$$E[\overset{y_i}{\cancel{X_i}}] = 1 \cdot \text{Prob}\,(P_i \text{ benutzt } e) + 0 \cdot \text{Prob}\,(P_i \text{ benutzt } e \text{ nicht}) =$$

$$= 1 \cdot \sum_{j:\, e \in P_{i,j}}{}' \alpha_{i,j} \le x_i(e) \longrightarrow$$ erwartete Belastung über e wegen P_i

erwartete Belastung über e wegen $P_1 P_2 \ldots P_i \ldots P_r$

$$E[\overset{y}{\cancel{X}}] \le \sum_{i=1}^{r} x_i(e) \le W_{OPT}^{frac.} \le W_{OPT}.$$

Die maximale erwartete Belastung über Kanten ist somit $\le W_{OPT}$

<u>Bemerkung:</u> Die erwartete <u>maximale</u> Belastung über Kanten könnte trotzdem groß werden. Aber: mit Prob $> 1-\varepsilon$ ist die maximale Belastung $\le W_{OPT} + \sqrt{3 W_{OPT} \cdot \ln \frac{n^2}{\varepsilon}}$

6.) MAX-SAT

<u>Beispiel:</u> eine Formel in konjunktiver Normalform (CNF)

$$(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2)$$

gesucht wird eine Wahrheitsbelegung

$$\{x_1, x_2\} \longrightarrow \{\text{TRUE}, \text{FALSE}\},$$

die eine maximale Anzahl von Klauseln erfüllt

Was ist das Optimum in unserem Beispiel?

(Variablen: $x_1\ x_2$

Literale: $x_1\ \neg x_1\ x_2\ \neg x_2$

Klausel: $(x_1 \vee \neg x_2)$ usw.)

<u>gewichtetes max-SAT</u>

<u>Eingabe:</u> eine Menge K von Klauseln, wobei Klausel k das Gewicht w_k besitzt mit Literalen aus $\{x_1, \neg x_1, x_2, \neg x_2 \; \ldots \; x_n, \neg x_n\}$

<u>Ausgabe:</u> Eine Wahrheitsbelegung die eine Klauselmenge von größtmöglichem Gesamtgewicht erfüllt.

LP 55.

Algorithmus 1. deterministische 2-Approximation → ist einfach

siehe Übung

Algorithmus 2. Randomisierung für lange Klauseln

FOR $i=1$ to n DO:

setze $x_i = 1$ mit Wahrscheinlichkeit $\frac{1}{2}$ ~~und~~ bzw.

$x_i = 0$ mit Wahrscheinlichkeit $\frac{1}{2}$

Lange Klausel werden mit hoher Wahrscheinlichkeit erfüllt.
(z.B. $x_1 \vee x_2 \vee \neg x_4 \vee x_6 \vee \neg x_7$) $l=5$

Sei k eine Klausel der Länge l

$$\text{Prob}(k \text{ wird } \underline{\text{nicht}} \text{ erfüllt}) = \underbrace{\frac{1}{2}\cdot\frac{1}{2}\cdot\frac{1}{2}\cdot\ldots\cdot\frac{1}{2}}_{l} = \left(\frac{1}{2}\right)^l$$

$$\text{Prob}(k \text{ wird erfüllt}) = 1 - \frac{1}{2^l}$$

Insbesondere: Wenn alle Klauseln mindestens l Literale besitzen, dann ist das erwartete Gewicht erfüllter Klauseln

mindestens $$\sum_k \left(1-\frac{1}{2^l}\right)\cdot w_k = \left(1-\frac{1}{2^l}\right)\sum_k w_k \geq \left(1-\frac{1}{2^l}\right) W_0$$

(Warum?

Sei die Zufallsvariable $V_k = \begin{cases} 1 & \text{wenn Klausel } k \text{ erfüllt wurde} \\ 0 & \text{sonst} \end{cases}$

Das Gewicht aller erfüllten Klausel ist dann die Zufallsvariable

$\sum_k w_k \cdot V_k$, und das erwartete Gewicht

$$E\left[\sum_k w_k V_k\right] = \sum_k w_k \, E[V_k] \quad \text{wobei } E[V_k] \geq \left(1-\frac{1}{2^l}\right)\cdot 1 + \frac{1}{2^l}\cdot 0 = \left(1-\frac{1}{2^l}\right)$$

IP-Formulierung

- eine Variable y_i wird definiert für jede SAT-Variable x_i
- es wird auch eine Variable z_k für jede Klausel k definiert

Bedeutung: y_i ist die Belegung von x_i (0 oder 1)

$$z_k = \begin{cases} 1 & \text{falls Klausel } k \text{ erfüllt wird} \\ 0 & \text{sonst} \end{cases}$$

maximiere $$\sum_{k \in K} w_k \cdot z_k$$

so dass $$\sum_{x_i \in k} y_i + \sum_{\neg x_j \in k} (1 - y_j) \geq z_k \quad \forall k \in K$$

$$z_k, y_i \in \{0, 1\}$$

LP-Relaxierung $\quad 0 \leq z_k, y_i \leq 1$

Algorithmus 3: Randomisiertes Runden (für kurze Klauseln)

- löse die LP-Relaxierung; sei (y^*, z^*) eine optimale fraktionale Lösung
- FOR $i = 1$ to n DO
  setze $x_i = \text{TRUE}$ zufällig mit Wahrscheinlichkeit y_i^*

Bemerkung:

- für Klausel der Länge ≥ 3 ist die einfache Randomisierung (Alg. 2) besser! Warum?

für die Klausel $x_1 \vee x_2 \vee \ldots \vee x_\ell$ $y_i = \frac{1}{2}$ kann schon optimal sein für das LP

- Wenn es nur Klausel der Länge 2 gibt, dann ist die fraktionale Lösung $y_i = \frac{1}{2}$, $z_k = 1$ optimal, und die zwei Algorithmen sind äquivalent.

- für Klausel der Länge 1 ist das randomisierte Runden (Alg 3.) besser.

Algorithmus 4.

- bestimme eine Belegung mit Algorithmus 2.
- bestimme eine Belegung mit Algorithmus 3.
- gib die bessere Lösung aus!

Theorem: Das erwartete Gewicht belegter Klauseln ist mindestens

$$\frac{3}{4} \cdot \underbrace{\sum_{k \in K} w_k \cdot z_k^*}_{= \frac{3}{4} \mathrm{OPTfrac}(I)} \geq \frac{3}{4} \cdot \mathrm{OPT}(I)$$

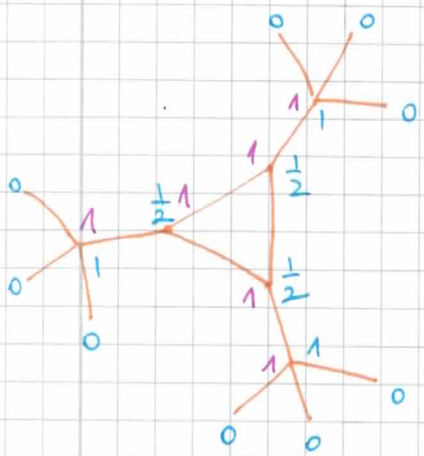
(ohne Beweis)

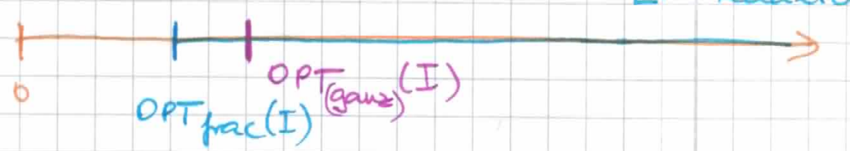
Die Integralitätslücke (integrality gap)

Def: Die Integralitätslücke ist der schlimmste mögliche (größtmögliche) Quotient des ganzzahligen (also eigentlichen) und des fraktionalen Optimum-Wertes für ein gegebenes Problem (-Typ).

Die Integralitätslücke

Beispiel 1. Ein Minimierungsproblem: VERTEX COVER





$OPT_{frac} \leq OPT$ gilt für jede Instanz

→ uns interessiert das (größt) mögliche Verhältnis $\dfrac{OPT}{OPT_{frac}}$ über alle Eingaben $G(V,E), w_i \ (i \in V)$

→ Finden wir einfache Eingaben für VERTEX COVER, wo dieses Verhältnis ganz klein (d.h. $OPT_{frac} \approx OPT$)

→ Wie groß kann das Verhältnis sein?
Wir wissen von der Analyse des deterministischen Rundens: das optimale fraktionale x^* aufgerundet ergibt eine ganzzahlige Lösung mit höchstens doppeltem Wert $2 \cdot OPT_{frac}$

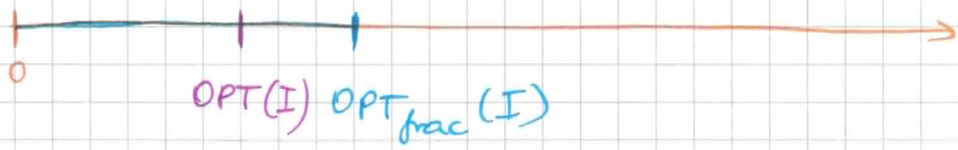
↳ für VERTEX COVER das Verhältnis $\dfrac{OPT(I)}{OPT_{frac}(I)}$
→ ist höchstens 2
→ kann $2 - \frac{1}{n}$ sein ⇝ für einen vollständigen Graph

Für das VERTEX COVER Problem die Integralitätslücke

$$\sup_I \frac{OPT(I)}{OPT_{frac}(I)}$$ ist zwischen $2-\frac{1}{n}$ und 2.

Beispiel 2. Ein Maximierungsproblem: INDEPENDENT SET

$I = G(V, E)$ Graph (gewichtet/ungewichtet)



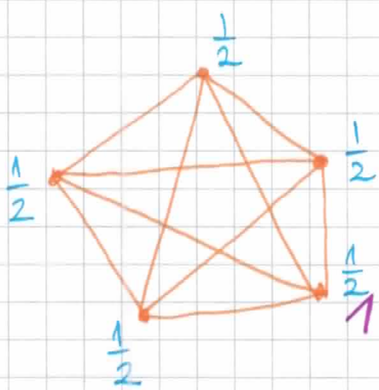
Wie groß kann das Verhältnis $\dfrac{OPT_{frac}(I)}{OPT(I)}$

über alle Eingabeinstanzen I werden?

→ Wir haben ein Beispiel I mit

$$\frac{OPT_{frac}(I)}{OPT(I)} = \frac{n}{2}$$

den vollständigen Graphen



⇝ die Integralitätslücke für das INDEPENDENT SET Problem ist mindestens $\frac{n}{2}$

(auch höchstens – Warum?)

Pelikan

→ Sei P ein Minimierungsproblem.
Für gegebene Instanz I sei $OPT(I)$ der optimale (ganzzahlige) Wert, und $OPT_{frac}(I)$ der optimale (minimale) Wert ~~einer~~ über fraktionale Lösungen der LP-Relaxierung.

→ Generell gilt $OPT_{frac}(I) \leq OPT(I)$ da eine ganzzahlige Lösung auch immer eine fraktionale Lösung ist.

→ Die <u>Integralitätslücke</u> ist das "Worst-Case" Verhältnis

$$\sup_I \frac{OPT(I)}{OPT_{frac}(I)} \quad \text{für Minimierungsprobleme,}$$

und

$$\sup_I \frac{OPT_{frac}(I)}{OPT(I)} \quad \text{für Maximierungsprobleme}$$

<u>Beispiel 1.</u> Für VERTEX COVER ist die Integralitätslücke

- <u>mindestens</u> $\frac{n-1}{\frac{n}{2}} = 2\left(1-\frac{1}{n}\right)$

weil für den vollständigen Graphen mit n Knoten $n-1$ ist das Optimum, und $\frac{n}{2}$ (für $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}\right)$) das fraktionale Optimum.

- aber auch <u>höchstens</u> 2 (für beliebige Instanz deterministisches Runden ergibt eine ganzzahlige Lösung mit $2 \cdot OPT_{frac}$ Wert.

<u>Beispiel 2.</u> Für INDEPENDENT SET ist die Integralitätslücke mindestens $\frac{n}{2}$ weil für den vollständigen Graphen

$$\frac{OPT_{frac}}{OPT} = \frac{\frac{n}{2}}{1} = \frac{n}{2}.$$

(auch höchstens $\frac{n}{2}$
Warum?)

max-BIPARTITE MATCHING

Für Bipartite Graphen hat die Inzidenzmatrix eine wichtige Eigenschaft:

Definition: Eine Matrix A heißt vollständig unimodular, falls die Determinante $\det B \in \{-1, 0, 1\}$ für jede quadratische Teilmatrix B von A gilt.
(Insbesondere sollen alle Einträge 1, 0 oder −1 sein)

Theorem 1. Die Inzidenzmatrix für einen ungerichteten Graphen ist genau dann vollständig unimodular, wenn der Graph bipartit ist. (ohne Beweis)

$\{(\Longrightarrow)$ ungerader Kreis: hat $\det = 2$

$$\begin{pmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ & & 1 & 1 & \\ & & & 1 & 1 \\ 1 & & & & 1 \end{pmatrix}$$

Theorem 2. Falls A vollständig unimodular, und b ein ganzzahliger Vektor ist, dann haben die Ecken des Lösungspolyeders $\{x \mid Ax = b, x \geq 0\}$ (bzw. $\{x \mid Ax \geq b\}$) nur ganzzahlige Koordinaten. (ohne Beweis)

$\Longrightarrow$ ~~das Optimum~~ ein solches ~~Optimierungsproblem~~ LP besitzt eine ganzzahlige optimale Lösung

Korollar: Das maximum-Gewicht MATCHING Problem für Bipartite Graphen ist (auch) mit linearer Programmierung exakt lösbar. (Warum?)

Theorem 3. Die Inzidenzmatrix eines gerichteten Graphen ist immer vollständig ganzzahl. unimodular.

$$\begin{array}{c|c|} & (v,w) \\ \hline v & 1 \\ w & -1 \\ \hline \end{array}$$

$\Longrightarrow$ Flussprobleme haben ganzzahlige Lösungen

4.) SET COVER

Eingabe: Grundmenge $\{1, 2, 3, \ldots, m\}$

n Teilmengen $S_j \subseteq \{1, 2, \ldots, m\}$ $\quad j = 1, 2, \ldots, n$

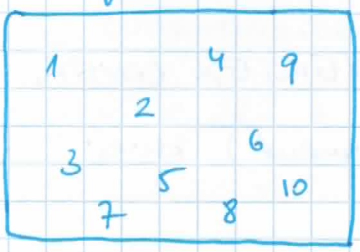
mit Gewichten w_j

Ausgabe: Eine Überdeckung von $\{1, 2, \ldots, m\}$ mit einer leichtesten Auswahl von Teilmengen $C \subseteq \{1, 2, \ldots n\}$

(C ist die Indexmenge der ausgewählten Teilmengen S_j.
C ist eine Überdeckung der Grundmenge, wenn jedes Element $1 \le i \le m$ ist Element von mindestens einer ausgewählten Teilmenge $i \in S_j$ ($j \in C$))

Beispiel:

Grundmenge



$S_1 = \{3, 4, 5, 8\}$

$S_2 = \{1\}$

$S_3 = \{1, 3, 5, 7, 9\}$

$S_4 = \{2, 4, 10\}$

$S_5 = \{2, 3, 4, 5, 6\}$

$S_6 = \{2, 6, 10\}$

$\{S_1, S_3, S_4, S_5\}$ und $\{S_1, S_3, S_6\}$ sind verschiedene Mengenüberdeckungen.

LP-Relaxierung:

minimiere $\displaystyle\sum_{j=1}^{n} x_j \cdot w_j$

Element i überdeckt wird $\xleftarrow{\text{so dass}}$ $\displaystyle\sum_{j:\, i \in S_j} x_j \ge 1 \quad \forall\, i = 1, 2, \ldots, m$

(m Bedingungen)

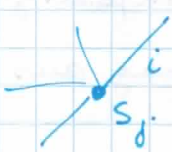
$0 \le x_j (\le 1)$

beabsichtigte Bedeutung: $x_j = \begin{cases} 1 & \text{falls } S_j \text{ in der Überdeckung } (j \in C) \\ 0 & \text{sonst.} \end{cases}$

SET COVER ist eine Verallgemeinerung des VERTEX COVER Problems

Kanten $\longleftrightarrow$ Elemente $1, 2, \ldots, m$

Knoten $\longleftrightarrow$ Teilmengen S_j.



Analog zur 2-Approximierung mit deterministem Runden für VERTEX COVER wird folgendes gezeigt:

<u>Thm:</u> Falls jedes Element i der Grundmenge in höchstens l Teilmengen S_j enthalten ist, dann ergibt ein deterministisches Runden der optimalen fraktionalen Lösung eine l-approximative Lösung. (siehe Übung)

(beachte, dass im Spezialfall vom VERTEX COVER $l=2$ gilt, da jede Kante (=Element) zwei Endknoten (=Teilmengen) hat.)

<u>Für den allgemeinen Fall ohne solche l:</u>

<u>Ein Approximationsalgorithmus</u> (<u>randomisiertes Runden</u>)

(<u>die Idee</u>: wir lösen die LP-Relaxierung. Die optimale fraktionale Lösung $(x_1^* x_2^* \dots x_n^*)$ bestimmt die Wahrscheinlichkeiten x_j^*, dass die entsprechende Teilmenge S_j in die Mengenüberdeckung gewählt wird $(Prob(j \in C) = x_j^*)$. ABER: erhalten wir so tatsächlich eine Überdeckung aller Elemente der Grundmenge? NEIN)

$\Longrightarrow$ mehrere (T) solche zufällige "Überdeckungen" werden vereinigt:

- löse die LP-Relaxierung;

  sei $x^* = [x_1^* x_2^* \dots x_n^*]$ eine optimale fraktionale Lösung

- FOR $t=1$ to T do
  - setze $C_t = \emptyset$
  - FOR $j=1$ to n do

    sei $j \in C_t$ mit Wahrscheinlichkeit x_j^*

- gib die Vereinigung $C = \bigcup_{t=1}^{T} C_t$ der T zufällig gewählten "Überdeckungen" C_t als Lösung aus.

Die Approximierbarkeit vom SET COVER Problem

Für SET COVER haben wir die folgenden Algorithmen gesehen:

m: Anzahl der ~~zu~~ überdeckenden Elemente

LP-basierte Algorithmen

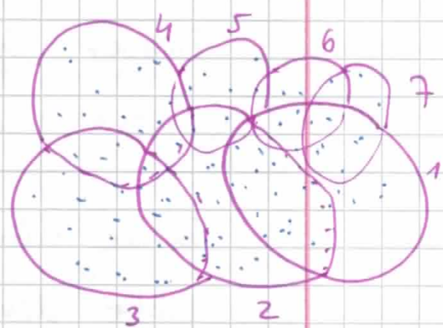
mit deterministischem Runden: ℓ-approximativ WENN jedes Element in max. ℓ Teilmengen enthalten

mit randomisiertem Runden: erwarteter Approximationsfaktor für beliebige Instanzen $\leq \ln m + 1$

Primal-Dual Algorithmus: ℓ-approximativ WENN jedes Element in max ℓ Teilmengen enthalten

Hier sehen wir noch einen einfachen Greedy Algorithmus:

Greedy Algorithmus für SET COVER



Aufwärmung:

für ungewichtetes SET COVER

→ die Mengen S_j der Überdeckung werden iterativ gewählt;

→ in jeder Runde wird die gewählte S_j die meisten noch nicht überdeckten Elemente überdecken

Greedy Algorithmus für SET COVER

$C := \emptyset$

WHILE nicht alle Elemente überdeckt DO

$C := C \cup \{j\}$ falls S_j die meisten noch nicht überdeckten Elemente überdeckt

---

<u>Gewichtete Variante mit</u> Gewichten $w_1, w_2 \ldots\ldots w_n$ der Teilmengen

bezeichne U die Menge der aktuell überdeckten Elemente

→ setze $C = \emptyset$ $U = \emptyset$

→ WHILE $U \neq \{1, 2, \ldots, m\}$ DO

→ bestimme eine Menge S_k so dass

$$\frac{|S_k \setminus U|}{w_k} \text{ größtmöglich}$$

$$\left(\text{Skript: } \frac{w_k}{|S_k \setminus U|} \text{ minimal}\right)$$

→ setze $U := U \cup S_k$; $C := C \cup \{k\}$

<u>Theorem</u>: Dieser Greedy Algorithmus für SET COVER ist $(\ln m + 1)$-approximativ. (ohne Beweis)

Siehe Übung für Beispiele, dass der Approximationsfaktor <u>dieses</u> Algorithmus <u>mindestens</u> $\ln m$ ist.
(Es gibt sogar Beispiele mit $l=2$, d.h. VERTEX COVER Beispiele, sogar ohne Knotengewichte.)