

Algorithmen für Routenplanung

8. Vorlesung, Sommersemester 2017

Ben Strasser | 29. Mai 2014

INSTITUT FÜR THEORETISCHE INFORMATIK · ALGORITHMIK



Heute:

- keine Routenplanung
- Womit sind die gesehenen Algorithmen verwandt?

Nächste Woche:

- Wieder Routenplanung

- Große quadratische Gleichungssysteme Lösen hat viele Anwendungen
- Schnelles Lösen ist wichtig
- Algorithmus von Gauß in $O(n^3)$ wobei n die Anzahl der Variablen ist
 - Oft zu langsam

- Oft sind Gleichungssysteme dünnbesetzt
 - d. h. viele Koeffizienten sind 0
- Idee: Speichere 0-Koeffizienten nicht ab
- Aber: Wie 0-Koeffizienten während des Lösens erhalten?

Ziel: Obere Dreiecksmatrix per Gaußelimination

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & -2 \\ 1 & 0 & -1 & -1 & 0 \\ -1 & 0 & -1 & -2 & 0 \\ 1 & -1 & 0 & 0 & 1 \end{bmatrix}$$

4 Nullen im oberen Dreieck

Ziel: Obere Dreiecksmatrix per Gaußelimination

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 1 & -2 & -2 & -1 \\ 0 & -1 & 0 & -1 & 1 \\ 0 & 0 & -1 & -1 & 0 \end{bmatrix}$$

Ziel: Obere Dreiecksmatrix per Gaußelimination

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 0 & -\frac{3}{2} & -\frac{3}{2} & \frac{1}{2} \\ 0 & 0 & -\frac{1}{2} & -\frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & -1 & -1 & 0 \end{bmatrix}$$

Ziel: Obere Dreiecksmatrix per Gaußelimination

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 0 & -\frac{3}{2} & -\frac{3}{2} & 1 \\ 0 & 0 & 0 & -1 & -\frac{3}{2} \\ 0 & 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

Keine Nullen übrig $\rightarrow :($

Idee: Spalten und Zeilen umsortieren

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & -2 \\ 1 & 0 & -1 & -1 & 0 \\ -1 & 0 & -1 & -2 & 0 \\ 1 & -1 & 0 & 0 & 1 \end{bmatrix}$$

Idee: Spalten und Zeilen umsortieren

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ -2 & 1 & 0 & 0 & 1 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & -2 & -1 \\ 1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

Idee: Spalten und Zeilen umsortieren

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & -2 & -1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Idee: Spalten und Zeilen umsortieren

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Idee: Spalten und Zeilen umsortieren

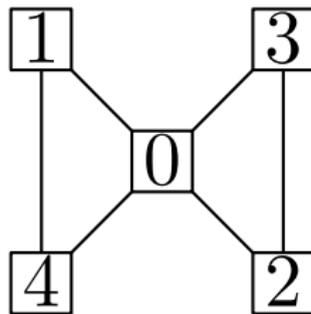
$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Alle 4 Nullen erhalten \rightarrow :)

Warum funktioniert das?

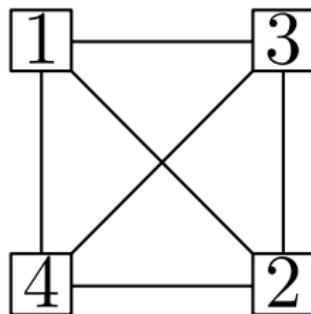
Jeder Eintrag der nicht Null ist entspricht einer Kante.

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & -2 \\ 1 & 0 & -1 & -1 & 0 \\ -1 & 0 & -1 & -2 & 0 \\ 1 & -1 & 0 & 0 & 1 \end{bmatrix}$$



Jeder Eintrag der nicht Null ist entspricht einer Kante.

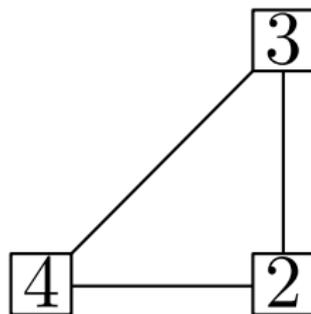
$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 1 & -2 & -2 & -1 \\ 0 & -1 & 0 & -1 & 1 \\ 0 & 0 & -1 & -1 & 0 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

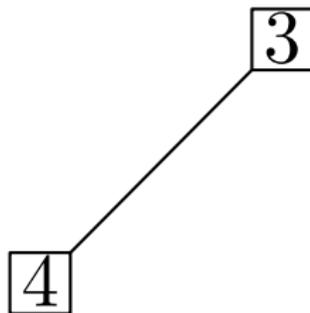
$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 0 & -\frac{3}{2} & -\frac{3}{2} & \frac{1}{2} \\ 0 & 0 & -\frac{1}{2} & -\frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & -1 & -1 & 0 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 0 & -\frac{3}{2} & -\frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & 0 & -1 & -\frac{1}{2} \\ 0 & 0 & 0 & 0 & -\frac{3}{2} \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

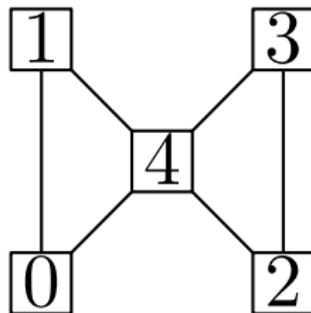
$$\begin{bmatrix} 1 & -1 & 1 & 1 & 1 \\ 0 & 2 & -1 & -1 & -3 \\ 0 & 0 & -\frac{3}{2} & -\frac{3}{2} & -\frac{1}{2} \\ 0 & 0 & 0 & -1 & -\frac{3}{2} \\ 0 & 0 & 0 & 0 & -\frac{3}{2} \end{bmatrix}$$

4

Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

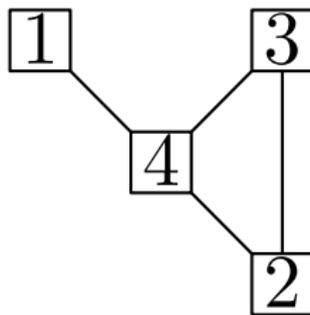
$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ -2 & 1 & 0 & 0 & 1 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & -2 & -1 \\ 1 & -1 & 1 & 1 & 1 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

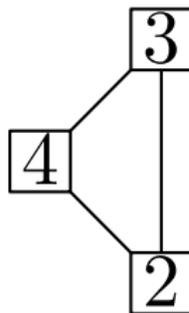
$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & -2 & -1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

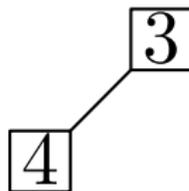
$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & -2 & -1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

Jeder Eintrag der nicht Null ist entspricht einer Kante.

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 3 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

4

Variablenelimination \leftrightarrow Knotenkontraktion
Jeder Shortcut zerstört eine Null

- Wenig Kanten im CCH-Graph \leftrightarrow viele 0-Koeffizienten in Matrix
- Wie hängt dies genau zusammen?
 - Baumzerlegung

Baumzerlegungen

- Sehr breites Themenfeld
- Schwer zu überblicken
- Sehr viel Literatur

Anwendungen

- Routenplanung
- Gleichungssysteme lösen
- Fixed Parameter Tractability (FPT) Algorithmen für NP-schwere Probleme
 - z. B. Vertex-Cover, 3-Color, Independent Set
 - Mehr dazu später

Eine Baumzerlegung (B, T) eines ungerichteten ungewichteten Graphen (V, E) besteht aus

- Menge von Bags
 - Jeder Bag ist eine Knotenmenge
- Backbone T
 - Graph mit Bags als Knoten
 - Bei Baumzerlegung ist T ein Baum
 - (Analog: Bei Pfadzerlegung ist T ein Pfad)

Eine Baumzerlegung (B, T) eines ungerichteten ungewichteten Graphen (V, E) besteht aus

- Menge von Bags
 - Jeder Bag ist eine Knotenmenge
- Backbone T
 - Graph mit Bags als Knoten
 - Bei Baumzerlegung ist T ein Baum
 - (Analog: Bei Pfadzerlegung ist T ein Pfad)

- T_x ist Teilgraph von T
- T_x wird induziert durch Bags die Knoten x enthalten

Jede Baumzerlegung muss drei Bedingungen erfüllen

- Jeder Knoten ist in einem oder mehreren Bags

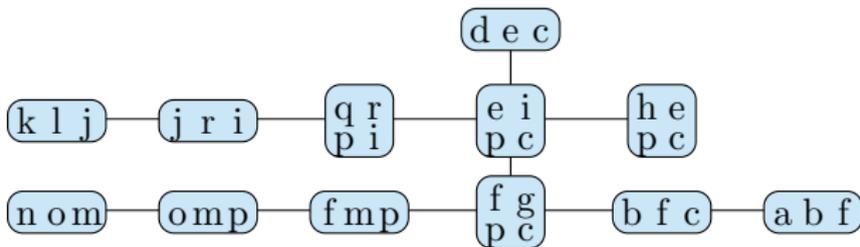
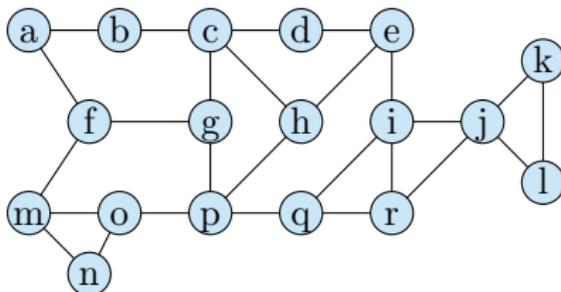
$$\bigcup_{b \in B} b = V$$

- Jede Kante ist in einem Bag

$$\forall \{x, y\} \in E : \exists b \in B : x \in b \wedge y \in b$$

- Für jeden Knoten x ist T_x ein Baum

Beispiel



- Breite einer Baumzerlegung ist maximale Baggröße minus 1
- Baumbreite tw eines Graphs ist minimale Breite einer Baumzerlegung

- Problem ist FPT in der Baumbreite wenn es Algorithmus gibt mit Laufzeit:

$$f(tw) \cdot p(n)$$

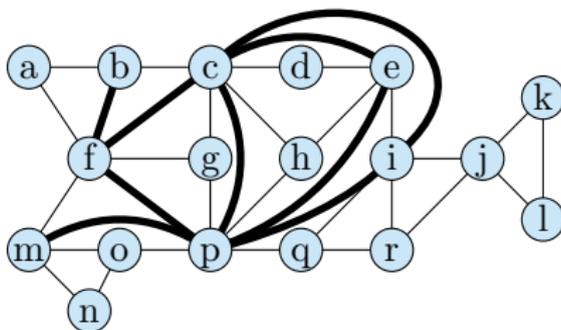
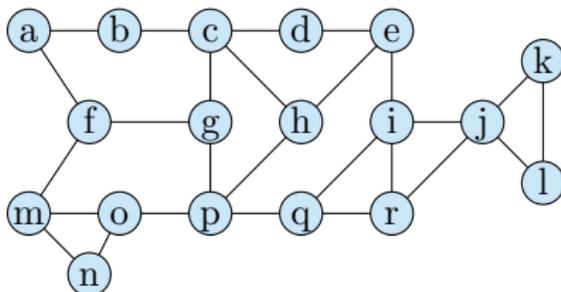
- tw ist Baumbreite, n ist Knotenanzahl
- f beliebige Funktion die nicht von n abhängt
- p beliebiges Polynom

Beispiel

- Vertex-Cover in Zeit $2^{tw} \cdot m$ lösbar

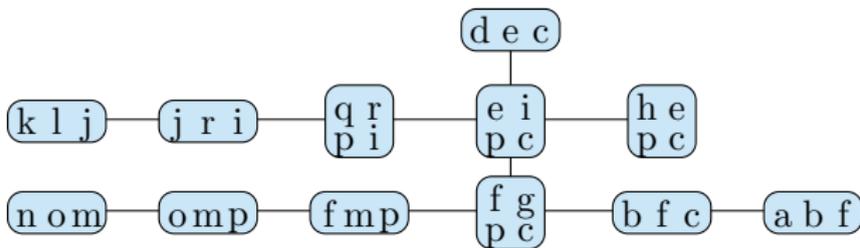
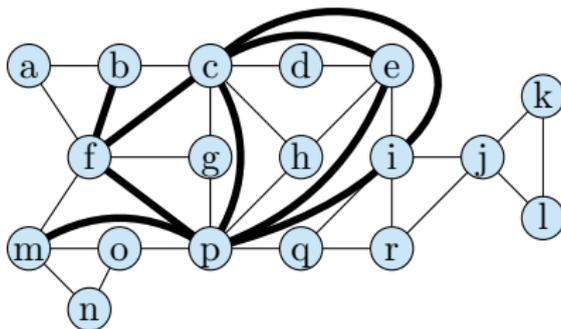
- Maximale Cliques in CCH-Graph sind Bags einer Baumzerlegung

Schritt 1: CCH-Graph bauen



Knotenordnung: k, l, a, b, d, n, j, o, m, f, g, h, e, q, r, i, p, c

Schritt 2: Maximale Cliques



Terminologie

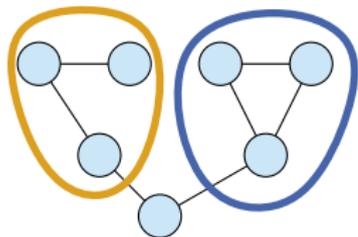
- Im Baumzerlegungskontext verwendet man folgende Begriffe:
 - Kontraktionsordnung \leftrightarrow (perfect) elimination order
 - CCH-Graph \leftrightarrow chordaler Supergraph

- Multilevel Partition P ist Menge von Zellen
- Jede Zelle z ist Menge von Knoten

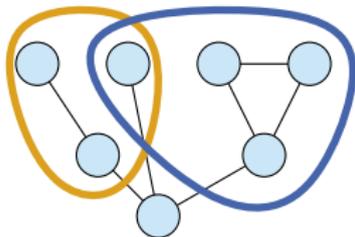
Wir fordern:

- Es gibt immer eine top-level Zelle die alle Knoten enthält
- Zellen die sich “berühren” sind verschachtelt

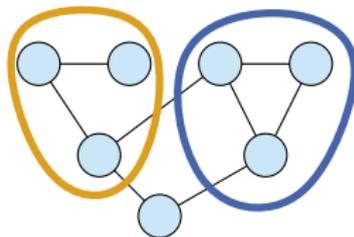
Wann berühren sich Zellen?



Keine Berührung



Berührung



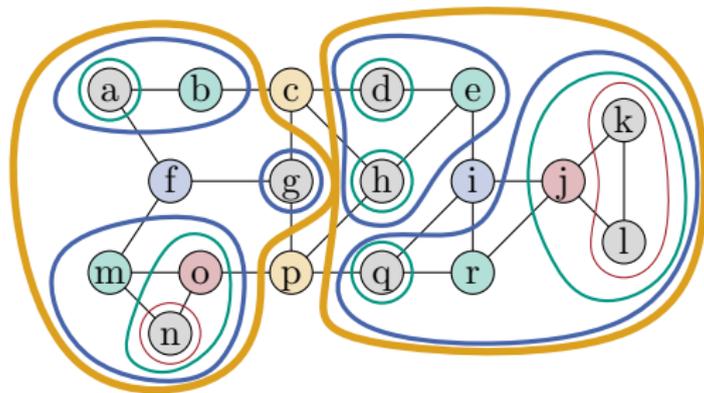
Berührung

Zwei Zellen berühren sich wenn sie

- einen gemeinsamen Knoten haben oder
- es eine Kante zwischen den Zellen gibt.

Warnung

- Zellen werden hier per Knotenseparator getrennt
- Bei MLD üblicherweise aber per Kantenschnitte
- Knotenseparatoren passen besser mit Baumzerlegung zusammen



- Kreise sind Zellen
- Toplevel Zelle nicht eingezeichnet

Zellenrand R einer Zelle Z sind Knoten die

- nicht in Z sind
- aber es eine Kante gibt zwischen Knoten in Z und in R

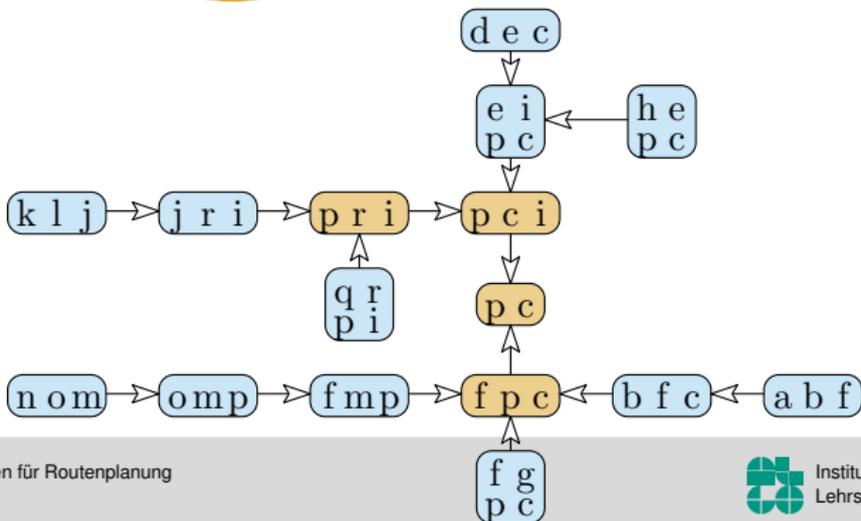
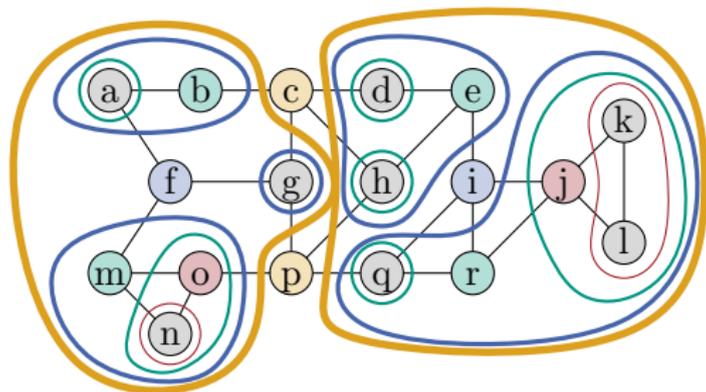
Zusammenhang zu Multilevel Partition

Zusammenhang

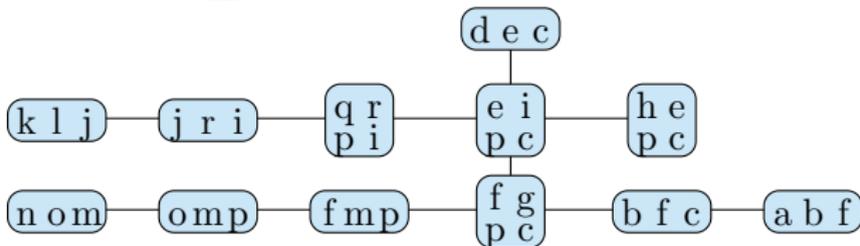
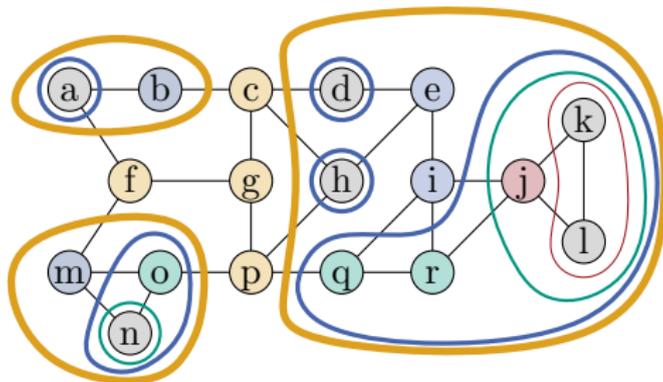
- Multilevel Partition entspricht gewurzelter Baumzerlegung
- Zellen entsprechen Bags
- Kind-Eltern-Relation entspricht Treebackbone

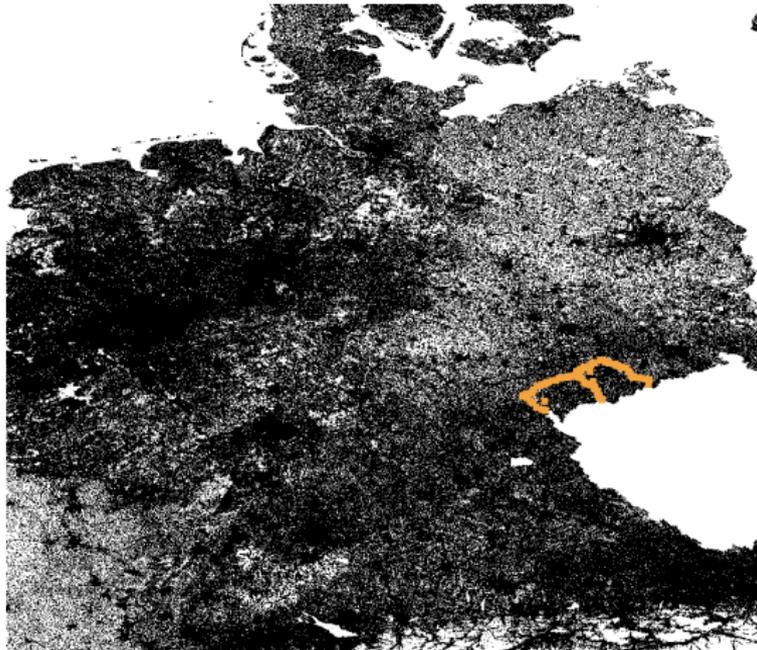
- Bag ist Vereinigung von
 - Zellenrand und Zell
 - ohne das Innere der Kinderzellen

Beispiel

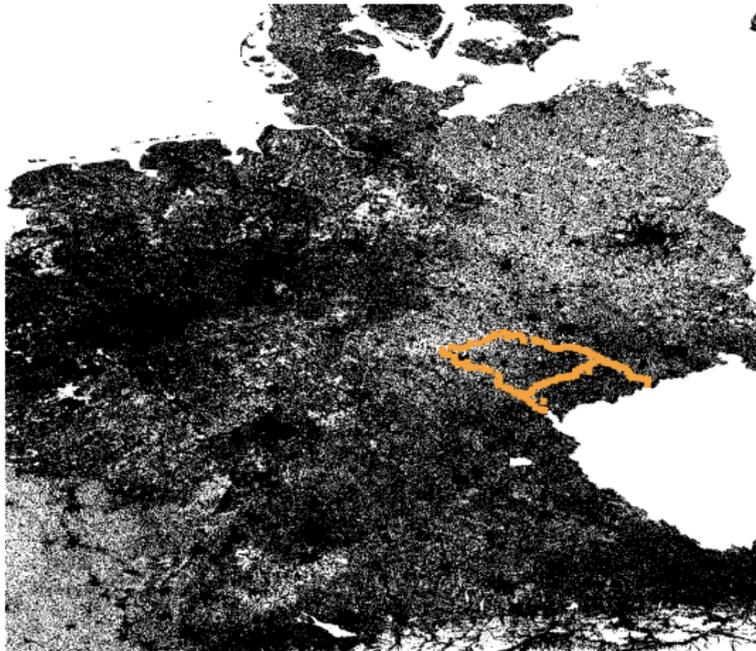


Beispiel

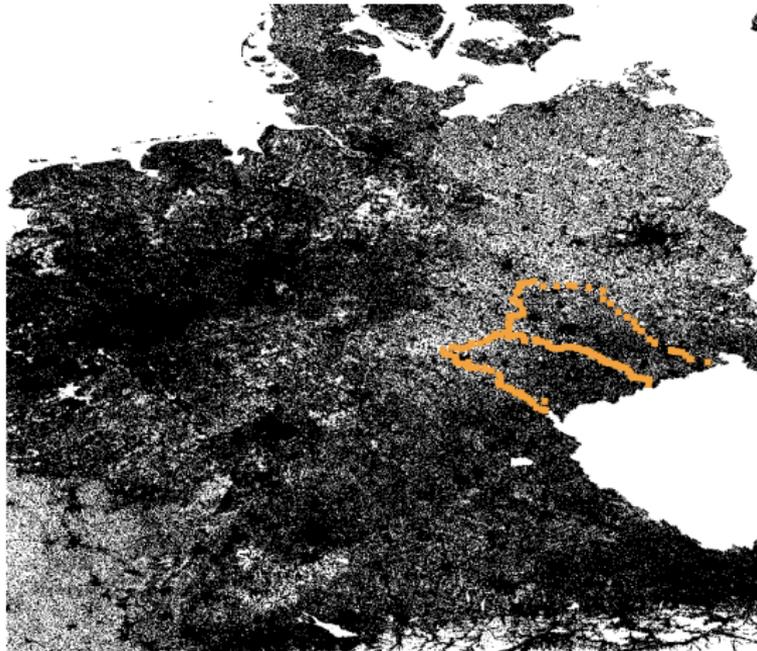




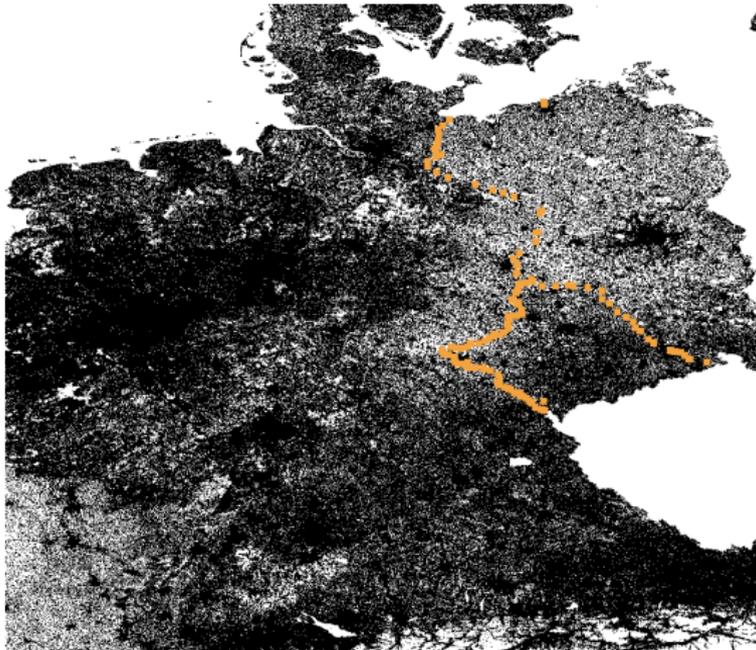
- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags



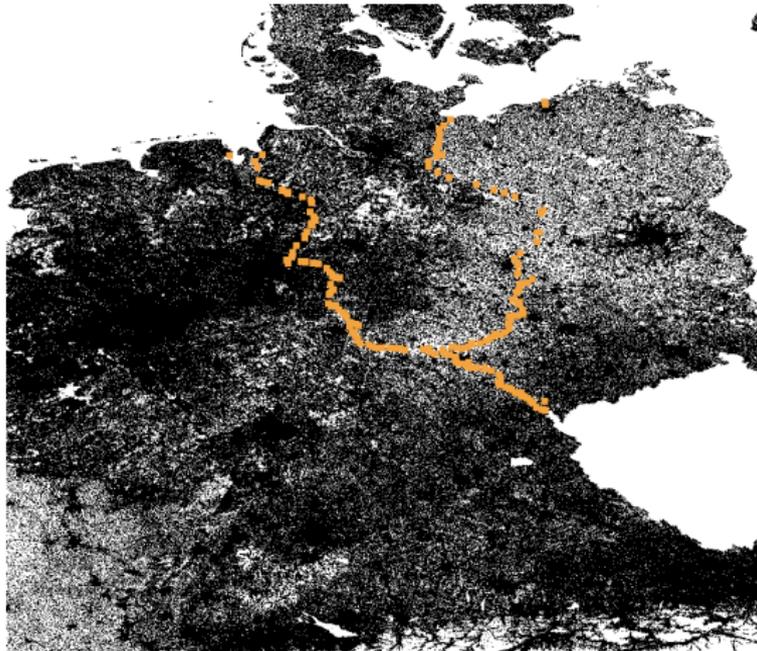
- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags



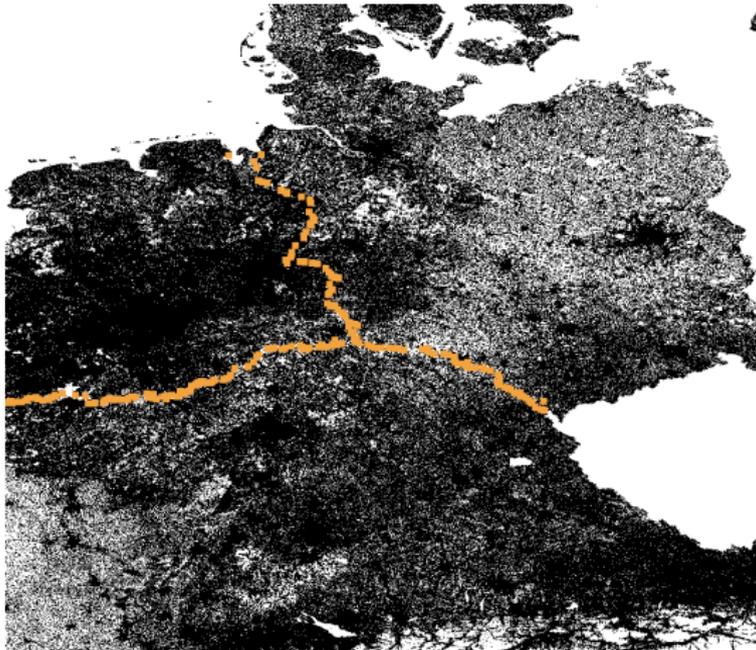
- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags



- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags



- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags



- orange Knoten sind in einem Bag
- Beispiel ist ein Pfad im Treebackbone von 6 Bags

Graph	Upper Treewidth Bound
DIMACS Colorado	87
DIMACS Kalifornien & Nevada	127
DIMACS Europa	449
DIMACS USA	311
OSM London	84
OSM Baden-Württemberg	107
OSM Deutschland	220

Eingabe:

- einfacher Graph mit Knoten und Kanten
- ungerichtet und ungewichtet

Ausgabe:

- Knotenmenge C
- Für jede Kante $\{x, y\}$ muss $x \in C$ oder $y \in C$ oder beide sein

Zielfunktion:

- C soll möglichst klein sein

Eigenschaften:

- klassisches NP-schweres Problem
- ist FPT in der Baumbreite
- Laufzeit $2^{tw} \cdot m$
- **Achtung:** Die Baumbreite von Straßengraphen ist zu groß für FPT Algorithmen

- Schritt 1: Find Baumzerlegung
- Schritt 2: Wähle irgendeinen Knoten als Wurzel aus

Idee:

- Speichere für jeden Bag b eine Tabelle mit $2^{|b|}$ Zeilen
- Tabelle enthält Zeile für jede Auswahl an Knoten aus b
- Jede Zeile enthält für jeden Knoten x aus b ein Bit ob $x \in C$
- Jede Zeile enthält Größe eines kleinsten Vertex-Covers
 - Nicht der ganze Graph wird überdeckt
 - Nur durch von Knoten im Teilbaum von b induzierter Subgraph überdeckt
 - Zellsichtweise: Graph induziert durch innere und Randknoten überdeckt
- Falls nicht möglich, dann Größe ∞

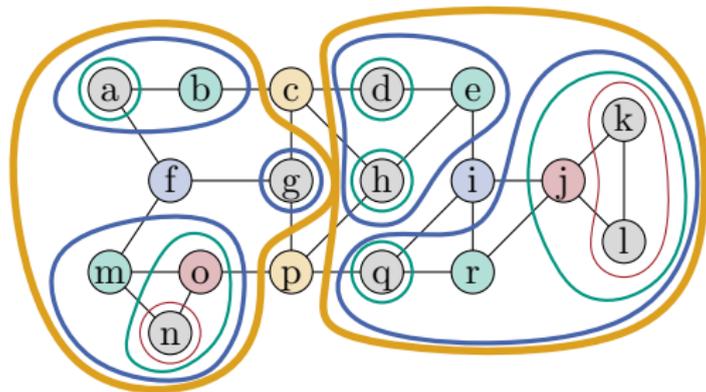


Tabelle für Bag $\{f, p, c\}$

$f \in C$	$p \in C$	$c \in C$	Größe	C (nicht gespeichert)
○	○	○	4	a, b, g, o
○	○	●	3	c, a, o
○	●	○	3	p, m, b
○	●	●	4	p, c, a, m
...				

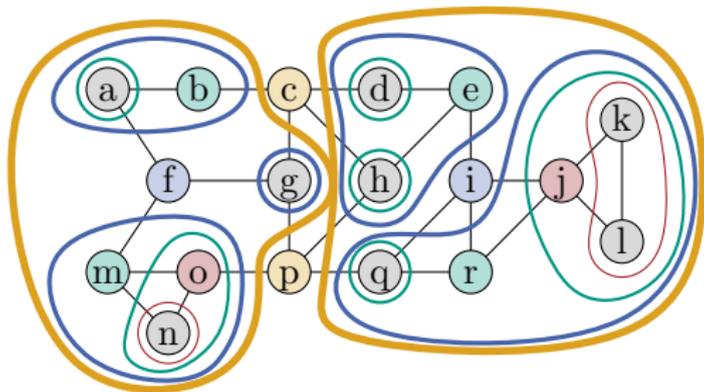


Tabelle für Bag $\{f, p, c\}$

$f \in C$	$p \in C$	$c \in C$	Größe	C (nicht gespeichert)
			...	
●	○	○	4	f, b, o, g
●	○	●	3	f, c, o
●	●	○	4	f, p, n, b
●	●	●	4	f, p, c, n

- Bottom-Up
 - Berechne erst die Tabellen der Blätter
 - Dann die Zwischenknoten
 - Schlussendlich die Tabelle der Wurzel
-
- Tabellen der Blätter werden mit durchprobieren berechnet
 - Andere Bags verwenden Tabellen ihrer Kinder
 - Kleinste Größe in Tabelle der Wurzel ist Endergebniss



Hans L. Bodlaender.
A tourist guide through treewidth.
Acta Cybernetica, 11:1–21, 1993.



Hans L. Bodlaender.
Treewidth: Structure and algorithms.
In *Proceedings of the 14th International Colloquium on Structural Information and Communication Complexity*, volume 4474 of *Lecture Notes in Computer Science*, pages 11–25. Springer, 2007.



Jean Blair and Barry Peyton.
An introduction to chordal graphs and clique trees.
In *Graph Theory and Sparse Matrix Computation*, volume 56 of *The IMA Volumes in Mathematics and its Applications*, pages 1–29. Springer, 1993.