Das Rucksackproblem

Definition Sprache Rucksack

Gegeben sind n Gegenstände mit Gewichten $W = \{w_1, \ldots, w_n\} \subset \mathbb{N}$ und Profiten $P = \{p_1, \ldots, p_n\} \subset \mathbb{N}$. Seien ferner $b, k \in \mathbb{N}$.

$$\text{RUCKSACK} := \{(W, P, b, k) \mid \exists I \subseteq [n] : \textstyle\sum_{i \in I} w_i \leq b \text{ und } \sum_{i \in I} p_i \geq k.\}$$

Satz

RUCKSACK ist $\mathcal{NP}$-vollständig.

Beweis: zu zeigen

1. RUCKSACK $\in \mathcal{NP}$ (bereits gezeigt)
2. SUBSETSUM $\leq_p$ RUCKSACK

Reduktion $f(M,t) = (W,P,b,k)$

Algorithmus M_f

EINGABE: M, t

1. Setze $b := t$ und $k := t$.
2. For $i = 1$ to n: Setze $w_i := m_i$ und $p_i := m_i$

AUSGABE: W, P, b, k

Laufzeit:

- Eingabelänge: $\log(t) + \sum_{i=1}^{n} \log(m_i)$
- Schritt 1: $\mathcal{O}(\log t)$, Schritt 2: $\mathcal{O}(\sum_{i=1}^{n} \log(m_i))$
- D.h. Gesamtlaufzeit ist polynomiell in der Eingabelänge.

$(M,t) \in$ SubsetSum $\Leftrightarrow f(M,t) \in$ Rucksack

Sei $(M,t) \in$ SubsetSum

- Dann gibt es eine Menge $I \subseteq [n]$ mit $\sum_{i \in I} m_i = t$.
- Damit gilt $\sum_{i \in I} m_i \leq t$ und $\sum_{i \in I} m_i \geq t$.
- Es folgt $\sum_{i \in I} w_i \leq b$ und $\sum_{i \in I} p_i \geq k$.
- Damit gilt $f(M,t) = (W,P,b,k) \in$ Rucksack

Sei $(W,P,B,k) = f(M,t) \in$ Rucksack

- Dann gibt es eine Menge $I \subseteq [n]$ mit $\sum_{i \in I} w_i \leq b$ und $\sum_{i \in I} p_i \geq k$.
- D.h. es gibt eine Menge $I \subseteq [n]$ mit $\sum_{i \in I} m_i \leq t$ und $\sum_{i \in I} m_i \geq t$.
- Setze $S = \{m_i \in M \mid i \in I\}$. Dann gilt $S \subseteq M$ und $\sum_{s \in S} s = t$.
- Damit ist $(M,t) \in$ SubsetSum

Exakte Überdeckung

Definition Exakte Überdeckung

Sei $U = \{u_1, \ldots, u_n\}$ und $F = \{S_1, \ldots, S_m\} \subseteq \mathcal{P}(U)$, d.h. $S_i \subseteq U$.
Eine Menge $C \subseteq F$ heißt *exakte Überdeckung* von U falls

1. $\bigcup_{S_i \in C} S_i = U$
2. $S_i \cap S_j = \emptyset$ für alle $S_i, S_j \in C$ mit $i \neq j$.

$$\textsc{Cover} := \{(U, F) \mid F \text{ enthält eine exakte Überdeckung von } U.\}$$

Bsp:

- $U = \{1, 2, 3, 4, 5\}, F = \{\{2, 3\}, \{1, 3\}, \{4, 5\}, \{1\}\}$
- $C = \{\{2, 3\}, \{4, 5\}, \{1\}\}$ ist eine exakte Überdeckung von U.
- F ist *keine* exakte Überdeckung von U.

$\mathcal{NP}$-Vollständigkeit der exakten Überdeckung

Satz

COVER ist $\mathcal{NP}$-vollständig.

Zeigen

1. COVER $\in \mathcal{NP}$ (Übung)
2. 3SAT $\leq_p$ COVER

Idee der Reduktion

- U enthält alle Variablen x_i, Klauseln K_j und Literale ℓ_{jk}.
- F enthält geeignete Mengen für Variablen, Klauseln und Literale.

Reduktion $f(\phi) = (U, F)$

Algorithmus M_f

EINGABE: $\phi(x_1, \ldots, x_n) = K_1 \wedge \ldots K_m$ mit $K_j = \ell_{j1} \vee \ell_{j2} \vee \ell_{j3}$

1. Setze $U = \{x_1, \ldots, x_n, K_1, \ldots, K_m, \ell_{11}, \ell_{12}, \ell_{13}, \ldots, \ell_{m1}, \ell_{m2}, \ell_{m3}\}$.
2. Definition von F als Vereinigung der Mengen
   - Variablen: $V_{i0} = \{x_i\} \cup \{\ell_{jk} \mid \ell_{jk} = x_i\}$ und
     $V_{i1} = \{x_i\} \cup \{\ell_{jk} \mid \ell_{jk} = \neg x_i\}$ für alle i, j, k.
   - Klauseln: $K_{jk} = \{K_j, \ell_{jk}\}$ für alle $j \in [m], k \in [3]$.
   - Literale: $L_{jk} = \{\ell_{jk}\}$ für alle $j \in [m], k \in [3]$.

AUSGABE: U, F

Laufzeit:

- Eingabelänge von ϕ ist $|\phi| = \Omega(m + n)$
- Schritt 1: $\mathcal{O}(n + m + |\phi|)$
- Schritt 2: Variablen $\mathcal{O}(n + |\phi|)$, Klauseln $\mathcal{O}(m)$, Literale $\mathcal{O}(|\phi|)$.
- D.h. die Laufzeit ist linear in der Eingabelänge.

Bsp.: $(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

- $U = \{x_1, x_2, x_3, K_1, K_2, \ell_{11}, \ell_{12}, \ell_{13}, \ell_{21}, \ell_{22}, \ell_{23}\}$
- $V_{i0} : V_{10} = \{x_1, \ell_{11}\}, V_{20} = \{x_2, \ell_{12}, \ell_{22}\}, V_{30} = \{x_3, \ell_{33}\}$
- $V_{i1} : V_{11} = \{x_1, \ell_{21}\}, V_{21} = \{x_2\}, V_{31} = \{x_3, \ell_{13}\}$
- $K_{1k} : K_{11} = \{K_1, \ell_{11}\}, K_{12} = \{K_1, \ell_{12}\}, K_{13} = \{K_1, \ell_{13}\}$
- $K_{2k} : K_{21} = \{K_2, \ell_{21}\}, K_{22} = \{K_2, \ell_{22}\}, K_{23} = \{K_2, \ell_{23}\}$
- $L_{1k} : L_{11} = \{\ell_{11}\}, L_{12} = \{\ell_{12}\}, L_{13} = \{\ell_{13}\}$
- $L_{2k} : L_{21} = \{\ell_{21}\}, L_{22} = \{\ell_{22}\}, L_{23} = \{\ell_{23}\}$
- Erfüllende Belegung von ϕ: $x_1 = 0, x_2 = 1, x_3 = 1$.

Korrektheit: $\phi \in 3SAT \Rightarrow f(\phi) = (U, F) \in$ COVER

Sei $\phi(x_1, \ldots, x_n) \in 3SAT$

- Dann gibt es eine erfüllende Belegung B der Variablen $x_1, \ldots, x_n$.
- B setzt in jeder Klausel K_j mindestens ein Literal ℓ_{jk} auf wahr.
- Definiere Menge $C \subseteq F$ mittels B:
  - Variablen: Falls $x_i = 0$, nimm V_{i0} in C auf. Sonst V_{i1}.
  - Klauseln: Nimm Menge K_{jk}, die ℓ_{jk} enthält, in C auf.
  - Literale: Für alle nicht von C abgedeckten $\ell_{jk'}$, nimm $L_{jk'}$ in C auf.
- C ist eine exakte Überdeckung, denn
  - Variablen x_i: Werden durch V_{i0} oder V_{i1} abgedeckt.
  - Klauseln K_j: Werden durch K_{jk} abgedeckt.
    Die paarweisen Schnitte der Mengen V_{i0}, V_{i1}, K_{jk} sind *leer*.
  - Literale $\ell_{jk'}$: Werden durch weitere erfüllte Literale aus $L_{jk'}$ abgedeckt.
- Damit ist $(U, F) \in$ COVER

Korrektheit: $f(\phi) = (U, F) \in$ Cover $\Rightarrow \phi \in$ 3SAT

Sei $f(\phi) = (U, F) \in$ Cover

- Dann gibt es eine Menge $C \subseteq F$ mit
  - Die Vereinigung der Mengen in C deckt U ab.
  - Der paarweise Schnitt von Mengen in C ist leer.
- Damit gilt für C
  - Variablen x_i: Entweder ist V_{i0} oder V_{i1} in C.
  - Klauseln K_j: Genau eine Klauselmenge K_{jk} ist in C.
- Definieren Variablen in B: $x_i = 0$ falls $V_{0i} \in C$, sonst $x_i = 1$.
  - Die von den V_{i0}, V_{i1} abgedeckten Literale sind auf falsch gesetzt.
  - Jede Klauselmenge K_{jk} muss ein wahres Literal ℓ_{jk} enthalten.
- D.h. B ist eine erfüllende Belegung.
- Damit gilt $\phi \in$ 3SAT.

Hamiltonscher Kreis

Definition Hamiltonscher Kreis

Sei G ein Graph. Ein Kreis in G, der jeden Knoten genau einmal enthält, heißt *Hamiltonscher Kreis*.
Für gerichtete Graphen definieren wir die Sprache

$$\text{GH-KREIS} := \{G \mid G \text{ gerichtet, } G \text{ besitzt einen Hamiltonschen Kreis.}\}$$

Für ungerichtete Graphen definieren wir analog

$$\text{UH-KREIS} := \{G \mid G \text{ ungerichtet, } G \text{ besitzt Hamiltonschen Kreis.}\}$$

Satz

GH-KREIS ist $\mathcal{NP}$-vollständig.

- Beweis kann mittels COVER $\leq_p$ GH-KREIS geführt werden.
- Wir verzichten hier auf den nicht-trivialen Beweis.

NP-Vollständigkeit von Hamiltonkreis

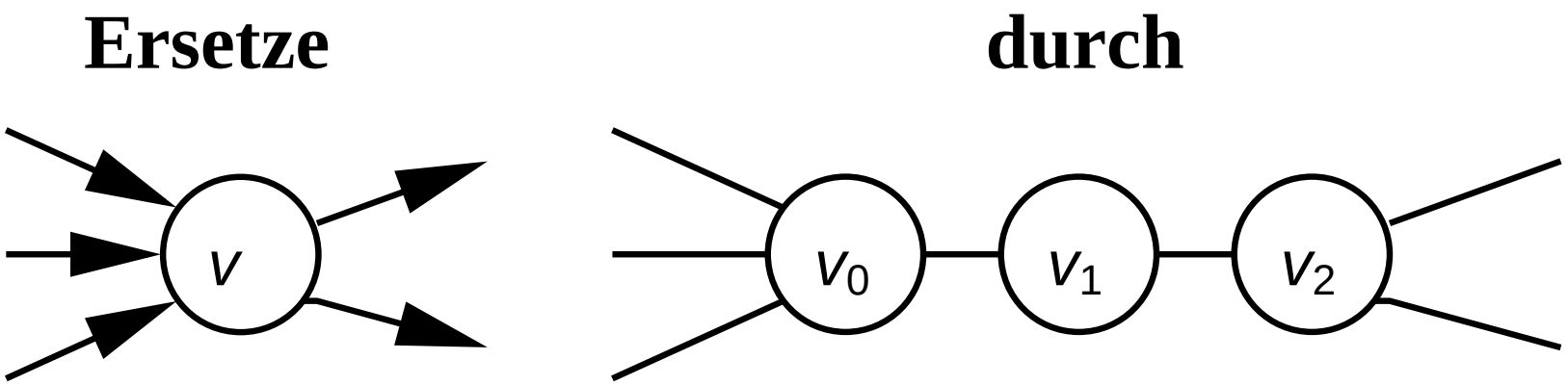
Satz

UH-KREIS ist $\mathcal{NP}$-vollständig.

Zeigen

1. UH-KREIS $\in \mathcal{NP}$ (Übung)
2. GH-KREIS $\leq_p$ UH-KREIS

Idee der Reduktion f:



Reduktion $f(G) = G'$

Algorithmus M_f

EINGABE: $G = (V, E)$ gerichteter Graph mit $V = [n], E = [m]$

1. Konstruktion der Knotenmenge V':
   - Für jeden Knoten $v \in V$ konstruiere v_0, v_1, v_2
2. Konstruktion der Kantenmenge E':
   - $E' = \{\{u_2, v_0\}, \{v_0, v_1\}, \{v_1, v_2\} \mid (u, v) \in E\}$.

AUSGABE: $G' = (V', E')$ ungerichteter Graph

Laufzeit:

- Eingabelänge $|G| = \Omega(n + m)$
- Schritt 1: $\mathcal{O}(n)$, Schritt 2: $\mathcal{O}(n + m)$
- D.h. die Gesamtlaufzeit ist linear in der Eingabelänge.

Korrektheit: $G \in$ GH-KREIS $\Leftrightarrow f(G) = G' \in$ UH-KREIS

Sei $G \in$ GH-KREIS

- Dann existiert eine Permutation $\pi : [n] \to [n]$, so dass G einen Hamiltonschen Kreis $H = (\pi(1), \pi(2), \ldots, \pi(n), \pi(1))$ enthält.
- G' enthält den Hamiltonschen Kreis $H' = (\pi(1)_0, \pi(1)_1, \pi(1)_2, \ldots, \pi(n)_0, \pi(n)_1, \pi(n)_2, \pi(1)_0)$.
- Damit ist $G' \in$ UH-KREIS

Sei $G' \in$ UH-KREIS

- G' enthält einen Hamiltonschen Kreis H'.
  - H' muss für alle $v \in V'$ die Kanten $\{v_0, v_1\}$ und $\{v_1, v_2\}$ enthalten, sonst könnte v_1 nicht in H' sein.
  - H' ist oBdA von der Form $(\pi(1)_0, \pi(1)_1, \pi(1)_2, \ldots, \pi(n)_0, \pi(n)_1, \pi(n)_2, \pi(1)_0)$.
- G besitzt Hamiltonschen Kreis $H = (\pi(1), \pi(2), \ldots, \pi(n), \pi(1))$.
- Damit ist $G \in$ GH-KREIS

Übersicht unserer $\mathcal{NP}$-vollständigen Probleme

Vorlesung:

- SAT
- 3SAT
- CLIQUE
- KNOTENÜBERDECKUNG
- SUBSETSUM
- RUCKSACK
- COVER
- GH-KREIS
- UH-KREIS

Übung:

- TEILGRAPH
- INDEPENDENT SET
- 0,1-PROGRAMMIERUNG
- LÄNGSTER PFAD
- HALF-CLIQUE

Diffie-Hellman Schlüsselaustausch (1976)

Öffentliche Parameter:

- Generator g einer multiplikativen Gruppe G mit primer Ordnung q.
- Die Beschreibungslänge von Elementen in G ist $\mathcal{O}(\log^2 q)$.
- Gruppenoperationen in G sollen Laufzeit $\mathcal{O}(\log^2 q)$ kosten.

Protokoll Diffie-Hellman Schlüsselaustausch

EINGABE: p, g

1. Alice wählt $a \in_R \mathbb{Z}_q$ und schickt g^a an Bob.
2. Bob wählt $b \in_R \mathbb{Z}_q$ und schickt g^b an Alice.
3. Alice berechnet $\left(g^b\right)^a = g^{ab}$, Bob analog $\left(g^a\right)^b = g^{ab}$.

Gemeinsamer geheimer DH-Schlüssel: g^{ab}.

Sicherheit gegenüber passive Angreifer

- Angreifer Eve für DH-Schlüsselaustausch erhält g, g^a, g^b.
- **Sicherheit:** Eve kann g^{ab} nicht von g^z, $z \in_R \mathbb{Z}_q$ unterscheiden.

Definition Decisional Diffie-Hellman (DDH)

Sei g Generator einer multiplikativen Gruppe G mit Ordnung q. Wir definieren die Sprache

$$\text{DDH} := \{(q, g, g^a, g^b, g^z) \mid g^z = g^{ab}\}.$$

Das ElGamal Kryptosystem (1984)

Algorithmus ElGamal

- **Schlüsselerzeugung**: Sei g Generator einer multiplikativen Gruppe G mit primer Ordnung q. Wähle $x \in_R \mathbb{Z}_q$. Setze $h := g^x$. Öffenlicher Schlüssel: q, g, h, geheimer Schlüssel: x,
- **Verschlüsselung:** Für Nachrichten $m \in G$ wähle $y \in_R \mathbb{Z}_q$ und berechne

$$Enc(m) = c = (c_1, c_2) = (g^y, m \cdot (h)^y).$$

- **Entschlüsselung:** Für einen Chiffretext $c = (c_1, c_2)$ berechne

$$Dec(c) = \frac{c_2}{c_1^x} = \frac{m \cdot g^{xy}}{g^{xy}} = m.$$

Laufzeit:

- Verschlüsselung: $\mathcal{O}(\log y \cdot \log^2 q) = \mathcal{O}(\log^3 q)$
- Entschlüsselung: $\mathcal{O}(\log x \cdot \log^2 q) = \mathcal{O}(\log^3 q)$