Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Visualization of Graphs

Lecture 12:
Octilinear Graph Drawing
Metro Map Layout

Johannes Zink

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
- 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)

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?

$\rightarrow$ 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.

$\rightarrow$ **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 of London voted Design Icon 2006 (2nd after Concorde)

A Bit of History

Berlin 1931
(redrawn by Maxwell Roberts)

A Bit of History

Complexity

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

Proof Sketch.

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

variable gadget (false)

clause gadget (false, false, false)

Remark.

- no efficient exact algorithms to expect
- same problem without diagonals (rectilinear) is efficiently solvable [Tamassia '87] (→ lecture 6)

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 2008]
For a path P of length n and an 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, Gemsa, Nöllenburg, Pajor 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 relationships 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, Nöllenburg, Pajor, Rutter 2011]

input

simplified

optimized by MIP

including length order constraint

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

Force-Based Schematization

Idea. Apply well-known force-based graph drawing approach ($\rightarrow$ lecture 2)

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, Merrick, do Nascimento 2006]

- contract degree-2 vertices into weighted edges **(R3)** (Draw metro lines simple and monotone.)
- define octilinear magnetic field attracting edges **(R2)** (Restrict edge orientations.)
- only apply topology preserving vertex moves **(R1)** (Do not change the network topology.)
- spring lengths model uniform edge lengths **(R7)** (Use uniform edge lengths.)
- vertex repulsion models feature separation **(R8)** (Keep unrelated features apart.)
- station labels are placed in an independent second step

Force-Based Schematization – Bézier

[Fink, Haverkort, Nöllenburg, Roberts, Schuhmann, Wolff 2012]

- 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) (Let lines pass straight through interchanges.)
- (standard) attractive and repulsive forces
  (R7) + **(R8)** (Uniform edge lengths + Keep unrelated features apart.)
- (weak) **(R6)** (M & displacement.)
- forces i urvature
  raw met one as possible.)
- merge) →
- forces i **(R5)** (U stations.)
- apply t y **(R1)** topology.)

Force-Based Schematization – Bézier

[Fink, Haverkort, Nöllenburg, Roberts, Schuhmann, Wolff 2012]

Vienna:

Force-Based Schematization – Discussion

Octilinear.

- guarantees topology **(R1)**
- slower than path-based algorithms, but still fast enough
- no strict enforcement of octilinearity **(R2)**
- quite unbalanced edge lengths **(R7)**
- bends in interchanges **(R4)**
- no distortion restriction **(R6)**

Bézier.

- guarantees topology **(R1)**
- first curvilinear metro map algorithm
- takes almost all design rules into account
- sometimes small angles and distances **(R5)+(R8)**
- works well on small and medium instances, but difficulties with more complex networks

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)**

Zürich 100 1,000 10,000

Local Schematization

Idea.

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

[Ware, Taylor, Anand, Thomas 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, Rodgers, Burkhard, Meier, Smis 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

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**

(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**

(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.

$y \leq 0.9x + 1.5$

$y \geq 1.4x - 1.3$

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 Positions

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**

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

to select exactly one of the three sectors.

Octilinearity and Relative Positions

Predecessor sector.

very large constant (how large, actually?)

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

How does this work?

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

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

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

$$
\left.
\begin{aligned}
y(u) - y(v) &\leq M \\
-y(u) + y(v) &\leq M \\
x(u) - x(v) &\geq -M + \ell
\end{aligned}
\right\} \text{all arbitrary}
$$

Octilinearity and Relative Positions

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
- 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)

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]

Paderborn

→ combine layout & labeling for optimal results!

- parallelogram as special metro line
- switching sides allowed

Example: Sydney

Input	$\lvert V \rvert$	$\lvert E \rvert$	faces	$\lvert \mathcal{L} \rvert$
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 & Tokyo

- multi-phase MIP model [Onda, Moriguchi, Imai 2018]

5 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
- 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

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 vertracke Problem.

VON ALKIENES POLATSCHEK

Das größte Netz: *So etwa würde der Londoner U-Bahn-Plan heute aussehen, wenn man die nach der ursprünglichen Abbildungsmethode fortgeschrieben hätte. Das Bild zeigt den geographisch korrekten Lauf der Linien. Das Londoner U-Bahn-Netz gilt mit 422 Kilometer Länge obschon weit die größte der Welt.*

Schönrechner: *Die geographischen Daten des Londoner Netzes gestern Karlsruher Forscher in ihre experimentelle Software zur automatischen Erzeugung von U-Bahn-Plänen ein. An dem gezeigten Beispiel rechnete die Optimierungsprogramm 15 Minuten lang. Es kann auch Stationen beseitigen, aber da konnte er für das riesige Londoner Netz auch nach zwei Tagen Berechnung noch kein gutes Ergebnis vorzeigen.*

DIE SCHÖNHEIT DES UNTERGRUNDES

Alexander Wolff ist Informatiker und muß ein glücklicher Mensch sein. Denn seit ein paar Monaten sieht er ein Problem gelöst, das er zu seinem Lebensthema gemacht hat – jedenfalls soweit, wie ein Wissenschaftler in einer Lösung etwas anderes erkennen kann als die Einladung, neue Stufen des Unglücktun auszuspähen. Schon Wolffs Diplomarbeit an der Freien Universität Berlin hatte ihn 1992 mit der optimalen Beschriftung von Lageplänen aller Art zusammengebracht; promoviert hat er darüber auch. Im Juni 2000 animierte er dann seine Studenten an der Universität Greifswald in einem Seminar: „Ich füllte die Themenliste mit Problemen, die ich selbst schon immer gern gelöst hätte, aber entweder keinen Erfolg damit hatte oder keine Zeit dafür. Das Thema von Graphendarstellung und Graphenbeschriftung, hier unter dem Begriff U-Bahn-Diagramme formuliert, hat von allen in der Praxis sicherlich die größte Wichtigkeit."

Computererzeugte U-Bahn-Pläne also. Nach seinem Wechsel an die Universität Karlsruhe verschärft Wolff an dieser Frage einen ausgewachsenen Doktoranden und ein paar eigene Blettstiftle, bis er in dem jungen Martin Nöllenburg einen kongenialen Partner fand, der die Frage nicht nur willig schulterte, sondern im Rahmen einer Diplomarbeit auch zu einem fulminanten Ergebnis brachte, wie Wolff lobend erzählt: „Die Arbeit hatte schon so eine Eins gekriegt, aber daß er darin auch noch den mathematischen Beweis schaffte, wie schwierig die Lösung eigentlich ist, das verdiente eine Eins mit Sternchen."

Doch worin besteht diese Schwierigkeit überhaupt? Wir alle sind mit schematischen Netzplänen von U-Bahnen – oder des öffentlichen Nahverkehrs generell – derart vertraut, daß uns das Spezielle daran kaum mehr auffällt. Es ist sogar so: Je besser sie sind, desto weniger wird einem die Qualität bewußt. Denn so ein Diagramm soll eben die mühelose Navigation innerhalb eines Nahverkehrssystems ermöglichen, es also möglichst einfach machen, die passende Station zum eigenen Ziel zu finden, die bunten Linien schnell nachzuverfolgen und alle Umsteigestellen zu sehen, die einem dorthin führen.

Es handelt sich hierbei sicherlich um kein weltmaximierendes Problem. Aber eines zum Verbleiben. Und es ist ein Problem für Handgreiflichkeiten, denn für den neugeschulten Informationsdesigner ist die Detailausführung angedachter Netzschemata ein Debattengegenstand ersten Ranges.

In der Tat werden die großen Pläne bis heute von Hand gezeichnet – natürlich inzwischen am Computer, aber eben gestützt vom Mausklick des Grafikers. Daß eine Software die systematisch mäandernden Linien, ist erst seit kurzem ein Thema. Meist nehmen sich theoretische Informatiker wie Wolff und Nöllenburg der Fragestellung an. Theoretische Informatiker sind häufig nicht so theoretisch, wie es klingt. Das Ruhr liegt nur meist vor ihrer Tür, weil Pläne aller Art sich eben als Bilder von etwas betrachten lassen, was der Mathematiker „Graphen" nennt. Das sind Punkte, die durch Linien, oder genauer gesagt „Kanten", verbunden sind. Für solche Graphen gibt es eine Graphentheorie, also eine Menge mit mathematischen Formeln und Algorithmen, ohne deren Kenntnis man bei konkreten Problemen dann schnell an seine Grenzen stößt.

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

Problemen dieser Sorte kann weiterkommt. Tatsächlich wird aus der Erzeugung eines scheinbar simplen U-Bahn-Schemas ein fetter Klumpen an Optimierungsaufgabe, an der auch ein schneller Computer schier endlos kauen kann.

Der Ausgangspunkt eines jeden U-Bahn-Plans sind je immer die echten Schienen, auf denen Züge fahren, also die geographische Realität. Wie zur Geographie ein Schema wird, und zwar ein möglichst übersichtliches, das ist die Frage, um die es hier geht. Und jeder, der sich ernsthaft mit der Frage befaßt, kennt als erstes die große historische Wunde kennen, nämlich die freudes- und tränenreiche Geschichte des technischen Zeichners Henry C. Beck, dessen revolutionärer Londoner U-Bahn-Plan von 1933 ein System etablierte, das sich die Welt so heftig zu eigen machte, daß wir Heutigen darin gar nichts Besonderes mehr erkennen (siehe „Der Mann, der die Nudeln geradezog").

Neben Alexander Wolff und Martin Nöllenburg haben sich seit ein paar Jahren noch einige Graphentforscher den U-Bahn-Plänen zugewandt. Wie in jeder Subdisziplin der Wissenschaft kennt man sich über die Kontinente hinweg, von Konferenzen und aus der Fachliteratur. Und alle kennen den Beckschen Urgrund. Wenn man eben Schienen, auf denen Züge fahren, also die geographische Realität, nicht für die Feinausführung. Doch denke man an die vielen schiefen Pläne, die in Zukunft ad hoc nach Nutzervorgaben, im Detail, als Ausschnitt, live und online, für Handy und fürs Web zu berechnen wären.

Als Wolff und Nöllenburg ihren historischen Henry Beck zu Ende studiert und lange über einem Stapel moderner U-Bahn-Pläne gebrütet hatten, erdachten sie sieben schlichte Regeln, die aus einem geographischen Plan ein Netzschema machen. „Das Problem so zu modellieren war eigentlich die Hauptarbeit", berichten sie. Von den sieben waren es eineinhalb vier harte Regeln, denen die neue „schöne Zeichnung", wie sie das nennen, ohne Ausnahme folgen muß. Sie muß der geographischen Vorlage topologisch gleichen, das heißt, jeder Knoten muß rundum wieder mit den gleichen Knoten durch je eine Kante verbunden sein. Jede Kante muß außerdem mit einer gewissen festgelegten Mindestlänge abgebildet werden. Jeder Knoten muß darüber hinaus einen festgelegten Mindestabstand von vorbeiführenden Kanten haben. Und schließlich muß die neue Zeichnung „oktilinear" werden. Jede Kante muß von jedem Knoten weg ausnahmslos in eine von acht Richtungen weisen – horizontal, vertikal oder diagonal entweder 45 Grad nach links oder nach rechts gerichtet; das war seinerzeit der große Kniff des Henry Beck.

Die drei weiteren Regeln sind eher weich. Das heißt: Je besser eine neue Zeichnung sie erfüllt, ein desto höheren Schönheitswert bekommt sie in der darauf berechneten Optimierung zuerkannt. Erstens soll jede U-Bahn-Linie darin möglichst wenige Knicke aufweisen, also jede ihrer Kanten möglichst in dieselbe Richtung weisen wie ihre Nachbarn auch. Zweitens sollte die Summe sämtlicher Kantenlängen in der Zeichnung möglichst klein sein (je kompakter das Ganze, desto hübscher). Drittens sollte jede Kante möglichst nahe der Richtung liegen, die sie in der ursprünglichen geographischen Abbildung hat.

Die Bilder, die der Computer nach diesen sieben Regeln am Ende auswirft, ähneln en gros wie en detail oft verblüffend den menschengemachten. Man versteht auf einmal, wovon die ziellosen Skizzenzeichnungen handelten, mit denen Henry Beck über Jahrzehnte hinweg sein Heim pflasterte. Das waren ebenfalls Optimierungsläufe, gekrümmte verbesserte Lösungen. Denn wenn Regeln einmal auf die beschriebene Weise formuliert sind, kann man sie fast ohne Umstände einem der verfügbaren Standardprogramme für eine sogenannte gemischt-ganzzahlige Optimierung vorwerfen. Dieses Programm wirft dann einen optimal schönen Plan aus.

Man muß nur manchmal Geduld haben. Mathematisch gehört die U-Bahn-Aufgabe nämlich zu einer ebenso häufigen wie unbeliebten Klasse von Rechenproblemen, die man „NP-vollständig" nennt. Das ist der Beweis, den Martin Nöllenburg führte. Nach heutiger Erkenntnis gibt es bei Problemen dieser Klasse keinen Trick für schnelle Lösungen. Es hilft nur schiere Rechenkraft, wenhalb unfolgendsche Aufgaben sich in der Praxis einer präzisen oder garantierten Lösung verweigern.

Damit taucht plötzlich ein mathematischer Grund dafür auf, warum Henry Beck in London viele Jahre mit dem Versuch zubrachte, möglichst wenige U-Bahn-Linien als Diagonalen abzubilden. Bekannt war, daß er als ausgebildeter technischer Zeichner sich elektrische Schaltpläne als Vorbild genommen hatte, die ja nur vier Richtungen, sprich horizontale und vertikale Linien kennen – also „Rektilinearität" statt „Oktilinearität". 1981 konnte der amerikanische Doktorand Roberto Tamassia, heute Professor für Informatik an der Brown University, zeigen, daß eine rein rektilinearе Zeichnung „effizient löslich" ist, also leichter zu berechnen und eben nicht NP-schwer. Seit Nöllenburg weiß man wenigstens: Mit man auch Diagonalen sehen, wird das Problem schwer. Henry Beck damals mag das beim Tüfteln gespürt haben. Die beiden Karlsruher Forscher sind offenbar die einzigen, die sich dieser Komplexität stellen. Die Verfahren der anderen Gruppen kommen zwar auch mal schneller zu einem Ergebnis, ihren Ansätzen ist jedoch gerade die entscheidende Oktilinearität gleichgültig oder weniger wichtig – ästhetischer Wert entsteht so höchstens zufällig. (Siehe „Der Grand Prix von Sydney").

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

Die Karlsruher Software läßt um gleichlechtwegs sich nicht tragfähig auf eine Optimierung warten. Bereits während des Laufs lassen sich neue Kriterien einrechnen. Das nutzen Wolff und Nöllenburg für den Trick, die Modell mit einer generellen Überprüfung der Frage zu belasten, ob sich irgendwelche Linien in der Zeichnung überschneiden. Interessante Kandidaten für eine schöne Zeichnung werden auf schneidende Linien geprüft, und eine solche unerlaubte Überlappung wird durch Setzen eines einzelnen zusätzlichen Kriteriums ausgeschlossen. Der Nutzer kann die Kandidaten jederzeit betrachten und wird in den meisten Fällen mit den ersten guten Lösungen zufrieden sein, auch wenn die noch gar nicht rechnerisch optimal sind. Für das vergleichsweise wenig verlassenen Wiener U-Bahn-Netz etwa lieferte der Karlsruher Software auf einem haushaltsüblichen PC schon nach 30 Sekunden eine erstklassige Lösung.

Die Software kann die U-Bahn-Stationen und den Plänen auch beschriften. Das ist eigentlich ein eigener Zweig der Forschung und keine geringe Kunst. Mathematisch gesehen erhält die Vorgabe, möglichst große (aber einheitlich große) Texte ohne Überschneidungen in ein Diagramm zu plazieren, die Komplexität der Aufgabe dramatisch. Doch lösen die Karlsruher die Anforderung durch einen eleganten Kniff. Beschriftungen werden durch eine Art Pseudo-U-Bahn umhüllt, die dann in der Gesamtoptimierung mitläuft. Weil dabei so viel mehr Eingabekriterien vorliegen, dauert die Suche nach einer schönen Zeichnung länger. Im Fall Wien berichtet Nöllenburg von 432 Stunden und ergänzt: „Die ersten vergleichbar guten Lösungen tauchten aber schon nach 24 Stunden auf, und machbare Lösungen schlechter Qualität gab es nach nur dreieinhalb Minuten."

Für die verzweigten Netze von Sydney und London fand sich mit zweitzigigen Rechenläufen noch gar kein schön beschriftbarer Plan. Hier bedarf die Optimierung also noch der Optimierung – oder einer massiv verstärkten Rechenkapazität. Auch im Detail mißfällt die Verschönerung jetzt erst anfangen Unmöglichgemachte anders zu markieren, den Linien eine beliebige Breite geben zu können. Wäre der man damit rechnen! „Wir werden die Thema wohl nicht weiterverfolgen", sagt Martin Nöllenburg. Alexander Wolff, der Anfang des Jahres in Karlsruhe habilitierte, nicht gerade weiter auf eine neue Stelle nach Eindhoven. „Dem Programmcode werden wir wohl freigeben, damit er nicht verschimmelt", sagt er. Vielleicht gibt es ja irgendwo jemandem, der nicht weiter U-Bahn-Pläne von Hand zeichnen will.

Quellen: „M Beck's Underground Map" von Ken Garland (1994); „Underground Maps After Beck" von Ken Garland (1994); „Mapping the Underground" von Mark Ovenden (2006); alle bei Capital Transport Publishing in London erschienen. **Website:** www.urbanrail.net und www.mi-co.com/metro

1 Geografie: *Am kartographisch korrekten Bild der Linien von Sydney CityRail (Stand 2005). Wie das Unternehmen selber bestätigt, gilt das Netz als „einer der verzwicktesten in der Welt" und soll bis 2010 mit hohen Investitionen in neue Schienenverbindungen durchlässiger gemacht werden.*

2 Anziehungskräfte: *Eine internationale Gruppe um Seok-Hee Hong, Spezialistin für Graphen-Darstellung an der Universität Sydney, veröffentlichte 2004 fünf schematische Versionen des Netzes. Ihr bestes Verfahren ist ein „Spring Embedder". Scheint für Schwerkraft rechnen dabei verbundene Knoten zusammen, nicht verbundene stoßen sich ab. Zeitdauer: 7,8 Sekunden.*

3 Gipfelsucher: *Jonathan Stott und Peter Rodgers von der Universität Kent rechnen gern länger als Hong (fast 18 Minuten lang). Sie benutzen ein „Hill-climbing-Verfahren" und optimieren schrittweise anhand mehrer Schönheitskriterien.*

4 Optimierung über alles: *eine Karlsruher Version des Netzes von Sydney. Perfekt oktilinear, und zwangsläufig in diesem Fall mit nur 30 Sekunden Rechenzeit schneller fertig als das Modell aus Kent.*

Der Grand Prix von Sydney

Am Beispiel der australischen Metropole läßt sich ein Vergleichstest der automatisch erzeugten U-Bahn-Pläne durchführen. Alle bekannten Forschergruppen, die Netzpläne optimieren, haben Sydney als einen ihrer Beispiele gerechnet – leider nicht alle im urbanen Jahr und manche in einer gekappten Version, so daß bestpotenzierte Farbe und Länge der Außenlinien vorenanden der abweichen. Trotzdem sind die benutzten Rechenprinzipien am einhaltlichen Beispiel am besten zu illustrieren. Die Kontrahenten des Karlsruher Modells schneiden sich mit ihren Verfahren weniger um die Oktilinearität, also um das beste Raster der acht Richtungen, und liefern daher Schema-Pläne, die um ungewohnt ersichtlichen. Man könnte sie aber auch einfach häßlich nennen.

Menschenwerk: *Das Grafiker von Sydney CityRail ziehen natürlich viel mehr Feinarbeit in die Netzschema. Das generelle Layout des Plans ist von einer automatischen Variante aber nicht weit entfernt, sie könnte also durchaus Grundlage für weitere manuelle Geistfeil werden.*

Der Mann, der die Nudeln geradezog

Mit Henry Beck begann die Ära der aufgeräumten Diagramme

Das Original: *Henry Becks erster Entwurf für die Londoner U-Bahn, veröffentlicht im Januar 1933 – heute ein Meilenstein der Design-Geschichte. Schematische Darstellungen einzelner Strecken gab es zwar damals zum Beispiel schon in der U-Bahn-Hagen, aber auf ein Gesamtschema war vorher nie jemand gekommen.*

Deutsche Revolution: *Außerhalb Englands findet man ein Diagramm, das Beck Ideen komplett vorwegnimmt. Es ist ein Berliner Netzschema von 1931, perfekt oktilinear aufgebaut, vom S-Bahn-Ring abgesehen. Der entscheidende Unterschied zu London: Dort wurde eine lokbinde Tradition und von vorherrscher Verhältnis geschaffen. In Berlin vorgeß man die Vorzüge des Schemas bald wieder und kehrte lange zur geographischen Karte zurück. Man hatte wohl mit anderen Commerzen zu tun.*

Verwirrung in Manhattan: *Die New Yorker U-Bahn gilt nach London als zweitgroßte, so hat sogar mehr Stationen. 1972 führte in ihnen neuen „Subway Guide" nach dem Muster von Beck ein, gestaltet vom Star-Designer Massimo Vignelli. Der Plan wurde sonder abgeschafft. Die New Yorker fanden die Abweichungen von der Geographie zu irriterend.*

Wer das U-Bahn-Wesen generell, so begannen nach seine Dienstjungsprobleme in London. Man schrieb 1908, da waren die damals privaten Londoner U-Bahnen in Not. Ihre sündteure Maulwurfarbeit in der Londoner Innenstadt war abgeschlossen, man mußten neue Passagiere anlocken. Deshalb gaben die Konkurrenten erstmals einen gemeinsamen Plan heraus. Diese geographische Karte zeigte den Verlauf der Linien unter Stadtteilen und Straßen.

Bald wuchsen die Linien aber weiter hinaus in die Vororte. Auf dem fortgeschrittenen Plan ewiggt es sich das wichtige Innenstadt-Netz auf so knapper Fläche zusammen, daß Nützliches kaum noch zu erkennen war. Die Zeichner hielten alle Einzelheiten weg, sie fingen sogar widerwillig an, der Karte einen gleitenden Mißstab zu geben, im Zentrum größer als außen. Trotzdem erinnerte der Plan schließlich noch an verwickelte Spaghetti berechnungsweise ihre noch dünnere Variante „Vermicelli", wie Henry Beck sie nannte.

Henry C. Beck („Harry" unter Freunden) wurde der Mann, der die Nudeln geradezog. Geboren Anfang des Jahrhunderts, fing er 1925 als junger technischer Zeichner bei der U-Bahn zu arbeiten an. Die Jahre waren hart, er bekam nur einen Zeitarbeitsvertrag; mal beschäftigt, mal gekündigt, die Jüngsten traf es immer zuerst.

In seiner freien Zeit, irgendwann 1931, traf den Henry Beck von dem Vermicelli-Plan ein Geduldsanfälle. Warum nicht die Geographie beiseite lassen? Die scheute den Untergrundpendler ja gar nicht. Alle Linien geradezuhen und aufrichten, sie horizontal und vertikal verteilen wie auf einem elektrischen Schaltbild, vielleicht auch Diagonalen, die Stationen gleichmäßig auf die Strecken verteilen – das war es doch! Ein übersichtliches Diagramm statt einer Karte.

Im Sommer 1933 zeigte er der Werbeabteilung seines Unternehmens den Entwurf. Kein Interesse, hieß es. Zu revolutionär. Henry Beck steckte das Blatt in die Schublade, ging es ein Jahr später wieder hervor und wagte einen neuen Anlauf. Überraschung: Nun wollte man es versuchen!

Der kleine Fähplan wurde im Januar 1935 gedruckt und vom Publikum ins Herz geschlossen. Die ersten 500 Stück waren rasch vergriffen, im Februar mußten schon 500 neue nachgedruckt werden. Im Sommer 1935 stellte die gesamte geradezugerichteten U-Bahn ihre Plakate nach dem Schema Beck um.

1994 verfaßte der einflußreiche britische Designer Ken Garland das Standardwerk über Becks Arbeiten. Er wollte der Geschäftsgleitung seines Freundes ein Denkmal setzen und stellte fest: „Es wird sonst nur wenige Einzelarbeiten geben, die so bahnbrechend und dauerhaft maßgebend waren." Es stimmt schon: Wie leicht zu erkennen, haben Nah-

London 1937, 1940, 1954: *Henry Beck war ein Tüftler der Geduld und der Pedanterie. Fast 30 Jahre lang schrieb er sein Prinzip in endloser Variation die Detailgestaltung fort, aus eigenem Antrieb oder irgendwann dann unter Auftraggeber. Hier nur drei Beispiele von vielen.*

verkehrsnetze in aller Welt sich die Idee angeeignet.

Und Beck? Für Entwurf und Arbeit, die Monate der Mühen, bekam er 20,50 Pfund als Honorar, was man sich in heutiger Kaufkraft als 1000 Euro vorstellen darf. Doch Henry Beck soll doppelt in der Falle. Wegen seines perfekten Arbeitsvertrages gestand man ihm die Ideen zu, hielt ihn aber weiterhin wie einen Domestiken. Schlimmer noch: Die Idee selbst nahm Beck in Heft, Jahrzehnte verbrachte er mit der Fortschreibung des Netzplans in mäßiser Detailverzweiseiheit. Mindestens einmal zeichnete er aus eigenem Antrieb die ganze Druckvorlage neu, nur um die Biegungen der Linien etwas enger zu fassen. Dazu kamen die Ideen der wechselnden Vertreter der U-Bahn. Mal mußte Beck die Umsteigekosten in Form von Kreisen zeigen statt als Rasten, dann mußte er Stationsnamen in vergrößerte Rasten schreiben oder Punkte in die Kreise setzen, wo auch ein Fernbahnhof lag. Mal wurden die zentralen Linien, dann alle Linien fetter und ihre Namen in der Linie verzweigt, und schließlich alles wieder retour, weil Beck den Ideen dann gutwilelich Ausführung ihren Irrsinn nachwies.

Ken Garland vermutet, daß sich Beck dem Irrsinn, um Beck zu hart by fest in Händen behalten zu können. Die Neugierigsamkeit hab, obwohl manche in der Werbeabteilung Beck nur als schwierigen und verdrießlichen Menschen sahen. Im April 1960 diptierte ihn dieses Chef Harold F. Hutchinson und veröffentlichte mit Fanfarenstoß einen komplett neu gestalteten Plan, der er selber angefertigt hatte. Es war bei nüherer Betrachtung eine amusanterhafte Karikatur der Beckschen Ideen. Noch zwei quälenden Jahren wurde die neue Schema nach Beckschen Prinzipien gebeift und bis heute fortgeschrieben.

Beck selber kam trotz verzweifelter Bittbriefe nie mehr zum Zug. Er starb 1974. Heute gilt er als großer Meister, gekürt mit einer eigenen Ausstellung im London Transport Museum, Anklicken von Erzwärtten am seiner Hand und zuletzt, in einer Publikums-Show der BBC, der Wahl seines revolutionären U-Bahn-Plans zur zweitwichtigsten Design-Ikone überhaupt – nach der Concorde. *kp*

Least-Squares Schematization

[Wang & Chi 2011]

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 a focus route, but 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
- take thick edge bundles and large vertices into account

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, Gemsa, Nöllenburg, Pajor '10] Path Schematization for Route Sketches
- [Brandes & Pampel '09] On the Hardness of Orthogonal-Order Preserv-ing Graph Drawing
- [Gemsa, Nöllenburg, Pajor, Rutter '11] On d-Regular Schematization of Embedded Paths
- [Hong, Merrick, do Nascimento '06] Automatic visualisation 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, Taylor, Anand, Thomas '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, Rodgers, Burkhard, Meier, Smis '11] Automatic 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