

1 Matching

1.1 Matching-Lemma

G Graph, $w : E \rightarrow \mathbb{R}$, $v \in V$, M' maximales Matching für $G' = G - v$; dann kann mit einer Berechnung eines erhöhenden Weges Matching M maximalen Gewichts von G berechnet werden.

1.2 Matching-Algorithmus für planaren Graphen G

1. Zerlege G in G_1, G_2 dank Separator S entsprechend Planar-Separator-Theorem und berechne rekursiv in G_1 und G_2 Matching M_1, M_2 maximalen Gewichts; definiere $M := M_1 \cup M_2$, $G' = G_1 \cup G_2$
2. Solange $S \neq \emptyset$:
 - wähle $v \in S$, $S := S - v$, und berechne mit Lemma aus M' Matching maximalen Gewichts in $G' + v$

Laufzeit t Laufzeit von Matching, t' von Lemma, $c_1, c_2 \leq \frac{2}{3}, c_3 \in \mathbb{N}$, $c_1 + c_2 \leq 1$

$$t(n) = t(c_1 \cdot n) + t(c_2 \cdot n) + c_3 \sqrt{nt'}(n)$$

Mit Mastertheorem kann $t(n)$ abgeschätzt werden durch

$$t(n) \in \mathcal{O}\left(n^{\frac{3}{2}}\right), \text{ falls } t'(n) \in \mathcal{O}(n), \text{ falls ungewichtet}$$

$$t(n) \in \mathcal{O}\left(n^{\frac{3}{2}} \log n\right), \text{ falls } t'(n) \in \mathcal{O}(n \log n), \text{ falls gewichtet}$$

2 Mixed-Max-Cut in planaren Graphen

2.1 Definition: Schnitt

$G = (V, E)$ Graph, $S \subset E$ heißt **Schnitt** von G , falls der durch $E - S$ induzierter Subgraph unzusammenhängend ist und für alle $(u, v) \in S$, u und v in verschiedenen Zusammenhangskomponenten liegen.

2.2 Definition: Mixed-Max Cut

Kantengewichte $w : E \rightarrow \mathbb{R}$

Mixed-Max Cut: Finde Schnitt S mit $w(S) = \sum_{s \in S} w(s)$ maximal.
ist in bel. Graphen NP-Schwer.

Beobachtung: MIXED-MAX CUT und MIXED-MIN CUT sind äquivalent. (Vorzeichen der Gewichte umdrehen.)

Spezialfall: MIN CUT Problem mit $w : E \rightarrow \mathbb{R}_{\geq 0}$ ist auch für beliebige Graphen in P.

2.3 Definition:

Matching M in G mit $|V|$ gerade heißt perfekt, falls $2|M| = |V|$

2.4 Polynomialer Algorithmus für Mixed-Max Cut in planaren Graphen

Verwende

- Dualität von Schnitten und Kreisen
- maximales Matching bzw. Planar Separator Theorem

Laufzeit in $\mathcal{O}(n^{3/2} \log n)$

Es gilt: G enthält Euler-Kreis g.d.w. E kantendisjunkte Vereinigung einfacher Kreise g.d.w. $\forall v \in V$ ist $d(v) \in 2\mathbb{Z}$

Dualität von Schnitt in G und Menge von einfachen Kreisen in Dualgraph G^* (bzgl. bel. pl. Einbettung) Menge von einfachen Kreisen = Kantenmenge, in der für alle Knoten der Knotengrad gerade ist =: gerade Menge

(maximaler Schnitt in G induziert maximalen Kreis in G^* und umgekehrt)

1. Trianguliere G in $\mathcal{O}(n)$; zusätzliche Kanten erhalten Gewicht 0
2. berechne in $\mathcal{O}(n)$ Dualgraph G^* bzgl. bel. pl. Einbettung; G^* ist dann 3-regulär, d.h. $\forall v \in V^* : d(v) = 3$
3. Konstruiere zu G^* Graph G' , so dass perfektes Matching min. Gewichts in G' eine gerade Menge (bzw. Menge von Kreisen) max. Gewichts in G^* induziert
4. berechne in $\mathcal{O}(n^{3/2} \log n)$ solch ein Matching bzw. gerade Menge
5. falls diese gerade Menge nichtleer, berechne daraus den entsprechenden Schnitt, ansonsten Sonderfall

2.4.1 Schritt 3

beachte, dass G^* 3-regulär; ersetze jeden Punkt in G^* durch ein Dreieck, erhalte G' ; Matching ergibt zwei Fälle: (1)

Sei m die Anzahl der Kanten in G^* und n die Anzahl der Knoten

$$\Rightarrow 3n = 2m \Rightarrow n \text{ gerade}$$

ergo hat G' eine gerade Anzahl an Knoten. Wir sehen, dass mindestens ein perfektes Matching für G' existiert.

2.4.2 Schritt 4

Konstruiere perfektes Matching minimalen Gewichts in G'

Beobachtung M perfektes Matching minimalen Gewichts in $G = (V, E)$ mit $w : E \rightarrow \mathbb{R}$, g.d.w. M perfektes Matching max. Gewichts in G bzgl. $y(e) = W - w(e)$, für W geeignet erzwingt, dass Matching max. Gewichts perfekt ist:

- zu M perfekt, betrachte $y(M) = \sum_{e \in M} y(e) = nW/2 - \sum_{e \in M} w(e) \geq n/2(W - w_{\max})$
- Zu N nicht perfekt, gilt $v(N) \leq (n/2 - 1)(W - w_{\min})$
- Wähle W so, dass

$$v(N) \leq (n/2 - 1)(W - w_{\min}) < n/2(W - w_{\max}) \leq y(M)$$

in $\mathcal{O}\left(n^{\frac{3}{2}} \log n\right)$

2.4.3 Schritt 5

Komplementmenge von perfektem Matching min. Gewichts in G' induziert gerade Menge max. Gewichts in G^* und damit max. Schnitt in G .

Es kann sein, dass resultierende Menge leer ist. Passiert, wenn max. Schnitt negatives Gewicht hat.

\rightsquigarrow **Sonderfall:** Wollen nichttrivialen Schnitt erzwingen;

betrachte wieder Schritt 3, erzwingt, dass in perfektem Matching minimalen Gewichts für mindestens einen Knoten $v \in G^*$, Fall 2 eintritt.

Vorgehensweise betrachte alle Knoten $v \in G^*$ und $G^* - v$ sowie durch perfektes Matching in G' induziertes Matching in $G^* - v$ und berechne mit Matching-Lemma ein Matching in G^* .

Wähle M mit $w(M) = \min_{v \in V^*} w(M_v)$

Frage Wie kann man dabei Fall 2 bei v erzwingen? (2)

Folien: Maximale s-t-Flüsse in Planaren Graphen

2.5 Lemma

Für jeden Kozykel C gilt: $\pi(C) \in \{-1, 0, 1\}$.

Ferner: $\pi(C) = 1 \iff C$ ist (s, t) -Schnitt

Beweis

Fall 1: s, t liegen auf derselben Seite von C^*

$$\iff P \text{ kreuzt } C^* \text{ gleich oft in jeder Richtung}$$

$$\iff C \text{ enthält dieselbe Zahl von Kanten in } P \text{ und } \bar{P}$$

$$\iff \pi(C) = 0$$

Fall 2: s liegt innen und t außen von C^*

$$\iff P \text{ kreuzt } C^* \text{ einmal mehr in die Richtung von } s \rightarrow t$$

$$\iff C \text{ enthält eine Kante mehr in } P \text{ als in } \bar{P}$$

$$\iff \pi(C) = 1$$

Fall 2: s liegt innen und t außen von C^*

analog wie Fall 2

2.6 Lemma

G besitzt einen gültigen $s-t$ -Fluss mit Wert λ genau dann, wenn G_λ^* keinen negativen Kreis enthält.

Beweis Zeige ' \implies ': Annahme: G_λ^* enthält negativen Kreis C^* , d.h. $0 > c(\lambda, C^*) = \sum_{e \in C^*} c(\lambda, e) = \sum_{e \in C^*} c(e) - \lambda \sum_{e \in C^*} \pi(e) = c(C^*) - \lambda \pi(C^*) \implies \pi(C^*) > c(C^*)/\lambda \leq 0 \implies \pi(C^*) = 1$
 $\implies C^*$ ist $s-t$ -Schnitt

Außerdem $c(C^*) < \lambda$, d.h. es existiert ein Schnitt mit Kapazität $< \lambda$, das ist ein Widerspruch

Zeige ' \impliedby ': G_λ^* enthält keinen negativen Kreis.

\implies kürzeste Wege wohldefiniert; sei x in G_λ^* beliebiger Ursprung, $dist(p, \lambda) :=$ Distanz von x zu p

Definition

$$\phi(\lambda, e) := dist(\lambda, head(e^*)) - dist(\lambda, tail(e^*)) + \lambda \pi(e)$$

Zeige ϕ ist gültiger st -Fluss

1. Für $v \in V$ gilt: $\sum_w \phi(v \rightarrow w) = \sum_w \lambda \pi(v \rightarrow w)$ es folgt: $\phi(\lambda, \cdot)$ ist Fluss mit Wert λ
2. $slack(\lambda, e^*) := dist(\lambda, tail(e^*)) + c(\lambda, e) - dist(\lambda, head(e^*))$ es gilt: $slack(\lambda, e) = c(e) - \phi(\lambda, e)$
 $\phi(\lambda, e) \leq c(e) \iff slack(\lambda, e) \geq 0$ Wäre $slack(\lambda, e) < 0$, dann folgt: $dist(\lambda, head(e^*)) > dist(\lambda, tail(e^*)) + c(\lambda, e^*)$, das wäre ein Widerspruch

2.7 Satz

Ein maximaler st -Fluss in einem st -planaren Graph kann in $\mathcal{O}(n \log n)$ Zeit berechnet werden.

Max λ , s.d. kein neg. Kreis in G_λ^* ist Länge des kürzesten ts -Weges in G_λ^*

3 Das Menger-Problem

3.1 Zur Erinnerung

$S \subset V$ heißt Separator in G , falls $G - S$ unzusammenhängend.

$S \subset E$ heißt Schnitt in G , falls $G - S$ unzusammenhängend.

3.2 Definitionen

Zu $u, v \in V$ definiere den **Knotenzusammenhang**

$$\kappa_G(u, v) := \begin{cases} |V| - 1, & \text{falls } \{u, v\} \in E \\ \min_{S \subset V} |S|, & \text{für } S \text{ Separator, der } u \text{ und } v \text{ trennt} \end{cases}$$

und $\kappa_G := \min_{u, v \in V} \kappa_G(u, v)$

$$\lambda_G(u, v) := \min_{S \subset E, S \text{ Schnitt und trennt } u \text{ und } v} |S|$$

und

$$\lambda(G) := \min_{u, v \in V} \lambda_G(u, v)$$

Zwei Wege heißen **kantendisjunkt**, wenn sie keine gemeinsame Kante enthalten, und (intern) **knotendisjunkt**, wenn sie außer Anfangs- und Endknoten keinen gemeinsamen Knoten enthalten.

3.3 Satz von Menger

Seien s und t Knoten in $G = (V, E)$ ($\{s, t\} \notin E$ bei knotendisjunkter Version)

- $\kappa_G(s, t) \geq k \iff \exists_{\geq k}$ paarweise knotendisjunkte st -Wege
- $\lambda_G(s, t) \geq k \iff \exists_{\geq k}$ paarweise kantendisjunkte st -Wege

3.4 Menger-Problem

Finde zu G, s, t maximale Anzahl knotendisjunkter bzw. kantendisjunkter st -Wege.

3.5 Menger-Problem in planaren Graphen: kantendisjunkte Variante

Linearzeitalgorithmus basierend auf RIGHT-FIRST-DFS.

Spezialfall s und t liegen auf derselben Facette:

RIGHT-FIRST = im Gegenuhrzeigersinn nächste freie Kante in Adjazenzliste (relativ zur aktuellen eingehenden Kante).

Algorithmus G planar eingebetteter Graph, OE t auf äußerer Facette

1. Ersetze G durch den gerichteten Graphen \vec{G} , indem $\{u, v\} \in E$ durch (u, v) und (v, u) ersetzt wird. (in $\mathcal{O}(n)$)
2. Berechne in $\mathcal{O}(n)$ Menge gerichteter einfacher kantendisjunkter Kreise $\vec{C}_1, \dots, \vec{C}_l$ und konstruiere aus \vec{G} den Graphen \vec{G}_C , indem die Richtung aller Kanten auf den \vec{C}_i umgedreht wird.
3. Berechne in \vec{G}_C in $\mathcal{O}(n)$ mittels RIGHT-FIRST-DFS eine maximale Anzahl kantendisjunkter gerichteter st -Wege.
4. Berechne aus den in Schritt 3 gefundenen st -Wegen in \vec{G}_C gleiche Anzahl kantendisjunkter st -Wege in G in $\mathcal{O}(n)$.

Schritt 1

3.6 Lemma

Seien P_1, \dots, P_r kantendisjunkte, gerichtete st -Wege in \vec{G} . Dann enthält

$$P = \{\{u, v\} \in E \mid \text{Genau eine der Kanten } (u, v) \text{ und } (v, u) \text{ liegt auf einem der } P_i\}$$

gerade r kantendisjunkte st -Wege in G .

Beweis Zwei Fälle: Wir konstruieren in beiden Fällen aus gegebenen st -Wegen unproblematische st -Wege

1. $(u, v) \in P_i \wedge (v, u) \in P_i$: Entferne (u, v, \dots, v, u) bzw. (v, u, \dots, u, v) aus P_i
2. $(u, v) \in P_i \wedge (v, u) \in P_j$: $P_i = (A, u, v, B)$, $P_j = (C, v, u, D)$; konstruiere $\tilde{P}_i = (A, D)$ und $\tilde{P}_j = (C, B)$

Schritt 2 C_1, \dots, C_l in \vec{G} , sodass

1. \vec{G}_C enthält keine Rechtskreise, d.h. keine Kreise, deren Inneres rechts liegt (aus Sicht einer Kante).
2. Sei $\vec{P}_C \subset \vec{E}_C$ Menge der Kanten auf kantendisj. $s - t$ Wegen in \vec{G}_C und $\vec{P} \subset \vec{E}$, wobei

$$\vec{P} := (\vec{P}_C \cap \vec{E}) \cup \{(u, v) \in \vec{E} : (u, v) \text{ auf einem der } \vec{C}_i \text{ und } (v, u)' \notin \vec{P}_C\}$$

Dann induziert \vec{P} k kantendisjunkte gerichtete st -Wege in \vec{G} g.d.w \vec{P}_C induziert k kantendisjunkte gerichtete st -Wege in \vec{G}_C .

Konstruktion der Kreise C_1, \dots, C_l Sei f Facette in G bzw. \vec{G} ; definiere Abstand von f zur äußerer Facette f_0 als

$dist(f) :=$ Länge eines kürzesten Weges des Dualknotens f^* zum Dualknoten der äußeren Facette f_0^* in G^*

Definiere C_i als Vereinigung der einfachen Kreise in G für die alle Facetten f im Inneren die Bedingung $dist(f) \geq i$ erfüllen. \vec{C}_i sei entsprechender Rechtskreis in \vec{G} . Drehe alle diese C_i um, erhalte \vec{G}_C .

\vec{G}_C enthält keine Rechtskreise, da für jeden Rechtskreis in \vec{G} beim Übergang zu \vec{G}_C mindestens eine Kante des Kreises umgedreht wird.

Sei $\vec{P}_C \subset \vec{E}_C$ Kantenmenge zu k st -Wegen in \vec{G}_C . Konstruiere dazu Kantenmenge \vec{P} in \vec{G} .

$$\vec{P} := (\vec{P}_C \cap \vec{E}) \cup \{(u, v) \in \vec{E} : (u, v) \text{ auf einem } \vec{C}_j \text{ und } (v, u)' \notin \vec{P}_C\}$$

Schritt 3 Berechnung einer maximalen Anzahl kantendisjunkter st -Wege in \vec{G}_C (in $\mathcal{O}(n)$)

Schleife über ausgehende Kanten aus s

RIGHTFIRSTDEPTHSEARCH:

Suchschritt: rechteste nicht markierte auslaufende Kante in Bezug auf Referenzkante

Zwei Variationen, wie die **Referenzkante** zu wählen ist

1. Weihe: aktuell einlaufende Kante
2. Coupry: erste einlaufende Kante

Korrektheitsbeweis zu Schritt 3 Beh.: \vec{P}_C enthält maximale Anzahl kantendisjunkter st -Wegen.

Benutze dazu gewichtete Variante des Satz v. Menger, d.h. konstruiere st -Schnitt der entsprechenden Kapazität.

Schnitt A wird induziert durch geeigneten Kreis $\vec{K} \subset \widehat{\vec{G}_C}$ mit:

1. $s \in \text{Innen}(\vec{K})$ oder auf \vec{K}
2. $t \in \text{Aussen}(\vec{K})$
3. $A := \{(u, v) \in \vec{E}_C \mid u \text{ liegt auf } \vec{K}, v \in \text{Aussen}(\vec{K})\}$, $|A| = \# st\text{-Wegen in } \vec{P}_C$

\vec{K} wird mittels LEFTFIRST-Rückwärtssuche von s aus in \vec{P}_C konstruiert.

Wie sieht \vec{K} aus:

Variante 1 \vec{K} geht von s nach s

Variante 2 \vec{K} geht von $s \neq v_0$ nach v_0 und s
In diesem Fall den Kreis, der von v_0 nach v_0 beschrieben wird.

3.7 Lemma

Betrachte $\vec{G}_C = (V, \vec{E}_C)$ und \vec{K} , dann ist jede Kante $(u, v) \in \vec{E}_C$ mit u auf \vec{K} und $v \in \text{Aussen}(\vec{K})$ durch einen st -Weg aus \vec{P}_C besetzt.

Beweis

1. Wenn P_1, \dots, P_l st -Wege sind und \vec{K} ein Linkskreis von s nach s , dann gehört keine der Kanten (x, y) , $x \in \text{Aussen}(\vec{K})$, $y \in \vec{K}$ zu einem der p_i .
Wegen LEFTFIRST in Graph indiziert durch p_1, \dots, p_l $(x, y) \in p_i$, für alle $1 \leq i \leq l$.
Deswegen: Kante (u, v) mit u auf \vec{K} , $v \in \text{Aussen}(\vec{K})$ kann nicht auf einem Linkskreis aus p_1, \dots, P_l liegen.
2. betrachte (u, v) mit u auf \vec{K} , $v \in \text{Aussen}(\vec{K})$ und (u, w) mit w auf \vec{K} .
Annahme: (u, v) gehört zu keinem der P_1, \dots, P_l .
Betrachte Referenzkante zu (u, w) in RIGHTFIRST-Suche (Schritt 3)
Referenzkante geht von Innerem zum Kreis oder liegt auf den Kreis,
aber dann hätte RIGHTFIRST nicht (u, w) sondern (u, v) gewählt. Das wäre ein Widerspruch.

4 Das Okamura & Seymour Problem

Sei G ein planarer Graph, $D = \{\{s_i, t_i\}, s_i, t_i \in V, 1 \leq i \leq k\}$, s_i, t_i liegen alle auf Rand der äußeren Facette.

Finde k paarweise kantendisjunkte Wege, die jeweils s_i mit t_i verbinden.

4.1 Kapazitätsbedingung

Für jeden Schnitt $X \subset V$ ist die **freie Kapazität** $fcap(X) \geq 0$,

$$fcap(X) := cap(X) - dens(X)$$

$$cap(X) = |\{\{x, y\} \in E \mid x \in X, y \in V \setminus X\}|$$

$$dens(X) = |\{\{s, t\} \in D \mid \#(\{s, t\} \cap X) = 1\}|$$

(**Kapazität** und **Dichte** von X)

Die Kapazitätsbedingung ist offensichtlich notwendig für Lösbarkeit des Problems. Allerdings ist sie i.A. nicht hinreichend.

Anti-Beispiel Ein Quadrat, mit den Ecken a, b, c, d mit Farben 1,2,1,2. Erfüllt Kapazitätsbedingung, löst aber Problem nicht. $fcap(X) = 1$, wobei $X = \{a \mid \}$.

4.2 Geradheitsbedingung

Für alle Schnitte $X \subset V$ gilt $fcap(X)$ ist gerade.

4.3 Satz von Okamura & Seymour

Falls die Geradheitsbedingung erfüllt ist, ist die Kapazitätsbedingung äquivalent zur Lösbarkeit des Problems.

4.4 Lemma

Es gilt:

$$fcap(X) \text{ gerade } \forall X \subset V \iff fcap(v) \text{ gerade } \forall v \in V$$

wobei

$$\begin{aligned} fcap(v) &= d(v) - dens(v) \\ dens(v) &= \# \{i \mid s_i = v\} + \# \{i \mid t_i = v\} \end{aligned}$$

Beweis \implies trivial.

\impliedby : Sei $fcap(v)$ gerade für alle v , $X \subset V$:

$$\begin{aligned} cap(X) &= \sum_{v \in X} cap(v) - 2 |\{\{u, v\} \in E \mid u, v \in X\}| \\ dens(X) &= \sum_{v \in X} dens(v) - 2 |\{\{s, t\} \in D \mid s, t \in X\}| \\ fcap(X) &= \sum_{v \in X} cap(v) - \sum_{v \in X} dens(v) - 2 |\{\{u, v\} \in E \mid u, v \in X\}| + 2 |\{\{s, t\} \in D \mid s, t \in X\}| = \sum_{v \in V} fcap(v) - 2N \in 2\mathbb{Z} \end{aligned}$$

$N \in \mathbb{N}$ \square

4.5 Linearzeitalgorithmus für planaren Graphen G

Terminalpaare D auf äußerer Facette und Geradheitsbedingung erfüllt.

Phase 1 Konstruiere aus G, D einfachere Instanz mit Klammerstruktur und berechne mittels RIGHT-FIRST-Tiefensuche kantendisjunkte Lösungswege q_1, \dots, q_k . Diese induzieren gerichteten Hilfsgraph.

Phase 2 Berechne mittels gerichteter RIGHTFIRST-Tiefensuche in Hilfsgraph Lösungswege p_1, \dots, p_k , die jeweils s_i mit t_i verbinden.

4.6 Phase 1: Instanz mit Klammerstruktur

$$G, D = \{\{s_i, t_i\} \mid \}$$

1. Wähle beliebiges Terminal als Startterminal.
2. Gehe im Gegenuhrzeigersinn: Dem ersten Terminal eines Paares, das einem begegnet, ordnet man ein aufgehende Klammer zu. Begegnet man dem zweiten, erhält dieses eine zugehende Klammer.
Entsprechende Klammerpaare ergeben neue Terminalpaare in D' (innere Klammerpaare haben kleineren Index).

3. Konstruiere mittels RIGHTFIRST-Suche Lösung q_1, \dots, q_k zu G, D' ; Reihenfolge, in der Wege berechnet werden nach Reihenfolge der t'_i , d.h. von innen nach außen in Klammerstruktur.

4.7 Korrektheit von Phase 1

Beobachtung

1. keine zwei Wege q_i, q_j kreuzen sich, wg. RIGHTFIRST-Auswahlregel
2. kein q_i kreuzt sich selbst
3. Sei G' der gerichtete Graph, der durch die q_i induziert wird. G' enthält keinen Rechtskreis. Angenommen G' hätte einen Rechtskreis, dann wären an diesem mind. zwei q_i, q_j beteiligt. Daraus folgt auf der Facette folgende Terminale: s_i, t_j, s_j, t_i , was der Klammerung $()()$ entspricht, was ein Widerspruch zur Paarung ist.
4. 1, 2 & 3 \implies induktiv über q_i kann gezeigt werden, dass q_i die richtigen Terminale verbindet.
5. 1.Phase in $\mathcal{O}(n)$ klar.

4.8 Phase 2

Ohne Einschränkung sei von Startterminal im Gegenuhrzeigersinn jeweils s_i vor t_i und Indizierung entsprechend Auftreten der t_i .

Für $i = 1, \dots, k$

1. $p_i :=$ führe RFS in G' von s_i aus bis zu einem j
2. Falls $i \neq j$, STOP.

gebe p_1, \dots, p_k aus.

Laufzeit $\mathcal{O}(n)$ amortisiert mit UNION-FIND wie beim kantendisjunkten Menger-Problem.

Korrektheit Der Algorithmus endet entweder

1. mit Wegen p_1, \dots, p_k , die jeweils s_i mit t_i verbinden.
2. p_1, \dots, p_{i-1} korrekt und p_i verbindet s_i mit t_j
 $\implies i < j$, da $i \neq j$
 Prozedur, die Weg p berechnet, so dass p einen Schnitt X induziert, der im Fall 1 **saturiert** ist, d.h. $fcap(X) =$ und der im Fall 2 **übersaturiert** ist, d.h. $fcap(X) < 0$.

Prozedur für Schnitt X :

Rückwärts-LFS startet von Terminal t_i bzw. t_j wo Weg p_i endet in Graph, der durch p_1, \dots, p_i induziert wird.

4.9 Lemma

Sei A Menge der Kanten $\{u, v\}$ aus G mit u auf p und v rechts von p . Jede Kante $\{u, v\} \in A$ gehört zu G' und genau dann in Orientierung (u, v) , wenn sie durch eine der p_1, \dots, p_i besetzt ist.

Beweis Wenn $\{u, v\}$ durch ein p_i besetzt, dann wegen Konstruktion von p in Orientierung (u, v) .

Fall 1 Es existiere (u, v) mit (u, v) nicht durch p_1, \dots, p_i besetzt. Mein Bild zeigt einen Widerspruch zu RFS Ur argument is invalid.

Fall 2 Es existiert $\{u, v\} \in A, (u, v), (v, u) \notin G'$. Bild. Ich bin müde as fuck... \square

4.10 Lemma

Sei X Schnitt induziert durch p (Knoten rechts von p). Falls p_i $s_i - t_i$ -Weg, so ist $fcap(X) = 0$, sonst $fcap(X) < 0$.

Beweis Kanten $\{u, v\}$ auf p, v rechts von p entweder zu Weg p_j gehört mit $1 \leq j < i, s_j \in V \setminus X$ und $t_j \in X$ oder zu Weg q_j aus erster Phase mit $s'_j \in X$ und $t'_j \in V \setminus X$.

Wenn p_i korrekter $s_i - t_i$ -Weg, so gilt

$$\begin{aligned} cap(X) &= \#\{\{s_j, t_j\} \mid s_j \notin X, t_j \in X, 1 \leq j \leq l\} + \#\{\{s'_j, t'_j\} \mid s'_j \in X, t'_j \notin X, \{s'_i, t'_j\} \notin D\} \\ &= dens(X) \end{aligned}$$

Wenn p_i nicht korrekt, d.h. s_i mit $t_j, i < j$, verbindet, so ist $cap(X) < dens(X)$, da $s_i \notin X, t_i \in X$.

\square