AnW – Woche 12

16.05.24
Nicolas Wehrli

Heute:

I. Kurze Kommentare zur TS
II. Theory Recap
- Flüsse
- Maxflow-Mincut Theorem
- Ford-Fulkerson
- Quiz Aufgaben
- Flüsse Anwendungen
  - Matching
  - (Kantendisjunkte Pfade
  - Bildsegmentierung)

III. Aufgabe
~~IV. Kahoot~~

Interesse an Big Kahoot ?

I. Kurze Kommentare zur TS

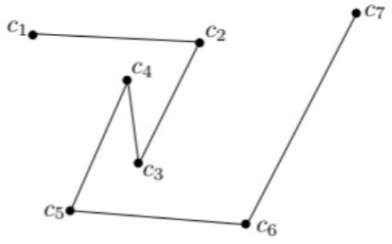
Sehr schwierige Aufgabe

Niemand hatte 2 Punkte

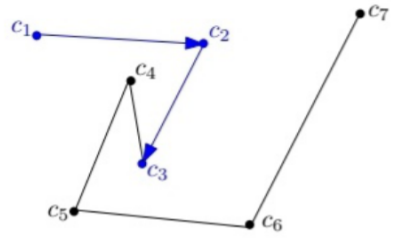
Weshalb müssen $(p_1, \ldots, p_n)$ sortiert sein für LocalRepair?

LocalRepair$(p_1, p_2, \ldots, p_n)$ — $(p_1, p_2, \ldots, p_n)$ sortiert

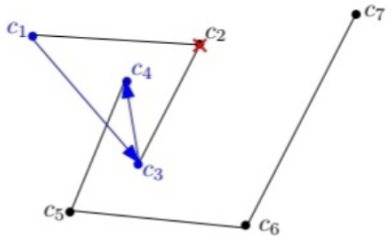
```
1:  q_0 ← p_1; h ← 0
2:  for i ← 2 to n do                ▷ unterer Rand, links nach rechts
3:      while h > 0 und q_h links von q_{h-1} p_i do
4:          h ← h − 1
5:      h ← h + 1; q_h ← p_i
6:              ▷ (q_0, ..., q_h) untere konvexe Hülle von {p_1, ..., p_i}
7:  h' ← h
8:  for i ← n − 1 downto 1 do        ▷ oberer Rand, rechts nach links
9:      while h > h' und q_h links von q_{h-1} p_i do
10:         h ← h − 1
11:     h ← h + 1; q_h ← p_i
12: return (q_0, q_1, ..., q_{h-1})
```



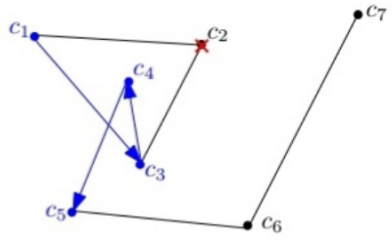
Point set C



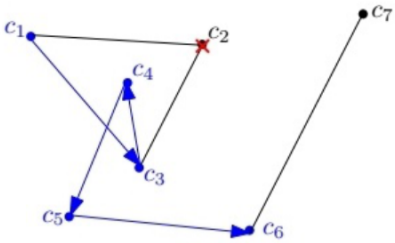
Iteration 1: Point c_2 is discarded



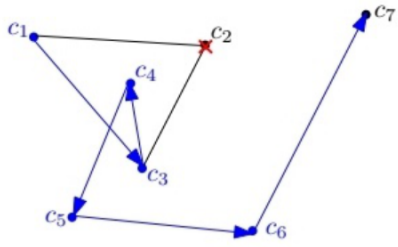
Iteration 2: Point c_4 is added, no point discarded



Iteration 3: Point c_5 is added, no point discarded



Iteration 3: Point c_6 is added, no point discarded



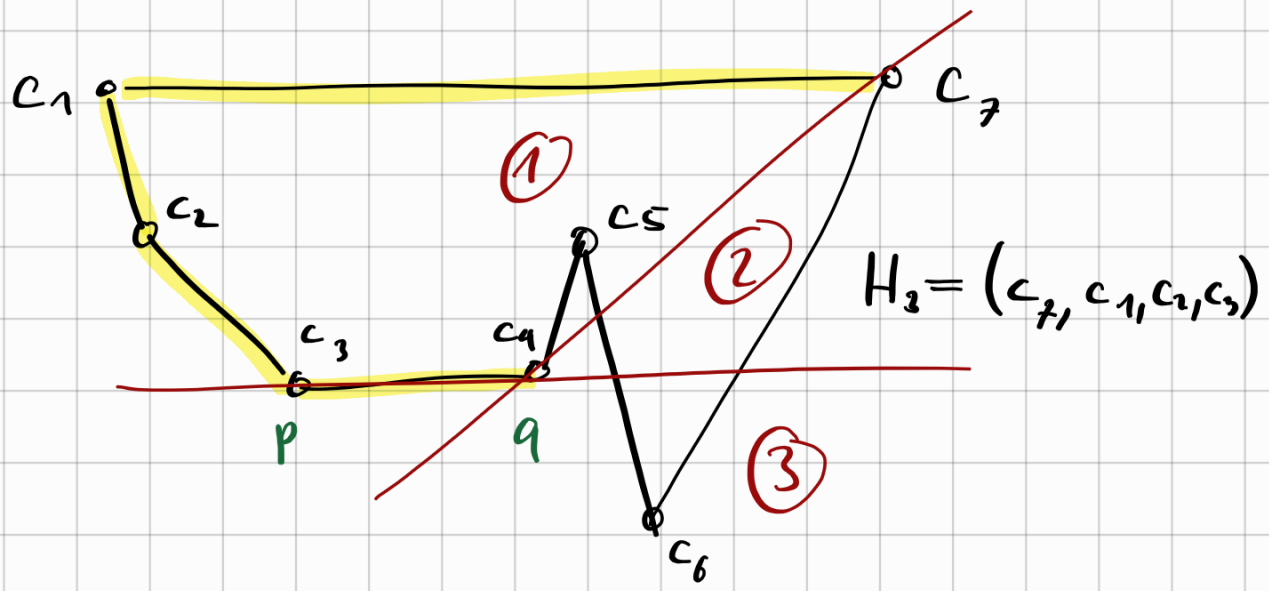
Iteration 3: Point c_7 is added, no point discarded

The final lower convex hull is $c_1, c_3, c_4, c_5, c_6, c_7$ which intersects itself, and is a wrong solution.

Alte Falsche Lösung

(b) Sei $C = \{c_1, c_2, \ldots, c_n\}$ eine Menge von Punkten, die eine nicht-überschneidende Kurve bilden, das heisst, das Segment (c_i, c_{i+1}) schneidet maximal zwei weitere Segmente: Segment (c_{i-1}, c_i) bei c_i (falls $i \geq 2$), und Segment (c_{i+1}, c_{i+2}) bei c_{i+1} (falls $i \leq n-2$). Wir nehmen ausserdem an, dass $c_1(x) < c_i(x)$ für $1 < i \leq n$ und $c_i(x) < c_n(x)$ für $1 \leq i < n$. Berechnen Sie die konvexe Hülle von C in Zeit $O(n)$.

Wir benutzen den Fakt, dass sich der Streckenzug $c_1, c_2, \ldots, c_n$ nicht schneidet.



In jeder Iteration des Algorithmus, ist

p der zweitletzte Punkt der bisherigen konvexen Hülle

q der letzte Punkt der bisherigen konvexen Hülle.

Case distinction:

for the next point c_i:

if c_i left of (c_1, c_2): skip to next

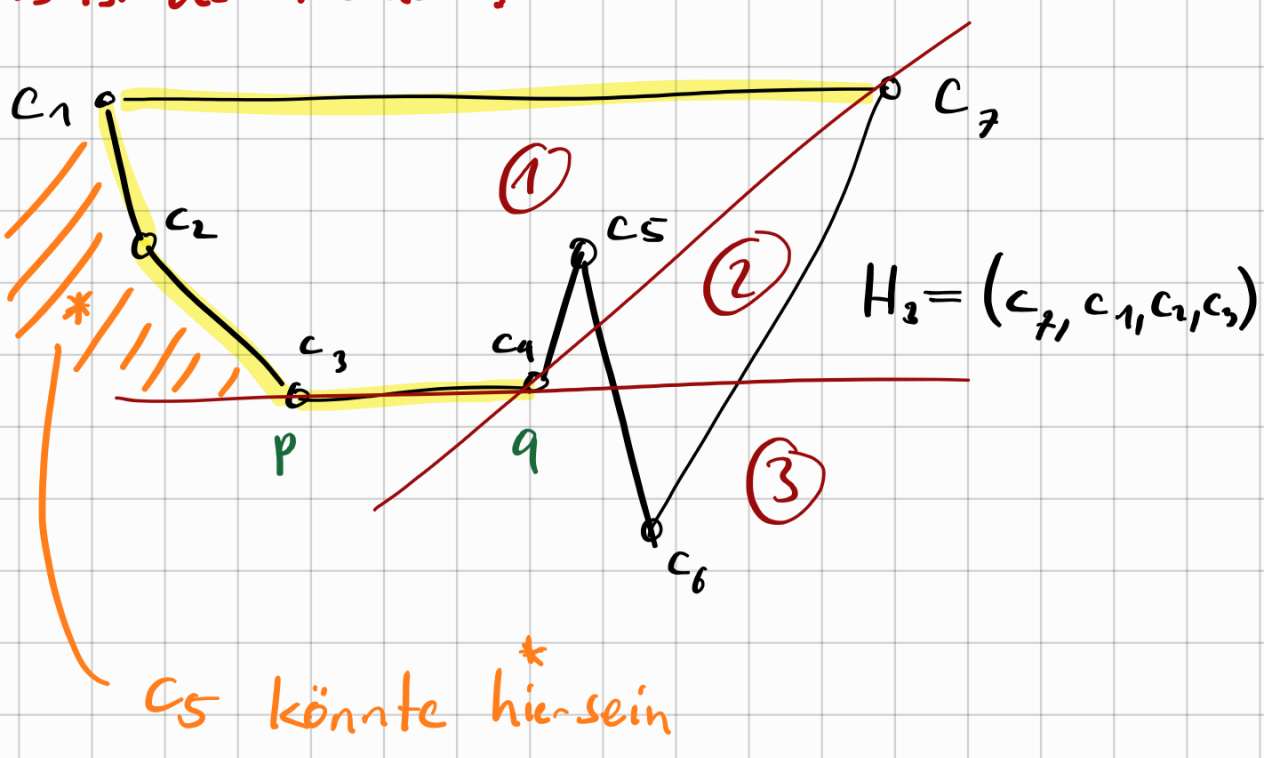
else

① left of (p, q), left of (q, c_2) } skip to next

② left of (p, q), right of (q, c_2) } add c_i

③ right of (p, q) } $q \leftarrow p$; $p \leftarrow$ predecessor of p; restart iteration for c_i

Was ist der Fehler?

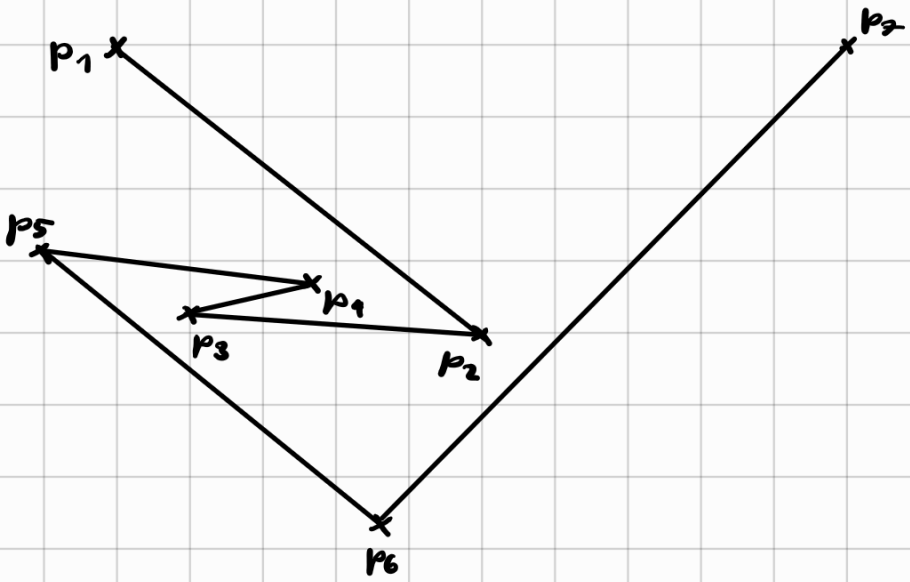


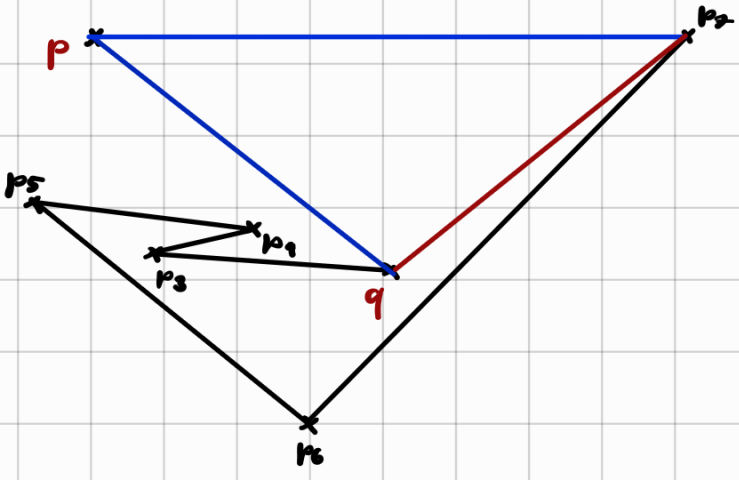
Aber wie? Der Streckenzug schneidet sich nicht per Annahme.

↝ Das gilt nur für den ursprünglichen Streckenzug.

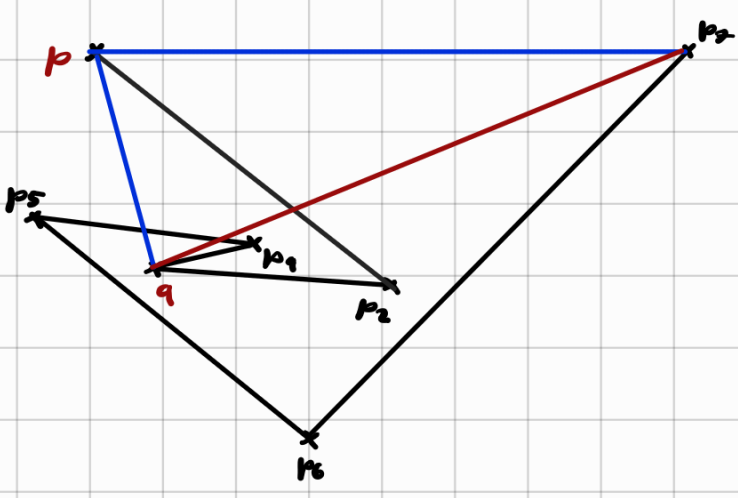
Inzwischen haben wir Abkürzungen gemacht, die den ursprünglichen Streckenzug durchstossen können.

Gegenbsp.

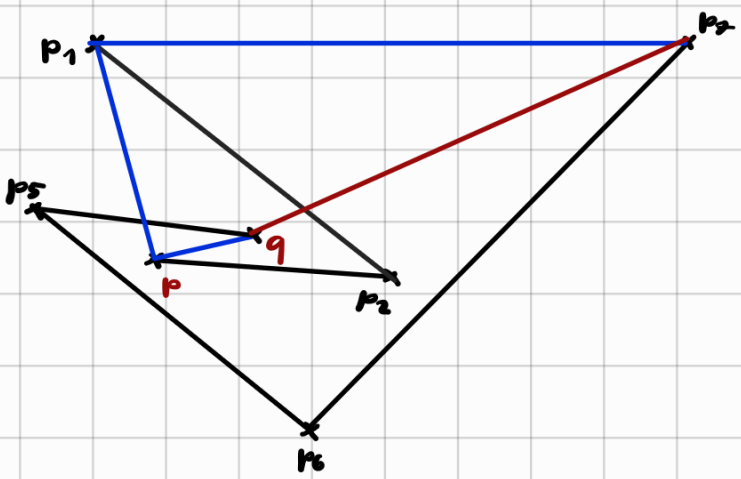




p_3 rechts von pq



p_4 links von pq
rechts von $q\,p_2$



p_5 links von pq
links von $q\,p_2$

aber p_5 nicht in bisherigen Polygon! ↯

Flüsse

Eher eines der einfacheren Themen.
Findet aber sehr viel Anwendung.

Ein Netzwerk ist ein Tupel $\mathcal{N} = (V, A, c, s, t)$, wobei
- (V, A) ein gerichteter Graph *
- $s \in V$, die Quelle (Source)
- $t \in V \setminus \{s\}$, die Senke (Sink)
- $c: A \to \mathbb{R}_0^+$, die Kapazitätsfunktion

* auf den Slides steht 'ohne Schleifen'. Diese Zusatzbedingung simplifiziert die Herleitung und die Beweise, ist aber nicht notwendig.

Mögliche Anwendungen: Matchings, Geld-/Verkehrs-/Stromflüsse, Bildsegmentierung

Gegeben sei ein Netzwerk $\mathcal{N} = (V, A, c, s, t)$.
Ein Fluss in $\mathcal{N}$ ist eine Funktion $f: A \to \mathbb{R}$ mit den Bedingungen
- Zulässigkeit: $0 \leq f(e) \leq c(e)$ für alle $e \in A$
- Flusserhaltung: Für alle $v \in V \setminus \{s,t\}$ gilt:

$$\sum_{u \in V : (u,v) \in A} f(u,v) = \sum_{u \in V : (v,u) \in A} f(v,u)$$

In-flow zu v | Out-flow von v

Der **Wert** eines Flusses f ist durch

$$val(f) := netoutflow(s) = \sum_{u \in V:\,(s,u)\in A} f(s,u) - \sum_{u \in V:\,(s,u)\in A} f(u,s)$$

definiert.

Der **Nettozufluss** der Senke t gleicht $val(f)$:

$$netinflow(t) := \sum_{u \in V:\,(u,t)\in A} f(u,t) - \sum_{u \in V:\,(t,u)\in A} f(t,u) = val(f)$$

$$\begin{aligned}
0 &= \sum_{(v,u)\in A} f(v,u) - \sum_{(u,v)\in A} f(u,v) \\
&= \sum_{v\in V} \underbrace{\left(\sum_{u\in V:\,(v,u)\in A} f(v,u) - \sum_{u\in V:\,(u,v)\in A} f(u,v) \right)}_{=0 \text{ für } v\notin\{s,t\}} \\
&= \underbrace{\left(\sum_{u\in V:\,(s,u)\in A} f(s,u) - \sum_{u\in V:\,(u,s)\in A} f(u,s) \right)}_{=val(f)} \\
&\quad + \underbrace{\left(\sum_{u\in V:\,(t,u)\in A} f(t,u) - \sum_{u\in V:\,(u,t)\in A} f(u,t) \right)}_{=-netinflow(t)}
\end{aligned}$$

Problemstellung: Suche nach dem Maxflow.

Gegeben $N = (V, A, c, s, t)$ finde den Fluss f in N mit dem grössten Wert ($val(f)$ maximal).

Es stellen sich folgende Fragen:

- Gibt es immer einen solchen maximalen Fluss? Es könnte ein ähnliches Phänomen auftreten wie beim offenen Intervall $(0, 1)$: Es gibt keine grösste Zahl, kein Maximum.
- Wie erkennt man, dass ein Fluss maximal ist? Was ist ein „einfacher“ Beweis für die Maximalität eines Flusses?

Um diese Fragen zu beantworten, brauchen wir vorerst noch eine Definition.

Ein s-t-Schnitt für ein Netzwerk $N = (V, A, c, s, t)$ ist eine Partition (S, T) von V (i.e. $S \cup T = V$ und $S \cap T = \emptyset$) mit $s \in S$ und $t \in T$.

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

Nur die Kanten von S zu T!

Lemma: Sei f ein Fluss und (S,T) ein s-t-Schnitt in einem Netzwerk $N=(V,A,c,s,t)$, so gilt

$$val(f) \leq cap(S,T)$$

Beweis:

Def: Für eine Partition (U,W) von V sei

$$f(U,W) := \sum_{(u,w)\in(U\times W)\cap A} f(u,w)$$

Wir zeigen zuerst $val(f) = f(S,T) - f(T,S)$ (i)

$$val(f) = \sum_{u\in V:(s,u)\in A} f(s,u) - \sum_{u\in V:(u,s)\in A} f(u,s) \qquad \text{(per Def. } val(f)\text{)}$$

$$= \sum_{u\in V:(s,u)\in A} f(s,u) - \sum_{u\in V:(u,s)\in A} f(u,s)$$

$$+ \sum_{v\in S\setminus\{s\}} \left(\sum_{u\in V:(v,u)\in A} f(v,u) - \sum_{u\in V:(u,v)\in A} f(u,v) \right)$$

$t\notin S$

da $t,s\notin S\setminus\{s\}$ gilt: $=0$ $\forall v\in S\setminus\{s\}$

$$= \sum_{v\in S} \left(\sum_{u\in V:(v,u)\in A} f(v,u) - \sum_{u\in V:(u,v)\in A} f(u,v) \right) = T_v = \sum_{v\in S} T_v$$

$=0$ für $v\neq s$

Für jede Kante $(x,y) \in A$ mit $x,y \in S$ gilt:

$f(x,y)$ erscheint 1-mal positiv in T_x

und 1-mal negativ in T_y.

$\Rightarrow$ Wir können alle Kanten $(x,y) \in A$ mit beiden Endpunkten in S weglassen.

$$= \sum_{(u,w) \in (S \times T) \cap A} f(u,w) - \sum_{(u,w) \in (T \times S) \cap A} f(u,w)$$

$$= f(S,T) - f(T,S) \qquad \square \text{ (i)}$$

Da per Zulässigkeit $f(T,S) \geq 0$ und

$f(S,T) \leq cap(S,T)$ gilt, folgt

$$val(f) \overset{\text{(i)}}{=} f(S,T) - f(T,S) \leq f(S,T) \leq cap(S,T)$$

$\square$

Maxflow-Mincut-Theorem

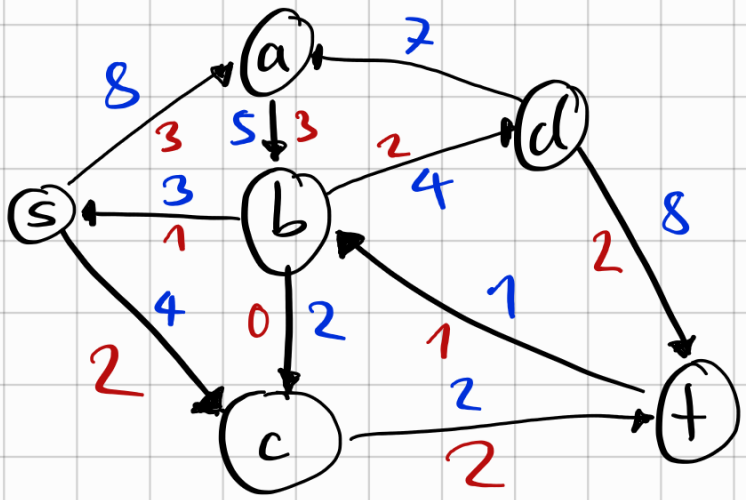
Satz 3.9 („Maxflow-Mincut Theorem"). Jedes Netzwerk $N = (V, A, c, s, t)$ erfüllt

$$\max_{f \text{ Fluss in } N} \text{val}(f) = \min_{(S,T) \text{ s-t-Schnitt in } N} \text{cap}(S, T)$$

Wir beweisen diese Aussage nur für Netzwerke mit ganzzahligen Kapazitäten und ohne entgegengesetzte Kanten.

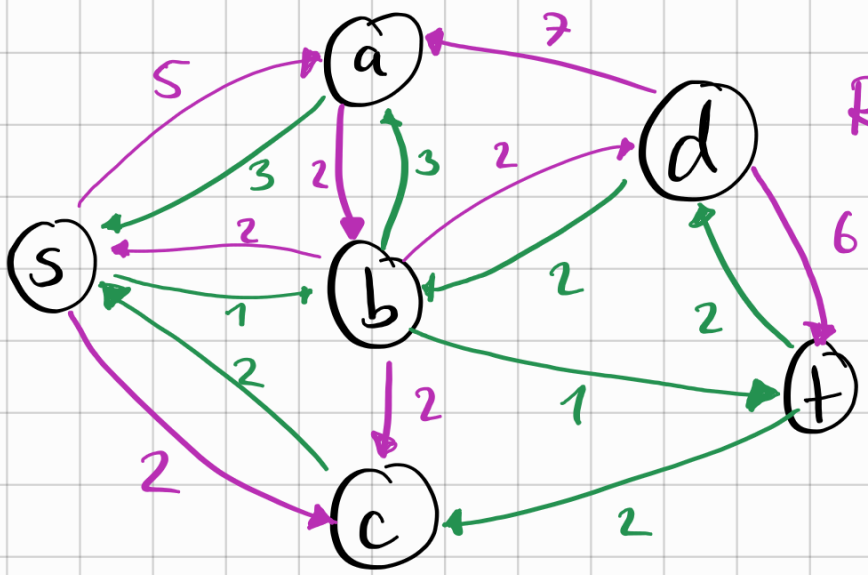
Es ist ein konstruktiver Beweis.

Zuerst ein Beispiel:



Kapazität c
Fluss f

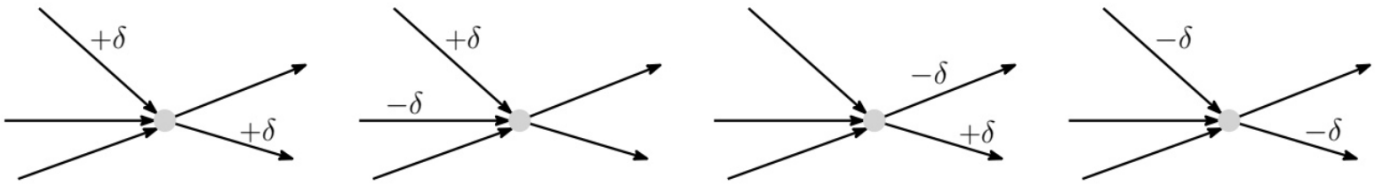
Restnetzwerk:



Restkapazität von e:
$c(e) - f(e)$

Fluss den wir reduzieren könnten.

Lokale Veränderungen des Flusses, die die Flusserhaltung erhalten:



Unter Beachtung { der Kapazität bei „$+\delta$“, und des aktuellen Flusses bei „$-\delta$“.

Das Restnetzwerk modelliert die möglichen Veränderungen unter Flusserhaltung.

Notation: Für eine Kante $e=(u,v)$ bezeichnen wir die entgegengesetzte Kante (v,u) mit e^{opp}.

Sei $\mathcal{N}=(V,A,c,s,t)$ ein Netzwerk ohne entgegengerichtete Kanten und sei f ein Fluss in $\mathcal{N}$.

Dann ist das Restnetzwerk $\mathcal{N}_f=(V,A_f,r_f,s,t)$ wie folgt definiert:

1. Für jedes $e\in A$ mit $f(e)<c(e)$, gilt: $e\in A_f$ mit $r_f(e):=c(e)-f(e)$.

2. Für jedes $e\in A$ mit $f(e)>0$, gilt: $e^{opp}\in A_f$ mit $r_f(e^{opp}):=f(e)$

3. Es befinden sich nur Kanten von (1) und (2) in A_f.

$r_f(e)$ für $e\in A_f$ nennen wir Restkapazität der Kante e.

Satz

Sei N ein Netzwerk (ohne entgegegen gerichtete Kanten).

Ein Fluss f ist maximaler Fluss

$\Leftrightarrow$

es im Restnetzwerk N_f keinen gerichteten s-t-Pfad gibt.

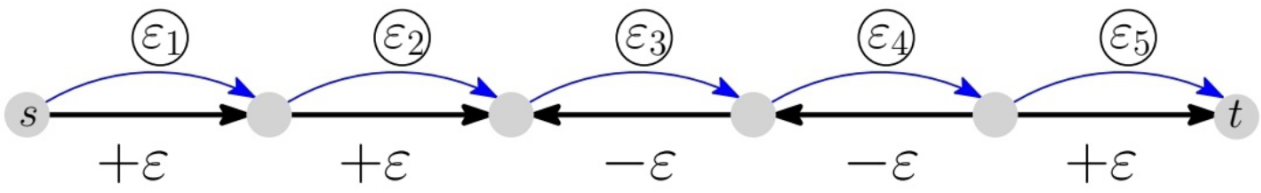
Für jeden maximalen Fluss f

gibt es einen s-t-Schnitt (S, T) mit $\text{val}(f) = \text{cap}(S, T)$.

Beweis:

1. N_f besitzt einen s-t Pfad:

Wir betrachten einen gerichteten s-t-Pfad in N_f:



Bestimme die kleinste Restkapazität $\varepsilon := \min_i \varepsilon_i$

Augmentiere f entlang des Pfades um ε.

$\Rightarrow$ f nicht maximal

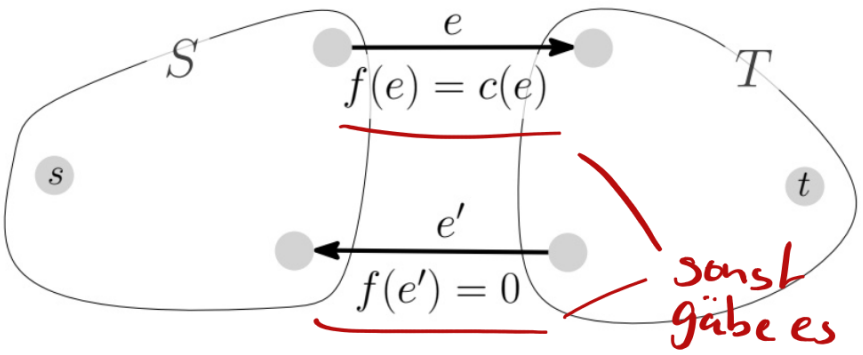
Per Kontraposition folgt:

f maximal $\Rightarrow \nexists$ s-t Pfad in N_f

2. N_f besitzt kein s-t Pfad

$S :=$ in N_f von s aus erreichbare Knoten; $T := V \setminus S$.

s von s aus in N_f erreichbar $\Rightarrow s \in S$
t von s aus nicht erreichbar $\Rightarrow t \notin S$
$\Rightarrow (S, T)$ ist s-t-Schnitt.



sonst gäbe es einen s-t-Pfad.

$f(S, T) = \text{cap}(S, T)$

$f(T, S) = 0$

$$\text{val}(f) \overset{(i)}{=} f(S,T) - \underbrace{f(T,S)}_{=0} = f(S,\bar{T}) = \text{cap}(S,\bar{T})$$

Per Lemma $\text{val}(f) \leq \text{cap}(S,T)$ folgt:

$\Longrightarrow$ f ist maximal. ☐

Wir haben nun gezeigt:

Für jeden maximalen Fluss existiert ein S-T-Schnitt, s.d. $\text{val}(f) = \text{cap}(S,T)$.

Aber existiert immer ein maximaler Fluss?

Auch begrenzte Mengen sind nicht eindeutig.

Bsp: $[0, 1)$ hat kein Maximum.

Konstruktiver Beweis per Algorithmus:

Ford Fulkerson

Ford-Fulkerson(V, A, c, s, t)

1: $f \leftarrow \mathbf{0}$ ▷ Fluss konstant 0
2: **while** $\exists$ s-t-Pfad P in N_f **do** ▷ augmentierender Pfad
3: Augmentiere den Fluss entlang P
4: **return** f ▷ maximaler Fluss

- Wir können nicht garantieren, dass der Algorithmus terminiert.
- Der Algorithmus kann bei Kapazitäten aus $\mathbb{R}$ unendlich laufen.
- Bei Kapazitäten aus $\mathbb{N}_0$ bleiben im Algorithmus Flüsse und Restkapazitäten ganzzahlig. In jedem Augmentierungsschritt wird der Fluss ganzzahlig ≥ 1 verbessert. D.h. insbesondere auch, dass das Ergebnis **ganzzahlig** $(A \to \mathbb{N}_0)$ ist.

Analyse:

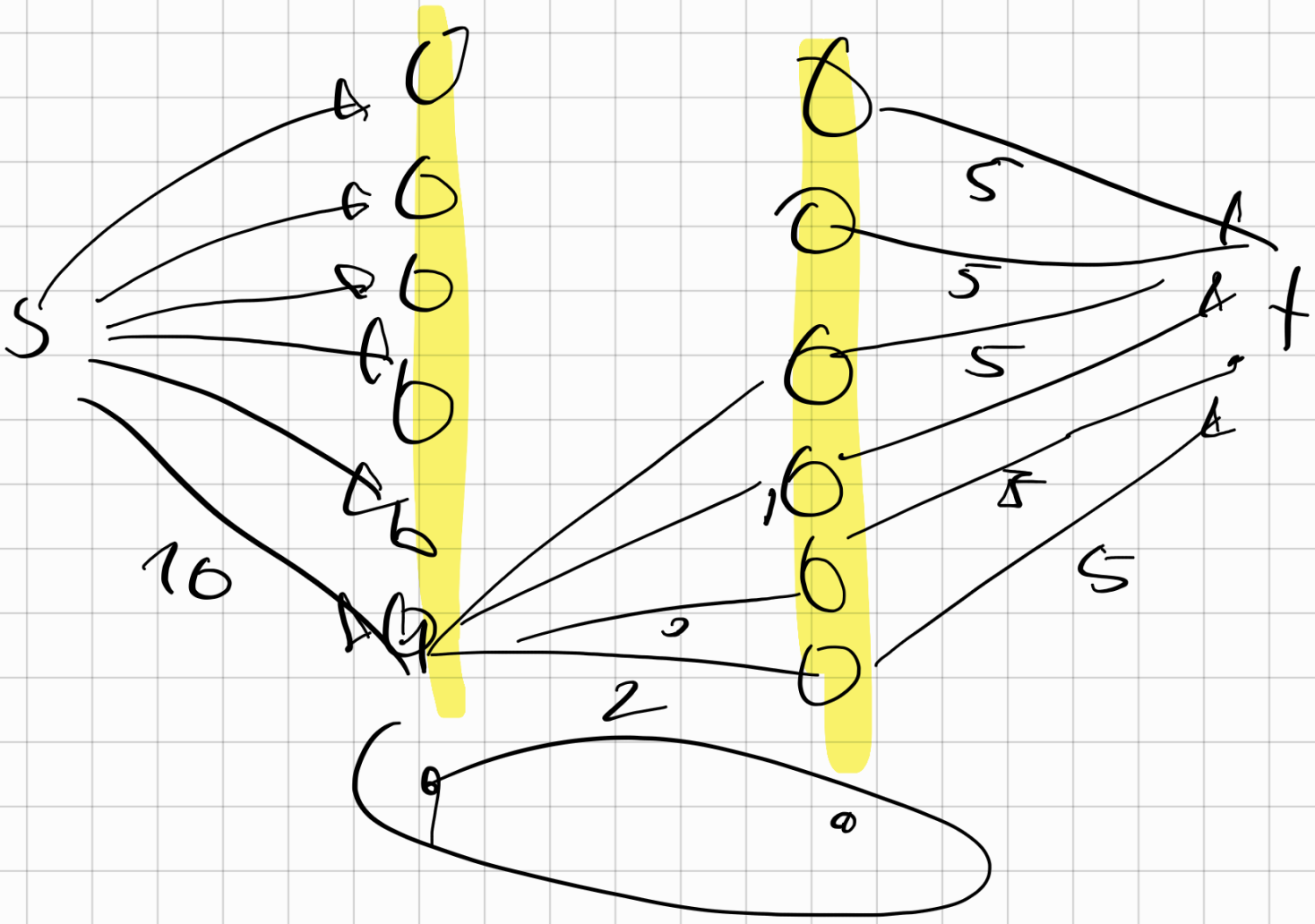
Sei $n := |V|$ und $m := |A|$ für Netzwerk $N = (V, A, c, s, t)$.

- Angenommen $c\colon A \to \mathbb{N}_0$ und $U := \max_{e \in A} c(e)$. Dann gilt
$$\text{val}(f) \leq \text{cap}(\{s\}, V \setminus \{s\}) \leq (n-1)U$$
und es gibt **höchstens** $(n-1)U$ **Augmentierungsschritte**.
- **Ein Augmentierungsschritt**
Suche s-t-Pfad in N_f, Augmentieren, Aktualisierung von N_f **benötigt** $O(m)$ **Zeit**.

Satz (Ford-Fulkerson mit ganzzahligen Kapazitäten)

Sei $N = (V, A, c, s, t)$ ein Netzwerk mit $c : A \to \mathbb{N}_0^{\leq U}$, $U \in \mathbb{N}$, ohne entgegen gerichtete Kanten.[2] *Dann gibt es einen ganzzahligen maximalen Fluss. Er kann in Zeit $O(mnU)$ berechnet werden.*

Somit haben wir konstruktiv das Maxflow-Mincut Theorem für ganzzahlige Kapazitäten und ohne Beachtung entgegengerichteter Kanten.



f ganzahlig $\iff$ $f : A \to \mathbb{N}_0$

$\iff$ f(e) ganzahlig für alle $e \in A$.

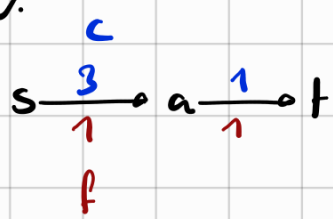
Quiz Aufgaben

Betrachten Sie ein Netzwerk $N = (V, A, c, s, t)$ ohne entgegegen gerichtete Kanten, einen Fluss f auf N, und das dazugehörige Restnetzwerk N_f. Wir sagen, dass ein Knoten $v \in V$ erreichbar ist, wenn es einen Pfad von s zu v in N_f gibt. Wir bezeichnen X die Menge der erreichbaren Knoten. Welche der folgenden Aussagen treffen immer zu?

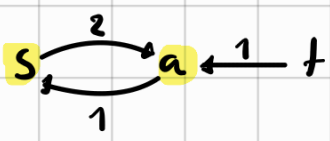
abhängig von f

True	False	
○	○	$t \in X$.

N:



N_f:

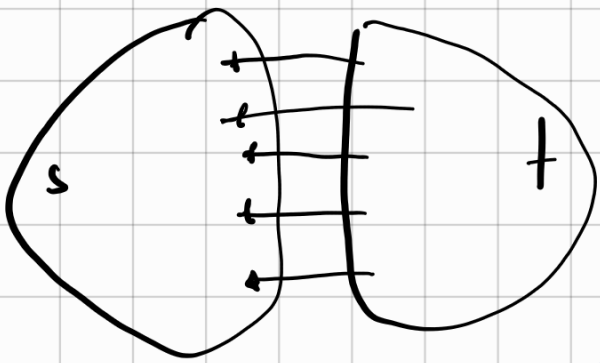


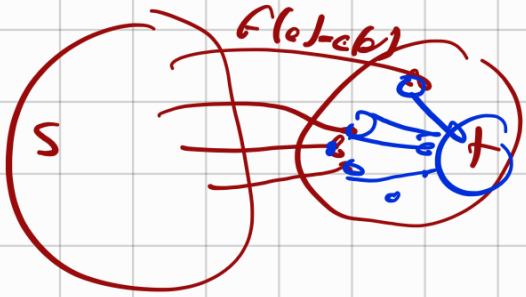
$X = \{s, a\} \Rightarrow t \notin X$

○	○	Falls $(X, V \setminus X)$ ein s-t Schnitt ist, dann gilt $\text{val}(f) = \text{cap}(X, V \setminus X)$.

$(X, V\setminus X)$ - s-t Schnitt $\Rightarrow$ $s \in X$, $t \notin X$.

$\Rightarrow$ In N_f:





$cap(X, V\setminus X)$ - net influent

$\Rightarrow$ $\forall (u,w) \in A$ st. $u \in X, w \in V\setminus X$: $f((u,w)) = c((u,w))$

$\forall (u,w) \in A$ st. $u \in V\setminus X$, $w \in X$: $f((u,w)) = 0$

$val(f) \geq cap(X, V\setminus X)$

per Maxflow - Mincut gilt für alle s-t Schnitte (S,T)

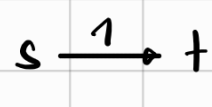
$val(f) \leq cap(S,T)$

$\Rightarrow val(f) = cap(X, V \setminus X)$

- ○ ○ $(X, V \setminus X)$ ist ein s-t Schnitt.

N:

N_f:



$\Rightarrow X = \{s,t\}$

$\Rightarrow$ kein s-t Schnitt

Sei $N = (V, A, c, s, t)$ ein Netzwerk ohne entgegen gerichtete Kanten und f ein Fluss so dass, $val(f) > 0$. Sei N_f das Restnetzwerk.

N_f enthält einen (gerichteten) Weg von t nach s.

- ○ Wahr
- ○ Falsch

$val\ f > 0 \Rightarrow \exists$ s-t Pfad P s.d. $\forall e \in P: f(e) > 0$

$\Rightarrow \forall e \in P \quad r(e^{orr}) = f(e) > 0$

$\Rightarrow \exists$ t-s Pfad in N_f mit $r(e^{orr}) > 0$.

$\emptyset$

Intuitiv: Restnetzwerk gibt uns für jede Kante, wie wir diesen Fluss ändern können.

Da etwas von s zu t fliesst, könnten wir auf jeder Kante

Sei $N = (V, A, c, s, t)$ ein Netzwerk ohne entgegen gerichtete Kanten und so, dass $c(e) > 0$ für alle Kanten $e \in A$. Sei f der 0-Fluss, d.h. $f(e) = 0, \forall e \in A$. Dann ist das Restnetzwerk $N_f = (V, A_f, r_f, s, t)$ dasselbe wie N.

Bemerkung: hier "ist das Restnetzwerk $N_f = (V, A_f, r_f, s, t)$ dasselbe wie N" bedeutet dass $A = A_f, c = r_f$.

- ○ Wahr
- ○ Falsch

Wahr

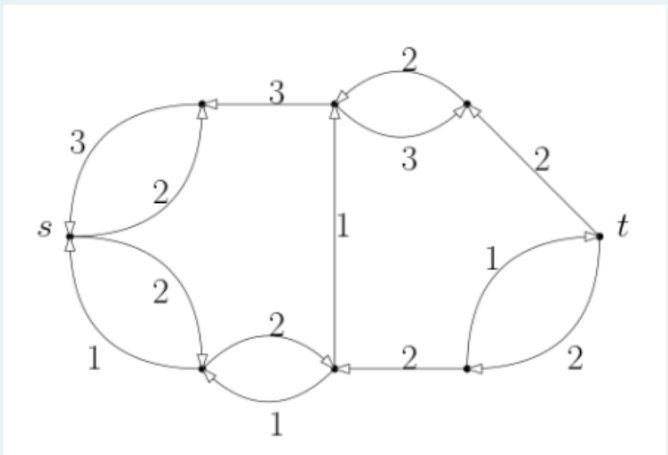
Sei $\mathcal{N} = (V, A, c, s, t)$ ein Netzwerk ohne entgegengerichtete Kanten und sei f ein Fluss in $\mathcal{N}$.

Dann ist das Restnetzwerk $\mathcal{N}_f = (V, A_f, r_f, s, t)$ wie folgt definiert:

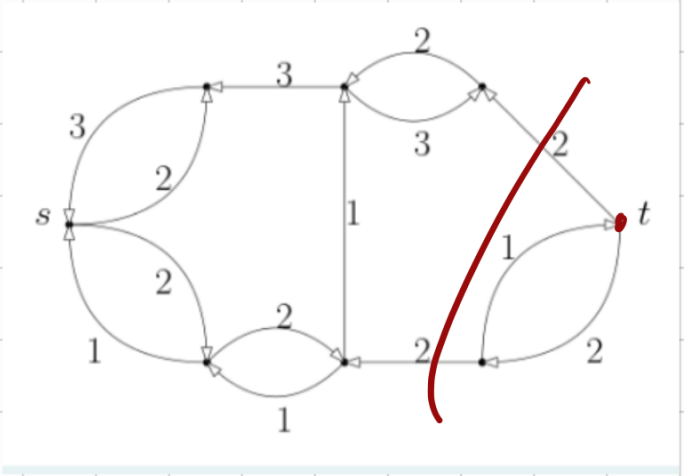
1. Für jedes $e \in A$ mit $f(e) < c(e)$, gilt: $e \in A_f$ mit $r_f(e) := c(e) - f(e)$.
2. Für jedes $e \in A$ mit $f(e) > 0$, gilt: $e^{opp} \in A_f$ mit $r_f(e^{opp}) := f(e)$
3. Es befinden sich nur Kanten von (1) und (2) in A_f.

$r_f(e)$ für $e \in A_f$ nennen wir Restkapazität der Kante e.

Sei $N = (V, A, c, s, t)$ ein Netzwerk ohne entgegen gerichtete Kanten und f ein Fluss. Das Restnetzwerk N_f ist wie folgt gegeben:



Ist der Fluss maximal?



Wahr. Von t schauen hilft.

Jedes Netzwerk erfüllt
$\max_{f \text{ Fluss}} \text{val}(f) = \min_{(S,T) \text{ s-t-Schnitt}} \text{cap}(S,T)$

1. Wahr – Maxflow-MinCut

Es gibt einen s-t-Schnitt (S,T) und einen Fluss f, so dass
$\text{val}(f) \geq \text{cap}(S,T)$

2. Wahr – folgt aus 1.

Jedes Netzwerk erfüllt
$\min_{f \text{ Fluss}} \text{val}(f) = \max_{(S,T) \text{ s-t-Schnitt}} \text{cap}(S,T)$

3. Falsch – $\min \text{val}(f) = 0 \neq \text{cap}(S,T)$ wenn $\exists$ s-t Pfad in N.

Es gibt einen s-t-Schnitt (S,T) und einen Fluss f, so dass
$\text{val}(f) > \text{cap}(S,T)$

4. Falsch – Maxflow-Mincut

Matching (mit Flüssen)

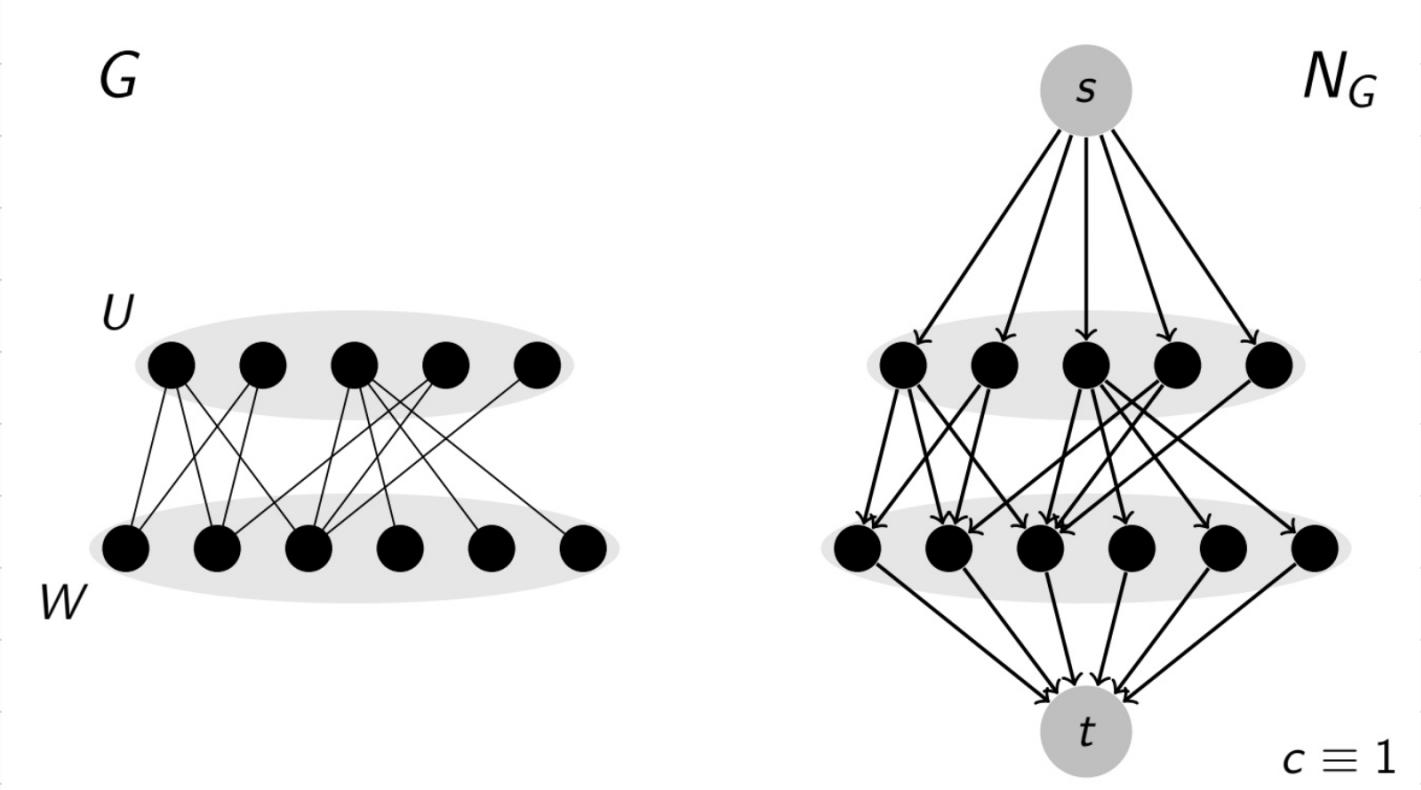
Wir betrachten ungerichtete, ungewichtete Graphen.
Zur Erinnerung:

Eine **Kantenmenge** $M \subseteq E$ heisst **Matching**, falls kein Knoten des Graphen zu mehr als einer Kante aus M inzident ist, d.h. wenn

$$e \cap f = \emptyset \quad \text{für alle } e, f \in M \text{ mit } e \neq f$$

Sei $G = (U \dot\cup W, E)$ bipartit.
Dann definiere wir das Netzwerk $\mathcal{N}_G = (U \dot\cup W \dot\cup \{s,t\}, A, c, s, t)$

- $s \neq t$, $(s,t \notin U$ und $s,t \notin W)$
- $A := \{s\} \times U \cup \{(u,w) \in U \times W \mid \{u,w\} \in E\} \cup W \times \{t\}$
- $c(e) := 1 \quad \forall e \in A$



Von letzter Woche wissen wir:

Satz (Ford-Fulkerson mit ganzzahligen Kapazitäten)

Sei $N = (V, A, c, s, t)$ ein Netzwerk mit $c : A \to \mathbb{N}_0^{\leq U}$, $U \in \mathbb{N}$, ohne entgegen gerichtete Kanten.[2] Dann gibt es einen ganzzahligen maximalen Fluss. Er kann in Zeit $O(mnU)$ berechnet werden.

Hier ist $U=1$. $\Rightarrow O(mn)$

Matching M in G $\mapsto$ Fluss f_M in N_G mit $\text{val}(f_M) = |M|$.

Ganzz. Fluss f in N_G $\mapsto$ Matching M in G mit $|M| = \text{val}(f)$.

Maximum Matching in G „$\simeq$“ ganzz. Maxflow in N_G.

$$\max_{M \text{ Matching in } G} |M| = \max_{f \text{ Fluss in } N_G} \text{val}(f)$$

Kantendisjunkte Pfade

Kantendisjunkte Pfade Problem. Gegeben ein Graph G mit zwei ausgezeichneten Knoten u und v, $v \neq u$, bestimme eine möglichst grosse Menge kantendisjunkter u-v-Pfade.

Weshalb ist dies relevant? $\Rightarrow$ Kantenzusammenhang

Zur Erinnerung:

Satz (Menger)

Sei $G = (V, E)$ ein Graph. G ist genau dann k-kantenzusammenhängend, wenn es für alle Paare von Knoten $u, v \in V$, $u \neq v$, mindestens k kantendisjunkte u-v-Pfade gibt.

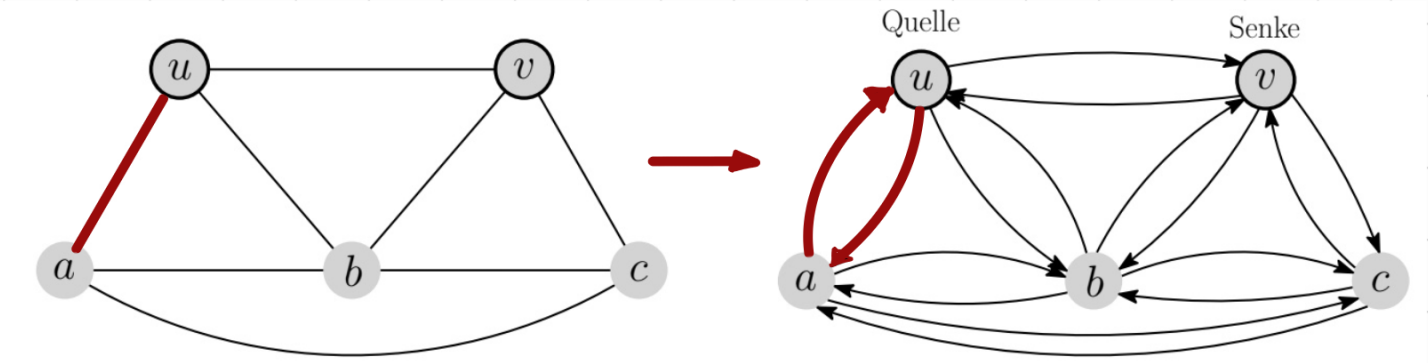
Sei $G=(V,E)$ mit $|V| \geq 2$ beliebig ungerichtet.

Sei $u, v \in V$, $u \neq v$ beliebig.

Dann definieren wir das Netzwerk $N_G^* = (V, A, c, u, v)$.

$$A := \{(x,y),(y,x) \mid \{x,y\} \in E\}$$

$$c(e) := 1 \;,\; \forall e \in A.$$



Anmerkung: Entgegengesetzte Kanten? Alle erarbeiteten Resultate gelten trotzdem. Es war nur eine vereinfachende Anahme.

Nun berechnen wir den maximalen Fluss mit Ford-Fulkerson (ganzzahlig)

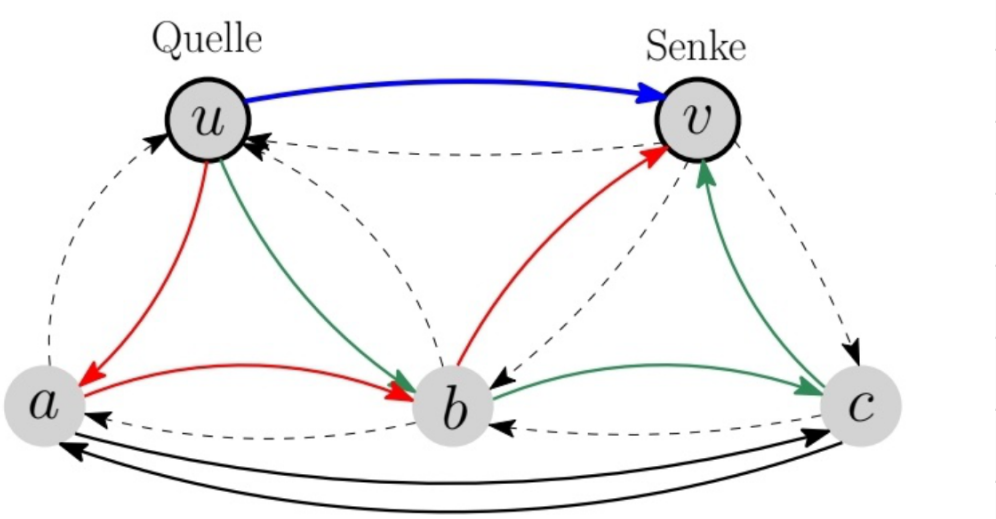
Wegen der Kapizitätsfunktion $c(e)=1, \forall e \in A$ folgt:

- Aus Flusserhaltung $\left(\sum_{x \in V:(w,x)} f(w,x) - \sum_{x \in V:(x,w)} f(x,w) = 0 \;,\; \forall w \in V \setminus \{u,v\}\right)$

$$indeg_f(w) = outdeg_f(w) \;,\; \forall w \in V \setminus \{u,v\}$$

- Aus $val(f) = netoutflow(u) = netinflow(v)$

$$val(f) = outdeg_f(u) - indeg_f(u) = indeg_f(v) - outdeg_f(v)$$



- Beginnend bei u laufe entlang gerichteten ungebrauchten Kanten mit Fluss 1 bis man bei v ankommt. Unterwegs durchlaufene Kanten werden als gebraucht markiert.
- Wiederhole val(f) Mal. Das gibt val(f) kantendisjunkte Pfade (nach Entfernen von Kreisen).

Liste, ...

↳ Laufe den Pfad ab und füge jeden Knoten x zu P hinzu.

Wenn wir auf einen Knoten y stossen, so dass y schon in P, entferne den Kreis von y zu y.

I.e. $\langle \ldots y, x, t, z, y \ldots \rangle \to \langle \ldots, y, \ldots \rangle$

usw. bis wir auf v stossen (Senke).

Mit dem Maxflow-Mincut Theorem und diesem Netzwerk, können wir auch den Satz von Menger beweisen.

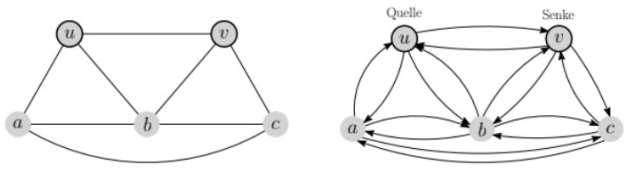
Satz (Maxflow-Mincut)

Jedes Netzwerk erfüllt

$$\max_{f \text{ Fluss}} \text{val}(f) = \min_{(S,T) \text{ s-t-Schnitt}} \text{cap}(S, T) \ .$$

Und ganzzahlige Netzwerke haben ganzzahlige maximale Flüsse.

⇓



Satz (Menger, Variante)

Sei G ein Graph mit Knoten u und v, $u \neq v$.

$$\max \# \text{ kantendisjunkter } u\text{-}v\text{-Pfade in } G$$
$$=$$
$$\min \# \text{ Kanten, die } u \text{ und } v \text{ trennen}$$

(„trennen“ heisst, nach Entfernen der Kanten sind u und v in verschiedenen Zusammenhangskomponenten des Graphen).

Bildsegmentierung

Gegeben ein Bild, trenne Hinter/-Vordergrund.

Modelliert:

Ein Bild ist ein Graph (P,E) mit Farbinformation $X: P \mapsto$ Farben.

P: Pixel; E: Kanten zw. benachbarten Pixeln

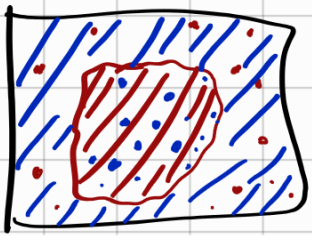
Jemand extrahiert aus den Farben der Pixel individuell eine Einschätzung, ob das Pixel im Vordergrund oder Hintergrund liegt:

$$\alpha : P \to \mathbb{R}_0^+ \qquad \alpha_p \text{ grösser} \Rightarrow \text{eher im Vordergrund}$$
$$\beta : P \to \mathbb{R}_0^+ \qquad \beta_p \text{ grösser} \Rightarrow \text{eher im Hintergrund}$$

Vordergrund $A = \{p \in P \mid \alpha_p > \beta_p\}$

Hintergrund $B = P \setminus A$

Zu feinkörnig:



Wir erhalten eine dritte Einschätzung, ob benachbarte Pixel eher im gleichen Teil (Vorder-/Hintergund) liegen.

$$\alpha : P \to \mathbb{R}_0^+ \qquad \alpha_p \text{ grösser} \Rightarrow \text{eher im Vordergrund}$$
$$\beta : P \to \mathbb{R}_0^+ \qquad \beta_p \text{ grösser} \Rightarrow \text{eher im Hintergrund}$$
$$\gamma : E \to \mathbb{R}_0^+ \qquad \gamma_e \text{ grösser} \Rightarrow \text{eher im gleichen Teil}$$

Qualitätsfunktion für Vorder-/Hintergrundspartition (A, B) von P:

$$q(A,B) := \sum_{p \in A} \alpha_p + \sum_{p \in B} \beta_p - \sum_{e \in E,\, |e \cap A| = 1} \gamma_e \,.$$

Neue Problemstellung:

Bildsegmentierung. Gegeben ein Bild (P, E) mit

$$\alpha : P \to \mathbb{R}_0^+, \quad \beta : P \to \mathbb{R}_0^+, \quad \gamma : E \to \mathbb{R}_0^+ ,$$

finde eine Partition (A, B) von P die

$$q(A, B) := \sum_{p \in A} \alpha_p + \sum_{p \in B} \beta_p - \sum_{e \in E,\, |e \cap A| = 1} \gamma_e \ .$$

maximiert.

Umformung

$$q(A, B) := \sum_{p \in A} \alpha_p + \sum_{p \in B} \beta_p - \sum_{e \in E,\, |e \cap A| = 1} \gamma_e \ .$$

Mit $Q := \sum_{p \in P} (\alpha_p + \beta_p)$ gilt

$$q(A, B) = Q - \sum_{p \in A} \beta_p - \sum_{p \in B} \alpha_p - \sum_{e \in E,\, |e \cap A| = 1} \gamma_e \ .$$

$q(A, B)$ zu maximieren is äquivalent zur Minimierung von

$$q'(A, B) := \sum_{p \in A} \beta_p + \sum_{p \in B} \alpha_p + \sum_{e \in E,\, |e \cap A| = 1} \gamma_e \ .$$

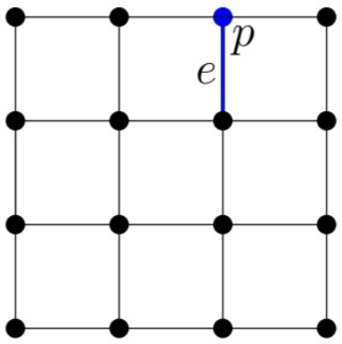
MinCut !

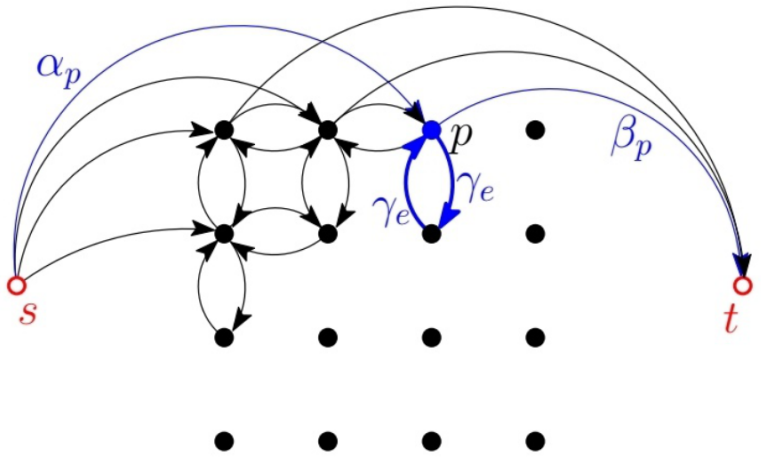
Wir definieren $N := (P \cup \{s,t\}, \vec{E}, c, s, t)$ wobei:

- $s, t \notin P$
- Alpha $= \{s\} \times P$, Beta $= P \times \{t\}$,
- Gamma $= \{(p,p'), (p',p) \mid \{p,p'\} \in E\}$

- $\vec{E} :=$ Alpha $\cup$ Gamma $\cup$ Beta

- $c(x,y) = \begin{cases} \alpha_y & \text{falls } (x,y) \in \text{Alpha} \\ \beta_x & \text{falls } (x,y) \in \text{Beta} \\ \gamma_{(x,y)} & \text{falls } (x,y) \in \text{Gamma} \end{cases}$





Sei (S, T) ein s-t-Schnitt, und $A := S \setminus \{s\}$ und $B := T \setminus \{t\}$.
Welche Kanten mit welchem Beitrag sind in diesem s-t-Schnitt?

- Kanten (s, p) mit $p \in B$; Beitrag zu $\text{cap}(S, T)$ ist $\sum_{p \in B} \alpha_p$.
- Kanten (p, t) mit $p \in A$; Beitrag zu $\text{cap}(S, T)$ ist $\sum_{p \in A} \beta_p$.
- Kanten (p, p') des Netzwerks in $A \times B$ mit Beitrag

$$\sum_{(p,p') \in A \times B, \{p,p'\} \in E} \gamma_{(p,p')}.$$

Es folgt

$$\begin{aligned} q'(A,B) &:= \sum_{p \in A} \beta_p + \sum_{p \in B} \alpha_p + \sum_{e \in E, |e \cap A| = 1} \gamma_e \\ &= \text{cap}(S,T) \end{aligned}$$

Minimaler s-t Schnitt, minimiert $q'(A,B)$.

$\Rightarrow$ Gibt uns die beste Partition.

III. Aufgabe

Aufgabe 1 – *Meisterschaft*

Die Mannschaften aus *Bedigliora*, *Caslano*, *Novaggio*, *Pura* und *Sessa* spielen um die Meisterschaft. Während der Saison muss jede Mannschaft sechs mal gegen jede andere Mannschaft antreten, und bei jedem Spiel wird genau ein Punkt an den Sieger der Begegnung vergeben (es gibt immer einen Sieger und einen Verlierer). Die aktuelle Tabelle ist rechts angegeben. Dabei hat Bedigliora noch 5 ausstehende Spiele gegen Caslano, 3 gegen Sessa und 6 gegen Novaggio. Caslano hat noch 2 ausstehende Spiele gegen Sessa und 5 gegen Novaggio. Sessa muss noch 3 mal gegen Novaggio antreten.

	Mannschaft	Punkte
C	Caslano	8
N	Novaggio	7
B	Bedigliora	6
S	Sessa	3
P	Pura	0

24

Pura verpflichtet mit sofortiger Wirkung einen Superstar, um doch noch die Meisterschaft zu gewinnen. Aber können sie das überhaupt noch? *Gibt es einen weiteren Saisonverlauf, bei dem Pura am Ende auf einem (womöglich geteilten) ersten Platz landet?*

Modellieren Sie das Problem als Flussproblem in einem Netzwerk.

Hinweis: Wie viele Spiele müsste Pura noch gewinnen, und wie viele dürfen die anderen Mannschaften noch gewinnen? Benutzen Sie diese Informationen, um die Netzwerkkapazitäten geeignet festzulegen.

B vs C	–	5	C vs S	–	2
B vs S	–	3	C vs N	–	5
B vs N	–	6	S vs N	–	3

24

Gespielt: 24

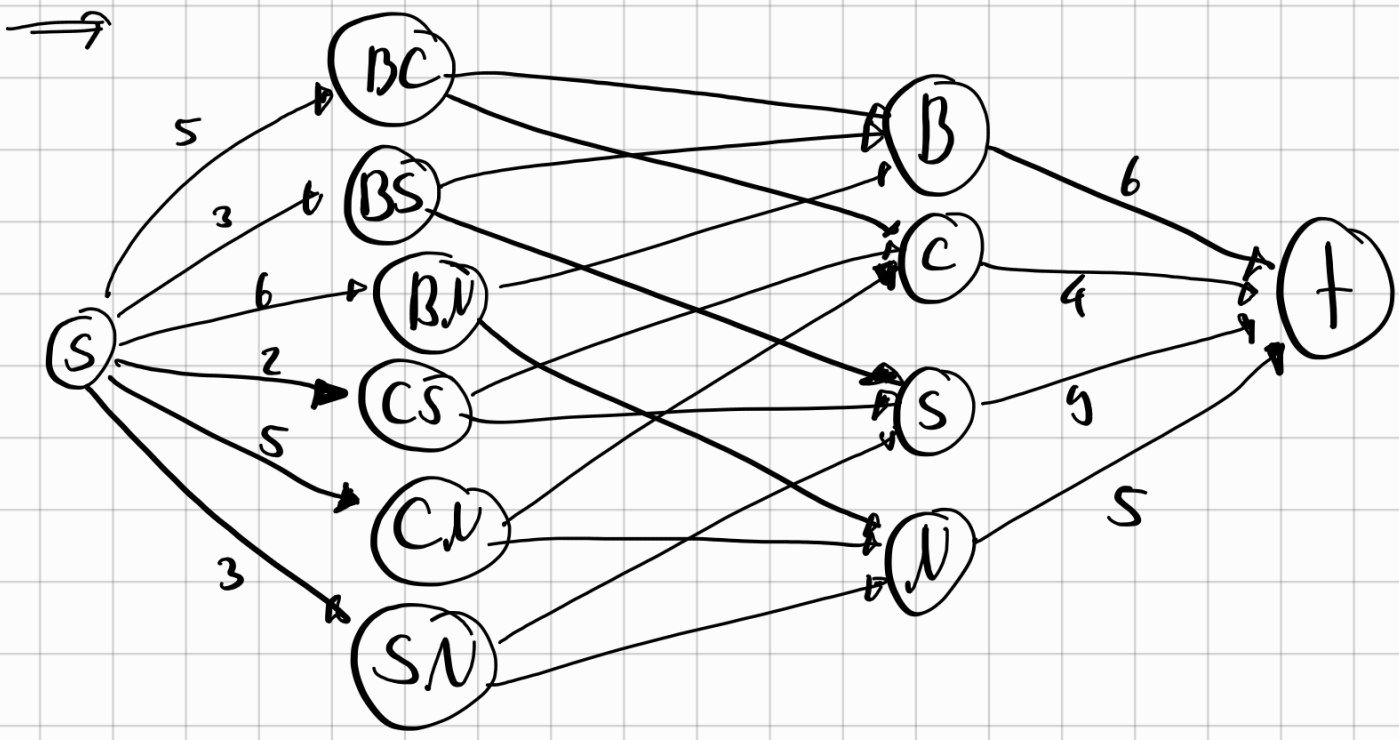
Geplant ohne Pura: 24

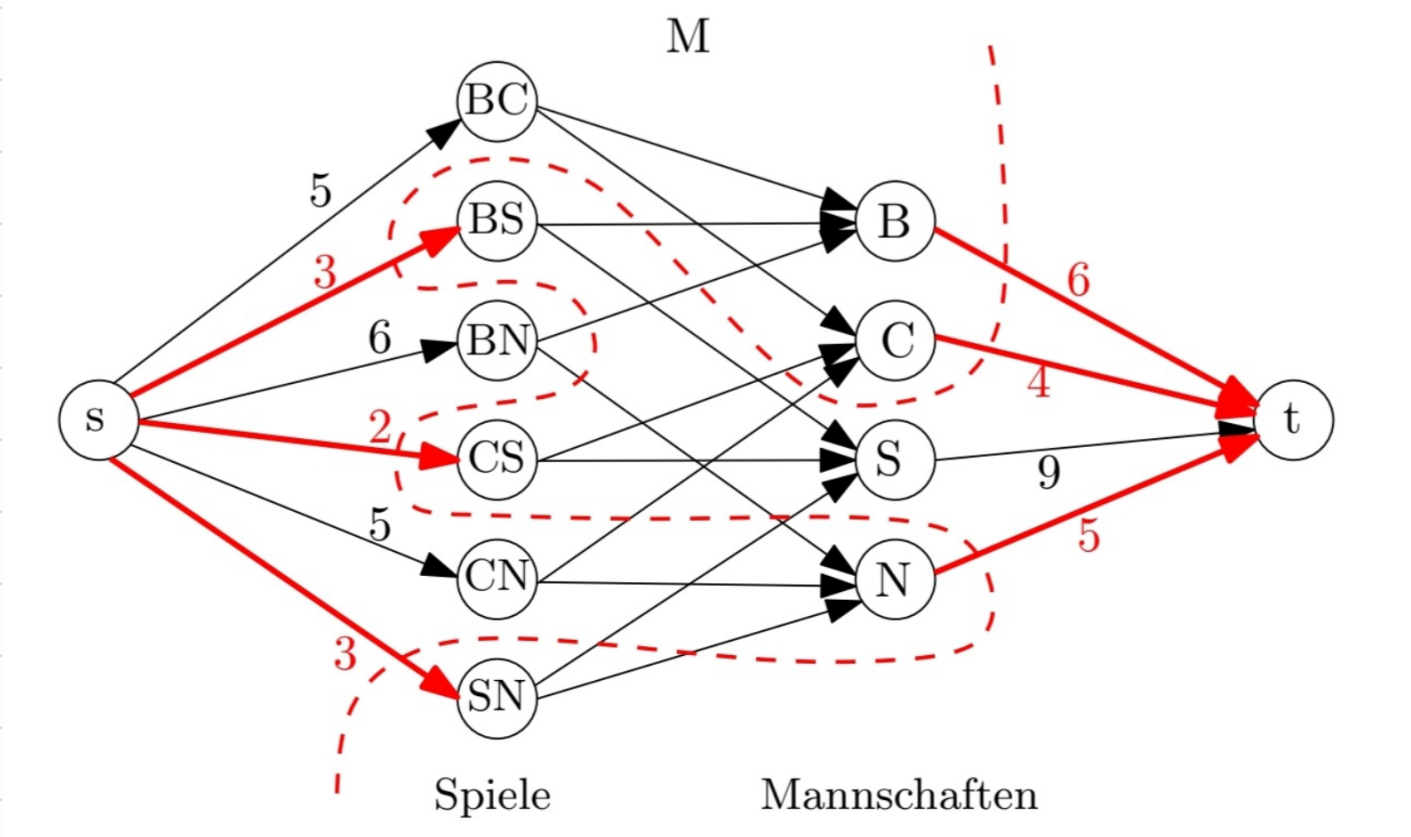
$\frac{5 \cdot 4 \cdot 6}{2} = 60$ Spiele total

$\Rightarrow$ Pura hat noch $60-24-24=12$ Spiele.

Pura holt im besten Fall 12 Punkte.

Damit darf C höchstens noch 4, N noch 5, B noch 6 und S noch 9 Punkte gewinnen.


BC
BS
BN
CS
CN
SN
B
C
S
N
s
t
5
3
6
2
5
3
6
4
9
5


M
BC
BS
BN
CS
CN
SN
B
C
S
N
s
t
5
3
6
2
5
3
6
4
9
5
Spiele
Mannschaften