# 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 $B, k \in \mathbb{N}$.

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

## 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 $B \leftarrow t$ und $k \leftarrow t$.
2. For $i \leftarrow 1$ to $n$: Setze $w_i \leftarrow m_i$ und $p_i \leftarrow m_i$

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 $\sum_{i \in I} p_i \geq t$.
- 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 $S = \sum_{i \in I} m_i$. 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 $\ell_{jk}$.
- $F$ 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: $V_{0i} = \{x_i\} \cup \{\ell_{jk} \mid \ell_{jk} = x_i\}$ und
     $V_{1i} = \{x_i\} \cup \{\ell_{jk} \mid \ell_{jk} = \neg x_i\}$ 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 $\phi$ 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.: $(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

- $U = \{x_1, x_2, x_3, K_1, K_2, \ell_{11}, \ell_{12}, \ell_{13}, \ell_{21}, \ell_{22}, \ell_{23}\}$
- $V_{0i}$ : $T_{01} = \{x_1, \ell_{11}\}, T_{02} = \{x_2, \ell_{12}, \ell_{22}\}, T_{30} = \{x_3, \ell_{33}\}$
- $V_{1i}$ : $T_{11} = \{x_1, \ell_{21}\}, T_{21} = \{x_2\}, T_{31} = \{x_3, \ell_{13}\}$
- $K_{1k}$ : $K_{11} = \{K_1, \ell_{11}\}, K_{12} = \{K_1, \ell_{12}\}, K_{13} = \{K_1, \ell_{13}\}$
- $K_{2k}$ : $K_{21} = \{K_2, \ell_{21}\}, K_{22} = \{K_2, \ell_{22}\}, K_{23} = \{K_2, \ell_{23}\}$
- $L_{1k}$ : $L_{11} = \{\ell_{11}\}, L_{12} = \{\ell_{12}\}, L_{13} = \{\ell_{13}\}$
- $L_{2k}$ : $L_{21} = \{\ell_{21}\}, L_{22} = \{\ell_{22}\}, L_{23} = \{\ell_{23}\}$
- Erfüllende Belegung von $\phi$: $x_1 = 0, x_2 = 1, x_3 = 1$.

# 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 $\ell_{jk}$ auf wahr.
- Definiere Menge $C \subseteq F$ mittels $B$:
  - Variablen: Falls $x_i = 0$, nimm $V_{0i}$ in $C$ auf. Sonst $V_{1i}$.
  - Klauseln: Nimm Menge $K_{jk}$, die $\ell_{jk}$ 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 $V_{0i}, V_{1i}$ abgedeckt.
  - Klauseln $K_j$: Werden durch $K_{jk}$ abgedeckt.
    Die paarweisen Schnitte der Mengen $V_{0i}, V_{1i}, K_{jk}$ sind *leer*.
  - Literale $\ell'_{jk}$: Werden durch $L'_{jk}$ 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 $V_{0i}$ oder $V_{1i}$ 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 $V_{0i}, V_{1i}$ abgedeckten Literale sind auf falsch gesetzt.
  - Jede Klauselmenge $K_{jk}$ muss ein wahres Literal $\ell_{jk}$ 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

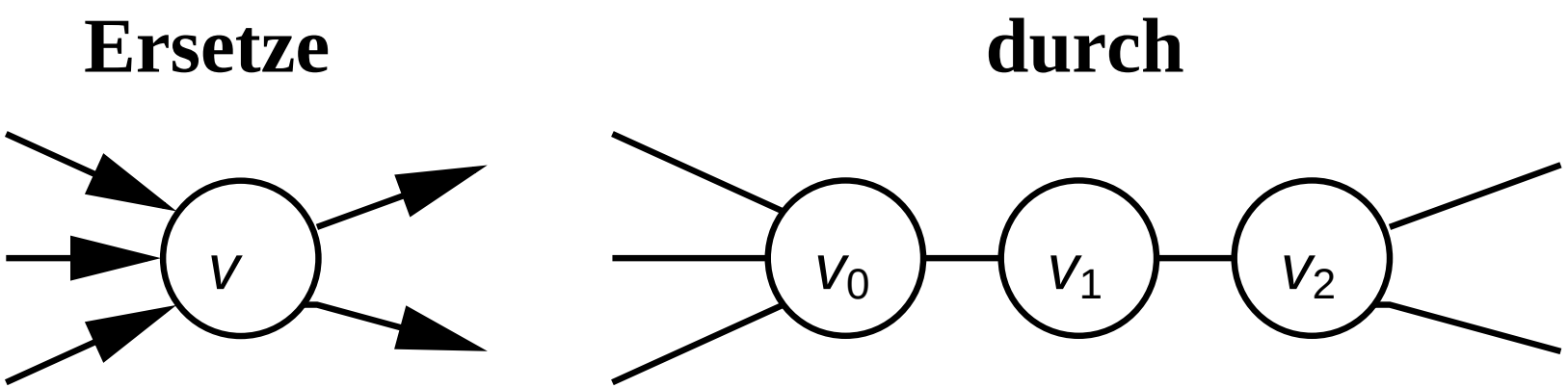
**Satz**

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 $V'$:
   - Ersetze alle $v \in V$ durch $v_0, v_1, v_2$
2. Konstruktion der Kantenmenge $E'$:
   - $E' = \{\{u_2, v_0\} \mid (u, v) \in E\} \cup \{\{v_0, v_1\}, \{v_1, v_2\} \mid v \in V\}$.

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:** Primzahl $p$, Generator $g$ von $\mathbb{Z}_p^*$

## **Protokoll** Diffie-Hellman Schlüsselaustausch

EINGABE: $p$, $g$

1. Alice wählt $\alpha \in_R \mathbb{Z}_{p-1}$ und schickt $g^\alpha \bmod p$ an Bob.
2. Bob wählt $\beta \in_R \mathbb{Z}_{p-1}$ und schickt $g^\beta \bmod p$ an Alice.
3. Alice berechnet $\left(g^\beta\right)^\alpha = g^{\alpha\beta}$, Bob analog $(g^\alpha)^\beta = g^{\alpha\beta}$.

Gemeinsamer geheimer DH-Schlüssel: $g^{\alpha\beta}$.

- Angreifer Eve erhält $g, g^\alpha, g^\beta$.
- **Sicherheit:** Eve kann $g^{\alpha\beta}$ nicht von $g^y$, $y \in_R \mathbb{Z}_{p-1}$ unterscheiden.

## **Definition** Decisional Diffie-Hellman (DDH)

Sei $p$ prim, $g$ Generator von $\mathbb{Z}_p^*$. Wir definieren die Sprache

$$\text{DDH} := \{(g^\alpha, g^\beta, g^y) \mid g^y = g^{\alpha\beta}\}.$$

# Das ElGamal Kryptosystem (1984)

**Parameter des ElGamal Kryptosystems:**

öffentlich: $p$ prim, $g$ Generator von $\mathbb{Z}_p^*$, $g^a$

geheim: $a \in \mathbb{Z}_{p-1}$

## **Algorithmus** ElGamal Ver- und Entschlüsselung

- Verschlüsselung von $m \in \mathbb{Z}_p$ unter Verwendung von $p, g, g^a$.
  - Wähle $r \in_R \mathbb{Z}_{p-1}$.
  - Berechne $Enc(m) = (\gamma, \delta) = (g^r, m \cdot (g^a)^r) \in \mathbb{Z}_p^* \times \mathbb{Z}_p$.
- Entschlüsselung von $Enc(m)$ unter Verwendung von $p, a$.
  - Berechne $Dec(Enc(m)) = \frac{\delta}{\gamma^a} = \frac{m \cdot g^{ar}}{g^{ar}} = m$.

**Laufzeit:**

- Verschlüsselung: $\mathcal{O}(\log r \cdot \log^2 p) = \mathcal{O}(\log^3 p)$
- Entschlüsselung: $\mathcal{O}(\log a \cdot \log^2 p) = \mathcal{O}(\log^3 p)$

# Sicherheit von ElGamal

**Intuitiv:** Eve soll $\delta = m \cdot g^{ab}$ nicht von $x \in_R \mathbb{Z}_p$ unterscheiden können.

## **Protokoll** Unterscheider

EINGABE: $p, g, g^a$

1. Eve wählt $m \in \mathbb{Z}_p^*$ und schickt $m$ an Alice. (Man beachte: $m \neq 0$.)
2. Alice wählt $b \in \{0, 1\}$:
   - Falls $b = 0$: Sende $Enc(m) = (g^r, m \cdot g^{ar})$ an Eve zurück.
   - Falls $b = 1$: Sende $(g^r, x) \in_R \mathbb{Z}_p^* \times \mathbb{Z}_p^*$ an Eve zurück.

Eves AUSGABE: $b' \in \{0, 1\}$

- Eve gewinnt das Spiel gdw $b' = b$.
- D.h. Eve muss $\delta$ von einer Zufallszahl $x$ unterscheiden.

## **Definition** Sprache ElGamal

Sei $p$ prim und $g$ ein Generator von $\mathbb{Z}_p^*$. Wir definieren

$$\text{ELGAMAL} := \{g^a, g^r, m, x \mid x = m \cdot g^{ar} \bmod p\}.$$

# Sicherheitsbeweis per Reduktion

**Satz** Sicherheit von ElGamal unter DDH

Das ElGamal Kryptosystem ist sicher gegen polynomielle Angreifer unter der Annahme, dass DDH nicht effizient entscheidbar ist.

**Logik des Beweises:**

- Zeigen: DDH $\leq_p$ ELGAMAL
- D.h. jeder polynomielle Algorithmus für ELGAMAL liefert einen polynomiellen Algorithmus für DDH.
- **Annahme:** Es existiert ein polynomieller Angreifer $A$, der Verschlüsselungen von Zufallszahlen unterscheiden kann.
- Dann gibt es einen Algorithmus, der in polynomieller Zeit DH-Schlüssel $g^{\alpha\beta}$ von Zufallszahlen unterscheidet.
- **Widerspruch:** Nach Annahme gibt es keinen effizienten Algorithmus zum Entscheiden von DH-Schlüsseln $g^{\alpha\beta}$.
- Daher kann es auch keinen polynomiellen Angreifer $A$ geben.

# Reduktion $f$

## Algorithmus $M_f$

EINGABE: $g^\alpha, g^\beta, g^y \in \mathbb{Z}_p^*$

1. Setze $g^a \leftarrow g^\alpha$ und $g^r \leftarrow g^\beta$.
2. Wähle $m \in_R \mathbb{Z}_p^*$.
3. Berechne $x = m \cdot g^y \bmod p$.

AUSGABE: $g^a, g^r, m, x$

**Laufzeit:**

- Eingabelänge: $\Omega(\log p)$
- Gesamtlaufzeit: $\mathcal{O}(\log^2(p))$

# Korrektheit Reduktion: $w \in \text{DDH} \leq_p f(w) \in \text{ElGamal}$

Sei $(g^\alpha, g^\beta, g^\gamma) \in \text{DDH}$.

- Dann gilt $g^\gamma = g^{\alpha\beta} = g^{ar}$.
- Damit ist $x = m \cdot g^\gamma = m \cdot g^{ar}$ korrekte Verschlüsselung von $m$.
- D.h. $(g^a, g^r, m, x) \in \text{ElGamal}$

Sei $f(g^\alpha, g^\beta, g^\gamma) = (g^a, g^r, m, x) \in \text{ElGamal}$.

- Dann ist $x = m \cdot g^\gamma$ eine korrekte Verschlüsselung von $m$.
- D.h. $d(m) = \frac{m \cdot g^\gamma}{g^{ar}} = m$ und damit $g^\gamma = g^{ar} = g^{\alpha\beta}$.
- Dann ist $(g^\alpha, g^\beta, g^\gamma) \in \text{DDH}$.

# Brechen von ElGamal ist nicht schwerer als DDH

## Satz

ElGamal $\leq_p$ DDH

**Beweis:** Wir definieren die folgende Reduktion $f$.

## Algorithmus $M_f$

EINGABE: $g^a, g^r, m, x \in \mathbb{Z}_p^*$

1. Setze $g^\alpha \leftarrow g^a$ und $g^\beta \leftarrow g^r$.
2. Berechne $g^y = \frac{x}{m}$.

AUSGABE: $g^\alpha, g^\beta, g^y$

**Laufzeit:**

- Eingabelänge: $\Omega(\log p)$
- Laufzeit: $\mathcal{O}(\log^2 p)$

# Korrektheit von $f$: $w \in$ ElGamal $\Leftrightarrow f(w) \in$ DDH

Sei $(g^a, g^r, m, x) \in$ ElGamal.

- Dann ist $x = m \cdot g^{ar}$ korrekte Verschlüsselung von $m$.
- Damit gilt $\frac{x}{m} = g^{ar} = g^{\alpha\beta} = g^y$.
- D.h. $(g^\alpha, g^\beta, g^y) \in$ DDH.

Sei $f(g^a, g^r, m, x) = (g^\alpha, g^\beta, g^y) \in$ DDH.

- Dann gilt $g^y = g^{\alpha\beta} = g^{ar}$.
- Damit folgt $x = m \cdot g^y = m \cdot g^{ar}$ ist Verschlüsselung von $m$.
- D.h. $(g^a, g^r, m, x) \in$ ElGamal.