Algorithmen & Wahrscheinlichkeit

Übungsstunde 10

Michelle Honegger - ML F34

Ablauf

- Theory Recap
- InClass Exercise
- Kahoot!

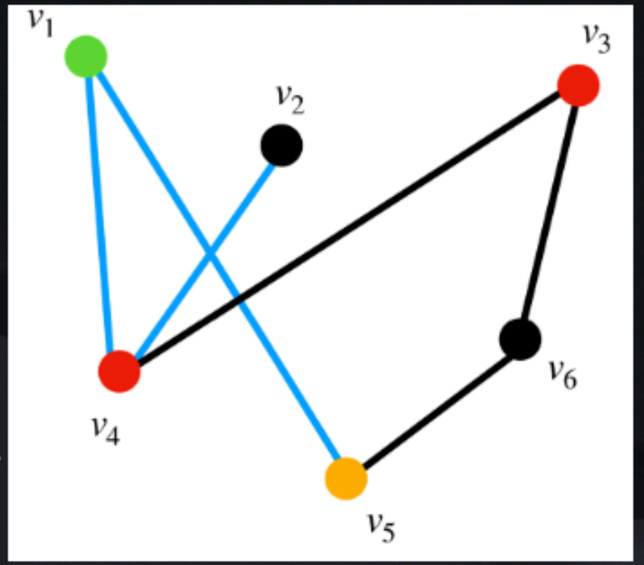
Lange Pfade

- Gegeben Graph G und Zahl B, gibt es einen Pfad in G mit länge B?
- Hamiltonkreis schwer -> Lange Pfade schwer
  - Erstelle G’ indem wir v und inzidente Kanten {v,wi} durch {wi’,wi} ersetzen
  - Hat ≤ 2n - 2 Knoten
  - Zeige Konstruktion von Hamiltonkreis

Lange Pfade

Bunte Pfade

- **gegeben**: ein Graph G und eine Färbung $\gamma : V \to [k]$ (die Färbung muss nicht gültig sein nach def aus Kapitel 1)
- **gesucht**: enthält G einen bunten Pfad der Läange k-1
  - bunter Pfad: alle Knoten haben unterschiedliche Farben
- $P_i(v) :=$ alle bunten Pfade der Länge i, die in v enden
- $P_i(v) := \{S \in \binom{[k]}{i+1} \mid \exists$ bunter Pfad, der in v endet und genau mit S gefärbt ist$\}$
- **gesucht**: ob $\bigcup_{v \in V} P_{k-1}(v) \neq \emptyset$



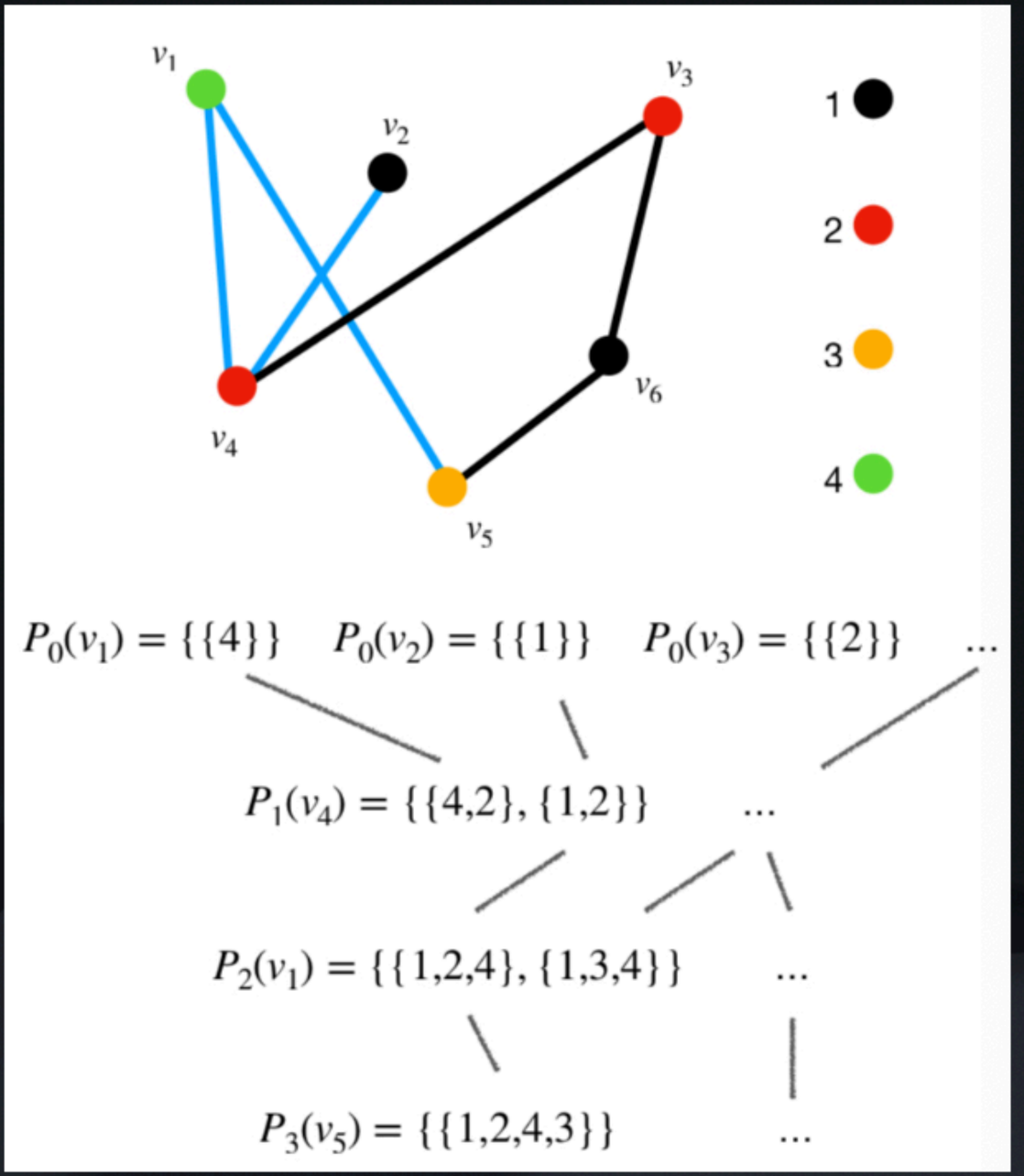
Lange Pfade

Bunte Pfade

BUNT(G, i)

1: **for all** $v \in V$ **do**
2: $\quad P_i(v) \leftarrow \emptyset$
3: $\quad$ **for all** $x \in N(v)$ **do**
4: $\quad\quad$ **for all** $R \in P_{i-1}(x)$ **mit** $\gamma(v) \notin R$ **do**
5: $\quad\quad\quad P_i(v) \leftarrow P_i(v) \cup \{R \cup \{\gamma(v)\}\}$

- findet alle bunten Pfade der Länge i, basierend auf bunten Pfaden der Länge i-1.
- $O(\binom{k}{i} \cdot i \cdot m)$



Lange Pfade

Bunte Pfade

BUNT(G, i)

1: **for all** $v \in V$ **do**
2: $\quad P_i(v) \leftarrow \emptyset$
3: $\quad$ **for all** $x \in N(v)$ **do**
4: $\quad\quad$ **for all** $R \in P_{i-1}(x)$ **mit** $\gamma(v) \notin R$ **do**
5: $\quad\quad\quad P_i(v) \leftarrow P_i(v) \cup \{R \cup \{\gamma(v)\}\}$

- findet alle bunten Pfade der Länge i, basierend auf bunten Pfaden der Länge i-1.
- $O(\binom{k}{i} \cdot i \cdot m)$

Regenbogen(G, γ)

1: **for all** $v \in V$ **do** $P_0(v) \leftarrow \{\{\gamma(v)\}\}$
2: **for** $i = 1..k-1$ **do** Bunt(G, i)
3: **return** $\bigcup_{v \in V} P_{k-1}(v) \neq \emptyset$

findet, ob es einen bunten Pfad der Länge k-1 gibt.

$O(2^k km)$

Lange Pfade

- **gegeben**: ein Graph G und k $\in \mathbb{N}_0$
- **gesucht**: ob G einen Pfad der Länge k enthält: NP-schwer zeigen durch Reduktion von HamiltonPfad-Problem
- Wir lösen Regenbogen (G, γ'), wobei $\gamma' : V \to [k+1]$ ist eine gleichverteilte Färbung mit k+1 Farben
-

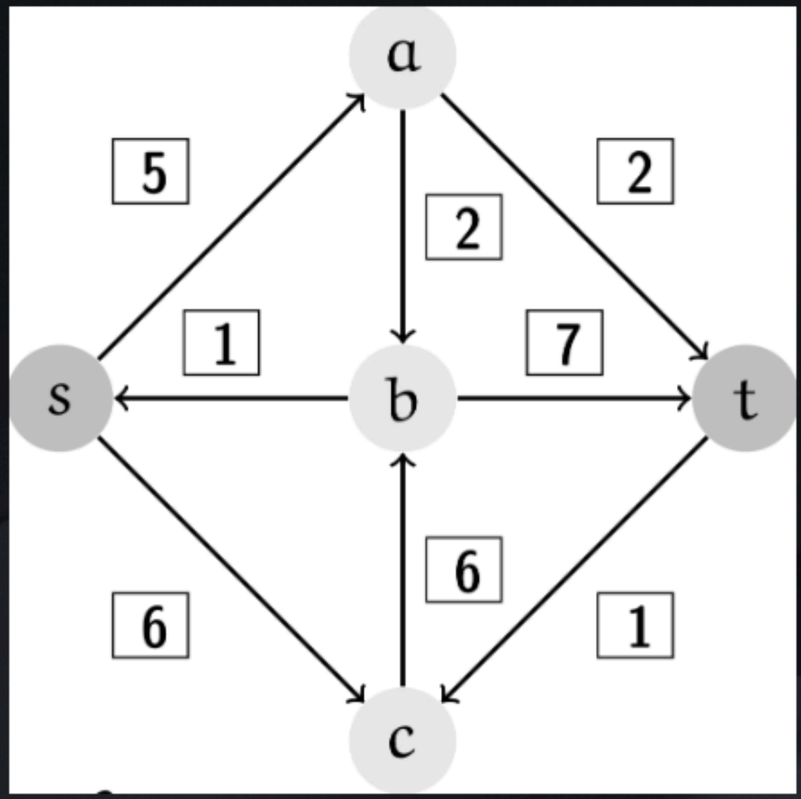
1 Versuch	$\lceil \lambda e^k \rceil$ Versuche
$O(2^k km)$	$O(\lambda e^k \cdot 2^k km)$
$P_{Erfolg} \geq e^{-k}$	$P_{Erfolg} \geq 1 - e^{-\lambda}$

Probability DP

- Leetcode: 688. Knight Probability in Chessboard

Flüsse in Netzwerken

- Ein Netzwerk ist ein Tupel N = (V,A,c,s,t), wobei gilt:
  - (V,A) ist ein gerichteter Graph
  - s ∈ V, die Quelle
  - t ∈ V\{s}, die Senke
  - c: A $\rightarrow \mathbb{R}_0^+$, die Kapazitätsfunktion



Flüsse in Netzwerken

- Sei N = (V,A,c,s,t) euin Netzwerk. Ein Fluss in N ist eine Finktion f : A $\rightarrow \mathbb{R}$ mit den Bedingungen:
  - Zulässigkeit: 0 ≤ f(e) ≤ c(e)
  - Flusserhaltung: Für alle v ∈ V \ {s,t} gilt: $\sum_{u \in V:(u,v) \in A} f(u,v) = \sum_{u \in V:(v,u) \in A} f(v,u)$
- Der Wert eines Flusses f ist definiert als: val(f) := netoutflow(s) := $\sum_{u \in V:(s,u) \in A} f(s,u) - \sum_{u \in V:(u,s) \in A} f(u,s)$

Flüsse in Netzwerken

- Der Nettozufluss der Senke t ist gleich dem Wert des Flusses, dh.
  netinflow(t) := $\sum_{u \in V:(u,t) \in A} f(u,t) - \sum_{u \in V:(t,u) \in A} f(t,u)$ = val(f)

Schnitte

- Ein s-t-Schnitt für ein Netzwerk (V,A,c,s,t) ist eine Pertition (S,T) von V mit s ∈ S und t ∈ T. Die Kapazität eines s-t-Schnitts (S,T) ist durch
  cap(S,T) := $\sum_{(u,w)\in(S\times T)\cap A} c(u,w)$ definiert. Wobei Partition $(S,T) : S \cup T = V$ und $S \cap T = \varnothing$

- Die Kapazität eines Schnitts (S,T) ignoriert die Kanten von T nach S

- Ist f ein Fluss und (S,T) ein s-t-Schnitt in einem Netzwerk (V,A,c,s,t) so gilt val(f) ≤ cap(S,T)

val(f) ≤ cap(S,T) - Beweis

- Sei f(U,W) := $\sum_{(u,w)\in(U\times W)\cap A} f(u,w)$
- $val(f) \overset{(1)}{=} f(S,T) - f(T,S) \overset{(2)}{\leq} f(S,T) \overset{(3)}{\leq} cap(S,T)$
  - (2) Nichtnegativität Fluss
  - (3) Kapazitätsbeschränkung f(u,w) ≤ c(u,w)
  - (1) für S = {s} per Def. und für T = {t} wg. netinflow(t) = val(t)

Restnetzwerke

- Sei N = (V,A,c,s,t) ein Netzwerk ohne entgegen gerichtete Kanten und sei f ein Fluss in N. Das restnetzwerk N_f := (V,A_f,r_fs,t) ist wie folgt definiert:
  - Ist e ∈ A mit f(e) < c(e), dann ist e auch iene Kante in A_f, mit $r_f(e)$:= c(e) - f(e)
  - Ist e ∈ A mit f(e) > 0, dann ist e^{opp} in A_f, mit $r_f(e^{opp})$ = f(e)
  - Nur oben beschirebene Kanten sind in A_f
- $r_f(e)$, e ∈ A_f, nennen wir die restkapazität der Kante e

Maximaler Fluss

- Ein Fluss f in einem Netzwerk N ist ein maximaler Fluss gdw. es im Restnetzwerk N_f keinen gerichteten Pfad von der Quelle s zur Senke t gibt. Für jeden solchen maximalen Fluss gibt es einen s-t-Schnitt (S,T) mit val(f) = cap(S,T)

- Es gibt einen s-t-Pfad ⇒ wir können entlang dieses Pfades augmentieren, Fluss war nicht maximal

- Es gibt keinen s-t-Pfad ⇒zeigen, dass es s-t-Schnitt mit cap(S,T) = val(f) gibt

Maximaler Fluss

- S :=. alle von s erreichbaren Knoten im Restnetzwerk N_f
- T:= V\S
- e = (u,v) ∈ A mit u ∈ S und v ∉ S dann keine Kante in A_f von u nach v und f(e) = c(e
- val(f) = f(S,T) - f(T,S) = cap(S,T) - 0 = cap(S, T) ⇒ Fluss ist maximal

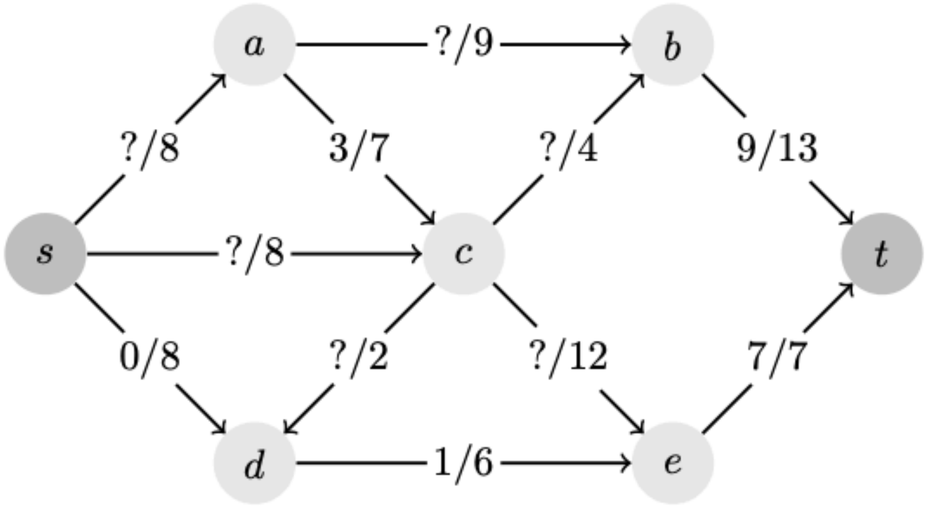
Ford-Fulkerson

```
FORD-FULKERSON(V, A, c, s, t)
1: f ← 0                                          ▷ Fluss konstant 0
2: while ∃ s-t-Pfad P in (V, A_f) do              ▷ augmentierender Pfad
3:     Erhöhe den Fluss entlang P                 ▷ wie in Beweis zu Satz 3.11
4: return f                                       ▷ maximaler Fluss
```

- In ganzzahligem Netzwerk O(mnU), wobei U maximales Kantenkapazität darstellt
- Capacity Scaling in O(mn*(1+log(U))
- Dynamic Trees in (O(mn*log(n))

Aufgaben

(a) Fill in the missing values so that they form a feasible (not necessarily maximum) flow:



(b) Construct the residual network ("Restnetzwerk").

(c) Find an augmenting s-t-path and augment the flow along this path. If necessary, repeat this step until you have found a maximum flow.

(d) Prove that your flow is a maximum flow by finding a minimum cut in N.

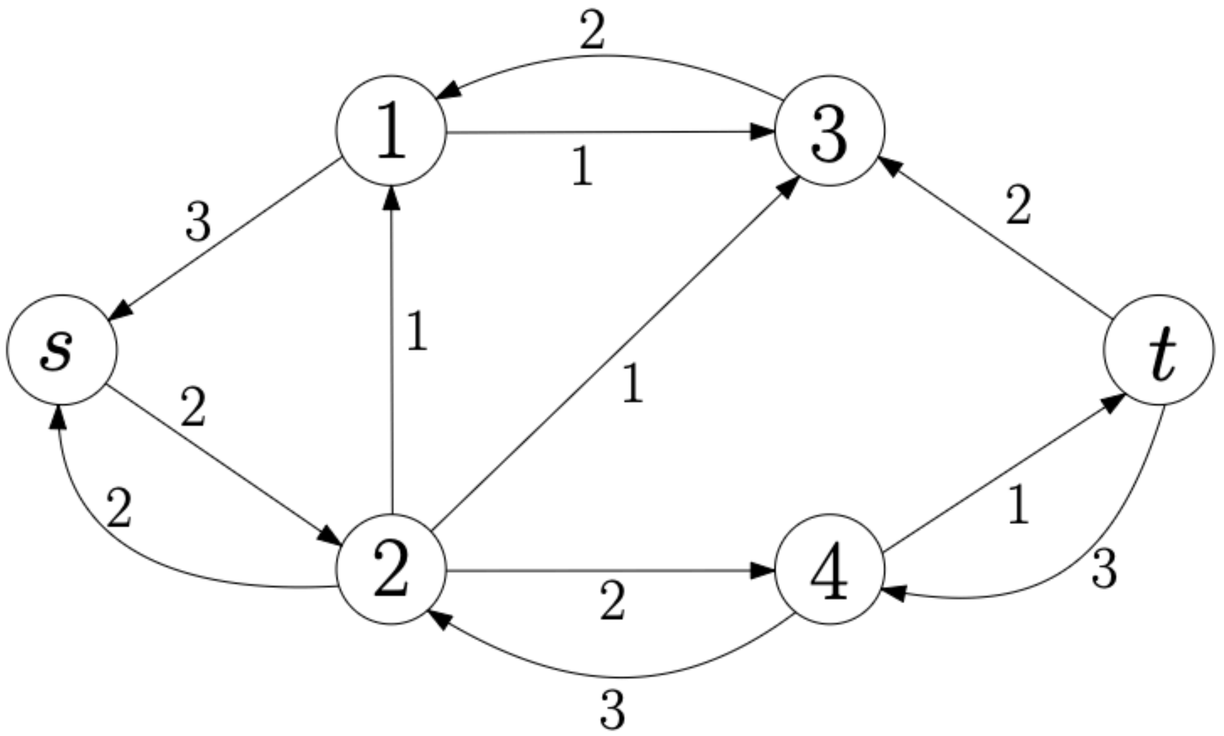
Aufgaben

(b) Sei f ein maximaler Fluss in einem Netzwerk $N = (V, A, c, s, t)$. Dann gibt es keine zwei Knoten $u, v \in V$, sodass sowohl die Kante (u, v) als auch die Kante (v, u) im Restnetzwerk N_f vorkommt.

WAHR ☐ FALSCH ☐

Begründung/Gegenbeispiel: *(2 Punkte)*

Sei N ein Netzwerk ohne entgegengesetzte Kanten und sei f ein Fluss in G. Unten abgebildet sehen Sie das Restnetzwerk R_f.



(a) Ist f maximal? Geben Sie eine kurze Begründung für Ihre Antwort. *(1 Punkte)*

(b) Rekonstruieren sie N und f.