V.) Komplementäre Slackness

Wir betrachten wieder die Ungleichungen die für jedes (P)–(D) Paar gelten sollen für jede Lösung x von (P) und jede Lösung y von (D):

$$(c_j - y^T \cdot a^j) \cdot x_j \geq 0 \qquad \text{für } j = 1, 2, \ldots, n$$

und

$$y_i \cdot (a_i^T \cdot x - b_i) \geq 0 \qquad \text{für } i = 1, 2, \ldots, m$$

Der Beweis der schwachen Dualität war im Grunde genommen

$$c^T \cdot x - y^T \cdot b = \sum_{j=1}^{n} (c_j - y^T \cdot a^j) \cdot x_j + \sum_{i=1}^{m} y_i \cdot (a_i^T \cdot x - b_i) \geq 0$$

Hier gilt also ≥ 0 für jeden der $n+m$ Summanden einzeln! Dies impliziert, dass x und y genau dann beide optimal sind, d.h. $c^T \cdot x = y^T \cdot b$ genau dann gilt, wenn

$$(c_j - y^T \cdot a^j) \cdot x_j = 0 \qquad \text{für alle } j$$

und $$y_i \cdot (a_i^T \cdot x - b_i) = 0 \qquad \text{für alle } i$$

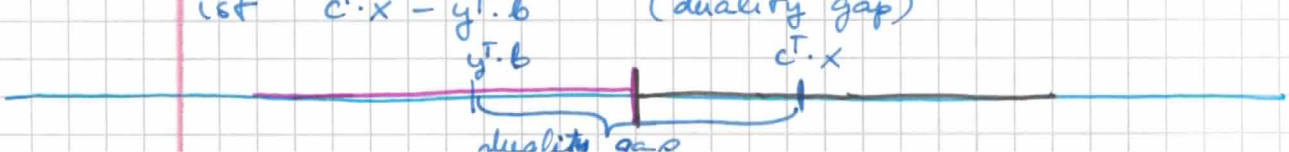
$\Updownarrow$

PKS: Für alle j: entweder $x_j = 0$ oder die entsprechende duale Ungleichung $c_j \geq y^T \cdot a^j$ exakt erfüllt wird $c_j = y^T \cdot a^j$

DKS: und analog für alle i: entweder $y_i = 0$ oder $a_i^T \cdot x = b_i$

Diese Bedingungen nennt man **Komplementäre Slackness Bedingungen** (complementary slackness conditions)

Für beliebige Lösungen x und y, die **Dualitätslücke** ist $c^T \cdot x - y^T \cdot b$ (duality gap)



vi.) Primal-duale Algorithmen (siehe Vazirani)

Diese Beobachtungen werden in den sogenannten primal-dualen Algorithmen benutzt. In diesen Heuristiken werden iterativ y und x Vektoren definiert, so dass die Dualitätslücke letztendlich möglichst klein wird, und die letzten x und y Lösungen von (P) bzw. von (D) sind. Wenn am Ende solche x und y gefunden werden, dass x ganzzahlig (also "echte" Lösung für das ursprüngliche kombinatorische Problem) ist, und

$$c^T \cdot x \leq \alpha \cdot y^T \cdot b \quad \text{für irgendein } \alpha,$$

dann ist x eine α-approximative Lösung des IP, weil $c^T \cdot x \leq \alpha \cdot y^T \cdot b \leq \alpha \cdot OPT_{(D)} \leq \alpha \cdot OPT_{IP}$

→ diese Algorithmen sind relativ schnell und einfach

→ oft versucht der Algorithmus eine ganzzahlige x Lösung Schritt für Schritt so zu definieren, dass $x_j \neq 0$ nur dann gilt, wenn für die aktuelle y $(c_j - y^T \cdot a^j) = 0$ gilt. So werden zumindest die primalen komplementäre Slackness Bedingungen $(c_j - y^T \cdot a^j) \cdot x_j = 0$ die ganze Zeit gelten (die dualen nicht, x wird kein Optimum des LP werden können) und man kann hoffen, dass $y^T \cdot b$ und $c^T \cdot x$ nah aneinander bleiben

→ das LP wird dabei typischerweise gar nicht gelöst

→ die y_i Variablen haben meistens auch eine kombinatorische Bedeutung:
P-D Algorithmen nutzen somit die kombinatorische Struktur des gegebenen Problems aus, und führen oft zu kombinatorischen Algorithmen ohne LP.

→ es gibt P-D Algorithmen für SET COVER, VERTEX COVER, NETZWERKFLUSS, MATCHING, SHORTEST PATH, STEINER BAUM, FACILITY LOCATION, K-MEDIAN, usw. siehe bei Shmoys u. bei Vazirani.

Primal-dualer Algorithmus: ein mögliches Schema

(P) $\min \; c^T \cdot x \qquad A \cdot x \geq b \qquad x \geq 0$

(D) $\max \; y^T \cdot b \qquad y^T \cdot A \leq c^T \qquad y \geq 0$

(wir nehmen o.B.d.A $A \geq 0 \quad c \geq 0 \quad b \geq 0$ an)

Grobstruktur:

- starte mit $x = 0 \quad y = 0$ (x keine Lösung von (P) aber y Lösung von (D))
- die primalen kompl. Slackness Bedingungen $(c_j - y^T \cdot a^j) \cdot x_j = 0$ gelten. Diese Invariante wollen wir behalten.
  (wir finden eine Lösung x Schritt für Schritt und ändern y parallel, s.d. $(c_j - y^T a^j) x_j$ gilt, wir müssen x_j Komponenten erhöhen damit x Lösung wird wir dürfen x_j erhöhen wenn $c_j = y^T \cdot a^j$ gilt:)
  - erhöhe Komponente(n) von y bis eine duale Ungleichung $c_j \geq y^T \cdot a^j$ exakt erfüllt wird; sei dies $c_k = y^T \cdot a^k$
  - erhöhe x_k um (weitere) Bedingungen $a_i^T \cdot x \geq b$ zu erfüllen

Beispiel 1: Primal-dualer Algorithmus für SET COVER

Die Elemente $\{1, 2, \ldots, m\}$ sind zu überdecken,

das Gewicht der Teilmenge S_j ist w_j.

Gesucht wird der charakteristische Vektor x der Überdeckung

$x_j = 1 \iff S_j$ ist in der Mengenüberdeckung $(j \in C)$

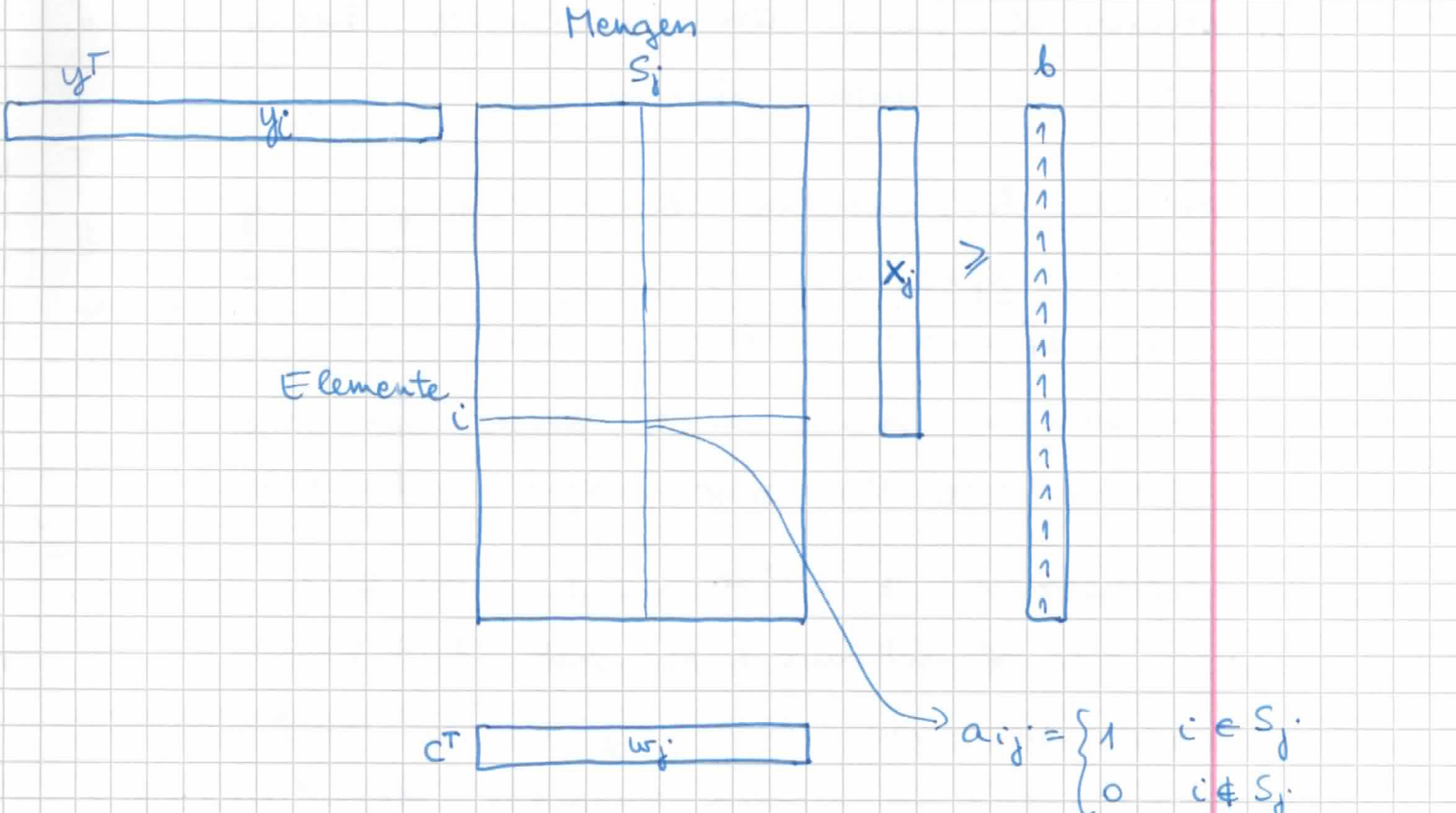
(P) minimiere $\sum_j w_j x_j$

„covering problem"

so dass $\sum_{j : i \in S_j} x_j \geq 1$ für jede $i = 1, 2, \ldots, m$

$x_j \geq 0$ $(x_j \leq 1$ wird in einer optimalen Lösung sowieso erfüllt)

Was sind A, b und c?



LPD 20. Was ist das duale Problem?

- für jedes Element i die Bedingung in (P) „überdecke i" entspricht einer dualen Variable $y_i \geq 0$
- maximiere $\sum_{i=1}^{m} y_i \cdot 1$
- jede Teilmenge S_j (x_j) entspricht der dualen Bedingung

„packing problem" (D) $\sum_{i \in S_j} y_i \leq w_j \qquad j = 1, 2, \ldots, n$

↓

die w_j werden „vollgepackt" mit y_i

Die Interpretation von y_i: „mindestens so viel kostet noch im Zielwert $\sum x_j \cdot w_j$ das Überdecken von Element i"

Unterschiedliche $y = (y_1\ y_2 \ldots y_m)$ entsprechen unterschiedlicher Aufteilung der Kosten über die Elemente (keine genaue Aufteilung, nur untere Schranke)

Die primale komplementäre Slackness Bedingung sagt:

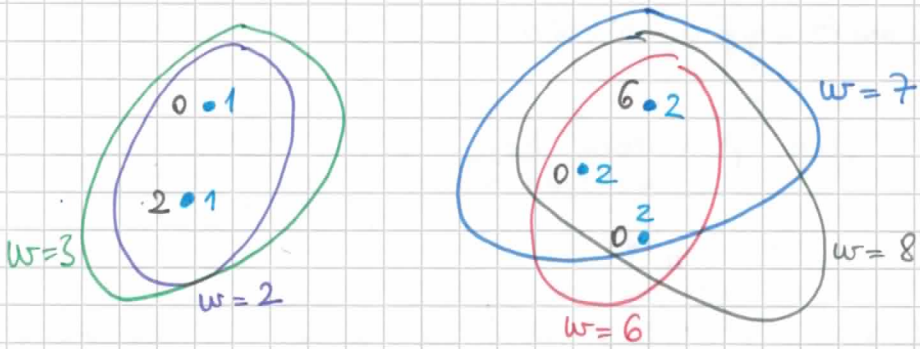
$\forall j \quad x_j = 0$ oder $\sum_{i \in S_j} y_i = w_j \rightarrow$ S_j gewählt und w_j aufgefüllt mit Preisen

S_j nicht gewählt

in unserem primal-dualen Algorithmus wird dies stets erfüllt. Wir starten mit $\underline{x} = \underline{0}$ $\underline{y} = \underline{0}$

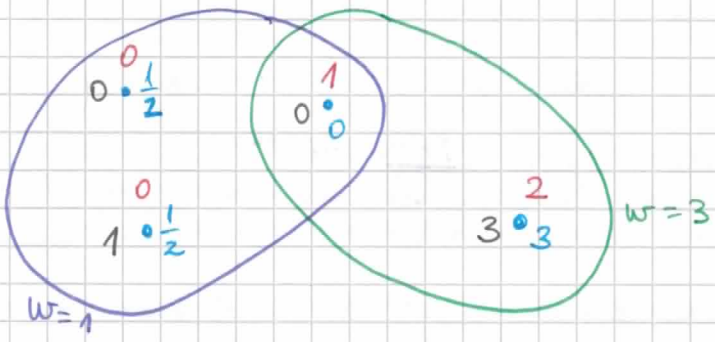
Wir illustrieren zunächst durch zwei Beispiele, wie der P-D Algorithmus funktioniert. ACHTUNG! Generell wird eine Lösung y von (D) nicht iterativ gewählt, nur in diesem einfachen Algorithmus: für ein nicht-überdecktes Element i, wird y_i hochgesetzt bis ein w_j gefüllt wird. Anschließend wird S_j gewählt ($x_j = 1$ gesetzt) um i zu überdecken.

Beispiel 1. Die Elemente i werden durch y_i beschriftet für eine (optimale) Lösung y von (D).



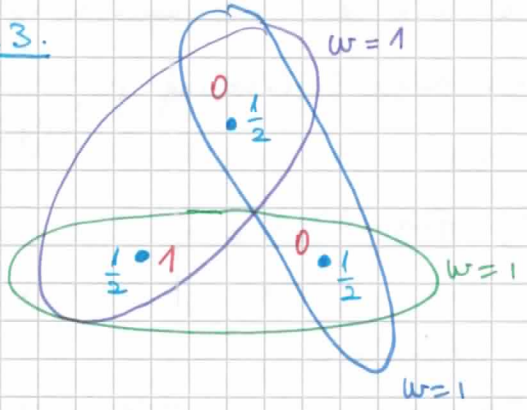
eine (optimale) Lösung y die der P-D Alg finden kann, ist schwarz

Beispiel 2.



nicht-optimales y gefunden vom P-D Algorithmus
(aber die Überdeckung d.h. das I.P wird hier optimal gelöst)

Beispiel 3.



Der P-D Alg. kann hier kein optimales y finden, aber die (ganzzahlige) Überdeckung die er findet (zwei Mengen) ist natürlich optimal.

LPD 22.

Primal-Dualer Algorithmus für SET COVER

- starte mit $y = 0 \quad x = 0$

es gelten die primalen komp. Slack. Bedingungen

$$\left(w_j - \underbrace{y^T \cdot a_j}_{\sum_{i \in S_j} y_i}\right) \cdot x_j = 0 \qquad (*)$$

y ist Lösung von (D) aber x keine Lösung von (P)

d.h. noch keine Mengenüberdeckung

→ $\exists\, k \in \{1, 2, \ldots, m\}$ noch nicht überdeckt

→ wir müssen x_j erhöhen für eine j mit $k \in S_j$

→ da $x_j \neq 0$, soll $w_j - y^T \cdot a^j = 0$ werden damit $(*)$ weiter gilt ABER: x_j zuerst, und dann y erhöhen wird schief gehen bei schlechter Wahl von x_j!

→ wir machen umgekehrt: wir erhöhen y_k für das unbedeckte Element k, und beim ersten j wo $w_j - y^T \cdot a^j = 0$ wird, setzen wir $x_j = 1$

(dies kann nur für j mit $k \in S_j$ vorkommen)

WHILE es gibt $k \in \{1, 2, \ldots, m\}$ nicht überdeckt

- erhöhe y_k so hoch wie die Bedingungen in (D) erlauben, d.h. bis eine Bedingung j $\sum_{i : i \in S_j} y_i \leq w_j$ exakt erfüllt wird.

(Dann gilt $k \in S_j$ für diese j.)

die Erhöhung von y_k beeinflusst nur die Bedingungen j für $k \in S_j$

- setze $x_j = 1$ (wir nehmen S_j in die Überdeckung

$(w_j - y^T \cdot a^j) \cdot x_j = 0$ gilt weiterhin: „$y^T \cdot b$ ist nicht weit von $c^T \cdot x$"

- gib x als Überdeckung aus

$$C := \{j \mid x_j = 1\}$$

Laufzeit: $\leq m$ Iterationen, da jede Iteration mindestens ein Element der Grundmenge überdeckt.

Analyse der Approximation:

Thm: Wenn jedes Element in höchstens l Teilmengen enthalten ist, dann ist die primal-duale Lösung l-approximativ.

Beweis: Für jede S_j in der Mengenüberdeckung bezahlen wir w_j in der Zielfunktion.

Idee: Zählen wir (summieren wir) nicht alle diese w_j, sondern die $\sum_{i : i \in S_j} y_i$ für die Elemente in S_j.

Dann wird jedes y_i maximal l-mal in der Summe auftreten, weil i in höchstens l Teilmengen enthalten ist. Wir nutzen aus, dass für alle Teilmengen S_j für die $x_j = 1$ gesetzt wurde, $\sum_{i : i \in S_j} y_i = w_j$ gilt.

(diese y_i Werte werden später im Algorithmus nicht modifiziert, da diese Elemente von S_j schon überdeckt wurden, bzw. erhöhen dürfte man sie eh nicht)

$$\sum_{j=1}^{n} w_j x_j = \sum_{j \in C} w_j = \sum_{j \in C} \sum_{i : i \in S_j} y_i = \sum_{i=1}^{m} \sum_{\substack{j : i \in S_j \\ j \in C}} y_i \leq \sum_{i=1}^{m} l \cdot y_i =$$

$$= l \sum_{i=1}^{m} y_i = l \, y^T \cdot b \leq l \cdot OPT_{frac} \leq l \cdot OPT_{ganz}$$

die y_i Preise werden spaltenweise summiert

die y_i werden zeilenweise summiert (für jede $a_{ij} = 1$ in A)

□

Was war im obigen P-D Algorithmus mit den dualen Komplementäre-Slackness-Bedingungen los?

diese sind / wären

$$y_i \underbrace{(a_i^T \cdot x}_{\text{wieviele Mengen überdecken Element } i \text{ in der Überdeckung } x} - b_i) = 0 \qquad i = 1, 2, \dots, m$$

$$\Big\| \; 1$$

→ die können nicht exakt gelten, weil die primalen $(c_j - y^T a^j) x_j = 0$ gelten, und eine ganzzahlige Lösung x die der Alg. findet, allgemein keine minimale (fraktionale) Lösung des LP sein kann. Wir wollen aber diese Terme auch möglichst klein haben wegen der Approximationsfaktor. (damit $c^T \cdot x$ und $y^T \cdot b$ nahe sind)

→ Wenn jedes Element i in maximal l Teilmengen S_j enthalten ist, dann

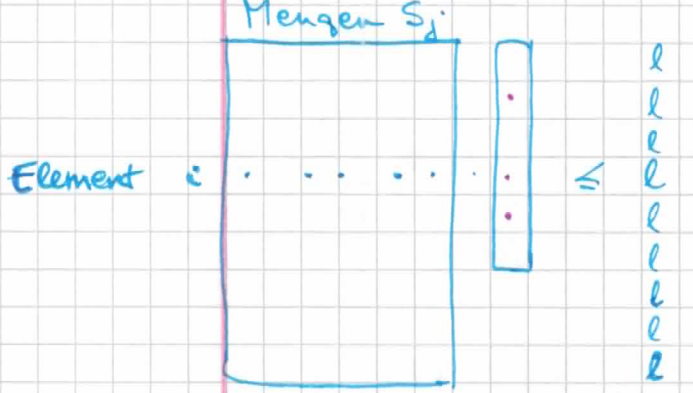
STATT $\quad y_i > 0 \Rightarrow a_i^T \cdot x = b_i$ (genau eine Menge überdeckt Element i)

(duale compl. slackness)

GILT $\quad (y_i > 0 \Rightarrow) \; a_i^T \cdot x \leq l \cdot b_i$

weil $b_i = 1$

$a_i^T \cdot x$ = Anzahl der gewählten Mengen die Element i enthalten



Dies ermöglichte den Approximationsfaktor l

(Shmoys-Williamson Chapter 7.3)

Beispiel 2. Primal-dual Algorithmus für das shortest-s-t-Path Problem

(Schulbeispiel für ein exakt (effizient) approximierbares Problem)

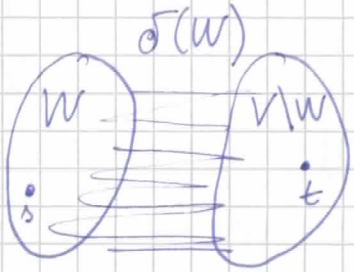
Eingabe: ungerichteter Graph $G(V, E)$
- mit einer Länge $l_e \geq 0$ für jede Kante $e \in E$
- zwei ausgezeichnete Knoten $s, t \in V$

Ausgabe: ein kürzester s-t Pfad

Definitionen:

→ eine Knotenmenge $W \subseteq V$ mit $s \in W$ $t \notin W$ definiert einen s-t Schnitt, d.h. eine Partition $(W, V \setminus W)$ so dass $s \in W$ und $t \in V \setminus W$.
Der Einfachheit halber nennen wir eine solche W auch einen s-t-Schnitt. (nicht nur die Partition)

→ Bezeichne $\delta(W)$ die Menge aller kreuzenden Kanten im Schnitt $(W, V \setminus W)$



→ Sei $\mathcal{S} = \{W \subseteq V \mid s \in W, t \notin W\}$ die Menge aller s-t-Schnitte

24/6

Eine IP-Formulierung des Shortest-s-t-Path Problems

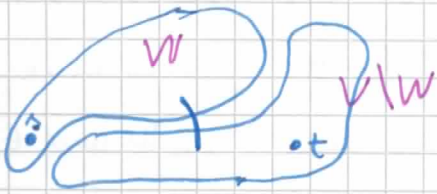
(für konkrete Instanz (G, l, s, t))

→ wir nehmen eine Variable x_e für jede Kante $e \in E$

Wir möchten, dass die Kanten mit $x_e = 1$ in der (optimalen) Lösung einen s-t-Pfad ergeben, oder zumindest einen s-t-Pfad enthalten.

→ Diesmal fordern wir nicht die Flusserhaltung-Bedingungen!

Wir fordern, dass jeder s-t-Schnitt von mindestens einer Kante e mit $x_e = 1$ überquert wird



Somit wird jede Lösung (Kanten mit $x_e = 1$) einen s-t-Pfad enthalten (siehe Behauptung 1).

IP-Formulierung:

$$\text{minimiere} \sum_{e \in E} l_e \cdot x_e$$

$$\text{so dass} \sum_{\{e \mid e \in \delta(W)\}} x_e \geq 1 \qquad \forall\, W \subsetneq S$$

$\{e \mid e \in \delta(W)\}$: e kreuzt $(W, V \setminus W)$

$\forall\, W \subsetneq S$: für jeden s-t-Schnitt

$$x_e \in \{0, 1\} \qquad \forall\, e \in E$$

Behauptung 1: Die Kanten mit $x_e = 1$ enthalten genau dann einen s-t-Pfad, wenn x die Bedingungen im IP für jeden s-t-Schnitt W erfüllt.

Beweis:

1. Die Bedingungen, dass jeder s-t-Schnitt von einer Kante überquert wird, sind notwendig:

   Trivial. x enthält einen s-t Pfad, und dieser muss jeden s-t-Schnitt überqueren (mindestens einmal)

2. Die Bedingungen sind auch hinreichend:

   Dh. eine solche Lösung x enthält mindestens einen s-t Pfad. (d.h. die Kantenmenge mit $x_e = 1$ enthält einen Pfad)

   Wir zeigen 2.: Sei x eine Lösung des IP, und seien

   $$E' = \{e \in E \mid x_e = 1\}$$

   $$W' = \{v \in V \mid v \text{ erreichbar aus } s \text{ über Kanten in } E'\}$$

   Angenommen, dass E' keinen s-t Pfad enthält (durch Widerspruch), dann $t \notin W'$

   $\Rightarrow (W', V \setminus W')$ ist ein s-t-Schnitt, ohne kreuzende Kante aus E', also ohne kreuzende Kante mit $x_e = 1$ (weil die Knoten ausserhalb W' nicht erreichbar sind)

   $\Rightarrow$ die lineare Nebenbedingung $\sum_{\{e \mid e \in \delta(W')\}} x_e \geq 1$ für diesen W' s-t-Schnitt wird verletzt, also x war doch keine Lösung. $\lightning$ □

→ in der LP-Relaxierung des IP wird $x_e \in \{0,1\}$ durch $0 \leq x_e$ ersetzt für jede $e \in E$.

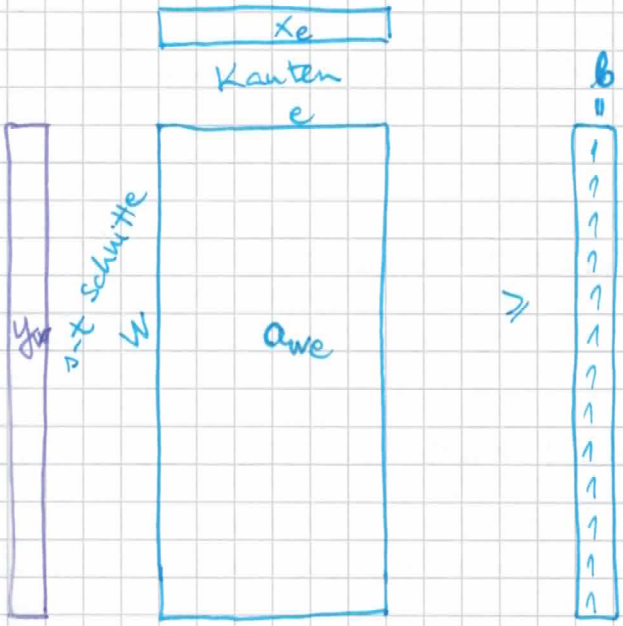
($x_e \leq 1$ wird in einer optimalen Lösung automatisch erfüllt.)

→ Beachte: Wir haben exponentiell viele Nebenbedingungen! (Warum? Wieviele s-t-Schnitte gibt es?)

ABER: wir werden das LP gar nicht lösen müssen!

<u>Wir formulieren das duale Programm der LP-Relaxierung</u>

(P)



$c^T =$ | l_e |

a_{We}: ob e kreuzt Schnitt W

→ Im dualen Programm (D) haben wir eine y_W Variable für jeden s-t-Schnitt W

→ und eine Nebenbedingung für jede Kante $e \in E$

maximiere $\sum_{w \in S} y_w$ $(y^T \cdot b)$

(D) so dass

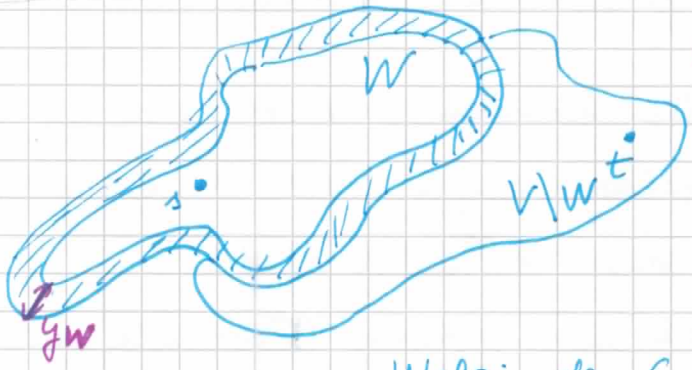
$$\sum_{\{W \mid e \in \delta(W)\}} y_w \leq l_e \qquad \forall e \in E$$

alle s-t-Schnitte die e kreuzt

$y_w \geq 0$

(Die Summe der y_w für alle s-t-Schnitte W die e kreuzt ist höchstens l_e)

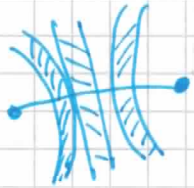
→ Interpretation der y_w: mindestens so viel kostet es noch im Zielwert (Pfadlänge) den Schnitt $(W, V \setminus W)$ zu überqueren.



(y_w: die Breite eines "Grabens" um W herum, wobei die Kanten "Brücken" entsprechen.

Wobei der Graph allgemein nicht planar ist; W nicht "links" und $V \setminus W$ nicht "rechts" ist sondern überall im Graph sein kann

Die Graben verschiedener s-t-Schnitte überlappen sich nicht ⇒ jede Kante e kann Graben mit Gesamtbreite $\leq l_e$ überqueren.

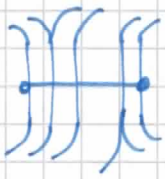


Ein s-t Pfad muss jeden s-t-Schnitt (jeden Graben) überqueren, und hat Gesamtlänge $\geq \sum_{w \in S} y_w$)

→ die PKS-Bedingungen: (primale komplementäre-Slackness Bedingungen)

entweder $x_e = 0$ (Kante e nicht gewählt in der Lösung)

oder $\sum_{W \mid e \in \delta(W)} y_W = l_e$



die Gesamtbreite der Graben die e kreuzt, ist genau die Länge der Kante e

Primal-dualer Algorithmus

Eingabe: $G(V,E)$, $l: E \to \mathbb{R}_+$, $s,t \in V$

- starte mit $y=0$ $x=0$ $F=\emptyset$

(die Kanten e mit $x_e=1$ bilden nach jeder Runde einen Baum $F \subseteq E$ (statt Pfad))

- WHILE x noch keine Lösung (F enthält noch keinen s-t Pfad)
  - ~~sei $F = \{e \in E \mid x_e = 1\}$~~
  - sei W die Menge aller Knoten erreichbar über Kanten in F aus s (Zusammenhangskomponente von s)

(wir müssen $(W, V \setminus W)$ mit einer weiteren Kante überqueren

aber mit welcher?

Wir wählen die neue Kante nicht beliebig, sondern die y Lösung hilft zu wählen.)

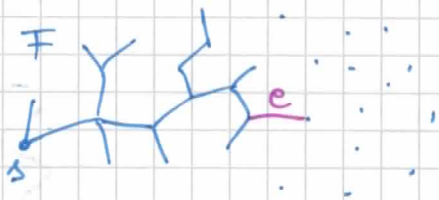
- erhöhe y_W bis für irgendeine Kante $e \in \delta(W)$ gilt:

$$\sum_{W' \mid e \in \delta(W')} y_{W'} = l_e$$

- setze $x_e = 1$ für diese Kante
  $F = F \cup \{e\}$
  (e kommt zu den Baumkanten F)

(die PKS-Bedingungen bleiben erfüllt)

Wichtige Bemerkung: F bleibt ein Baum:



weil e den bisherigen Baum F mit einem nicht-Baum-Knoten verbindet

— am Ende enthält F einen s-t Pfad, nenne ihn P; da F ein Baum ist, ist dieser s-t Pfad $P = P^F$ eindeutig (der einzige s-t Pfad in F), und ihn gibt der Algorithmus aus

---

Theorem: Dieser Primal-dual Algorithmus gibt einen kürzesten s-t-Pfad aus.

Beweis: Wir nutzen wieder, dass die PKS-Bedingungen gelten, also, dass für jede Kante $e \in F$ (d.h. mit $x_e = 1$) gilt

$$l_e = \sum_{\{W' \mid e \in \delta(W')\}} y_{W'}$$

P C F war die Kantenmenge des Pfades ausgegeben vom Algorithmus. Sei $x' \leq x$ der 0-1 Vektor, der P entspricht (statt x, der F entspricht)

Länge des s-t Pfades P $= \sum_{e \in P} l_e = \sum_{e \in P} \sum_{\{W \mid e \in \delta(W)\}} y_W = \sum_{W \in S} \sum_{\substack{e \in P \\ e \text{ kreuzt } W}} y_W =$

$\sum_e x'_e \cdot l_e = c^T \cdot x'$

Summe der y_W Spaltenweise

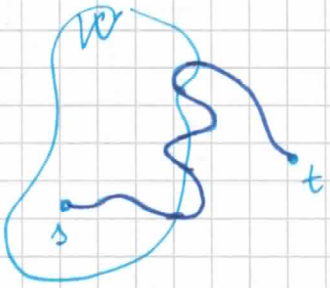
jeder s-t-Schnitt

Summe der y_W Zeilenweise

wie oft kreuzt P den $(W, V \setminus W)$

$= \sum_{W \in S} |\delta(W) \cap P| \cdot y_W$

$= 1$ falls $y_W > 0$



Brauchen: P kreuzt W genau 1 mal, falls $y_W > 0$ ♦

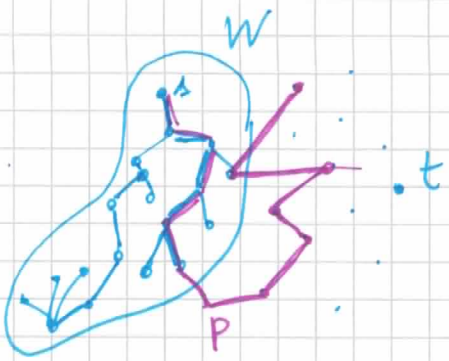
$= \sum_{W \in S} y_W = \sum_{W \in S} y_W \cdot 1 = y^T \cdot b$

↘ Wert der gebildeten dualen Lösung y

$\Rightarrow$ der karakteristische Vektor x' des Pfades P ist eine optimale Primale Lösung, weil eine duale Lösung y den selben Wert hat.

♦ sagt, dass in diesem Algorithmus sogar die DKS (Duale Komplementäre Slackness Bedingungen) gelten für die gefundenen Lösungen x und y (wobei x nur für den Pfad P der kar. Vektor ist)

Warum gilt ◆ ?



Pfadkante

Kante des damaligen F als y_w erhöht wurde

- als der Algorithmus y_w hochgesetzt hat ($y_w > 0$)

enthielt ~~bestand~~ der damalige Baum, gespannt durch die Kanten von F, genau ~~an~~ die Knoten W

- wenn der Pfad P ~~den~~ den Schnitt W ≥2 mal kreuzen würde, dann würde F am Ende einen Kreis enthalten ↯.

(weil $F_{damals} \subset F_{letzter}$, $P \subset F_{letzter}$)

Allgemein:

→ da x' und y beide optimal waren (da $c^T \cdot x = y^T \cdot b$)

gelten auch die DKS (Duale Komplementäre Schlaffheits-Bedingungen), weil (mit allgemeiner Notation)

$$c^T \cdot x' - y^T \cdot b = \underbrace{\sum_{j=1}^{n} (c_j - y^T \cdot a^j) \cdot x'_j}_{=0 \text{ (PKS)}} + \underbrace{\sum_{i=1}^{m} y_i \, (a_i^T \cdot x' - b_i)}_{=0 \text{ (DKS)}} = 0$$

in unserem Fall $a_w^T \cdot x'$ ist die Anzahl der Kanten von P die W kreuzen und $b_w = 1$

DKS: $y_w = 0$, oder genau eine Kante von P kreuzt W ◆

→ Anders geschrieben: die Analyse des P-D Algorithmus entspricht:

$$c^T \cdot x' = (y^T \cdot A) \cdot x' = y^T \cdot (A x') = y^T \cdot b$$

$\Uparrow$ (PKS) $\qquad$ $\Uparrow$ (DKS)

→ Jetzt zurück zur Analyse vom P-D-Algorithmus für SET COVER:

DKS für SET COVER:

in der LP-Formulierung (P) und (D) für SET COVER, $a_i^T \cdot x$ ist die Anzahl der Teilmengen in der Überdeckung C, die Element i überdecken, und $b_i = 1$

DKS: $y_i = 0$, oder genau eine Teilmenge aus der Lösung C enthält Element i

→ die Analyse des P-D Algorithmus für SET COVER entspricht

$$c^T \cdot x = (y^T \cdot A) \cdot x = y^T \cdot (Ax) \leq \ell \cdot y^T \cdot b$$

$\Uparrow$ (PKS)

$\Uparrow$

$a_i^T \cdot x \leq \ell \cdot b_i = \ell$

weil jedes Element in höchstens ℓ Teilmengen enthalten

Statt der DKS: $y_i = 0$ oder $a_i^T \cdot x = b_i$

hatten wir dort

relaxierte DKS-Bedingungen: $y_i = 0$ oder $a_i^T \cdot x \leq \beta \cdot b_i$

(mit $\beta = \ell$). Diese ergeben einen β-approximativen Primal-Dual Algorithmus.

Wir verallgemeinern diese Beobachtungen:

Theorem:

Wenn für die Lösungen x von (P) und y von (D) gelten die primalen Komplementäre-Slackness (PKS) Bedingungen:

$$x_j > 0 \Rightarrow y^T \cdot a^j = c_j$$

und die relaxierten dualen Komplementäre Slackness (DKS) Bedingung

$$y_i > 0 \Rightarrow a_i^T \cdot x \leq \beta \cdot b_i$$

für irgendein $\beta > 0$, dann ist die Lösung x (und auch y) β-approximativ.

Wann? mit analogem Beweis wie beim Komplementären Slackness oben

$$c^T \cdot x - y^T \cdot \beta \cdot b = (c^T - y^T \cdot A) \cdot x + y^T \cdot (A \cdot x - \beta \cdot b) =$$

$$= \underbrace{\sum_{j=1}^{n} (c_j - y^T \cdot a^j) \cdot x_j}_{=0} + \underbrace{\sum_{i=1}^{m} y_i \cdot (a_i^T \cdot x - \beta b_i)}_{\leq 0} \leq 0$$

$$\Rightarrow \quad c^T \cdot x \leq \beta \cdot y^T \cdot b \leq \beta \cdot OPT_{frac.} \leq \beta \cdot OPT_{ganz}$$

(wobei OPT_{ganz} das primale ganzzahlige Optimum bedeutet)