



Technische
Universität
Braunschweig



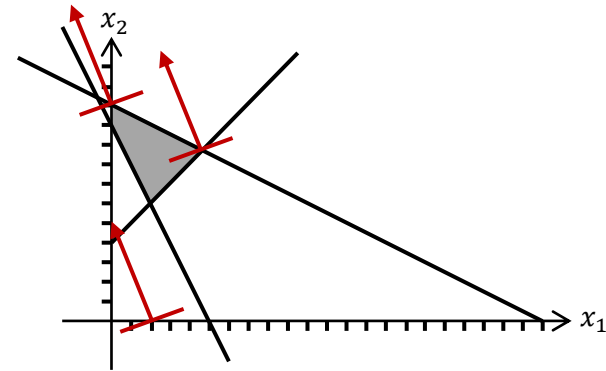
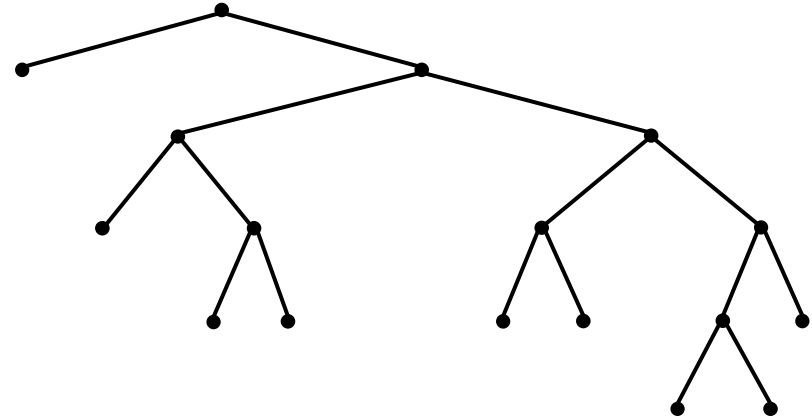
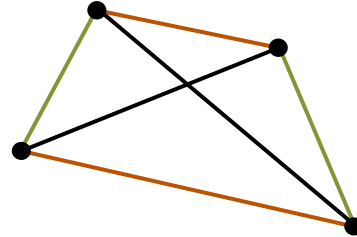
Algorithmen und Datenstrukturen 2 – Übung #3

Matthias Konitzny

01.06.2022

Heute

- Branch-and-Bound für Knapsack
- Euklidisches TSP
 - Definition
 - Schranken
 - Branch-and-Bound
 - Linear Programming



Branch-and-Bound

Allgemein:

1. Test auf Zulässigkeit

$$\sum_{i=1}^{l-1} b_i z_i \leq Z$$

2. Schranken berechnen

$$L = LB(I, b_1, \dots, b_{\ell-1})$$
$$U = UB(I, b_1, \dots, b_{\ell-1})$$

3. Test auf bessere Lösung

$$L > P?$$

4. Verzweigen falls nötig

1. Teste $b_l := 0$
2. Teste $b_l := 1$

Herausforderung

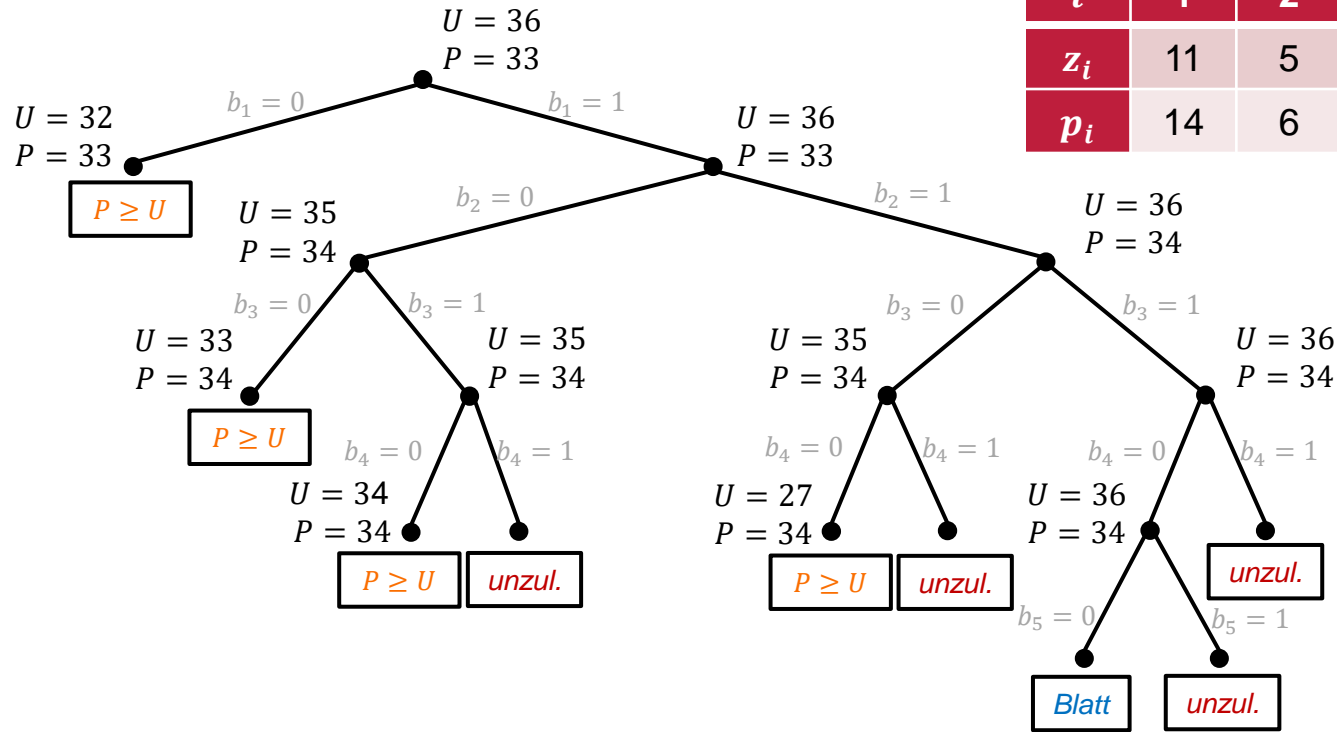
Finde gute Schranken, d.h. finde **kleine obere** und **große untere** Schranken.

MAXIMUM
KNAPSACK

Branch-and-Bound – Beispiel

i	1	2	3	4	5
z_i	11	5	13	18	9
p_i	14	6	13	16	7

$Z = 33$



Menge	Wert

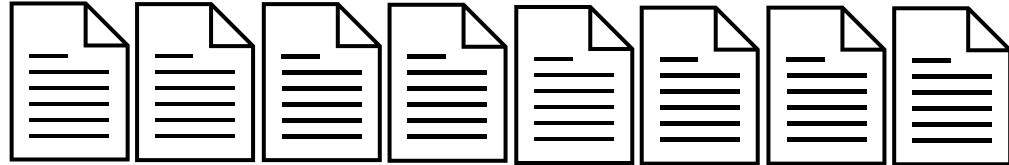
Branch-and-Bound – Laufzeit

Satz 3.2. Algorithmus [3.1](#) (als rekursiv arbeitende Unterroutine) berechnet eine Optimale Lösung für das Knapsackproblem in einer Worst-Case-Laufzeit $O(2^n f(n))$, wobei $f(n)$ die Zeit für die Berechnung der Schranken ist.

In unserem Fall: $f(n) \in O(n \log n)$

2^n ist hierbei der schlimmste Teil!

Achtung:
Jetzt ist euer
Bauchgefühl gefragt!



2^{100} : 100 mal verdoppeln und stapeln. Wie hoch ist der Papierstapel?

Dicke von Papier: $\sim 0.1\text{mm}$

2^{100} Blätter gestapelt



6.371 km



384.400 km

Distanz Sonne-Erde: 150 mio. km

Distanz Sonne-Pluto: 6 mrd. km

Distanz zu Alpha Centauri: 4,2 Lichtjahre

Distanz zum Andromeda Nebel: 2,5 mio. Lichtjahre

Distanz zum Virgohaufen: 60 mio. Lichtjahre

Wie hoch ist nun der Stapel mit 2^{100} Blättern?

Die Antwort: Etwa 14 mrd. Lichtjahre $\approx 10^{23}$ km

Euclidean Traveling Salesman Problem

Gegeben:

Punkte $p_1, \dots, p_n \in \mathbb{R}^2$

Gesucht:

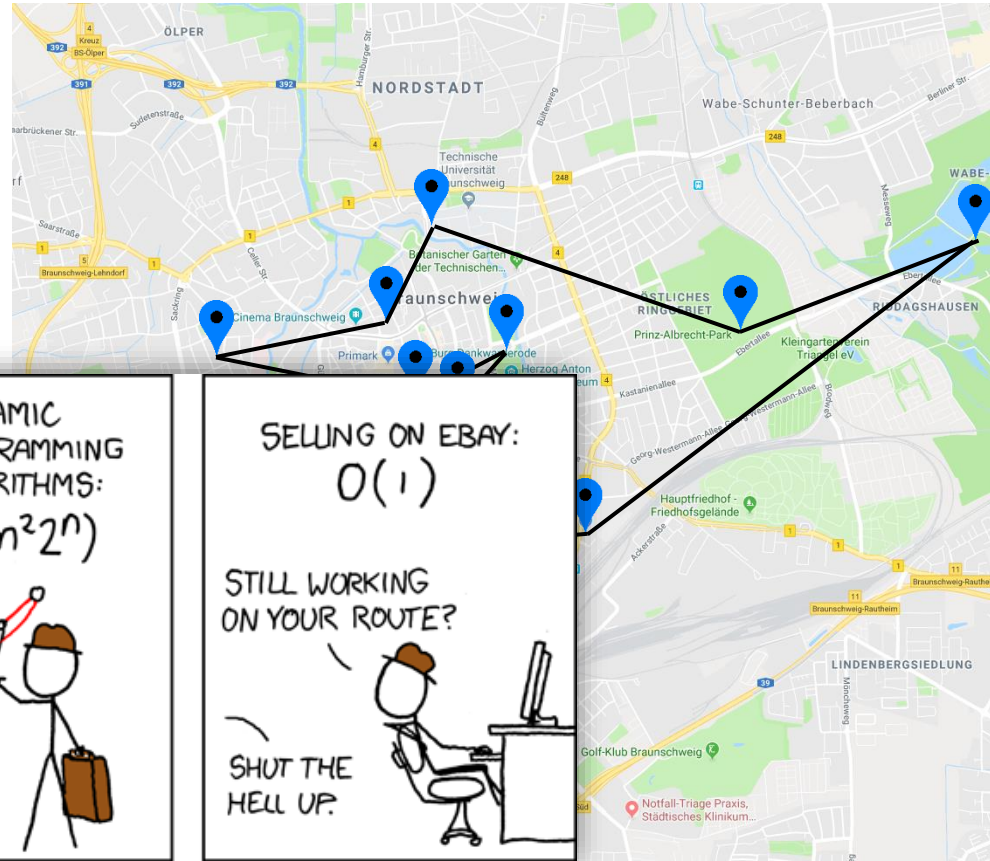
Permutation π

$$\sum_{i=1}^n \|p_{\pi(i)} - p_{\pi(i+1)}\|$$

minimal. Dabei

Finde also eine

die alle Punkte



https://imgs.xkcd.com/comics/travelling_salesman_problem.png

ETSP – Schranken

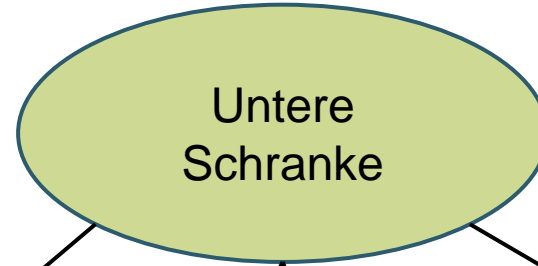


2x Minimal
aufspannender
Baum

Jede *zulässige*
Lösung

Greedy

2-OPT



1x Minimal
aufspannender
Baum

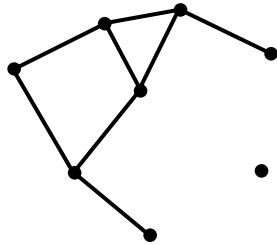
...

Summe der n
kürzesten
Kanten

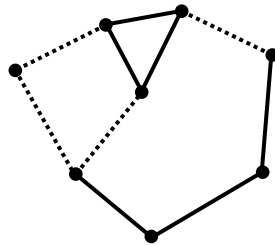
Hälfte der Summe
der zwei kleinsten
Kanten pro
Knoten

ETSP – Untere Schranken

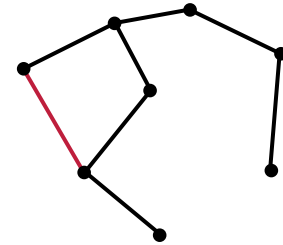
Summe der n
kürzesten Kanten



Hälfte der
Summe der zwei
kleinsten Kanten
pro Knoten



1x Minimal
aufspannender
Baum
(+ n . kürzeste Kante)



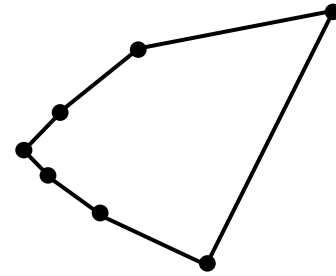
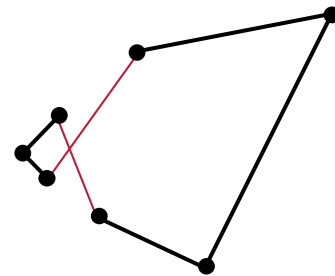
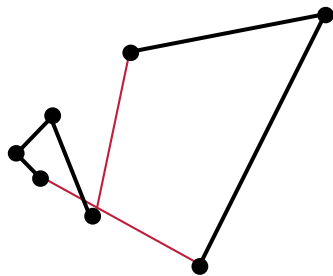
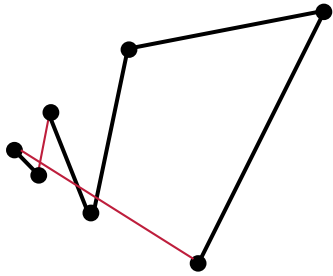
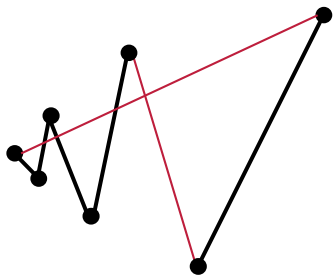
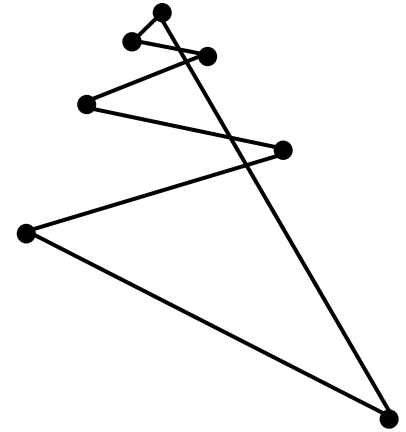
ETSP – Obere Schranken

Greedy
Nearest-Neighbor

Starte bei p_1 und gehe iterativ zum nächstgelegenen Punkt.

2-OPT

In einer beliebigen Tour, tausche zwei existierende Kanten mit zwei nicht-existierenden Kanten, um eine bessere Tour zu erhalten.



ETSP – Branch-and-Bound

Zunächst: Worüber verzweigen? Wo können wir Entscheidungen treffen?

Knoten

- Entscheide für jeden Knoten, wann er besucht wird.
- Verschiedene Knoten dürfen nicht gleichzeitig besucht werden.

Kanten

- Kante ist entweder da, oder nicht.
- Entscheidungen sind unabhängig.
- Einfach Schranken zu finden.

(Teil-)Kreise/Pfade

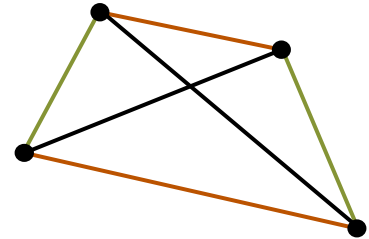
- Also alle Permutationen testen?
- Vorgänger von Knoten raten?
- Wie kann ich darüber Schranken finden?

ETSP – zulässige Teillösungen

Betrachte eine beliebige Reihenfolge der Kanten. Sei $x_i \in \{0,1\}$ die Entscheidungsvariable für die i -te Kante.

Wann ist eine Auswahl von Kanten nicht zulässig?

1. Maximal n Kanten ausgewählt, also
$$\sum_i x_i \leq n$$
2. An jedem Punkt p dürfen nur zwei Kanten liegen, also
$$\sum_{i \in \delta(p)} x_i \leq 2$$
3. Falls $< n$ Kanten ausgewählt sind, darf kein Kreis existieren. (Erkennbar mit BFS/DFS aus AuD)
4. Falls $= n$ Kanten ausgewählt sind, muss exakt ein Kreis existieren. (Wieder mit BFS/DFS)
5. Es dürfen sich keine zwei Kanten schneiden.



Die grünen bzw. braunen Kanten sind höchstens so lang wie die schwarzen. Grund: Dreiecksungleichung
$$d(a,b) + d(b,c) \geq d(a,c)$$

ETSP – Lineares Ungleichungssystem

Etwas anderer Ansatz:

$$\min \sum_{e \in E} c_e x_e$$

s.t.

$$\sum_{e \in \delta(p)} x_e = 2 \quad \text{für alle } p \in P$$

$$\sum_{e \in \delta(S)} x_e \geq 2 \quad \text{für alle } \emptyset \neq S \subsetneq P$$

$$x_e \in \{0,1\} \quad \text{für alle } e \in E$$

E : Menge aller Knotenpaare
 c_e : Distanz des Knotenpaares e

Das ist ein
Integer Linear Programm

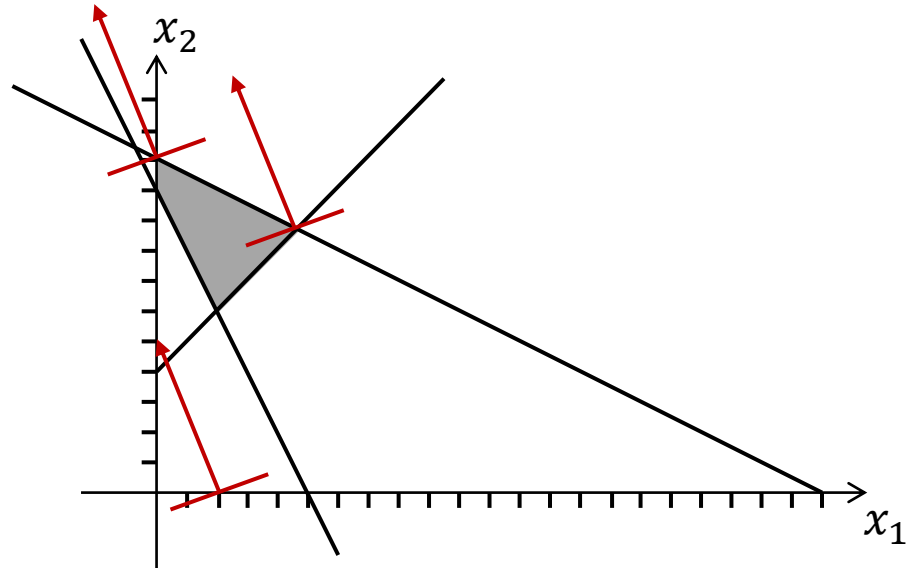
Linear/Integer Programming

Man stellt ein (Un-)Gleichungssystem mit m Variablen in der folgenden Form auf:

$$\begin{aligned} \min c^T x \\ \text{s.t. } Ax \leq b \\ x \in \mathbb{R}^m \end{aligned}$$

Beispiel:

$$\begin{aligned} \min 2x_1 - 5x_2 \\ \text{s.t. } x_1 + 2x_2 \leq 22 \\ 2x_1 + x_2 \geq 10 \\ -x_1 + x_2 \geq 4 \\ x_1, x_2 \geq 0 \end{aligned}$$



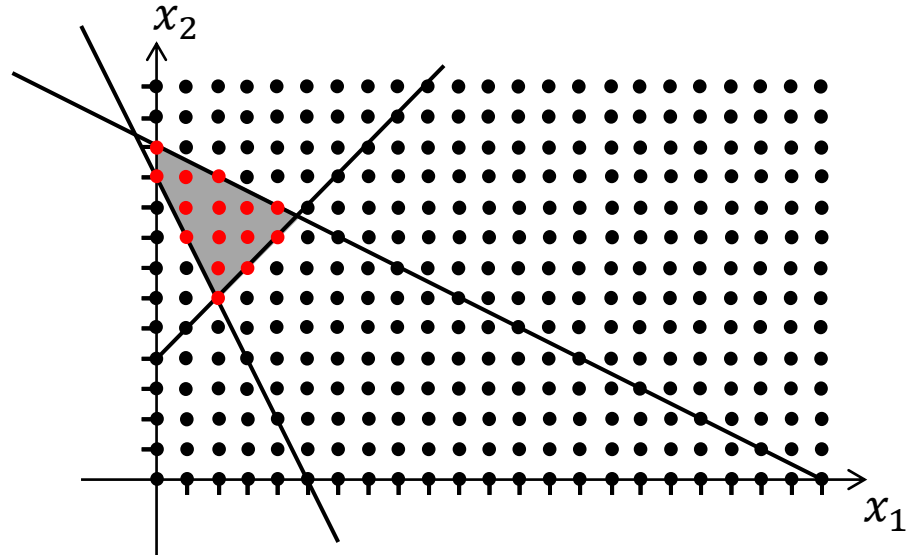
Linear/Integer Programming

Man stellt ein (Un-)Gleichungssystem mit m Variablen in der folgenden Form auf:

$$\begin{aligned} \min c^T x \\ \text{s.t. } Ax \leq b \\ x \in \mathbb{R}^m \end{aligned}$$

Beispiel:

$$\begin{aligned} \min 2x_1 - 5x_2 \\ \text{s.t. } x_1 + 2x_2 \leq 22 \\ 2x_1 + x_2 \geq 10 \\ -x_1 + x_2 \geq 4 \\ x_1, x_2 \in \mathbb{N} \end{aligned}$$

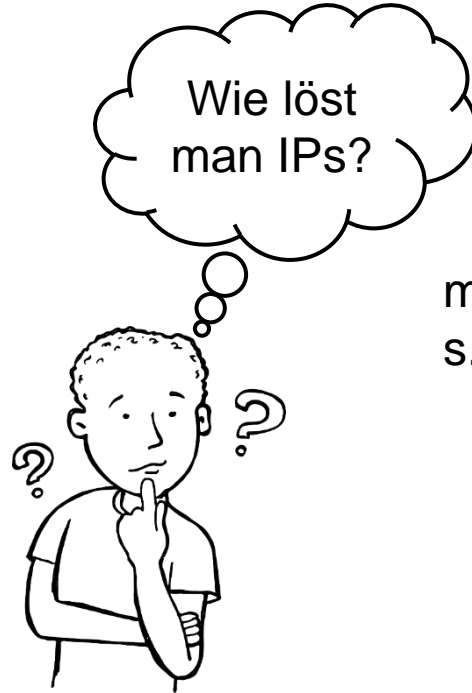


Linear/Integer Programming

LP

$$\begin{aligned} \min \quad & 2x_1 - 5x_2 \\ \text{s.t.} \quad & x_1 + 2x_2 \leq 22 \\ & 2x_1 + x_2 \geq 10 \\ & -x_1 + x_2 \geq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Effizient lösbar

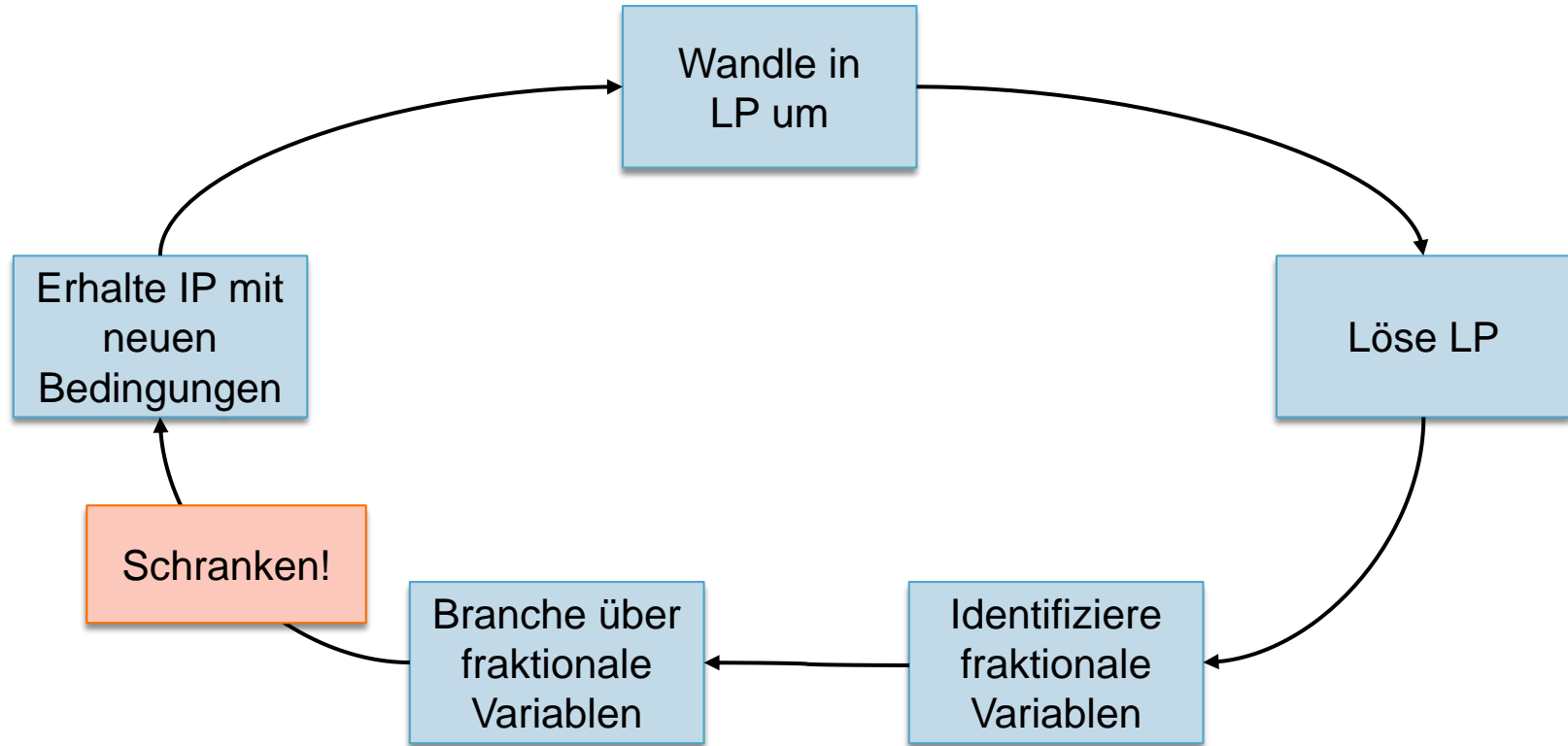


IP

$$\begin{aligned} \min \quad & 2x_1 - 5x_2 \\ \text{s.t.} \quad & x_1 + 2x_2 \leq 22 \\ & 2x_1 + x_2 \geq 10 \\ & -x_1 + x_2 \geq 4 \\ & x_1, x_2 \in \mathbb{N} \end{aligned}$$

Vermutlich nicht
effizient lösbar

Lösen von IPs (Vereinfachte Darstellung)



Solver für Linear/Integer Programming



Microsoft Excel



$$\min \sum_{e \in E} c_e x_e$$

s.t.

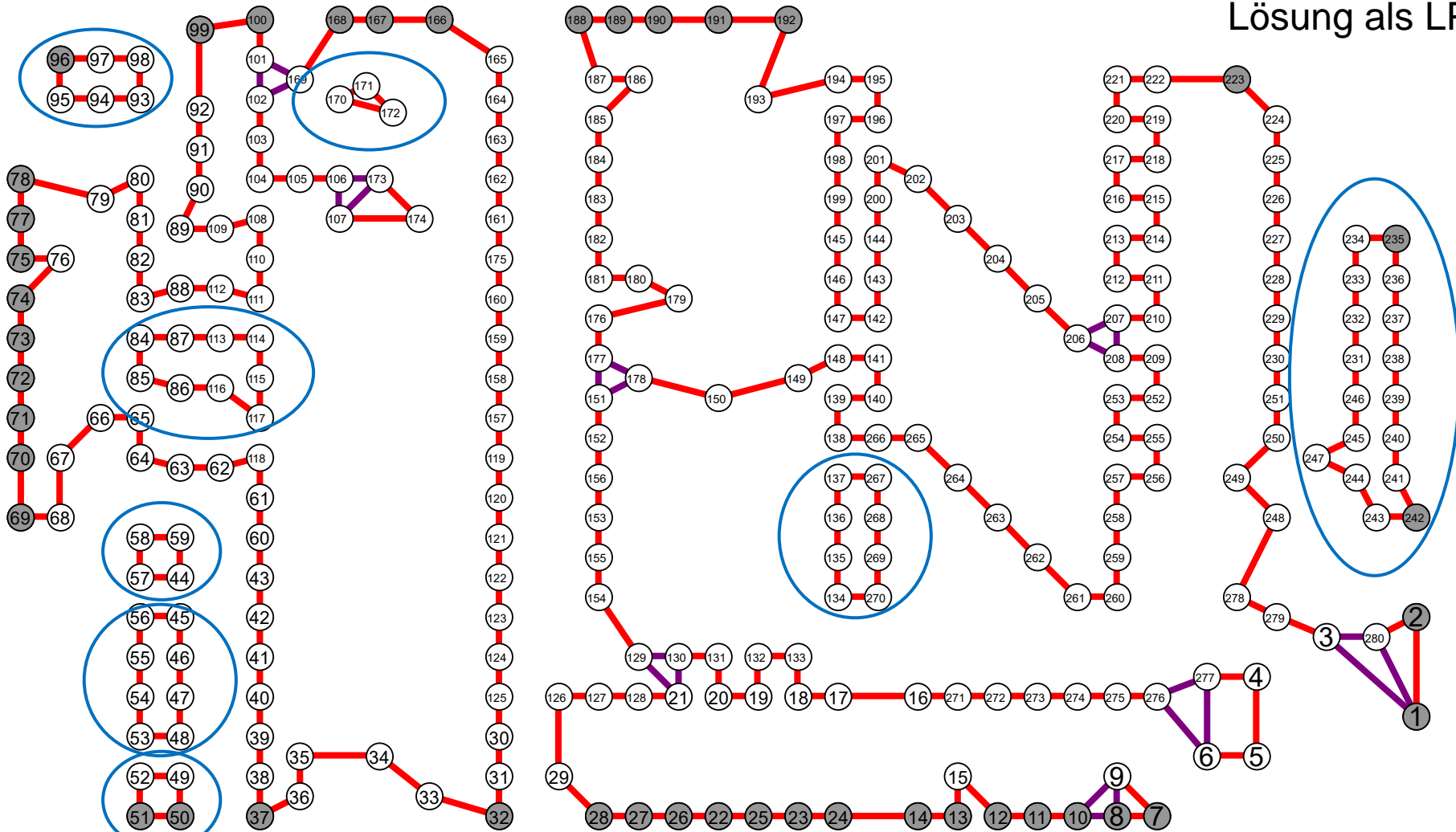
$$\sum_{e \in \delta(p)} x_e = 2 \quad \text{für alle } p \in P$$

$$\sum_{e \in \delta(S)} x_e \geq 2 \quad \text{für alle } \emptyset \neq S \subsetneq P$$

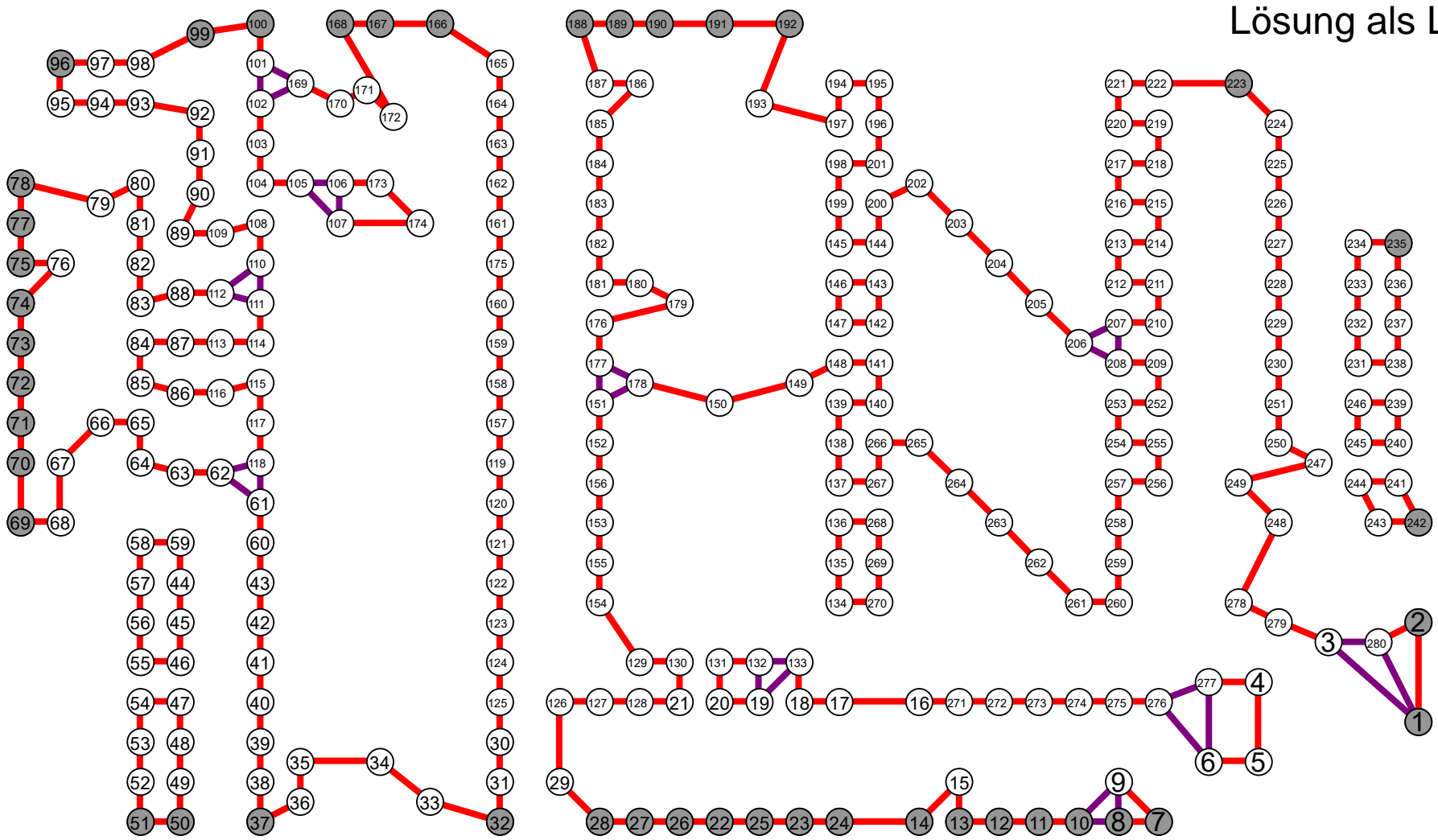
$$x_e \in \{0,1\} \quad \text{für alle } e \in E$$

Exponentiell viele...
Füge sie während des
Lösungsprozesses hinzu!

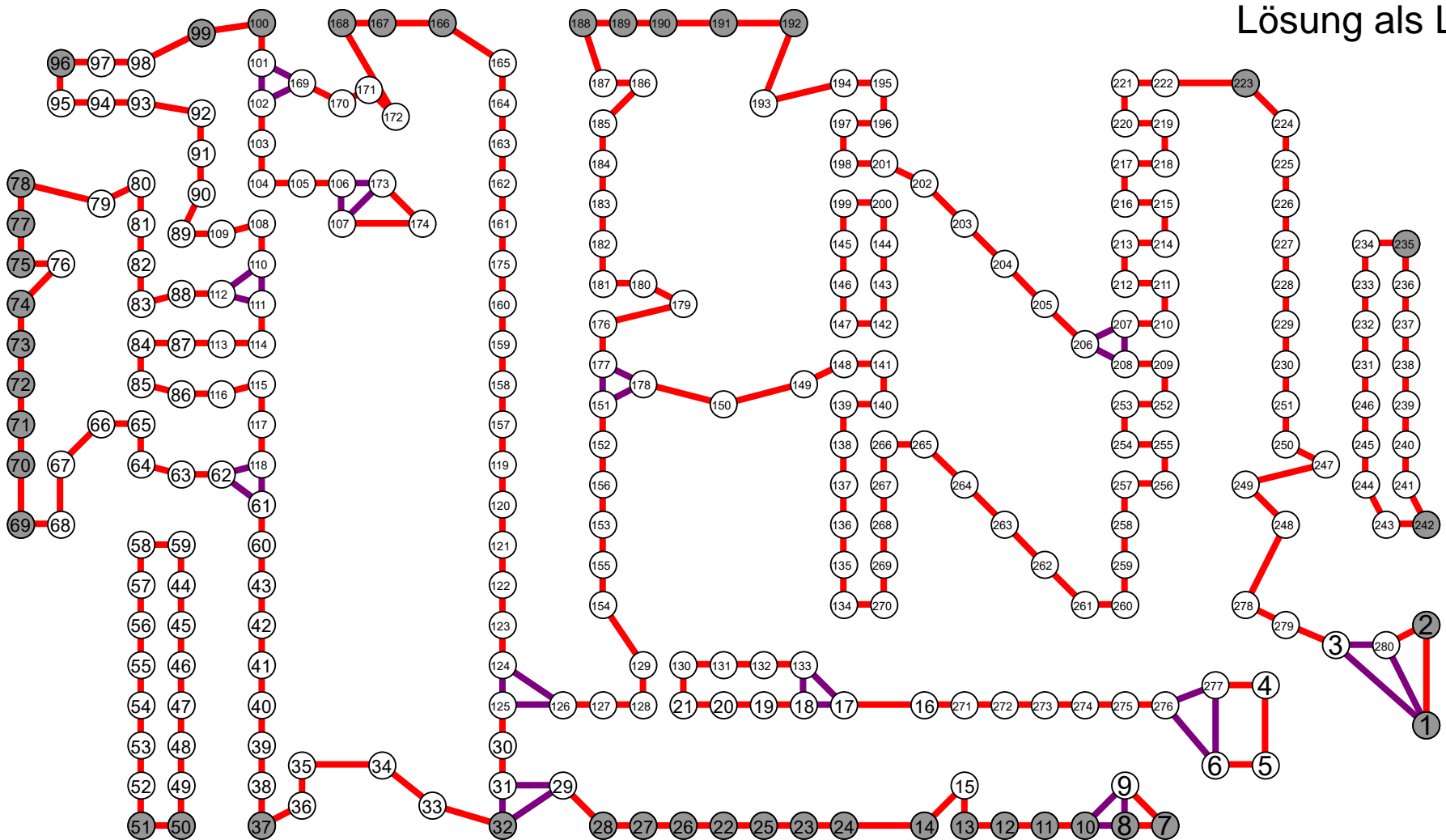
Lösung als LP



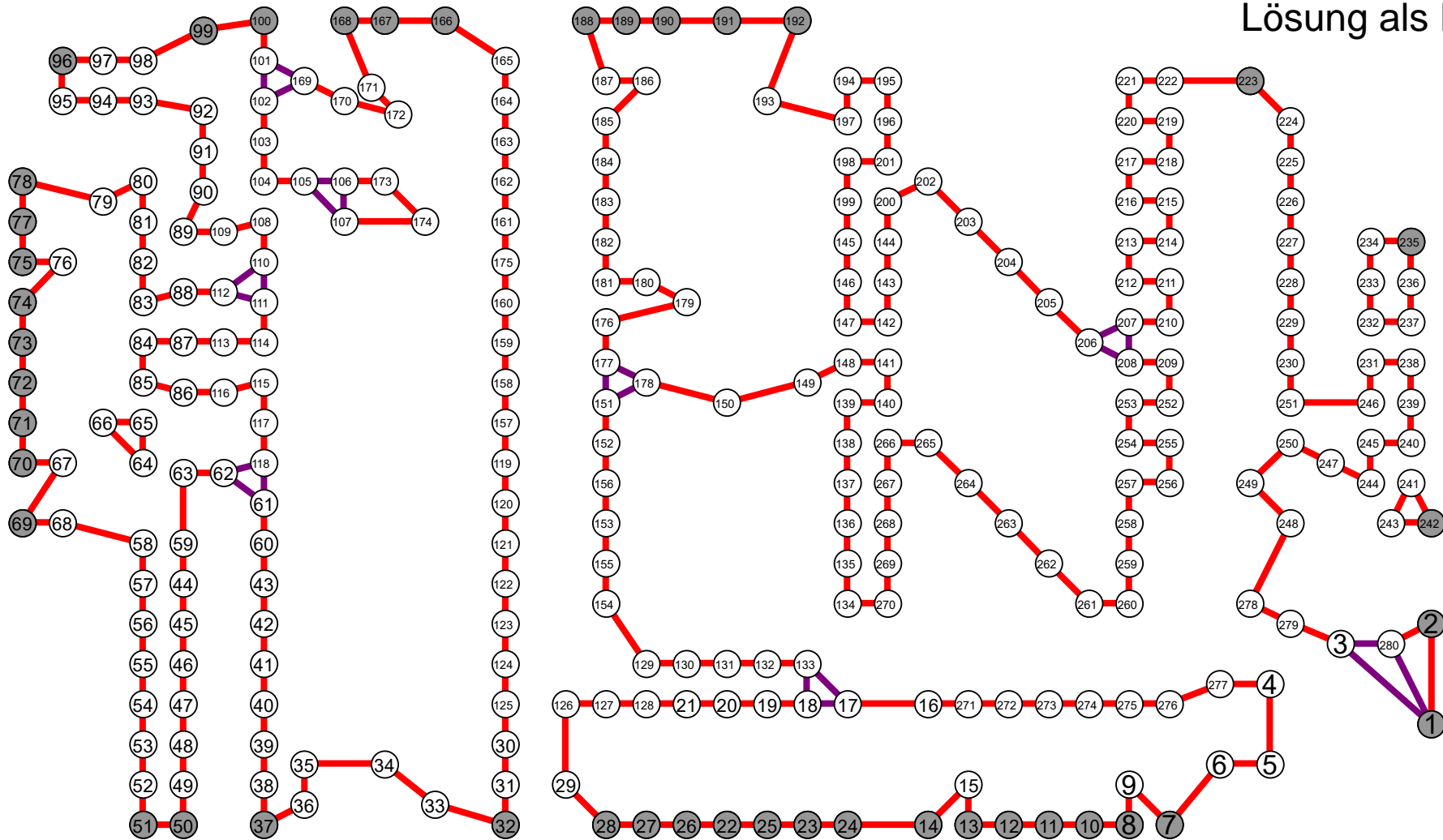
Lösung als LP



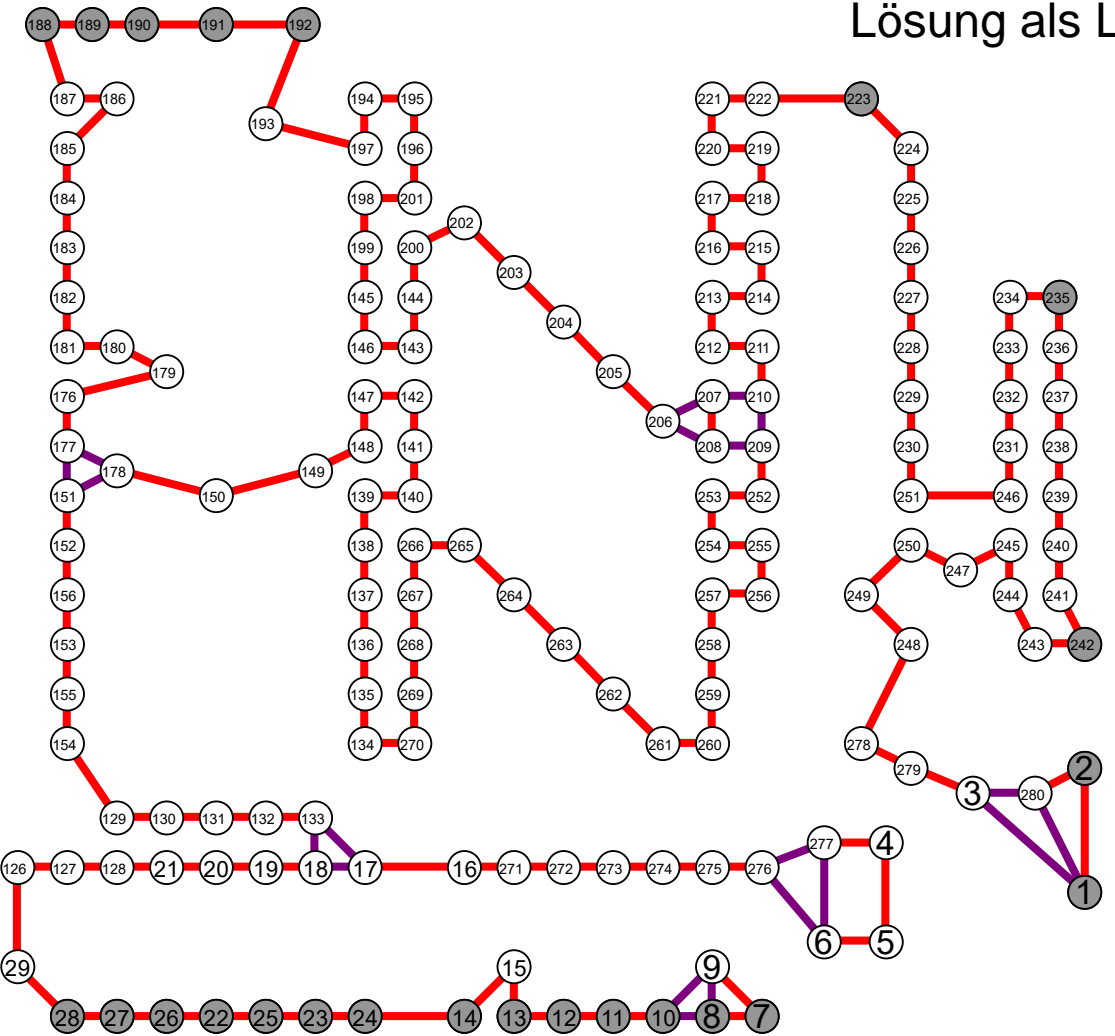
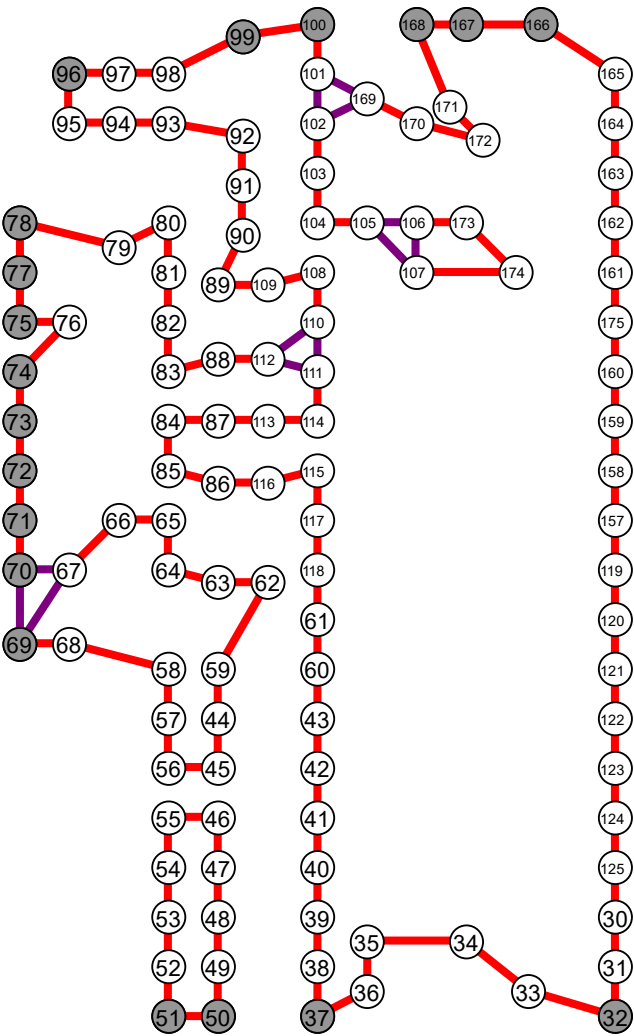
Lösung als LP



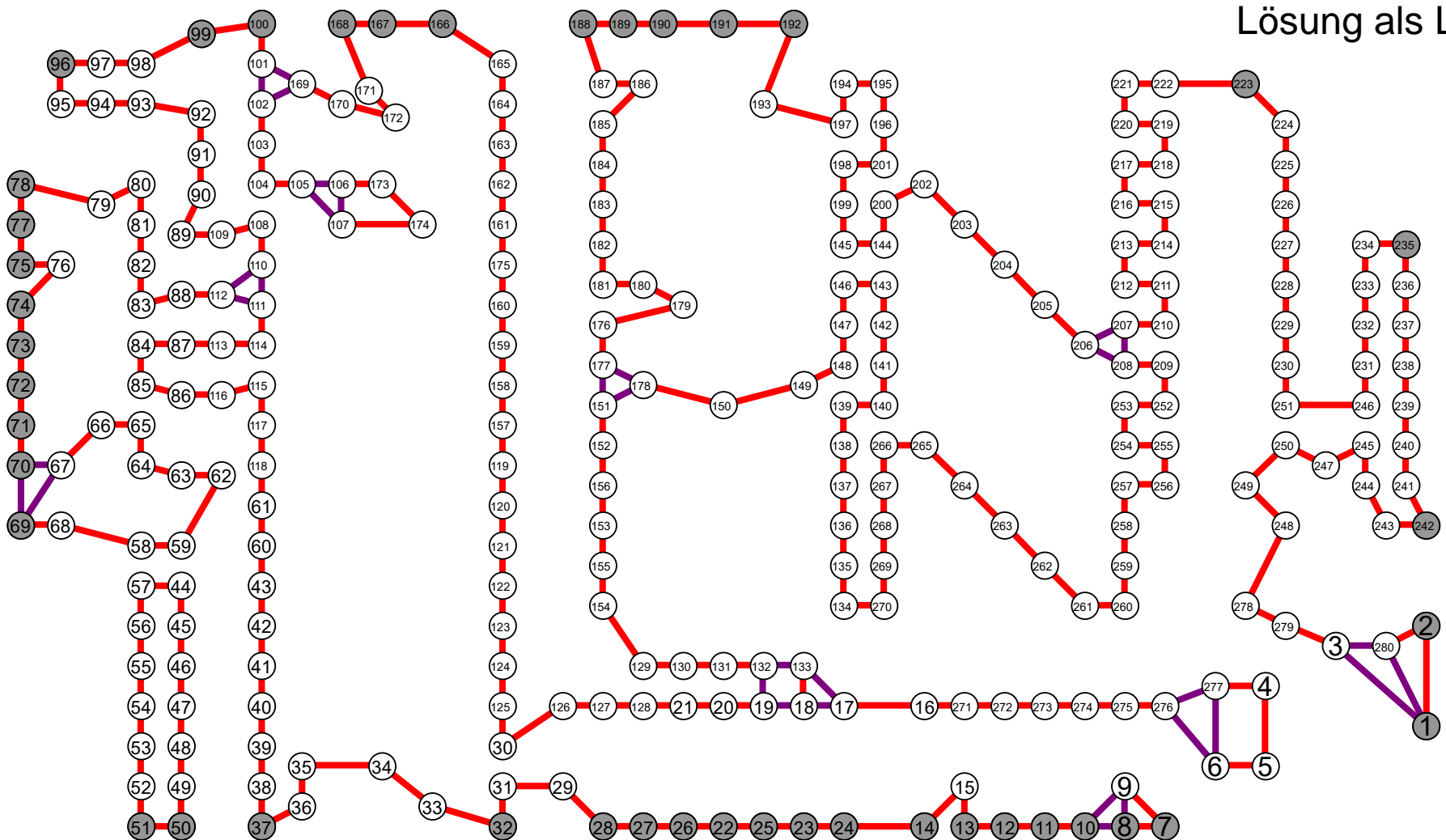
Lösung als LP



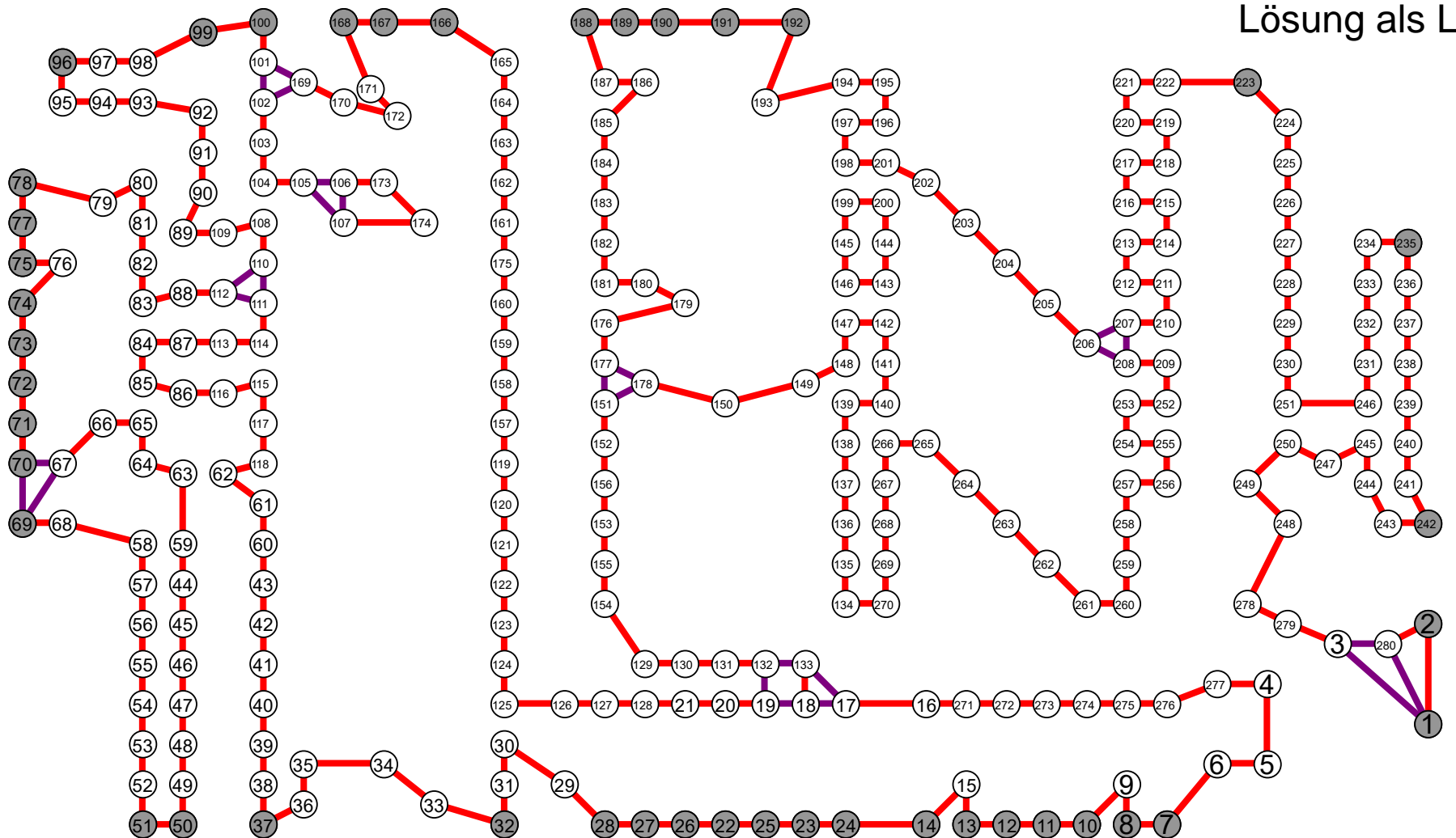
Lösung als LP



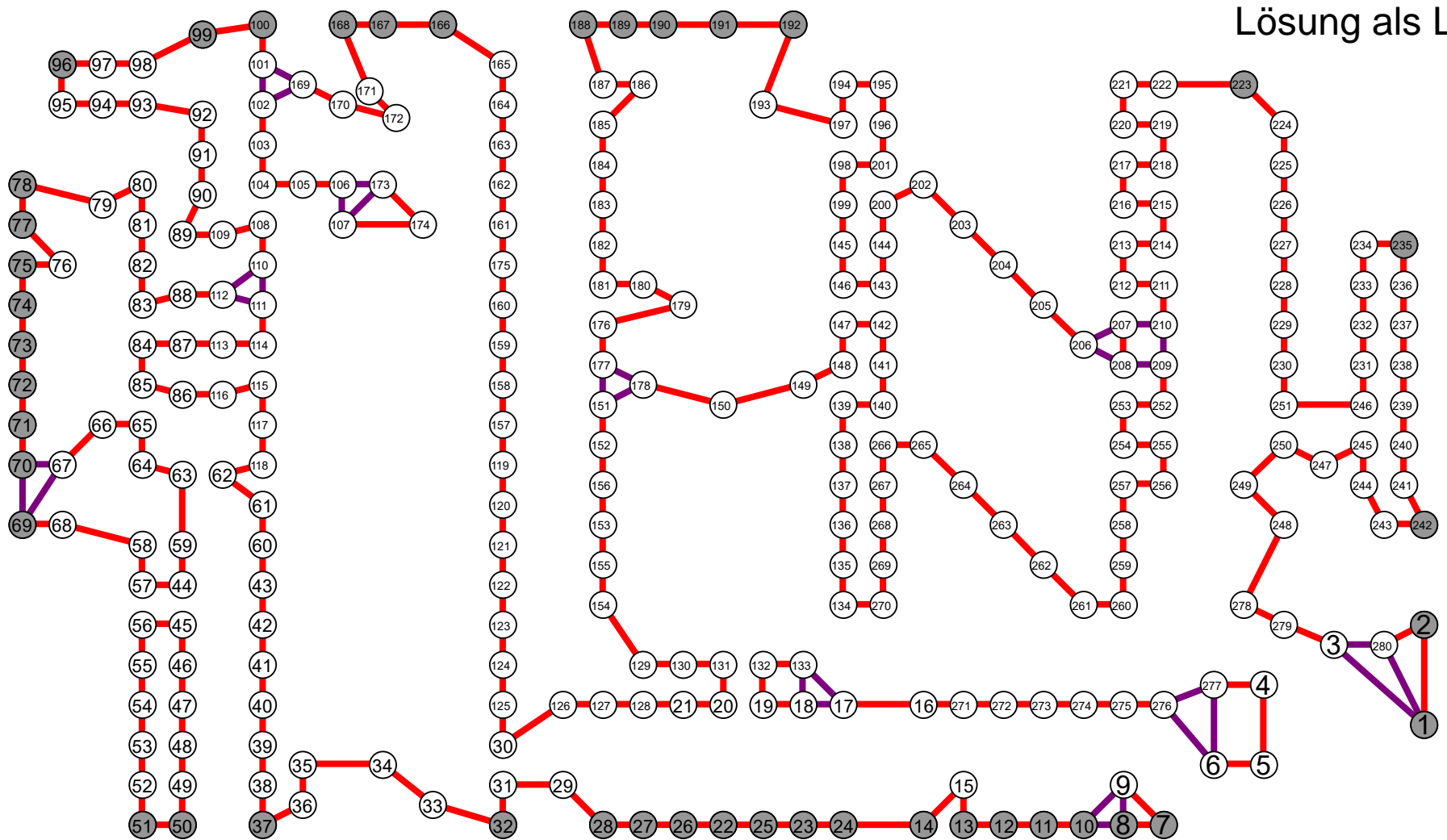
Lösung als LP



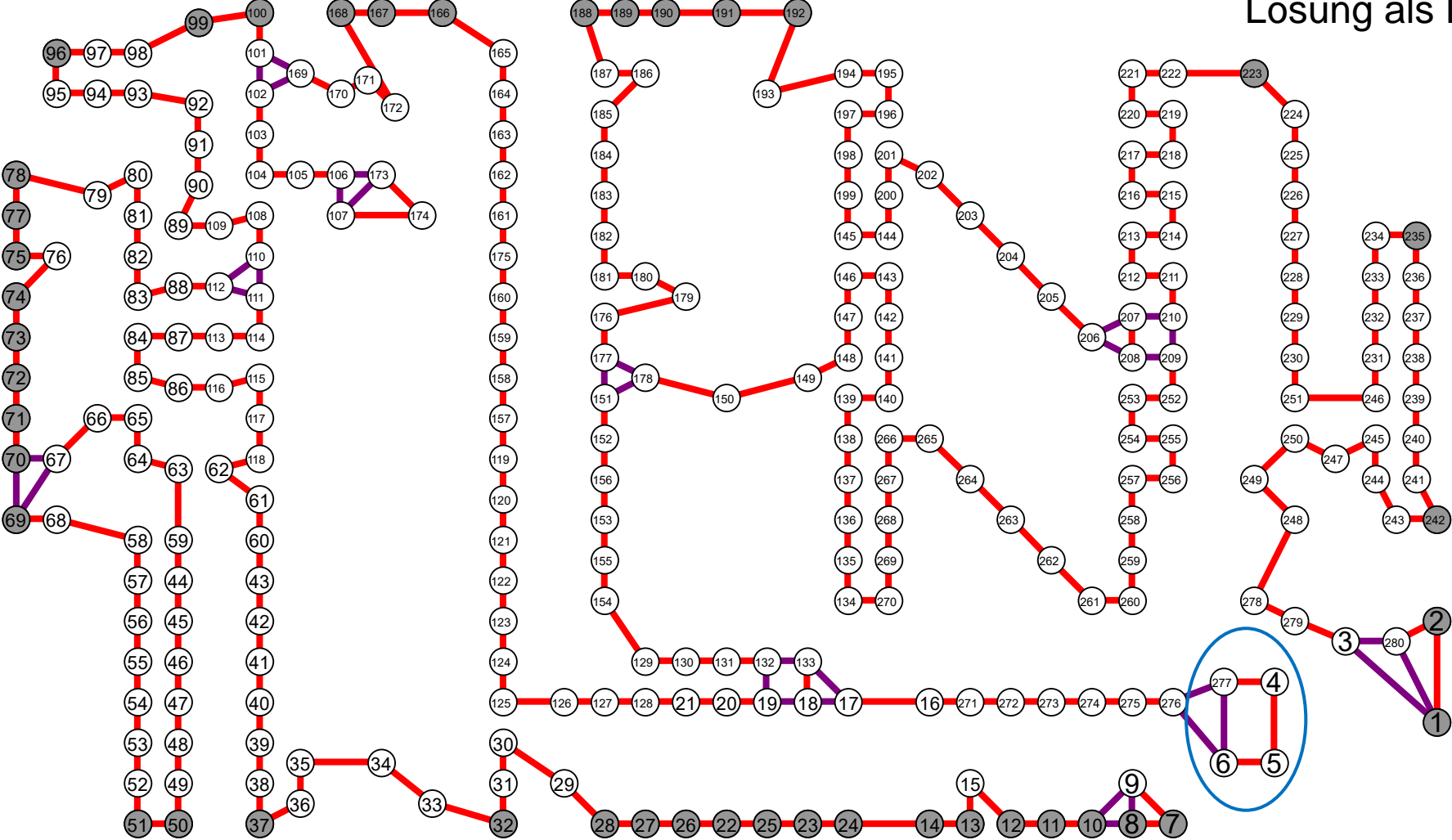
Lösung als LP



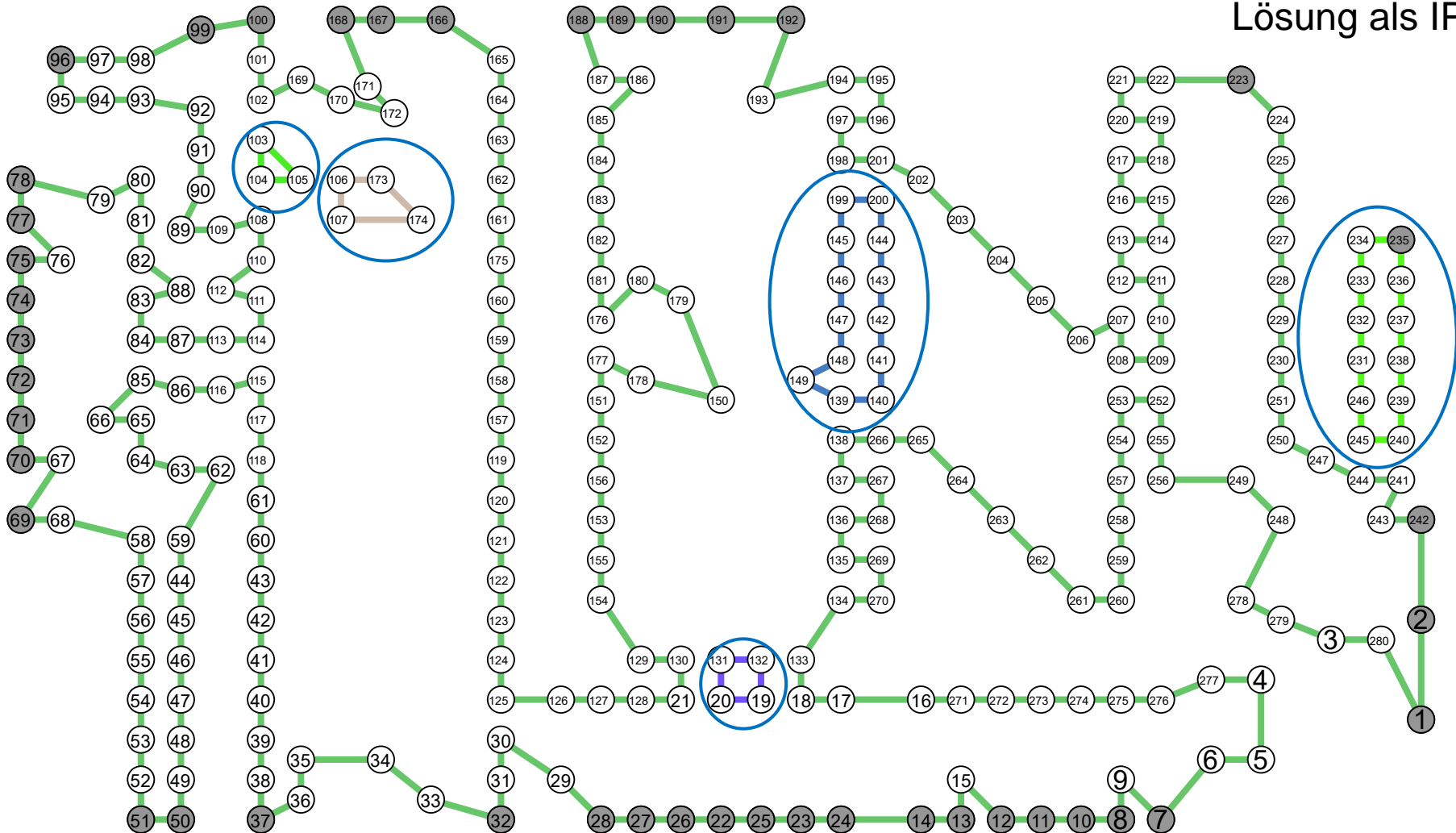
Lösung als LP

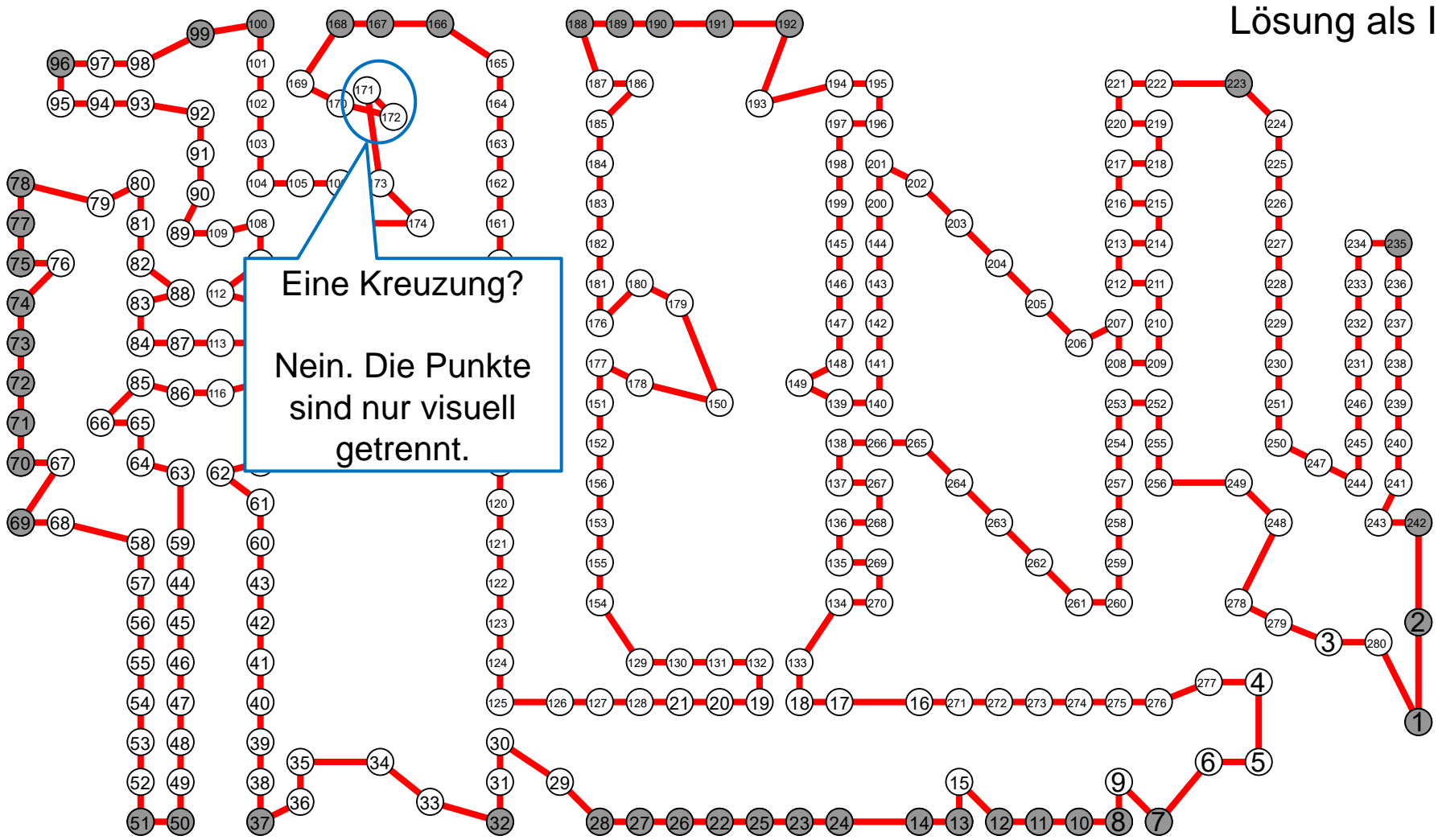


Lösung als LP



Lösung als IP





Interesse an mehr?

Mathematische Methoden der Algorithmik

Semester Wintersemester 2020/2021 ▾

Studiengänge Master Wirtschaftsinformatik, Master Informations-Systemtechnik, Master Informatik

IBR Gruppe ALG (Prof. Fekete)

Art Vorlesung/Übung

Dozent



Dr. Linda Kleist

Wissenschaftliche Mitarbeiterin

✉ kleist@ibr.cs.tu-bs.de

☎ +49 531 3913118

📍 Raum 331

IBR Login

↕ Institut für Betriebssysteme
und Rechnerverbund

News

▼ Wir über uns

▼ Connected and Mobile
Systems

▼ Verteilte Systeme

▼ Algorithmik

Mikroprozesslabor

Master-Veranstaltung

Algorithmikpraktikum: Solving TSP

Semester Sommersemester 2019 ▾

Studiengang Bachelor Informatik

IBR Gruppe ALG (Prof. Fekete)

Art Vorlesung/Übung

Dozent



Prof. Dr. Sándor P. Fekete

Abteilungsleiter

✉ s.fekete@tu-bs.de

☎ +49 531 3913111

📍 Raum 335

IBR Login

↕ Institut für Betriebssysteme
und Rechnerverbund

News

▼ Wir über uns

▼ Connected and Mobile
Systems

▼ Verteilte Systeme

▼ Algorithmik

Bachelor-Veranstaltung