Osnabrück, 28. Juni 2016

Computergrafik SS 2016

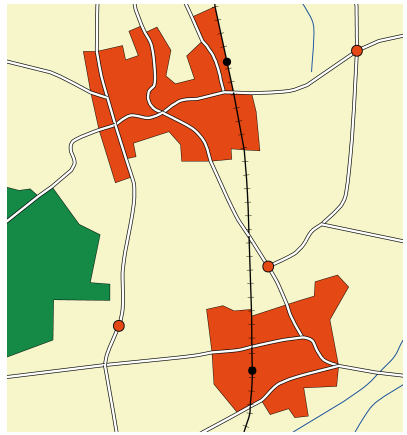
Kapitel 26: Geovisualisierung

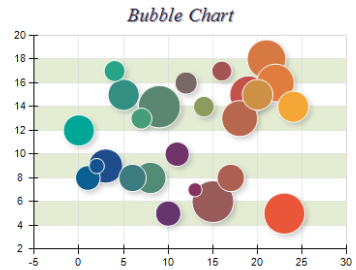
Prof. Dr.-Ing. Jan-Henrik Haunert
AG Geoinformatik
Institut für Informatik
Universität Osnabrück

Computergrafik vs. Informationsvisualisierung

- schnelles Rendern realistischer 3D-Modelle

- automatische Erzeugung abstrahierter grafischer Modelle (Diagramme, Graphen, Karten)

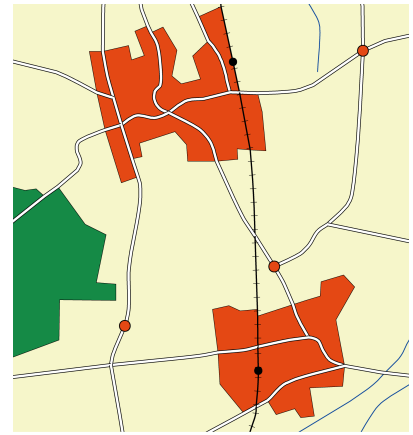


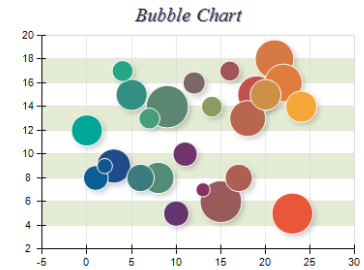


Computergrafik vs. Informationsvisualisierung

- schnelles Rendern realistischer 3D-Modelle

- automatische Erzeugung abstrahierter grafischer Modelle (Diagramme, Graphen, Karten)



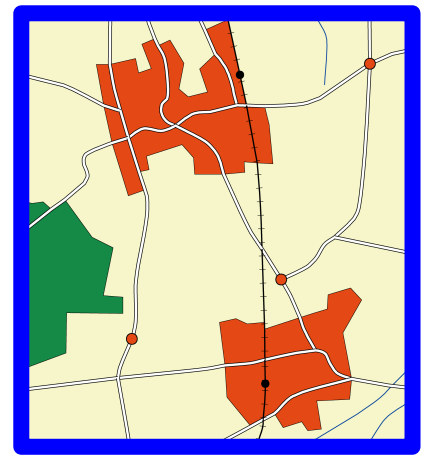


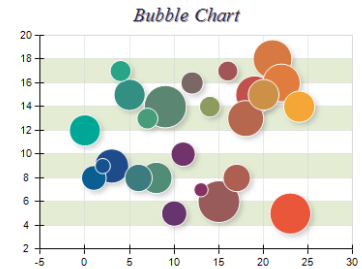
- gemeinsame Konferenzen und Zeitschriften, z.B. *IEEE Transactions on Visualization and Computer Graphics*

Computergrafik vs. Informationsvisualisierung

- schnelles Rendern realistischer 3D-Modelle

- automatische Erzeugung abstrahierter grafischer Modelle (Diagramme, Graphen, Karten)





- gemeinsame Konferenzen und Zeitschriften, z.B. *IEEE Transactions on Visualization and Computer Graphics*
- Geovisualisierung: Spezialisierung von CG und InfoVis auf Geo-Themen, insbes. mit abstrahierten Modellen (Karten).

Was ist eine Karte?

Definition der International Cartographic Association (ICA)
beschlossen auf der International Cartographic Conference 1995 (ICC'95)

„A map is a symbolised image of geographical reality,
representing selected features or characteristics,
resulting from the creative effort
of its author's execution of choices,
and is designed for use
when spatial relationships are of primary relevance."

Was ist eine Karte?

Definition der International Cartographic Association (ICA)
beschlossen auf der International Cartographic Conference 1995 (ICC'95)

„A map is a symbolised image of geographical reality,
representing selected features or characteristics,
resulting from the creative effort
of its author's execution of choices,
and is designed for use
when spatial relationships are of primary relevance."

(Diercke Weltatlas, 2007)

- Ein Foto ist keine Karte!
- Entscheidend sind graphische Abstraktion und Auswahl.

Was ist eine Karte?

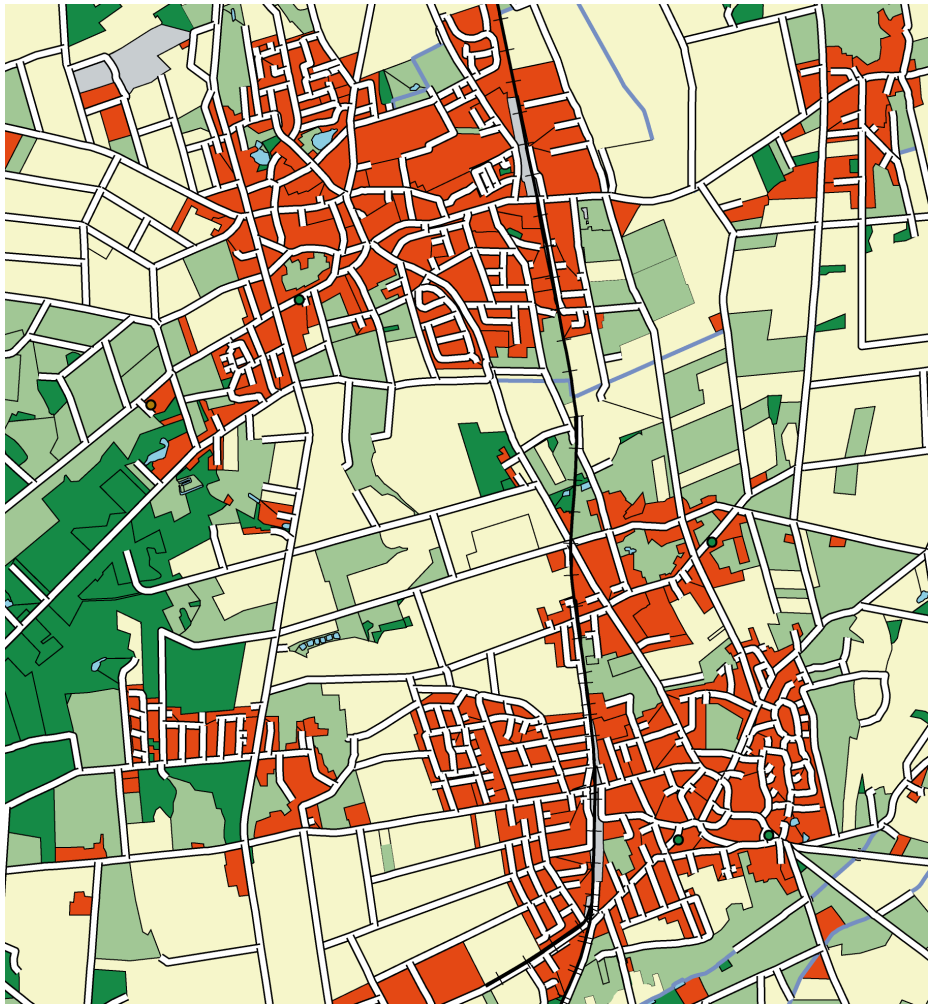
Definition der International Cartographic Association (ICA)
beschlossen auf der International Cartographic Conference 1995 (ICC'95)

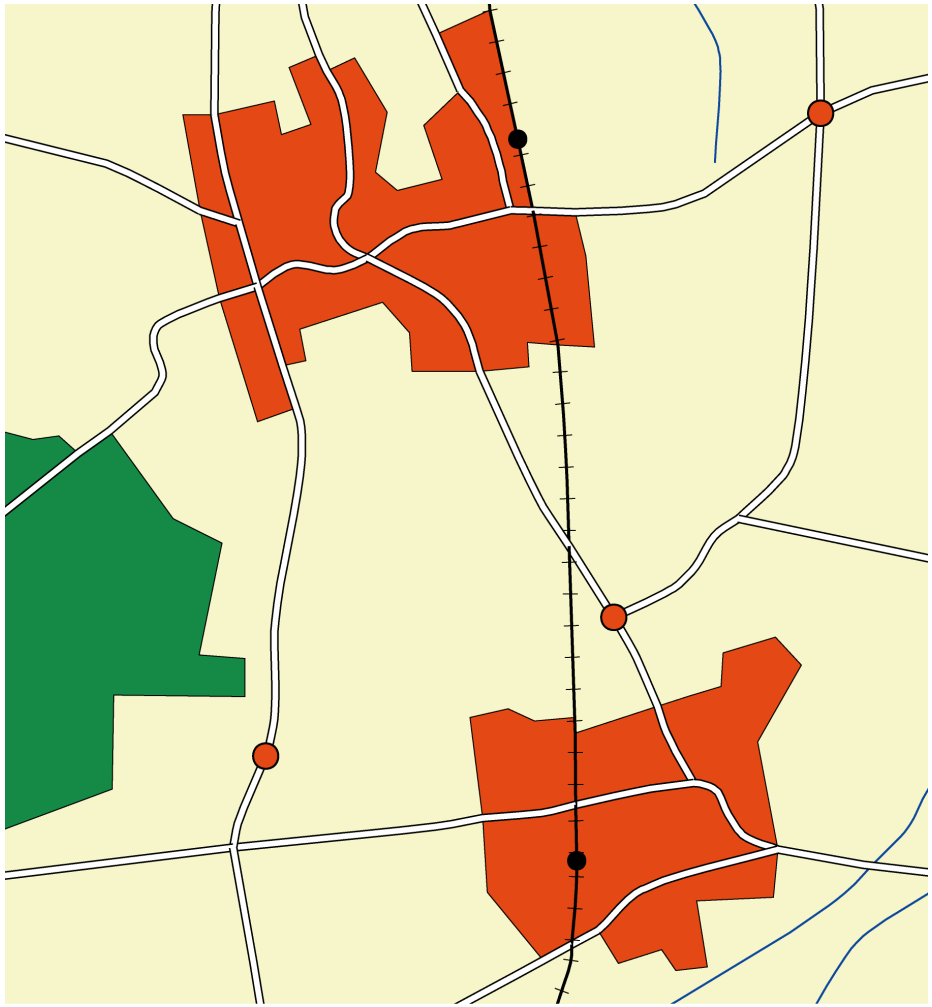
*„A map is a symbolised image of geographical reality,
representing selected features or characteristics,
resulting from the creative effort
of its author's execution of choices,
and is designed for use
when spatial relationships are of primary relevance.“*

(Diercke Weltatlas, 2007)

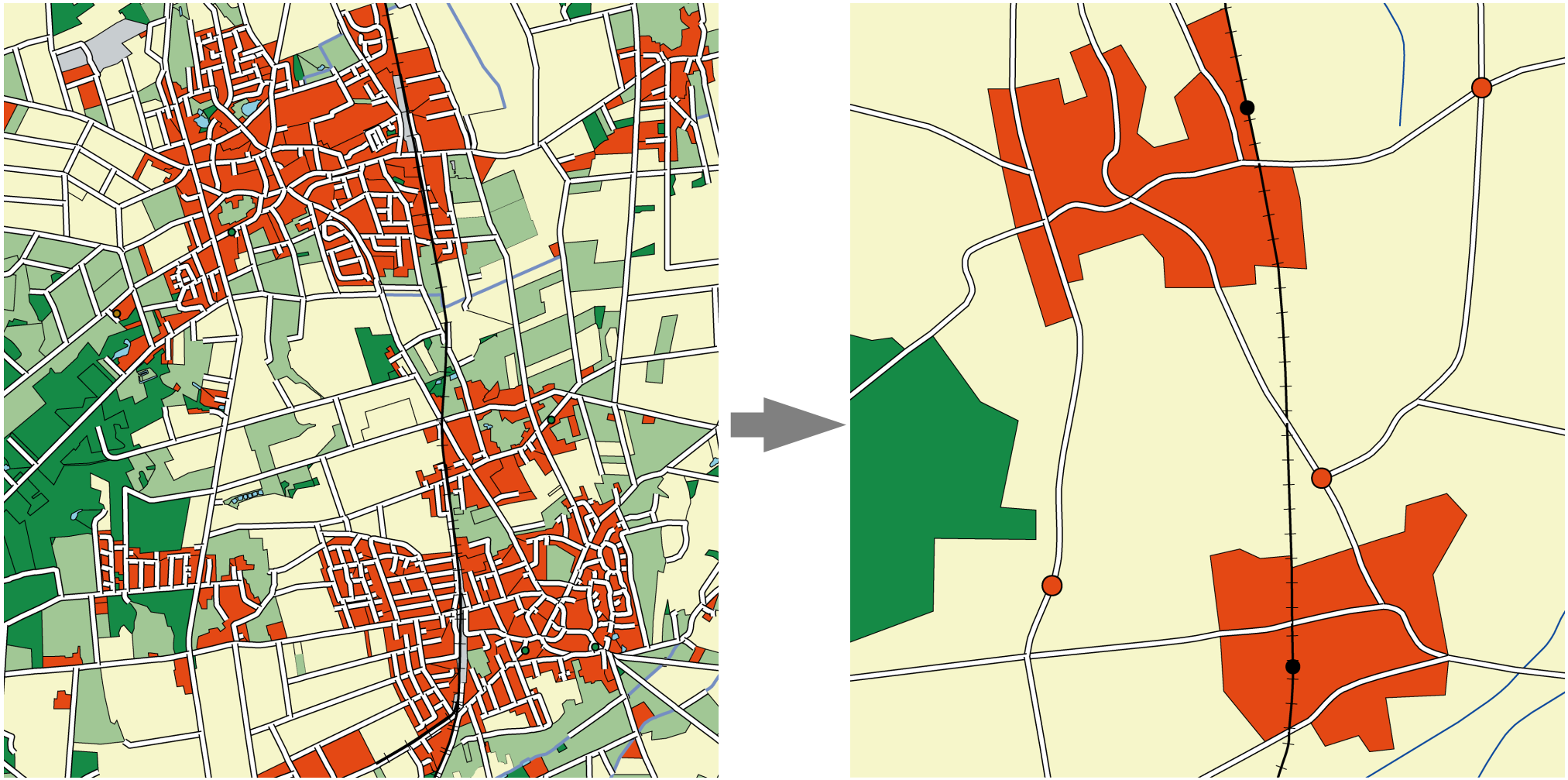
- Ein Foto ist keine Karte!
- Entscheidend sind graphische Abstraktion und Auswahl.

klassische Probleme der Kartographie



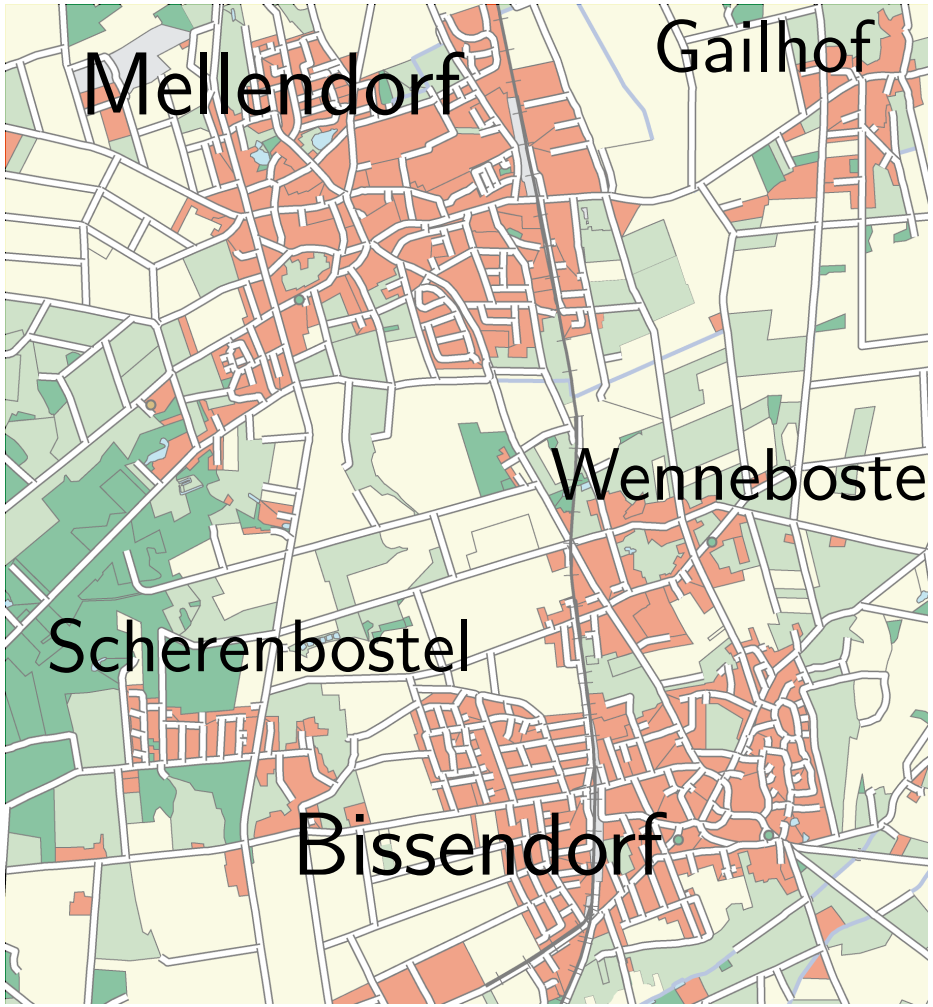


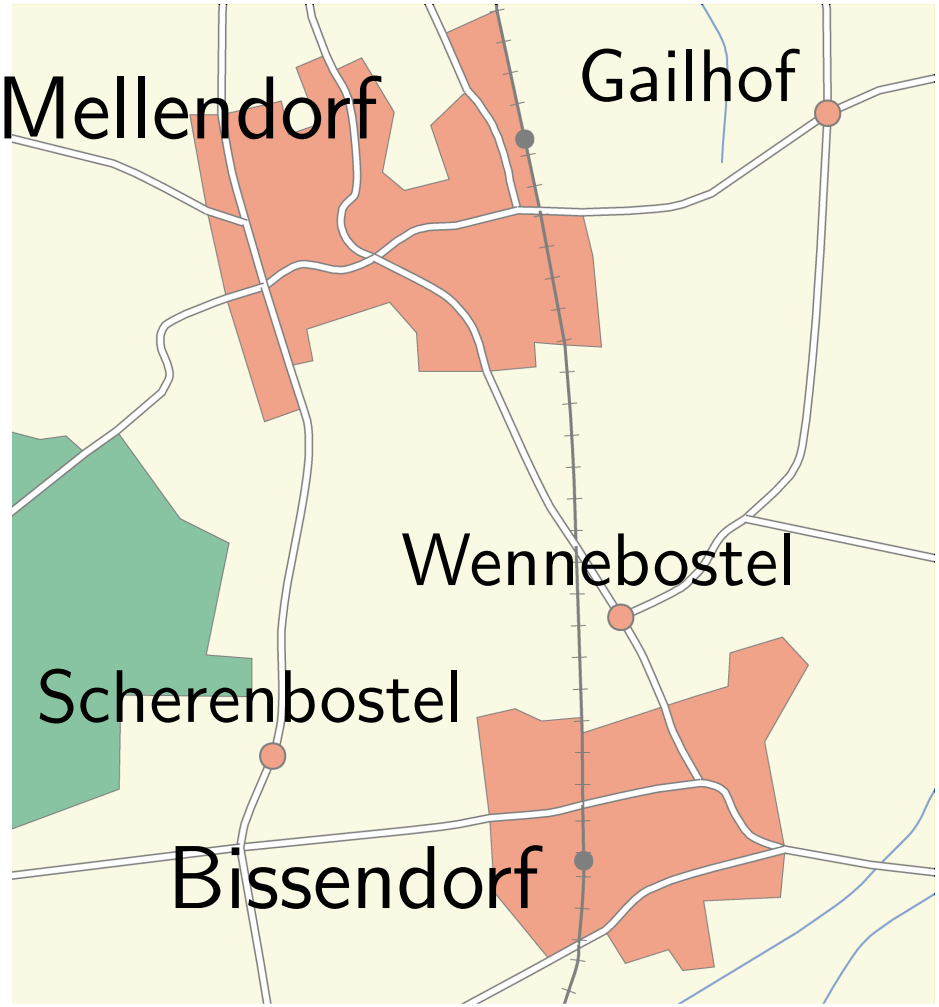
klassische Probleme der Kartographie



1. Generalisierung

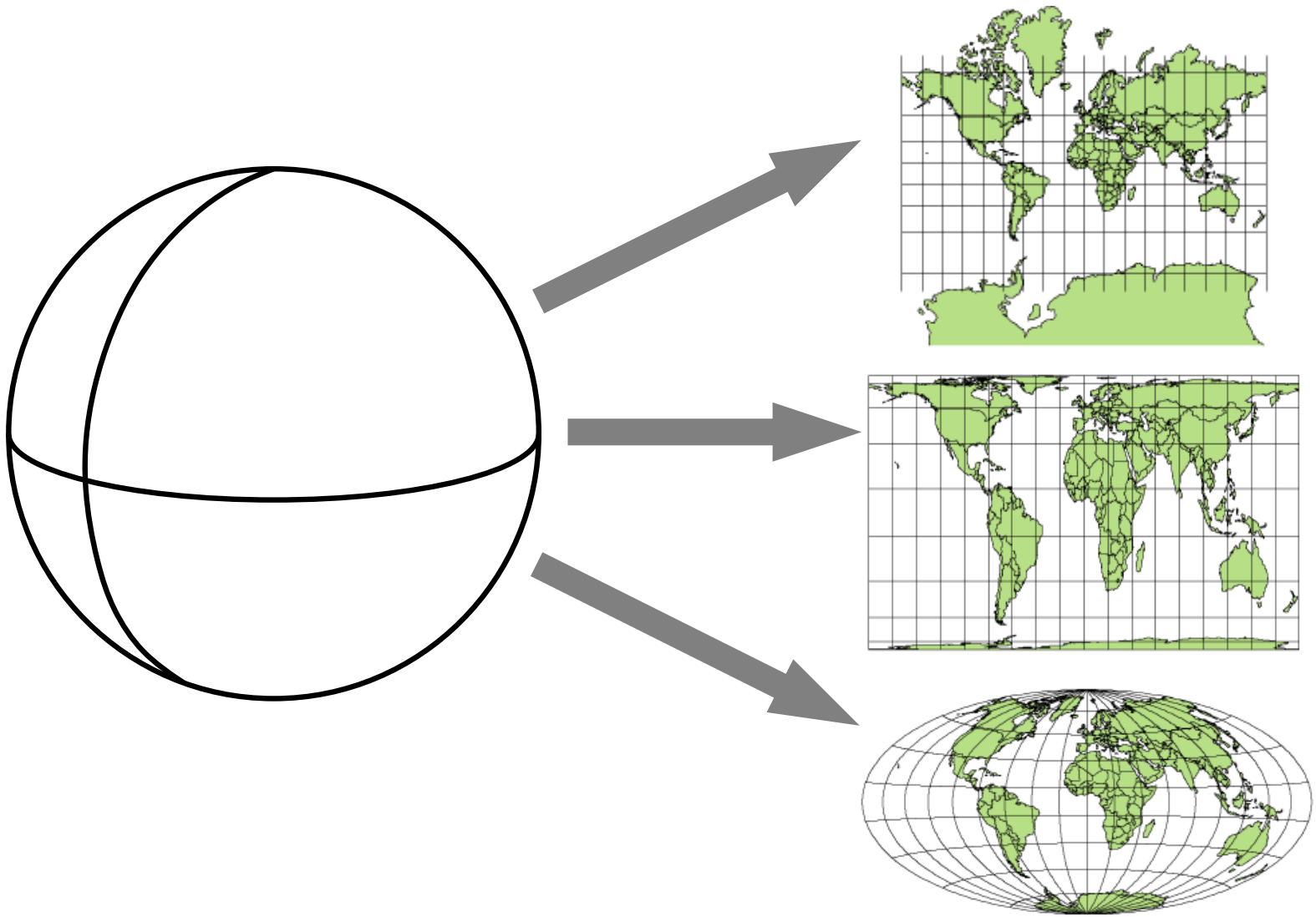
klassische Probleme der Kartographie





2. Beschriftung

klassische Probleme der Kartographie



3. Herstellung des Raumbezugs

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.

Aufgabe II: *Lösung*

Entwirf einen Algorithmus, der eine Lösung berechnet,

- die alle Restriktionen erfüllt.

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen und Ziele.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.
- Ein Ziel drückt die Qualität einer Lösung als Zahl aus.

Aufgabe II: *Lösung*

Entwirf einen Algorithmus, der eine Lösung berechnet,

- die alle Restriktionen erfüllt.

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen und Ziele.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.
- Ein Ziel drückt die Qualität einer Lösung als Zahl aus.
- Mehrere Ziele werden i.d.R. durch eine (gewichtete) Summe zu einem Gesamtziel zusammengefasst.

$$Z_{\text{ges}} = w_1 \cdot Z_1 + w_2 \cdot Z_2 + \ldots + w_k \cdot Z_k$$

Aufgabe II: *Lösung*

Entwirf einen Algorithmus, der eine Lösung berechnet,

- die alle Restriktionen erfüllt.

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen und Ziele.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.
- Ein Ziel drückt die Qualität einer Lösung als Zahl aus.
- Mehrere Ziele werden i.d.R. durch eine (gewichtete) Summe zu einem Gesamtziel zusammengefasst.

$$Z_{\text{ges}} = w_1 \cdot Z_1 + w_2 \cdot Z_2 + \ldots + w_k \cdot Z_k$$

Aufgabe II: *Lösung*

Entwirf einen Algorithmus, der eine Lösung berechnet,

- die alle Restriktionen erfüllt
- und bzgl. des Gesamtziels am besten ist.

Automation

Aufgabe I: *Modellierung*

Formalisiere Restriktionen und Ziele.

- Restriktionen müssen von der Ausgabe *zwingend* erfüllt werden.
- Ein Ziel drückt die Qualität einer Lösung als Zahl aus.
- Mehrere Ziele werden i.d.R. durch eine (gewichtete) Summe zu einem Gesamtziel zusammengefasst.

$$Z_{\text{ges}} = w_1 \cdot Z_1 + w_2 \cdot Z_2 + \ldots + w_k \cdot Z_k$$

Aufgabe II: *Lösung*

Entwirf einen Algorithmus, der eine Lösung berechnet,

- die alle Restriktionen erfüllt
- und bzgl. des Gesamtziels am besten ist.

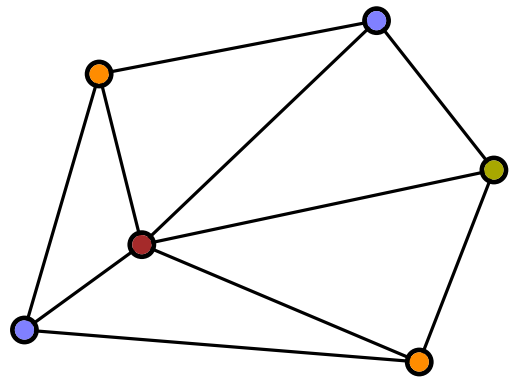
= **Optimierung**

Automation

- Für viele Probleme der Kartographie bieten sich *graphentheoretische Modelle und Algorithmen* an.

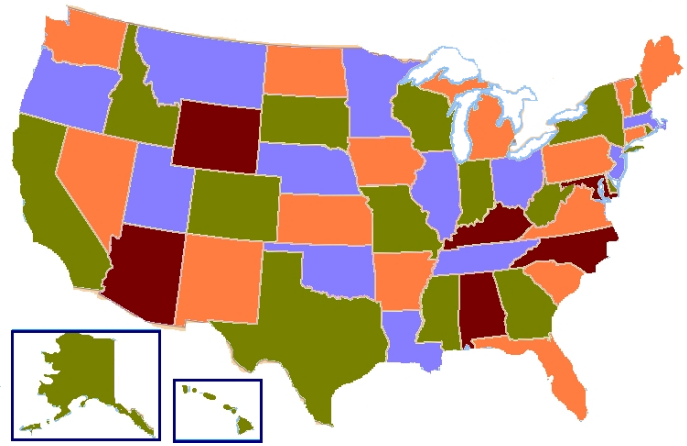
Automation

- Für viele Probleme der Kartographie bieten sich *graphentheoretische Modelle und Algorithmen* an.
- z.B. *Vier-Farben-Theorem:*
  Vier Farben genügen, um die Ecken eines planaren Graphen so zu färben, dass keine zwei benachbarten Ecken dieselbe Farbe haben.



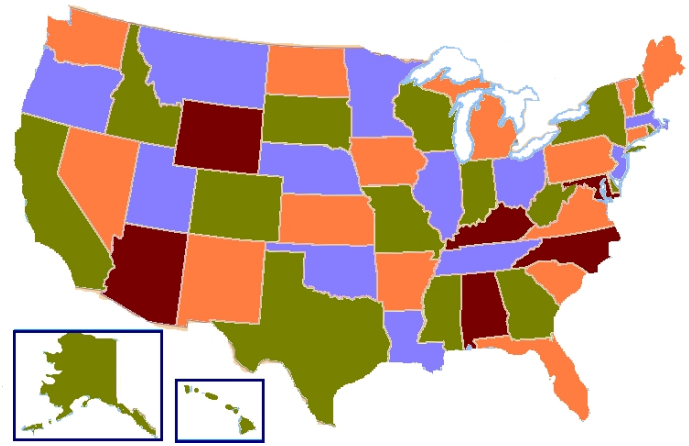
Automation

- Für viele Probleme der Kartographie bieten sich *graphentheoretische Modelle und Algorithmen* an.
- z.B. *Vier-Farben-Theorem:*
  Vier Farben genügen, um die Ecken eines planaren Graphen so zu färben, dass keine zwei benachbarten Ecken dieselbe Farbe haben.



Automation

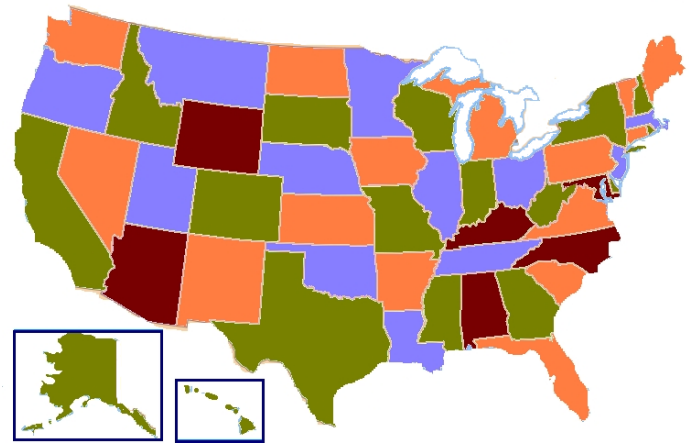
- Für viele Probleme der Kartographie bieten sich *graphentheoretische Modelle und Algorithmen* an.
- z.B. *Vier-Farben-Theorem:*
  Vier Farben genügen, um die Ecken eines planaren Graphen so zu färben, dass keine zwei benachbarten Ecken dieselbe Farbe haben.



- in dieser Vorlesung:
  - unabhängige Mengen
  - Matchings
  - kürzeste Wege
  - Flüsse

Automation

- Für viele Probleme der Kartographie bieten sich *graphentheoretische Modelle und Algorithmen* an.
- z.B. *Vier-Farben-Theorem:*
  Vier Farben genügen, um die Ecken eines planaren Graphen so zu färben, dass keine zwei benachbarten Ecken dieselbe Farbe haben.



- in dieser Vorlesung:
  - unabhängige Mengen
  - Matchings
  - kürzeste Wege
  - Flüsse

Gliederung:

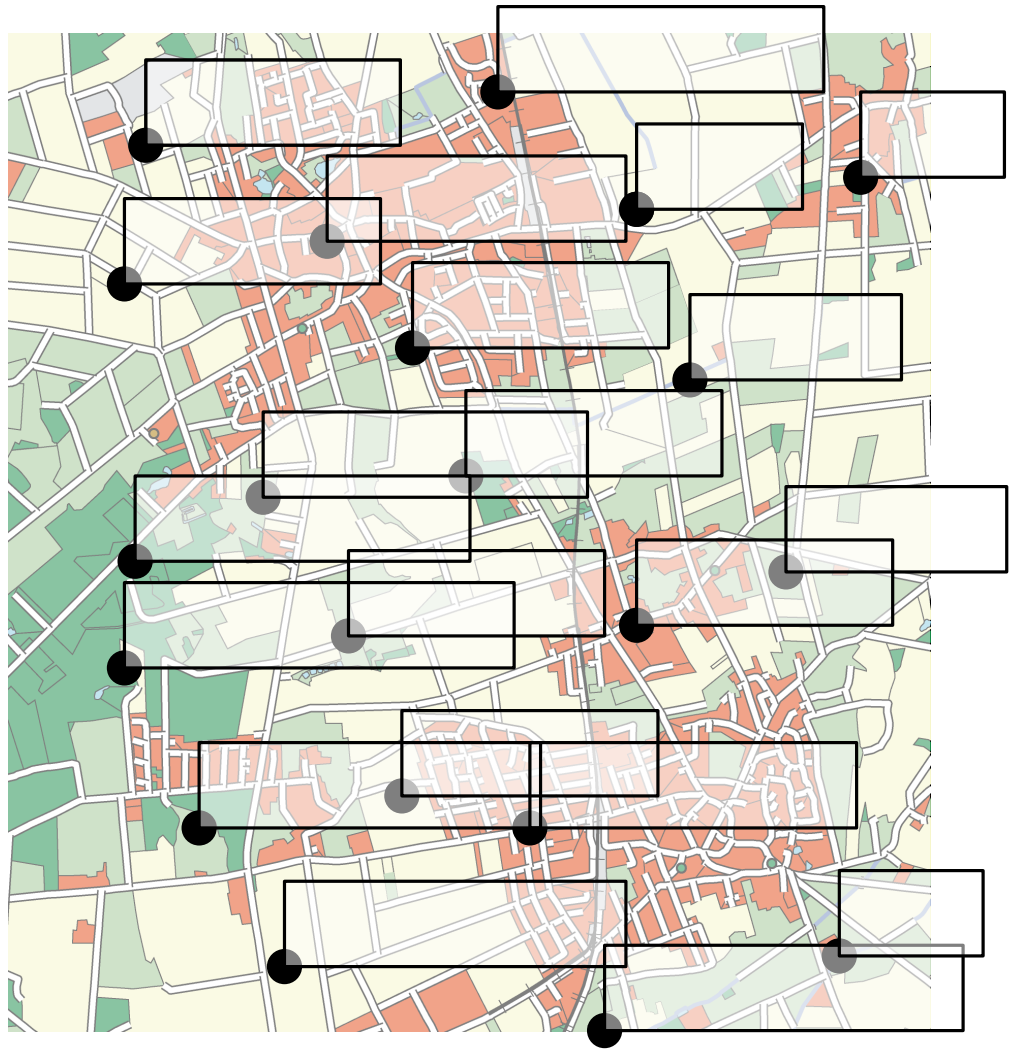
- Beschriftung
- Erzeugung von Grid Maps
- Linienvereinfachung
- Selektion
- Fazit

Beschriftung

Allgemeiner Ansatz

See, e.g., Shieber et al. (1995):

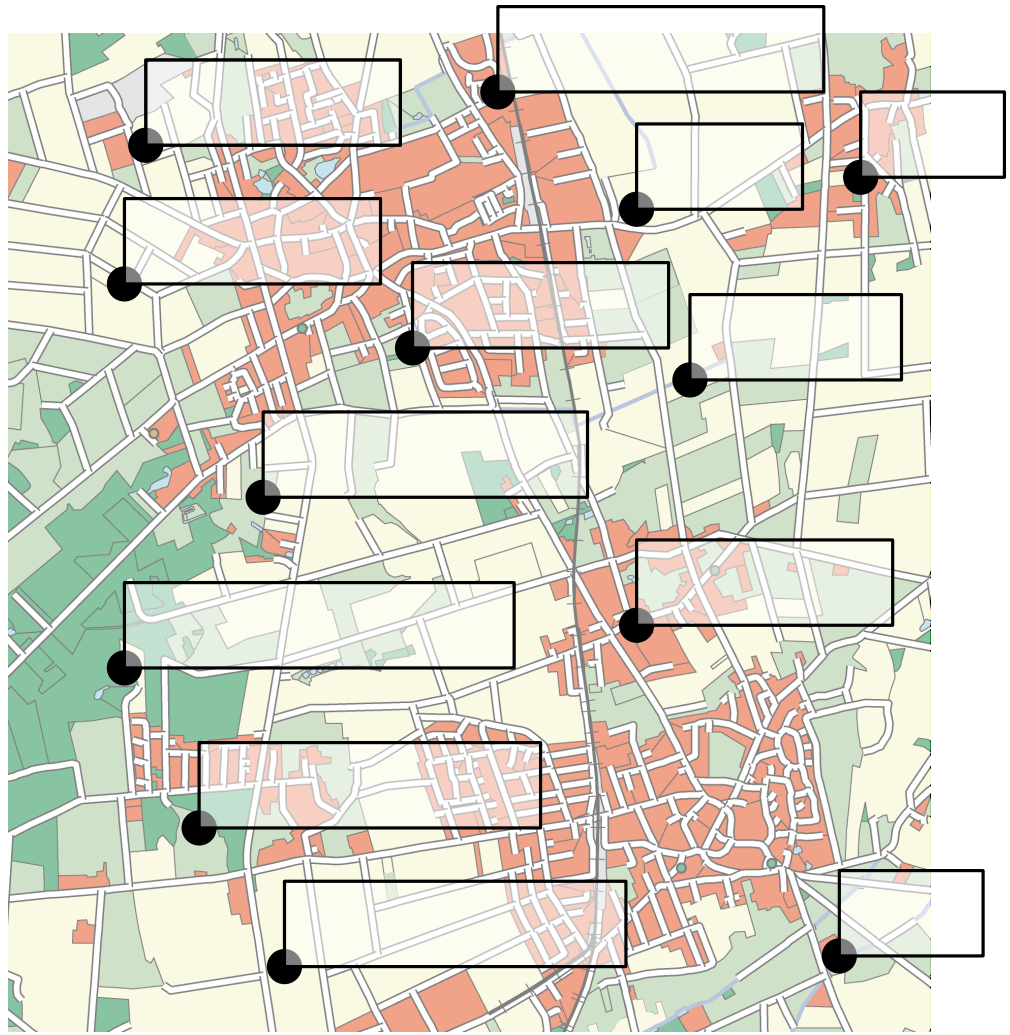
- Generate rectangular candidate labels.
- Define weight (importance) of each label.



Allgemeiner Ansatz

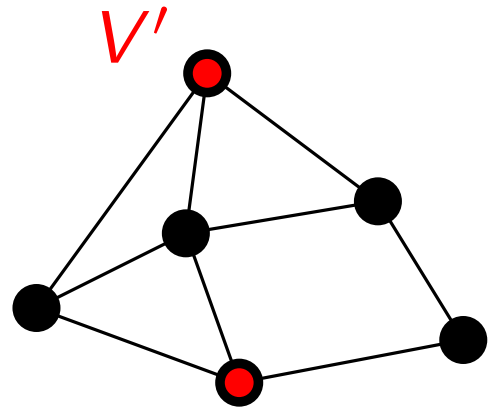
See, e.g., Shieber et al. (1995):

- Generate rectangular candidate labels.
- Define weight (importance) of each label.
- Compute maximum-weight subset of labels without label–label conflicts.



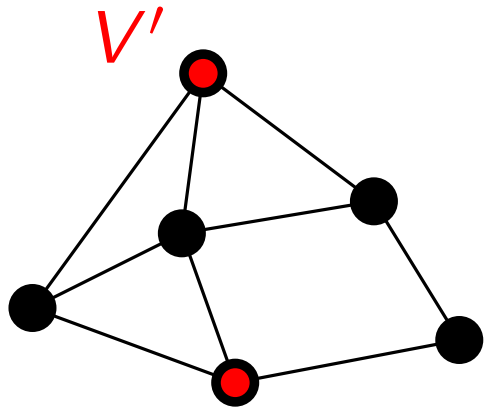
unabhängige Mengen

- Eine *unabhängige Menge* eines Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante $\{u, v\} \in E$ $u \notin V'$ oder $v \notin V'$.



$G = (V, E)$

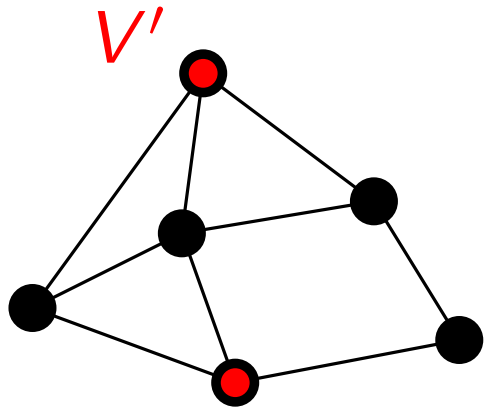
unabhängige Mengen



$G = (V, E)$

- Eine *unabhängige Menge* eines Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante $\{u, v\} \in E$ $u \notin V'$ oder $v \notin V'$.
- in der Anwendung:
  - Jede Ecke in V steht für ein Label (Rechteck).
  - Eine Kante $\{u, v\}$ in E steht für zwei Rechtecke, die sich schneiden.

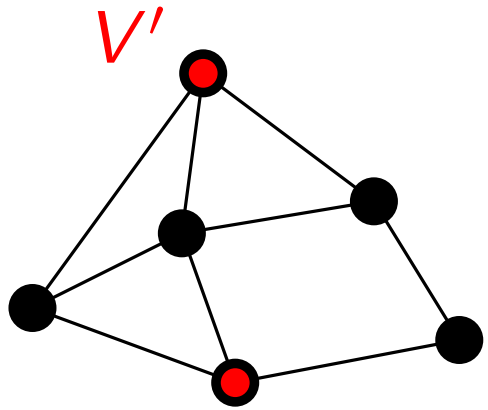
unabhängige Mengen



$G = (V, E)$

- Eine *unabhängige Menge* eines Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante $\{u, v\} \in E$ $u \notin V'$ oder $v \notin V'$.
- in der Anwendung:
  - Jede Ecke in V steht für ein Label (Rechteck).
  - Eine Kante $\{u, v\}$ in E steht für zwei Rechtecke, die sich schneiden.
- Gesucht: (gewichts-)maximale unabhängige Menge in G

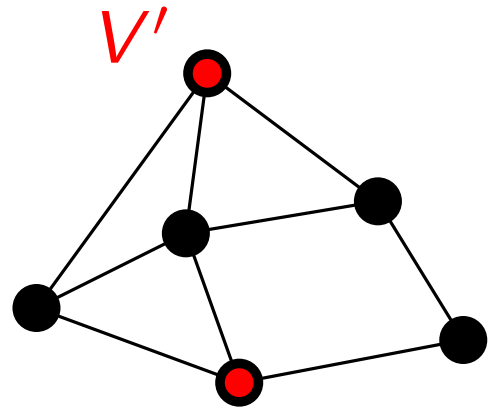
unabhängige Mengen



$G = (V, E)$

- Eine *unabhängige Menge* eines Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante $\{u, v\} \in E$ $u \notin V'$ oder $v \notin V'$.
- in der Anwendung:
  - Jede Ecke in V steht für ein Label (Rechteck).
  - Eine Kante $\{u, v\}$ in E steht für zwei Rechtecke, die sich schneiden.
- Gesucht: (gewichts-)maximale unabhängige Menge in G
- Problem ist NP-schwer für allgemeine Graphen, insbes. aber auch für Graphen, die Rechtecks-Konflikte repräsentieren (Formann & Wagner 1991).

unabhängige Mengen



$G = (V, E)$

- Eine *unabhängige Menge* eines Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante $\{u, v\} \in E$ $u \notin V'$ oder $v \notin V'$.
- in der Anwendung:
  - Jede Ecke in V steht für ein Label (Rechteck).
  - Eine Kante $\{u, v\}$ in E steht für zwei Rechtecke, die sich schneiden.
- Gesucht: (gewichts-)maximale unabhängige Menge in G
- Problem ist NP-schwer für allgemeine Graphen, insbes. aber auch für Graphen, die Rechtecks-Konflikte repräsentieren (Formann & Wagner 1991).
- Für die Praxis relevant: Heuristiken (z.B. Greedy, Simulated Annealing) und effizient lösbare Spezialfälle

max. unabhängige Mengen

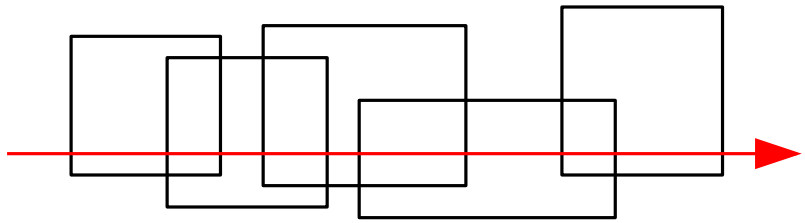
Tractable (= efficiently solvable) cases:

max. unabhängige Mengen

Tractable (= efficiently solvable) cases:

- A horizontal line intersecting all rectangles exists.

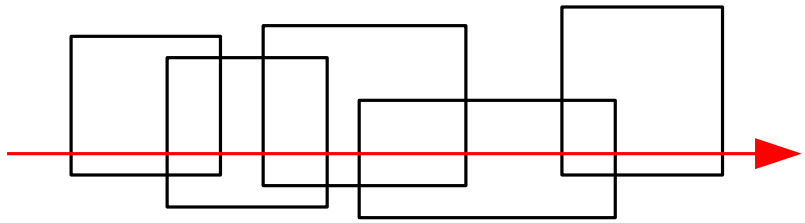
  (van Kreveld et. al 1999)

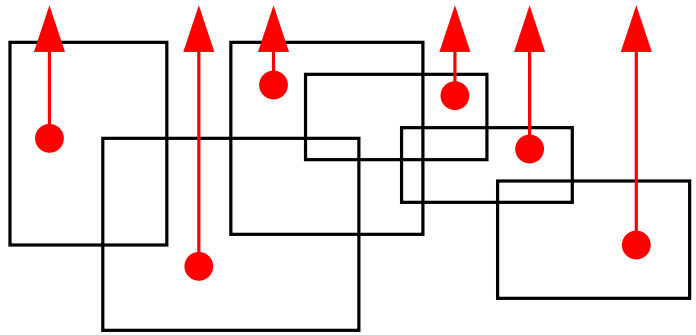


max. unabhängige Mengen

Tractable (= efficiently solvable) cases:

- A horizontal line intersecting all rectangles exists.
  (van Kreveld et. al 1999)
- Each rectangle contains a point above which no other rectangle lies.
  $\rightarrow$ solvable in $O(n^4)$ time
  (Bonsma et. al 2011)



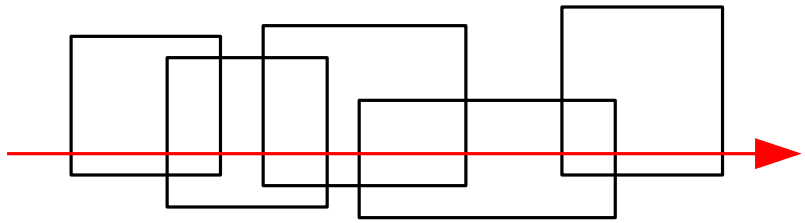


max. unabhängige Mengen

Tractable (= efficiently solvable) cases:

- A horizontal line intersecting all rectangles exists.

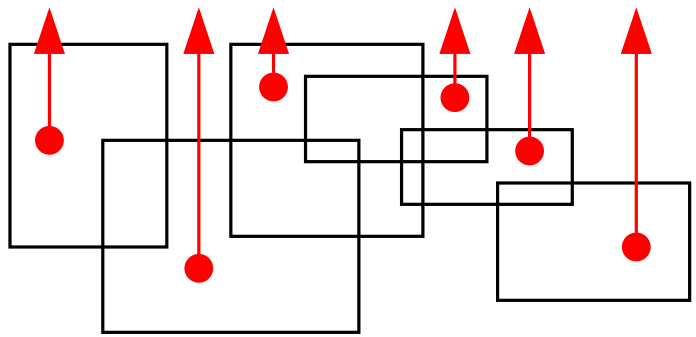
  (van Kreveld et. al 1999)



- Each rectangle contains a point above which no other rectangle lies.

  $\rightarrow$ solvable in $O(n^4)$ time

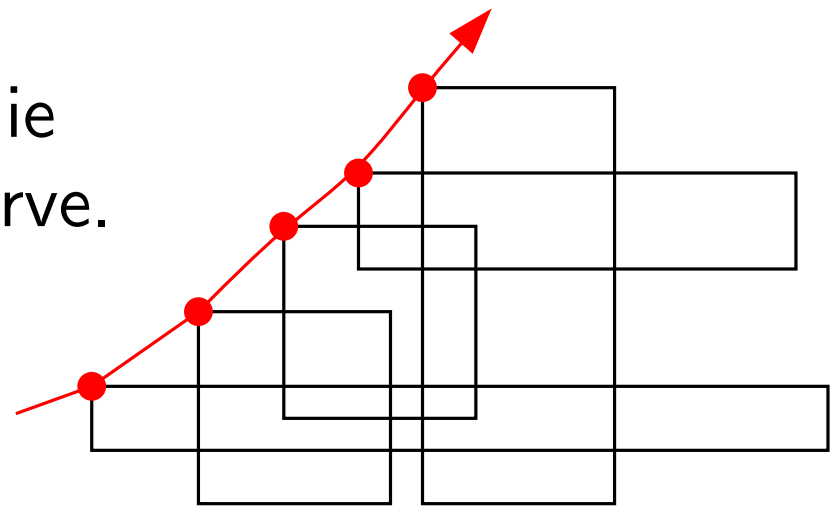
  (Bonsma et. al 2011)



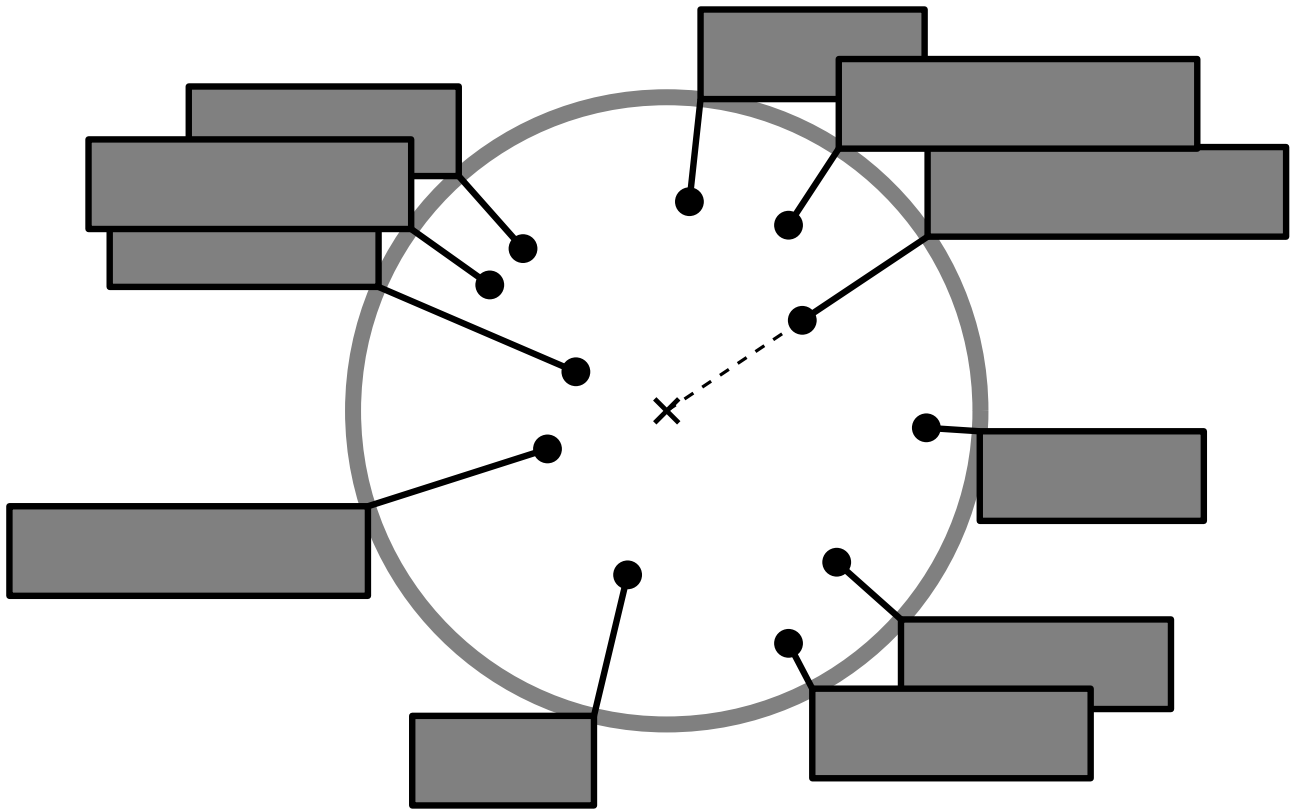
- Upper left corners of rectangles lie on a monotonically ascending curve.

  $\rightarrow$ solvable in $O(n^2)$ time

  (Correa et al. 2015)



Labeling Model

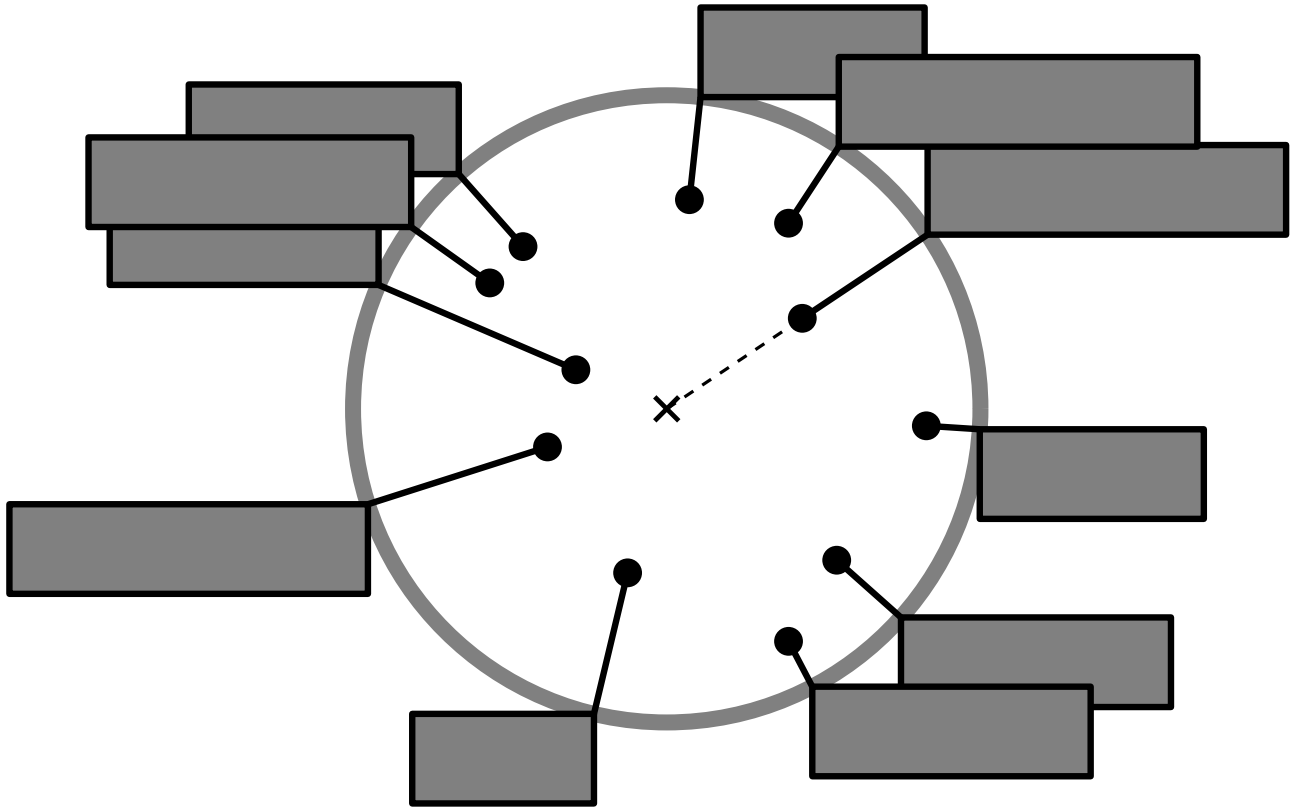


Task: Label POIs in a circular focus region F.

Idea: Project POIs from center of F to boundary of F; place labels there.

To Do: Avoid conflicts.

Algorithms

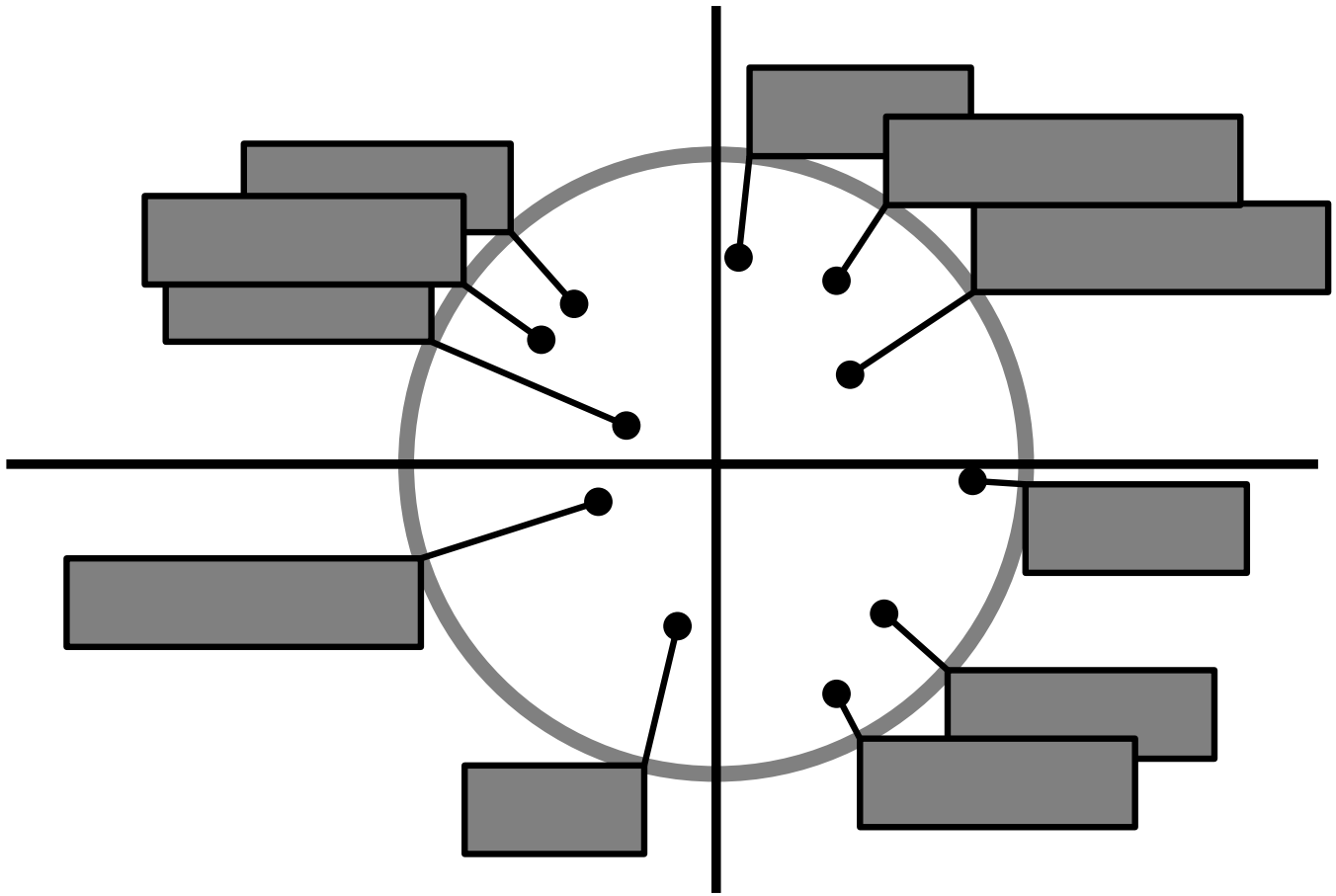


Task: Label POIs in a circular focus region F.

Idea: Project POIs from center of F to boundary of F; place labels there.

To Do: Avoid conflicts. $\rightarrow$ solve MWISR

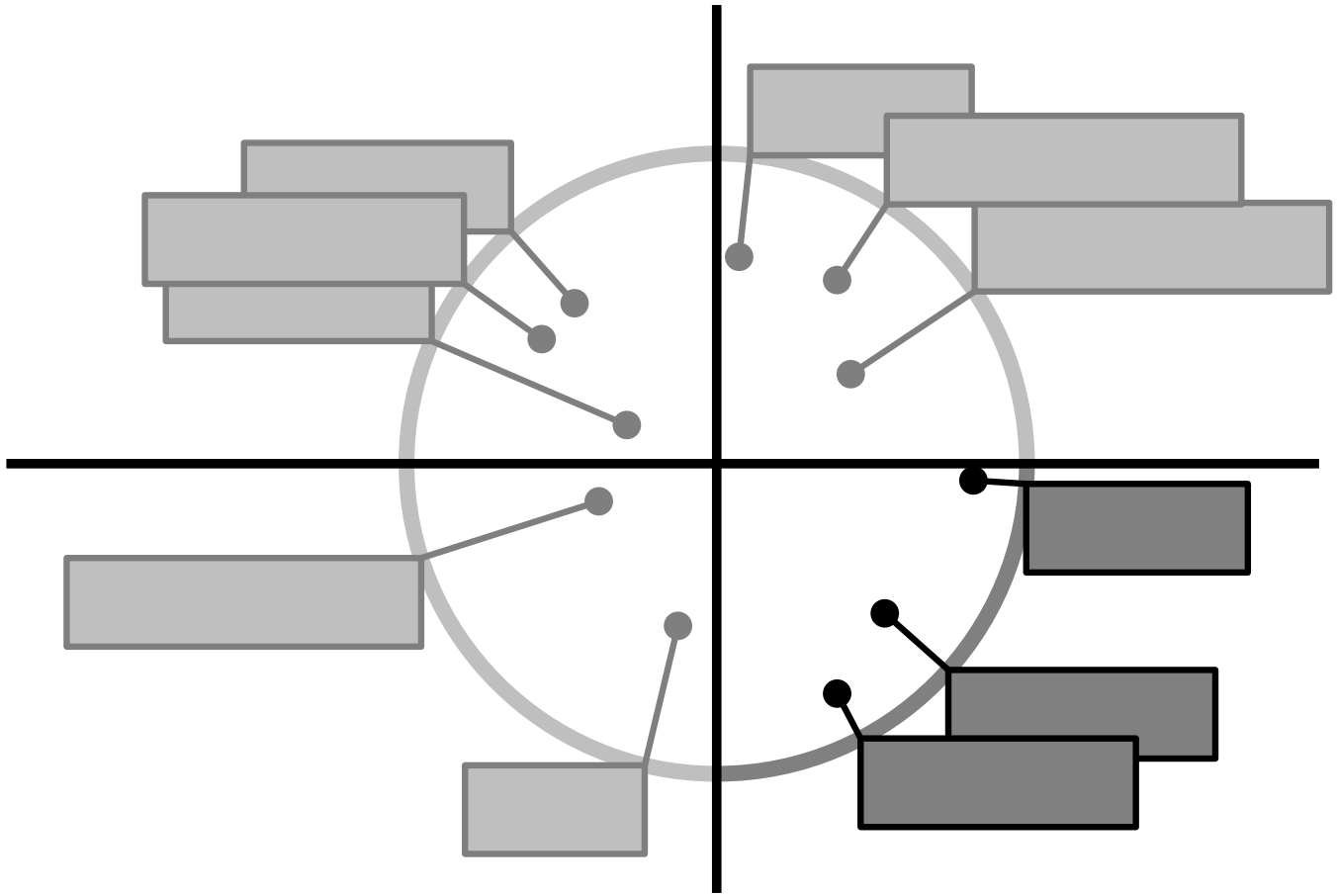
Algorithms



Observation:

Each of the four quandrants of the focus region can be solved independently of the other quadrants, using our algorithm for MWISR.

Algorithms



Observation:

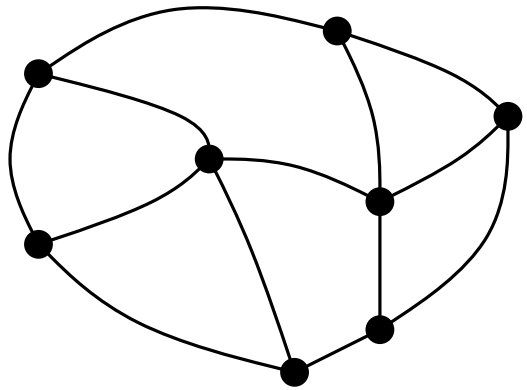
Each of the four quandrants of the focus region can be solved independently of the other quadrants, using our algorithm for MWISR.

Matching

- geläufig aus “Image Matching” oder “Map Matching”

Matching

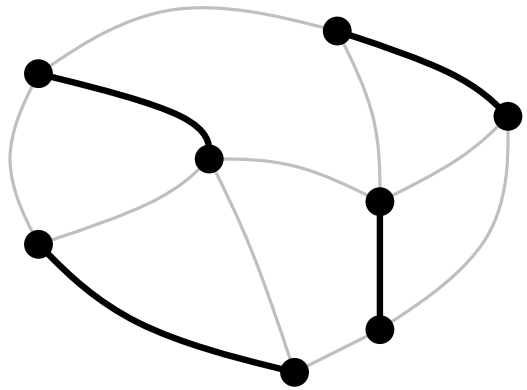
- geläufig aus “Image Matching” oder “Map Matching”
- im graphentheoretischen Sinn:



$G = (V, E)$

Matching

- geläufig aus “Image Matching” oder “Map Matching”
- im graphentheoretischen Sinn:

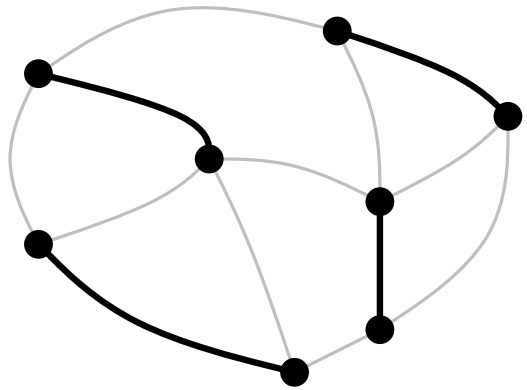


$G = (V, E)$

$M \subseteq E$ ist ein Matching.
$\Leftrightarrow$ Keine zwei Kanten aus M teilen sich einen Knoten.

Matching

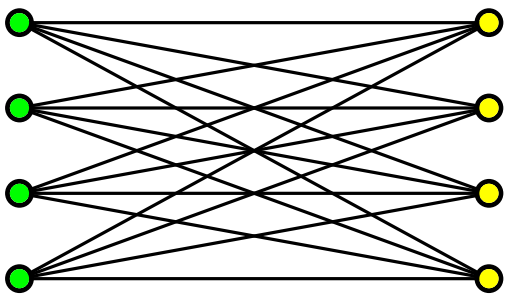
- geläufig aus “Image Matching” oder “Map Matching”
- im graphentheoretischen Sinn:



$G = (V, E)$

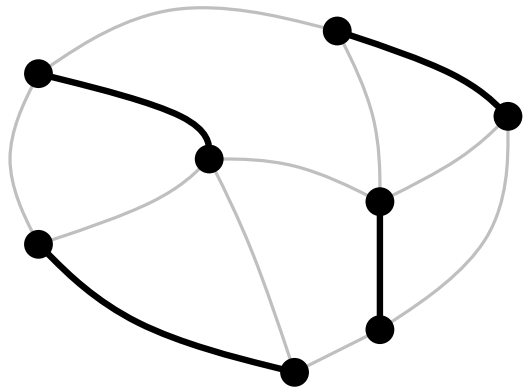
$M \subseteq E$ ist ein Matching.
$\Leftrightarrow$ Keine zwei Kanten aus M teilen sich einen Knoten.

- G ist häufig zweifärbbar (bipartit)



Matching

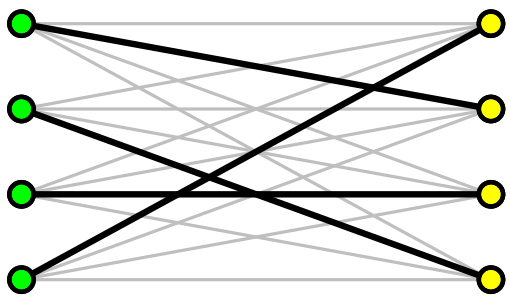
- geläufig aus “Image Matching” oder “Map Matching”
- im graphentheoretischen Sinn:



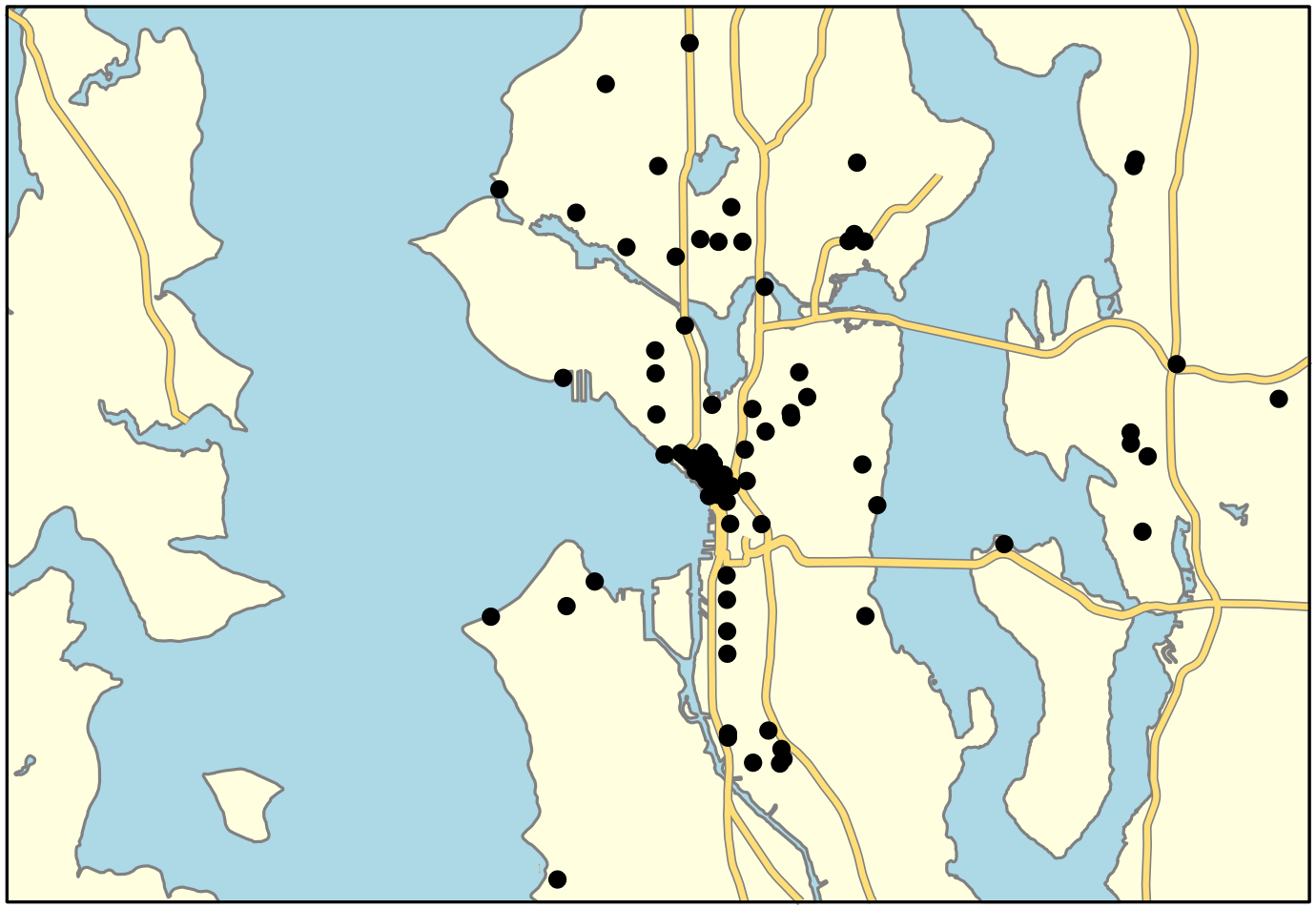
$G = (V, E)$

$M \subseteq E$ ist ein Matching.
$\Leftrightarrow$ Keine zwei Kanten aus M teilen sich einen Knoten.

- G ist häufig zweifärbbar (bipartit)

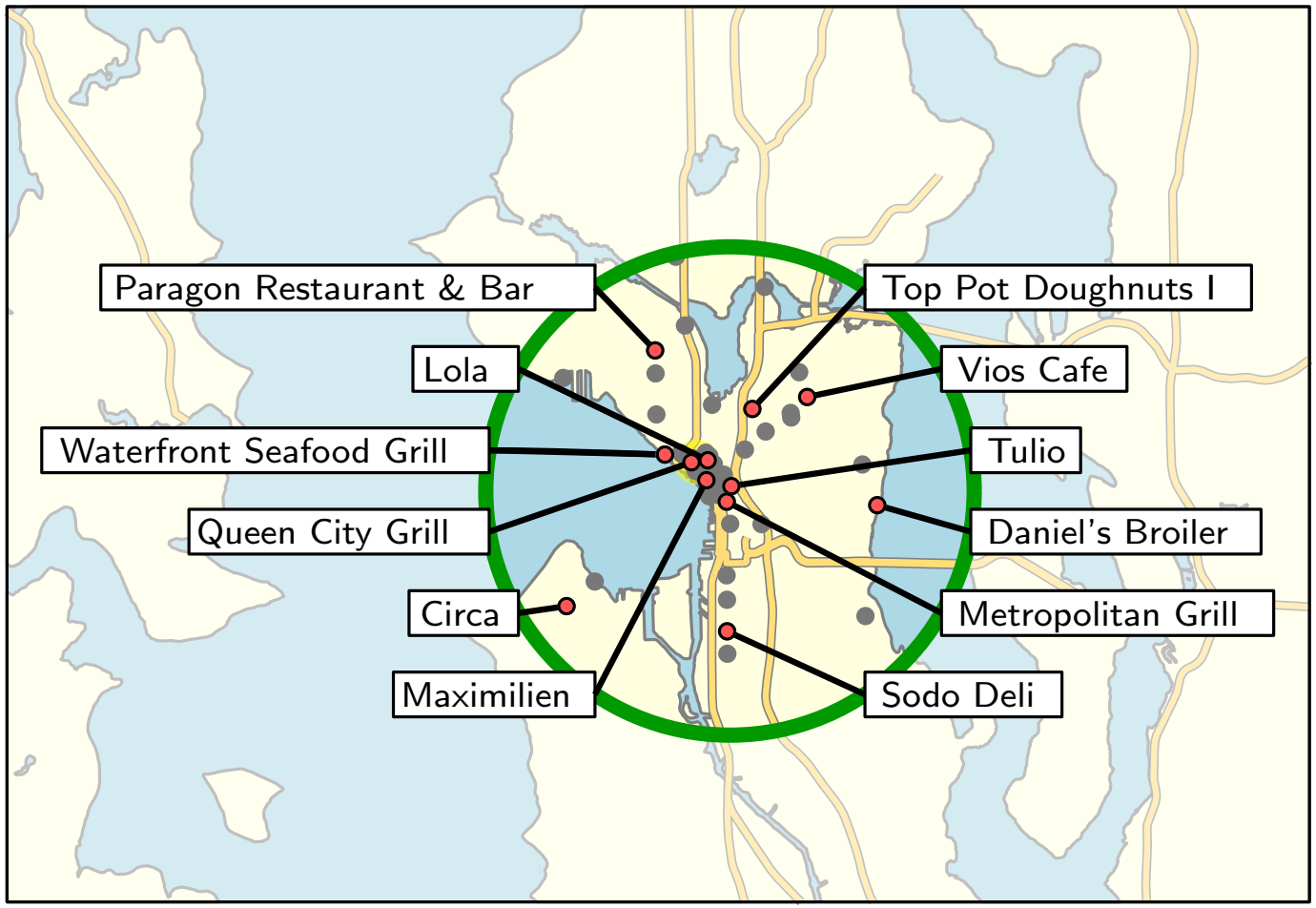


Matching – Anwendung in der Beschriftung



M. Fink, J.-H. Haunert, A. Schulz, J. Spoerhase und A. Wolff.
Algorithms for labeling focus regions.
IEEE Transactions on Visualization and Computer Graphics (Proc. InfoVis'12), 18(12):2583–2592, 2012.

Matching – Anwendung in der Beschriftung

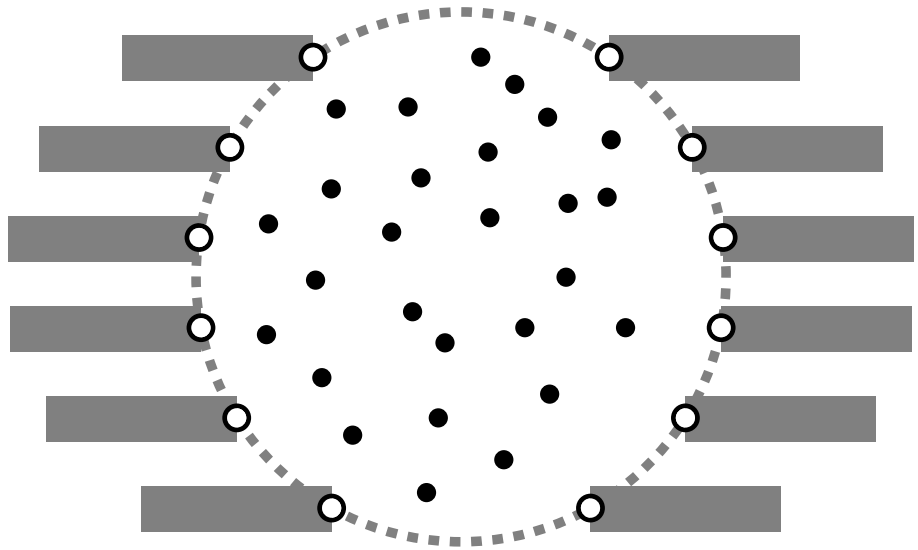


M. Fink, J.-H. Haunert, A. Schulz, J. Spoerhase und A. Wolff.
Algorithms for labeling focus regions.
IEEE Transactions on Visualization and Computer Graphics (Proc. InfoVis'12), 18(12):2583–2592, 2012.

Matching – Anwendung in der Beschriftung

Problem:

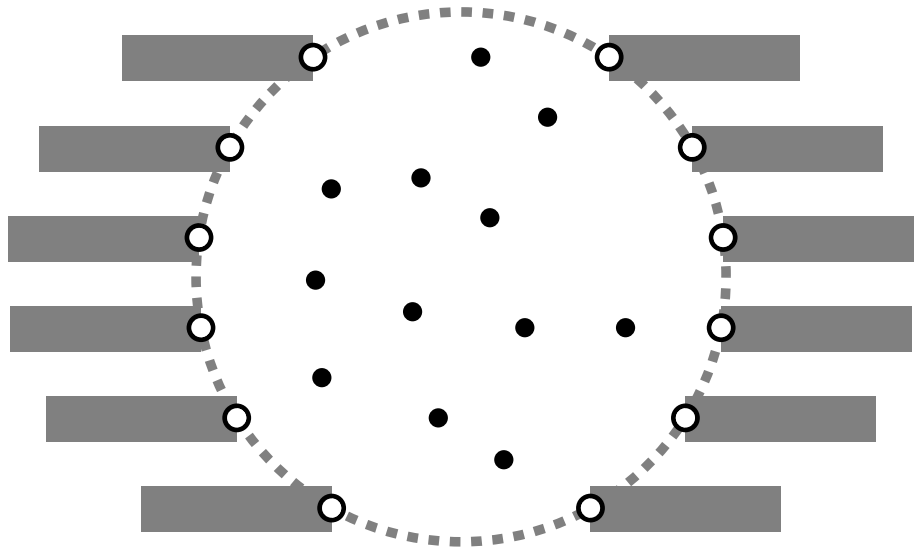
- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$



Matching – Anwendung in der Beschriftung

Problem:

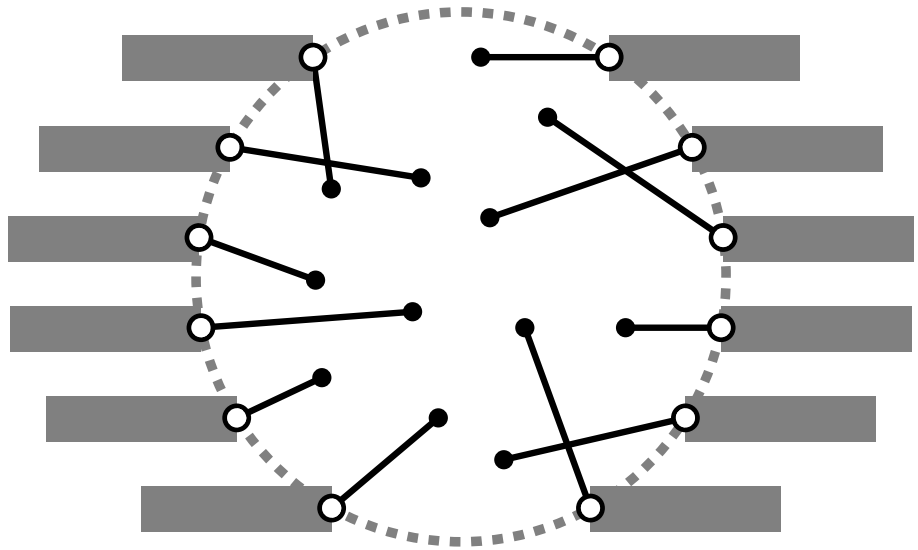
- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.



Matching – Anwendung in der Beschriftung

Problem:

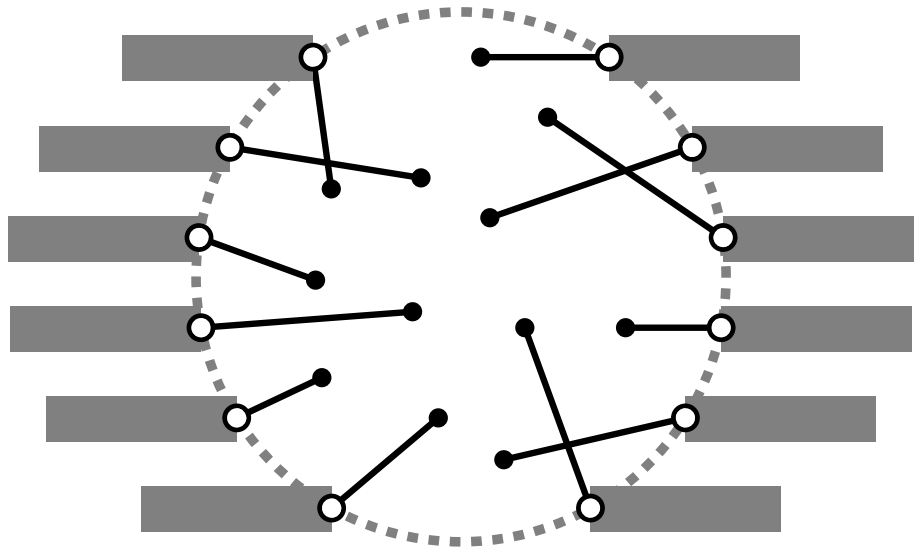
- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.



Matching – Anwendung in der Beschriftung

Problem:

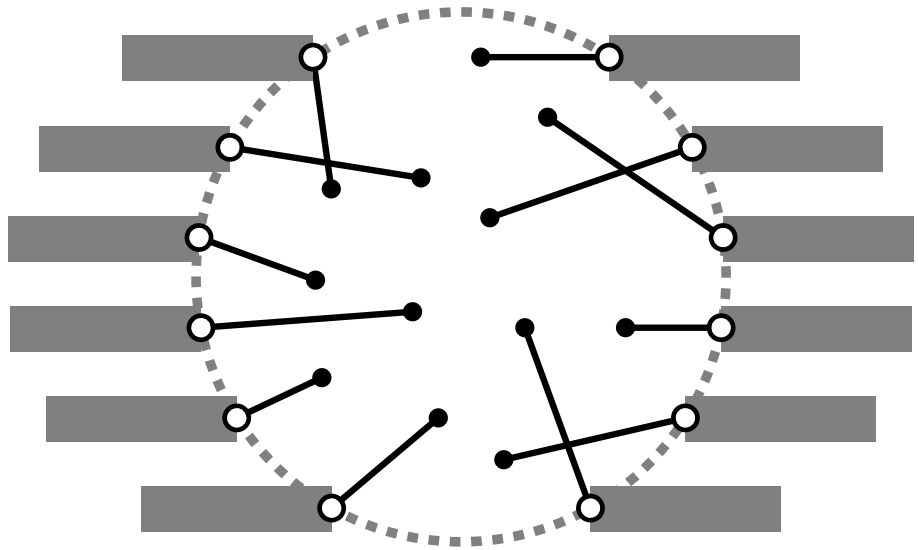
- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.
- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.



Matching – Anwendung in der Beschriftung

Problem:

- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.



- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.

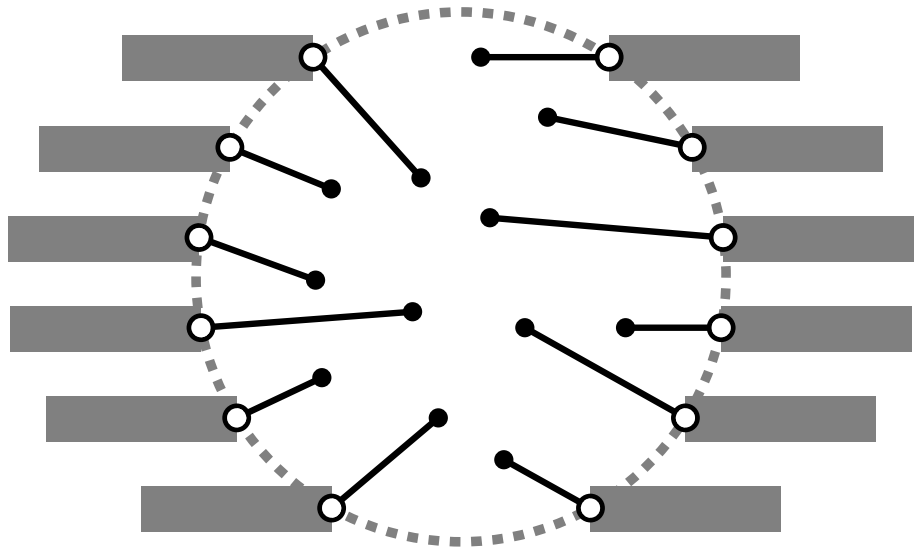
  $W =$ Gesamtgewicht der ausgewählten Punkte

  $L =$ Gesamtlänge der Leader

Matching – Anwendung in der Beschriftung

Problem:

- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.
- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.



Leader weisen keine Schnitte auf!

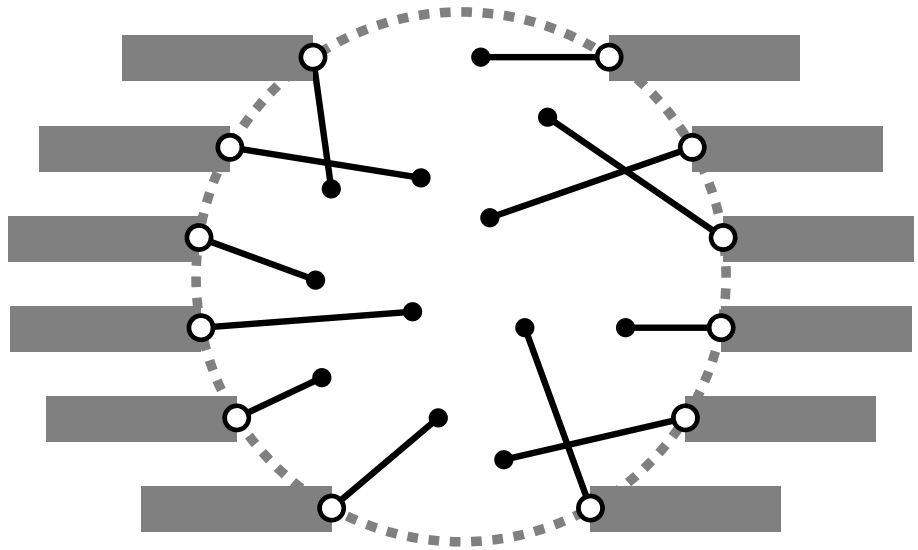
$W =$ Gesamtgewicht der ausgewählten Punkte

$L =$ Gesamtlänge der Leader

Matching – Anwendung in der Beschriftung

Problem:

- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.
- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.



Leader weisen keine Schnitte auf!

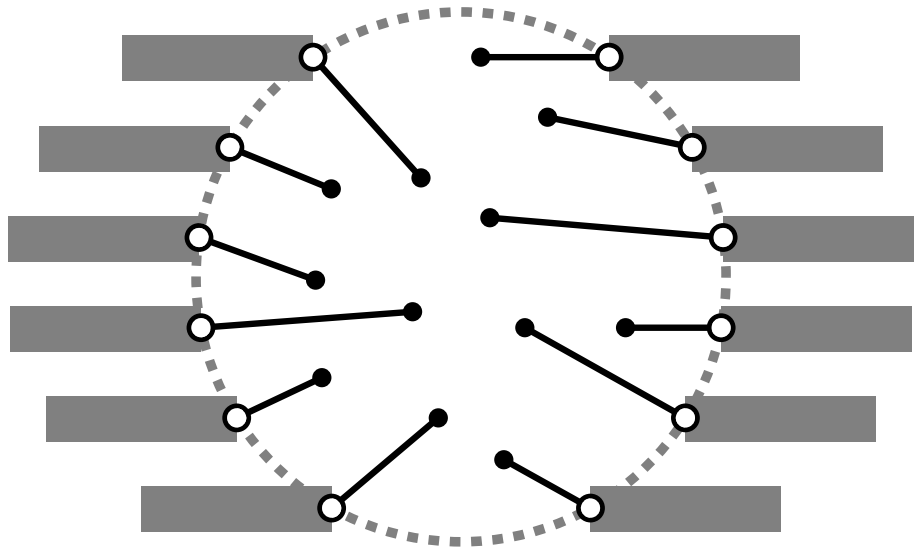
$W =$ Gesamtgewicht der ausgewählten Punkte

$L =$ Gesamtlänge der Leader

Matching – Anwendung in der Beschriftung

Problem:

- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.
- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.



Leader weisen keine Schnitte auf!

$W =$ Gesamtgewicht der ausgewählten Punkte

$L =$ Gesamtlänge der Leader

Matching – Anwendung in der Beschriftung

Problem:

- gegeben:
  - n Punkte ($\bullet$) mit Gewichten
  - $k \leq n$ Labelpositionen ($\circ$) auf Rand der Fokusregion
  - Gewichtsfaktor $0 < \lambda < 1$
- Wähle k Punkte.
- Bilde 1-zu-1-Beziehungen zwischen ausgewählten Punkten und Labelpositionen.
- Maximiere $\lambda \cdot W - (1 - \lambda) \cdot L$.

  $W =$ Gesamtgewicht der ausgewählten Punkte

  $L =$ Gesamtlänge der Leader

Algorithmus:

Ungarische Methode
(gewichtsmaximales Matching
in bipartitem Graph)

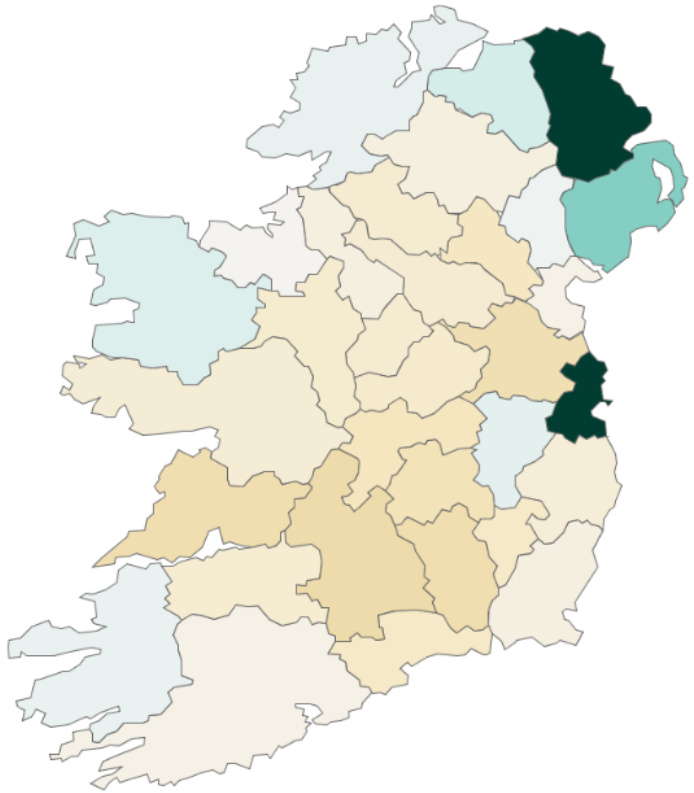
Laufzeit:

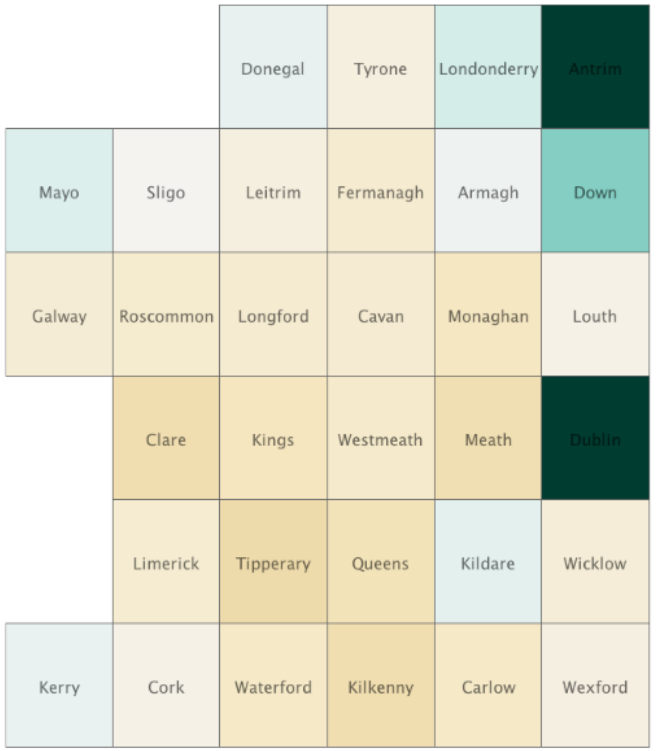
$$\mathcal{O}(n^3)$$

Erzeugung von Grid Maps

Grid Maps – Examples

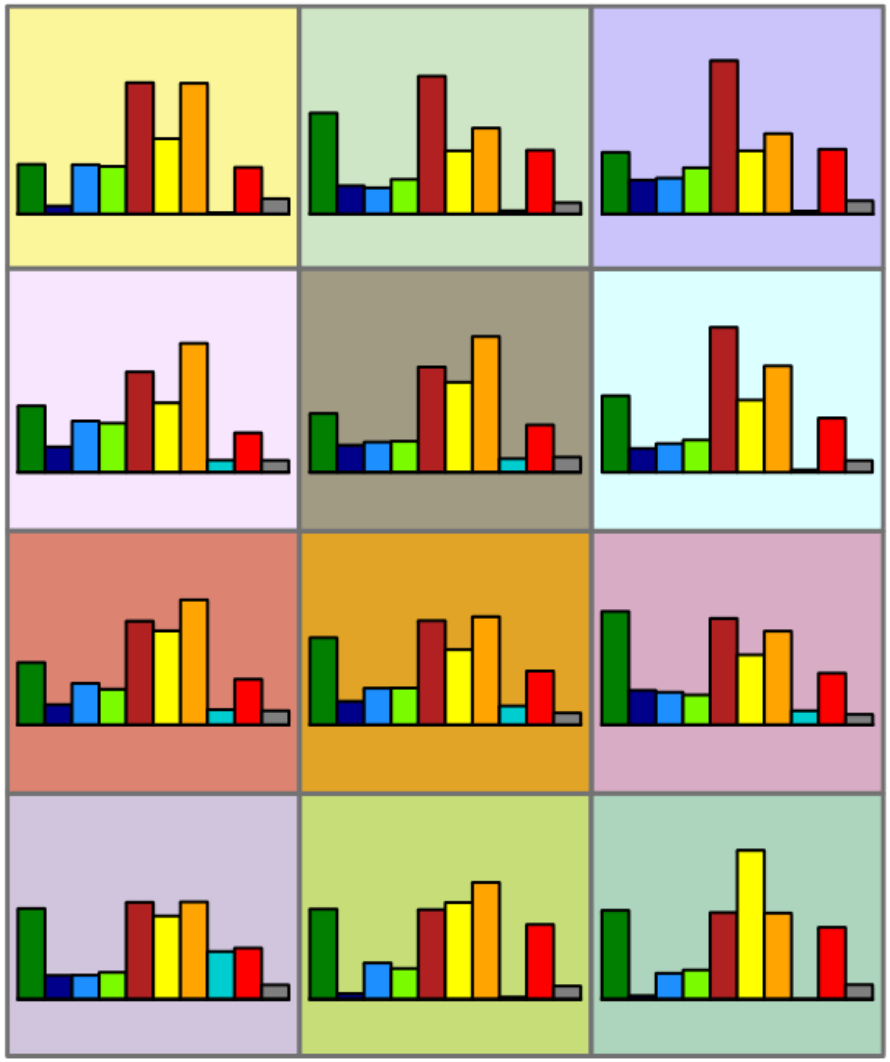
population change per county between 1851 and 1911





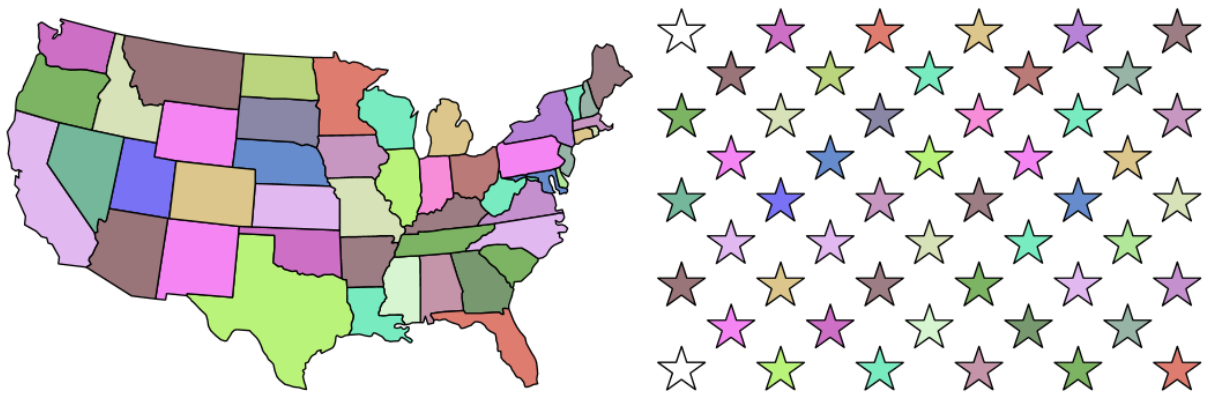
M. Kelly, A. Slingsby, J. Dykes, J. Wood.
Historical Internal Migration in Ireland.
In: Proc. GIS Research UK Conference (GISRUK'13), 2013.

Grid Maps – Examples



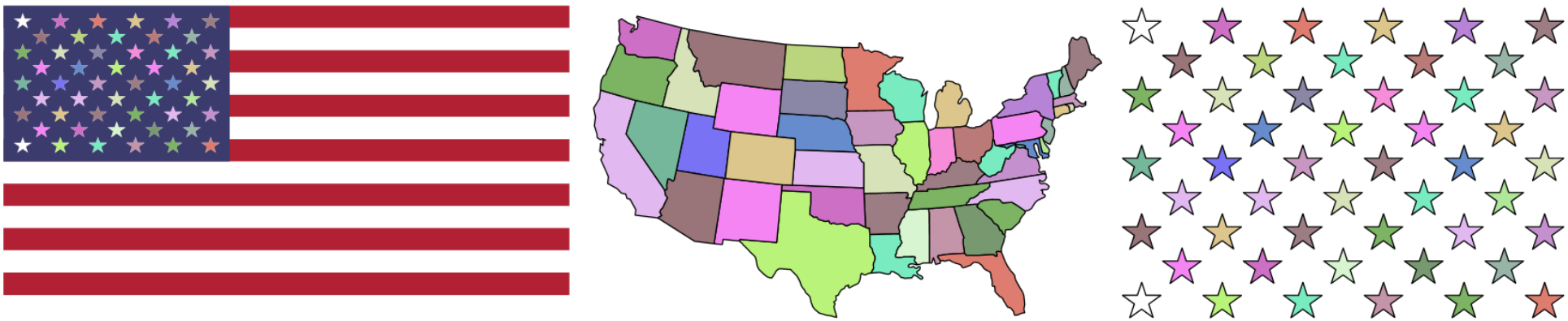
D. Eppstein, M. van Kreveld, B. Speckmann, F. Staals.
Improved Grid Map Layout by Point Set Matching.
In: Proc. IEEE Pacific Visualization Symposium 2013.

Grid Maps – Examples



D. Eppstein, M. van Kreveld, B. Speckmann, F. Staals.
Improved Grid Map Layout by Point Set Matching.
In: Proc. IEEE Pacific Visualization Symposium 2013.

Grid Maps – Examples



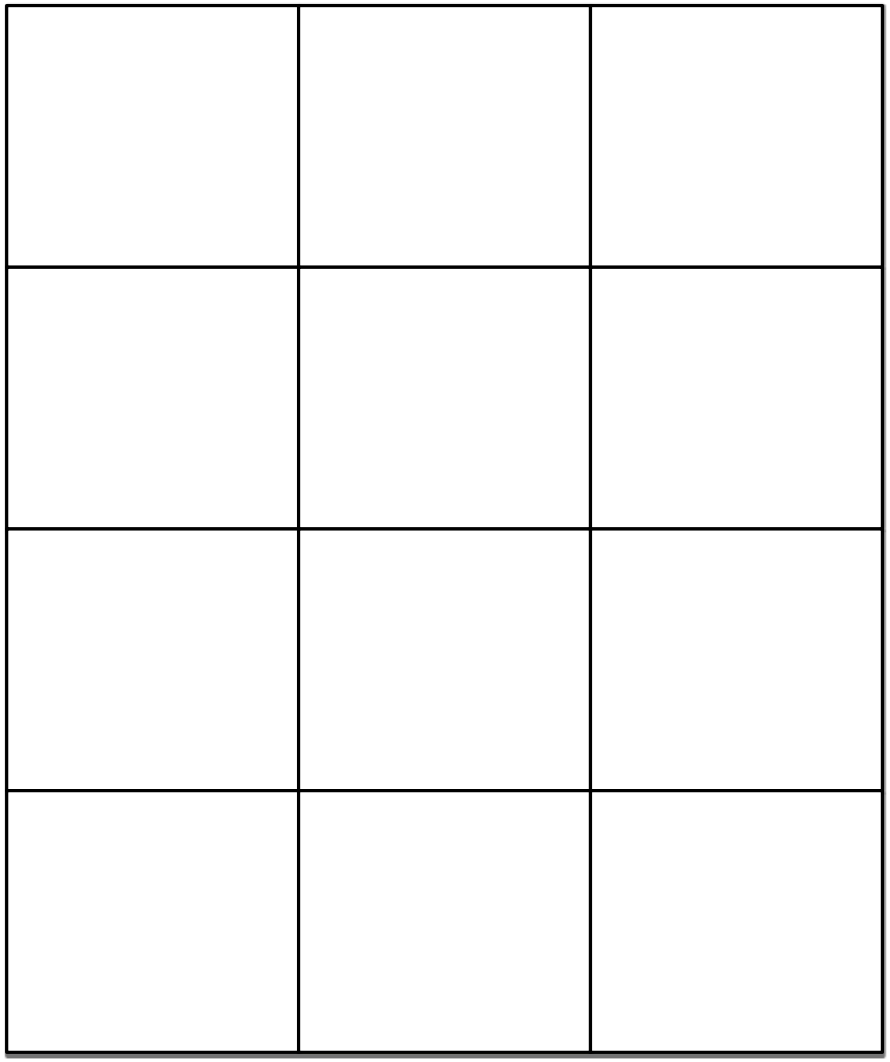
D. Eppstein, M. van Kreveld, B. Speckmann, F. Staals.
Improved Grid Map Layout by Point Set Matching.
In: Proc. IEEE Pacific Visualization Symposium 2013.

Grid Maps

- display a raster in which every cell represents a geographic region.
- pros: overview; equal space for every region
- cons: geometric distortion; adjacency relationships not preserved

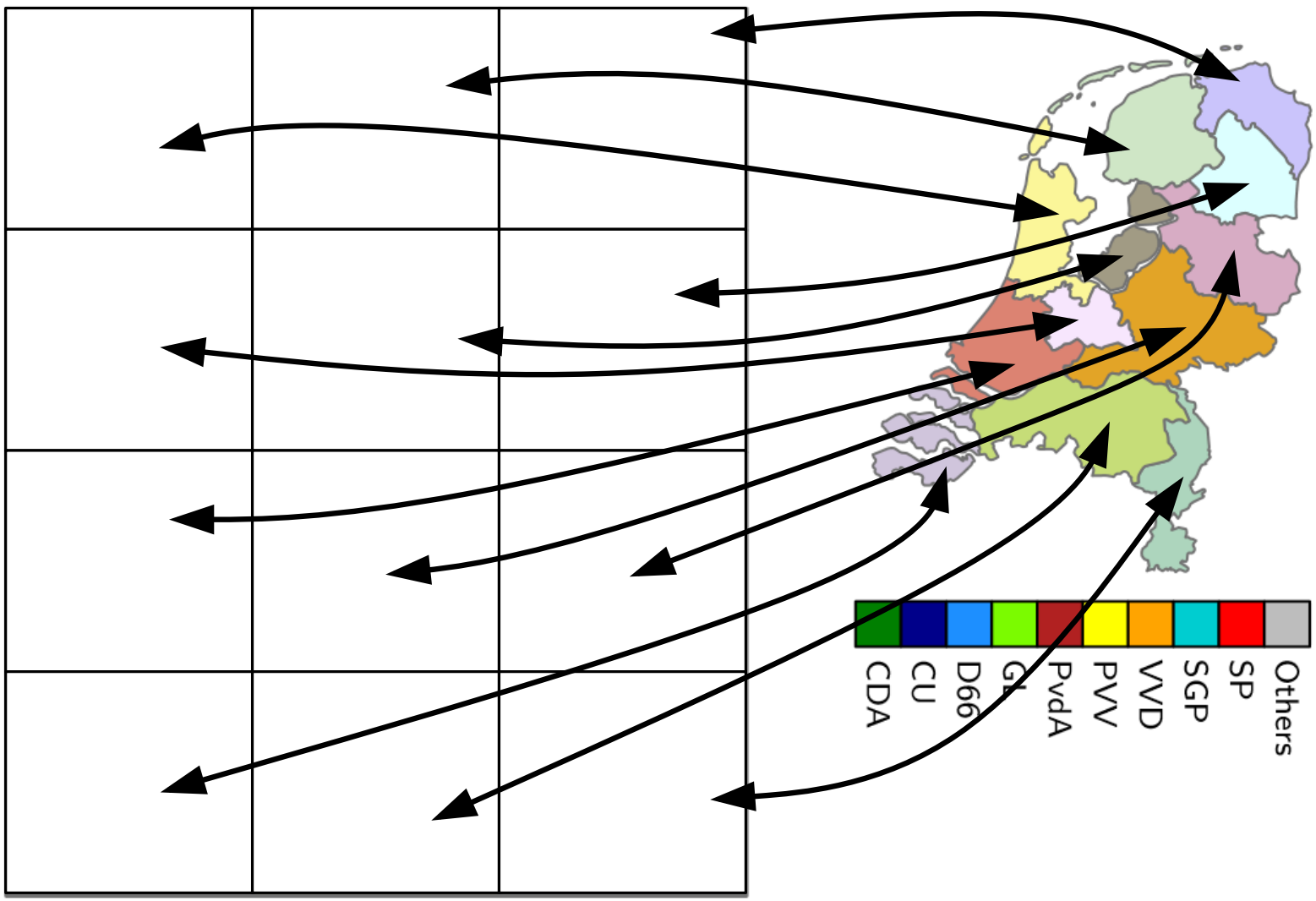
Problem

- Input: A grid of n cells and a map of n regions



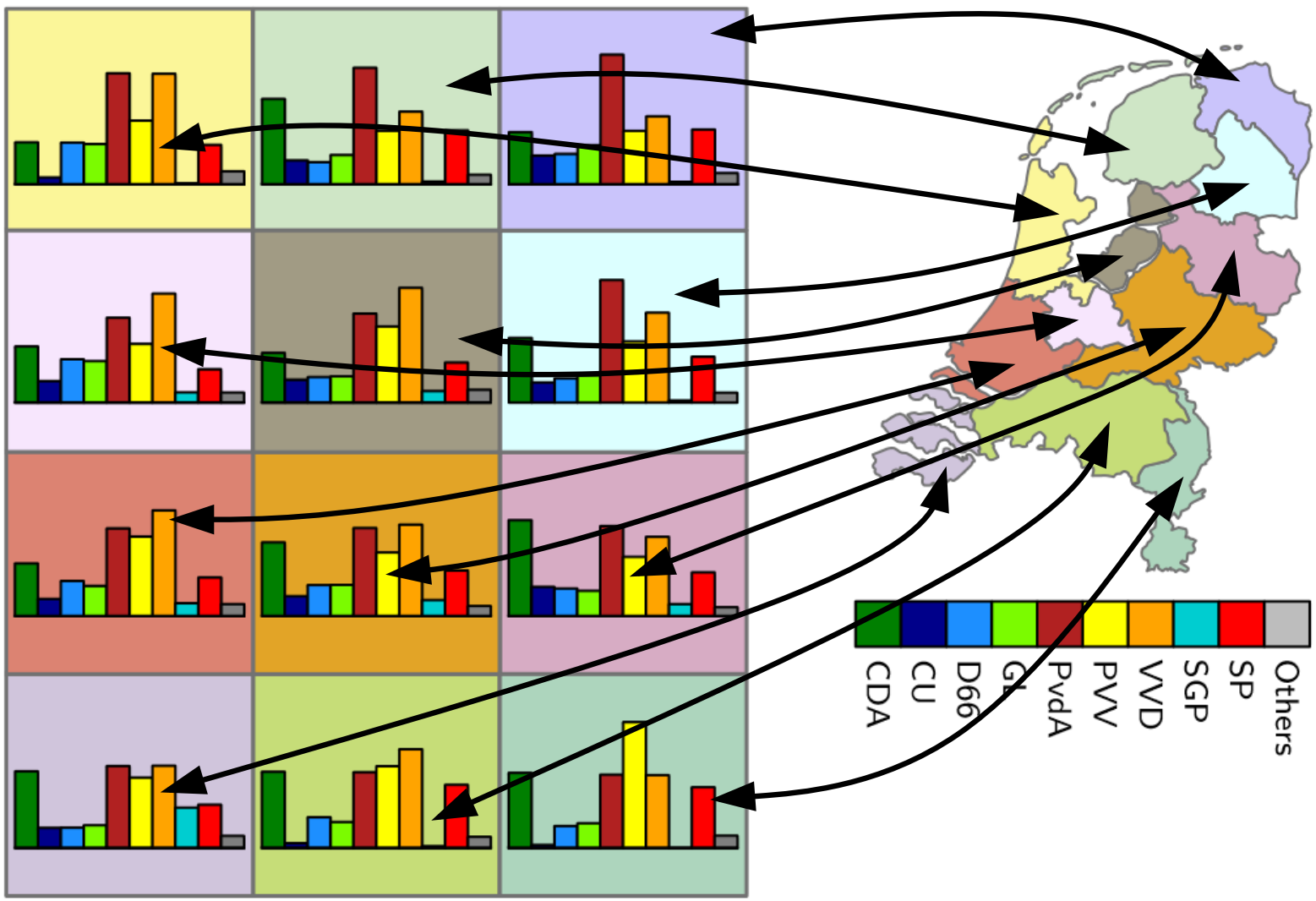
Problem

- Input: A grid of n cells and a map of n regions
- Problem: Define correspondences between regions and cells



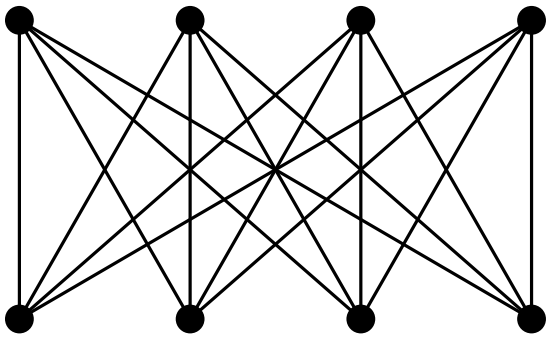
Problem

- Input: A grid of n cells and a map of n regions
- Problem: Define correspondences between regions and cells



General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



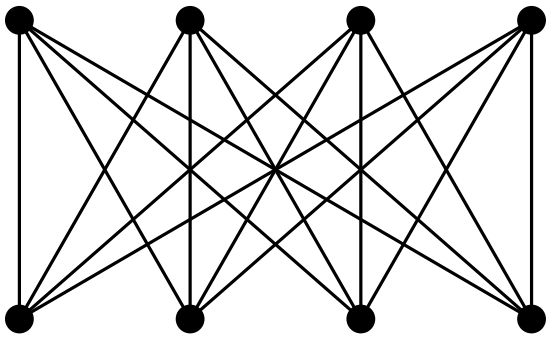
V_1: geographic regions

E: possible assignments

V_2: raster cells

General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



V_1: geographic regions

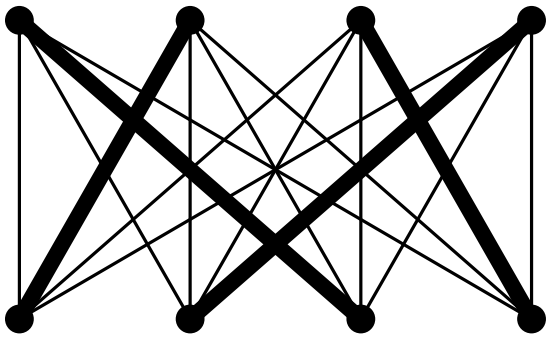
E: possible assignments

V_2: raster cells

- Define cost $c(uv)$ of every edge $uv \in E$, for example, $c(uv) =$ Distance between centroids of u and v.

General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



V_1: geographic regions

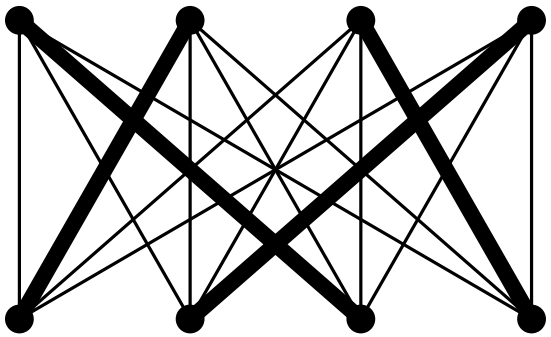
E: possible assignments

V_2: raster cells

- Define cost $c(uv)$ of every edge $uv \in E$, for example, $c(uv) =$ Distance between centroids of u and v.
- Compute minimum-cost perfect matching in G.

General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



V_1: geographic regions

E: possible assignments

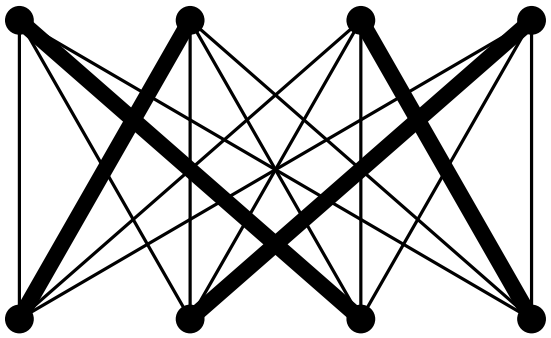
V_2: raster cells

- Define cost $c(uv)$ of every edge $uv \in E$, for example, $c(uv) =$ Distance between centroids of u and v.
- Compute minimum-cost perfect matching in G.

= assignment problem

General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



V_1: geographic regions

E: possible assignments

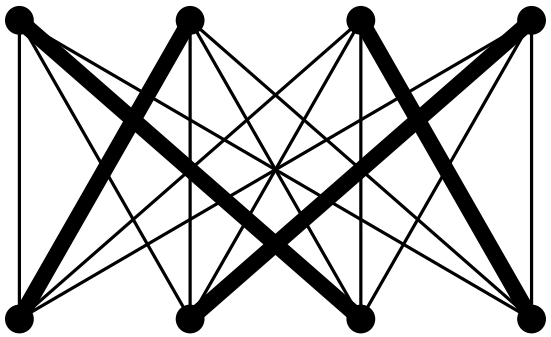
V_2: raster cells

- Define cost $c(uv)$ of every edge $uv \in E$, for example, $c(uv) =$ Distance between centroids of u and v.
- Compute minimum-cost perfect matching in G.

= assignment problem

General Approach

- Define a complete bipartite graph $G = (V, E)$ with $V = V_1 \cup V_2$ and $E = V_1 \times V_2$



V_1: geographic regions

E: possible assignments

V_2: raster cells

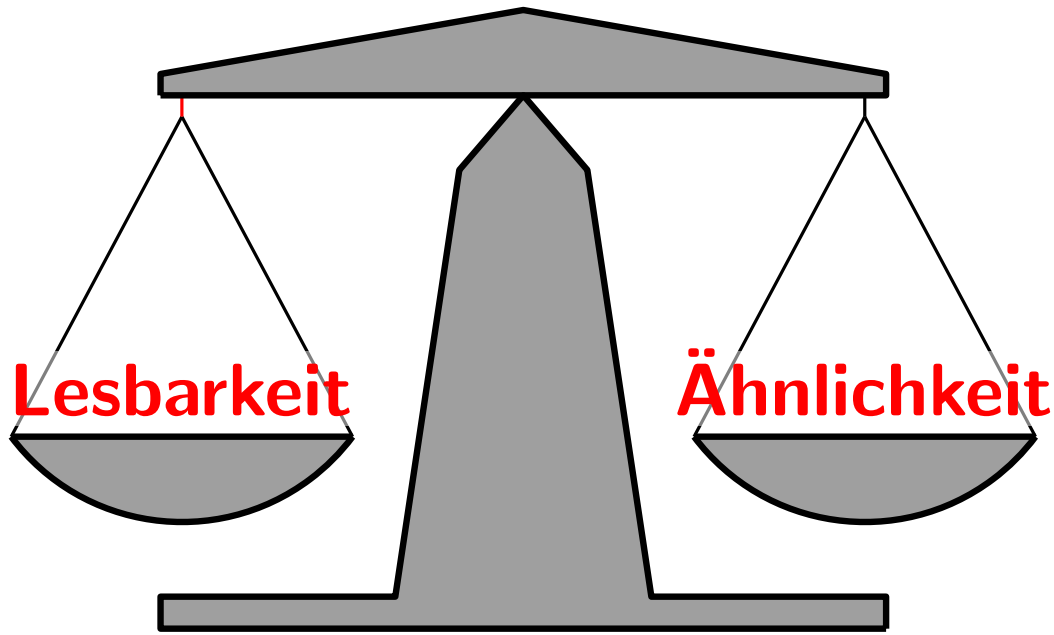
- Define cost $c(uv)$ of every edge $uv \in E$, for example, $c(uv) =$ Distance between centroids of u and v.
- Compute minimum-cost perfect matching in G.

= assignment problem

http://vimeo.com/50448511

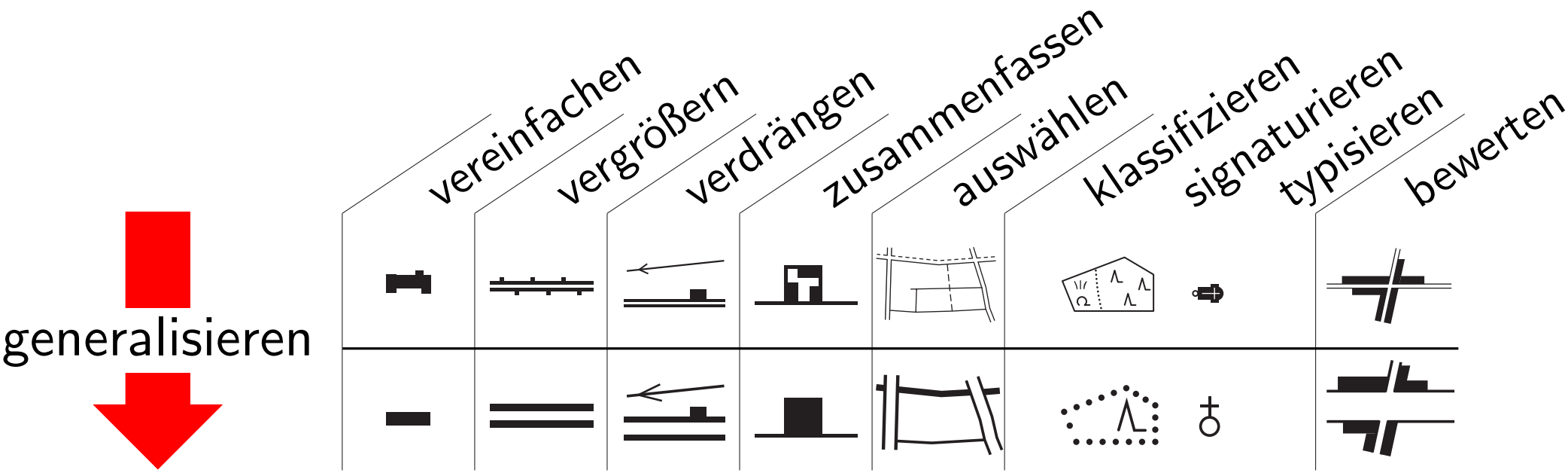
Linienvereinfachung

Ziele der Generalisierung



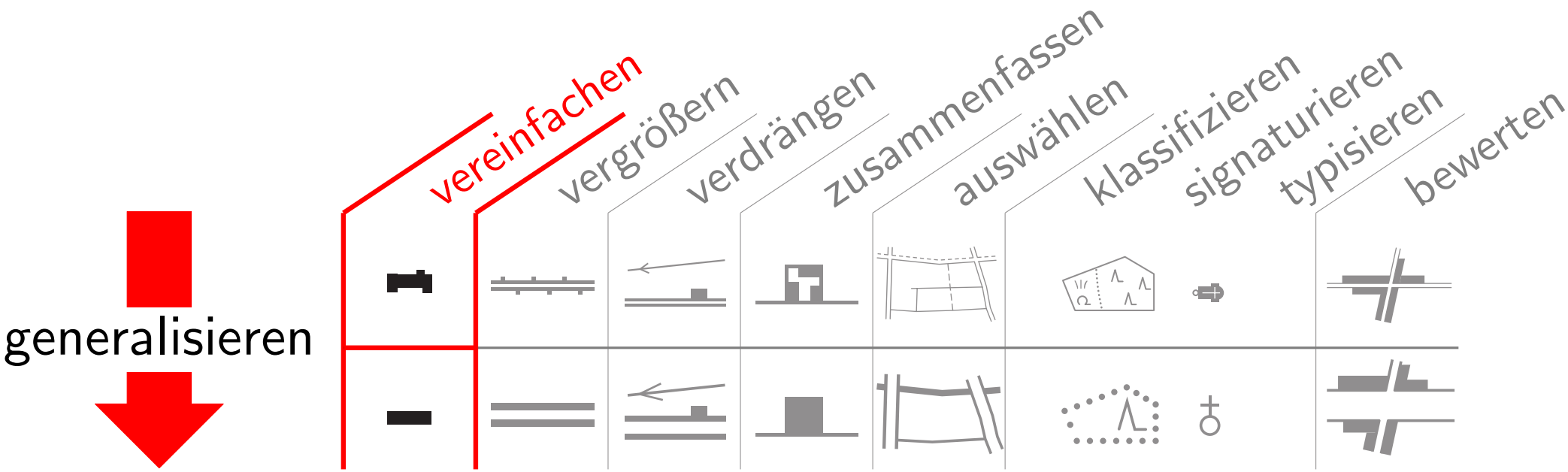
(Burghardt et al., 2007)

Veinfachung in der Generalisierung



nach Hake et al. (1994)

Veinfachung in der Generalisierung

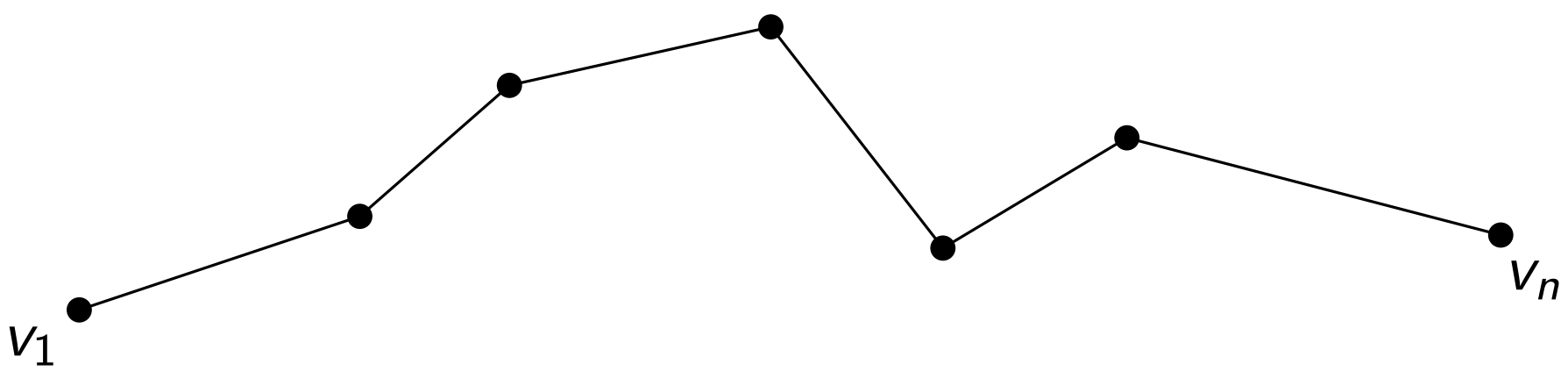


nach Hake et al. (1994)

Linienvereinfachung

Problem (Imai & Iri, 1988)

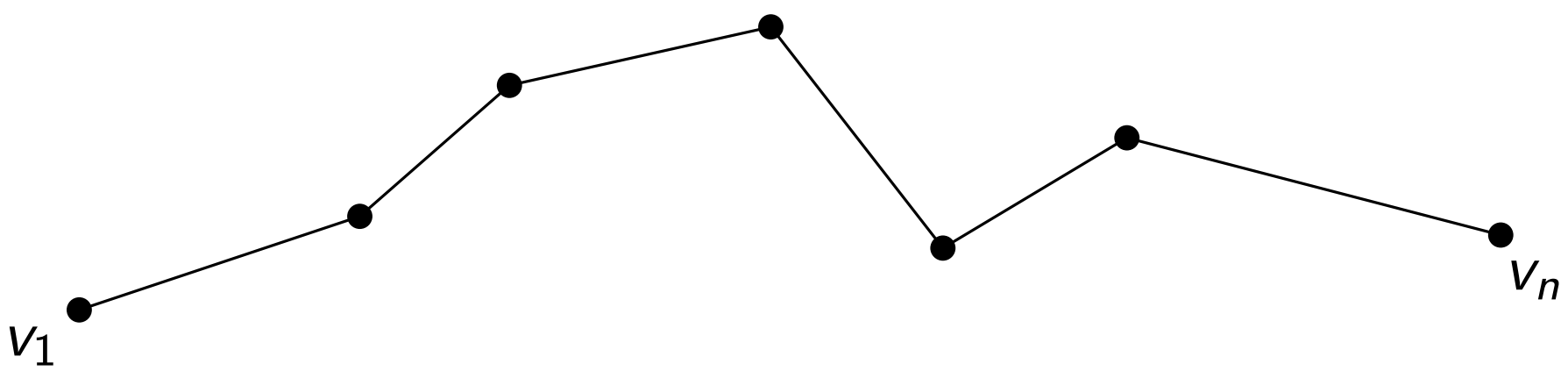
- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken



Linienvereinfachung

Problem (Imai & Iri, 1988)

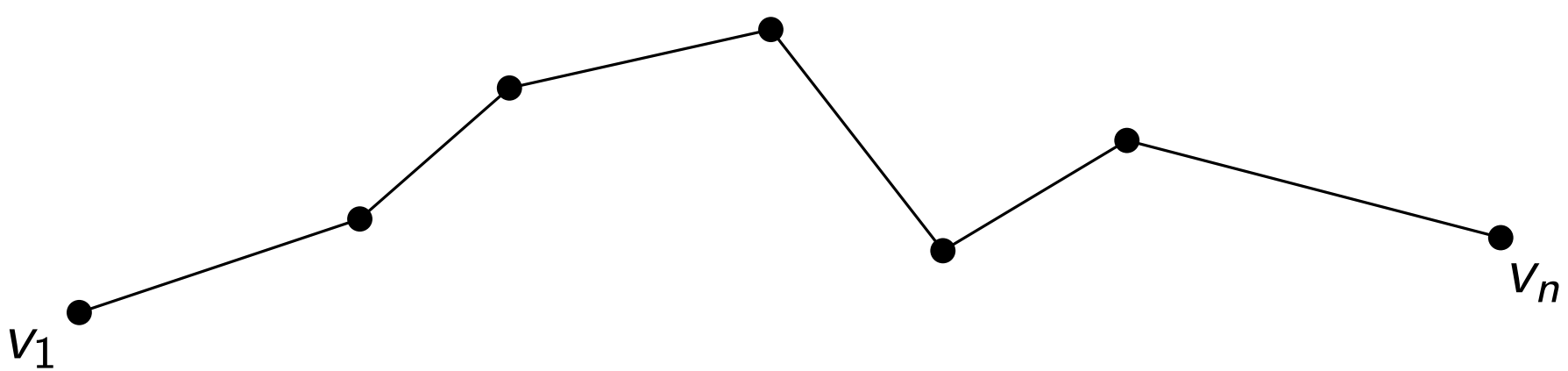
- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken
  - Fehlertoleranz $\varepsilon =$ ⟼⟻



Linienvereinfachung

Problem (Imai & Iri, 1988)

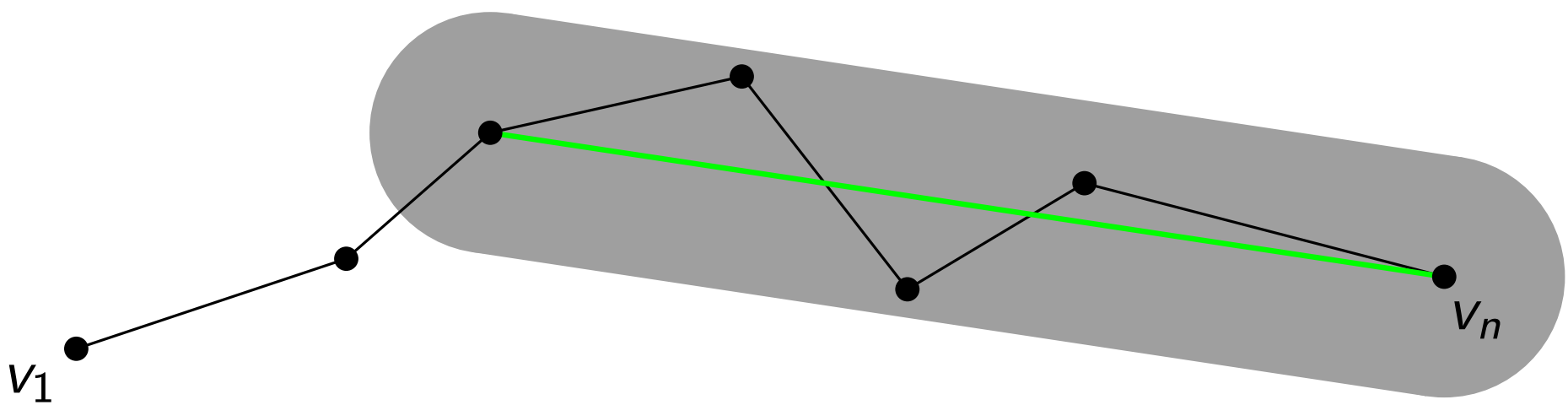
- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken
  - Fehlertoleranz $\varepsilon =$ ⟼⟻
- Berechne möglichst kurze Teilfolge V' von V, so dass
  - V' die Ecken v_1 und v_n enthält
  - für jede Kante $e = (v_i, v_k)$ in V' und jede Ecke v_j mit $i < j < k$ gilt: $\text{Distanz}(e, v_j) \leq \varepsilon$.



Linienvereinfachung

Problem (Imai & Iri, 1988)

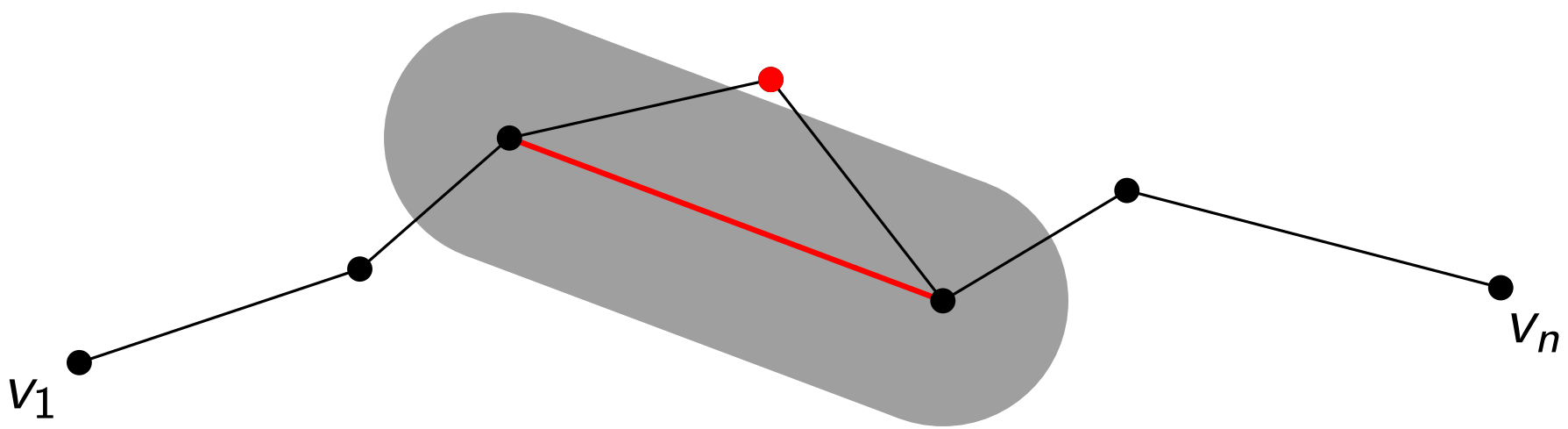
- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken
  - Fehlertoleranz $\varepsilon =$ ⟼⟻
- Berechne möglichst kurze Teilfolge V' von V, so dass
  - V' die Ecken v_1 und v_n enthält
  - für jede Kante $e = (v_i, v_k)$ in V' und jede Ecke v_j mit $i < j < k$ gilt: $\text{Distanz}(e, v_j) \leq \varepsilon$.



Linienvereinfachung

Problem (Imai & Iri, 1988)

- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken
  - Fehlertoleranz $\varepsilon =$ ⟼
- Berechne möglichst kurze Teilfolge V' von V, so dass
  - V' die Ecken v_1 und v_n enthält
  - für jede Kante $e = (v_i, v_k)$ in V' und jede Ecke v_j mit $i < j < k$ gilt: $\text{Distanz}(e, v_j) \leq \varepsilon$.



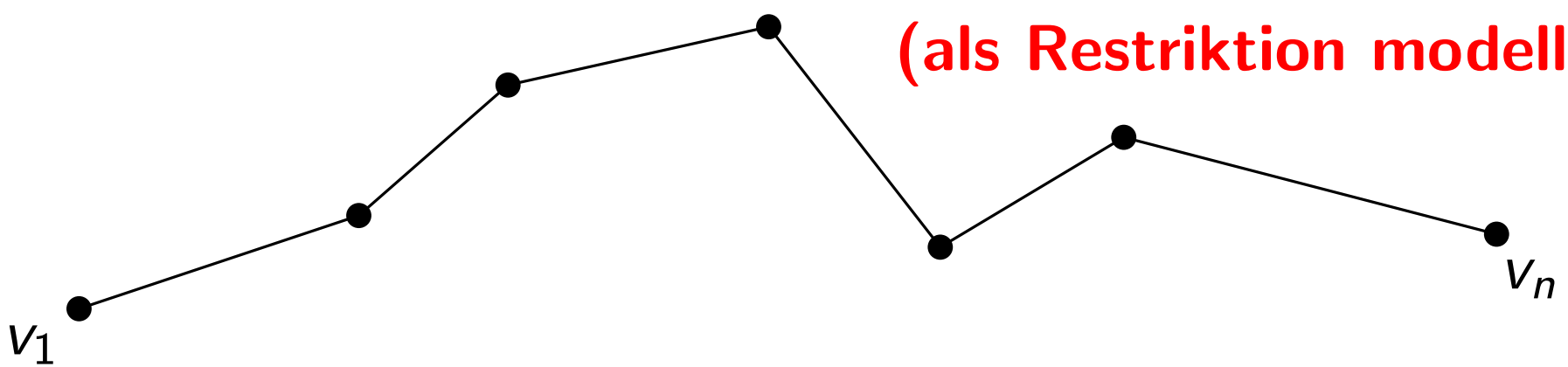
Linienvereinfachung

Problem (Imai & Iri, 1988)

- Gegeben:
  - Polylinie als Folge $V = \langle v_1, \ldots, v_n \rangle$ von Ecken
  - Fehlertoleranz $\varepsilon =$ ⟼
- Berechne möglichst kurze Teilfolge V' von V, so dass
  - V' die Ecken v_1 und v_n enthält
  - für jede Kante $e = (v_i, v_k)$ in V' und jede Ecke v_j mit $i < j < k$ gilt: $\text{Distanz}(e, v_j) \leq \varepsilon$.

Lesbarkeit (als Ziel modelliert)

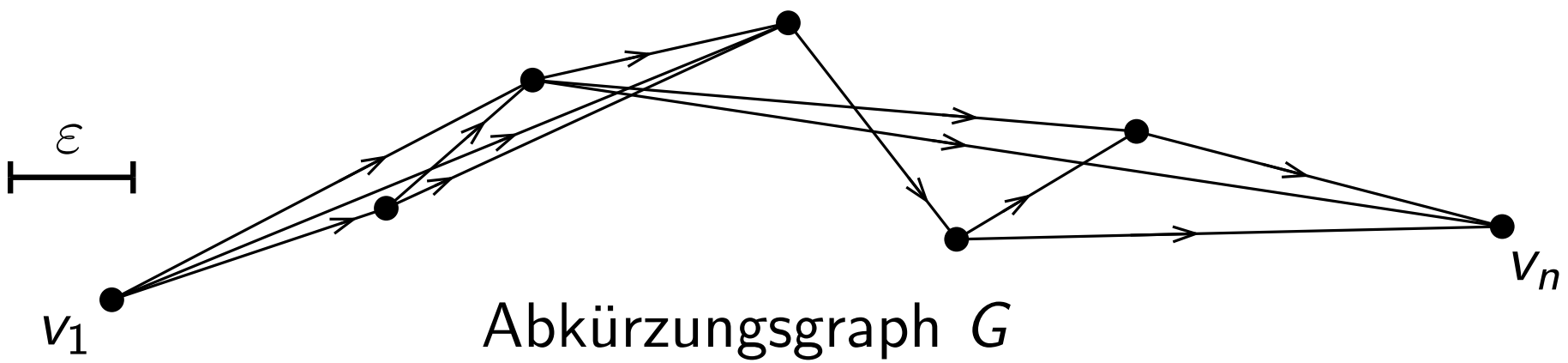
Ähnlichkeit (als Restriktion modelliert)



Linienvereinfachung

Algorithmus (Imai & Iri, 1988)

- Berechne alle zulässigen Kanten.

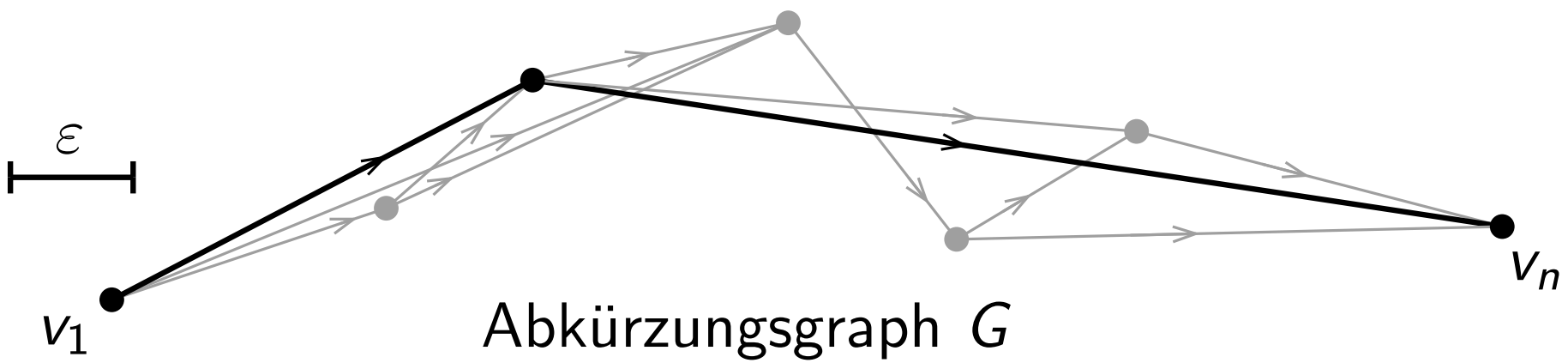


Abkürzungsgraph G

Linienvereinfachung

Algorithmus (Imai & Iri, 1988)

- Berechne alle zulässigen Kanten.
- Berechne kürzesten Weg in G von v_1 nach v_n.

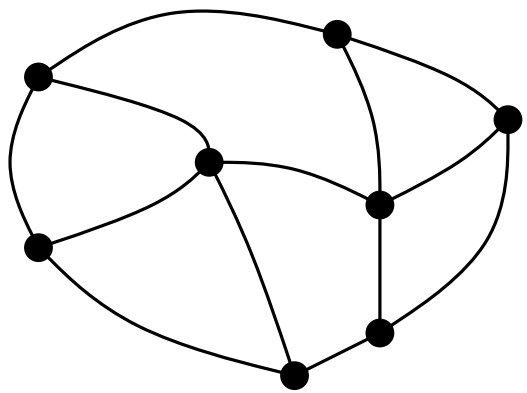


Abkürzungsgraph G

Selektion

Flüsse

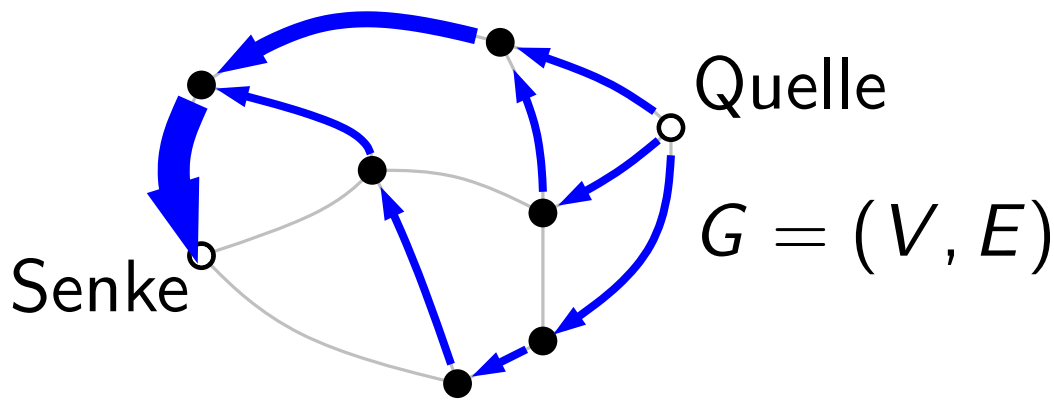
- Jede Kante kann einen Fluss tragen. Kapazität ist begrenzt.
- In jedem Knoten (außer Quelle & Senke) bleibt Fluss erhalten.



$G = (V, E)$

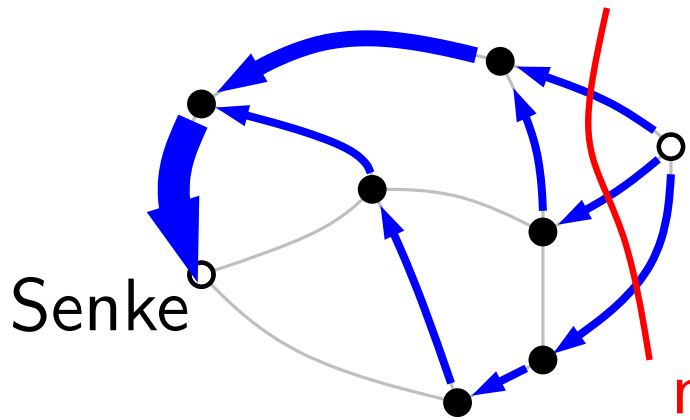
Flüsse

- Jede Kante kann einen Fluss tragen. Kapazität ist begrenzt.
- In jedem Knoten (außer Quelle & Senke) bleibt Fluss erhalten.



Flüsse

- Jede Kante kann einen Fluss tragen. Kapazität ist begrenzt.
- In jedem Knoten (außer Quelle & Senke) bleibt Fluss erhalten.

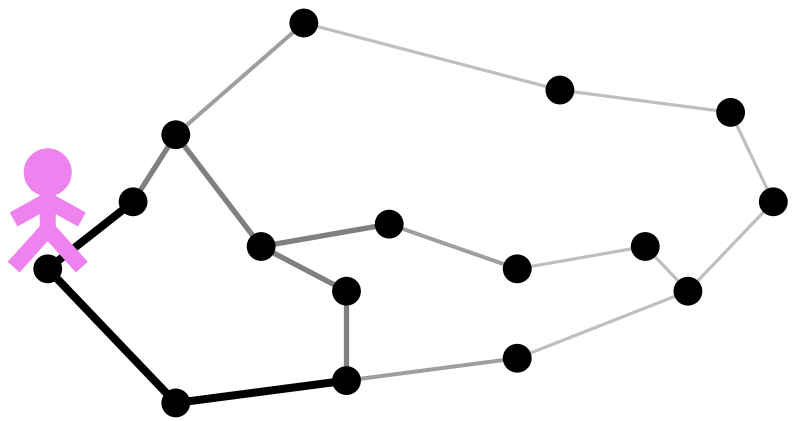


$G = (V, E)$

maximaler Fluss = minimaler Cut

Flüsse – Anwendung in der Generalisierung

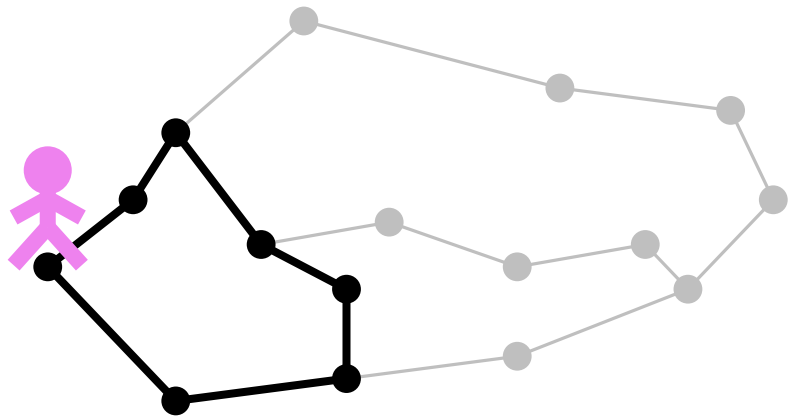
- Nutzer befindet sich in Knoten v von G (= Straßennetz).
- Jede Kante hat ein Gewicht (= Wichtigkeit der Darstellung).



T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

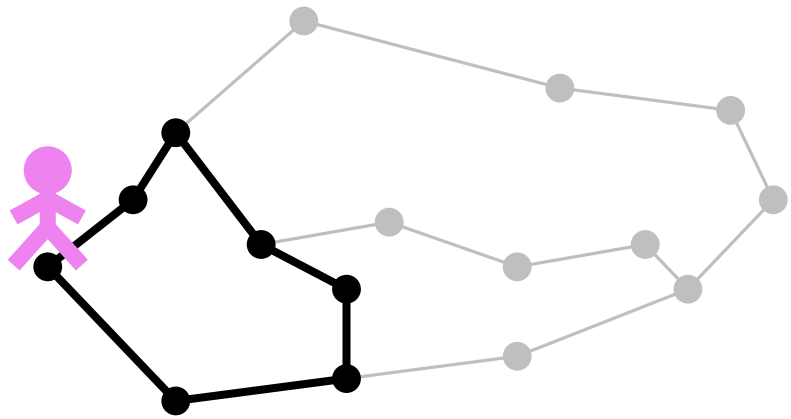
- Nutzer befindet sich in Knoten v von G (= Straßennetz).
- Jede Kante hat ein Gewicht (= Wichtigkeit der Darstellung).
- Wähle k Kanten aus, so dass Gesamtgewicht maximal ist.



T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

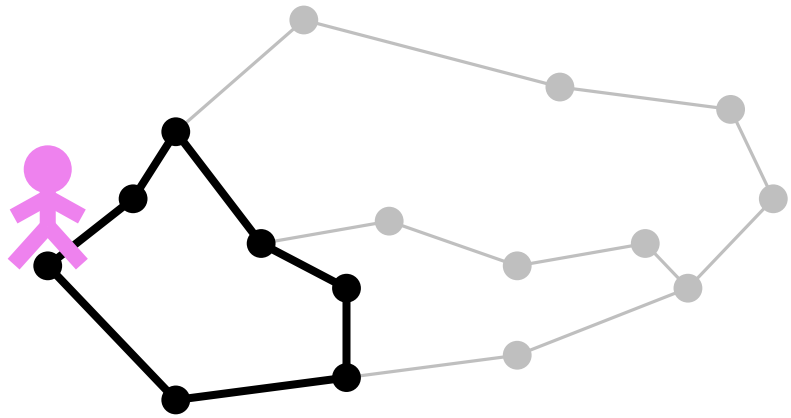
- Nutzer befindet sich in Knoten v von G (= Straßennetz).
- Jede Kante hat ein Gewicht (= Wichtigkeit der Darstellung).
- Wähle k Kanten aus, so dass Gesamtgewicht maximal ist.
  (ohne weitere Bedingungen trivial lösbar)



T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

- Nutzer befindet sich in Knoten v von G (= Straßennetz).
- Jede Kante hat ein Gewicht (= Wichtigkeit der Darstellung).
- Wähle k Kanten aus, so dass Gesamtgewicht maximal ist.
  (ohne weitere Bedingungen trivial lösbar)



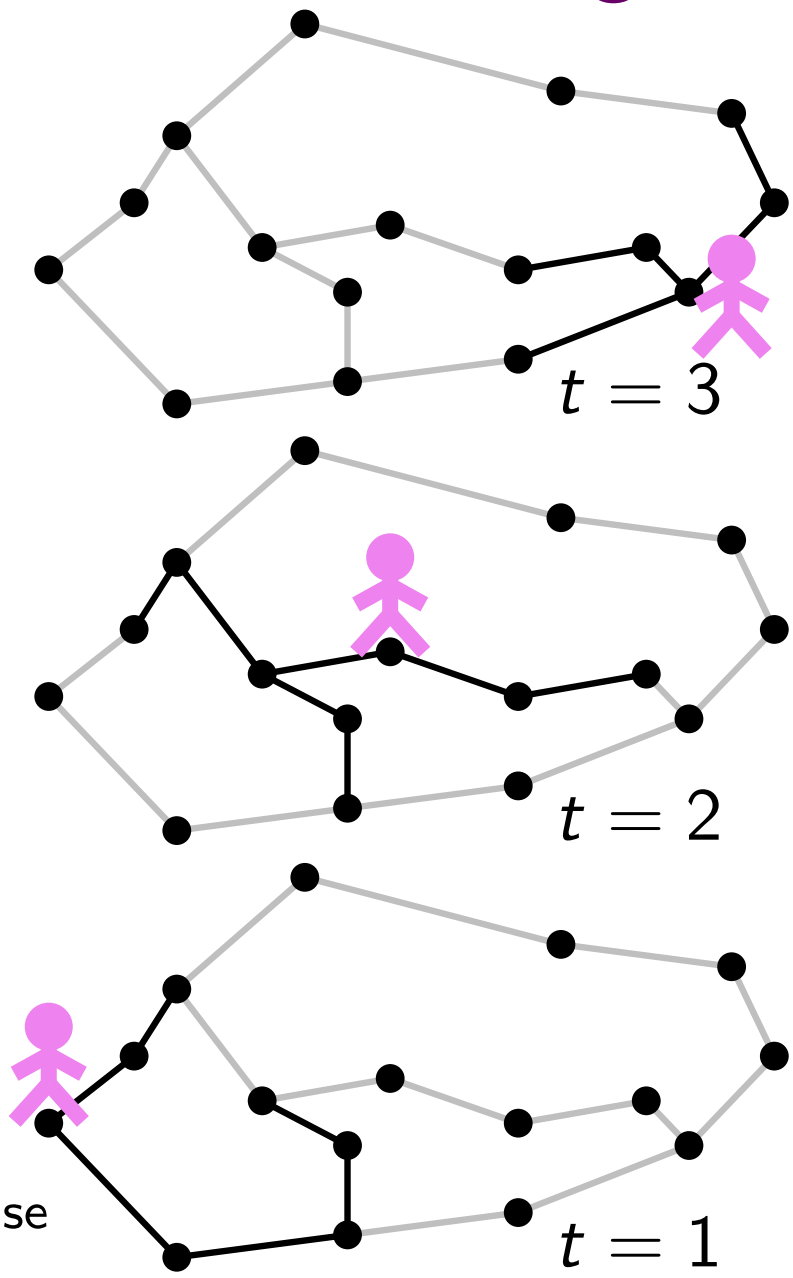
Dynamik behandeln!

T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

dynamische Variante:

- mehrere (Zeit-)Ebenen
- Gewichtung abhängig von Ebene



T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

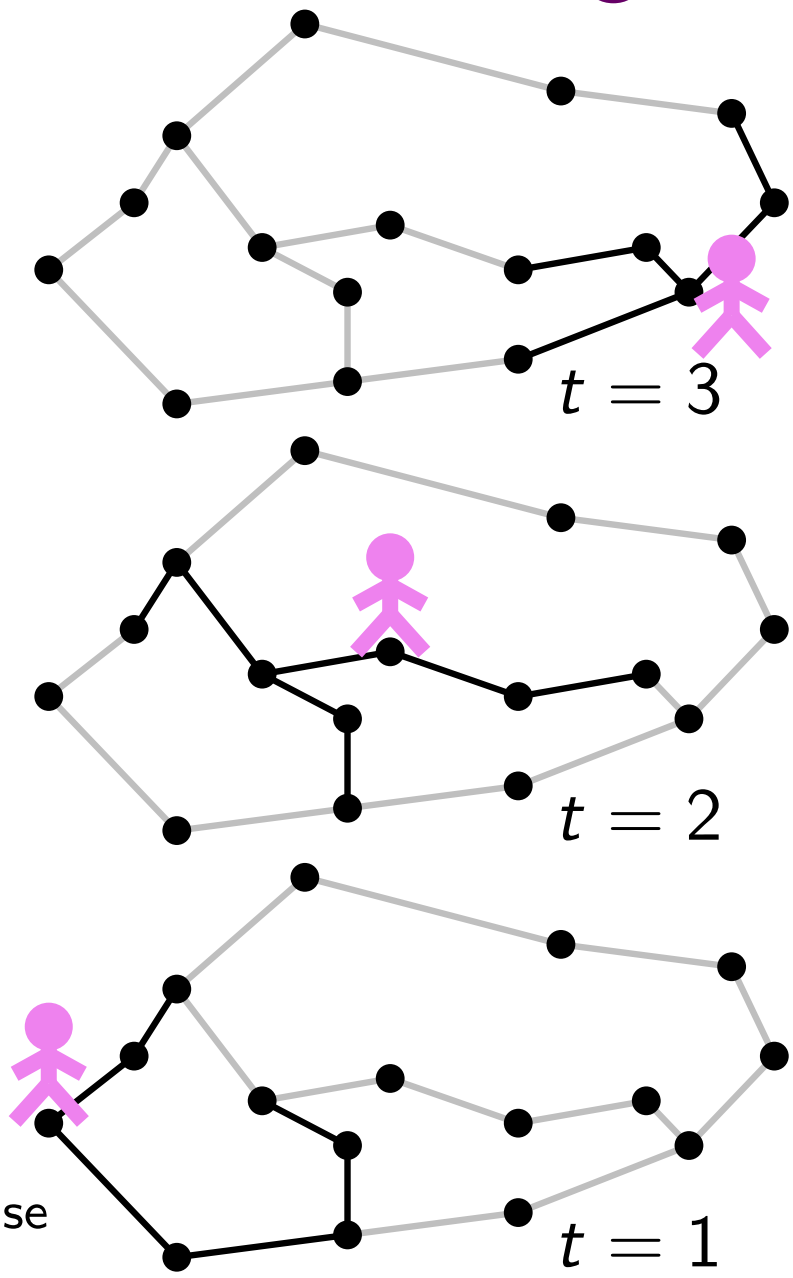
Flüsse – Anwendung in der Generalisierung

dynamische Variante:

- mehrere (Zeit-)Ebenen
- Gewichtung abhängig von Ebene

gesucht:

- pro Zeitebene k Kanten
- Maximierung Gesamtgewicht
- Minimierung Anzahl An/Aus-Änderungen



T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

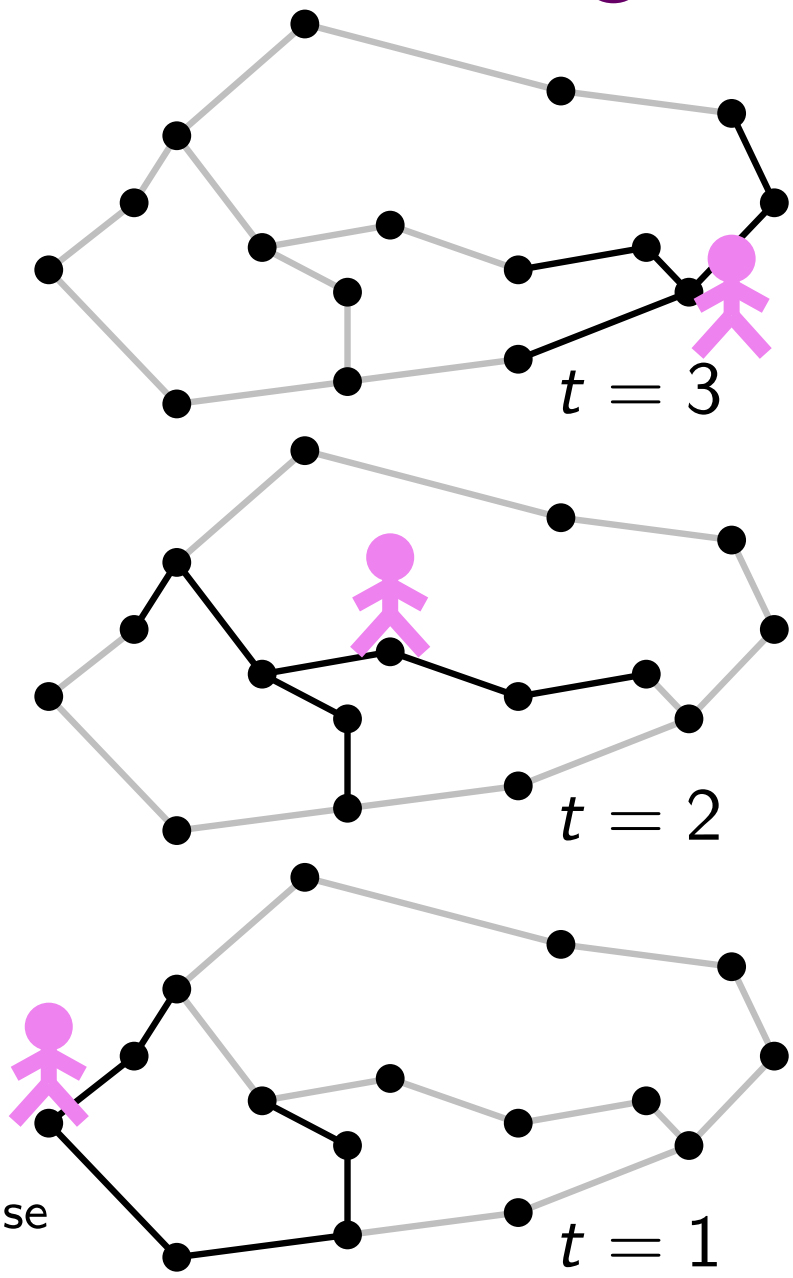
dynamische Variante:

- mehrere (Zeit-)Ebenen
- Gewichtung abhängig von Ebene

gesucht:

- pro Zeitebene k Kanten
- Maximierung Gesamtgewicht
- Minimierung Anzahl An/Aus-Änderungen

Min. gewichtete Summe

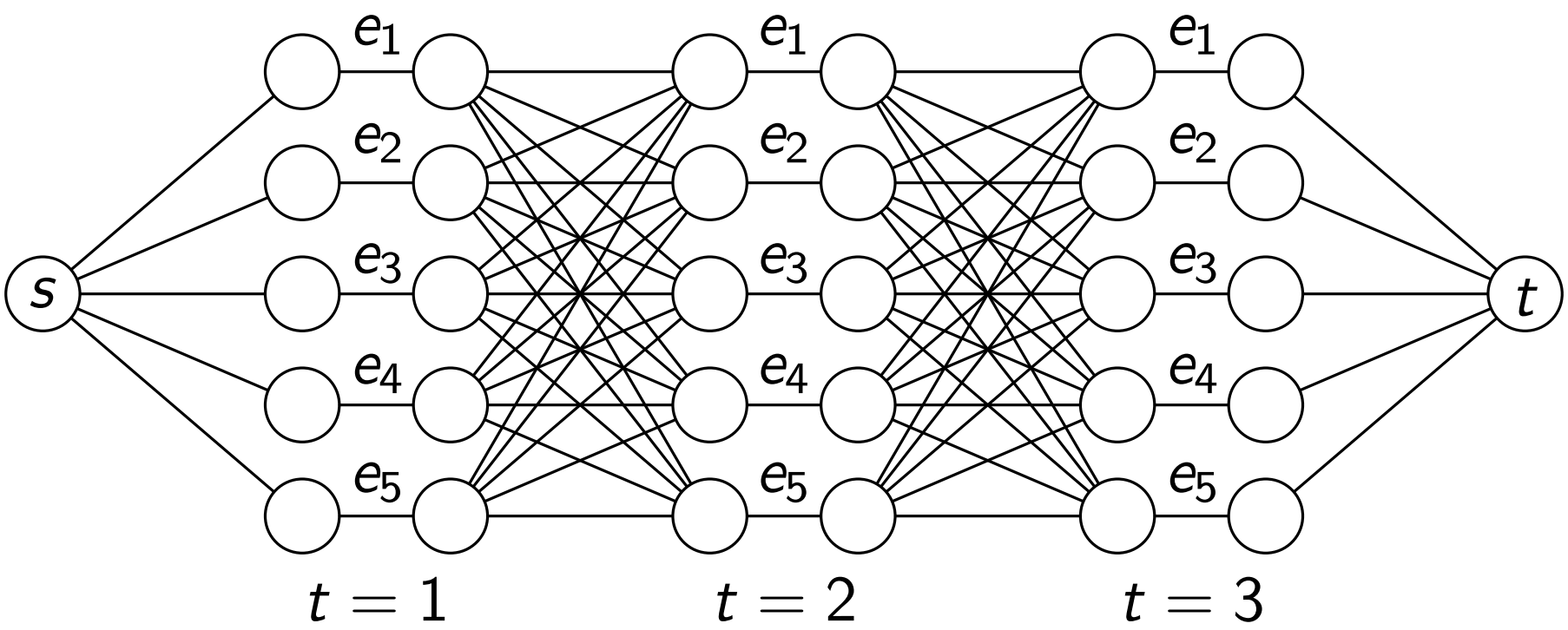


T. C. van Dijk, K. Fleszar, J.-H. Haunert, J. Spoerhase
Road segment selection with strokes and stability.
Proc. 1st ACM Workshop on Map Interaction, 2013.

Flüsse – Anwendung in der Generalisierung

Algorithmus:

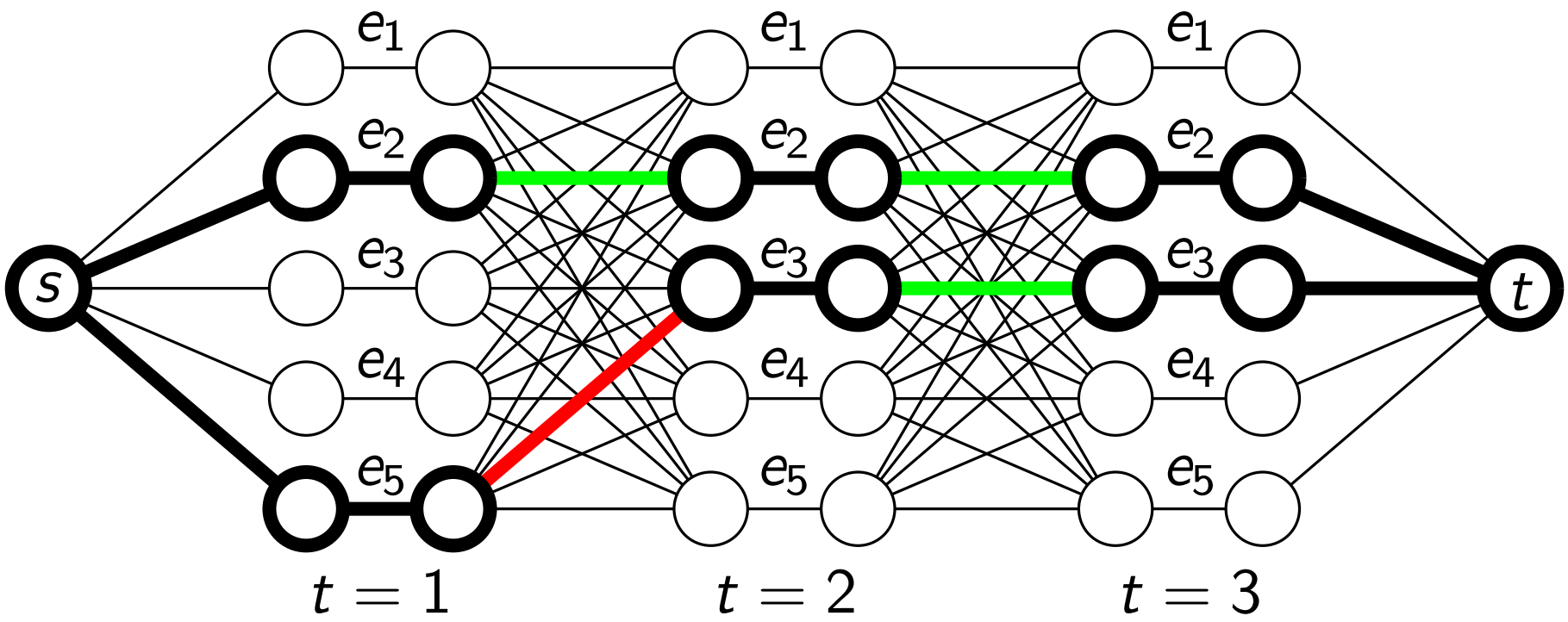
- Berechne Graph wie in Abbildung.
- Jede Kante hat Kapazität 1 und individuelle Kosten.



Flüsse – Anwendung in der Generalisierung

Algorithmus:

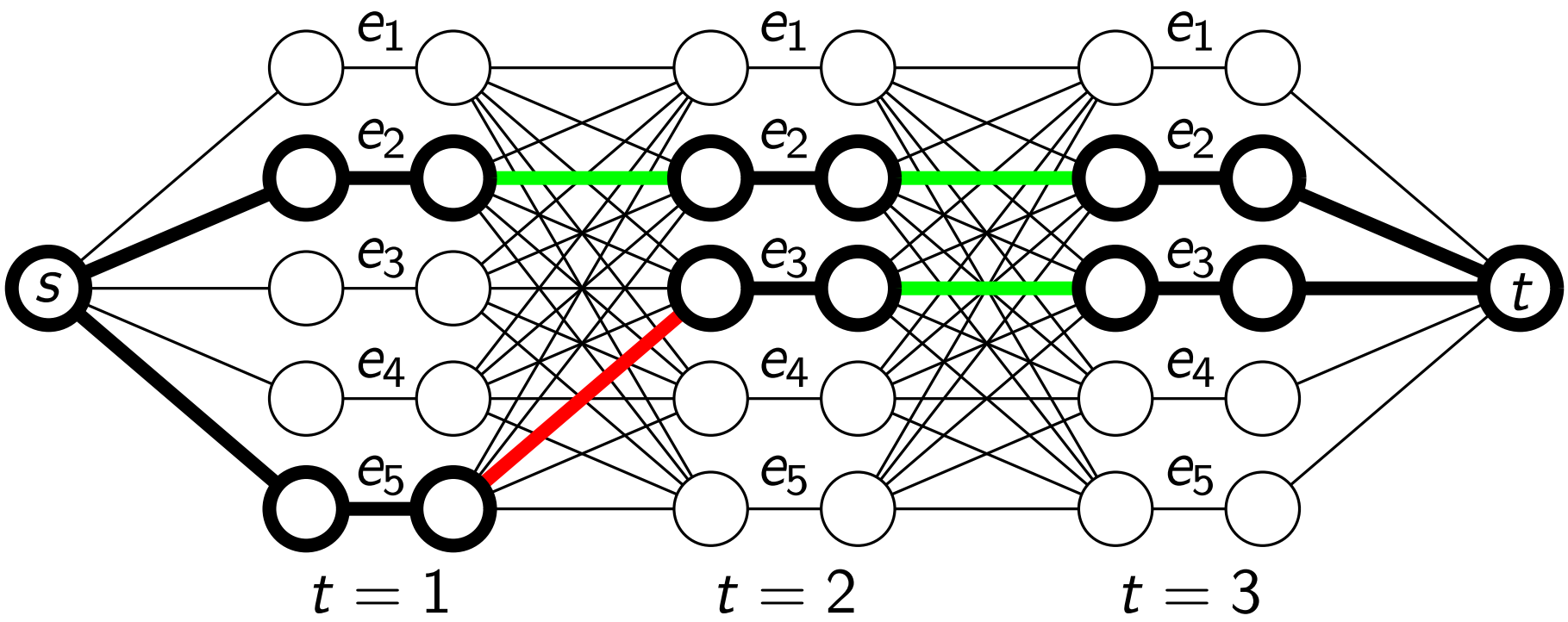
- Berechne Graph wie in Abbildung.
- Jede Kante hat Kapazität 1 und individuelle Kosten.
- Berechne Fluss mit Wert k und minimalen Kosten.



Flüsse – Anwendung in der Generalisierung

Algorithmus:

- Berechne Graph wie in Abbildung.
- Jede Kante hat Kapazität 1 und individuelle Kosten.
- Berechne Fluss mit Wert k und minimalen Kosten.

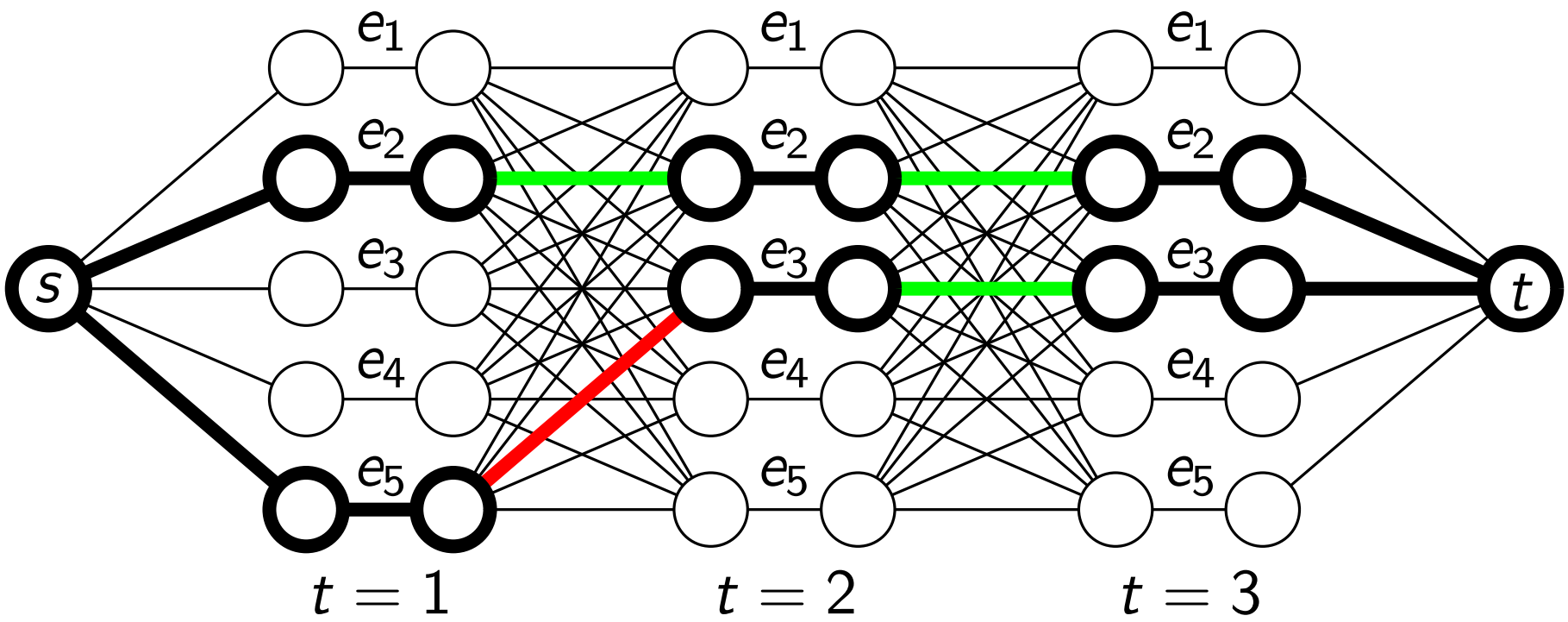


- Ergebnis: -10% Gewicht, dafür -84% an An/Aus-Events

Flüsse – Anwendung in der Generalisierung

Algorithmus:

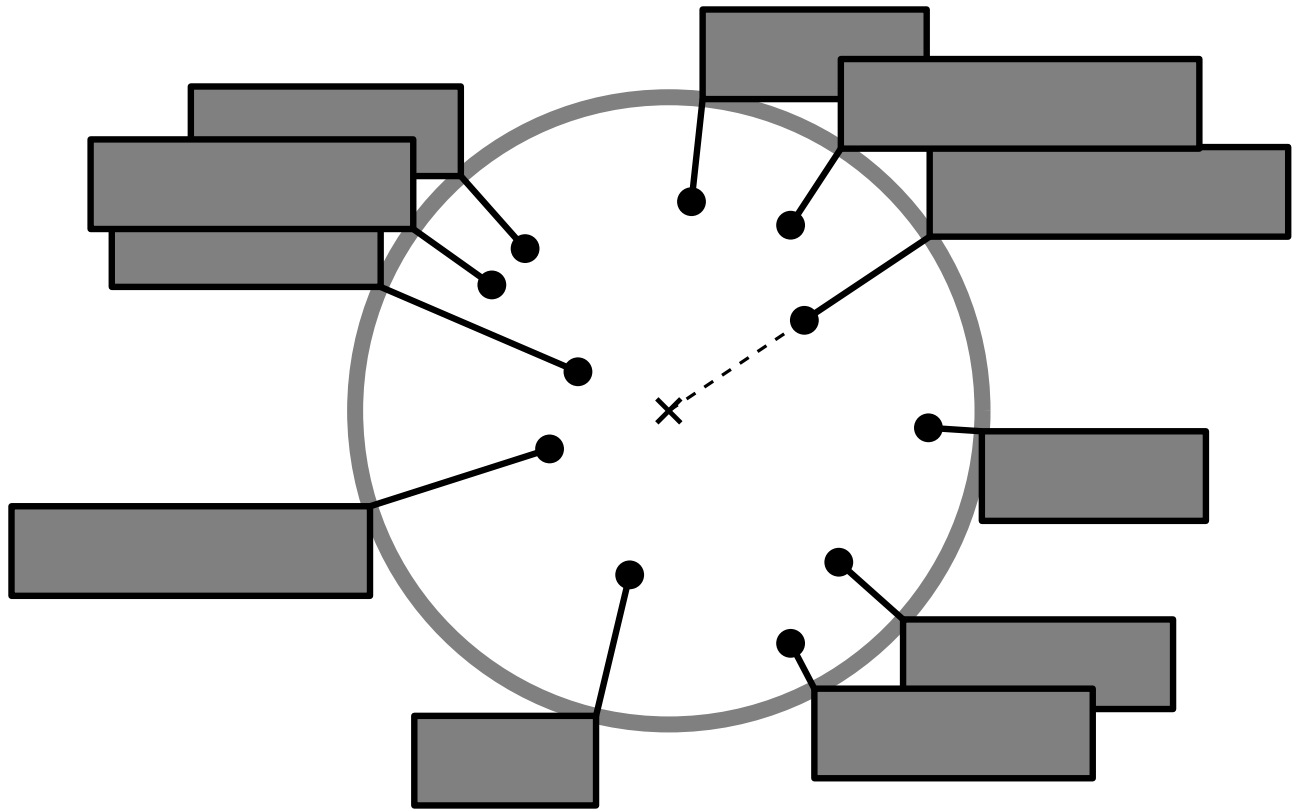
- Berechne Graph wie in Abbildung.
- Jede Kante hat Kapazität 1 und individuelle Kosten.
- Berechne Fluss mit Wert k und minimalen Kosten.



- Ergebnis: -10% Gewicht, dafür -84% an An/Aus-Events
- To Do: Restriktionen (z.B. topologischer Zusammenhang)

Aktuelle Forschung

- Vermeidung von sprunghaften Änderungen *und* Konflikten von Labels.



Fazit

- Viele kartographische Probleme lassen sich mit graphentheoretischen Konzepten/Algorithmen lösen.

Fazit

- Viele kartographische Probleme lassen sich mit graphentheoretischen Konzepten/Algorithmen lösen.
- Viele der Probleme sind NP-schwer, aber es lassen sich Heuristiken und effizient lösbare Spezialfälle (mit Relevanz für die Praxis!) finden.

Fazit

- Viele kartographische Probleme lassen sich mit graphentheoretischen Konzepten/Algorithmen lösen.
- Viele der Probleme sind NP-schwer, aber es lassen sich Heuristiken und effizient lösbare Spezialfälle (mit Relevanz für die Praxis!) finden.
- Wichtig ist die *Modellierung des Problems* – und *danach* die Suche nach geeigneten Algorithmen.

Fazit

- Viele kartographische Probleme lassen sich mit graphentheoretischen Konzepten/Algorithmen lösen.
- Viele der Probleme sind NP-schwer, aber es lassen sich Heuristiken und effizient lösbare Spezialfälle (mit Relevanz für die Praxis!) finden.
- Wichtig ist die *Modellierung des Problems* – und *danach* die Suche nach geeigneten Algorithmen.

- Geovisualisierung ist ein spannendes Forschungsfeld – auch für Informatiker!