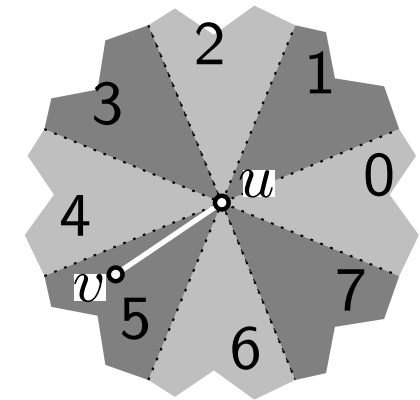
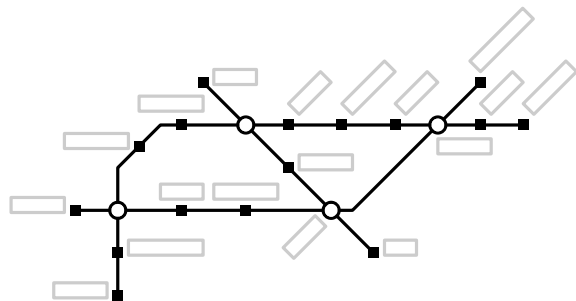
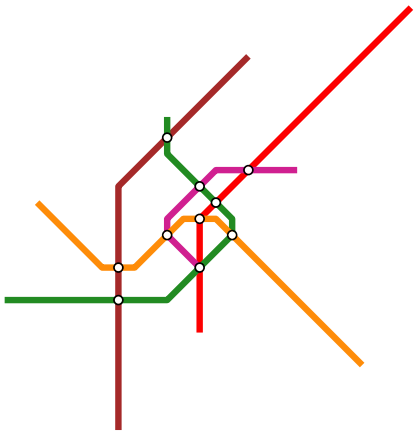


Visualization of Graphs

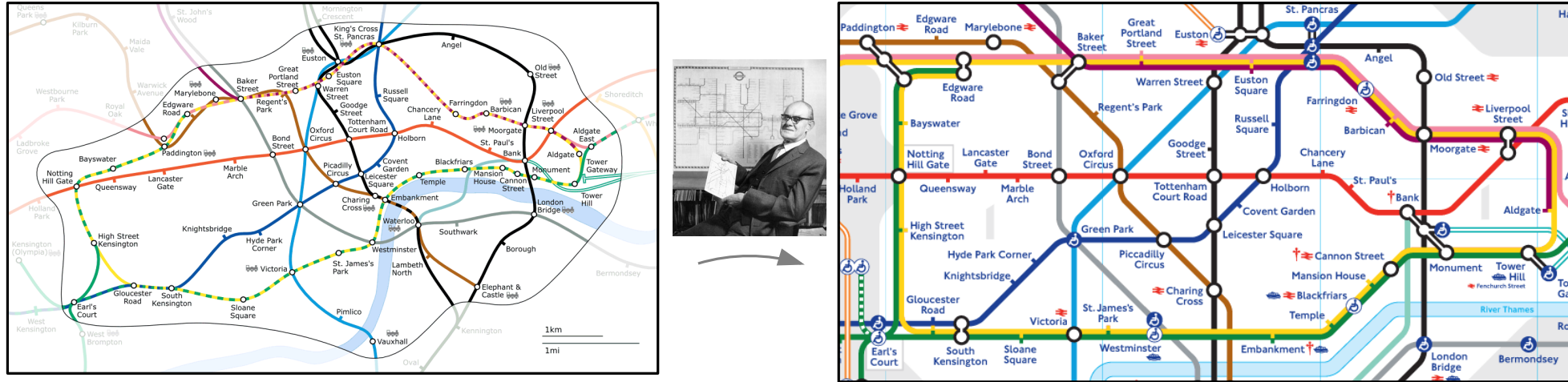
Lecture 12: Octilinear Graph Drawing Metro Map Layout

Part I: Schematic Metro Maps

Alexander Wolff

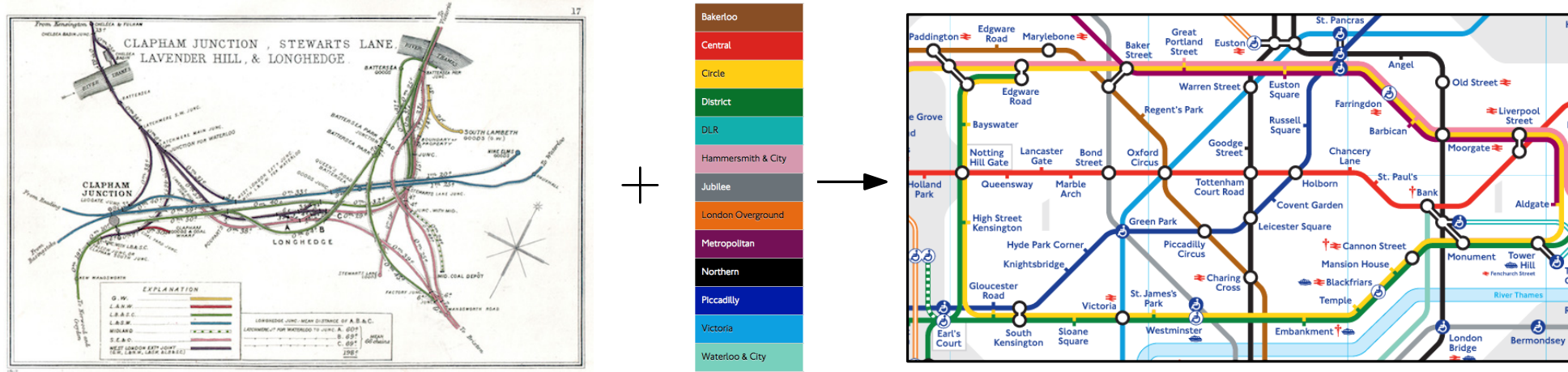


What is a Schematic Metro Map?



- map/diagram that shows stations connected by metro lines
- focus on topology rather than geography
- 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 $\mathcal{L}$ of metro lines 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

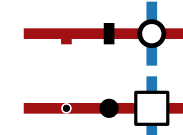
colors



fonts

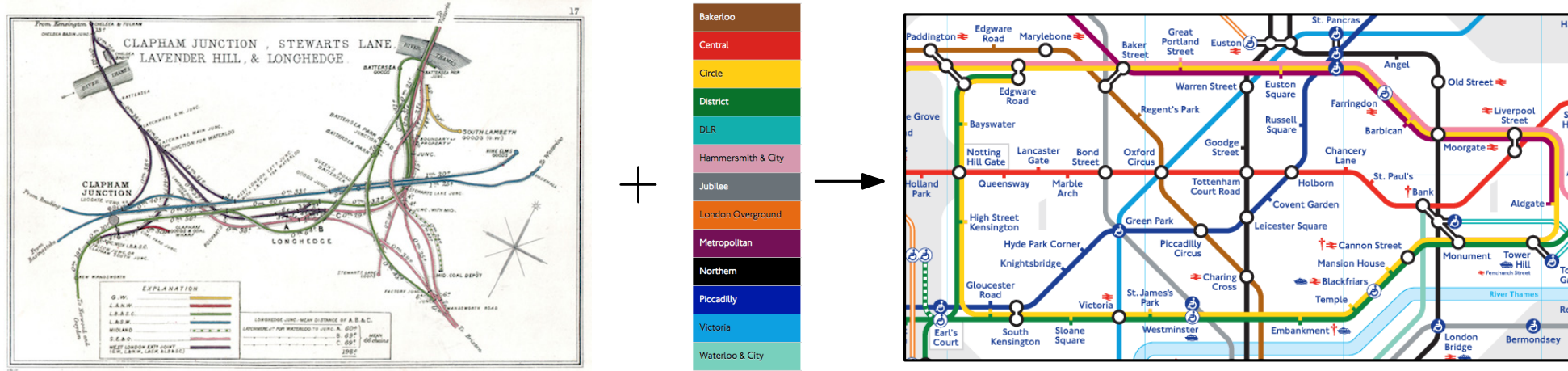
Paddington
PADDINGTON
Paddington

symbols



→ very salient,
but not a computational problem

Subtasks in Metro Map Layout



Input: ■ geographically embedded railway network G

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

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

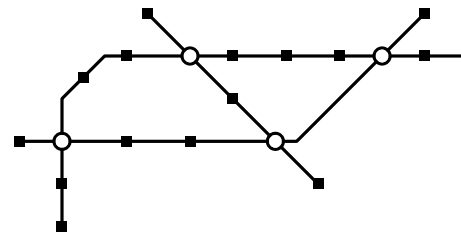
Divide the task into several subtasks:

■ rendering and design choices

■ network layout

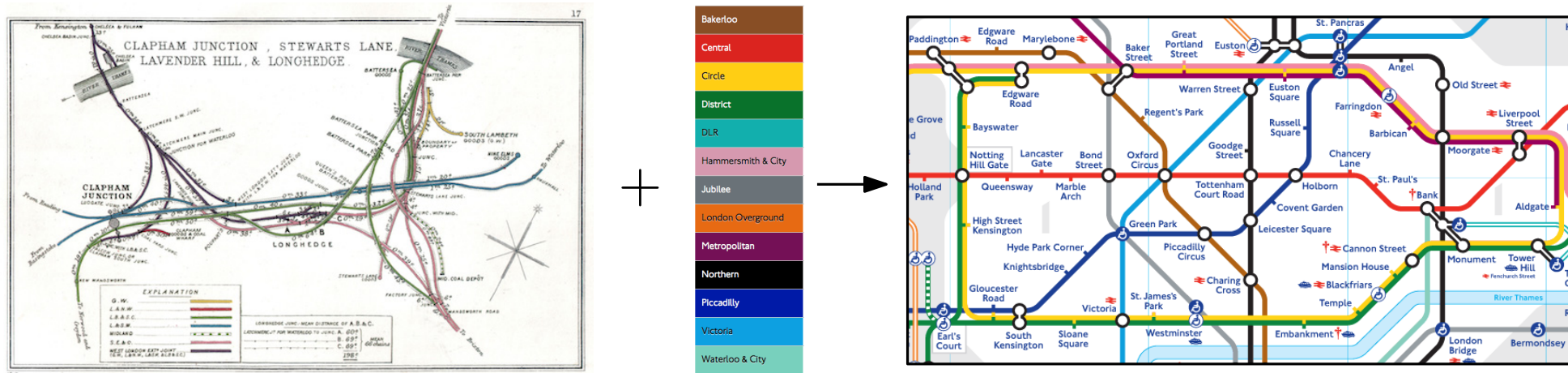
■ station labeling

■ metro line routing



determine geometry of network layout

Subtasks in Metro Map Layout



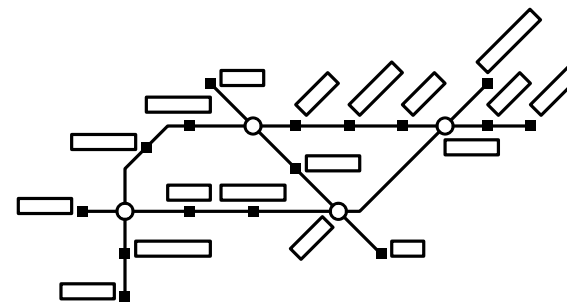
Input: ■ geographically embedded railway network G

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

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

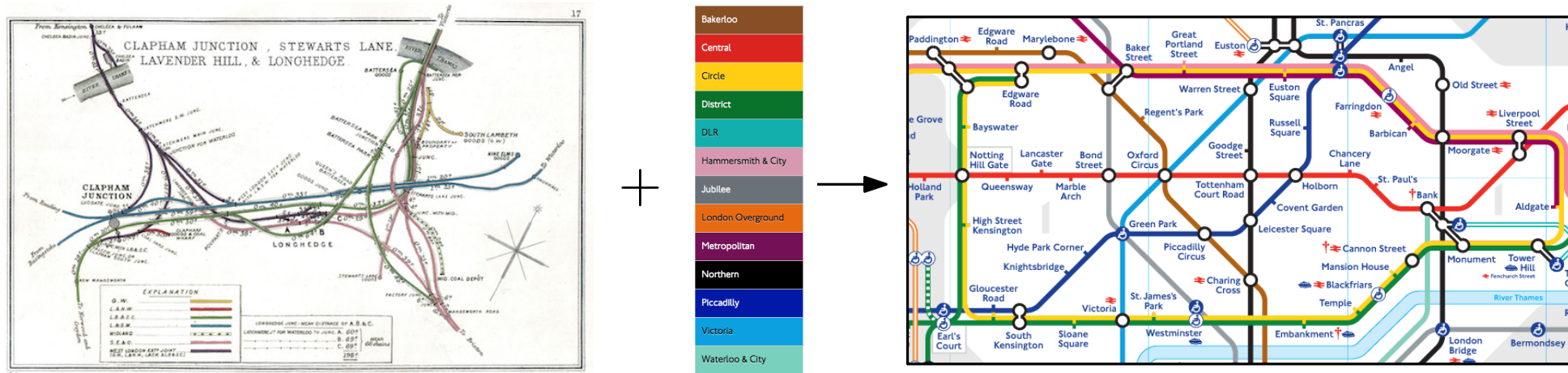
Divide the task into several subtasks:

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



determine positions of station names

Subtasks in Metro Map Layout



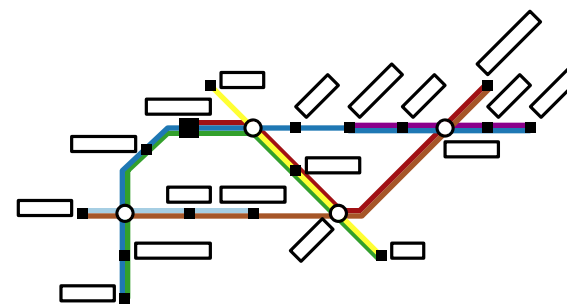
Input: ■ geographically embedded railway network G

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

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

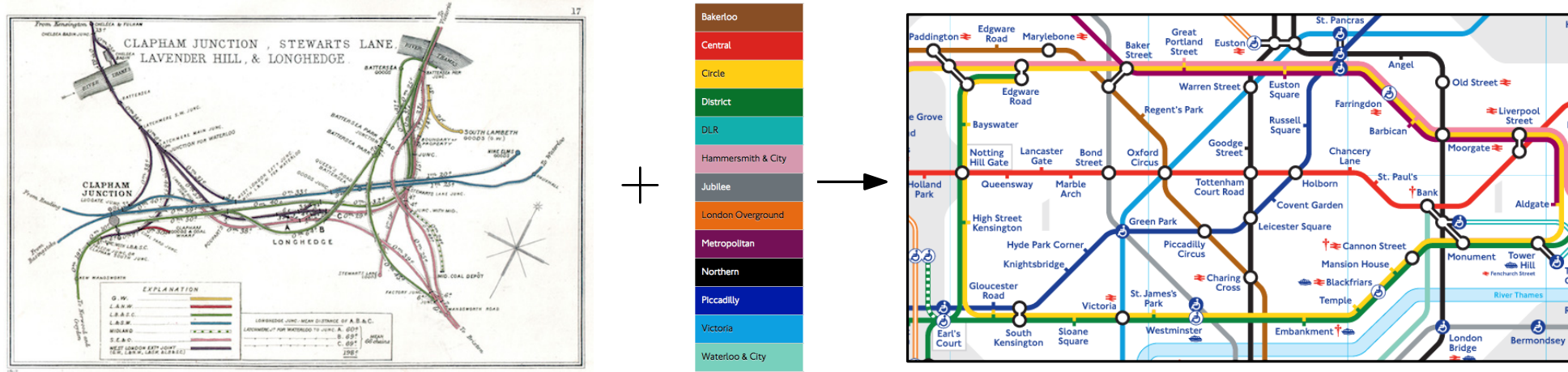
Divide the task into several subtasks:

- 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 $\mathcal{L}$ of metro lines serving G

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

Divide the task into several subtasks:

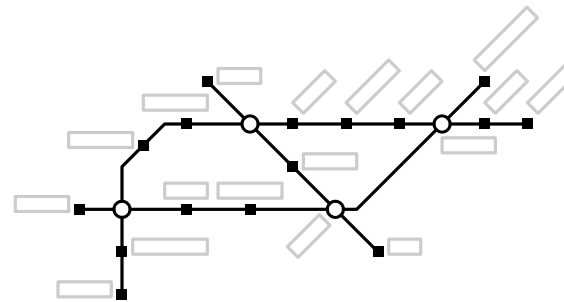
■ rendering and design choices

■ network layout

focus today

■ station labeling

■ metro line routing



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)

Find a **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?

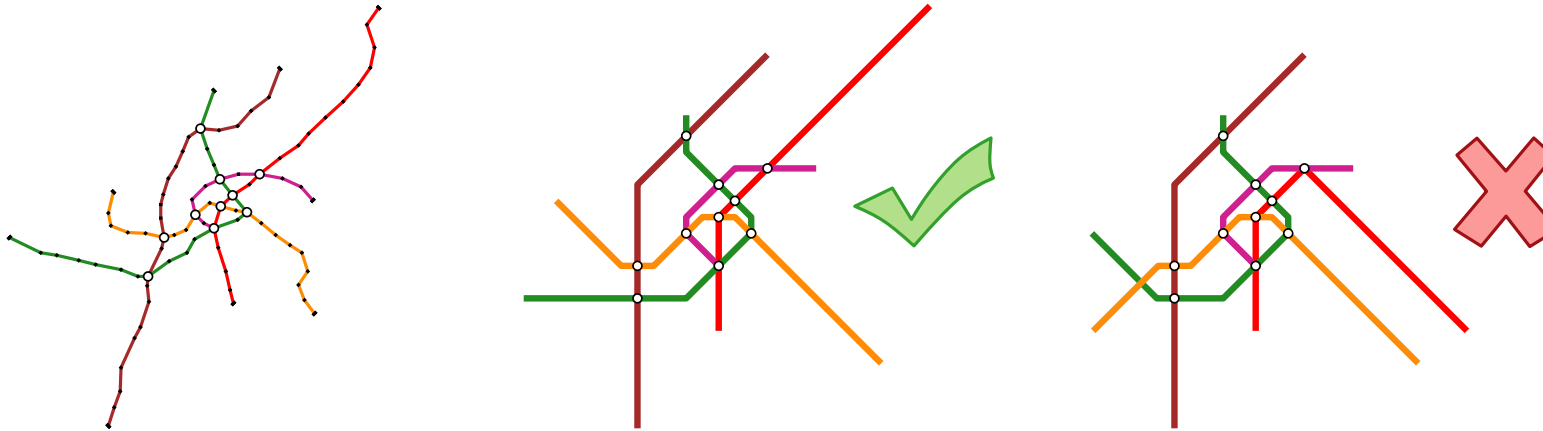
→ extract common principles of existing,
manually designed metro maps



Design Rules

(R1) Do not change the network topology.

- no new crossings
- no changes in circular vertex orders

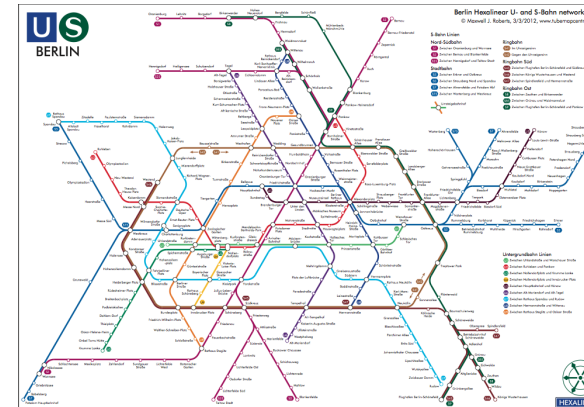
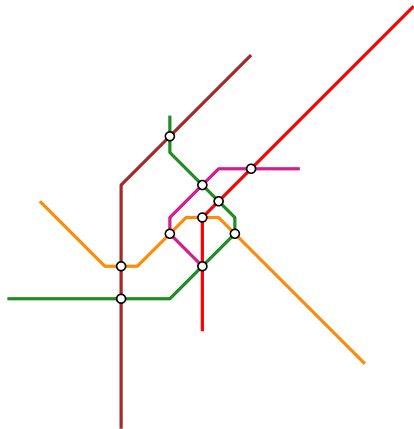
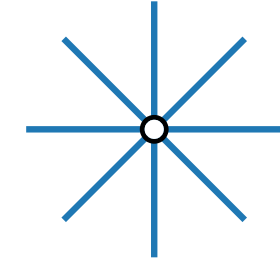


Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

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



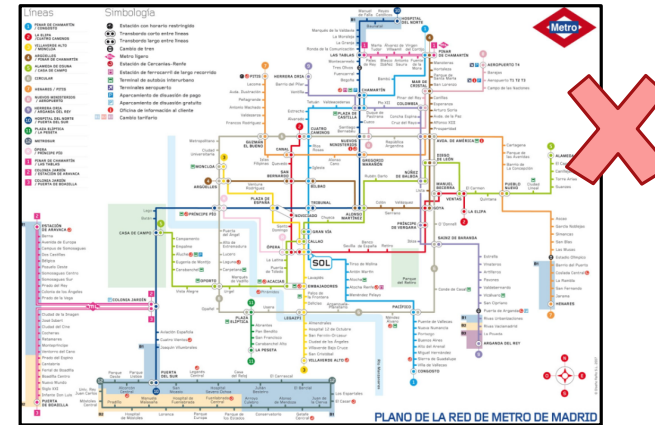
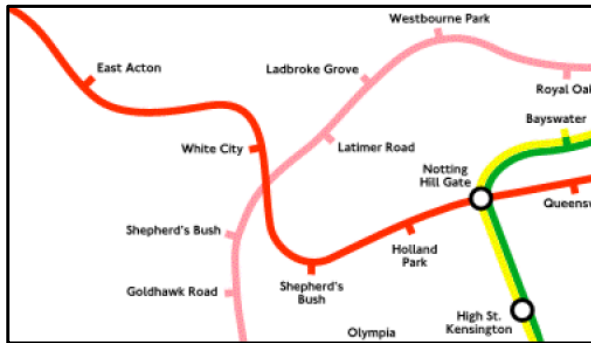
Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.

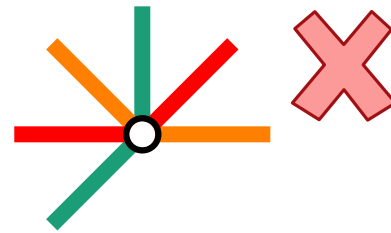
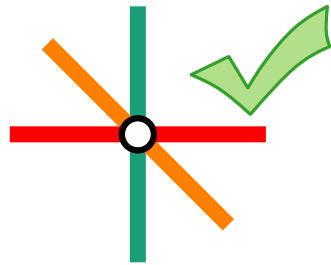
(R3) Draw metro lines as simple and monotone as possible.

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



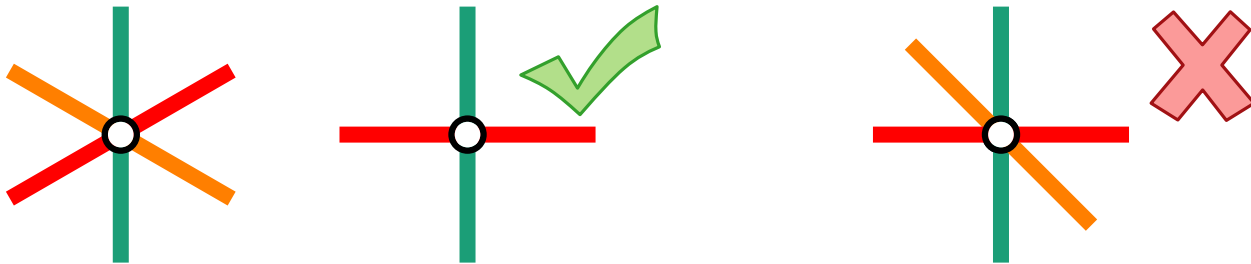
Design Rules

- (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

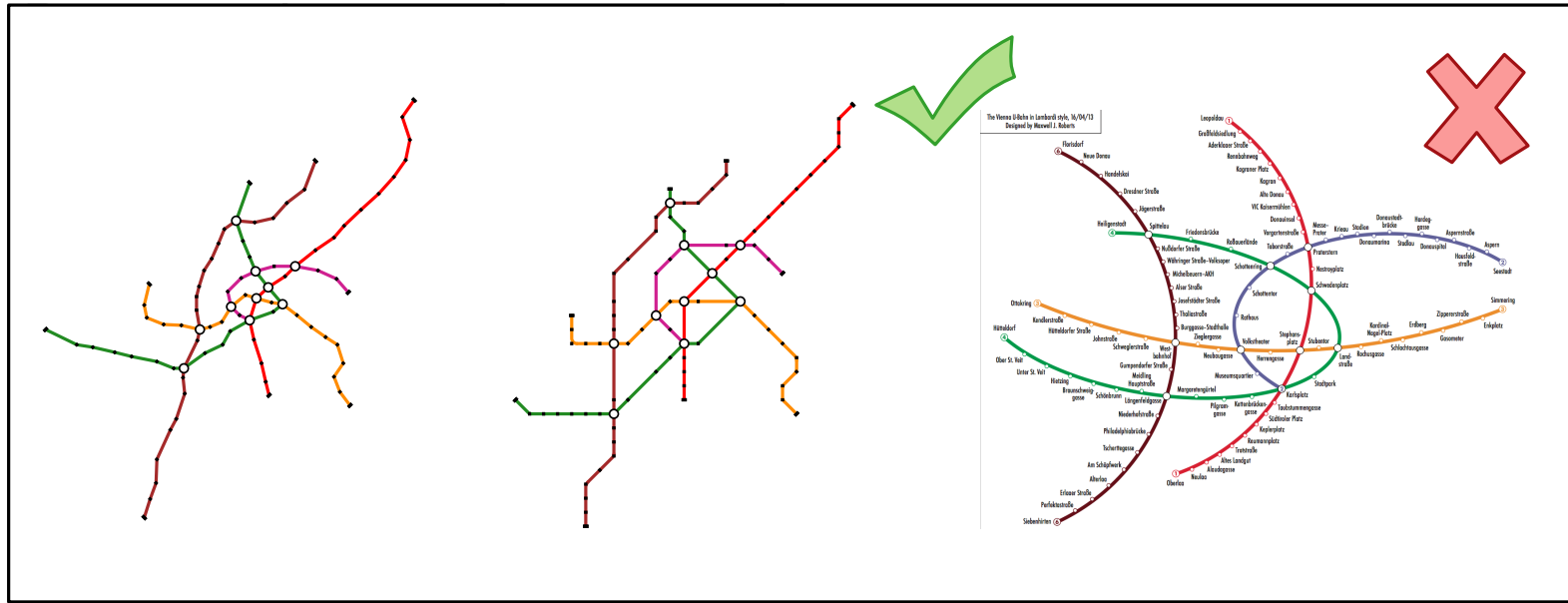
(R1)

(R2)

(R3)

(R4)

(R5)



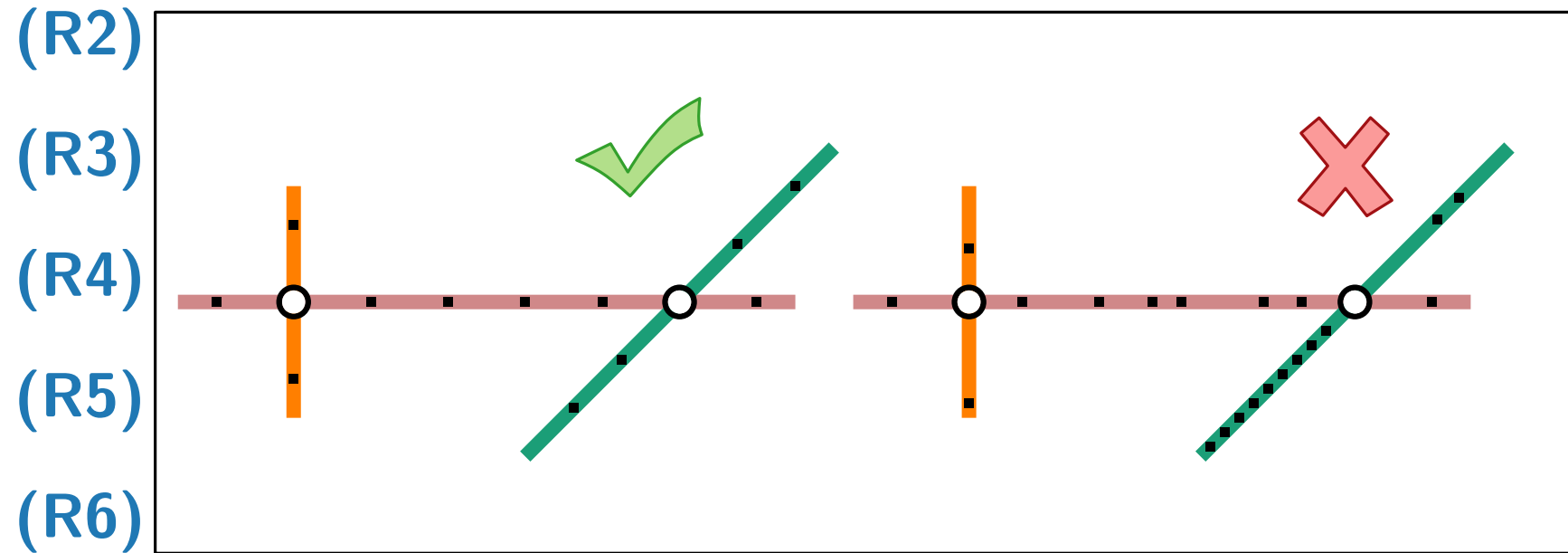
(R6) Minimize geometric distortion and displacement.

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

topographicity

Design Rules

(R1) Do not change the network topology.



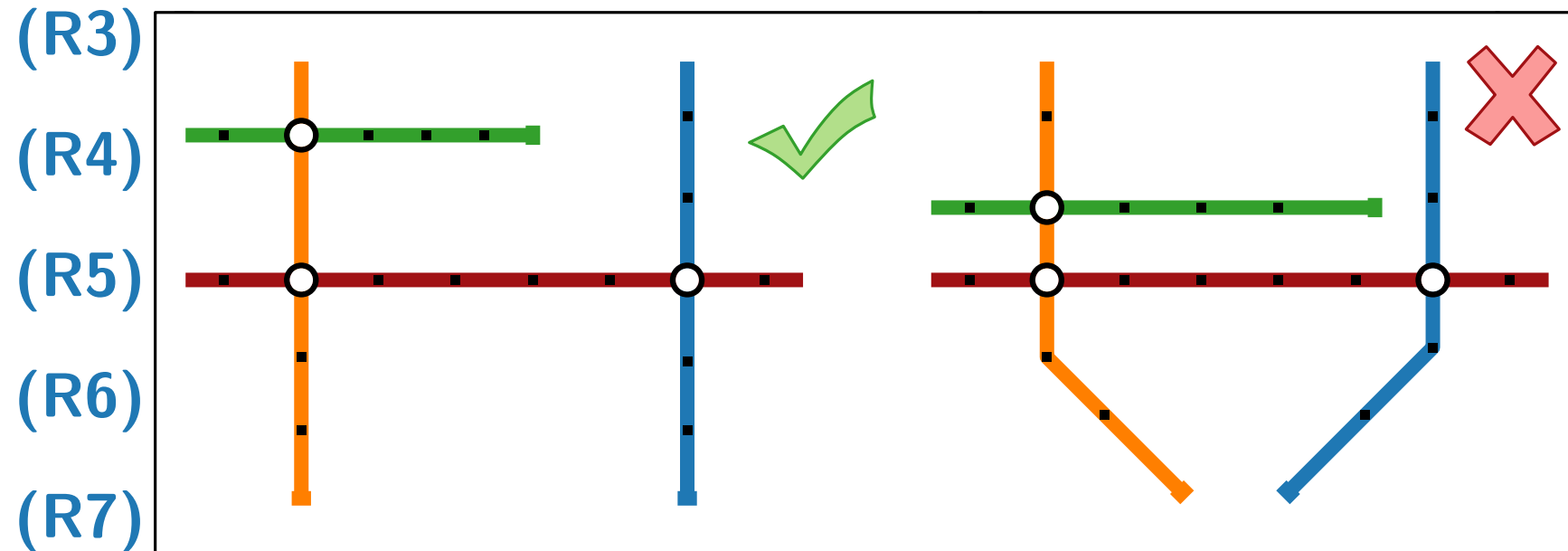
(R7) Use uniform edge lengths.

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

Design Rules

(R1) Do not change the network topology.

(R2) Restrict edge orientations.



(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)

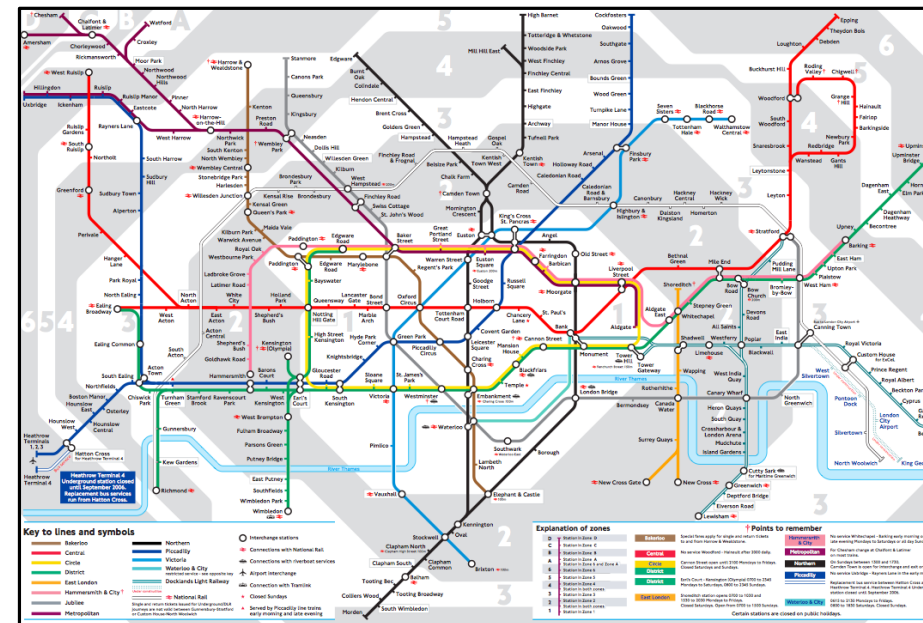
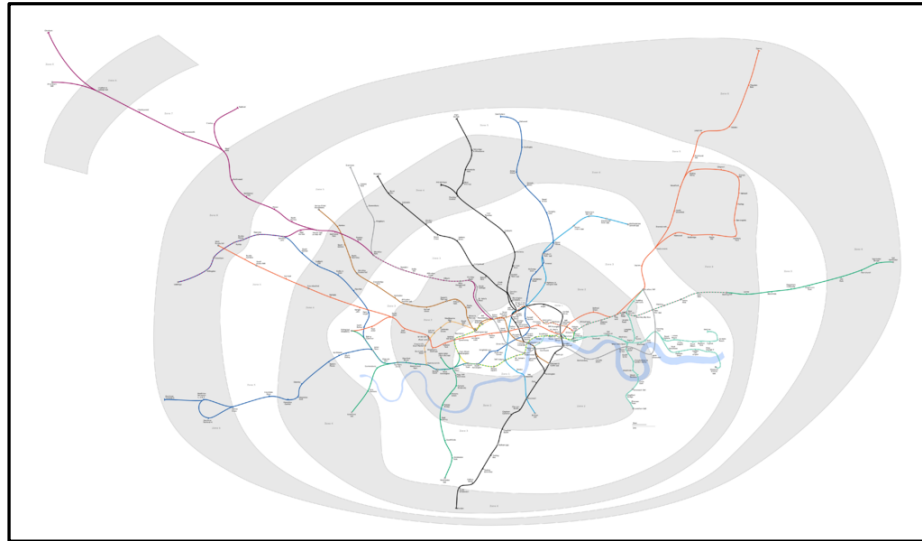
(R4)

(R5)

(R6)

(R7)

(R8)



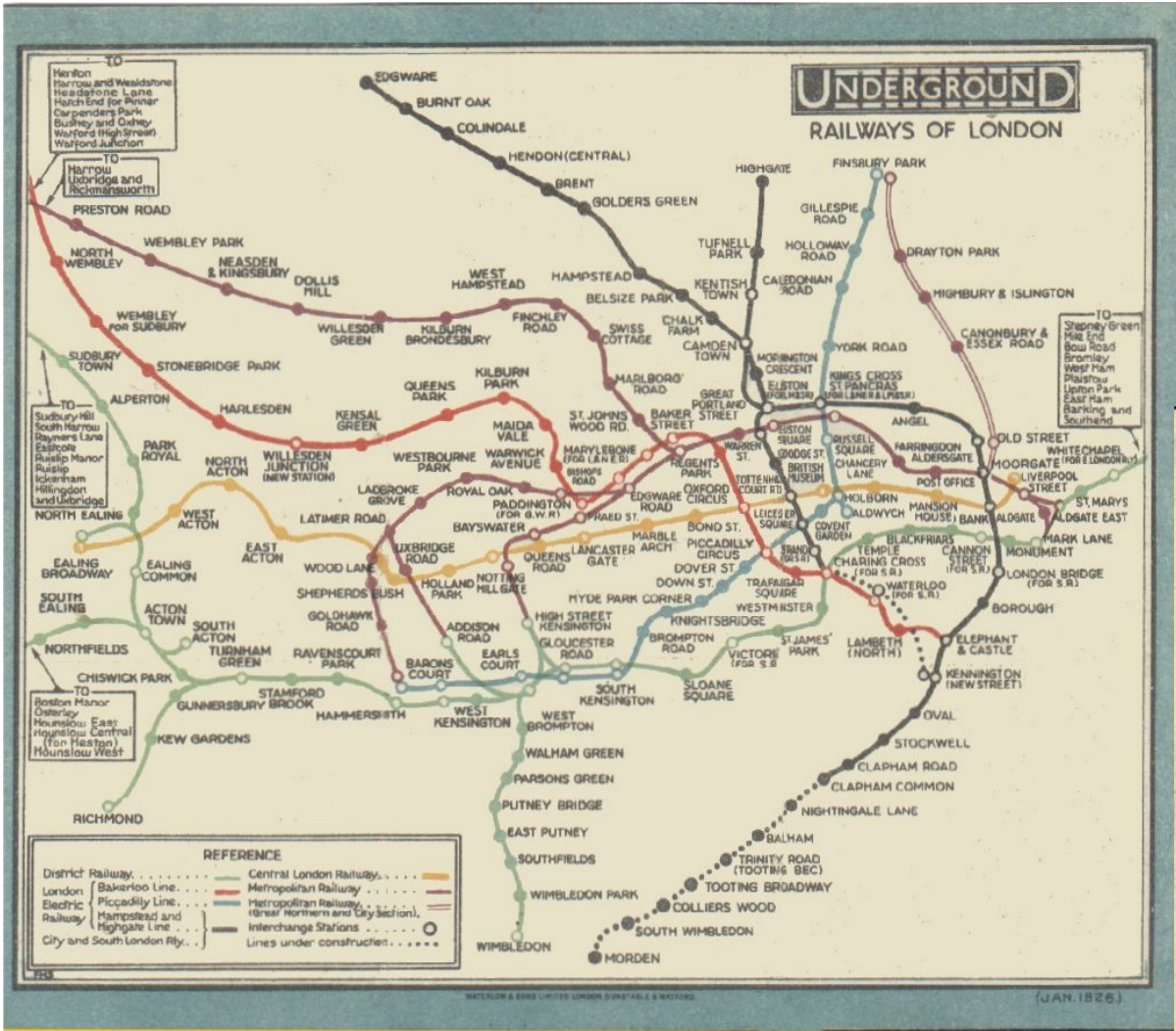
(R9) Avoid large empty spaces.

- 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.
- rules are potentially conflicting and need priorities

A Bit of History



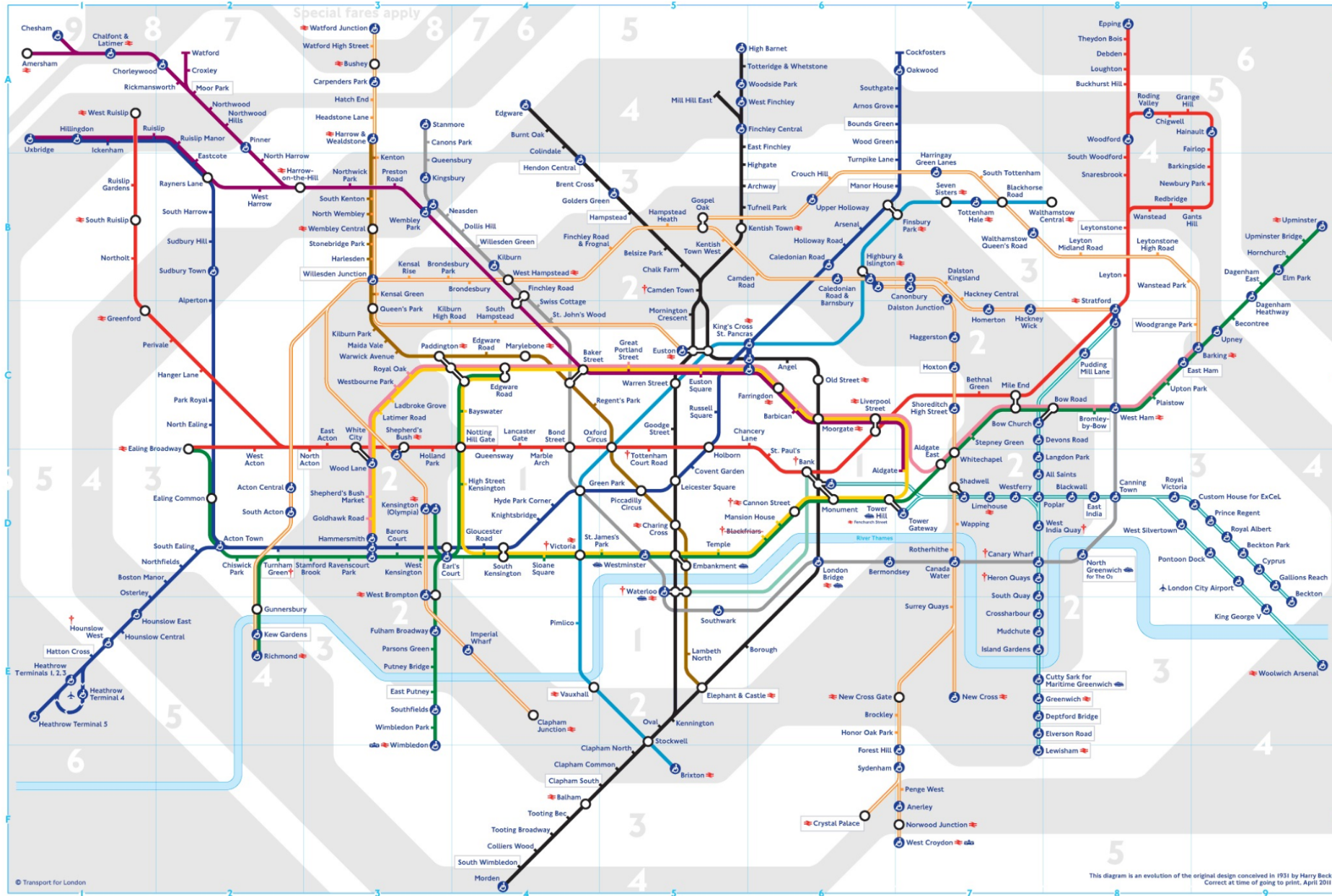
London 1927 (Fred H. Stingemore)

(c) Mike Yashworth

Henry Beck
London 1933



A Bit of History



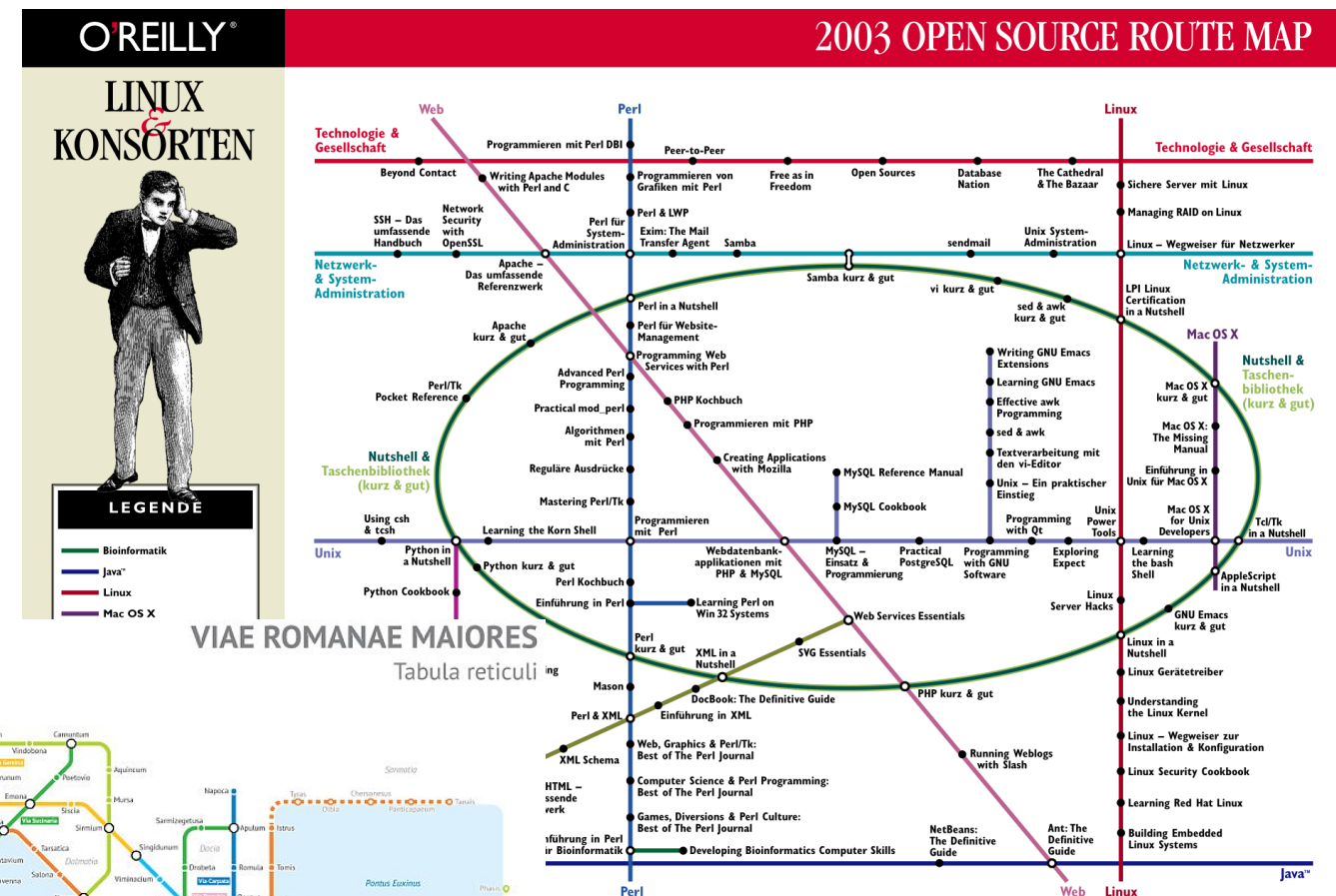
Tube Map
voted Design Icon 2006
(2nd after Concorde)

A Bit of History

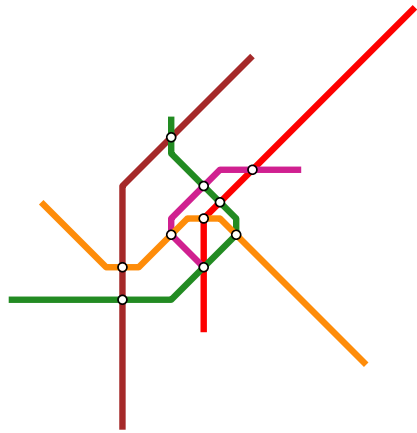


Berlin 1931 (redrawn by Maxwell Roberts)

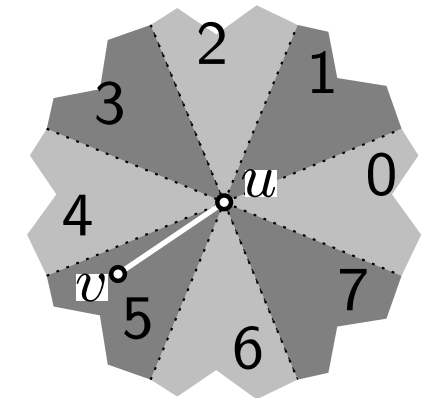
xkcd



Visualization of Graphs

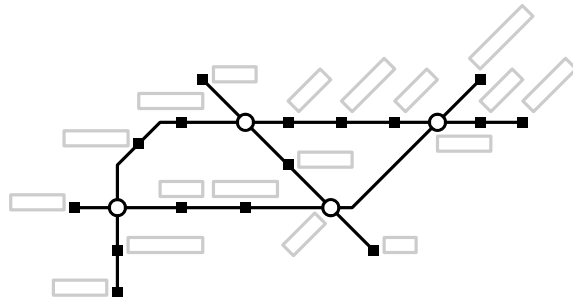


Lecture 12: Octilinear Graph Drawing Metro Map Layout



Part II: Complexity and Path-Based Schematization

Alexander Wolff



Complexity

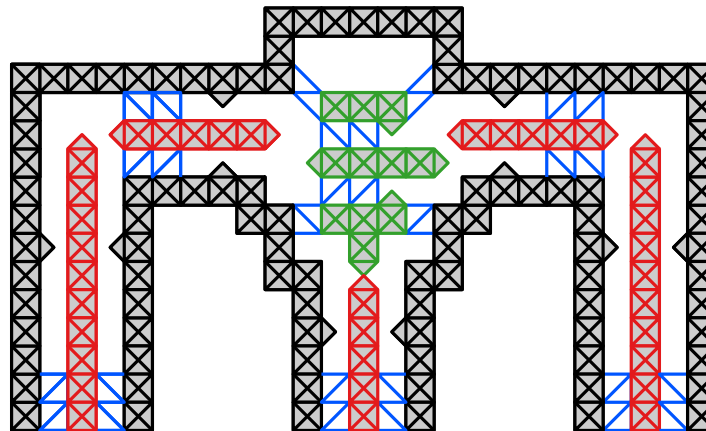
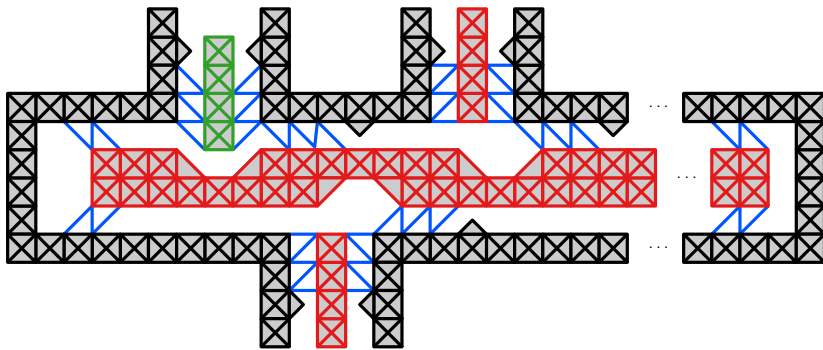
Theorem 1.

[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



Remark.

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

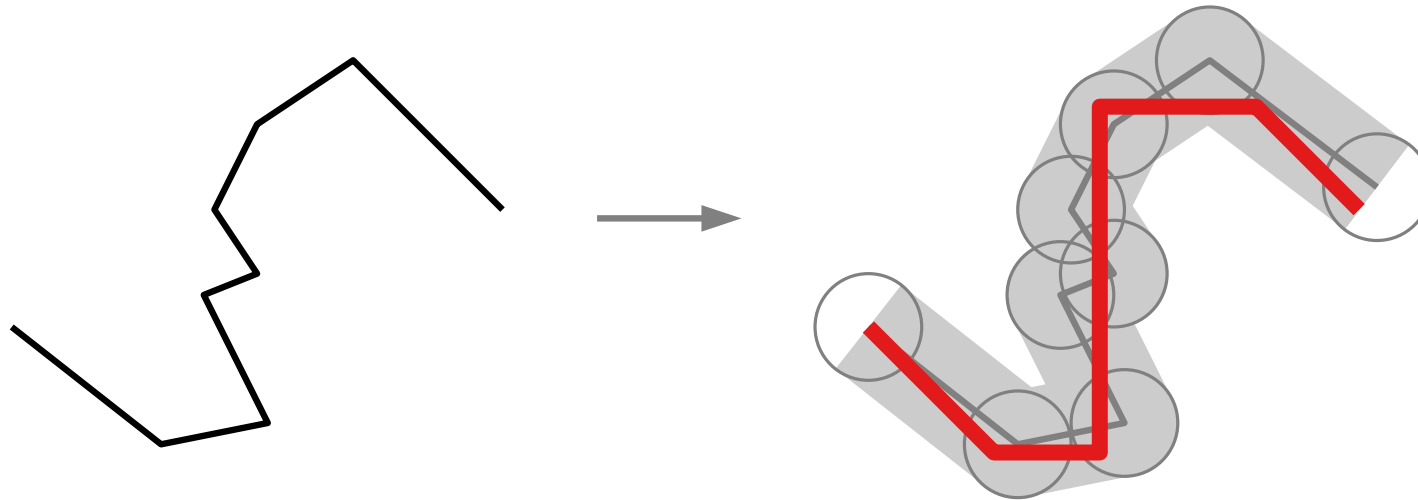
Path-Based Schematization

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

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

■ bounded displacement **(R6)**

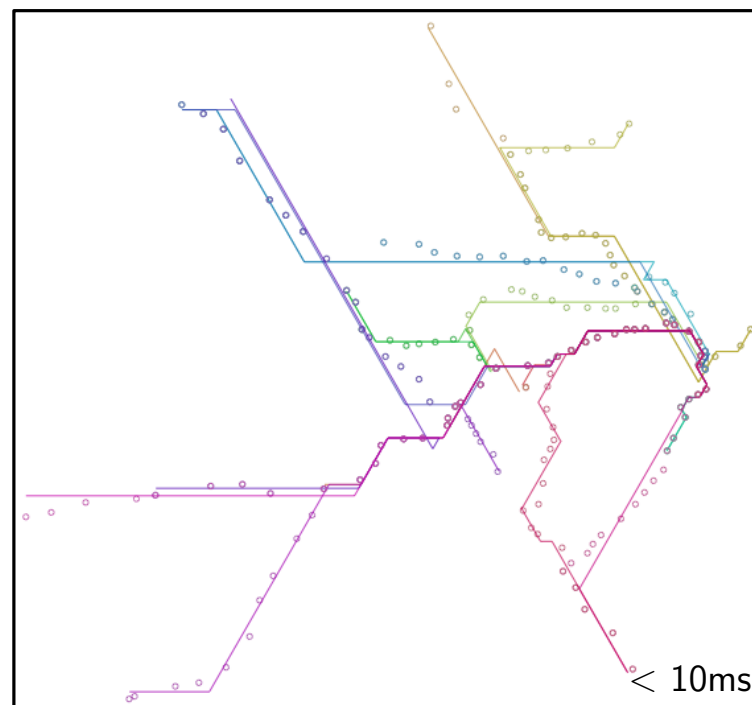
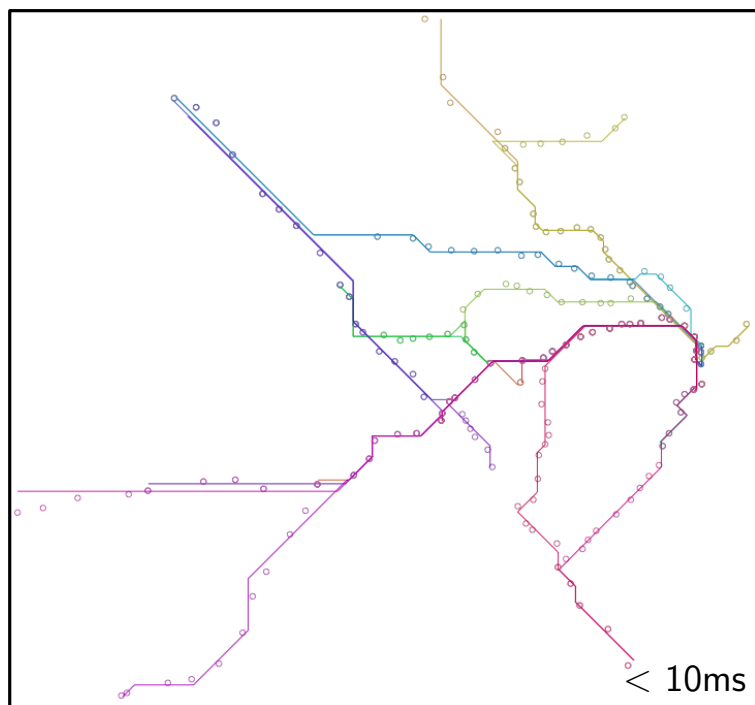
Criteria. ■ minimize number of links **(R3)**



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 $O(n^2) + \text{solve}(\text{LP})$ 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 $\mathcal{C} = \{i \cdot 90^\circ / d \mid i \in \mathbb{Z}\}$.

[Gemsa et al. 2011]



Path-Based Schematization – Discussion

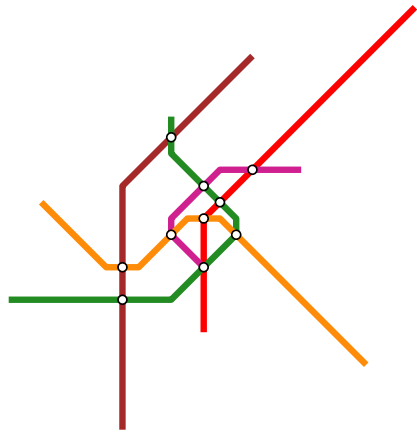
Pros.

- 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

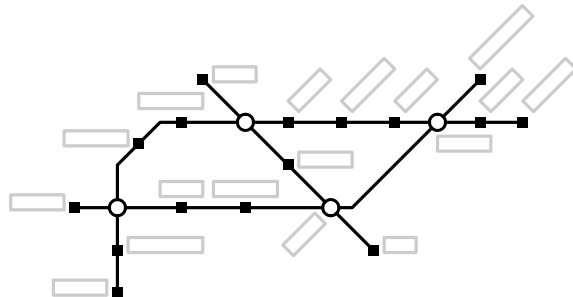
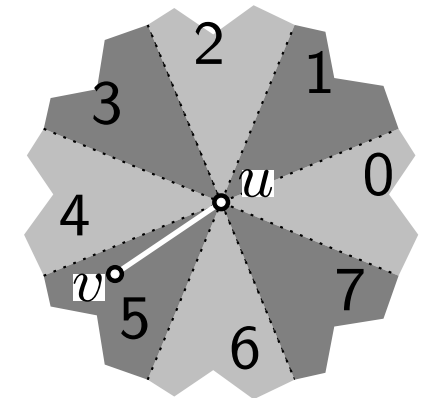
Visualization of Graphs



Lecture 12: Octilinear Graph Drawing Metro Map Layout

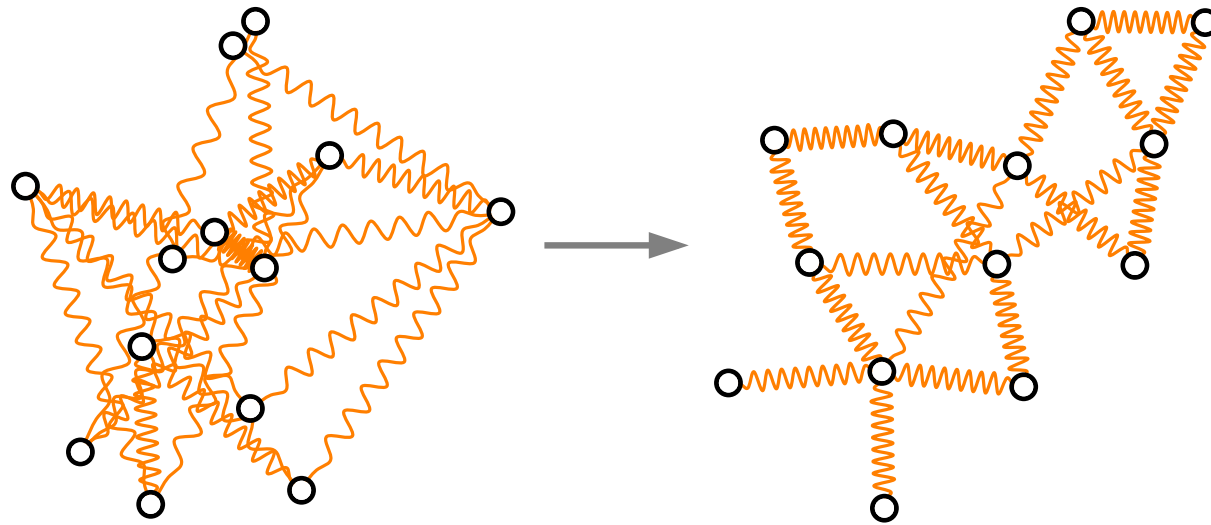
Part III: Force-Based Schematization

Alexander Wolff



Force-Based Schematization

Idea. Apply well known force-based graph drawing approach



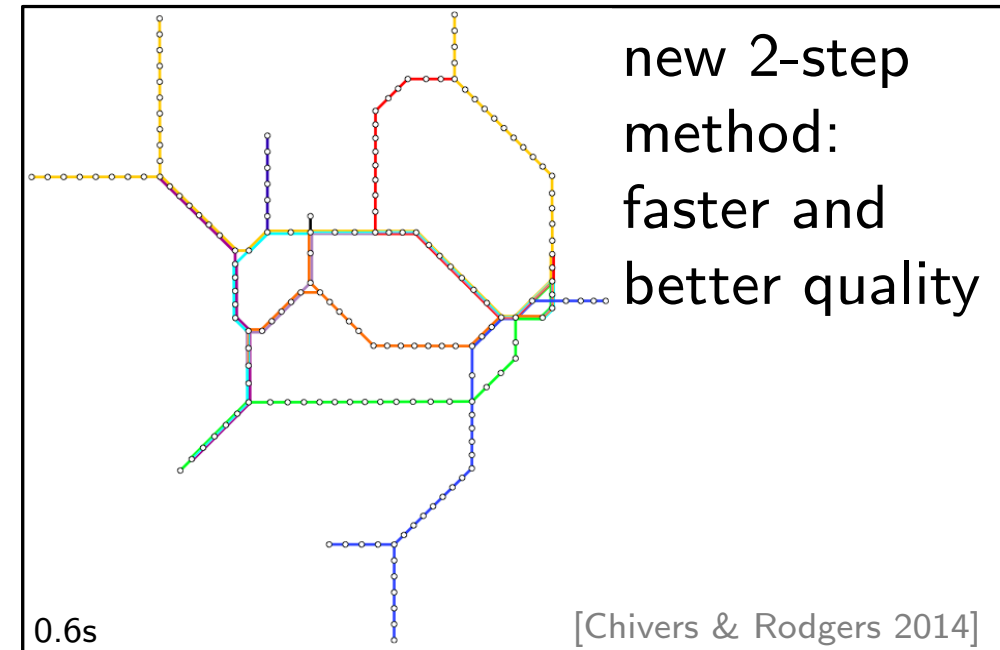
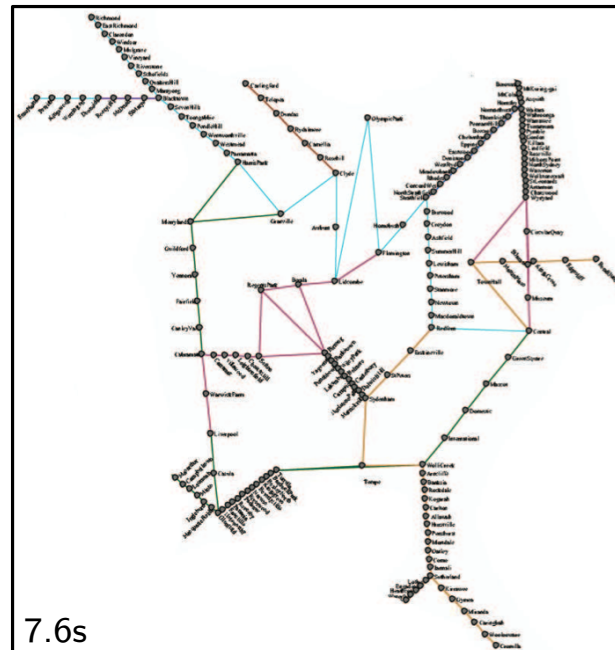
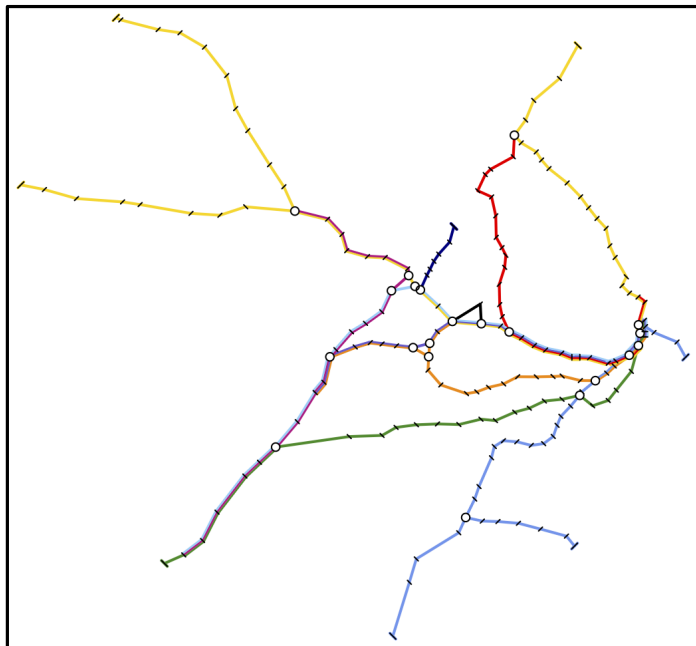
Recall.

- vertices are charged particles repelling each other
- edges are springs pulling edges into target length
- iteratively calculate and apply forces until system stabilizes
 - define additional forces to model subset of metro map design rules

Force-Based Schematization – Octilinear

[Hong et al. 2006]

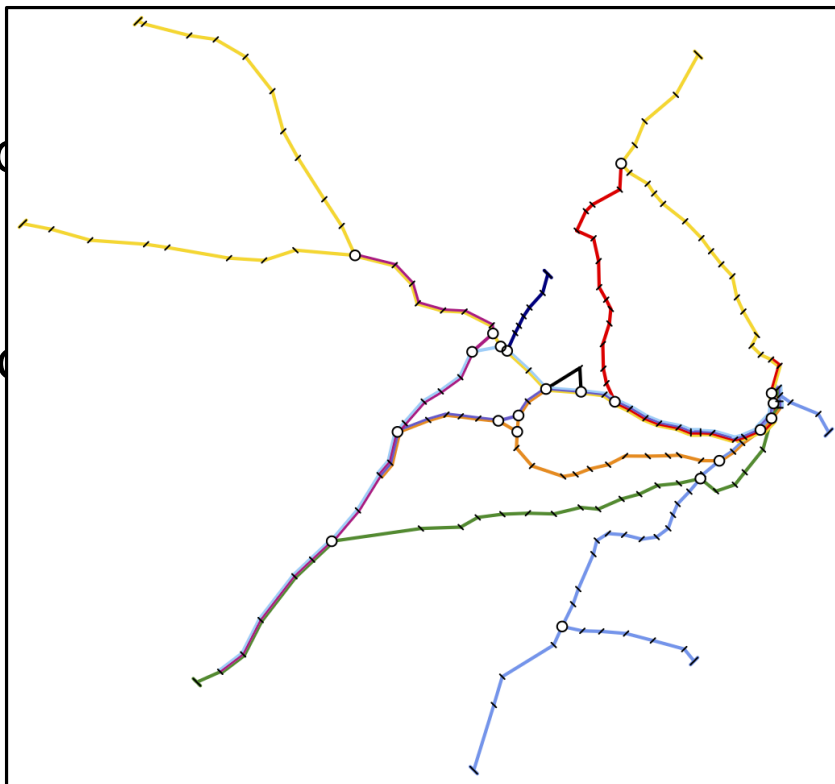
- contract degree-2 vertices into weighted edges (R3)
- define octilinear magnetic field attracting edges (R2)
- only apply topology preserving vertex moves (R1)
- spring lengths model uniform edge lengths (R7)
- vertex repulsion models feature separation (R8)
- station labels placed in independent 2nd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



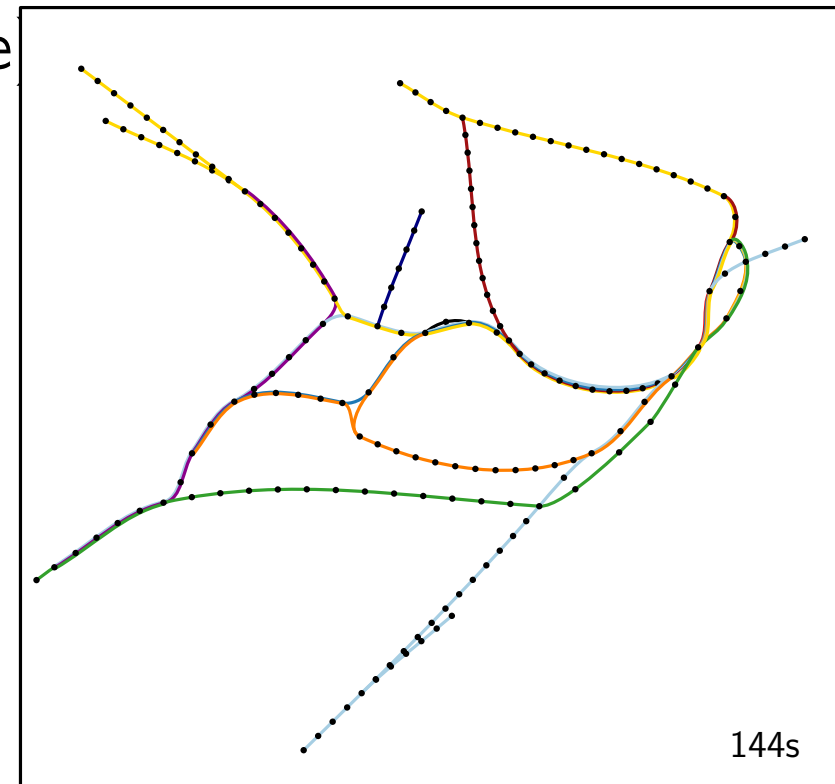
Force-Based Schematization – Bézier

[Fink et al. 2013]

- convert (octilinear) input layout into Bézier curves
→ 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
- merge c
- forces i
- apply to

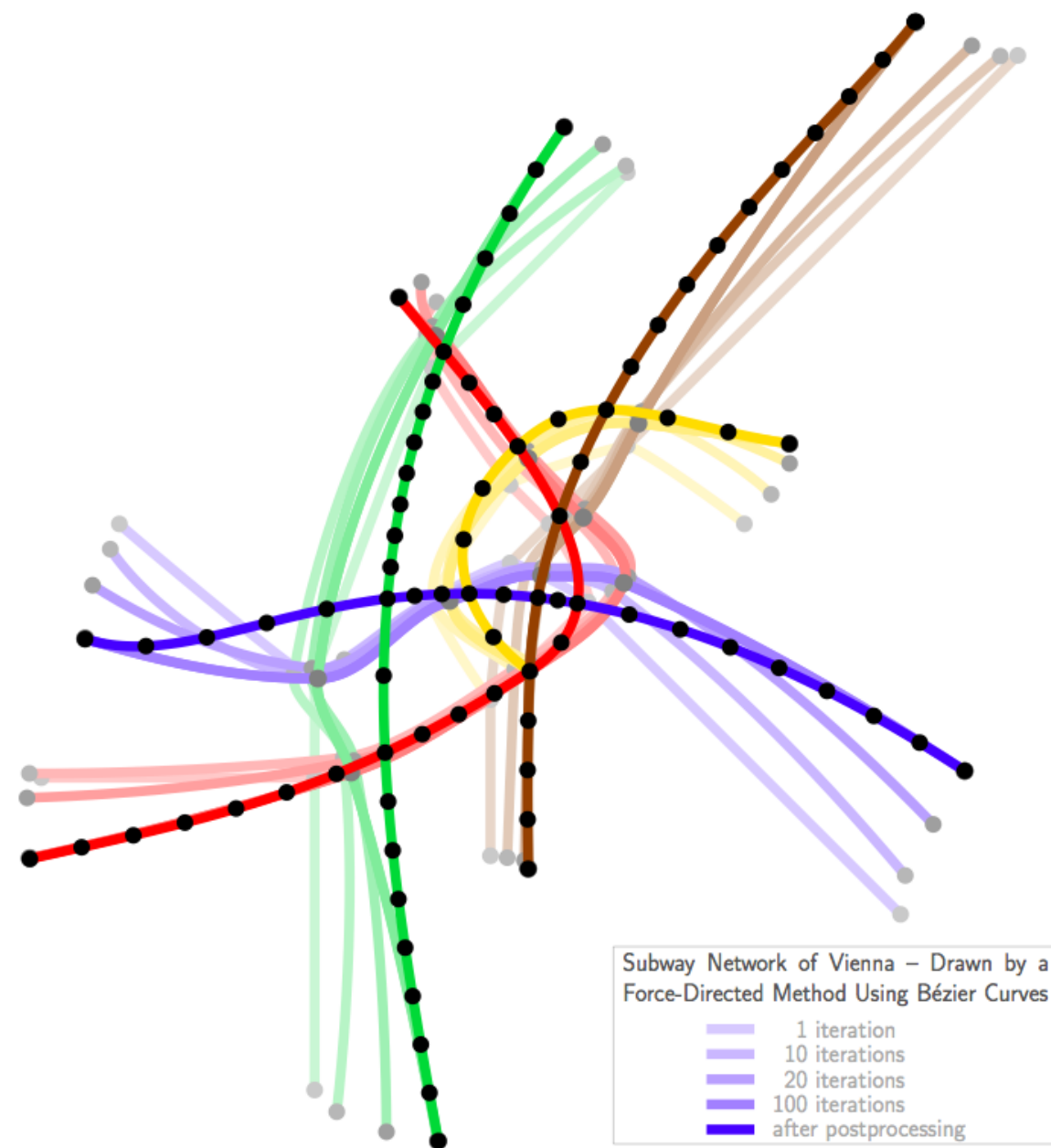
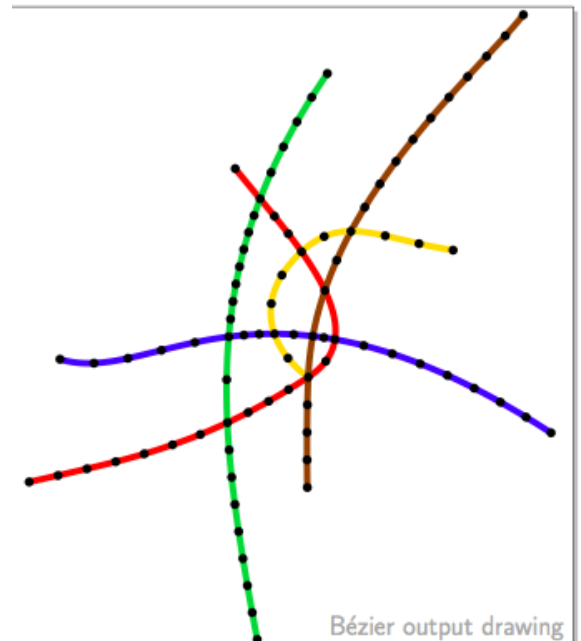
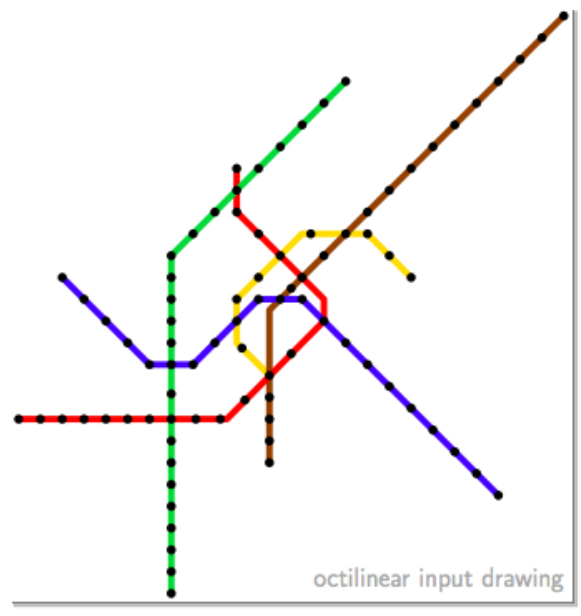


curvature
)
(R5)
y **(R1)**



Force-Based Schematization – Bézier

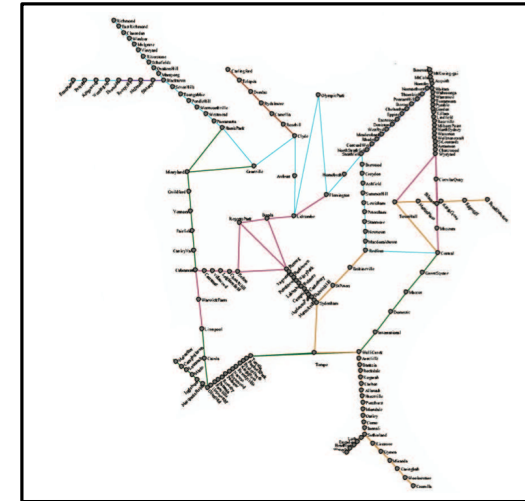
[Fink et al. 2013]



Force-Based Schematization – Discussion

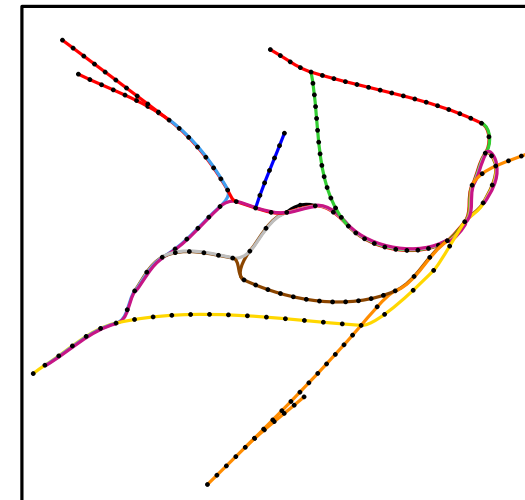
Octilinear.

- 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

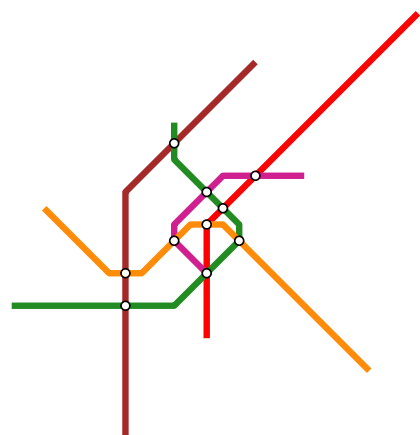


Bézier.

- 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

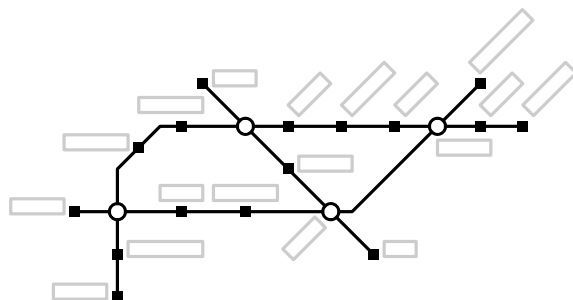
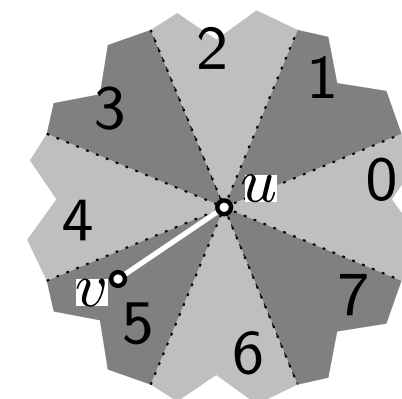


Visualization of Graphs



Lecture 12: Octilinear Graph Drawing Metro Map Layout

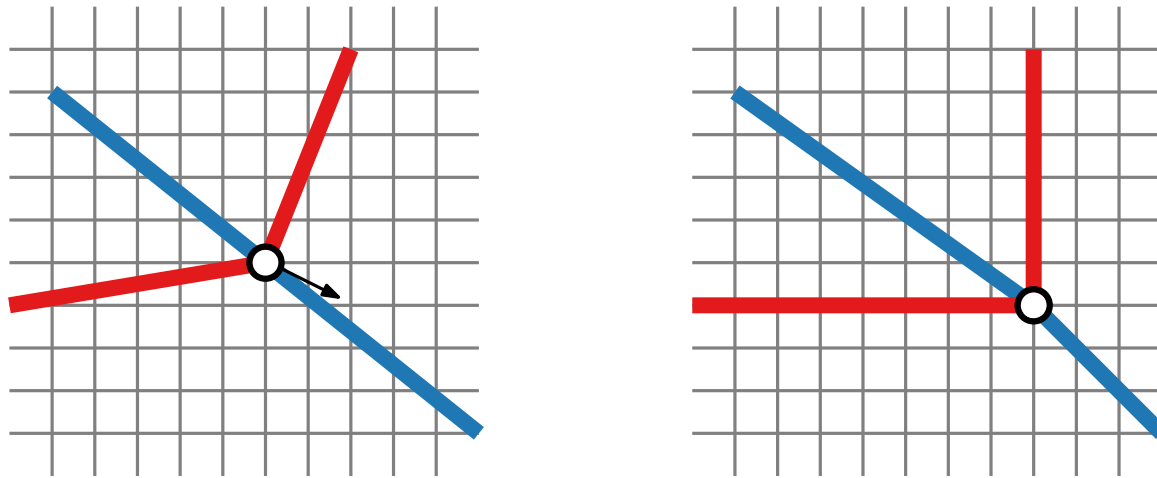
Part IV: Local Schematization



Alexander Wolff

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, ...

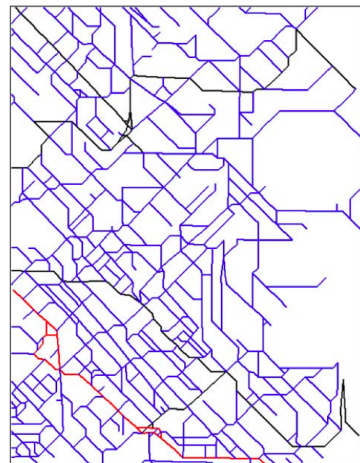


Local Schematization

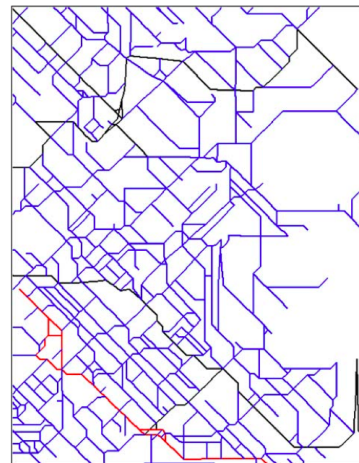
- Idea.**
- define a (multi-criteria) layout quality function
 - improve layout quality step-by-step by locally moving vertices to better nearby (grid) positions
 - applicable optimization techniques: hill climbing, simulated annealing, ant colony optimization, ...

[Avelar & Müller 2000]

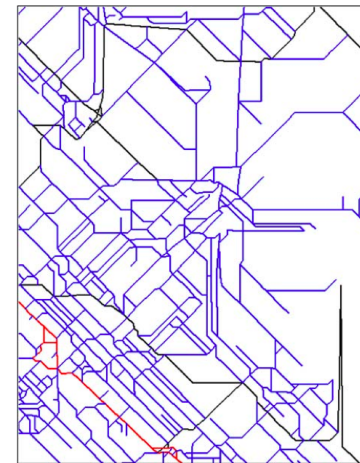
- calculate best vertex position in each criterion (octilinearity **(R2)**, min. separation **(R8)**)
- move vertex to average of positions without violating topology **(R1)**



100



1,000



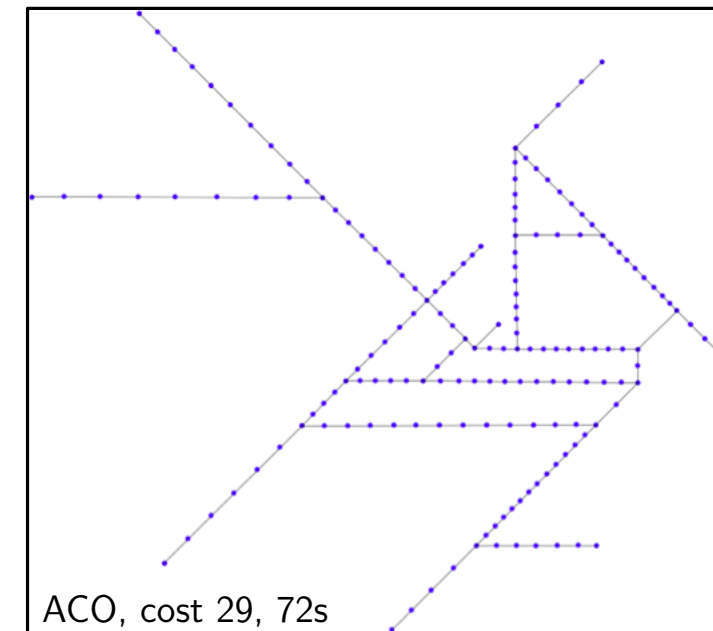
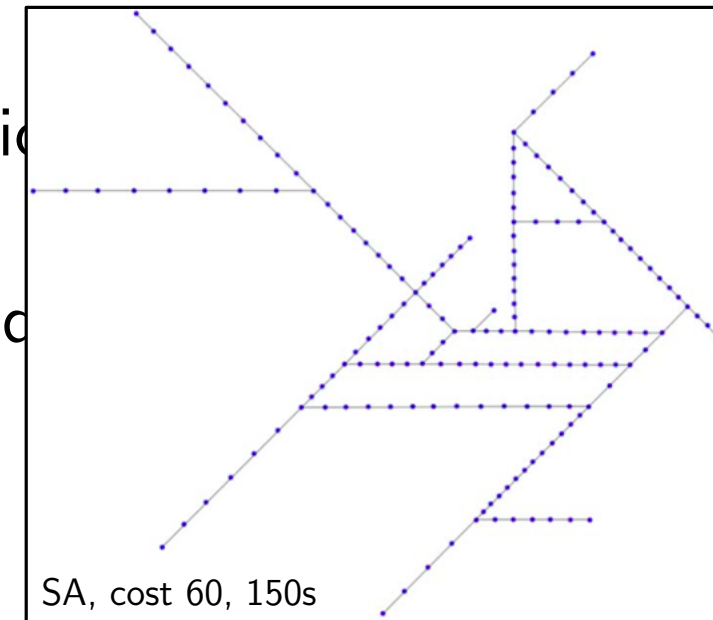
10,000

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, ...

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

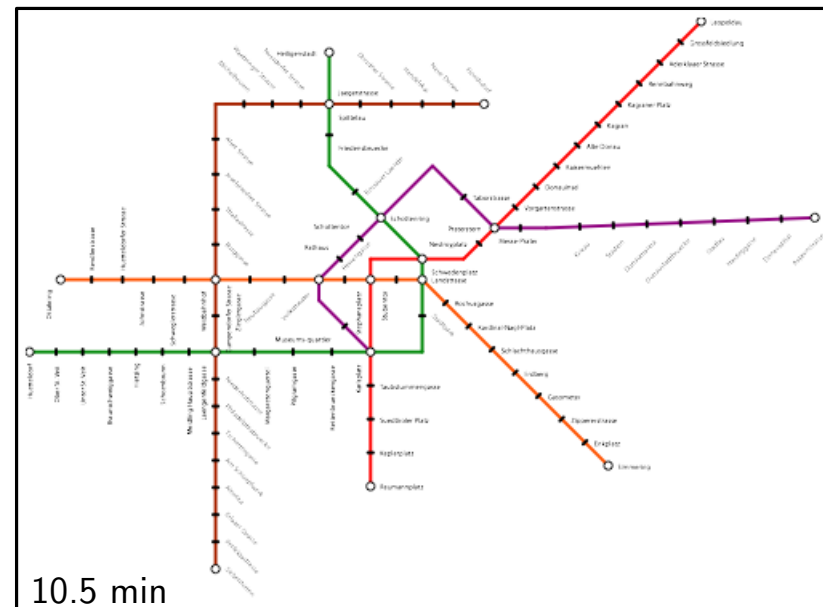
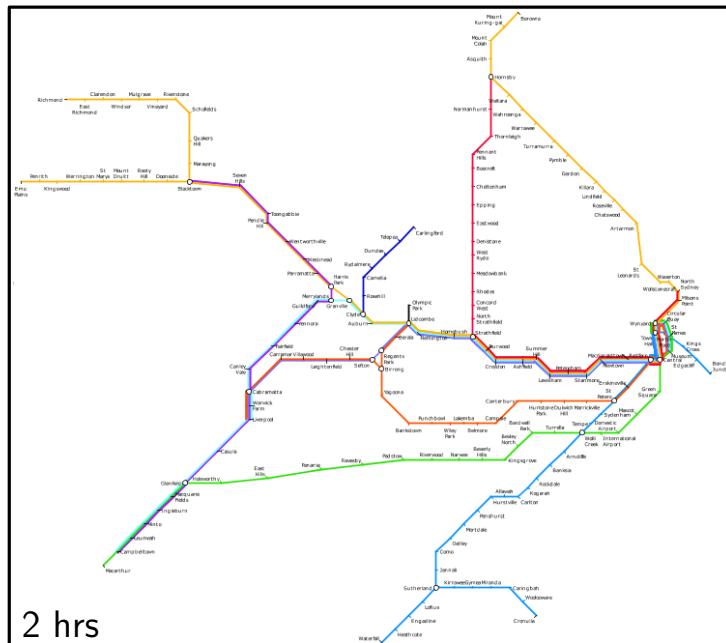
- weighted multicriteria function
- contract degree-2 vertices prior to optimization
- implemented more design rules (topology **(R1)**, octilinearity **(R2)**, displacement **(R6)**, edge lengths **(R7)**, separation **(R8)**)
- simulated annealing (2006) and ant colony optimization (2013)



Local Schematization

[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

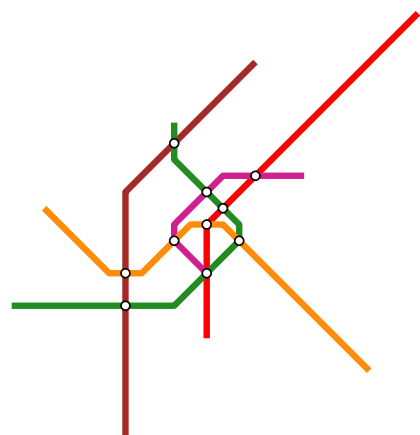
Pros.

- 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

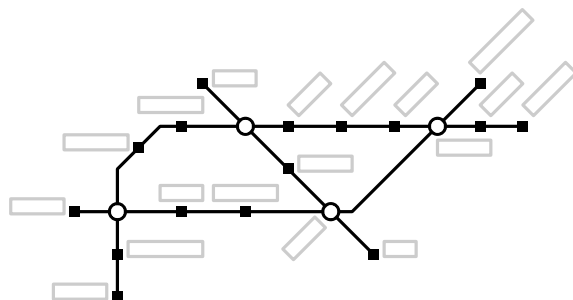
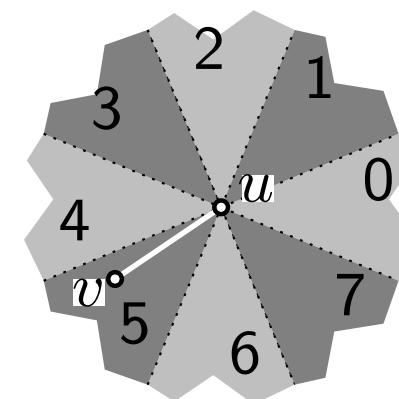
Visualization of Graphs



Lecture 12: Octilinear Graph Drawing Metro Map Layout

Part V: Mixed-Integer Programming

Alexander Wolff



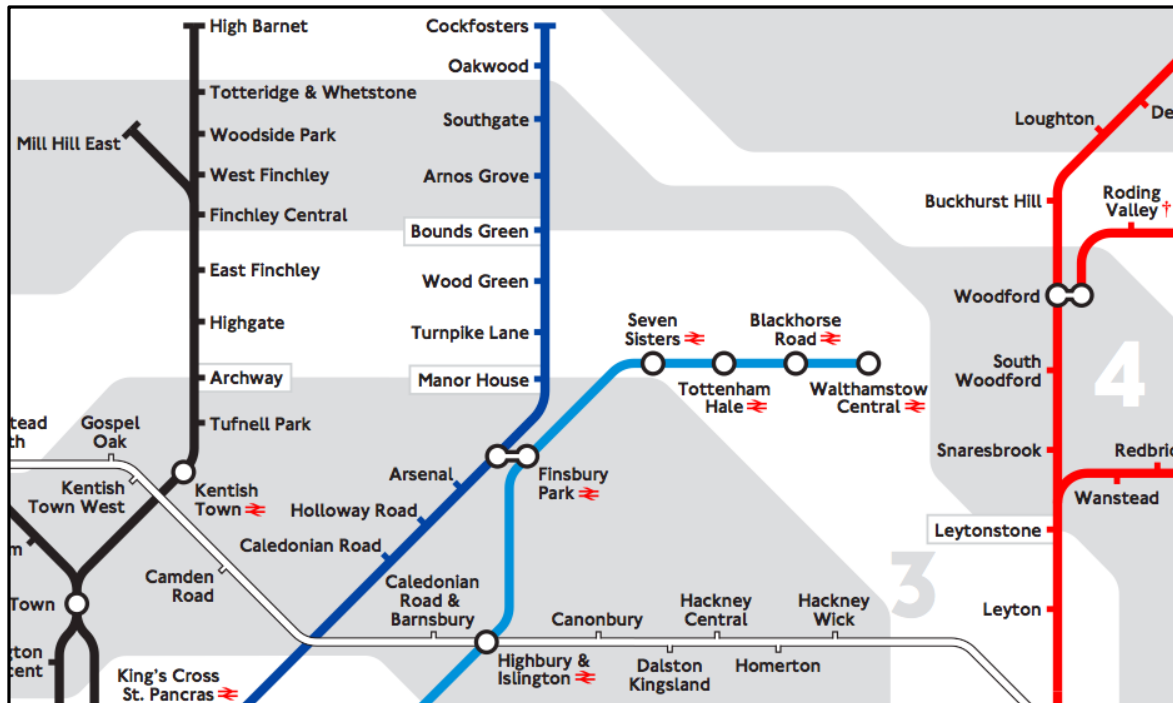
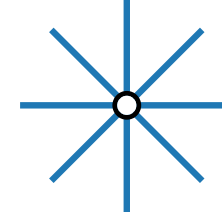
Mixed-Integer Programming

[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
→ 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)

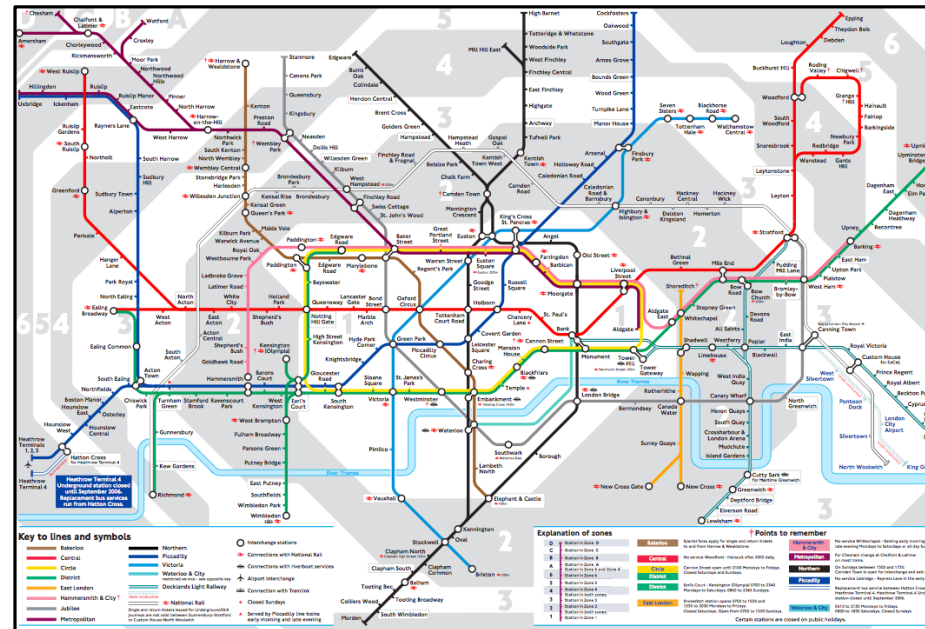
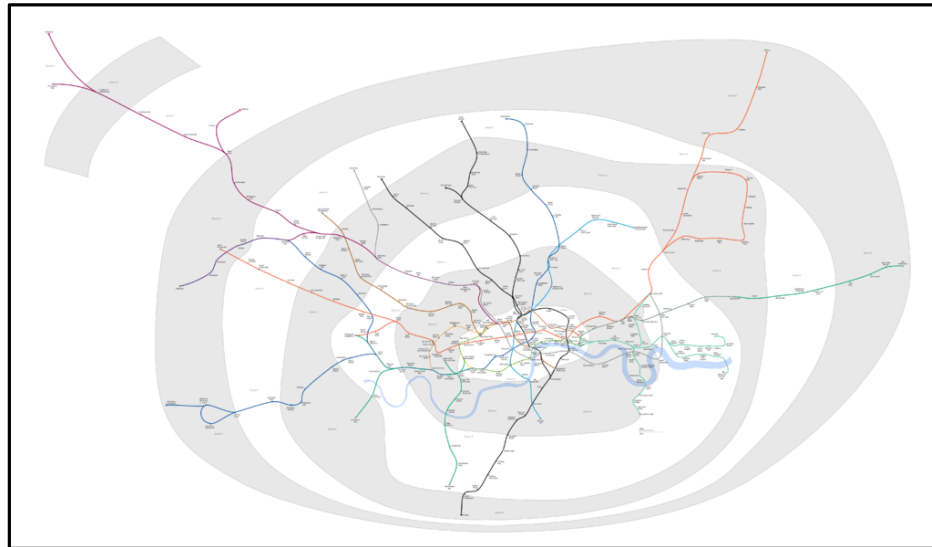
Hard Constraints

- (R1) preserve embedding/**topology** and **planarity**
- (R2) draw all edges **octilinearly**
- (R7) enforce **minimum edge length** $\ell_{\min}$
- (R8) enforce **minimum feature separation** $d_{\min}$



(R6) minimize geometric **distortion**

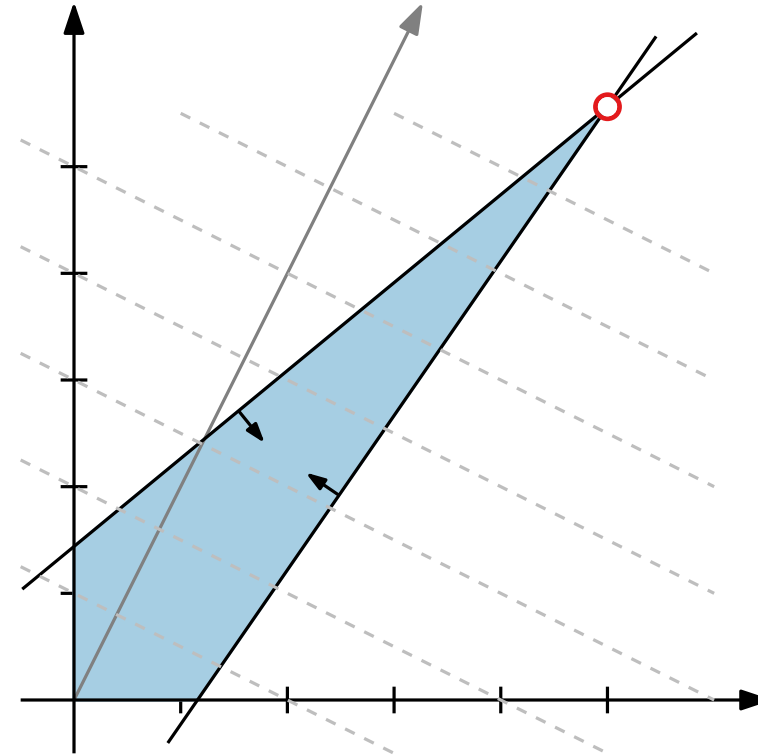
(R7) minimize total **edge length**



Linear Programming

Linear Programming (LP) is an efficient optimization method for

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



Example.

$$y \leq 0.9x + 1.5$$

$$y \geq 1.4x - 1.3$$

$$\text{maximize } x + 2y$$

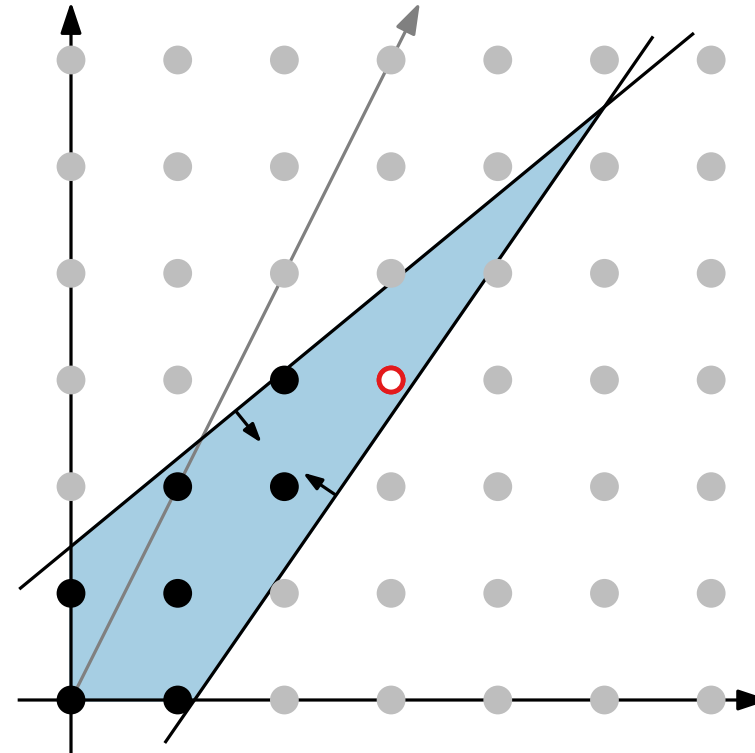
Linear Programming

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

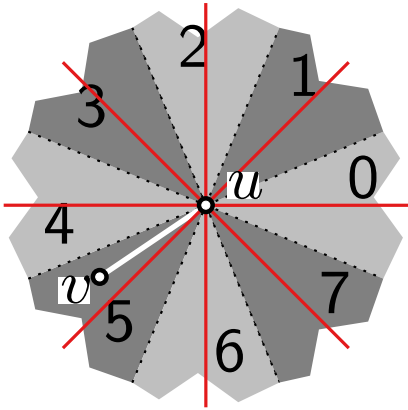


Theorem 5.

Metro map layout can be modeled as MIP such that

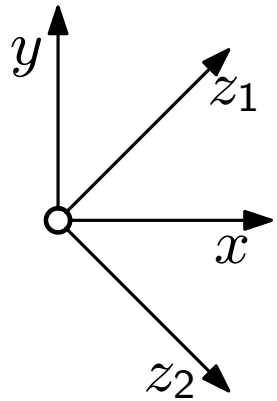
- hard constraints $\rightarrow$ linear constraints
- soft constraints $\rightarrow$ linear objective function

Sectors and Coordinates



Sectors.

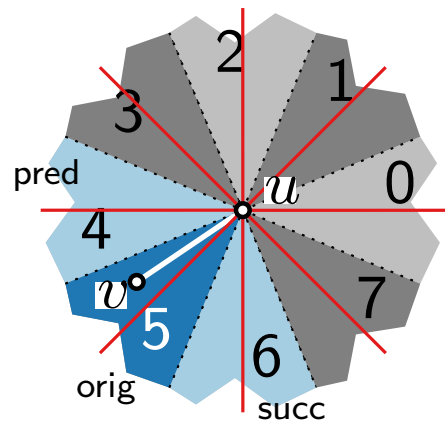
- for each vertex u partition the plane into eight sectors numbered 0–7
here: $\text{sec}(u, v) = 5$ in the input
- number octilinear edge directions accordingly
here, e.g., $\text{dir}(u, v) = 5$ in the solution (output)



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

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

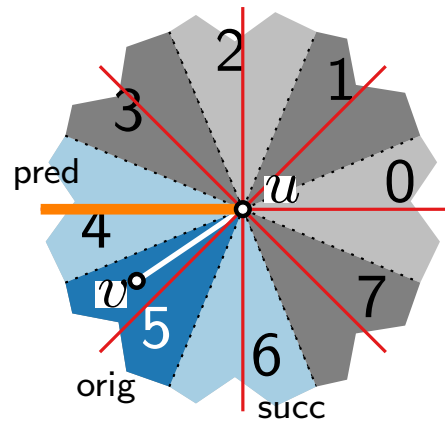
How to model this using linear constraints in a MIP?

Introduce **binary variables**

$$\alpha_{\text{pred}}(u, v) \quad \alpha_{\text{orig}}(u, v) \quad \alpha_{\text{succ}}(u, v)$$

to select exactly one of the three sectors.

Octilinearity and Relative Position



Predecessor sector.

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

very large constant (how large, actually?)

How does this work?

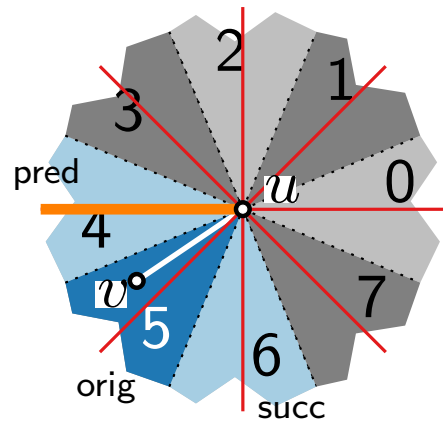
Case 1: $\alpha_{\text{pred}}(u, v) = 1$

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

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



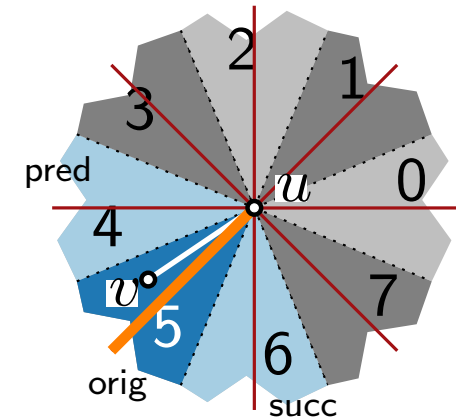
Predecessor sector.

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

very large constant

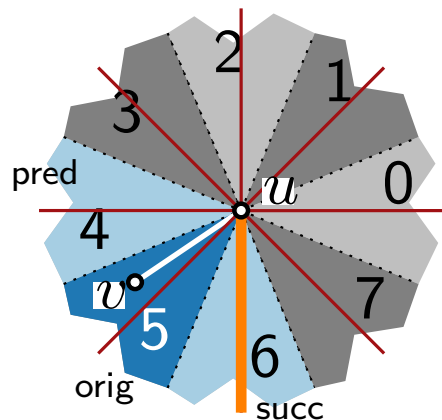
Original sector.

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



Successor sector.

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

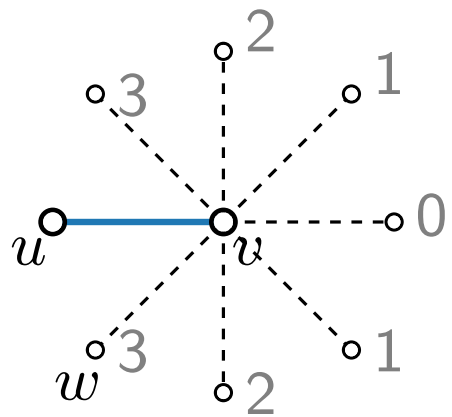


Objective Function

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

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

Example: line bends (R3/R4)



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

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

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

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

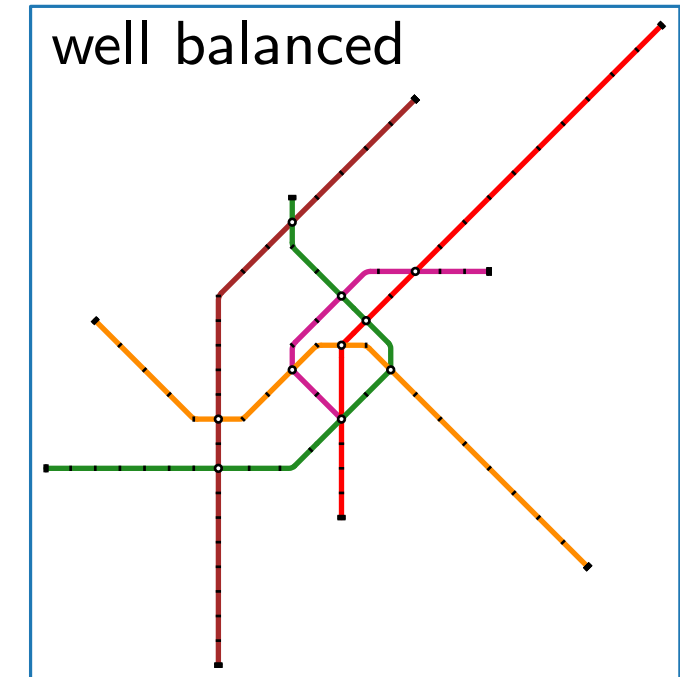
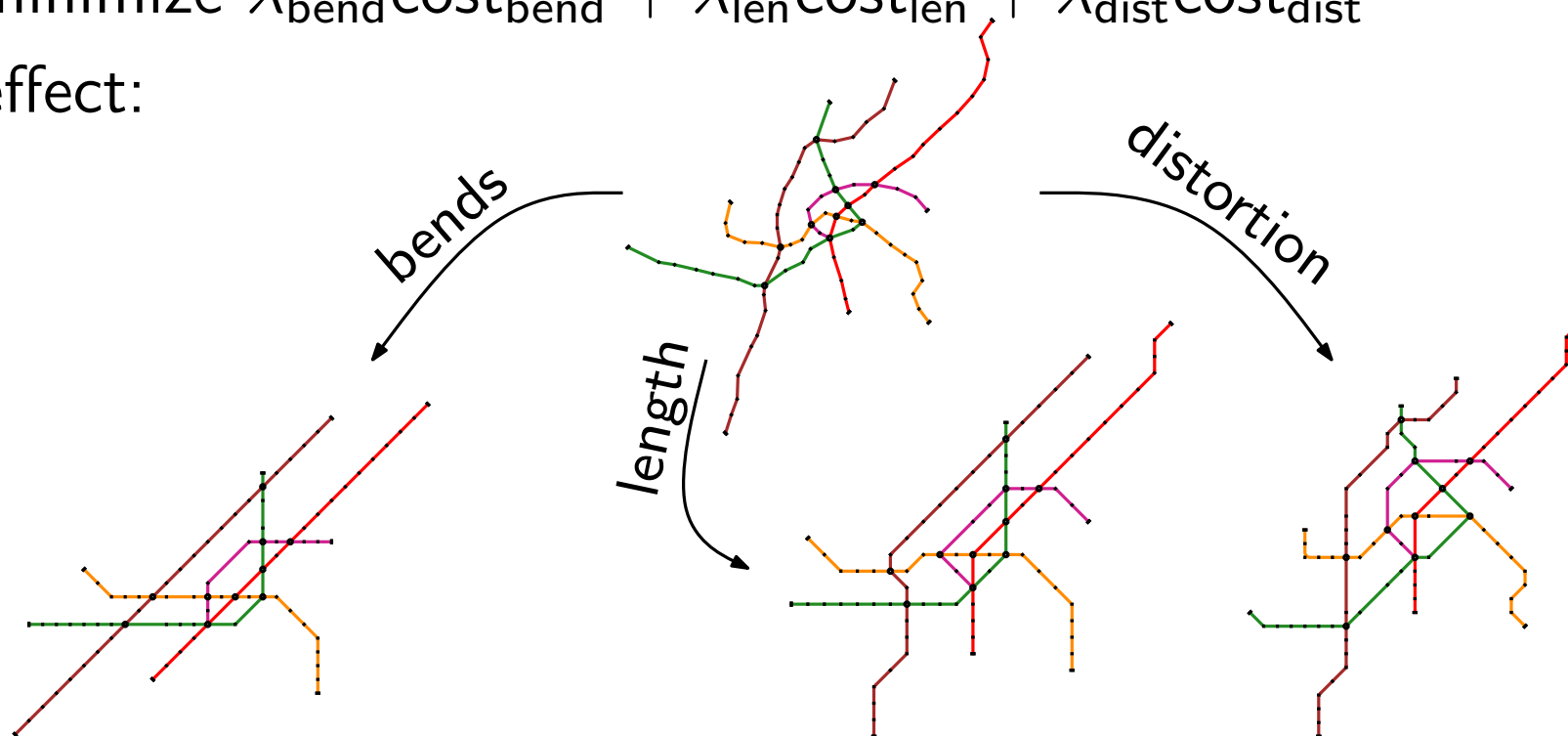
Overview MIP model

Constraints.

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

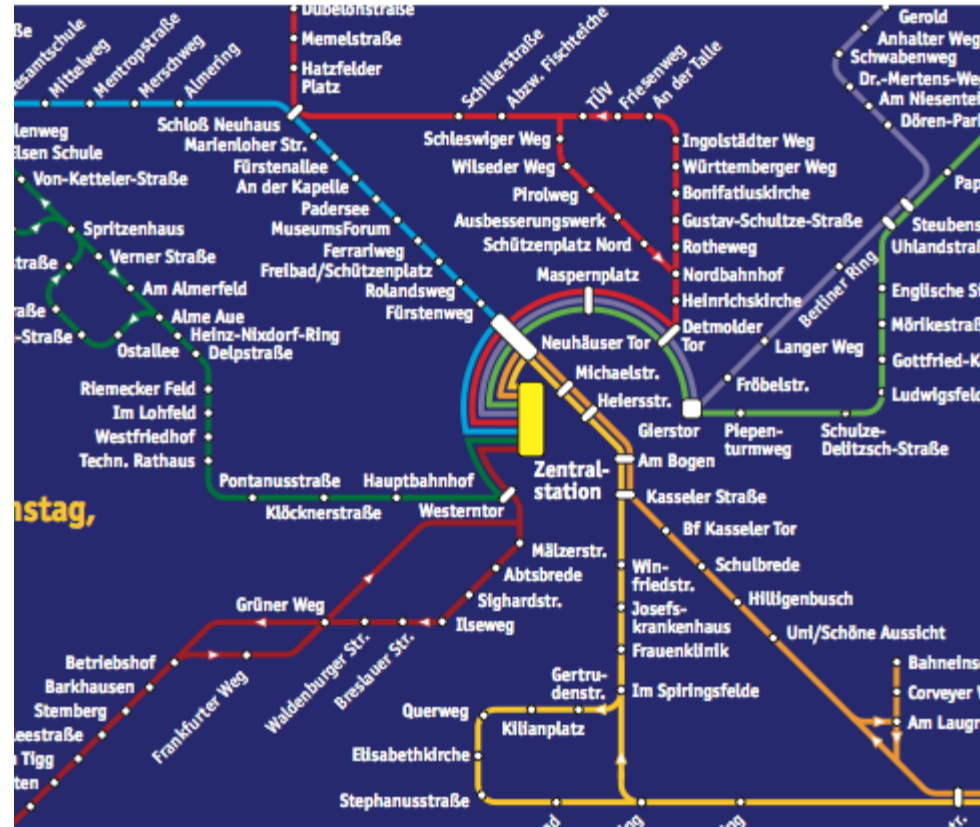
Objective function.

- 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:

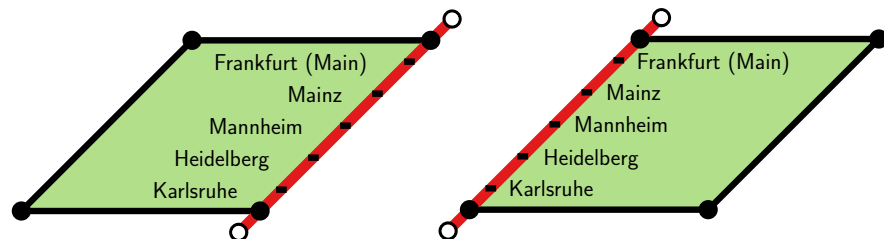


Station Labeling

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

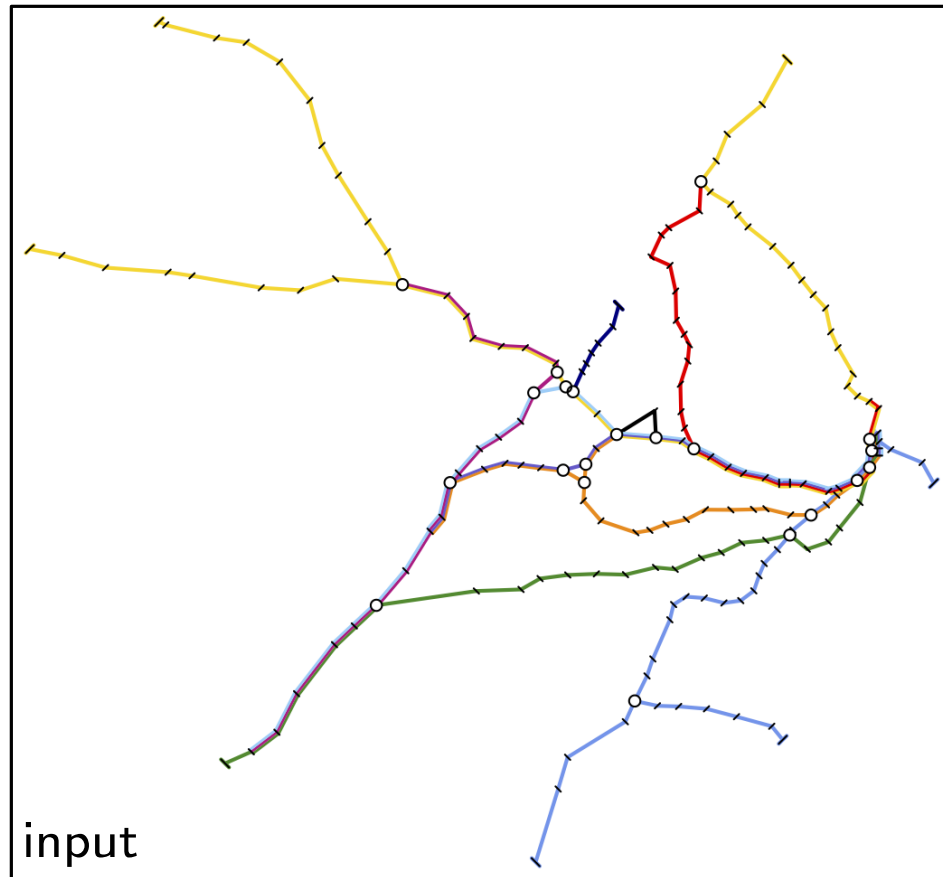


→ combine layout & labeling for optimal results!



- parallelogram as special metro line
- switching sides allowed

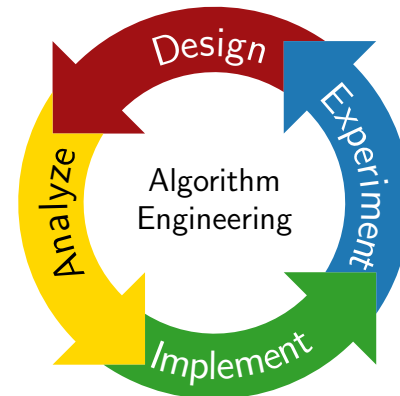
Example: Sydney &



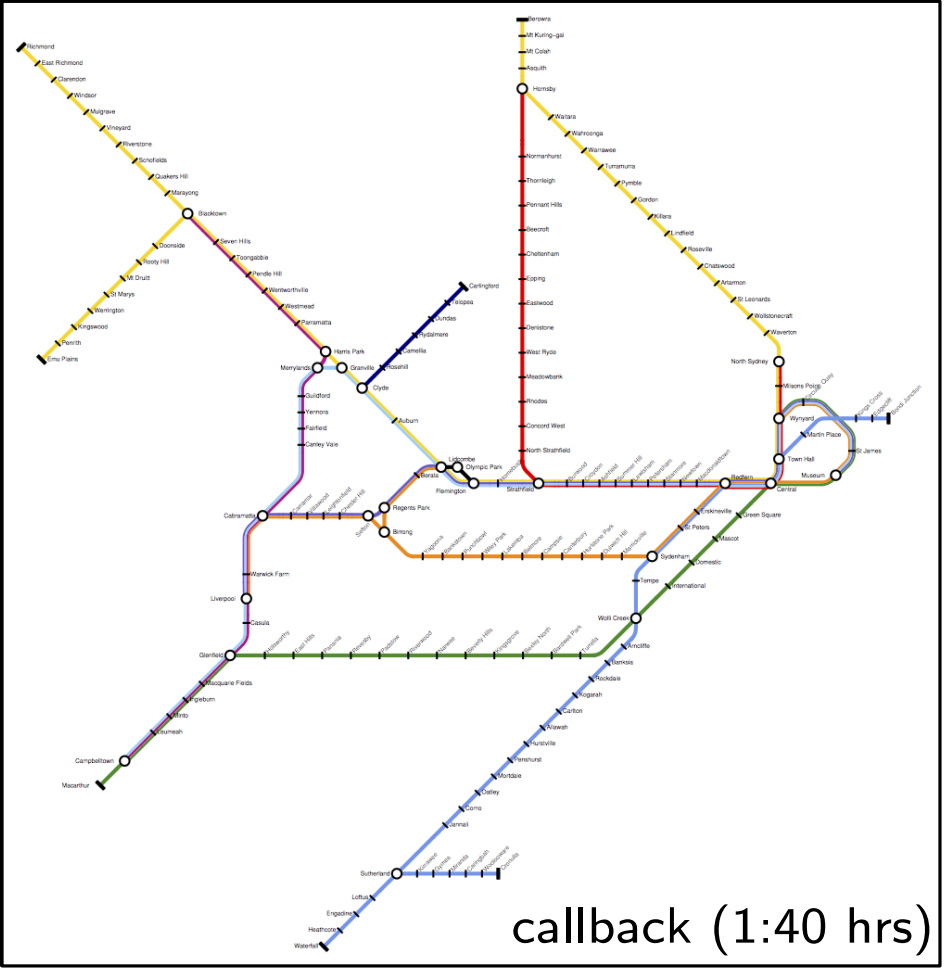
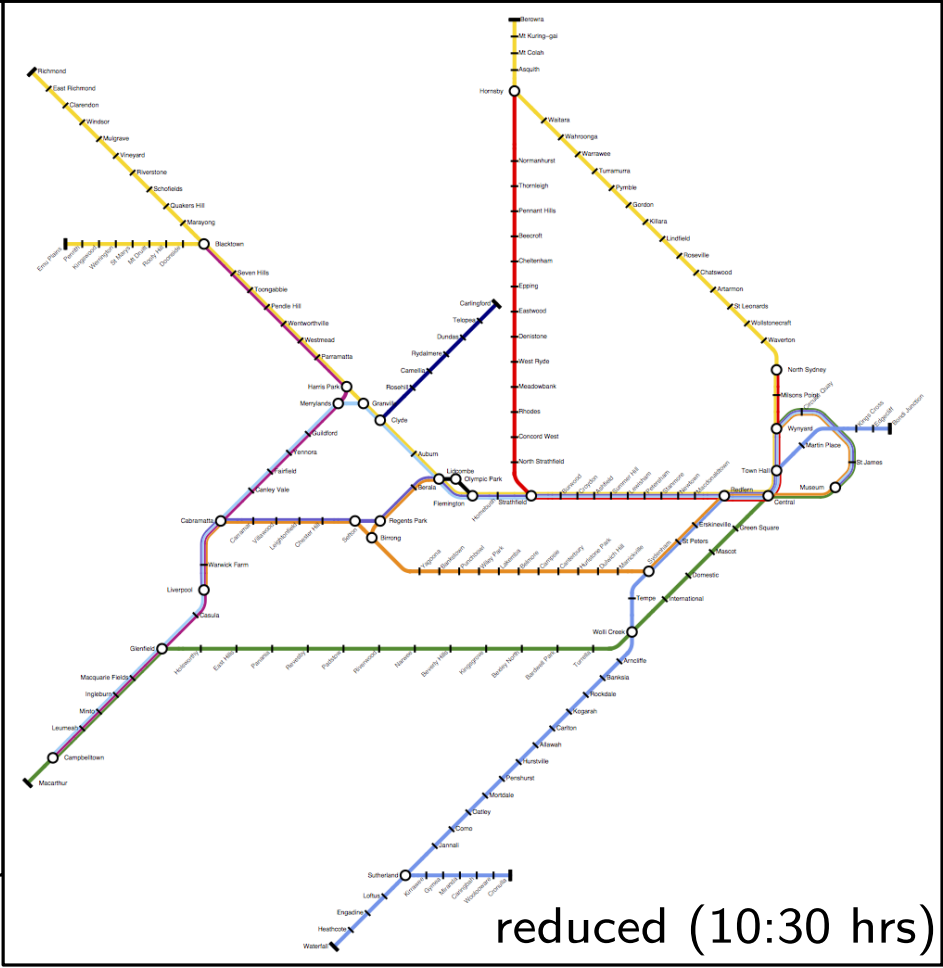
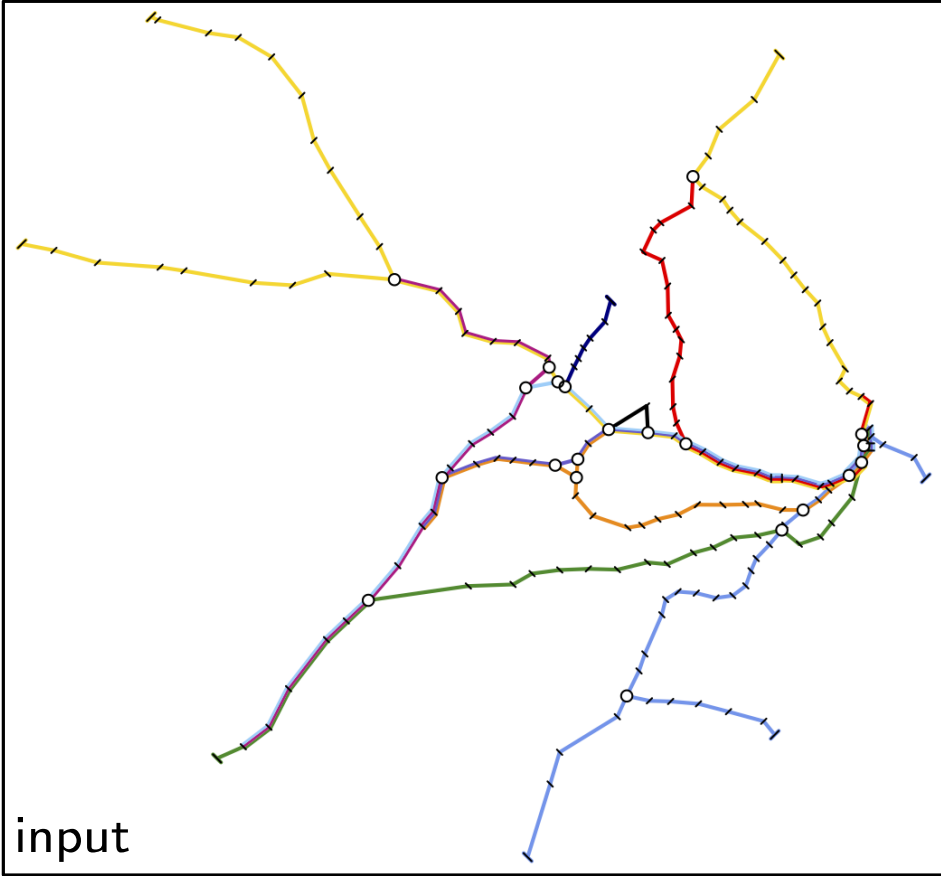
Input	$ V $	$ E $	faces	$ \mathcal{L} $
full	174	183	11	10
reduced	88	97		
labeled	242	270		



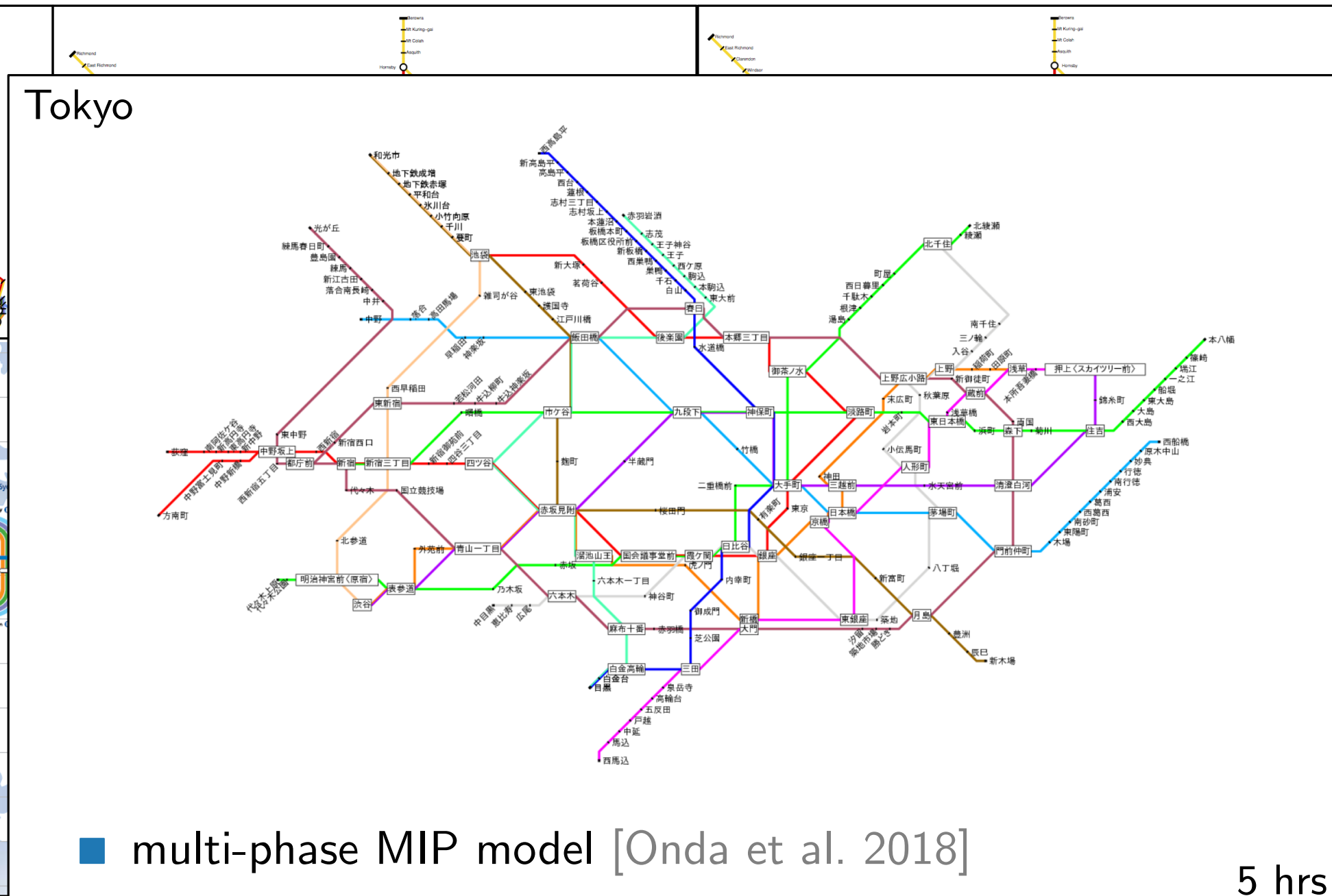
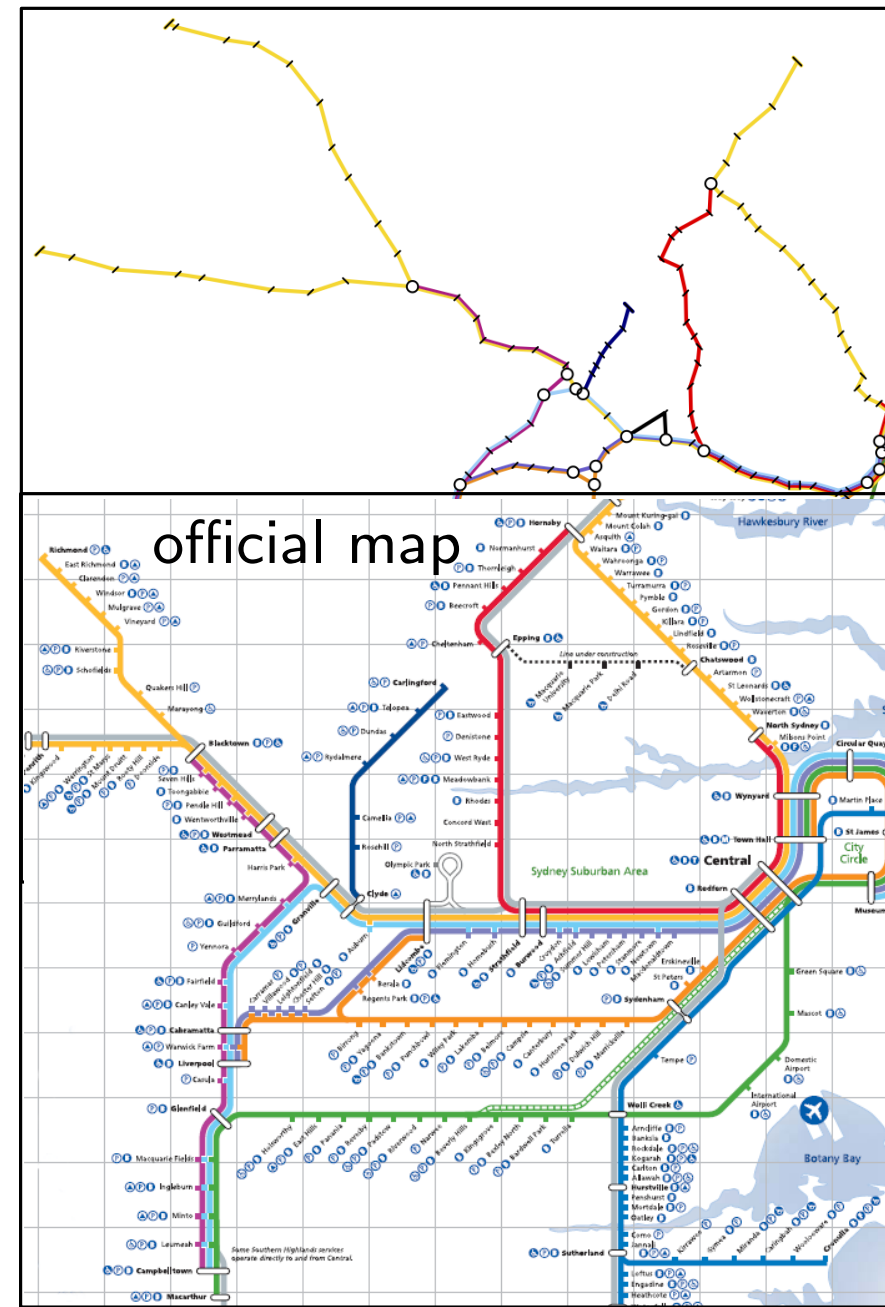
MIP	constraints	variables
full	1,191,406	290,137
callback	21,988	92,681



Example: Sydney &



Example: Sydney & Tokyo



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

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
- 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 Preserving Graph Drawing
- [Gema et al. '11] On d-Regular Schematization of Embedded Paths
- [Hong et al. '06] Automatic visualisation of metro maps
- [Chivers & Rodgers '14] Octilinear Force-Directed Layout with Mental Map Preservation for Schematic Diagrams
- [Fink, Haverkort, Nöllenburg, Roberts, Schuhmann, Wolff '12] Drawing metro maps using Bézier curves
- [Avelar & Müller '00] Generating topologically correct schematic maps
- [Ware et al. '06] Automated production of schematic maps for mobile application
- [Ware & Richards '13] An ant colony system algorithm for automatically 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

Oral Exams

- Focus on **methods** such as
 - (integer) linear programming
 - flow formulations
 - canonical order
 - Schnyder realizer
 - 2-SAT
 - (planar) 3-SAT for NP-hardness proofs
- Be prepared to “run” an algorithm on an example instance.
- You may pick your favorite topic for the first few questions.