Octilinear Graph Drawing: Metro Map Layout

Lecture Graph Drawing Algorithms · 192.053

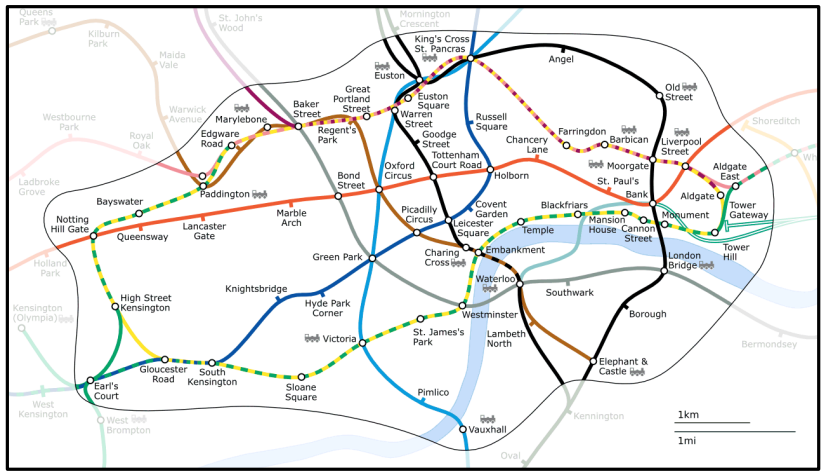
Martin Nöllenburg

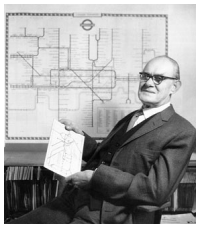
11.06.2018

TU WIEN

ac ALGORITHMS AND COMPLEXITY GROUP

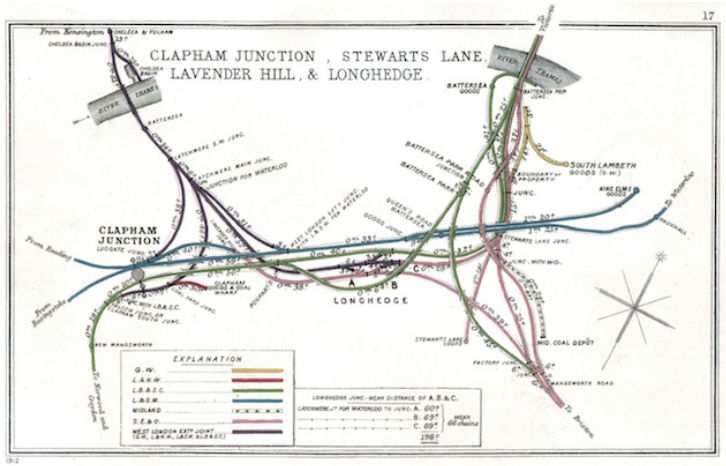
What is a Schematic Metro Map?





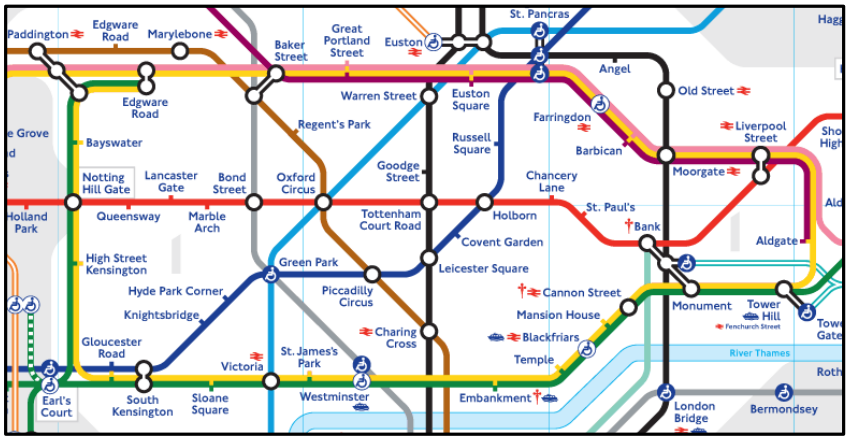
- map/diagram that shows stations connected by metro lines
- focus on topology rather than topography
- goal: easy-to-use visual navigation aid for passengers
  - „*How do I quickly get from A to B?*“
  - „*Where do I need to change trains?*“
- distorts scale and geometry
- metro map design still a largely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

Subtasks in Metro Map Layout



+

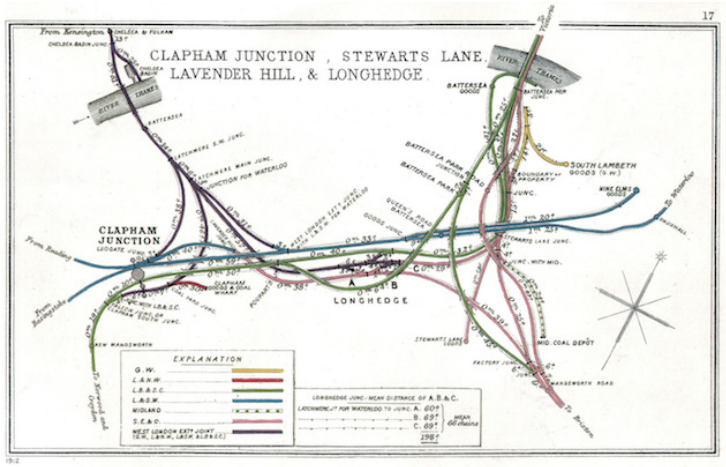
→



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

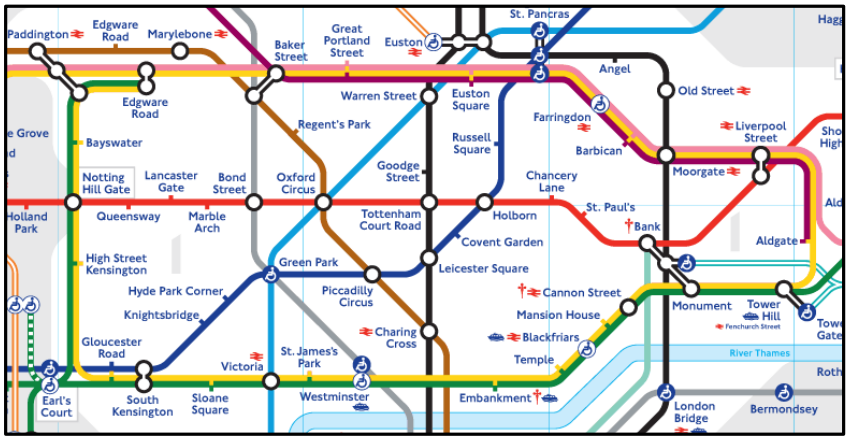
output:
- **optimum** metro map layout (whatever it means)

Subtasks in Metro Map Layout



+

→

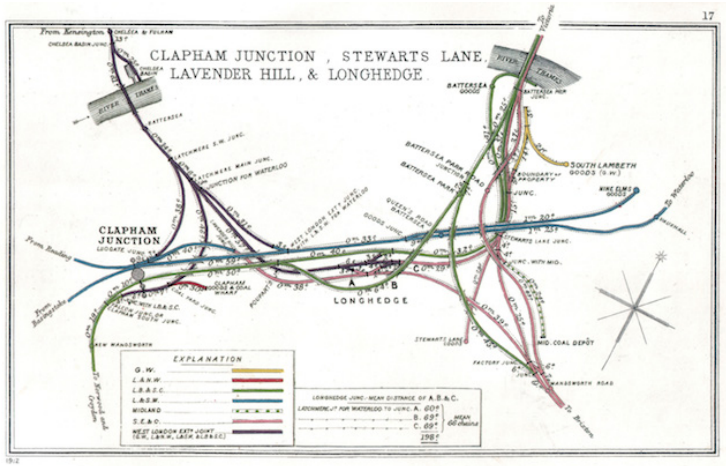


input:
- geographically embedded (railway) network G
- set of metro lines $\mathcal{L}$ serving G

output:
- **optimum** metro map layout (whatever it means)

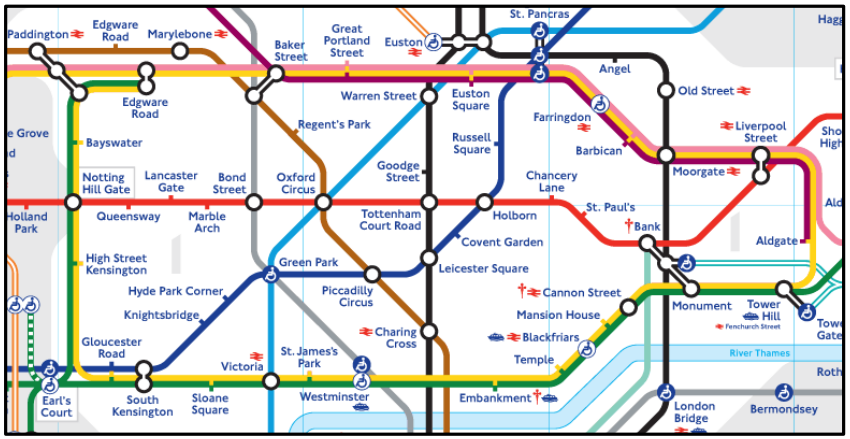
divide the task into several subtasks:
- rendering and design choices
- network layout
- station labeling
- metro line routing

Subtasks in Metro Map Layout



+

→



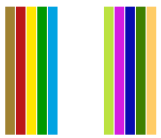
input:
- geographically embedded (railway) network G
- set of metro lines $\mathcal{L}$ serving G

output:
- **optimum** metro map layout (whatever it means)

divide the task into several subtasks:

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

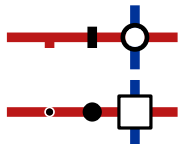
colors



fonts

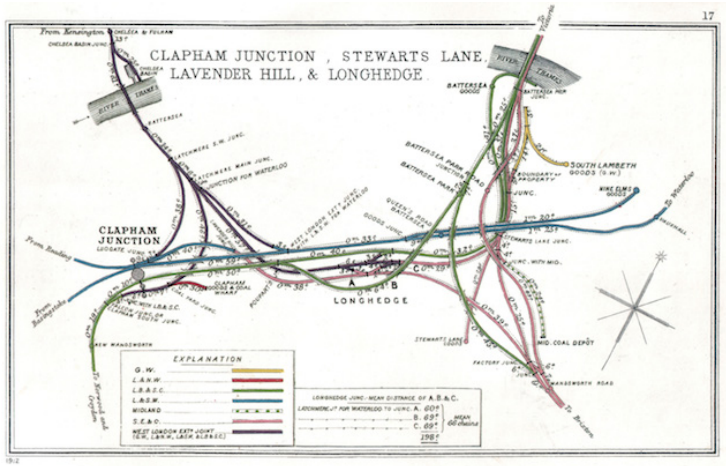
Paddington
PADDINGTON
Paddington

symbols



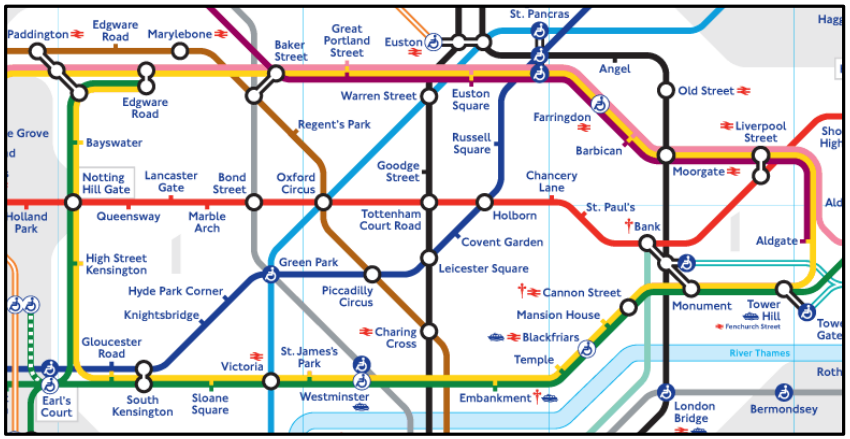
→ very salient,
but not a computational problem

Subtasks in Metro Map Layout



+

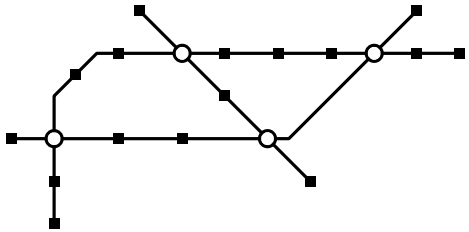
→



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

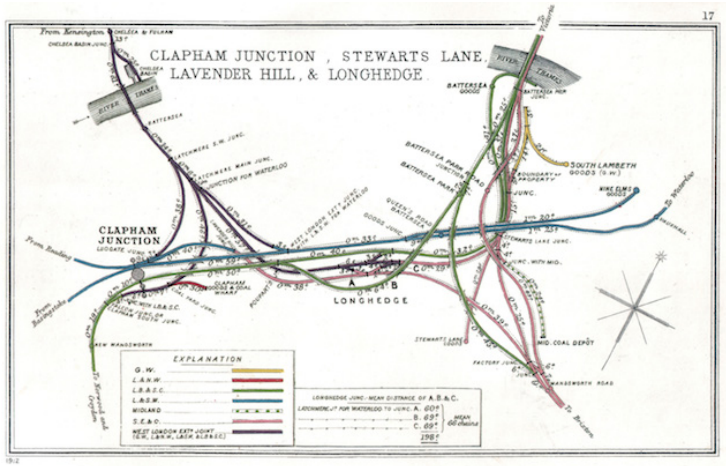
output:
- **optimum** metro map layout (whatever it means)

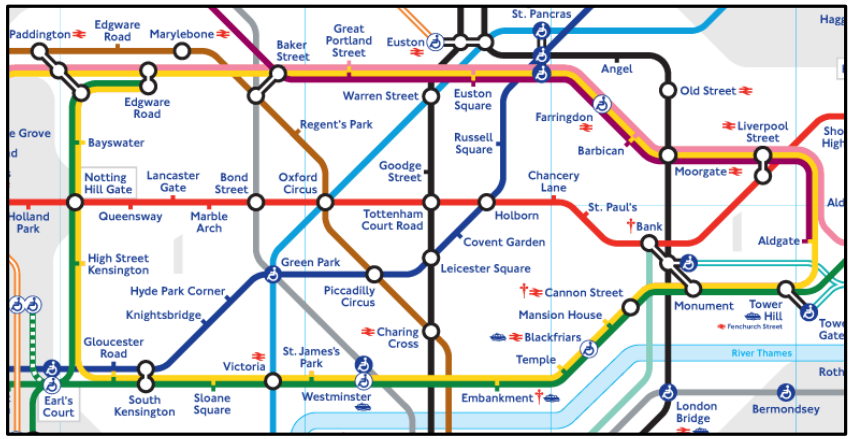
divide the task into several subtasks:
- rendering and design choices
- network layout
- station labeling
- metro line routing



determine geometry of network layout

Subtasks in Metro Map Layout



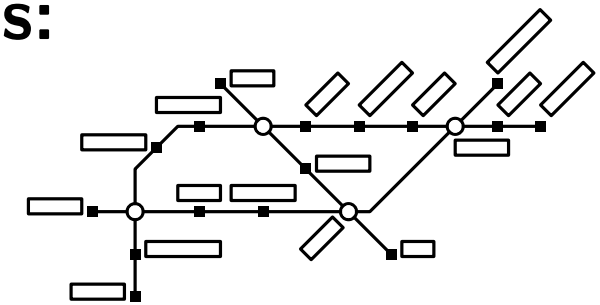


input:
- geographically embedded (railway) network G
- set of metro lines $\mathcal{L}$ serving G

output:
- **optimum** metro map layout (whatever it means)

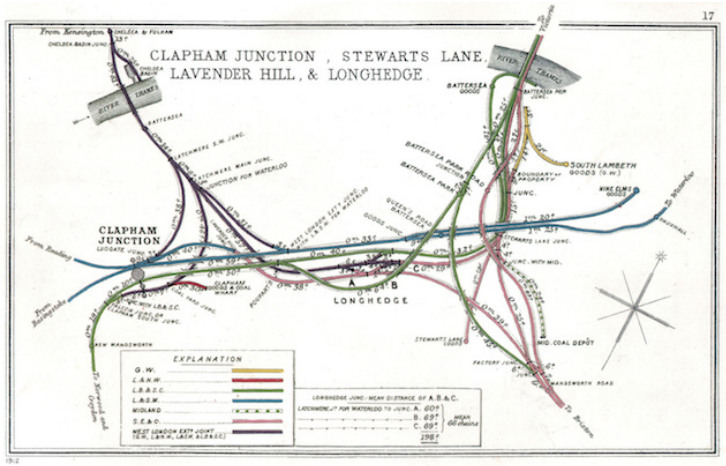
divide the task into several subtasks:

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



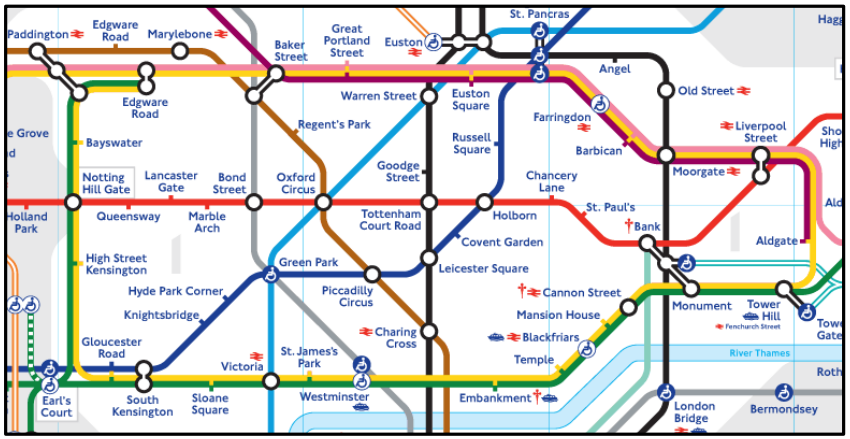
determine positions of station names

Subtasks in Metro Map Layout



+

→

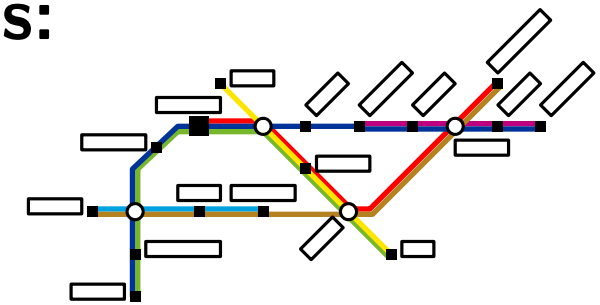


input:
- geographically embedded (railway) network G
- set of metro lines $\mathcal{L}$ serving G

output:
- **optimum** metro map layout (whatever it means)

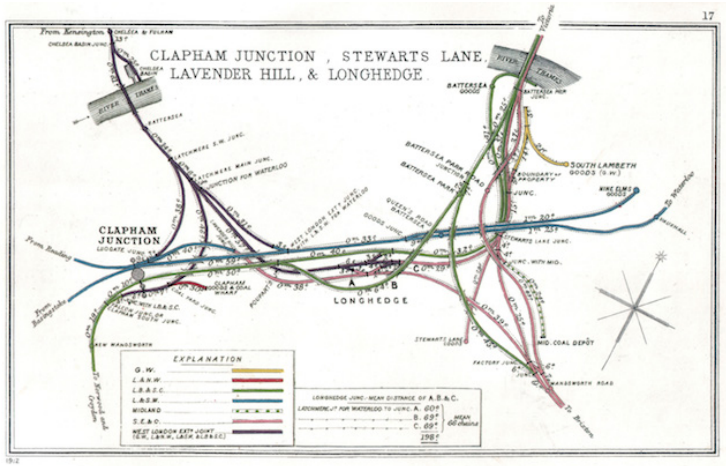
divide the task into several subtasks:

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



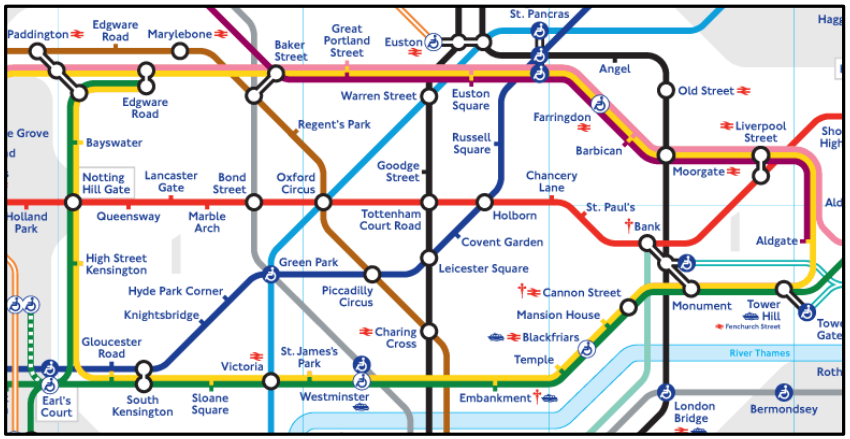
determine line routing and ordering of bundles

Subtasks in Metro Map Layout



+

→

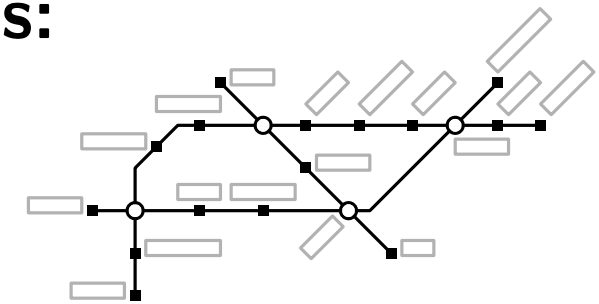


input:
- geographically embedded (railway) network G
- set of metro lines $\mathcal{L}$ serving G

output:
- **optimum** metro map layout (whatever it means)

divide the task into several subtasks:
- rendering and design choices
- network layout
- station labeling
- metro line routing

focus of this talk



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?

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

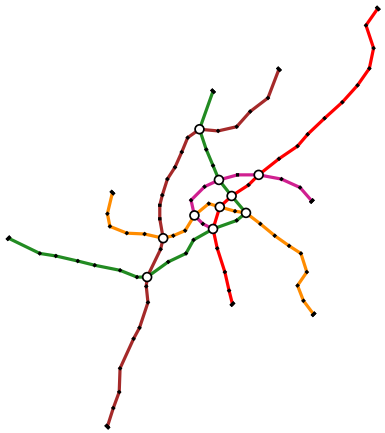
But what are the constraints and quality criteria?

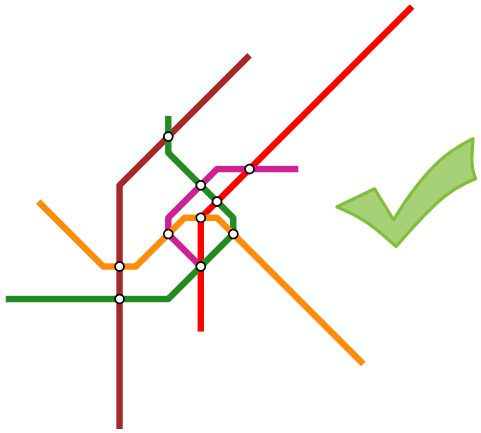
$\rightarrow$ extract common principles of existing, manually designed metro maps

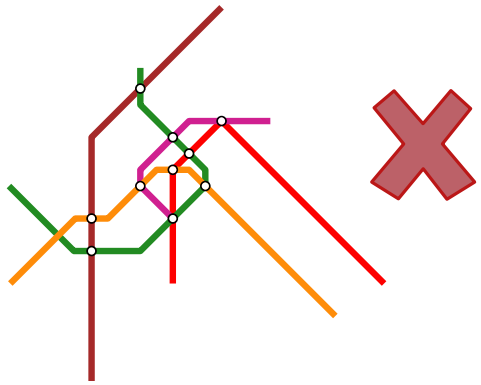
Design Rules

(R1) Do not change the network topology.

- no new crossings
- no changes in circular vertex orders





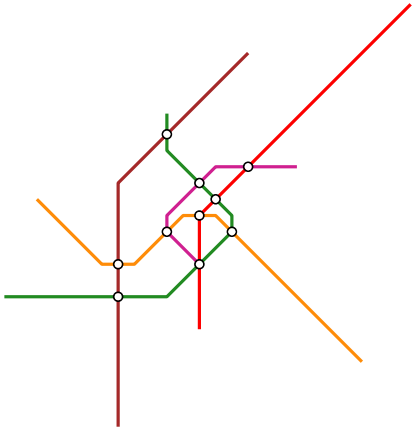


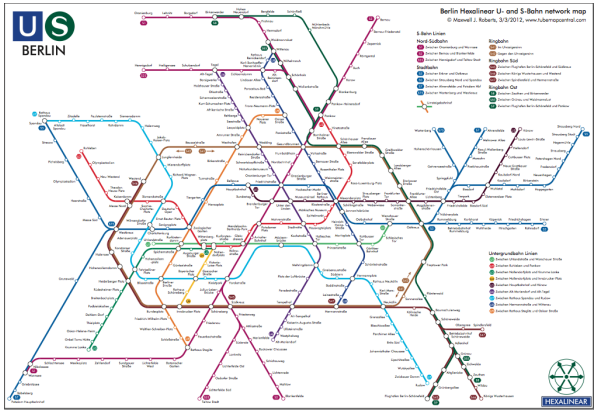
Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

- mostly octilinear orientation systems
- also curvilinear and other alternative orientation systems





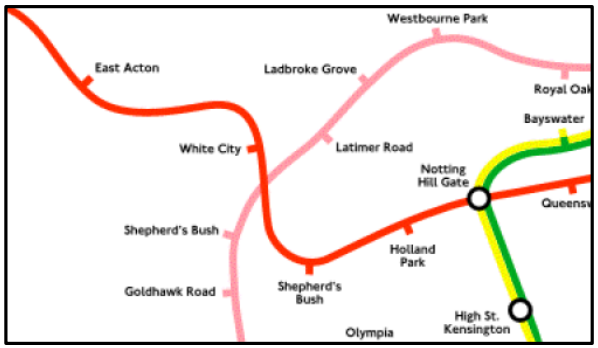
Design Rules

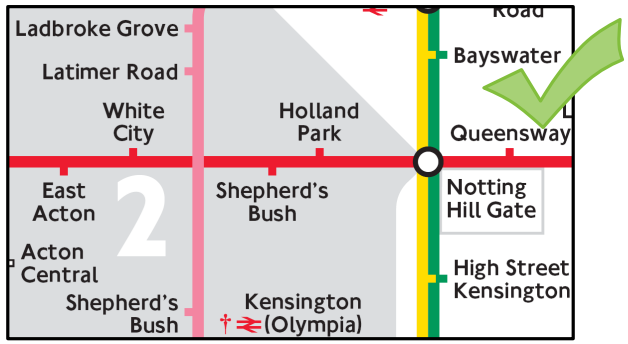
(R1) Do not change the network topology.

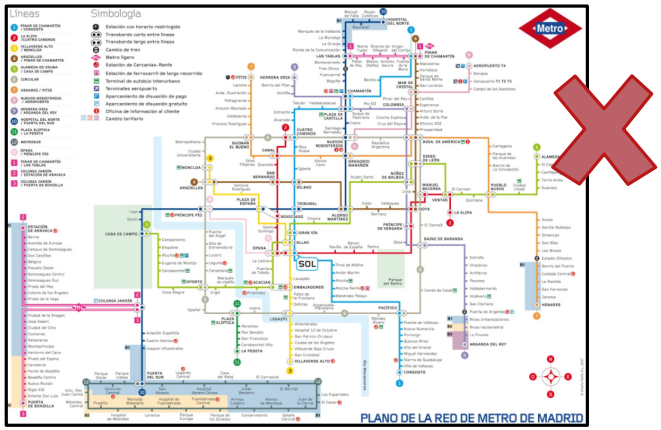
(R2) Restrict edge orientations.

(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

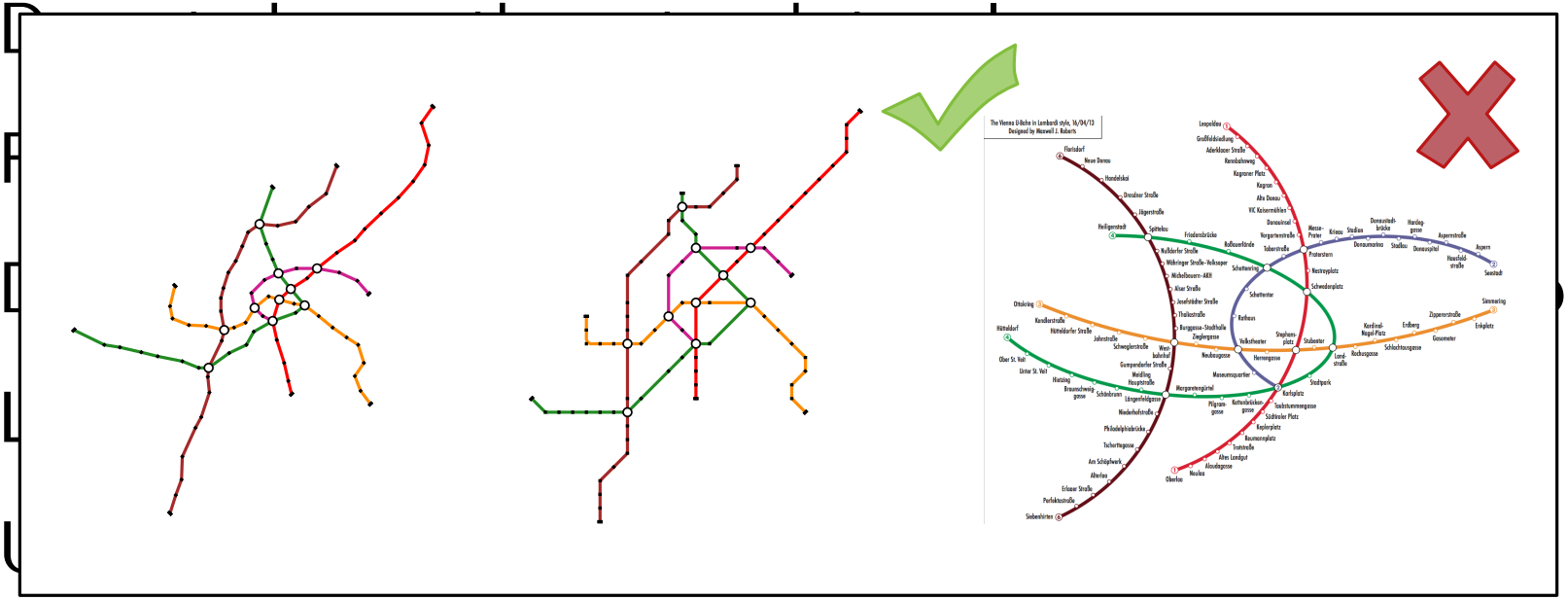
(R1) D

(R2) F

(R3) D ble.

(R4) L

(R5) U



(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.

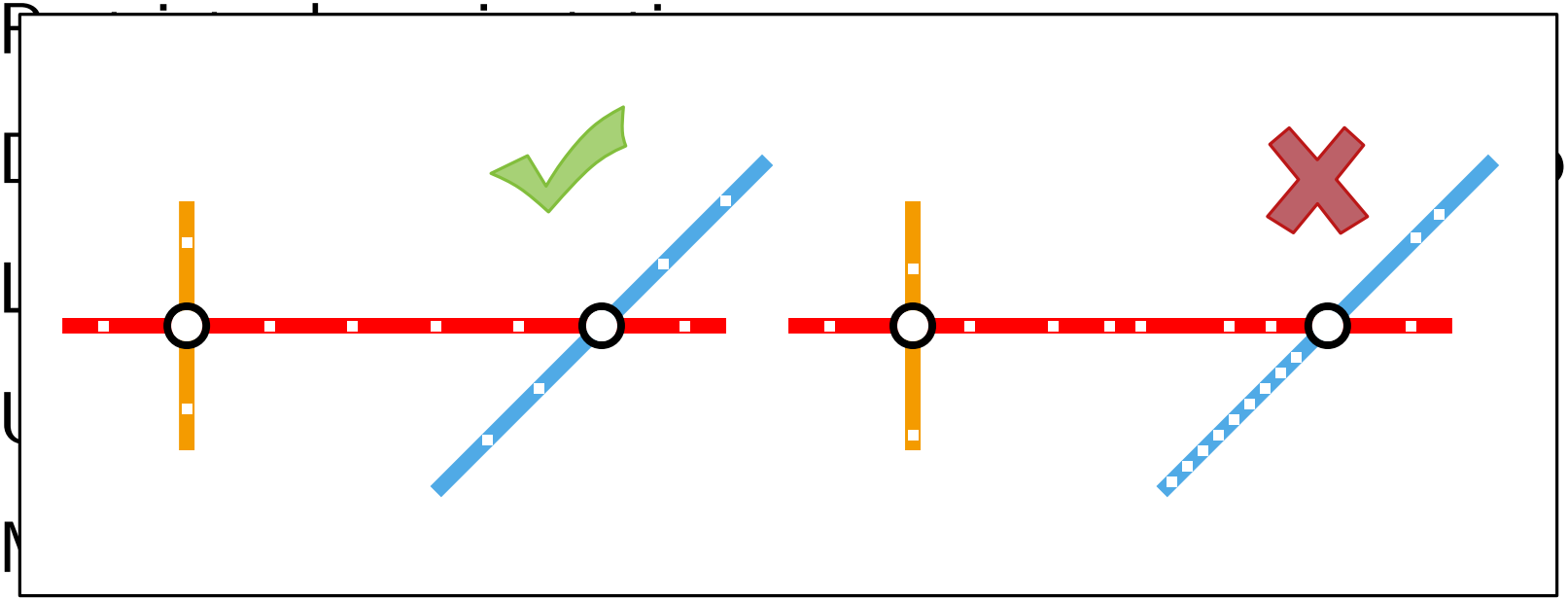
(R2) P

(R3) D ble.

(R4) L

(R5) U

(R6) M



(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.

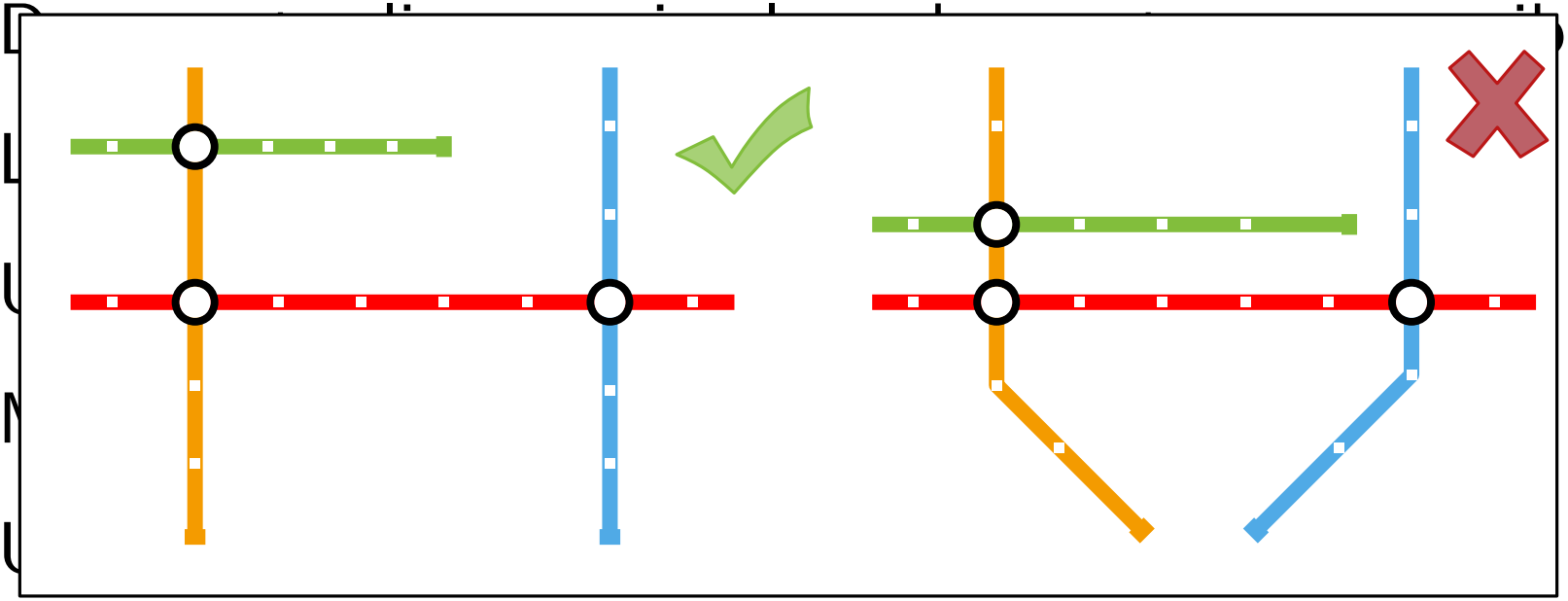
(R3) D…ble.

(R4) L…

(R5) U…

(R6) M…

(R7) U…



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

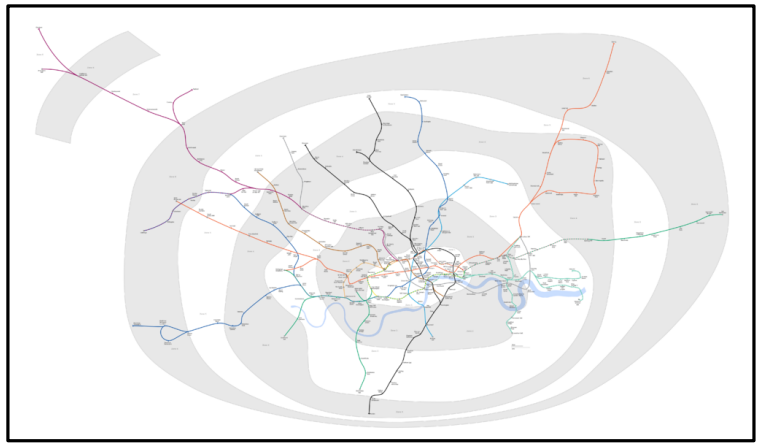
(R4) L

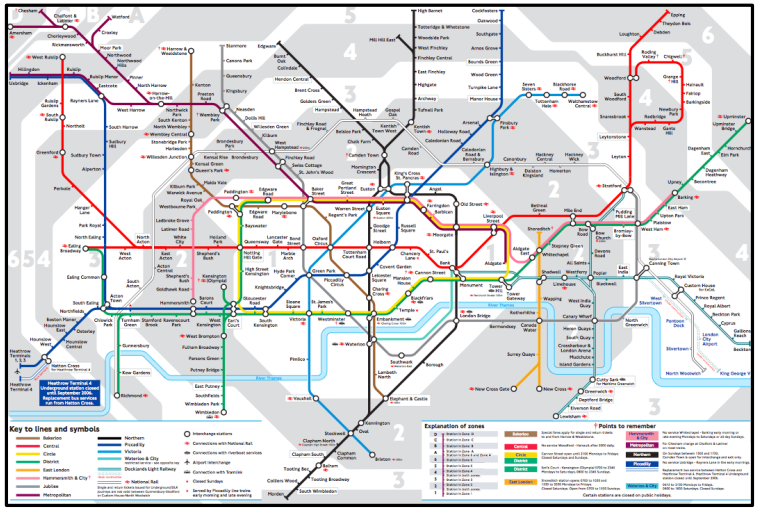
(R5) U

(R6) M

(R7) U

(R8) K





(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**

Complexity

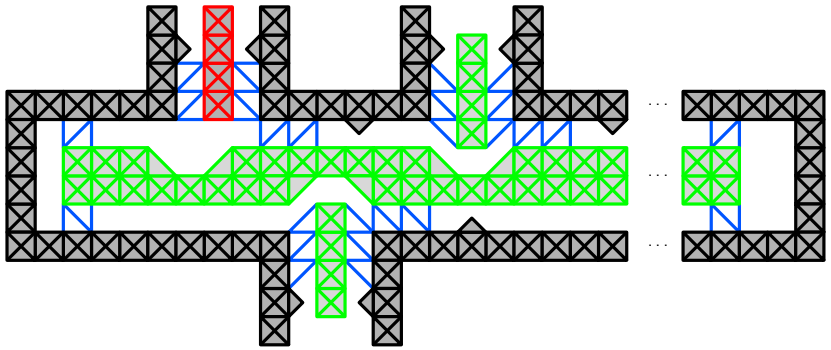
Theorem: For an embedded graph G (vertex degrees ≤ 8) **bend minimization** (R3) is NP-hard if **preserving topology** (R1) and **octilinearity** (R2) are required.

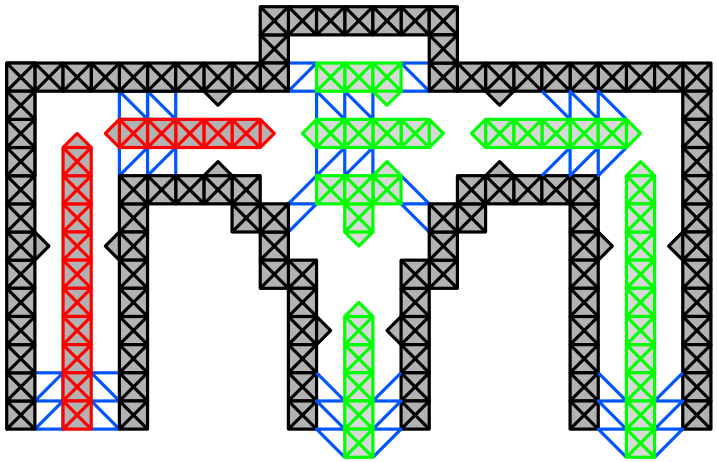
Complexity

Theorem: 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: (Nöllenburg 2005)

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



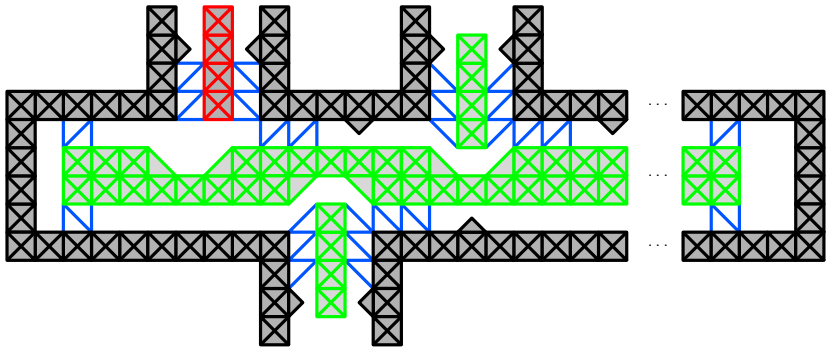


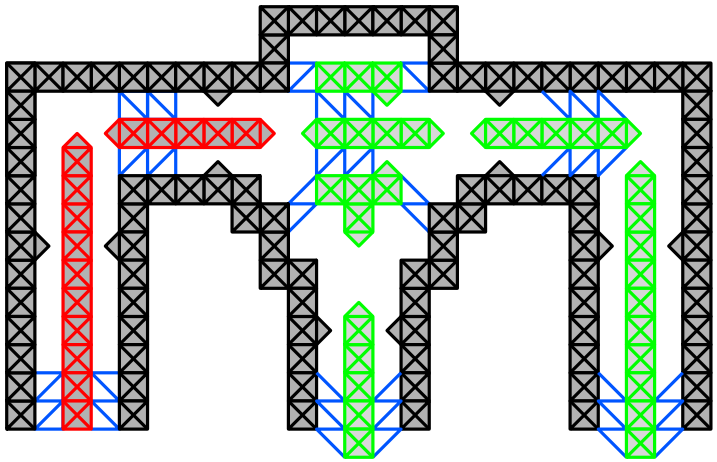
Complexity

Theorem: 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: (Nöllenburg 2005)

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





Remark:

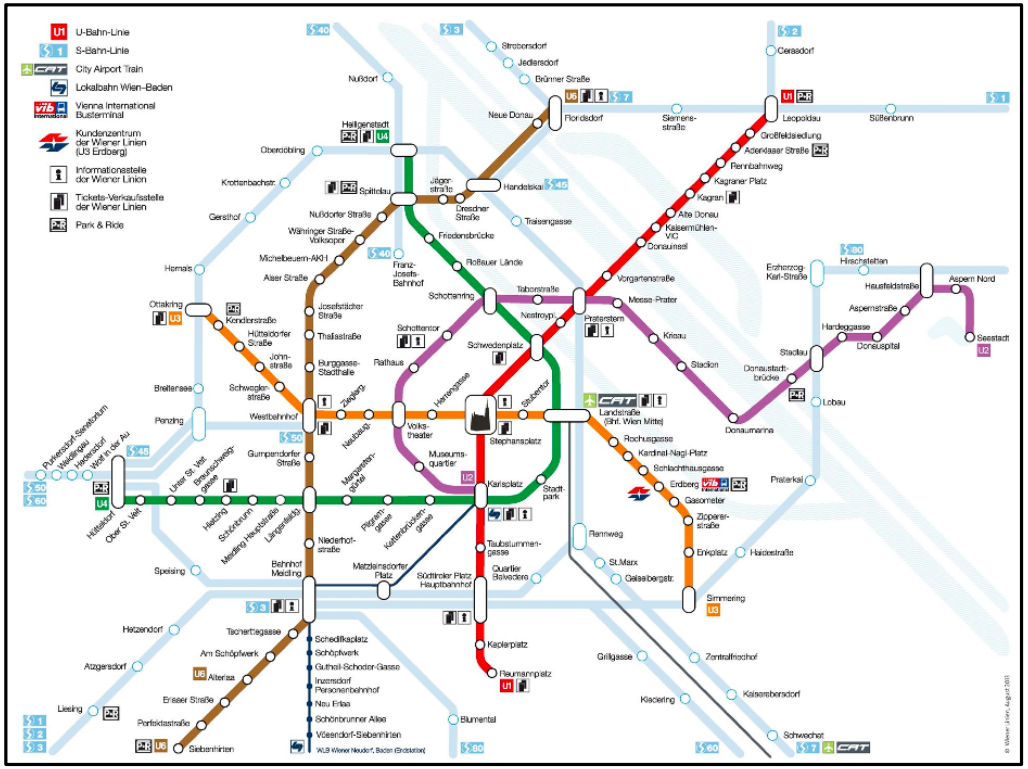
- no efficient exact algorithms to expect
- same problem without diagonals (rectilinear) is efficiently solvable (Tamassia 1987)

Mixed-Integer Programming (Nöllenburg, Wolff, 2011)

- find exact optimum solution using combinatorial optimization
- separate 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

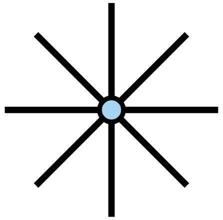
(R1) preserve embedding/**topology** and **planarity**

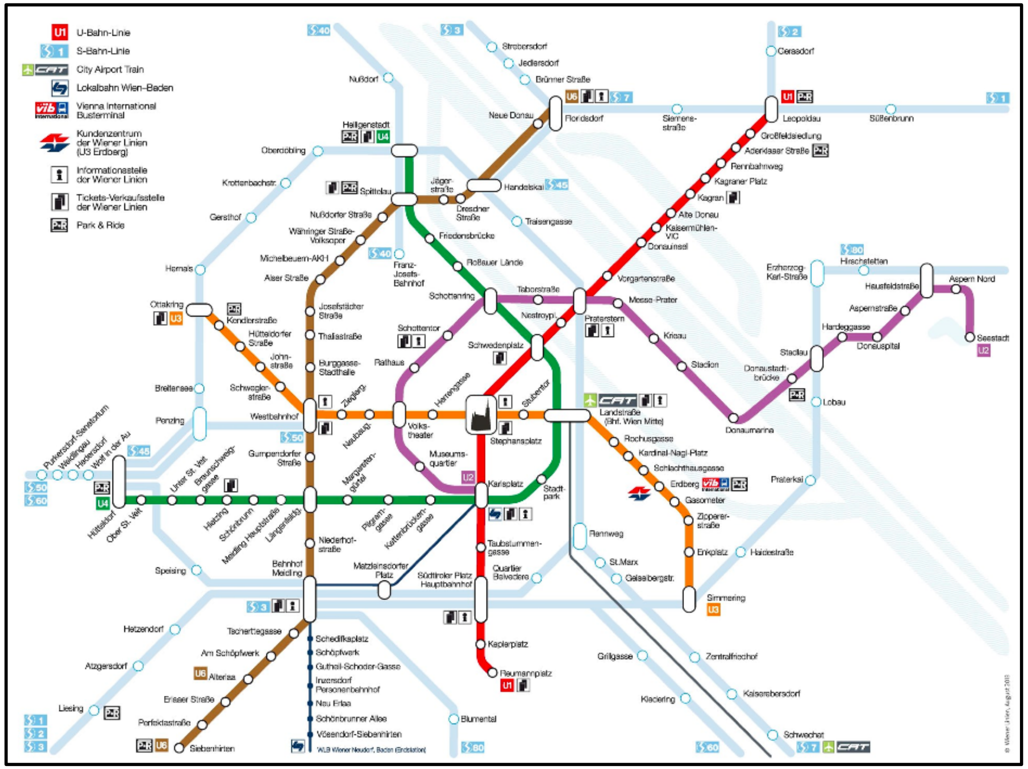


Hard Constraints

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

(R2) draw all edges **octilinearly**





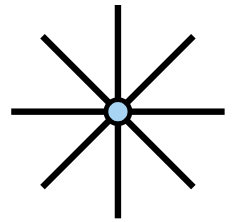
Hard Constraints

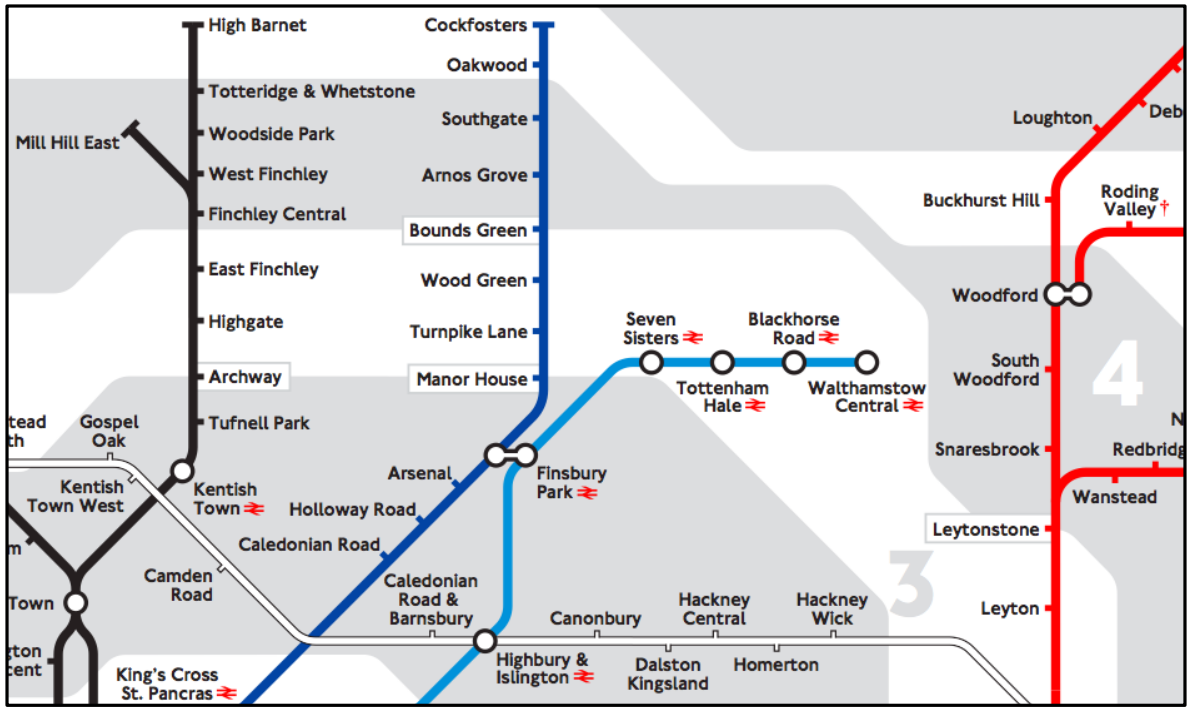
(R1) preserve embedding/**topology** and **planarity**

(R2) draw all edges **octilinearly**

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

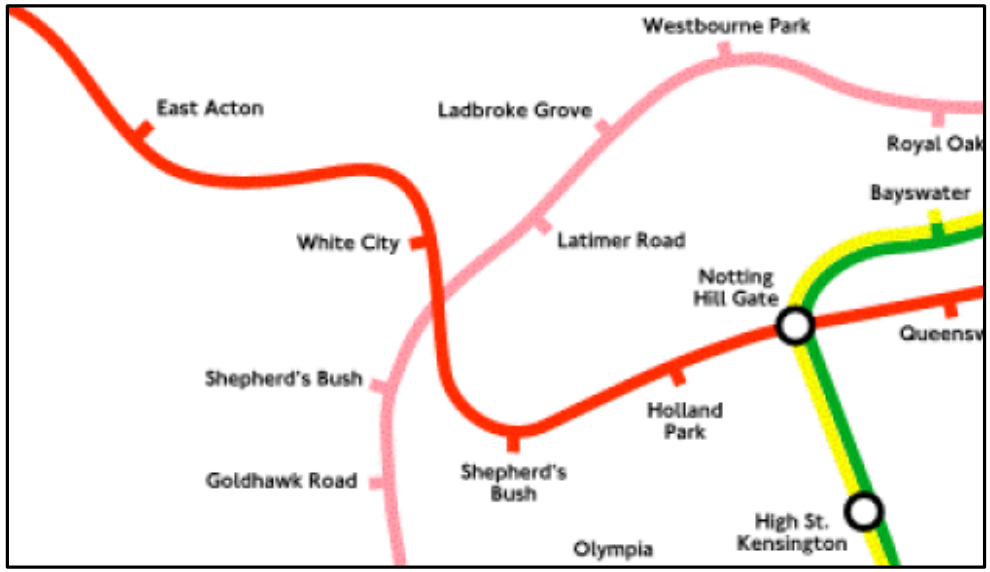
(R8) enforce **minimum feature separation** $d_{\min}$

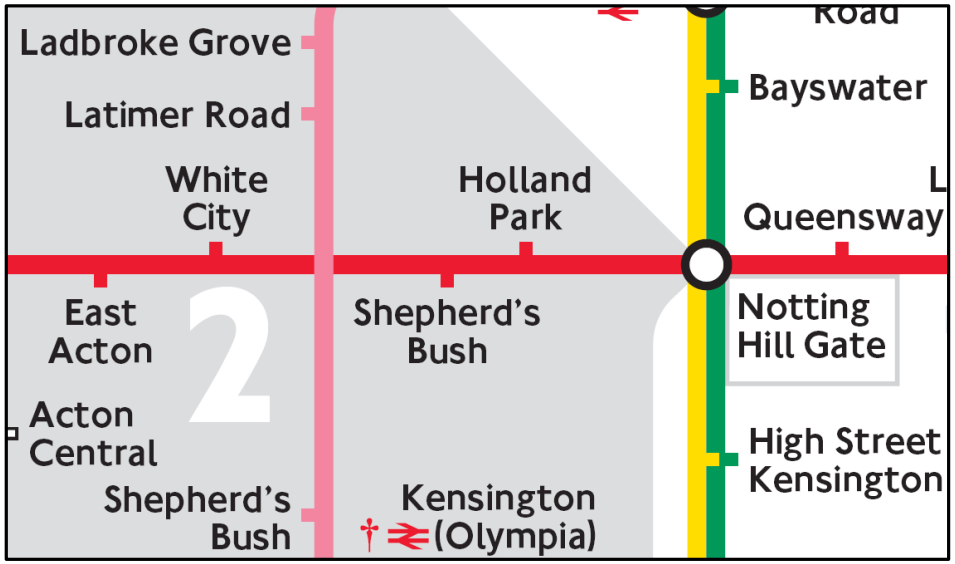




Soft Constraints

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

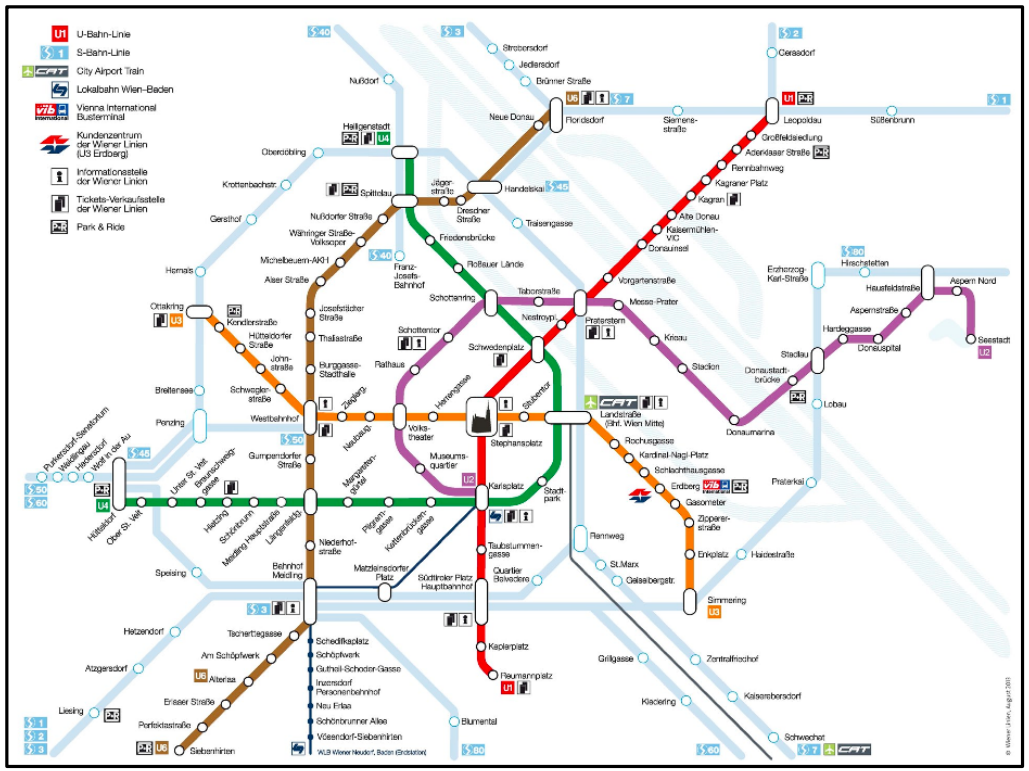




Soft Constraints

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

(R6) minimize geometric **distortion**

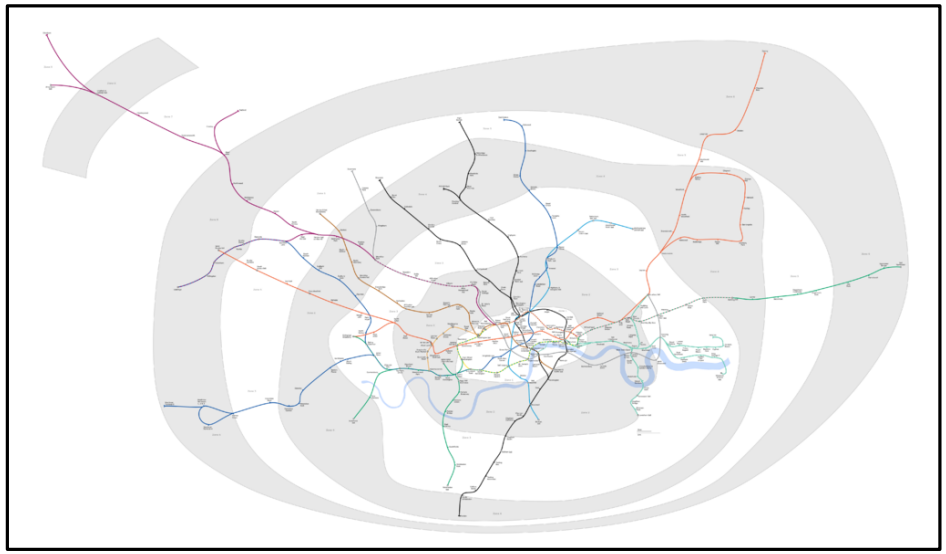


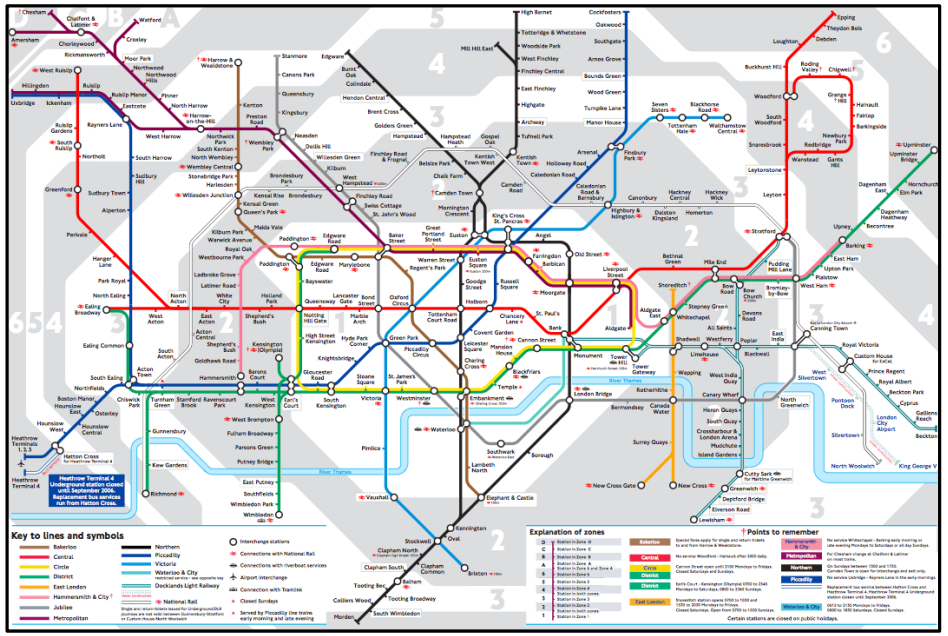
Soft Constraints

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

(R6) minimize geometric **distortion**

(R7) minimize total **edge length**

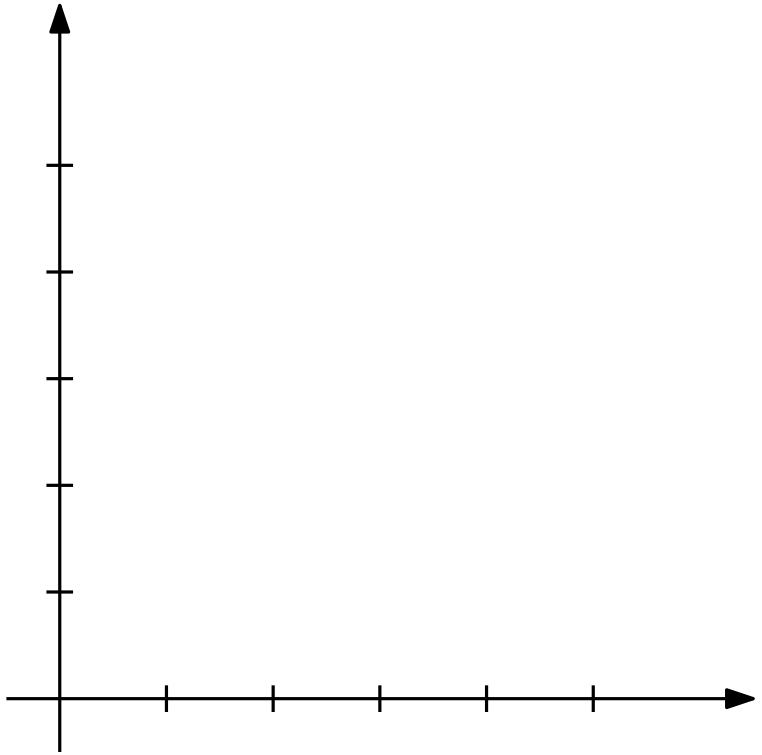




Linear Programming

Linear Programming (LP) is an efficient optimization method for

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



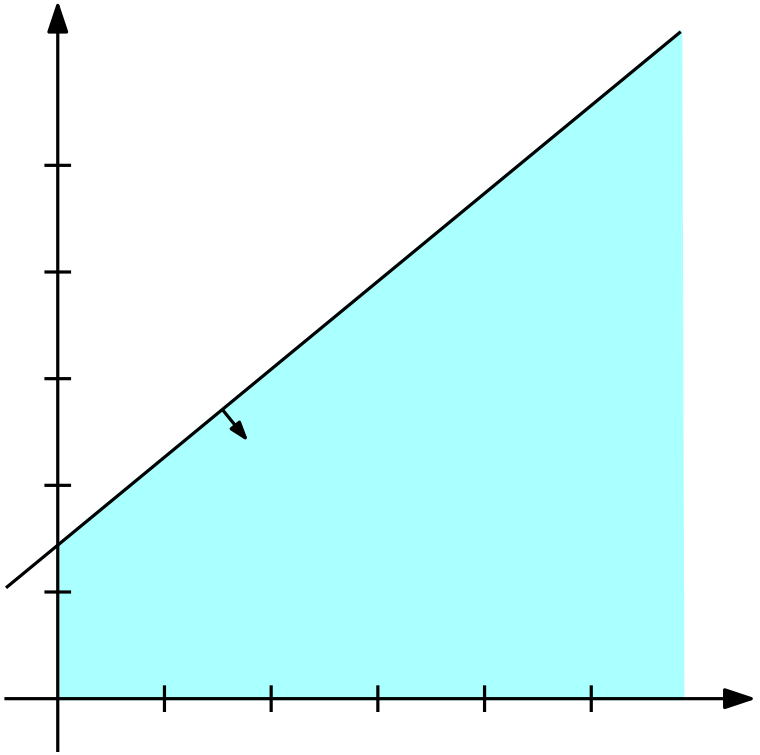
Linear Programming

Linear Programming (LP) is an efficient optimization method for

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

Example:

$y \leq 0.9x + 1.5$



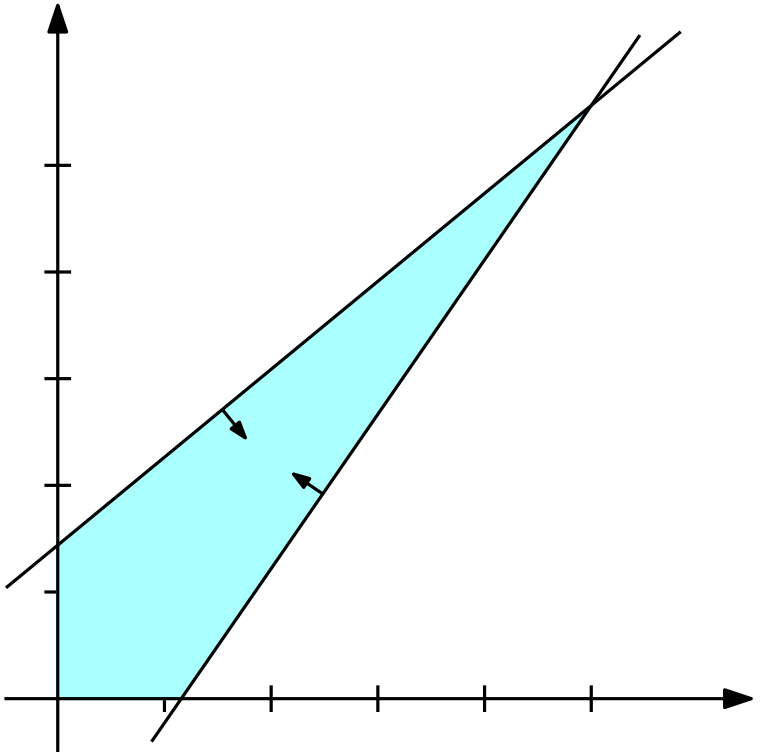
Linear Programming

Linear Programming (LP) is an efficient optimization method for

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

Example:

$$y \leq 0.9x + 1.5$$
$$y \geq 1.4x - 1.3$$



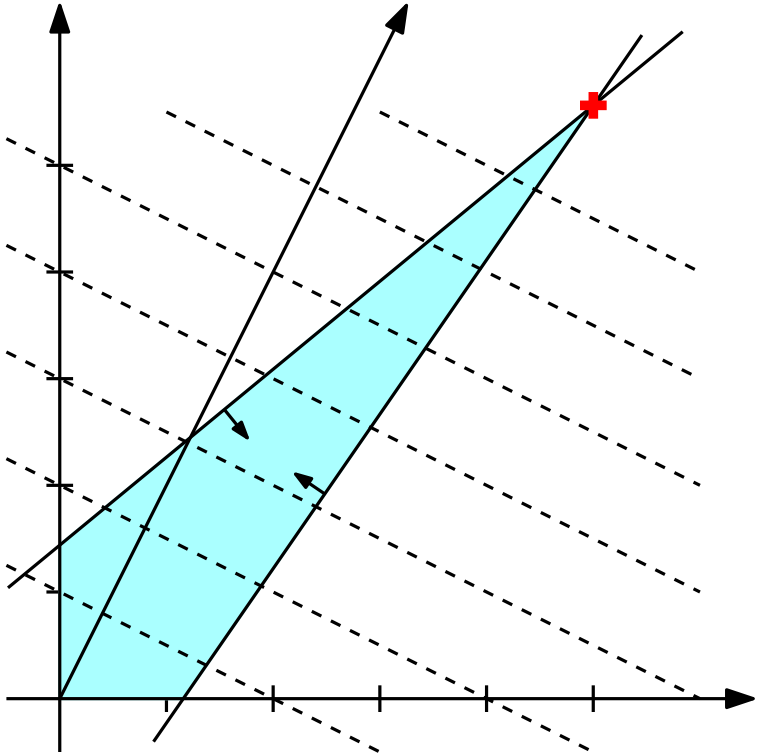
Linear Programming

Linear Programming (LP) is an efficient optimization method for

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

Example:

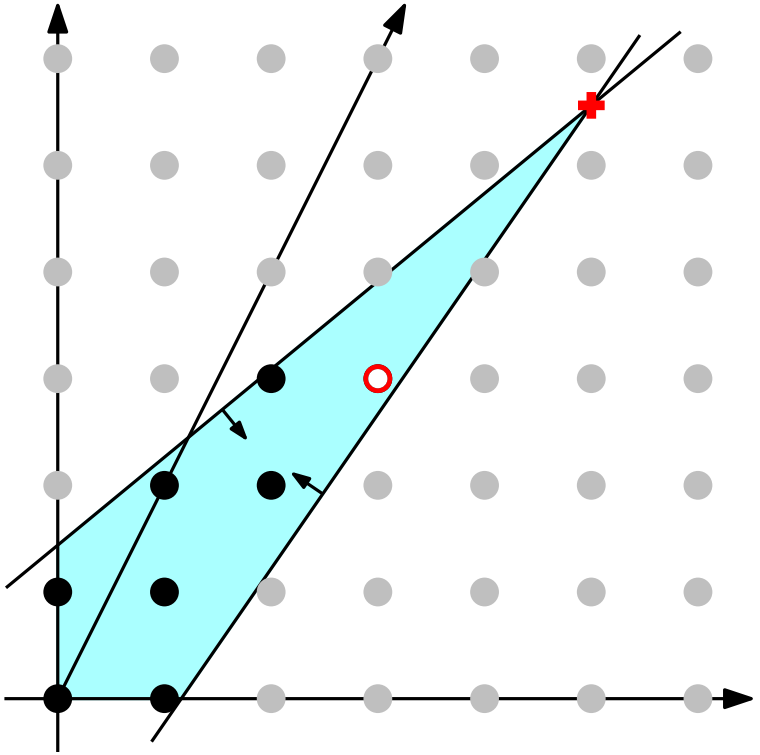
$y \leq 0.9x + 1.5$
$y \geq 1.4x - 1.3$
maximize $x + 2y$



Linear Programming

Linear Programming (LP) is an efficient optimization method for

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



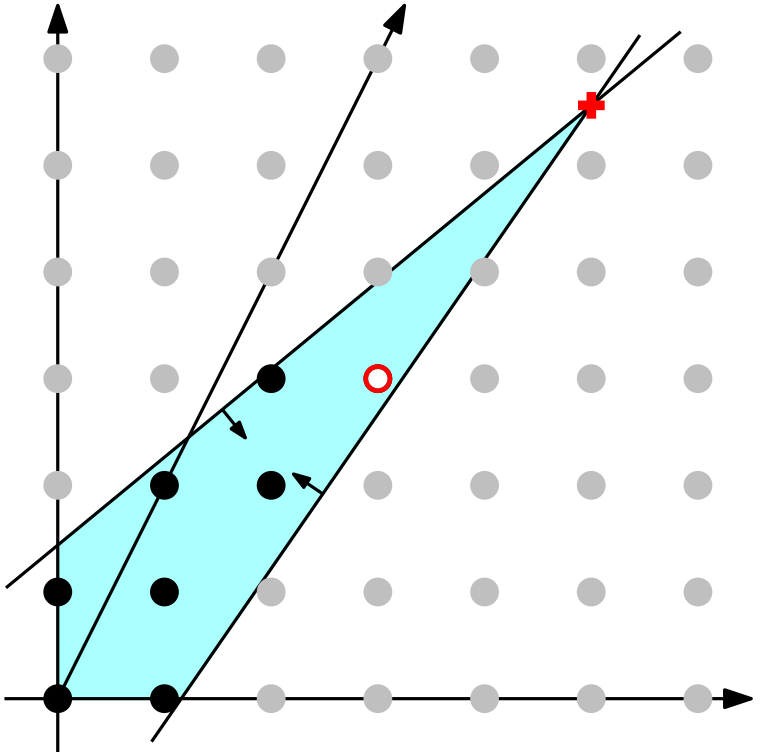
Mixed Integer Programming (MIP)

- in addition binary and integer variables
- 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



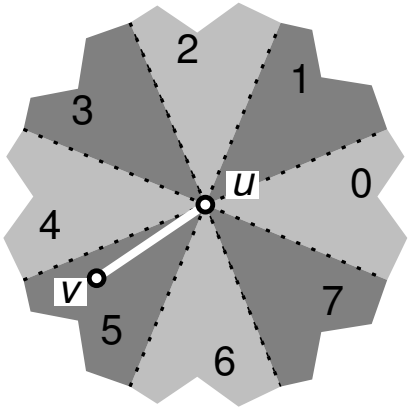
Mixed Integer Programming (MIP)

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

Theorem: Metro map layout can be modeled as MIP s.t.

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

Definitions: Sectors and Coordinates

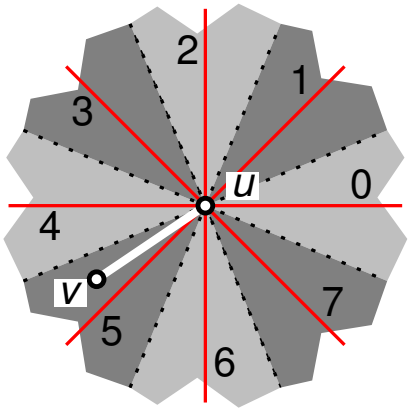


Sectors

– for each vtx. u partition plane into sectors 0–7

- here: $\mathrm{sec}(u, v) = 5$ (input)

Definitions: Sectors and Coordinates



Sectors

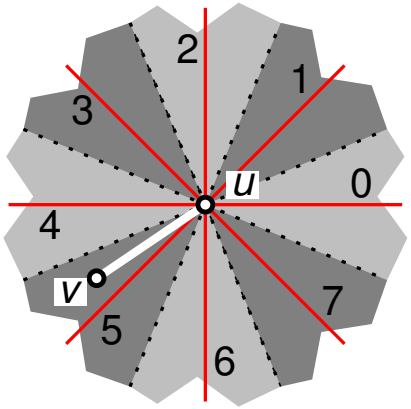
– for each vtx. u partition plane into sectors 0–7

- here: $\mathrm{sec}(u, v) = 5$ (input)

– number octilinear edge directions accordingly

- e.g. $\mathrm{dir}(u, v) = 4$ (output)

Definitions: Sectors and Coordinates



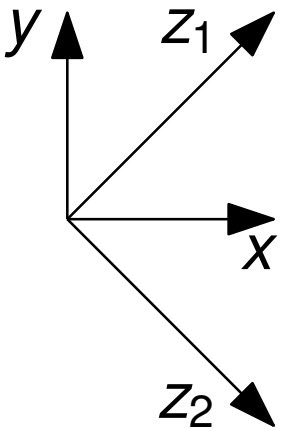
Sectors

– for each vtx. u partition plane into sectors 0–7

- here: $\mathrm{sec}(u, v) = 5$ (input)

– number octilinear edge directions accordingly

- e.g. $\mathrm{dir}(u, v) = 4$ (output)



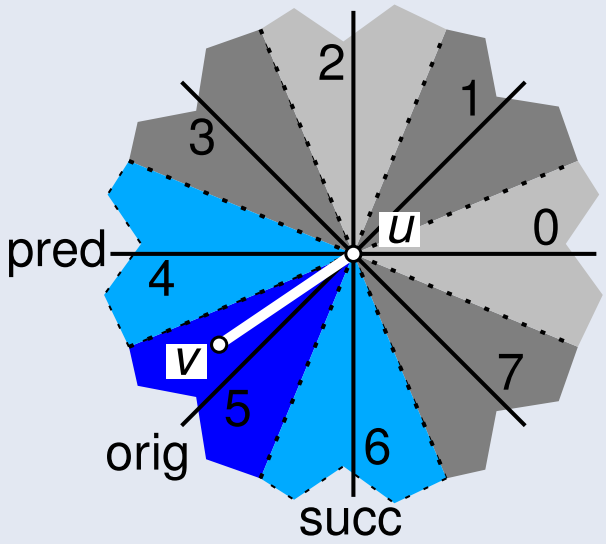
Coordinates

assign z_1- and z_2-coordinates to each vertex v:

- $z_1(v) = x(v) + y(v)$
- $z_2(v) = x(v) - y(v)$

Octilinearity and Relative Position

Goal

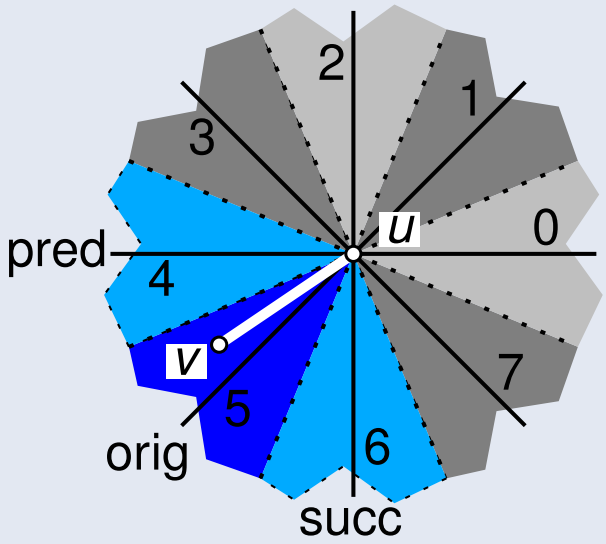


Draw edge uv

- octilinearly
- with minimum length ℓ_{uv}
- restricted to 3 directions

Octilinearity and Relative Position

Goal



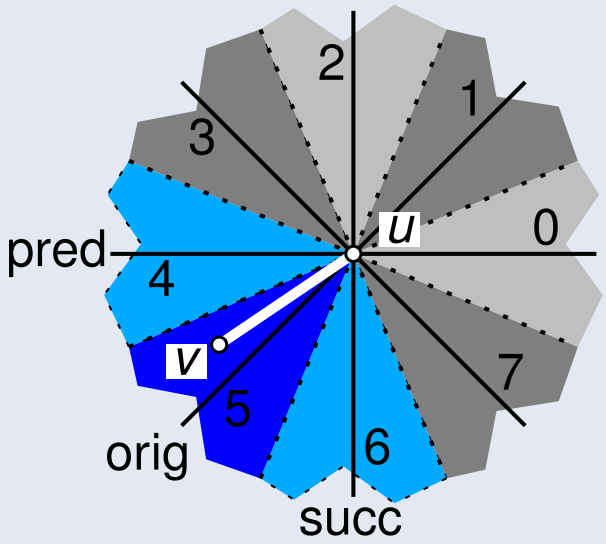
Draw edge uv

- octilinearly
- with minimum length ℓ_{uv}
- restricted to 3 directions

How to model this using linear constraints?

Octilinearity and Relative Position

Goal



Draw edge uv

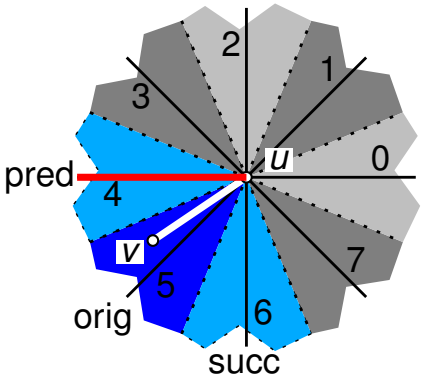
- octilinearly
- with minimum length ℓ_{uv}
- restricted to 3 directions

How to model this using linear constraints?

Binary Variables

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

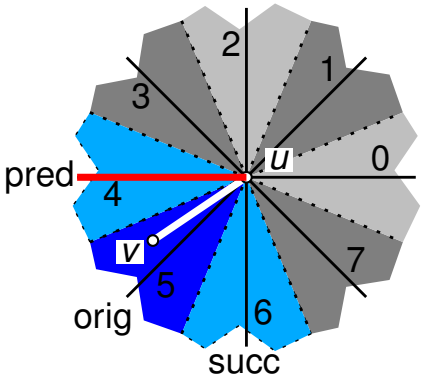
Octilinearity and Relative Position



Predecessor Sector

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

Octilinearity and Relative Position

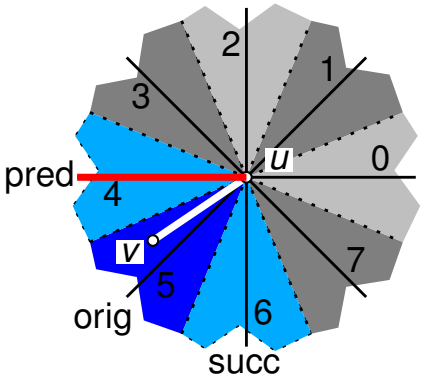


Predecessor Sector

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

How does this work?

Octilinearity and Relative Position



Predecessor Sector

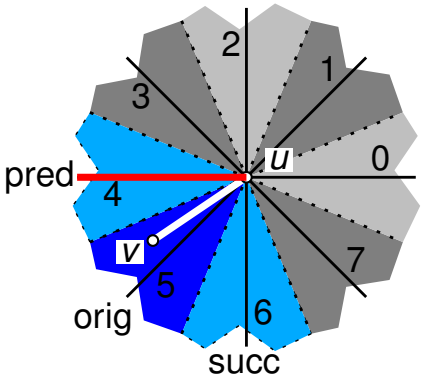
$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell_{uv} \end{aligned}$$

How does this work?

Case 1: $\alpha_{\text{pred}}(u,v) = 0$

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

Octilinearity and Relative Position



Predecessor Sector

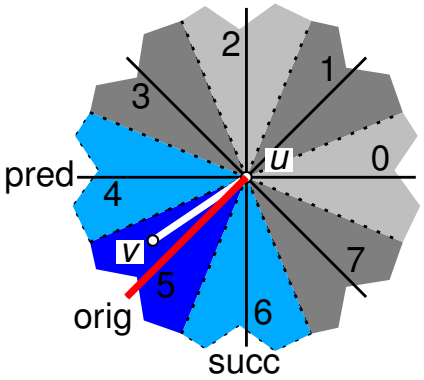
$$\begin{aligned} y(u) - y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ -y(u) + y(v) &\leq M(1 - \alpha_{\text{pred}}(u,v)) \\ x(u) - x(v) &\geq -M(1 - \alpha_{\text{pred}}(u,v)) + \ell_{uv} \end{aligned}$$

How does this work?

Case 2: $\alpha_{\text{pred}}(u,v) = 1$

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

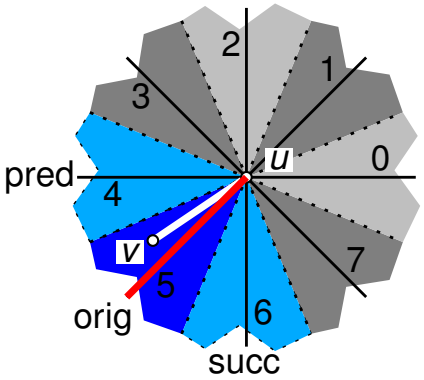
Octilinearity and Relative Position



Original Sector

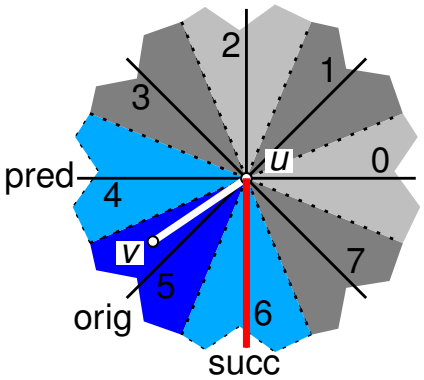
$$
\begin{aligned}
z_2(u) - z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\
-z_2(u) + z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\
z_1(u) - z_1(v) &\geq -M(1 - \alpha_{\text{orig}}(u,v)) + 2\ell_{uv}
\end{aligned}
$$

Octilinearity and Relative Position



Original Sector

$$
\begin{aligned}
z_2(u) - z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\
-z_2(u) + z_2(v) &\leq M(1 - \alpha_{\text{orig}}(u,v)) \\
z_1(u) - z_1(v) &\geq -M(1 - \alpha_{\text{orig}}(u,v)) + 2\ell_{uv}
\end{aligned}
$$



Successor Sector

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

Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

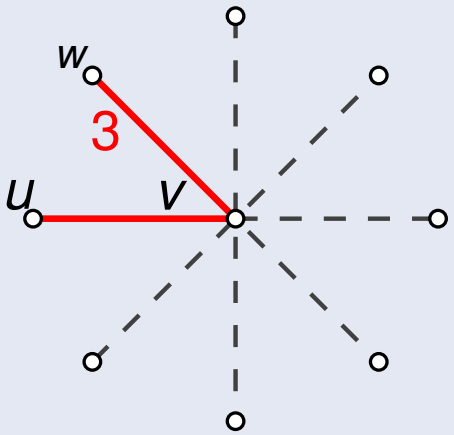
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

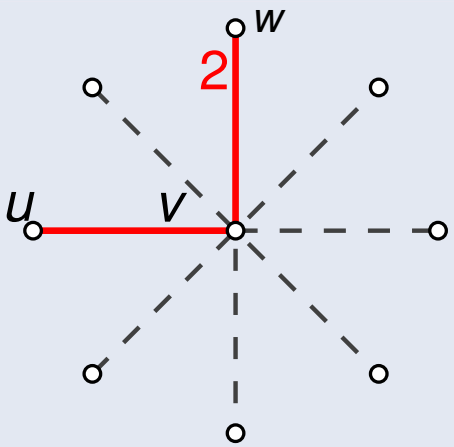
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

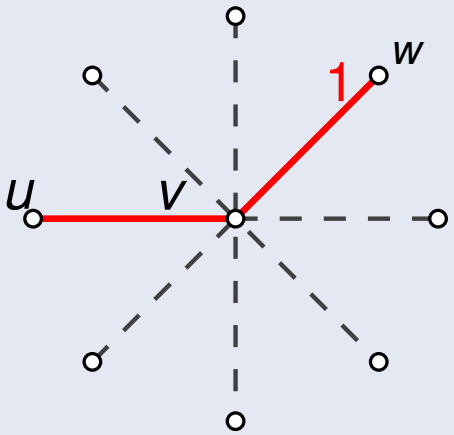
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

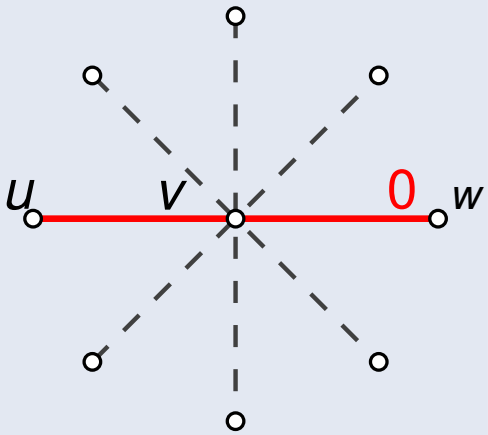
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

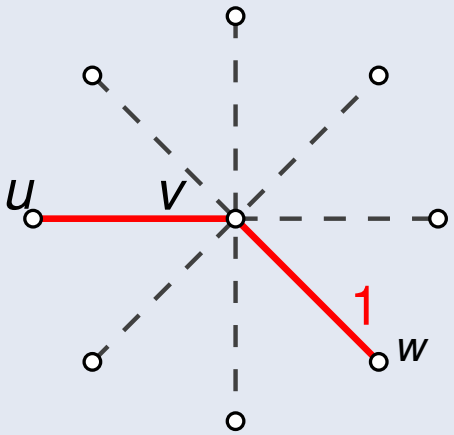
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

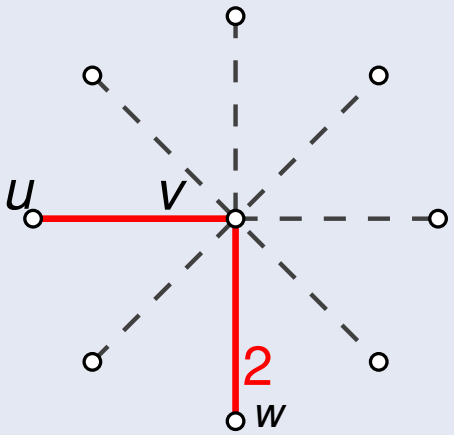
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

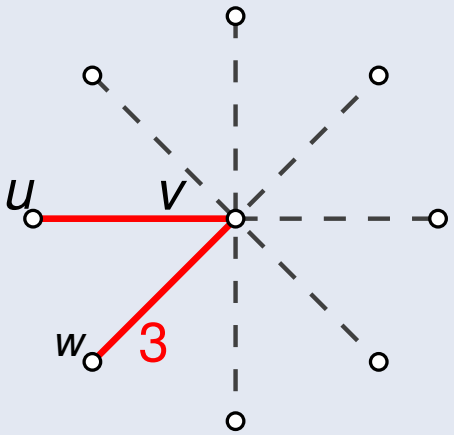
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

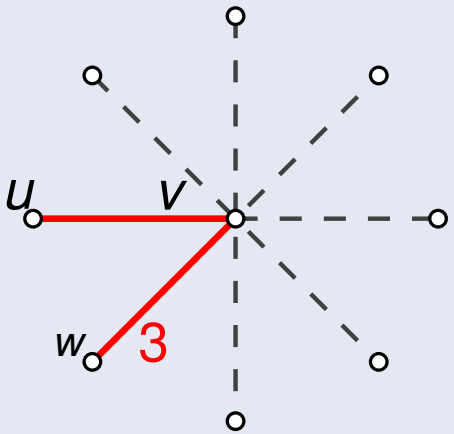
Objective Function

Objective Function

- corresponds to soft constraints (S1)–(S3)
- weighted sum of individual cost functions

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

Line Bends (S1)



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

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

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

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 $\lambda_{\text{bend}}\,\text{cost}_{\text{bend}} + \lambda_{\text{len}}\,\text{cost}_{\text{len}} + \lambda_{\text{dist}}\,\text{cost}_{\text{dist}}$

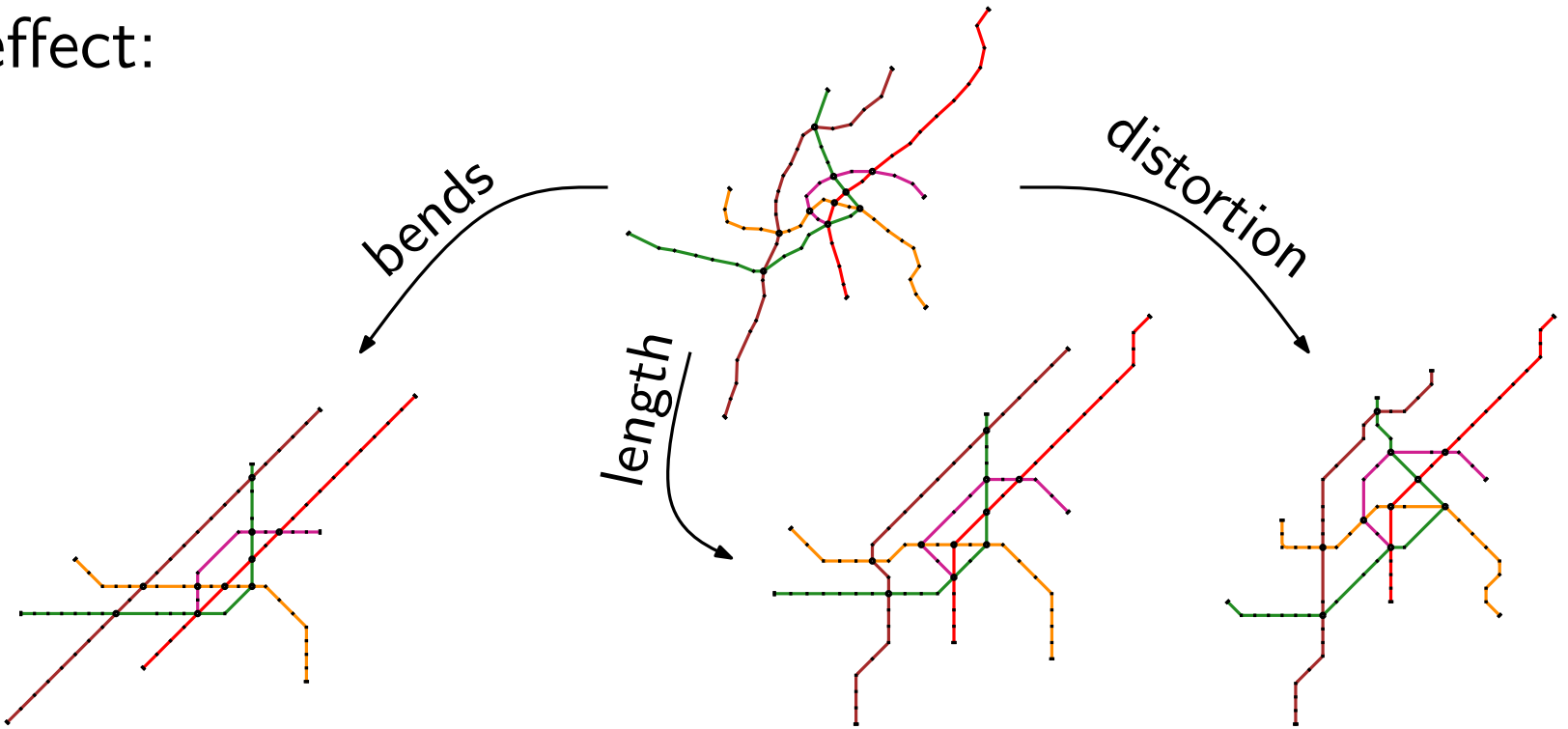
Overview MIP model

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:



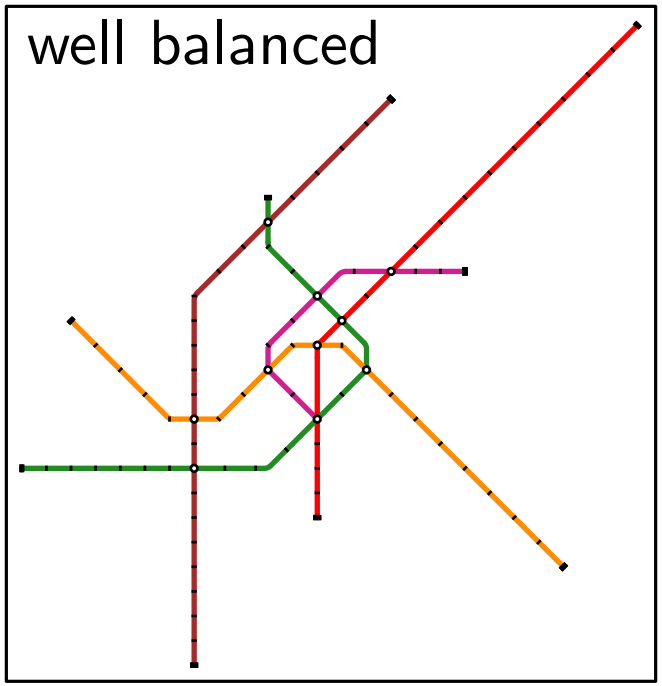
Overview MIP model

Constraints:

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

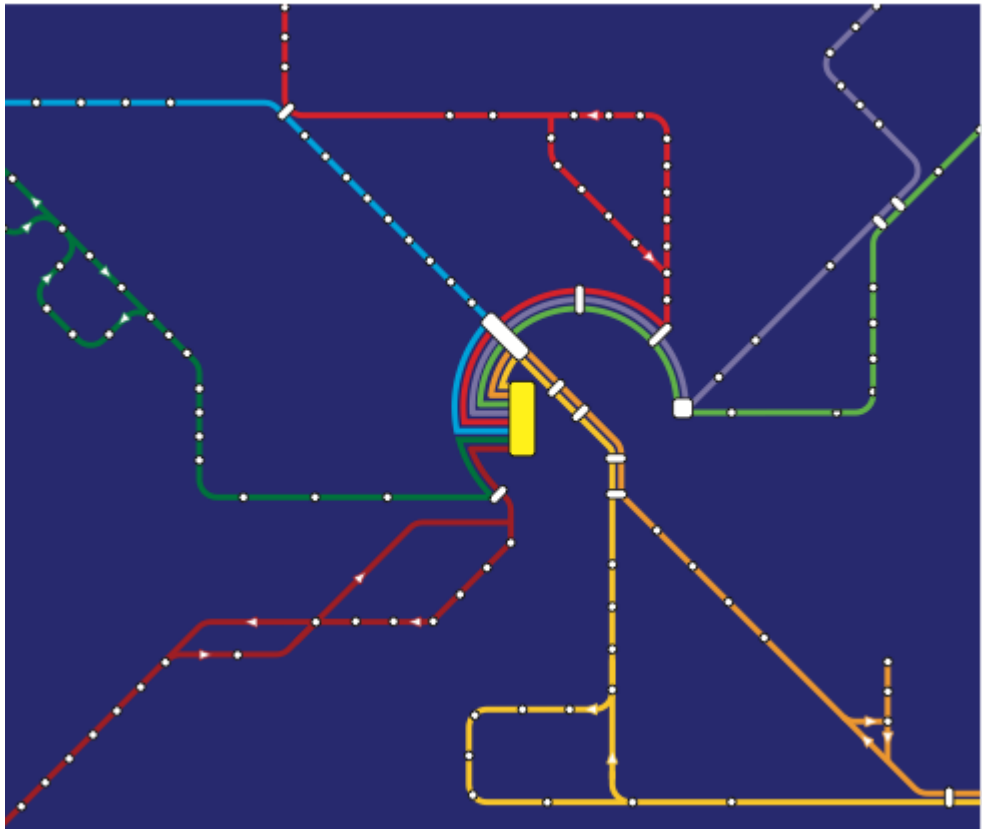
Objective function:

- 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:



Station Labeling

- unlabeled map mostly useless



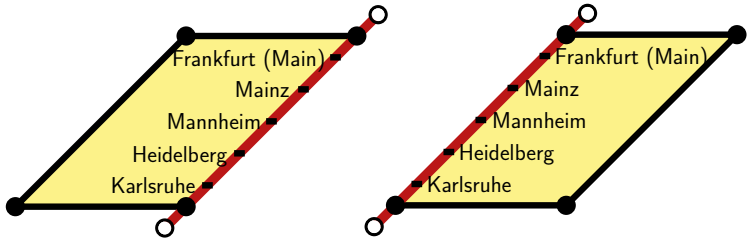
Station Labeling

- unlabeled map mostly useless
- labels need space
- labels may not overlap each other
- graph labeling problem is NP-hard (Tollis, Kakoulis 2001)

Station Labeling

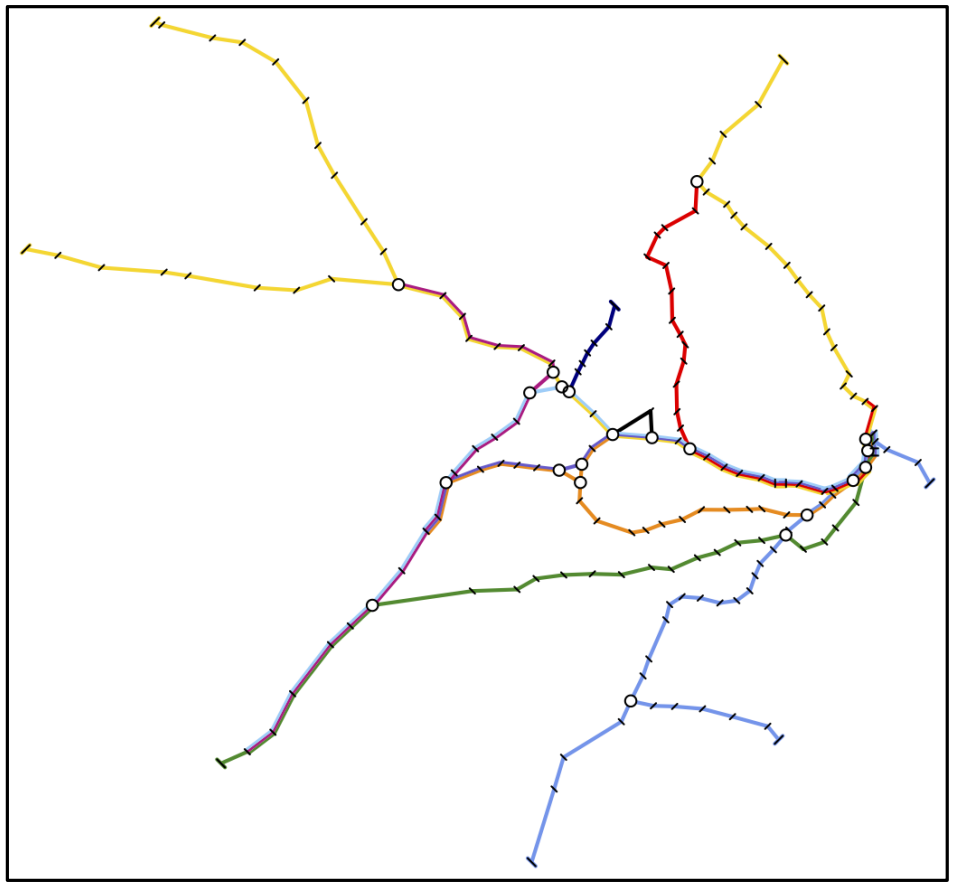
- unlabeled map mostly useless
- labels need space
- labels may not overlap each other
- graph labeling problem is NP-hard (Tollis, Kakoulis 2001)

→ combine layout & labeling for optimal results!



- parallelogram as special metro line
- switching sides allowed

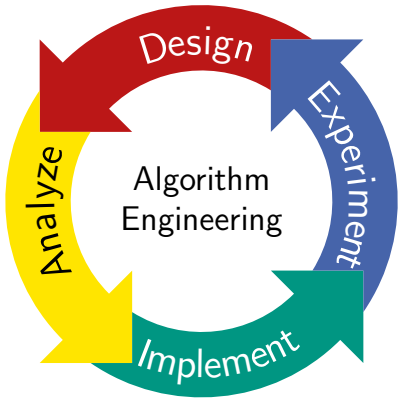
Example: Sydney



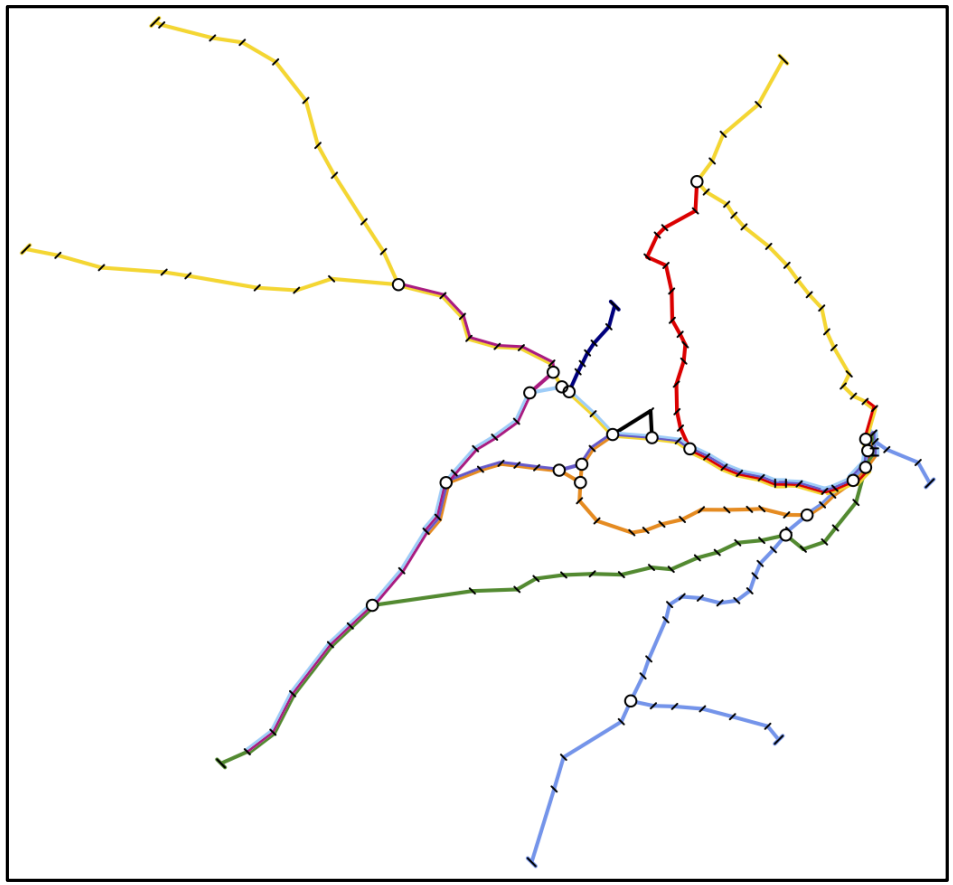
input

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

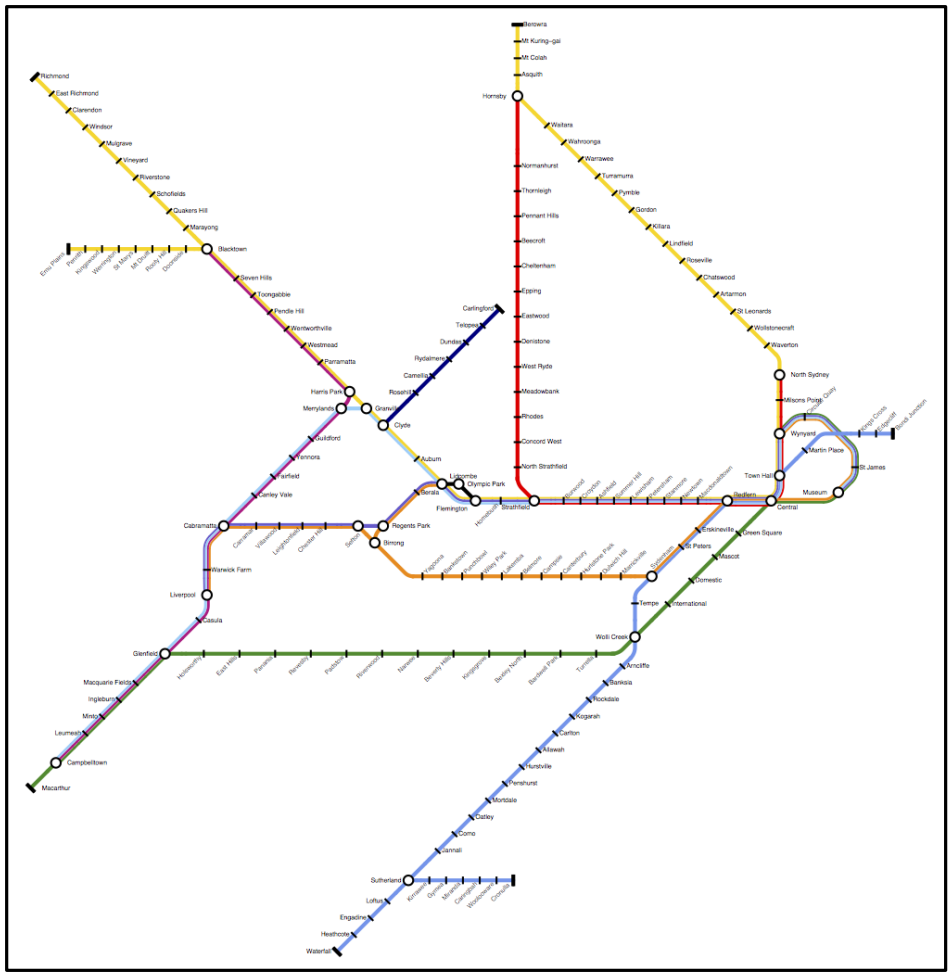
MIP	constraints.	variables
full	1,191,406	290,137
callback	21,988	92,681
skipped	6,838	2,969



Example: Sydney

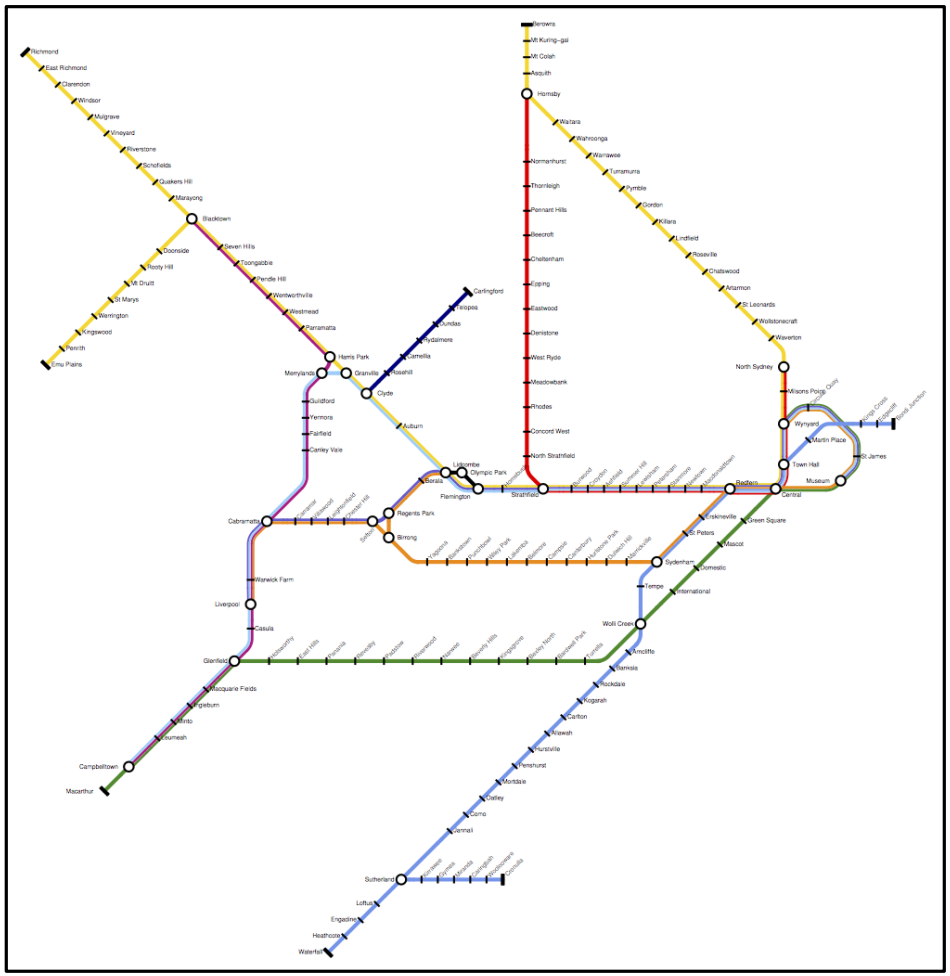


input

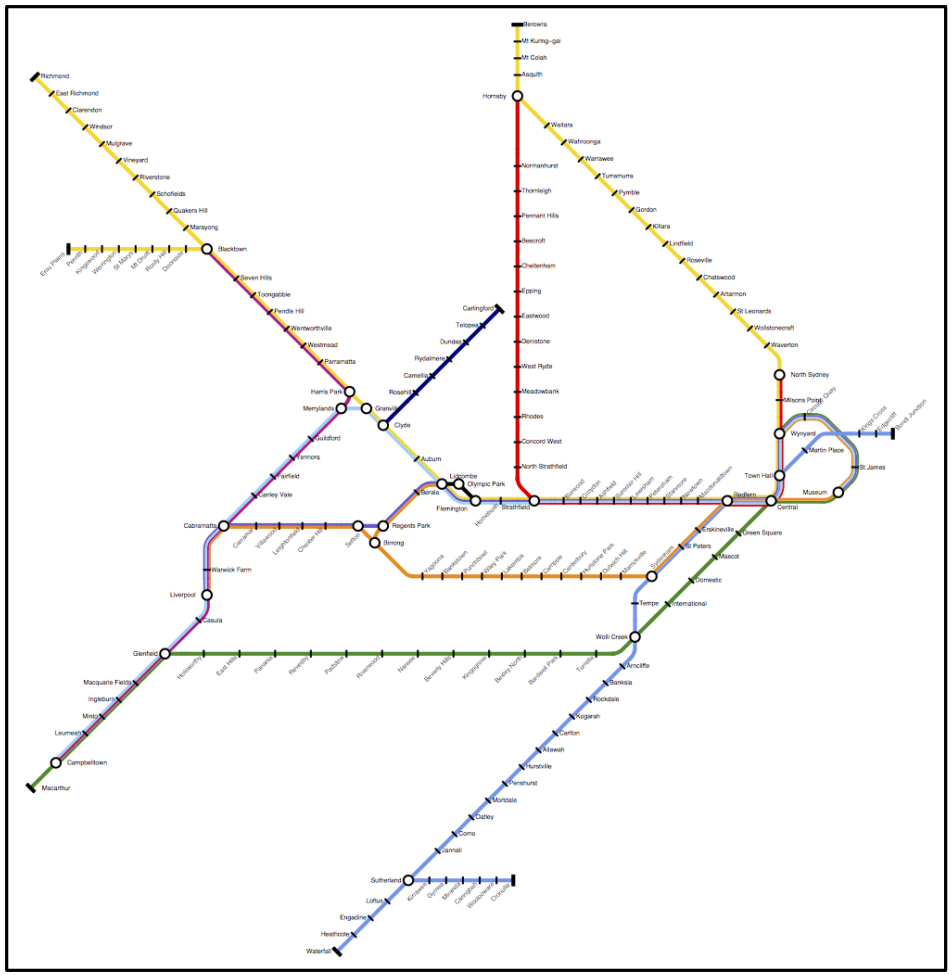


output (10:30 hrs)

Example: Sydney

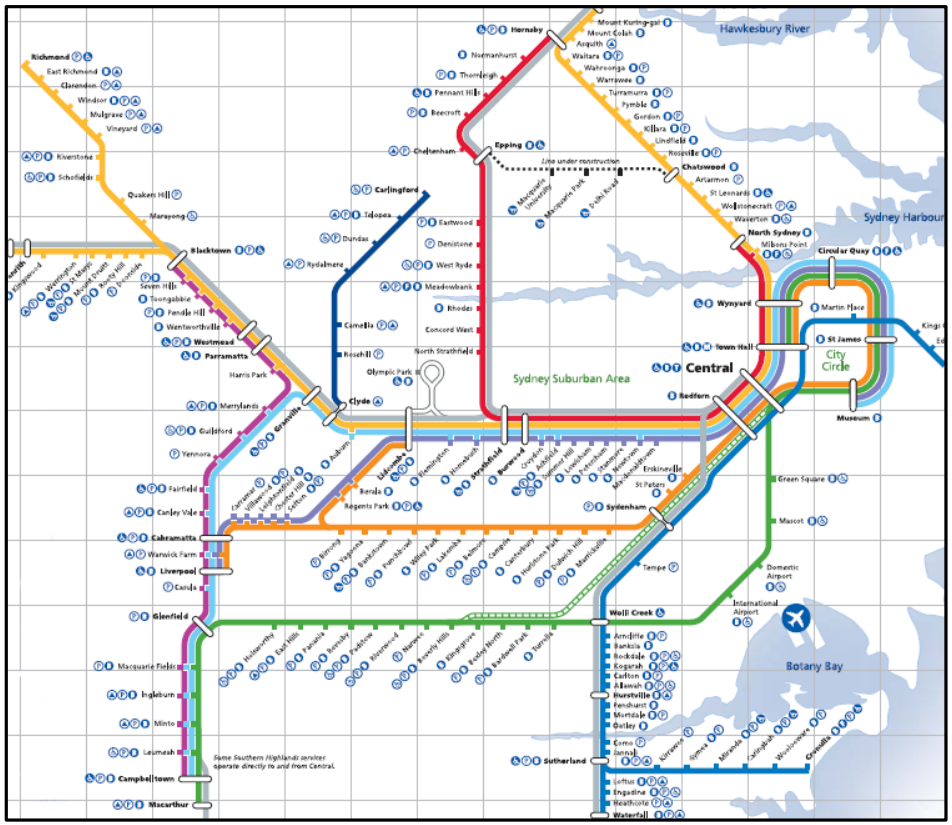


output (1:40 hrs)

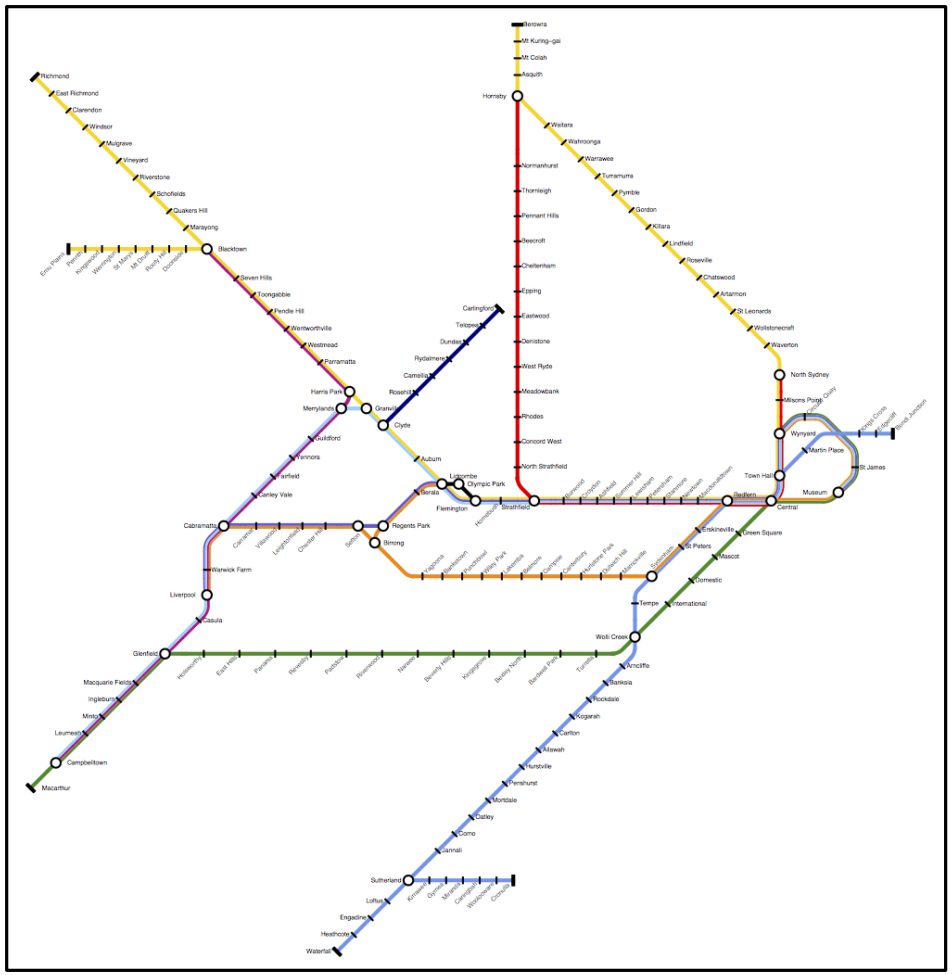


output (10:30 hrs)

Example: Sydney



official map



output (10:30 hrs)

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:

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