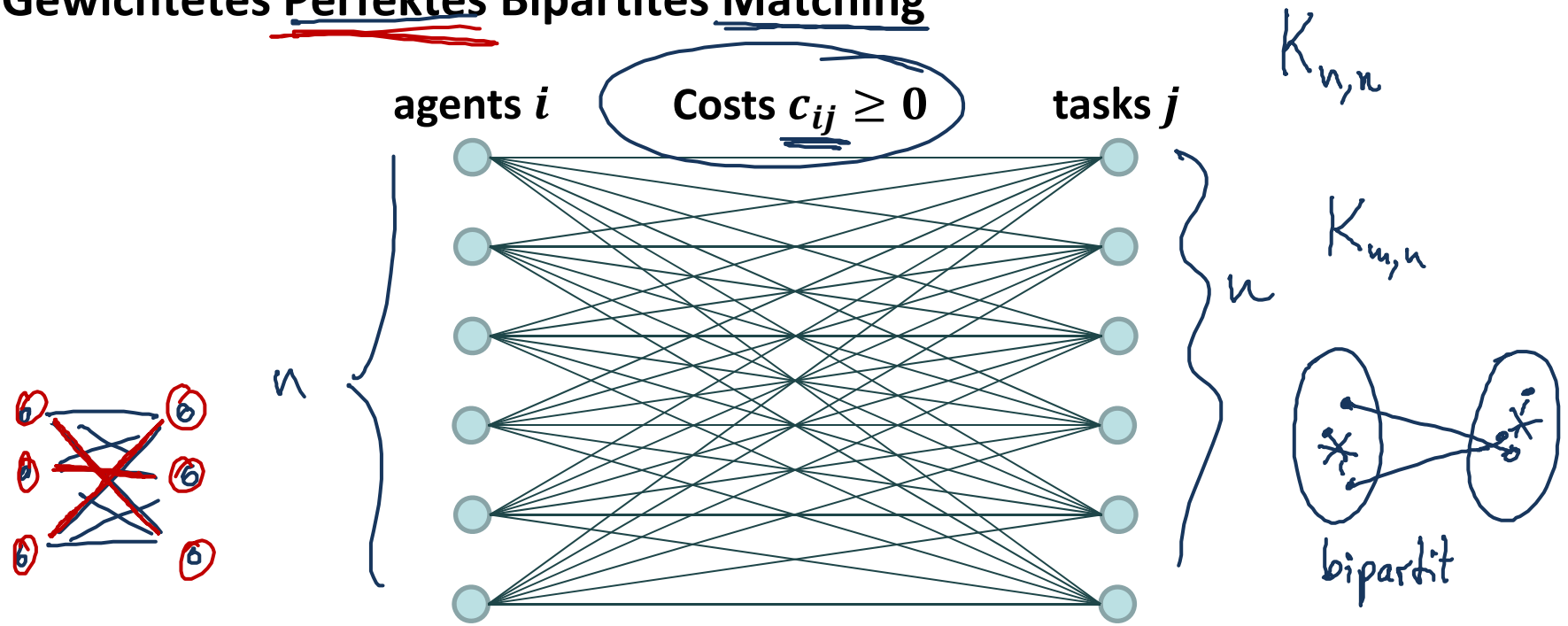
Optimierung

Vorlesung 9

Lineare Programmierung & Kombinatorische Optimierung

Assignment Problem (Zuordnungsproblem)

Gewichtetes Perfektes Bipartites Matching

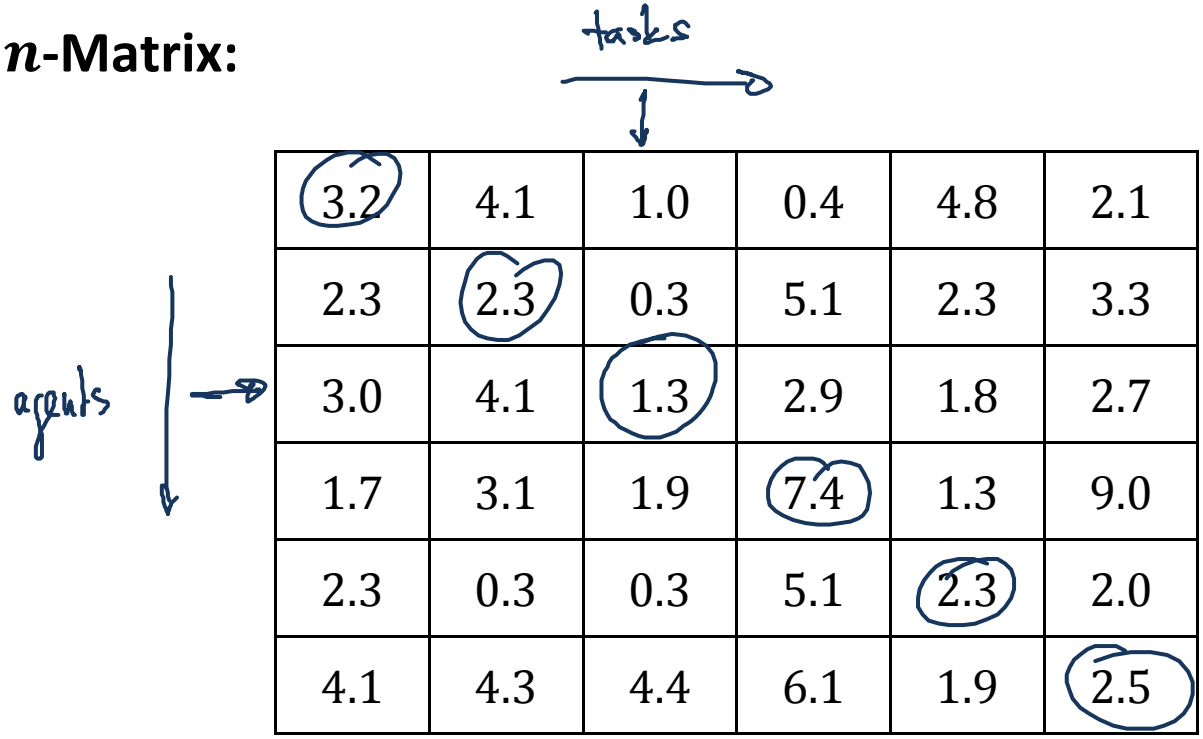


- Weise jedem Agenten genau eine Aufgabe zu
- Minimiere Gesamtkosten
- Kombinatorisches Optimierungsproblem:
  - Zulässige Lösungen: Kantenmengen, so dass jeder Agent und jede Aufgabe Knoten genau einer der Kanten ist

$\sum c_{ij} \cdots$

Assignment Problem (Zuordnungsproblem)

Als $n \times n$-Matrix:



3.2	4.1	1.0	0.4	4.8	2.1
2.3	2.3	0.3	5.1	2.3	3.3
3.0	4.1	1.3	2.9	1.8	2.7
1.7	3.1	1.9	7.4	1.3	9.0
2.3	0.3	0.3	5.1	2.3	2.0
4.1	4.3	4.4	6.1	1.9	2.5

- Wähle genau eine Zelle pro Zeile und Spalte aus (n Zellen)
- Minimiere Gesamtkosten (oder maximiere Gesamtgewicht)
- Kombinatorisches Optimierungsproblem:
  - Zulässige Lösungen: n Zellen, so dass jede Zeile und Spalte genau einmal gewählt ist

Assignment Problem (Zuordnungsproblem)

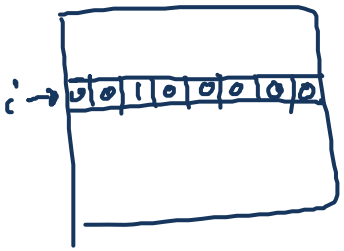
Als Integer Linear Program:

n^2 Variablen

- Variablen $x_{ij} \in \{0,1\}$, $x_{ij} = 1$ falls Zeile i und Spalte j Zelle der Lsg.
  - Falls Agent i Aufgabe j bekommt
- Zuordnungsproblem:

Gesamtkosten

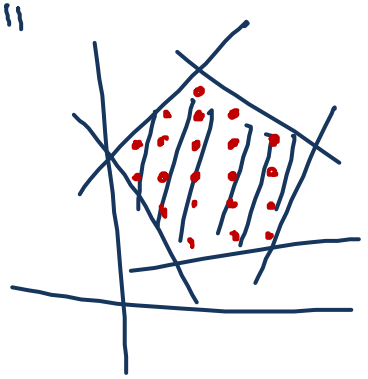
$$\min \sum_{i,j} c_{ij} x_{ij}$$ ← linear



$$\forall i \in \{1,\dots,n\}: \sum_{j=1}^{n} x_{ij} = 1$$ ← linear

$$\forall j \in \{1,\dots,n\}: \sum_{i=1}^{n} x_{ij} = 1$$

$$\forall i,j: \; x_{ij} \in \{0,1\}$$



- Lineares Programm ausser der Bedingung $x_{ij} \in \{0,1\}$

LP Relaxierung

- Schreibe Assignment Problem als lineares Programm:

$$\min \sum_{i,j} c_{ij} x_{ij}$$

$$\forall i \in \{1, \dots, n\}: \sum_{j=1}^{n} x_{ij} = 1$$

$$\forall j \in \{1, \dots, n\}: \sum_{i=1}^{n} x_{ij} = 1$$

$$\forall i, j: \; x_{ij} \geq 0$$

Relaxierung

- Nimm x_{ij}-Anteil der Zelle (i, j)
- Summe der Anteile in jeder Zeile/Spalte ist 1
- Vergrössert den Raum der zulässigen Lösungen
- Optimale Lösung ist untere Schranke für optimale Lösung des Ursprungsproblems

Ganzzahlige Lösungen

Satz: Jede nicht-ganzzahlige Lösung kann in eine ganzzahlige Lösung mit gleichen oder besseren Gesamtkosten transformiert werden.

Cost: 3 $x_{11} = 0.2$	Cost: 1 $x_{12} = 0.5$	Cost: 4 $x_{13} = 0$	Cost: 2 $x_{14} = 0.3$
Cost: 2 $x_{21} = 0.3$	Cost: 3 $x_{22} = 0$	Cost: 2 $x_{23} = 0.4$	Cost: 3 $x_{24} = 0.3$
Cost: 1 $x_{31} = 0.5$	Cost: 1 $x_{32} = 0.5$	Cost: 1 $x_{33} = 0$	Cost: 2 $x_{34} = 0$
Cost: 3 $x_{41} = 0$	Cost: 3 $x_{42} = 0$	Cost: 2 $x_{43} = 0.6$	Cost: 1 $x_{44} = 0.4$

Ganzzahlige Lösungen

Satz: Jede nicht-ganzzahlige Lösung kann in eine ganzzahlige Lösung mit gleichen oder besseren Gesamtkosten transformiert werden.

Cost: 3 $x_{11} = 0$	Cost: 1 $x_{12} = 0.7$	Cost: 4 $x_{13} = 0$	Cost: 2 $x_{14} = 0.3$
Cost: 2 $x_{21} = 0.3$	Cost: 3 $x_{22} = 0$	Cost: 2 $x_{23} = 0.4$	Cost: 3 $x_{24} = 0.3$
Cost: 1 $x_{31} = 0.7$	Cost: 1 $x_{32} = 0.3$	Cost: 1 $x_{33} = 0$	Cost: 2 $x_{34} = 0$
Cost: 3 $x_{41} = 0$	Cost: 3 $x_{42} = 0$	Cost: 2 $x_{43} = 0.6$	Cost: 1 $x_{44} = 0.4$

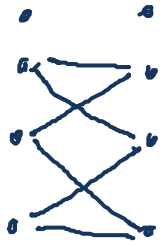
Ganzzahlige Lösungen

Satz: Jede nicht-ganzzahlige Lösung kann in eine ganzzahlige Lösung mit gleichen oder besseren Gesamtkosten transformiert werden.

Cost: 3 $x_{11} = 0$	Cost: 1 $x_{12} = 0.7$	Cost: 4 $x_{13} = 0$	Cost: 2 $x_{14} = 0.3$
Cost: 2 $x_{21} = 0.3$	Cost: 3 $x_{22} = 0$	Cost: 2 $x_{23} = \mathbf{0.7}$	Cost: 3 $x_{24} = \mathbf{0}$
Cost: 1 $x_{31} = 0.7$	Cost: 1 $x_{32} = 0.3$	Cost: 1 $x_{33} = 0$	Cost: 2 $x_{34} = 0$
Cost: 3 $x_{41} = 0$	Cost: 3 $x_{42} = 0$	Cost: 2 $x_{43} = \mathbf{0.3}$	Cost: 1 $x_{44} = \mathbf{0.7}$

Ganzzahlige Lösungen

Satz: Jede nicht-ganzzahlige Lösung kann in eine ganzzahlige Lösung mit gleichen oder besseren Gesamtkosten transformiert werden.



Cost: 3 $x_{11} = 0$	Cost: 1 $x_{12} = 1$	Cost: 4 $x_{13} = 0$	Cost: 2 $x_{14} = 0$
Cost: 2 $x_{21} = 0$	Cost: 3 $x_{22} = 0$	Cost: 2 $x_{23} = 1$	Cost: 3 $x_{24} = 0$
Cost: 1 $x_{31} = 1$	Cost: 1 $x_{32} = 0$	Cost: 1 $x_{33} = 0$	Cost: 2 $x_{34} = 0$
Cost: 3 $x_{41} = 0$	Cost: 3 $x_{42} = 0$	Cost: 2 $x_{43} = 0$	Cost: 1 $x_{44} = 1$

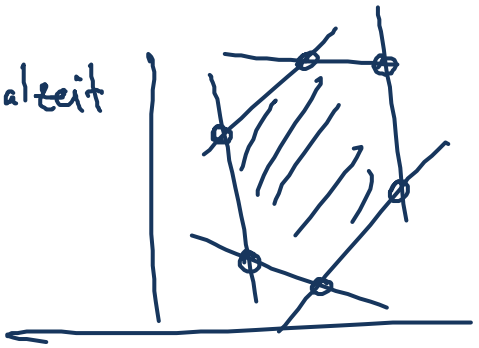
Allgemein:

- Es gibt immer einen Zyklus, so dass man alternierend um den gleichen Betrag erhöhen, reduzieren kann und die Kosten nicht steigen
  - Vgl: alternierende Pfade bei Matching-Formulierung

Lösung Assignment Problem

Mögliche Lösung des Assignment Problems:

1. Löse LP Relaxierung (LP mit $x_{ij} \geq 0$) ← Polynomialzeit
2. Mache Lösung ganzzahlig



Mit Simplex-Algorithmus:

- Der Simplex-Algorithmus gibt immer einen Extremalpunkt des konvexen Polyeders, welches durch die lin. Nebenbed. definiert ist.
- Mann kann zeigen: Alle Extremalpunkte des Assignment-Problems sind ganzzahlig
- Der Simplex-Algorithmus gibt direkt eine optimale ganzzahlige Lösung!

Ungarische Methode:

- Schnellerer Algorithmus, spezifisch für das Assignment Problem

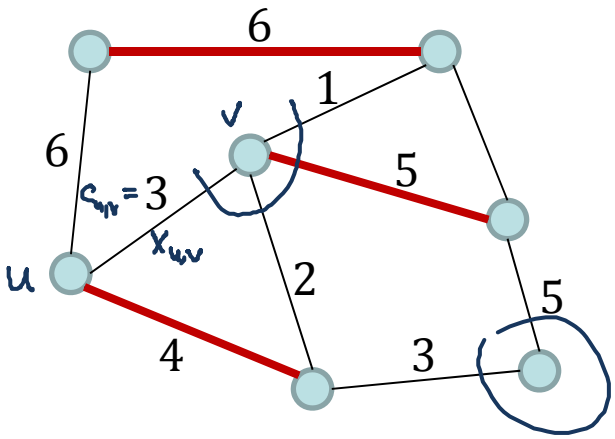
Kombinatorische Optimierung und LPs

Viele wichtige kombinatorische Optimierungsprobleme lassen sich als "Integer"-Varianten von linearen Programmen schreiben.

Assignment Problem (min gewichtetes perfektes Matching im $K_{n,n}$)

Maximum (Weighted) Matching:

- Gegeben ein (gewichteter) Graph $G = (V, E)$
- Matching: Teilmenge der Kanten, so dass sich keine 2 berühren
- Finde Matching mit maximalem Gewicht



$$\max \sum_{\{u,v\} \in E} c_{u,v} \cdot x_{u,v}$$

$$\forall u \in V: \quad \sum_{v \in N(u)} x_{u,v} \leq 1$$

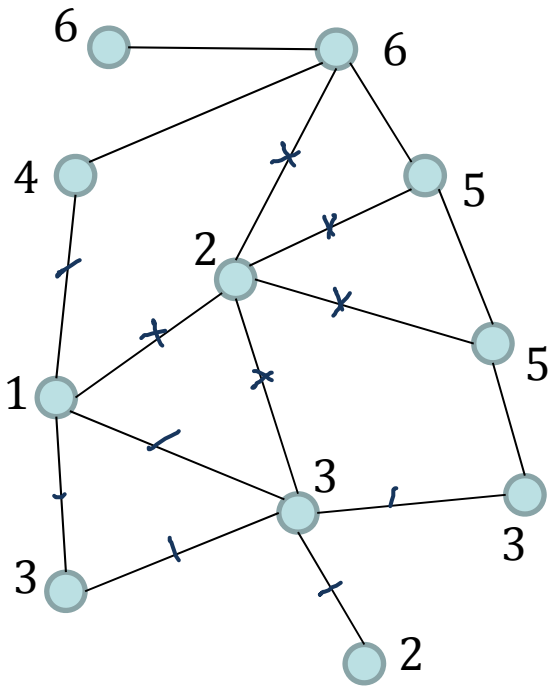
$$x_{u,v} \in \{0,1\}$$

$$x_{u,v} \geq 0 \quad \text{fractional matching}$$

Vertex Cover

~~Maximum~~ minimum (Weighted) Vertex Cover (~~VC~~):

- Gegeben ein Graph $G = (V, E)$ mit Knotengewichten
- Vertex Cover: Teilmenge der Knoten, so dass jede Kante abgedeckt ist
- Finde Vertex Cover mit minimalem Gewicht



$\forall v \in V: \; x_v \in \{0,1\}$

$$\min \sum_{v \in V} c_v \cdot x_v$$

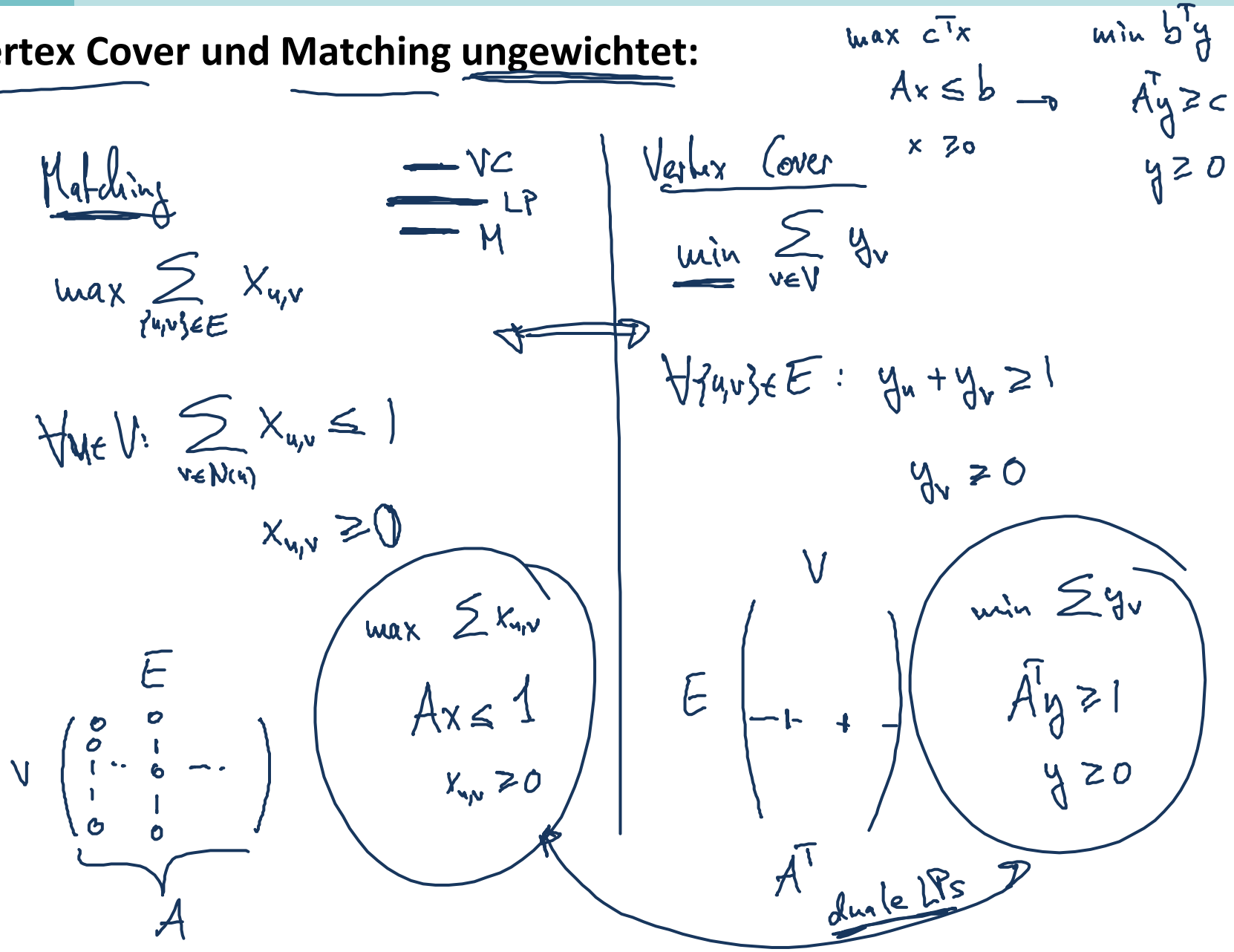
$$\forall \{u,v\} \in E: \; x_u + x_v \geq 1$$

$$x_v \in \{0,1\}$$

$$x_v \geq 0$$

Vertex Cover und Matching

Vertex Cover und Matching ungewichtet:



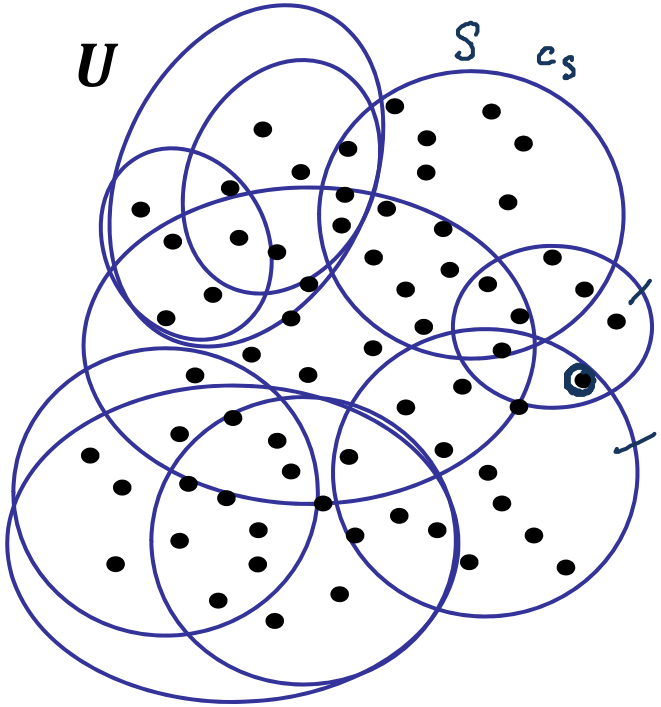
Set Cover

(Gewichtetes) Set Cover: $(U, \mathcal{S})$: set system

- Gegeben: Menge U und Menge von Teilmengen $\mathcal{S} \in 2^U$

$$\mathcal{S} = \{S_1, S_2, \dots, S_t\}, \quad S_i \subseteq U$$

- Jede Menge $S \in \mathcal{S}$ hat ein Gewicht c_S
- Set Cover: Teilmenge von $\mathcal{S}$, so dass alle Elemente $u \in U$ abgedeckt sind
- Ziel: Set Cover mit minimalem Gewicht



$$\forall S \in \mathcal{S}: x_S \in \{0,1\}$$

$$\min \sum_{S \in \mathcal{S}} c_S x_S$$

$$\forall u \in U: \sum_{S \in \mathcal{S}: u \in S} x_S \geq 1$$

$$x_S \in \{0,1\}$$

$$x_S \geq 0$$

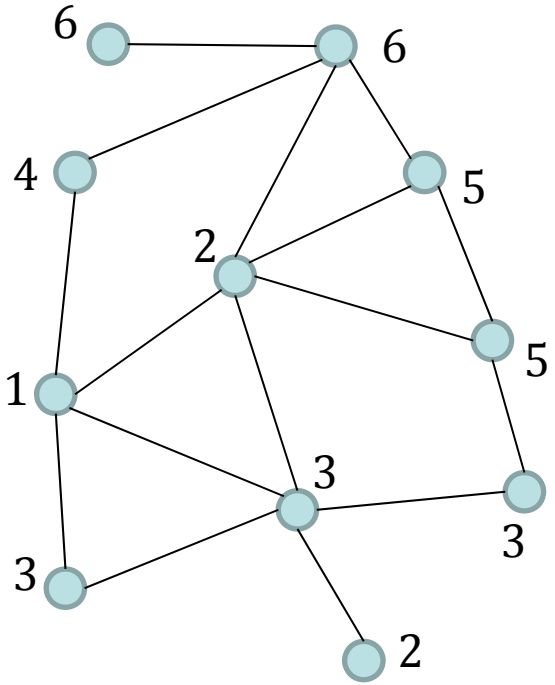
set cover

NP-hard

Independent Set

Maxium (gewichtetes) Independent Set (Stabile Menge):

- Gegeben Graph $G = (V, E)$ mit Knotengewichten
- Independent Set (unabhängige/stabile Menge):
  Teilmenge der Knoten, so dass keine 2 adjazent sind
- Ziel: Finde independent set mit maximalem Gewicht



$$\max \sum_{v \in V} c_v x_v$$

$$\forall \{u,v\} \in E: \quad x_u + x_v \leq 1$$

$$x_v \in \{0,1\}$$

$$x_v \geq 0$$

Maximaler Fluss

Flow Network:

- Directed graph $G = (V, E)$, $E \subseteq V^2$
- Each (directed) edge e has a capacity $c_e \geq 0$
  - Amount of flow (traffic) that the edge can carry
- A single source node $s \in V$ and a single sink node $t \in V$

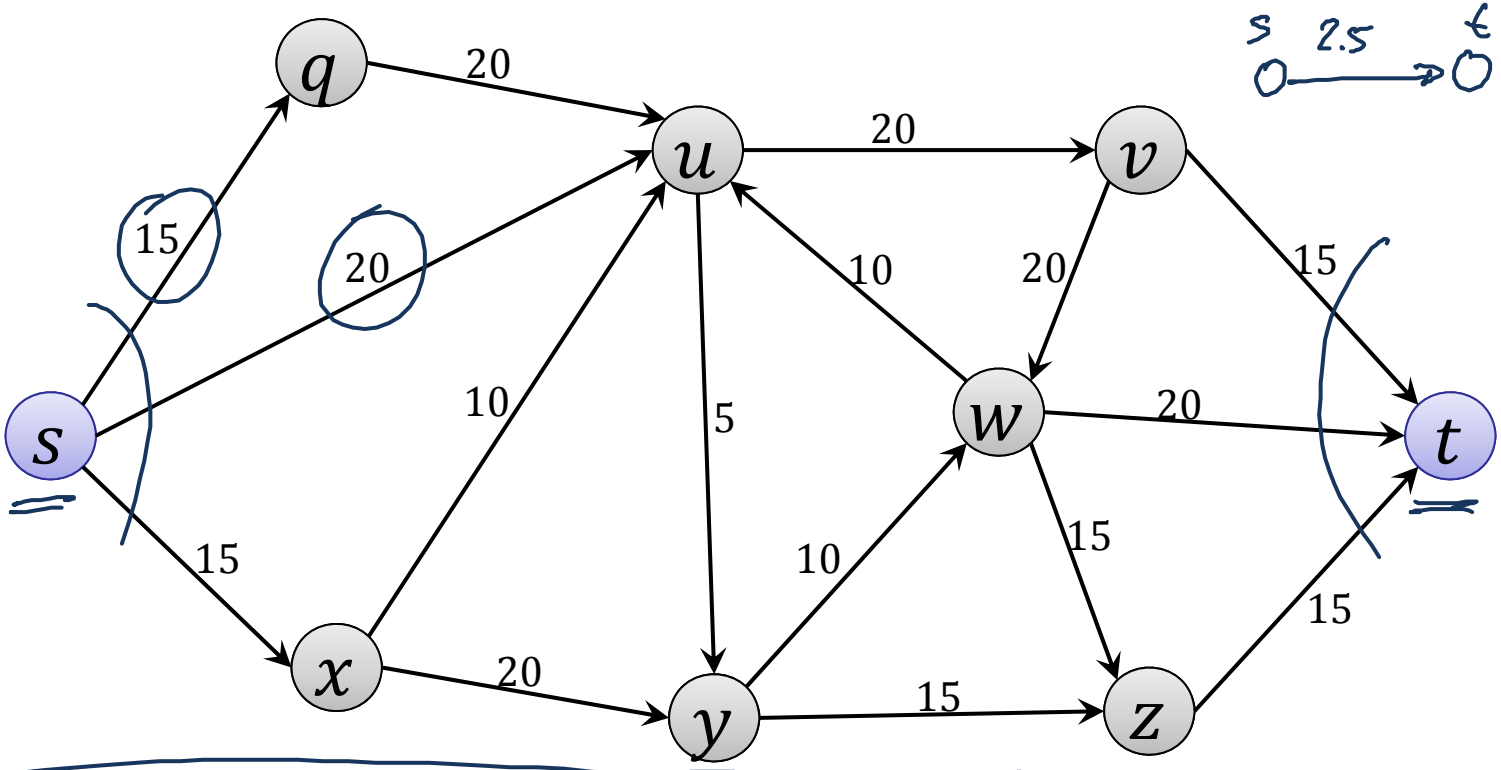
Flow: (informally)

- Traffic from s to t such that each edge carries at most its capacity

Examples:

- Highway system: edges are highways, flow is the traffic
- Computer network: edges are network links that can carry packets, nodes are switches
- Fluid network: edges are pipes that carry liquid

Maximaler Fluss



s 2.5 t

$0 \le f_{sq} \le 15$

$0 \le f_{xu} \le 10$

$\vdots$

$f_{uy} + f_{xy} = f_{yw} + f_{yz}$

$f_{uv} = f_{vw} + f_{vt}$

$\vdots$

$\max f_{sq} + f_{su} + f_{sx}$

$f_{uv} \in \mathbb{Z}$

LP Relaxierung

1. Löse LP Relaxierung (in Polynomialzeit!)
2. Falls möglich: Konvertiere in gleich gute ganzzahlige Lösung

Funktioniert bei:

- **Assignment Problem**
  (Maximum Weighted Matching in bipartiten Graphen)

- **Maximum Integer Flow**

- **Minimum Cost Flow**
  - Anstatt Kapazität, Kantenkosten
  - Erreiche gegebene Vorgaben (für alle Knoten) mit minmalen Kosten
  - Verallgemeinerung des Max Flow Problems

Allgemeines Matching

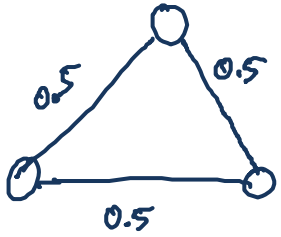
Matching in allgemeinem Graph $G = (V, E)$

$\forall \{u,v\} \in E : x_{u,v} \in \{0,1\}$

$\max \sum x_{u,v}$

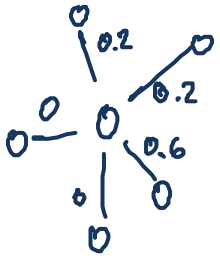
$\forall v \in V: \sum_{u \in N(v)} x_{u,v} \leq 1$

$x_{u,v} \geq 0$



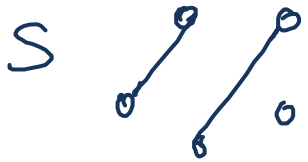
Wert: 1.5

Bestes Matching: 1



Allgemeines Matching

Zusätzliche Bedingungen:



Knotenmenge S

- Matching kann höchstens $\lfloor |S|/2 \rfloor$ Kanten $\{u, v\}$ mit $u, v \in S$ enthalten
- Variable $x_{u,v} = 1$: Kante $\{u, v\}$ im Matching
- Zusätzlich: Für alle ungeraden Knotenmengen S:

$$\sum_{u,v \in S, \{u,v\} \in E} x_{u,v} \leq \frac{|S| - 1}{2}$$

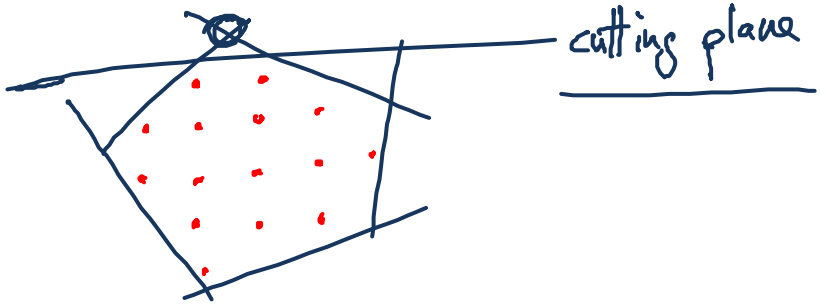
- Neues LP hat nur ganzzahlige Extremallösungen
- Aber: Exponentielle Anzahl Nebenbedingungen!
- Es gibt einen polynomiellen Algorithmus von Edmonds
  - Alg. findet duale optimale Lösung mit nur polynomiell vielen Variablen, welche nicht Null sind
  - Dient als Zertifikat für die Optimalität

Cutting Planes

Allgemeine Optimierungs-Technik:

1. Formuliere kombinatorisches Problem als Integer LP
2. Relaxiere auf ein LP
3. Löse LP mit Hilfe des Simplex- oder eines anderen Algorithmus
4. Falls Lösung ganzzahlig sind wir fertig
5. Sonst:

   Versuche weitere Nebenbedingungen hinzuzufügen, so dass
   - die momentane Lösung nicht mehr zulässig ist
   - alle ganzzahligen Lösungen weiterhin zulässig sind



Approximationsalgorithmen

Vertex Cover und Maximum Matching (ungewichtet):

NP-hart

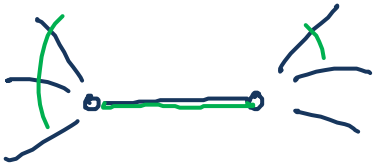
Dualität: max matching ≤ min vertex cover

LP

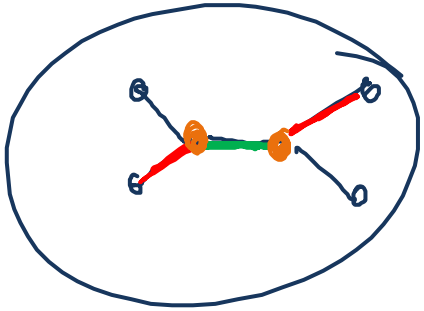
maximales Matching M

Matching, welches sich nicht erweitern lässt

Alle Knoten von M sind ein vertex cover



vertex covers der Grösse $\leq 2|M|$



M^*

C^*

2-Approximation

$$|M| \leq |M^*| \leq |C^*| \leq 2|M|$$

($|M^*| \leq |C^*|$: ≤ 2)

Vertex Cover und Maximum Matching (ungewichtet):

Approximationsalgorithmen

Gewichtetes Vertex Cover:

1. Löse LP Relaxierung
2. Runde auf ganzzahlige Lösung:

- Für jede Kante $\{u,v\} \in E$ gilt $x_u + x_v \geq 1$

x_u — x_v $\quad \max\{x_u, x_v\} \geq 0.5$ $\quad c_v \geq 0$

- Falls $x_u \geq 0.5$, runde x_u auf $x_u = 1$ auf
  Falls $x_u < 0.5$, runde x_u auf $x_u = 0$ ab

$\min \sum c_v \cdot x_v$

- Gibt eine 2-Approximation:

gültige Lsg.

höchstens Verdopplung

Integer LP und LP Relaxierung

Integrality Gap

- Verhältnis zwischen optimalem Zielfunktionswert des Integer LP und der Relaxierung

Maximum Matching

- Integrality Gap ist mindestens $3/2$

Minimum Vertex Cover (ungewichtet)

- Integrality Gap ist mind. $2 - 1/n$ (und höchstens 2)

opt. Lsg: $n-1$

LP-Lsg: $x_u = 0.5$ $n/2$

Maximum Clique (ungewichtet)

- Integrality Gap ist mindestens $n/2$.

K_u

opt. Lsg.: 1

$x_u + x_v \leq 1$

$x_v = 0.5$

opt. LP-Lsg: $\frac{n}{2}$

Minimum Set Cover (ungewichtet)

Greedy-Algorithmus:

- Füge Menge $S \in \mathcal{S}$ hinzu, welche die meisten neuen Elemente abdeckt

Minimum Set Cover (ungewichtet)

Greedy-Algorithmus:

- Füge Menge $S \in \mathcal{S}$ hinzu, welche die meisten neuen Elemente abdeckt

Minimum Set Cover (ungewichtet)

Greedy-Algorithmus:

- Füge Menge $S \in \mathcal{S}$ hinzu, welche die meisten neuen Elemente abdeckt

Minimum Set Cover (gewichtet)

Randomisiertes Runden:

1. Löse LP Relaxierung
   - Jede Menge $S \in \mathcal{S}$ hat Wert x_S, so dass

$$\forall u \in U : \sum_{S \in \mathcal{S}: u \in S} x_S \geq 1$$

2. Mit Wahrscheinlichkeit $\min\{1, x_S \cdot \ln n\}$ setze $x'_S = 1$ (sonst $x'_S = 0$)

Minimum Set Cover (gewichtet)

Randomisiertes Runden:

1. Löse LP Relaxierung
   - Jede Menge $S \in \mathcal{S}$ hat Wert x_S, so dass

$$\forall u \in U : \sum_{S \in \mathcal{S}: u \in S} x_S \geq 1$$

2. Mit Wahrscheinlichkeit $\min\{1, x_S \cdot \ln n\}$ setze $x'_S = 1$ (sonst $x'_S = 0$)