alpha_{\text{succ}}(u,v)) \\ -x(u) + x(v) &\leq M(1 - \alpha_{\text{succ}}(u,v)) \\ y(u) - y(v) &\geq -M(1 - \alpha_{\text{succ}}(u,v)) + \ell \end{aligned}$$

Objective Function

- models the three soft constraints

Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

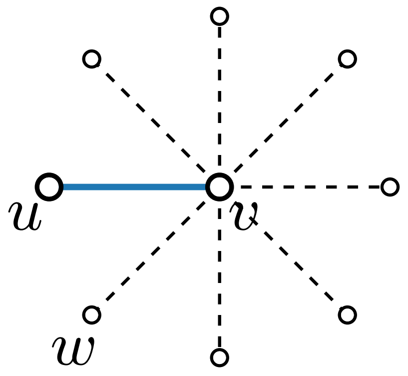
  minimize $\lambda_{\mathrm{bends}}\,\mathrm{cost}_{\mathrm{bends}} + \lambda_{\mathrm{length}}\,\mathrm{cost}_{\mathrm{length}} + \lambda_{\mathrm{dist}}\mathrm{cost}_{\mathrm{dist}}$

Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)



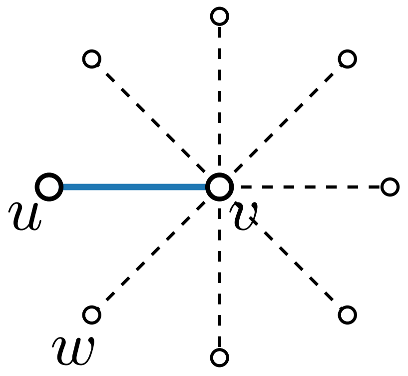
Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)

Edges uv and vw on a line $L \in \mathcal{L}$

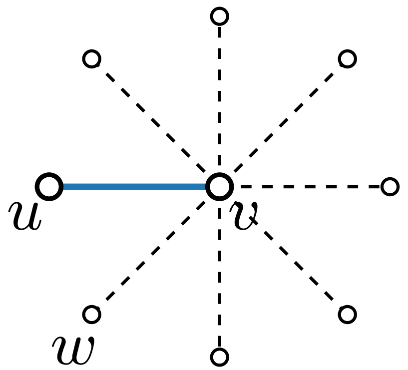


Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)



Edges uv and vw on a line $L \in \mathcal{L}$

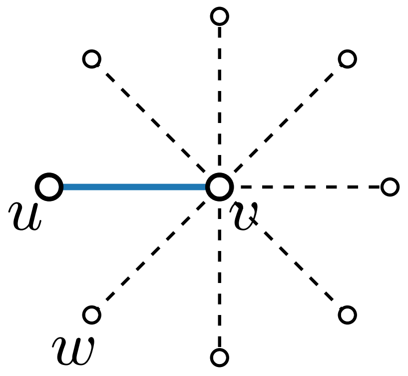
- draw as straight as possible

Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)



Edges uv and vw on a line $L \in \mathcal{L}$

- draw as straight as possible
- increasing cost bend(u, v, w) for increasing acuteness of $\angle(\overline{uv}, \overline{vw})$

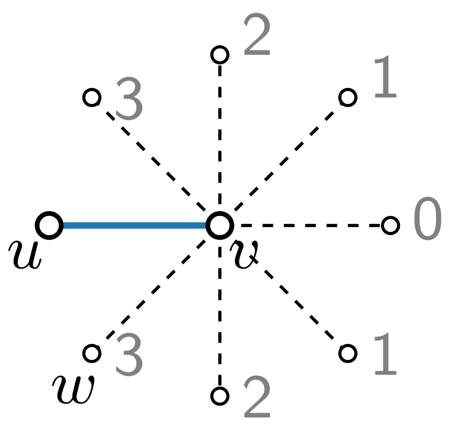
$$\text{cost}_{\text{bends}} = \sum_{L \in \mathcal{L}} \sum_{uv, vw \in L} \text{bend}(u, v, w)$$

Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)



Edges uv and vw on a line $L \in \mathcal{L}$

- draw as straight as possible
- increasing cost bend(u, v, w) for increasing acuteness of $\angle(\overline{uv}, \overline{vw})$

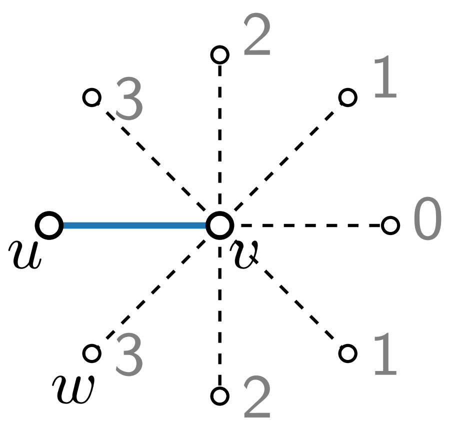
$$\text{cost}_{\text{bends}} = \sum_{L \in \mathcal{L}} \sum_{uv, vw \in L} \text{bend}(u, v, w)$$

Objective Function

- models the three soft constraints
- weighted sum of individual cost functions

  minimize $\lambda_{\text{bends}}\,\text{cost}_{\text{bends}} + \lambda_{\text{length}}\,\text{cost}_{\text{length}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

Example: line bends (R3/R4)



Edges uv and vw on a line $L \in \mathcal{L}$

- draw as straight as possible
- increasing cost bend(u, v, w) for increasing acuteness of $\angle(\overline{uv}, \overline{vw})$

$$\text{cost}_{\text{bends}} = \sum_{L \in \mathcal{L}} \sum_{uv, vw \in L} \text{bend}(u, v, w)$$

To assign bend(u, v, w) correctly, we need to define some linear constraints based on the direction variables dir(u, v) and dir(v, w).

Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\text{bend}}\text{cost}_{\text{bend}} + \lambda_{\text{len}}\text{cost}_{\text{len}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$

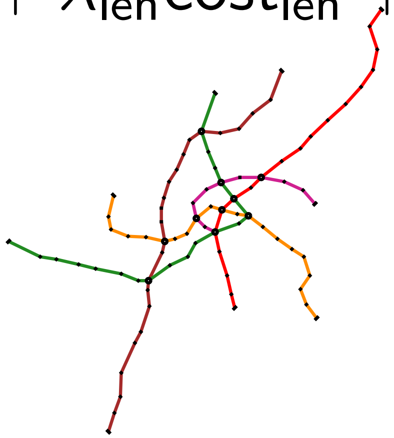
Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\text{bend}}\text{cost}_{\text{bend}} + \lambda_{\text{len}}\text{cost}_{\text{len}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$
- effect:



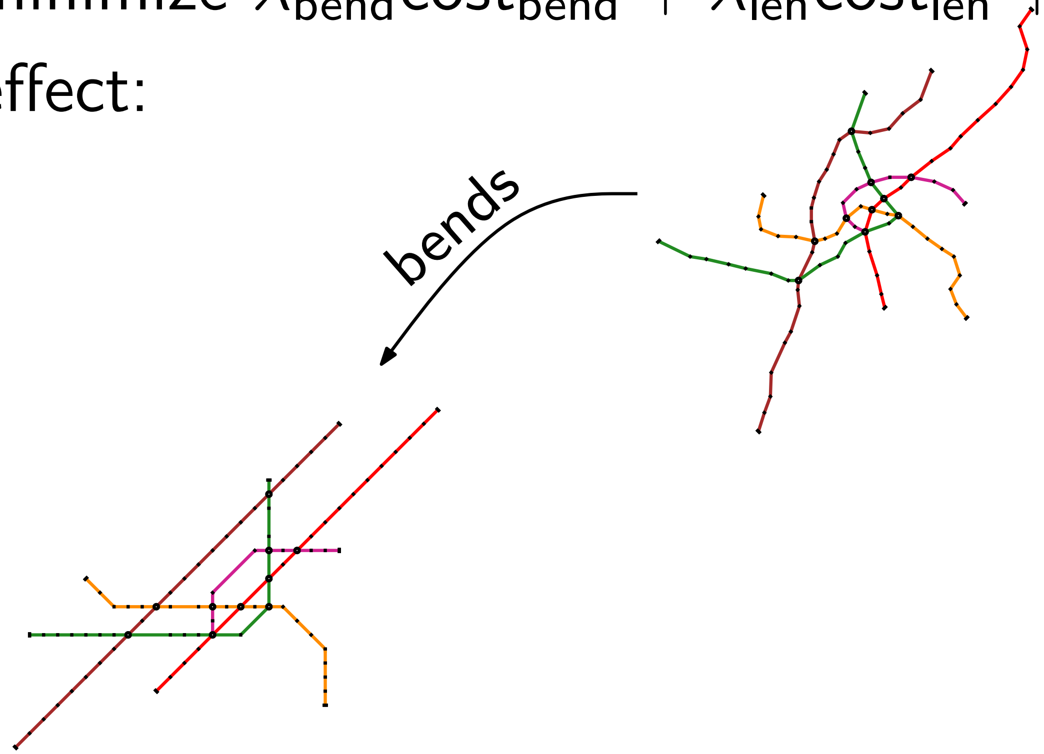
Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\text{bend}}\text{cost}_{\text{bend}} + \lambda_{\text{len}}\text{cost}_{\text{len}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$
- effect:



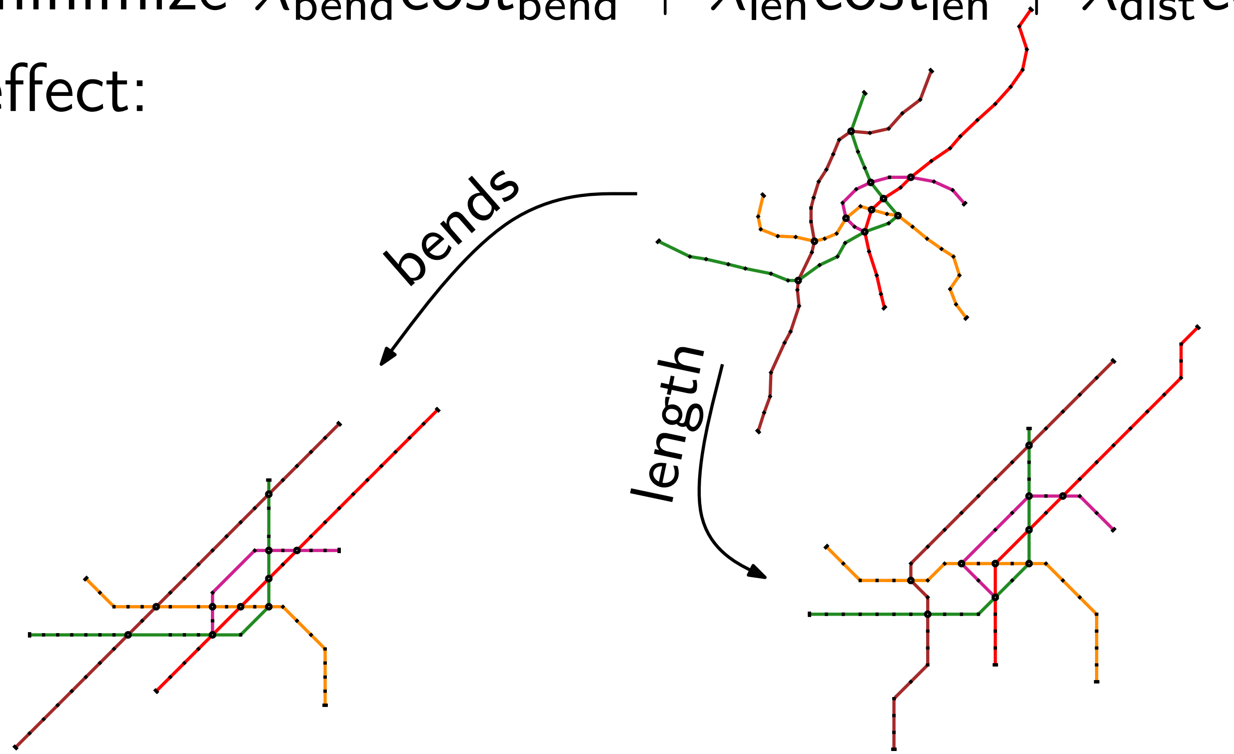
Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\mathsf{bend}}\mathsf{cost}_{\mathsf{bend}} + \lambda_{\mathsf{len}}\mathsf{cost}_{\mathsf{len}} + \lambda_{\mathsf{dist}}\mathsf{cost}_{\mathsf{dist}}$
- effect:



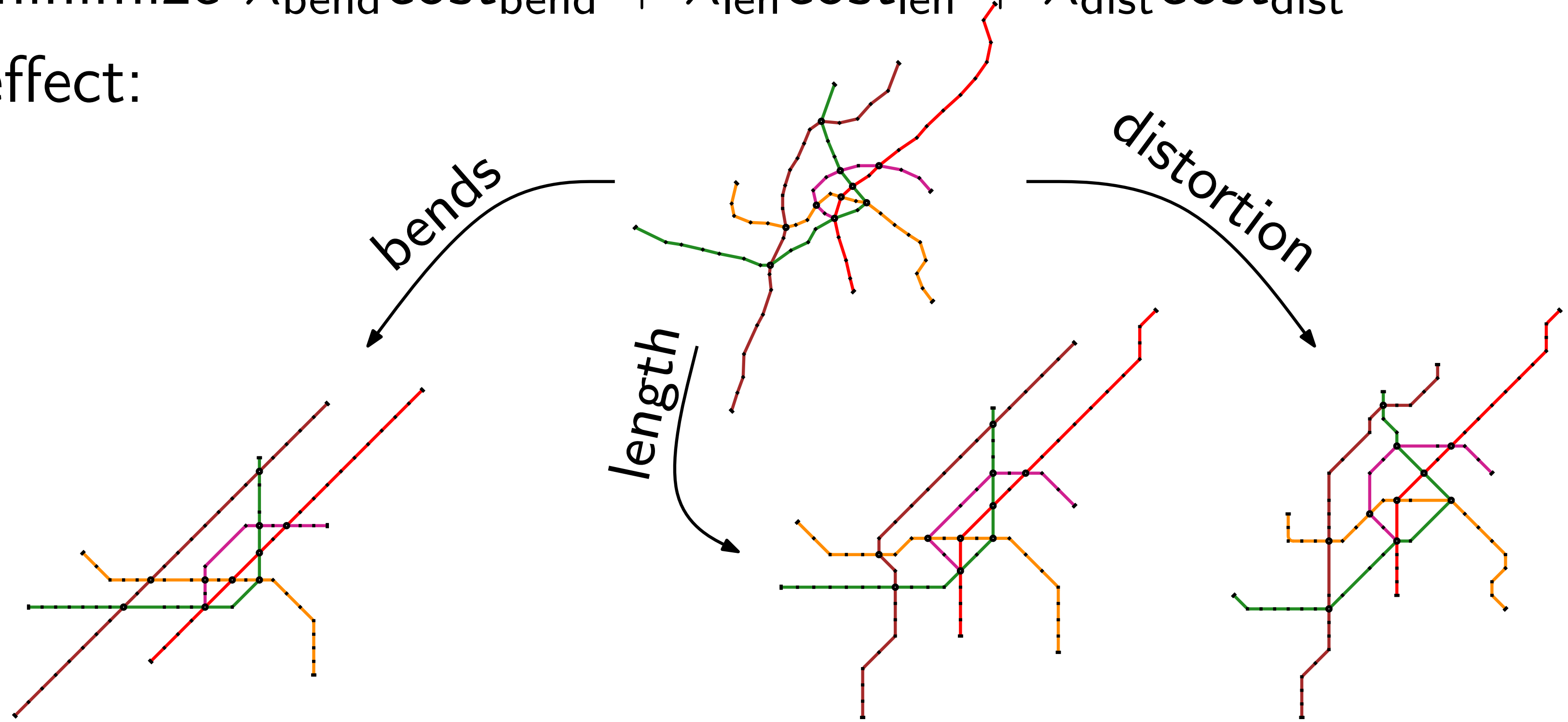
Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\text{bend}}\text{cost}_{\text{bend}} + \lambda_{\text{len}}\text{cost}_{\text{len}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$
- effect:



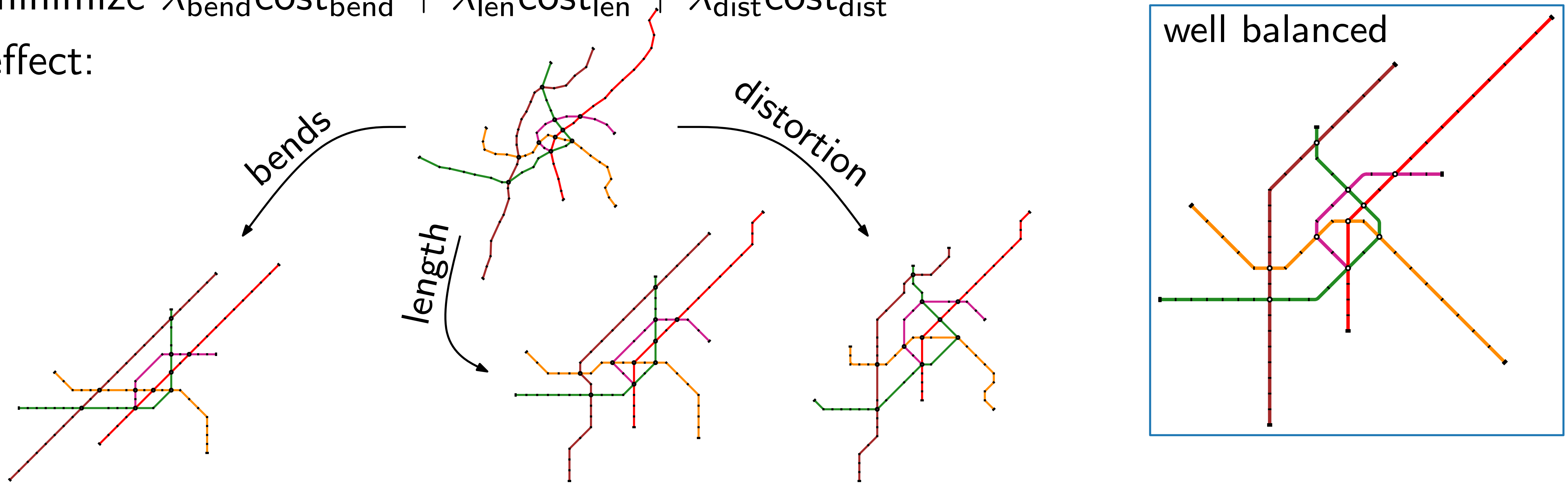
Overview MIP model

Constraints.

- linearization of all hard constraints
- $O(n^2)$ variables and constraints (due to planarity)

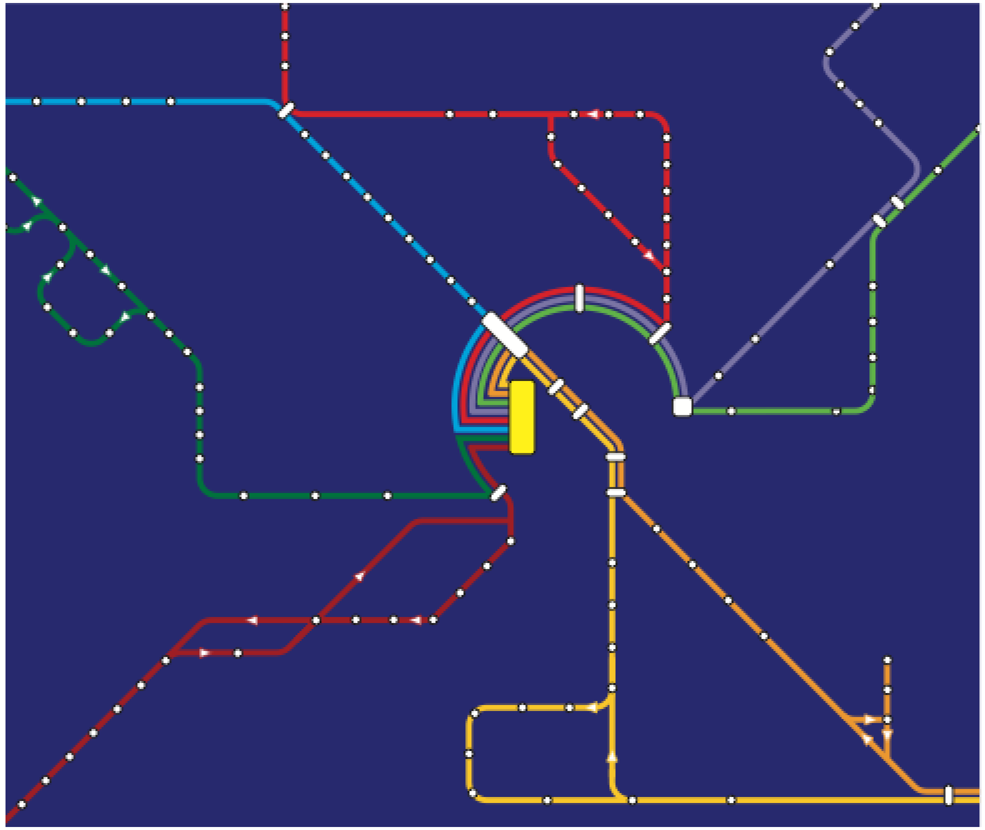
Objective function.

- weighted sum of the three soft constraints
- minimize $\lambda_{\text{bend}}\text{cost}_{\text{bend}} + \lambda_{\text{len}}\text{cost}_{\text{len}} + \lambda_{\text{dist}}\text{cost}_{\text{dist}}$
- effect:



Station Labeling

- unlabeled map mostly useless



Station Labeling

- unlabeled map mostly useless

Station Labeling

- unlabeled map mostly useless
- labels need space

Station Labeling

- unlabeled map mostly useless
- labels need space
- labels may not overlap each other

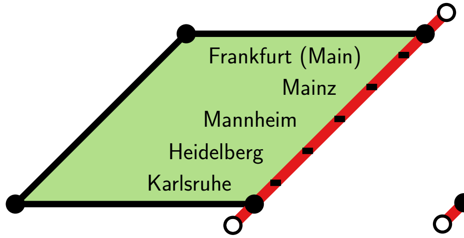
Station Labeling

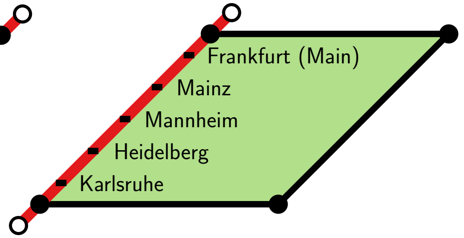
- unlabeled map mostly useless
- labels need space
- labels may not overlap each other
- graph labeling problem is NP-hard [Tollis & Kakoulis 2001]

Station Labeling

- unlabeled map mostly useless
- labels need space
- labels may not overlap each other
- graph labeling problem is NP-hard [Tollis & Kakoulis 2001]

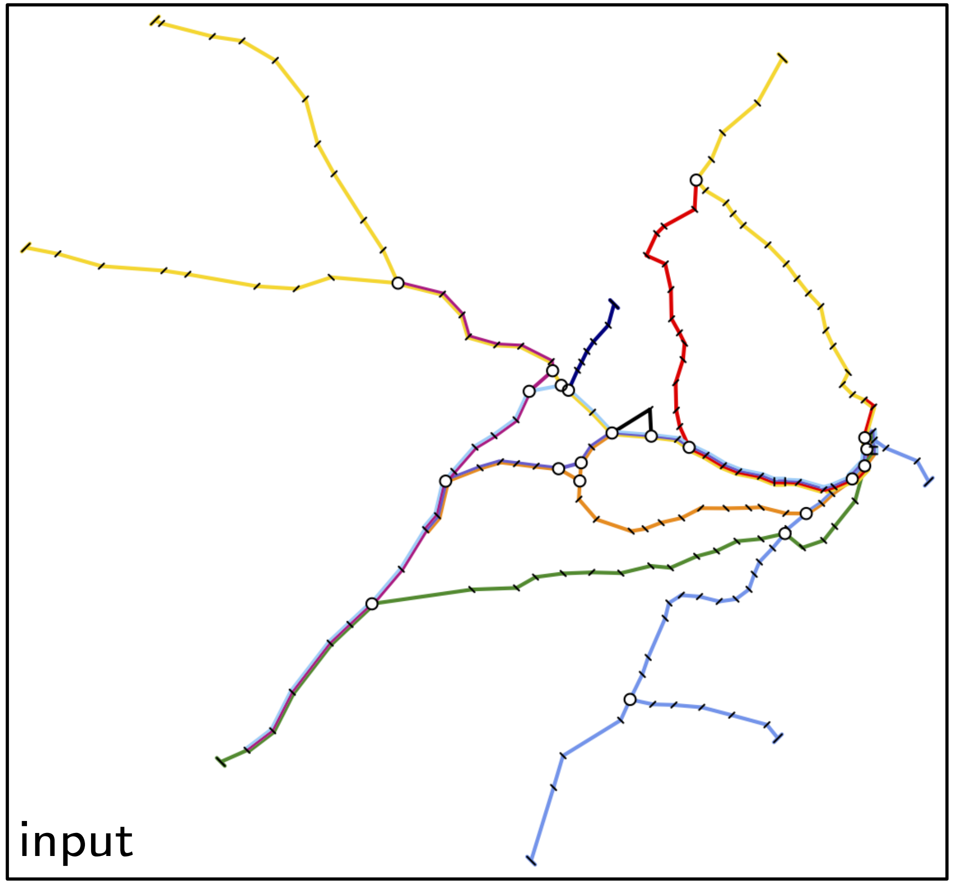
→ combine layout & labeling for optimal results!



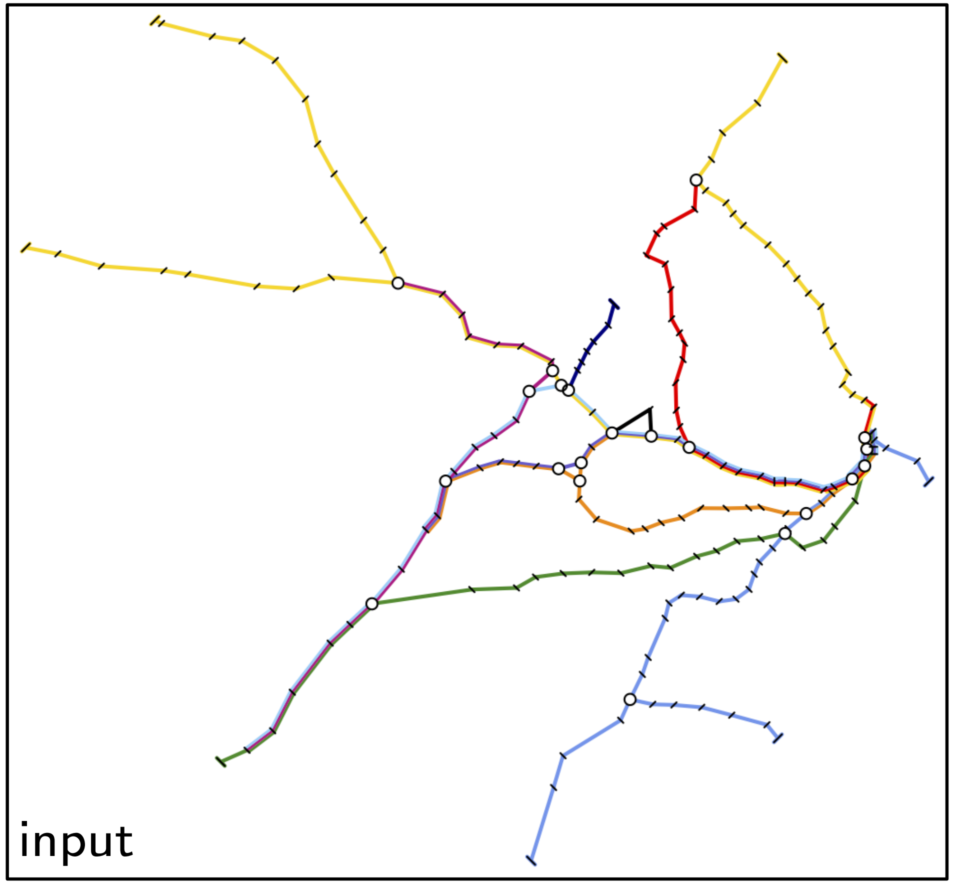


- parallelogram as special metro line
- switching sides allowed

Example: Sydney &

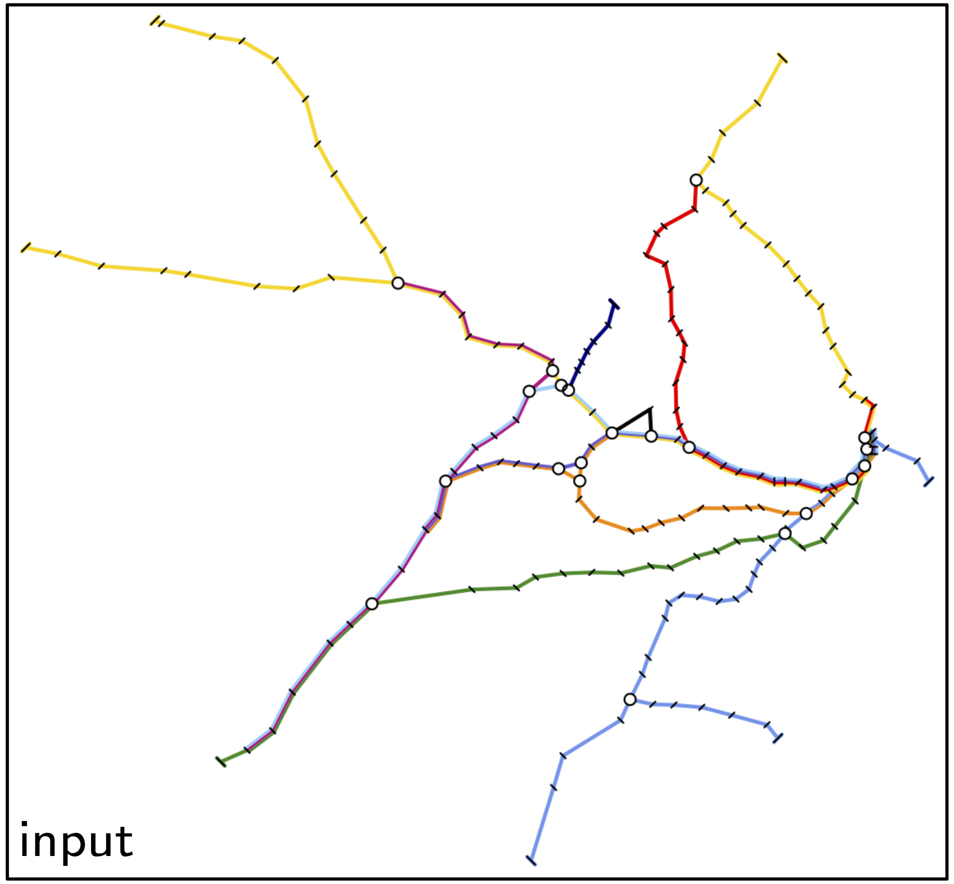


Example: Sydney &



Input	$\lvert V \rvert$	$\lvert E \rvert$	fcs.	$\lvert \mathcal{L} \rvert$
full	174	183	11	10
reduced	88	97		
labeled	242	270	30	

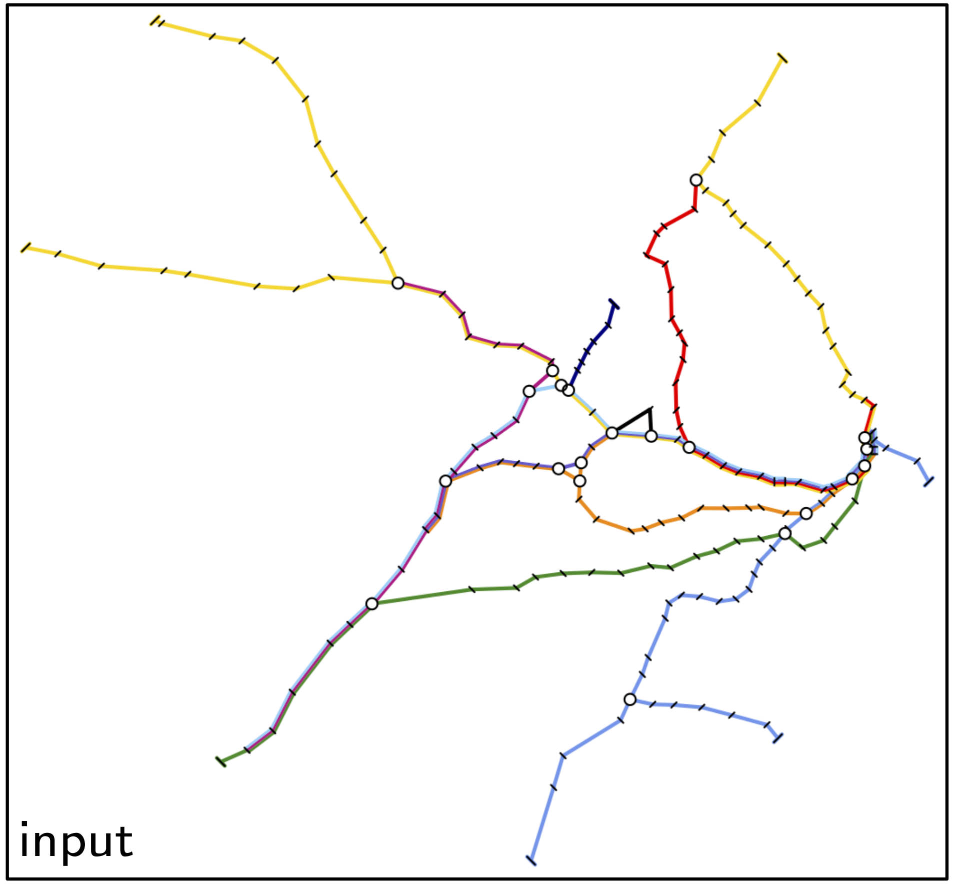
Example: Sydney &



Input	$\|V\|$	$\|E\|$	fcs.	$\|\mathcal{L}\|$
full	174	183	11	10
reduced	88	97		
labeled	242	270	30	

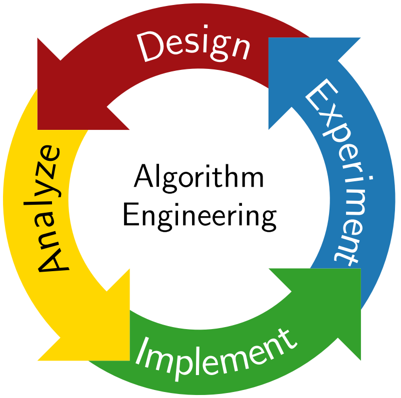
MIP	constraints	variables
full	1,191,406	290,137
callback	21,988	92,681

Example: Sydney &

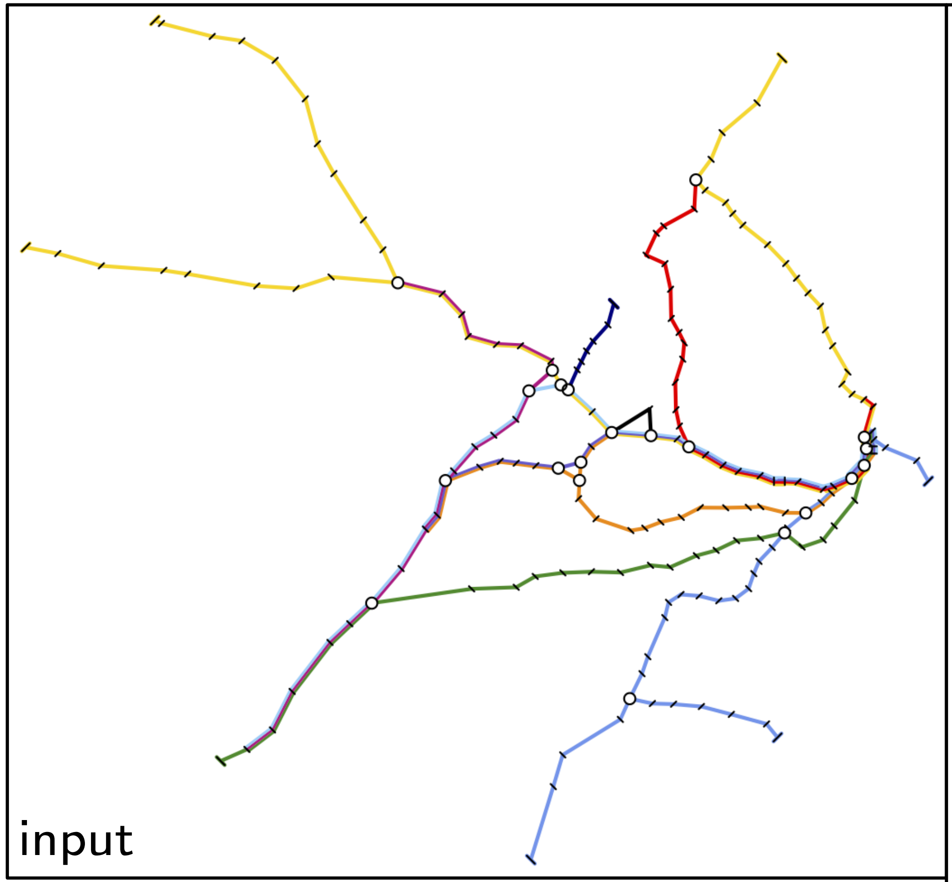


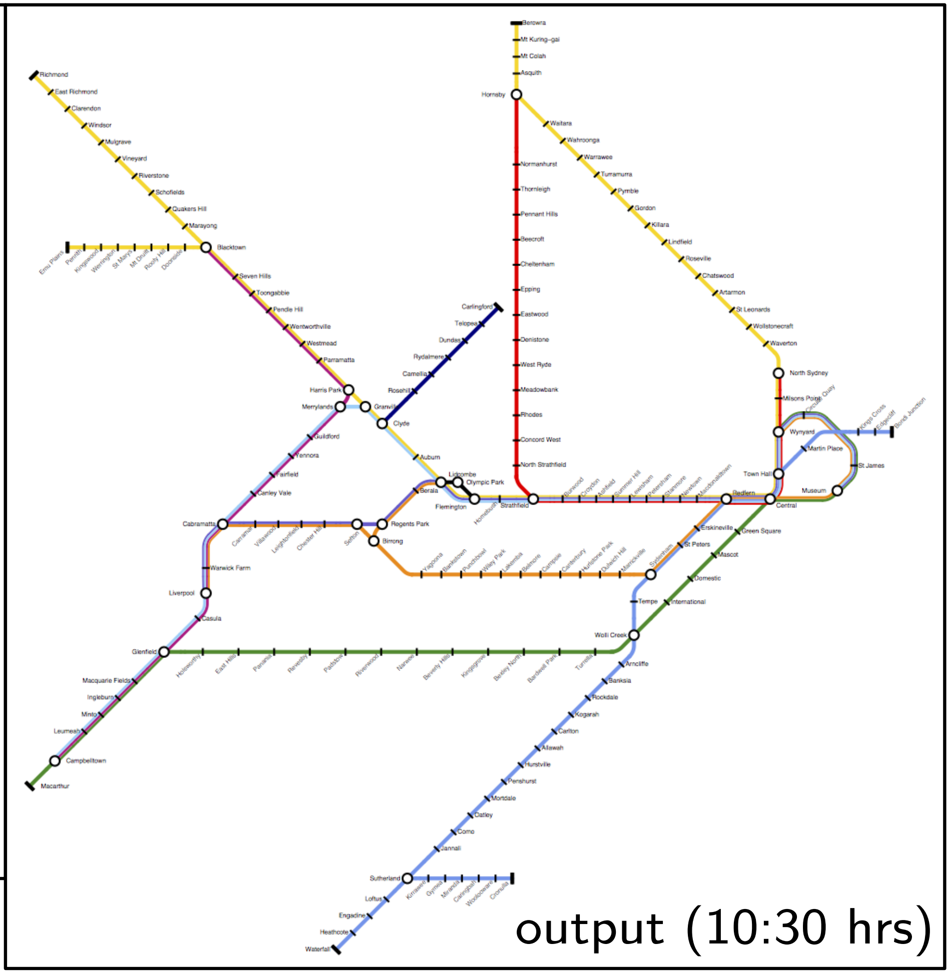
Input	$\lvert V \rvert$	$\lvert E \rvert$	fcs.	$\lvert \mathcal{L} \rvert$
full	174	183	11	10
reduced	88	97		
labeled	242	270	30	

MIP	constraints	variables
full	1,191,406	290,137
callback	21,988	92,681

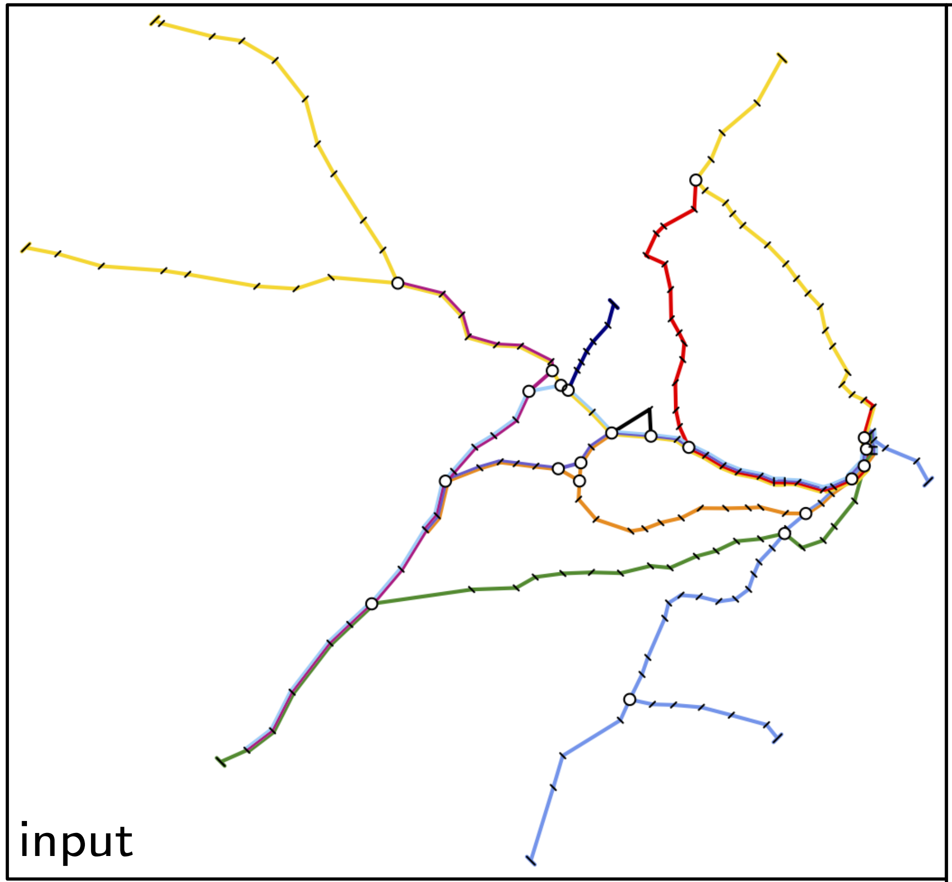


Example: Sydney &

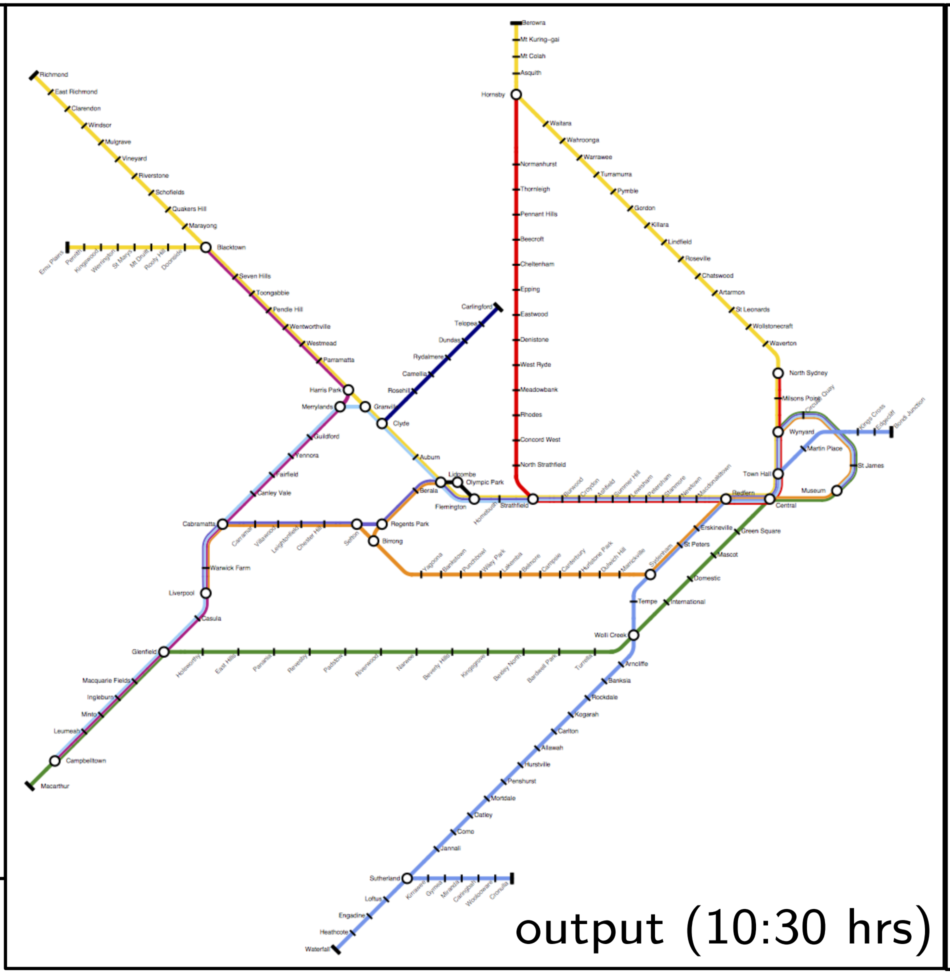




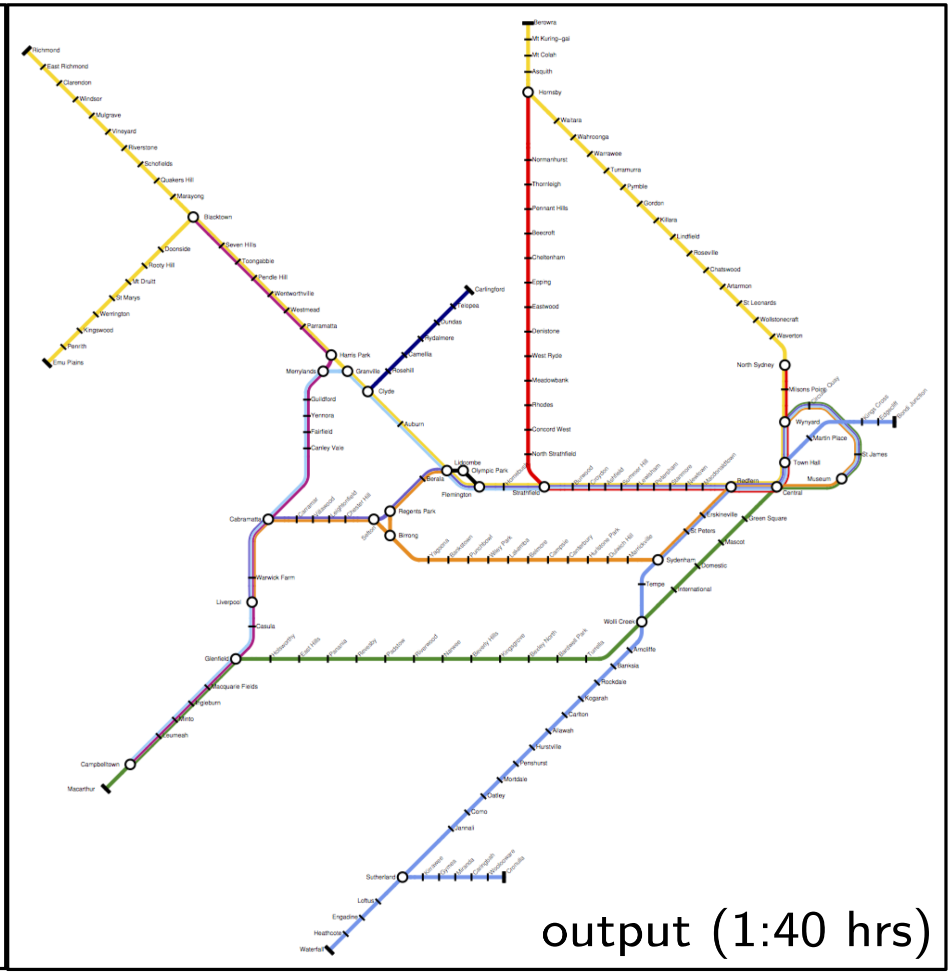
Example: Sydney &



input

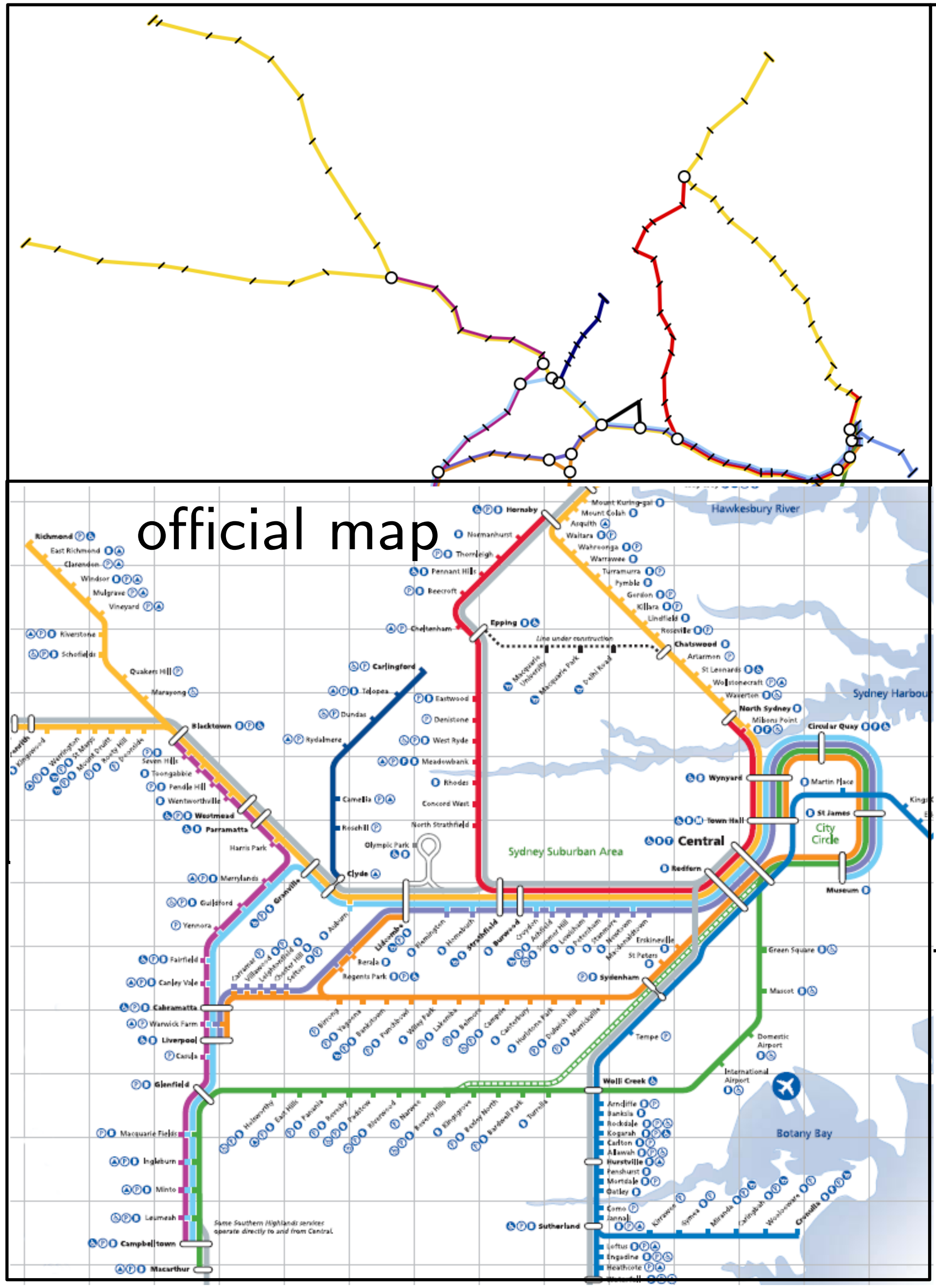


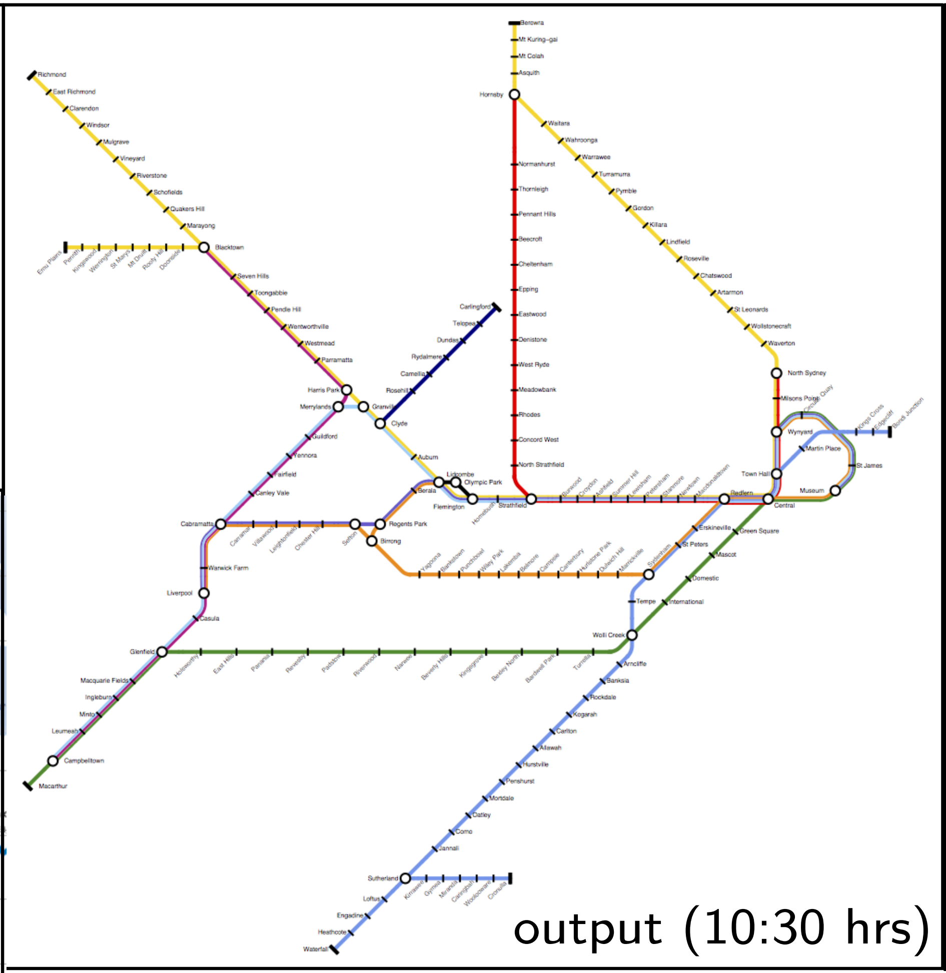
output (10:30 hrs)

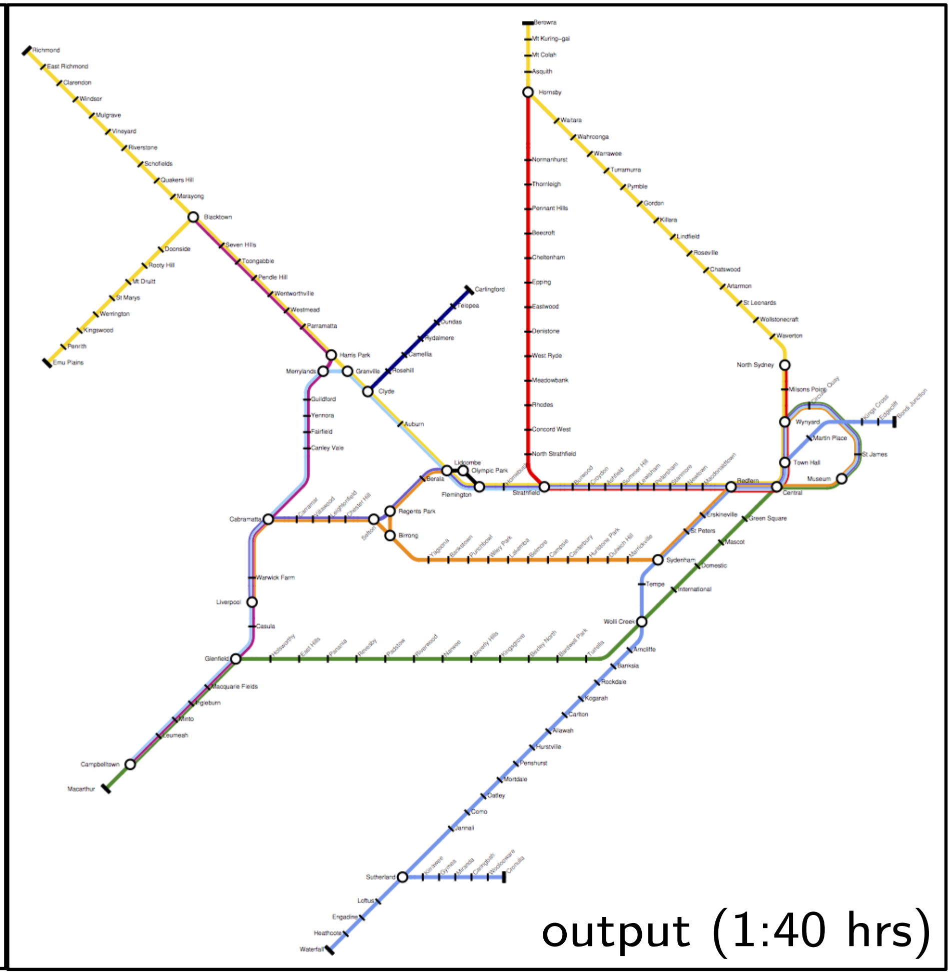


output (1:40 hrs)

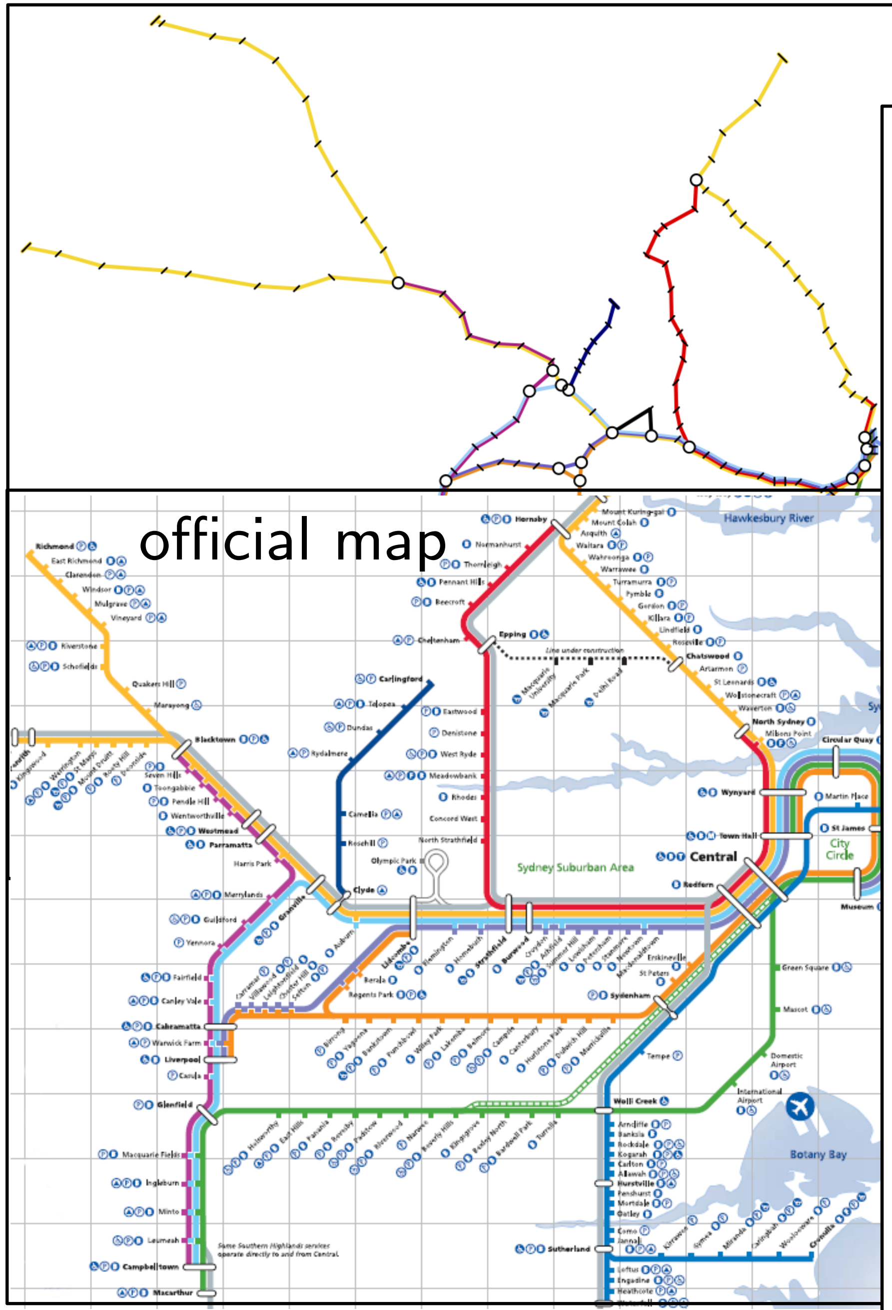
Example: Sydney &

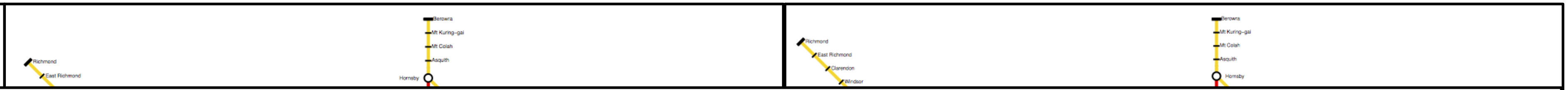

official map


output (10:30 hrs)

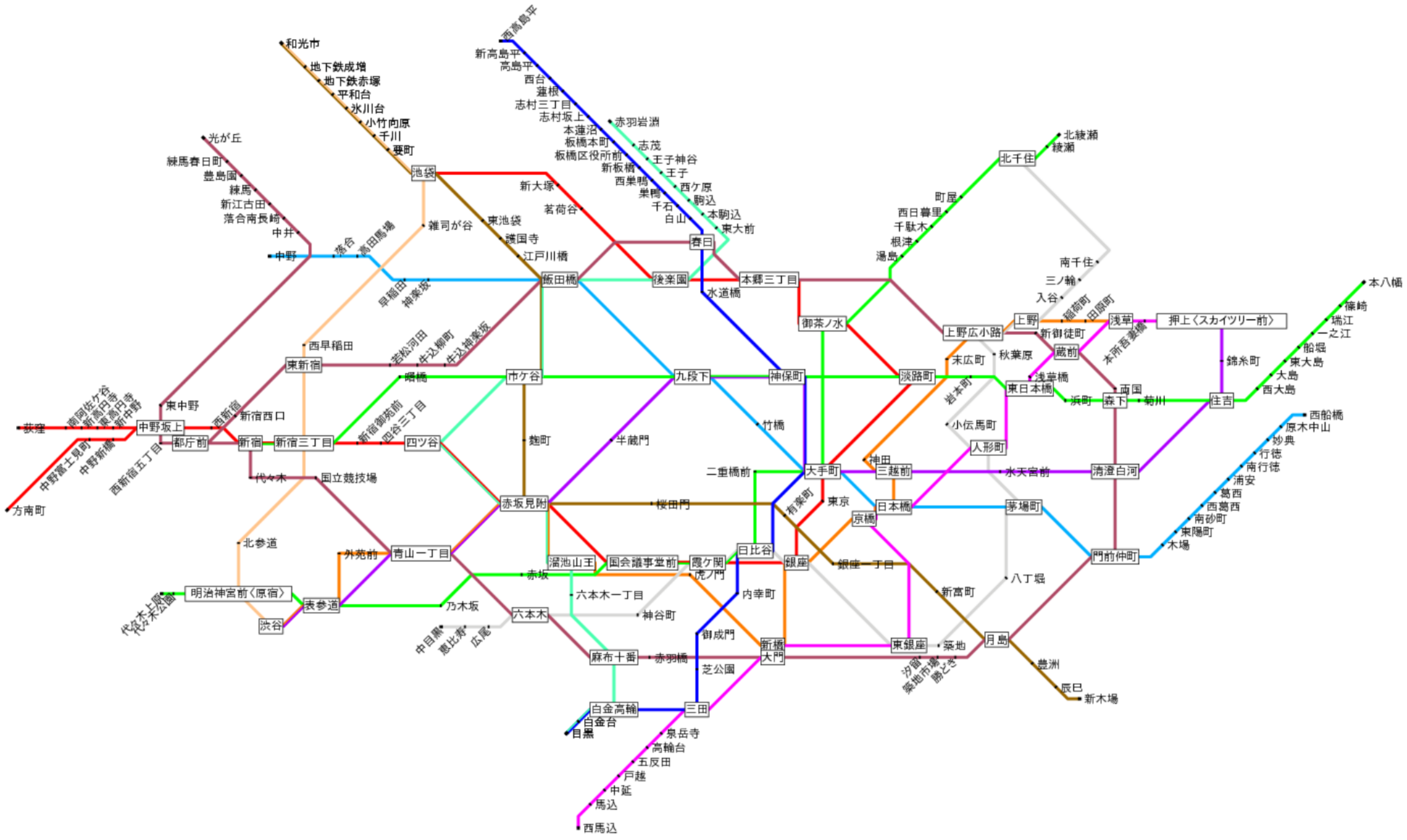

output (1:40 hrs)

Example: Sydney & Tokyo





Tokyo



- multi-phase MIP model [Onda et al. 2018]

5 hrs

Mixed-Integer Programming – Discussion

Pros.

Cons.

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort

Cons.

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality

Cons.

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality
- theoretical guarantees

Cons.

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality
- theoretical guarantees
- can integrate user constraints dynamically

Cons.

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality
- theoretical guarantees
- can integrate user constraints dynamically

Cons.

- long, sometimes unpredictable running times

Mixed-Integer Programming – Discussion

Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality
- theoretical guarantees
- can integrate user constraints dynamically

Cons.

- long, sometimes unpredictable running times
- for large labeled networks no proof of optimality

Mixed-Integer Programming – Discussion

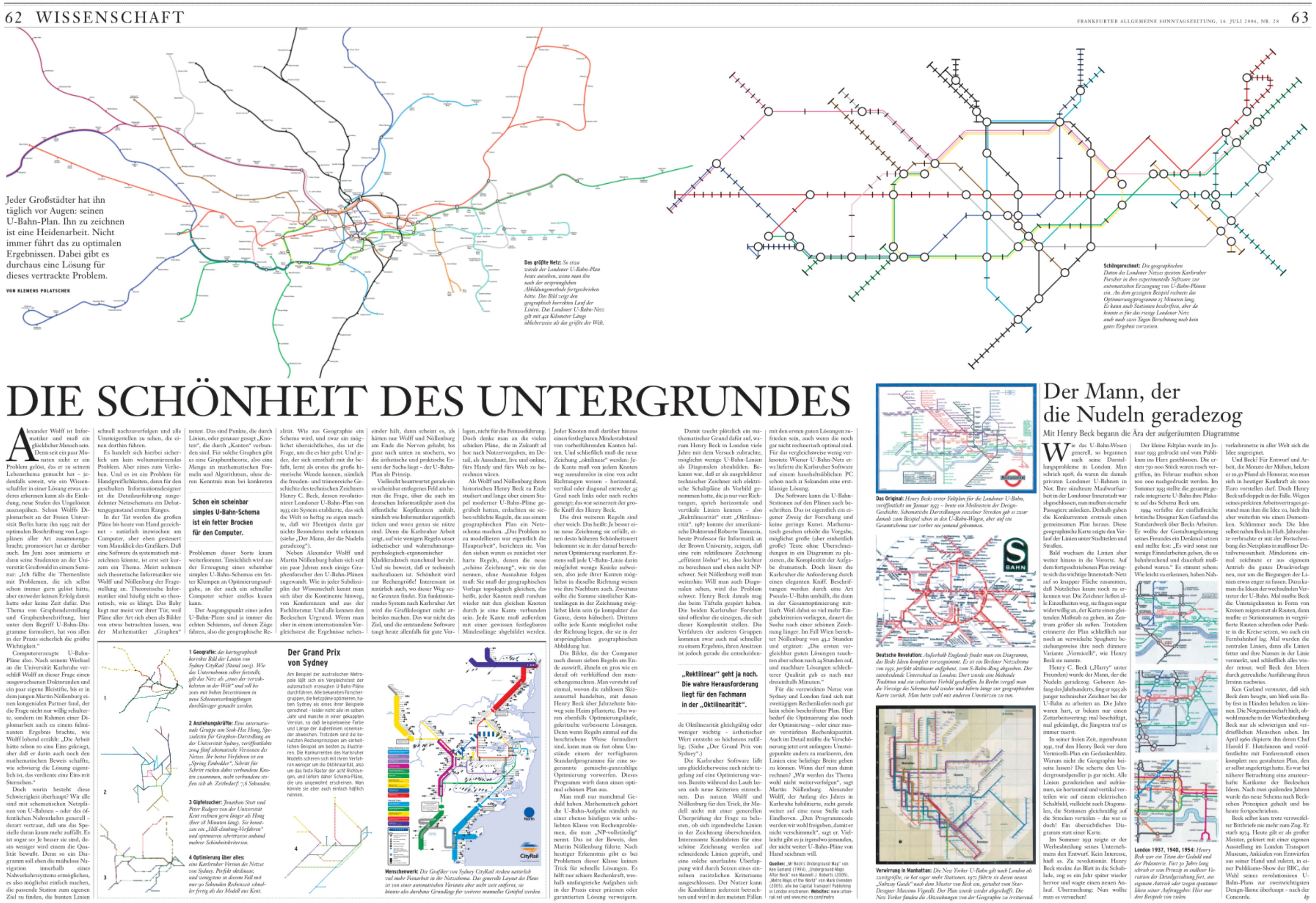
Pros.

- flexible framework, but integration and linearization of new criteria requires some effort
- high layout and labeling quality
- theoretical guarantees
- can integrate user constraints dynamically

Cons.

- long, sometimes unpredictable running times
- for large labeled networks no proof of optimality
- solutions only as good as the model specification

Mixed-Integer Programming – Discussion


62 WISSENSCHAFT

FRANKFURTER ALLGEMEINE SONNTAGSZEITUNG, 16. JULI 2006, NR. 28 63

Jeder Großstädter hat ihn täglich vor Augen: seinen U-Bahn-Plan. Ihn zu zeichnen ist eine Heidenarbeit. Nicht immer führt das zu optimalen Ergebnissen. Dabei gibt es durchaus eine Lösung für dieses vertrackte Problem.

VON KLEMENS POLATSCHEK

DIE SCHÖNHEIT DES UNTERGRUNDES

Schon ein scheinbar simples U-Bahn-Schema ist ein fetter Brocken für den Computer.

Der Grand Prix von Sydney

„Rektilinear" geht ja noch. Die wahre Herausforderung liegt für den Fachmann in der „Oktilinearität".

Der Mann, der die Nudeln geradezog

Mit Henry Beck begann die Ära der aufgeräumten Diagramme

Least-Squares Schematization

[Wang, Chi '11]

Idea.

- model layout problem as minimization of a set of (squared) energy terms
- variables for vertex positions and edge slopes
- use iterative numerical optimization method

Least-Squares Schematization

[Wang, Chi '11]

Idea.
- model layout problem as minimization of a set of (squared) energy terms
- variables for vertex positions and edge slopes
- use iterative numerical optimization method

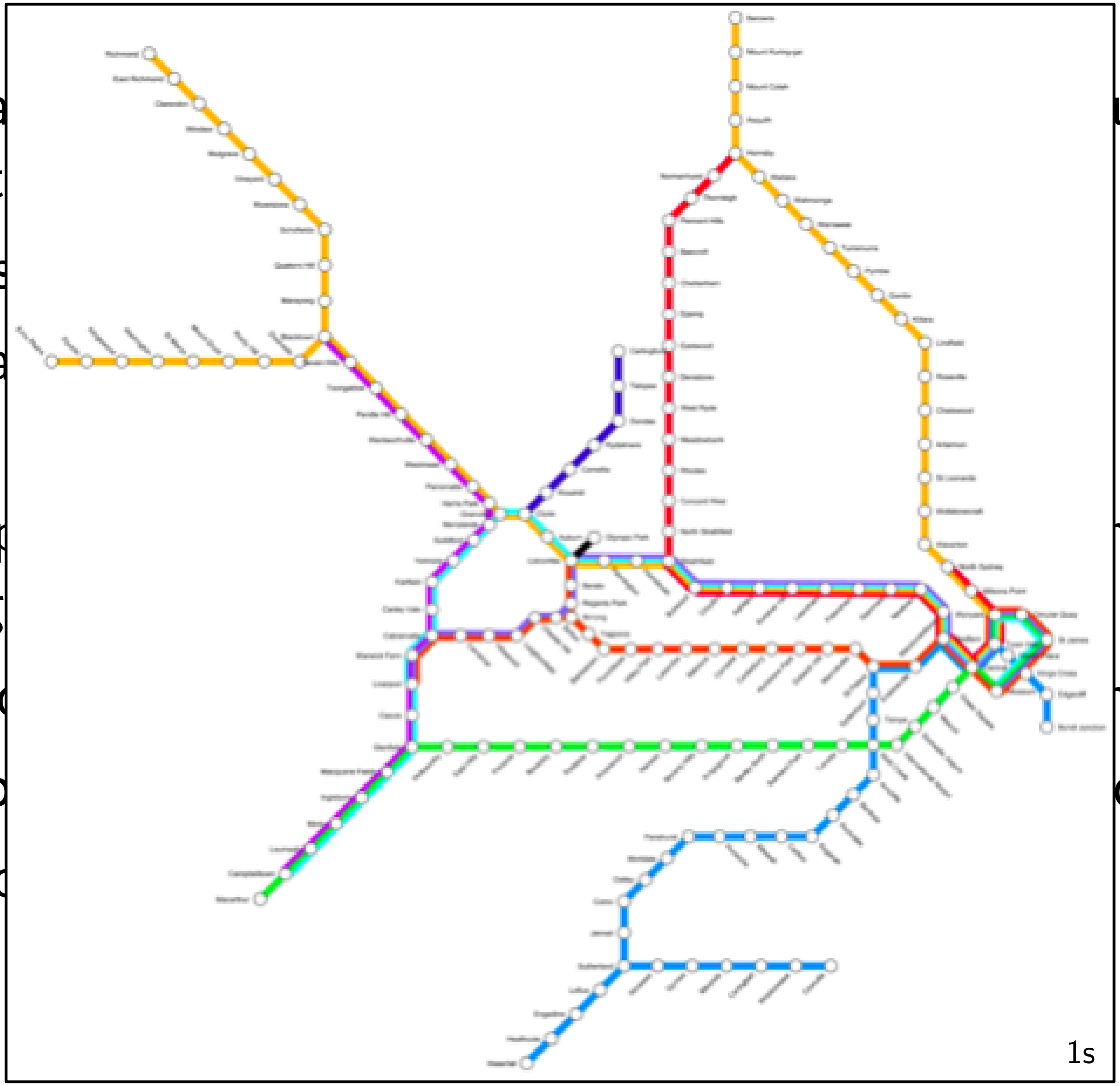
- 3-step approach
  - compute topologically correct non-octilinear layout optimizing angular resolution **(R5)**, uniform edge lengths **(R7)**, displacement **(R6)**
  - octilinearly discretize edge orientations **(R2)** by extra energy term
  - optimize label placement by energy minimization in fixed layout
- described for focus route, generalizes to entire maps

Least-Squares Schematization

[Wang, Chi '11]

Idea.
- model la … uared) energy t
- variables
- use itera

- 3-step approac
  - compute top … mizing angular resolution **(R5)**, uniform edg
  - octilinearly … nergy term
  - optimize lab … d layout
- described for fo



Least-Squares Schematization

[Wang, Chi '11]

Idea.
- model layout problem as minimization of a set of (squared) energy terms
- variables for vertex positions and edge slopes
- use iterative numerical optimization method

- 3-step approach
  - compute topologically correct non-octilinear layout optimizing angular resolution **(R5)**, uniform edge lengths **(R7)**, displacement **(R6)**
  - octilinearly discretize edge orientations **(R2)** by extra energy term
  - optimize label placement by energy minimization in fixed layout
- described for focus route, generalizes to entire maps

Discussion.
- very fast method for good quality layouts
- no guarantee on constraints unless final energy is zero

Summary

- variety of layout methods evolved over the last 15-20 years

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

- base layouts for graphic designers (semi-automated process)

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

- base layouts for graphic designers (semi-automated process)
- large quantities of individual or special-purpose maps

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

- base layouts for graphic designers (semi-automated process)
- large quantities of individual or special-purpose maps

Challenges

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

- base layouts for graphic designers (semi-automated process)
- large quantities of individual or special-purpose maps

Challenges

- global quality criteria like harmony, coherence, balance

Summary

- variety of layout methods evolved over the last 15-20 years
- many shared design rules
- trade-off between speed and quality, but quite reasonable maps can be computed in a matter of seconds to minutes
- many approaches are customizable and open to new criteria
- current trend: beyond octilinear metro maps

Why automated maps?

- base layouts for graphic designers (semi-automated process)
- large quantities of individual or special-purpose maps

Challenges

- global quality criteria like harmony, coherence, balance
- edge bundles and large vertices

Literature

- [Nöllenburg '14] A Survey on Automated Metro Map Layout Methods
- [Dwyer, Hurst, Merrick '08] A fast and simple heuristic for metro map path simplification
- [Delling et al. '14] On d-regular schematization of embedded paths
- [Brandes & Pampel '09] On the Hardness of Orthogonal-Order Preserv-ing Graph Drawing
- [Gema et al. '11] On d-Regular Schematization of Embedded Paths
- [Hong et al. '06] Automaticvisualisation of metro maps
- [Chivers & Rodgers '14] Octilinear Force-Directed Layout with Mental MapPreservation for Schematic Diagrams
- [Fink, Haverkort, Nöllenburg, Roberts, Schuhmann, Wolff '12] Drawing metro maps using Bézier curves
- [Avelar & Müller '00] Generating topologically correct schematicmaps
- [Ware et al. '06] Automatedproduction of schematic maps for mobile application
- [Ware & Richards '13] An ant colony system algorithm forautomatically schematizing transport network data set
- [Stott et al. '11] Auto-matic metro map layout using multicriteria optimization
- [Nöllenburg & Wolff '11] Drawing and labeling high-quality metro maps by mixed-integer programming
- [Onda, Moriguchi, Imai '18] Automatic Drawing for Tokyo Metro Map
- [Wang & Chi '11] Focus+context metro maps