

Theoretische Grundlagen der Informatik

Vorlesung am 06. Dezember 2016

INSTITUT FÜR THEORETISCHE INFORMATIK



Problem SUBSET SUM

Gegeben: Eine endliche Menge M , eine Gewichtsfunktion $w : M \rightarrow \mathbb{N}_0$ und $K \in \mathbb{N}_0$

Frage: Existiert eine Teilmenge $M' \subseteq M$ mit $\sum_{a \in M'} w(a) = K$?

Satz:

Problem SUBSET SUM ist \mathcal{NP} -vollständig.

NP-Vollständigkeit von SUBSET SUM

SUBSET SUM $\in \mathcal{NP}$.

- Es kann für eine gegebene Teilmenge $M' \subseteq M$ in Polynomialzeit der Wert $\sum_{a \in M'} w(a)$ ausgerechnet und mit K verglichen werden.

Beweis: NP-Vollständigkeit von SUBSET SUM

EXACT COVER α SUBSET SUM

- Sei $(X = \{0, 1, \dots, m-1\}, \mathcal{S})$ EXACT COVER-Instanz.
- Konstruiere SUBSET SUM Instanz (M, w, K)

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$p := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} p^x$$

$$K := \sum_{x=0}^{m-1} p^x$$

- Die Konstruktion benötigt nur Polynomialzeit.

Beweis: NP-Vollständigkeit von SUBSET SUM

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$p := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} p^x$$

$$K := \sum_{x=0}^{m-1} p^x$$

Veranschaulichung:

- Wir stellen die Mengenzugehörigkeiten als Zahlen zur Basis p dar.
- Kodiere $w(Y)$ für $Y \in \mathcal{S}$ als String aus Nullen und Einsen der Länge m , wobei an i -ter Stelle eine 1 steht genau dann, wenn $i \in Y$;
- entsprechend ist K ein String der Länge m aus Einsen.

Beweis: NP-Vollständigkeit von SUBSET SUM

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$\rho := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} \rho^x$$

$$K := \sum_{x=0}^{m-1} \rho^x$$

Veranschaulichung:

- Komponentenweise Addition der Strings $w(Y_1), \dots, w(Y_n)$ ergibt einen String der Länge m , an dessen i -ter Stelle steht in wievielen der $Y_j (j = 1, \dots, n)$ das Element i vorkommt.
- $\sum_{Y \in \mathcal{S}'} w(Y) = K$ bedeutet also, dass jedes $x \in X$ in genau einem $Y \in \mathcal{S}'$ vorkommt.

Beweis: NP-Vollständigkeit von SUBSET SUM

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$p := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} p^x$$

$$K := \sum_{x=0}^{m-1} p^x$$

Beispiel:

$$X = \{0, 1, 2, 3, 4, 5, 6\}$$

$$\mathcal{S} = \{Y_1 = \{0, 1, 2, 3\}, Y_2 = \{2, 5\}, Y_3 = \{3, 4, 5\}, Y_4 = \{4, 5, 6\}\}$$

$$\#0 = 1, \#1 = 1, \#2 = 2, \#3 = 2, \#4 = 2, \#5 = 3, \#6 = 1, p = 4$$

$$w(Y_1) = 0001111_4, w(Y_2) = 0100100_4$$

$$w(Y_3) = 0111000_4, w(Y_4) = 1110000_4.$$

Beweis: NP-Vollständigkeit von SUBSET SUM

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$\rho := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} \rho^x$$

$$K := \sum_{x=0}^{m-1} \rho^x$$

■ (X, \mathcal{S}) lösbar $\Rightarrow (M, w, K)$ lösbar.

Sei $\mathcal{S}' \subseteq \mathcal{S}$ exakte Überdeckung von (X, \mathcal{S}) . Dann gilt

$$\sum_{Y \in \mathcal{S}'} w(Y) = \sum_{Y \in \mathcal{S}'} \sum_{x \in Y} \rho^x = \sum_{x=0}^{m-1} \rho^x = K,$$

da jedes $x \in X$ genau einmal überdeckt wird. \mathcal{S}' erfüllt also die Bedingung für SUBSET SUM.

Beweis: NP-Vollständigkeit von SUBSET SUM

$$M := \mathcal{S}$$

$$\#x := |\{Y \in \mathcal{S} : x \in Y\}|$$

$$\rho := \max_{x \in X} \#x + 1$$

$$w(Y) := \sum_{x \in Y} \rho^x$$

$$K := \sum_{x=0}^{m-1} \rho^x$$

■ (X, \mathcal{S}) lösbar $\Leftrightarrow (M, w, K)$ lösbar.

Ist $\mathcal{S}' \subseteq M = \mathcal{S}$ eine geeignete Menge für SUBSET SUM, so gilt

$$\sum_{Y \in \mathcal{S}'} w(Y) = K = \sum_{x=0}^{m-1} \rho^x.$$

Also kommt jedes $x \in X$ in genau einem $Y \in \mathcal{S}'$ vor.

Damit ist \mathcal{S}' eine exakte Überdeckung.

Problem PARTITION

Gegeben: Eine endliche Menge M und eine Gewichtsfunktion $w : M \rightarrow \mathbb{N}_0$.

Frage: Existiert eine Teilmenge $M' \subseteq M$ mit $\sum_{a \in M'} w(a) = \sum_{a \in M \setminus M'} w(a)$?

Satz:
Problem PARTITION ist \mathcal{NP} -vollständig.

Beweis: NP-Vollständigkeit von PARTITION

PARTITION $\in \mathcal{NP}$.

- Für eine Menge M' können in Polynomialzeit die Werte $\sum_{a \in M'} w(a)$ und $\sum_{a \in M \setminus M'} w(a)$ ausgerechnet und verglichen werden.

Beweis: NP-Vollständigkeit von PARTITION

SUBSET SUM \propto PARTITION.

- Sei (M, w, K) eine SUBSET SUM-Instanz.
- Konstruiere PARTITION-Instanz (M^*, w^*)

$$N := \sum_{a \in M} w(a) + 1$$

$$M^* := M \cup \{b, c\}$$

$$w^*(a) = w(a) \quad \text{für } a \in M$$

$$w^*(b) := N - K$$

$$w^*(c) := K + 1$$

- Die Konstruktion benötigt nur Polynomialzeit.

Beweis: NP-Vollständigkeit von PARTITION

$$N := \sum_{a \in M} w(a) + 1$$

$$M^* := M \cup \{b, c\}$$

$$w^*(a) = w(a) \quad \text{für } a \in M$$

$$w^*(b) := N - K$$

$$w^*(c) := K + 1$$

- (M, w, K) Ja-Instanz genau dann, wenn (M^*, w^*) Ja-Instanz:
 $\exists M' \subseteq M^*$ mit $\sum_{a \in M'} w^*(a) = \sum_{a \in M^* \setminus M'} w^*(a) \iff \exists M'' \subseteq M$ mit $w(M'') = K$.
- Es können b und c nicht beide in M' bzw. $M^* \setminus M'$ enthalten sein.
- o.B.d.A. $b \in M'$

Beweis: NP-Vollständigkeit von PARTITION

$$N := \sum_{a \in M} w(a) + 1$$

$$M^* := M \cup \{b, c\}$$

$$w^*(a) = w(a) \quad \text{für } a \in M$$

$$w^*(b) := N - K$$

$$w^*(c) := K + 1$$

■ (M, w, K) Ja-Instanz genau dann, wenn (M^*, w^*) Ja-Instanz:

$$\exists M' \subseteq M^* \text{ mit } \sum_{a \in M'} w^*(a) = \sum_{a \in M^* \setminus M'} w^*(a) \iff \exists M'' \subseteq M \text{ mit } w(M'') = K.$$

\Rightarrow

■ Sei M' , so dass $\sum_{a \in M'} w^*(a) = \sum_{a \in M^* \setminus M'} w^*(a)$.

■ Dann gilt $w(M') = N$, da $w(M^*) = 2N$.

■ Damit erfüllt $M'' := M' \setminus \{b\}$ die Bedingung für SUBSET SUM.

Beweis: NP-Vollständigkeit von PARTITION

$$N := \sum_{a \in M} w(a) + 1$$

$$M^* := M \cup \{b, c\}$$

$$w^*(a) = w(a) \quad \text{für } a \in M$$

$$w^*(b) := N - K$$

$$w^*(c) := K + 1$$

■ (M, w, K) Ja-Instanz genau dann, wenn (M^*, w^*) Ja-Instanz:

$$\exists M' \subseteq M^* \text{ mit } \sum_{a \in M'} w^*(a) = \sum_{a \in M^* \setminus M'} w^*(a) \iff \exists M'' \subseteq M \text{ mit } w(M'') = K.$$

⇐

■ Sei M'' , so dass $w(M'') = K$.

■ Dann erfüllt $M' := M'' \cup \{b\}$ die Bedingung für PARTITION.

Problem KNAPSACK

Gegeben: Eine endliche Menge M ,
eine Gewichtsfunktion $w : M \rightarrow \mathbb{N}_0$,
eine Kostenfunktion $c : M \rightarrow \mathbb{N}_0$
 $W, C \in \mathbb{N}_0$.

Frage: Existiert eine Teilmenge $M' \subseteq M$ mit $\sum_{a \in M'} w(a) \leq W$
und $\sum_{a \in M'} c(a) \geq C$?

Satz:
Problem KNAPSACK ist \mathcal{NP} -vollständig.

Beweis: NP-Vollständigkeit von KNAPSACK

KNAPSACK $\in \mathcal{NP}$.

- Für eine Menge M' kann in Polynomialzeit überprüft werden, ob
 - $\sum_{a \in M'} w(a) \leq W$ und
 - $\sum_{a \in M'} c(a) \geq C$
- gilt.

Beweis: NP-Vollständigkeit von KNAPSACK

PARTITION \propto KNAPSACK.

- Sei (M, w) eine PARTITION-Instanz.
- Konstruiere KNAPSACK-Instanz (M, w', c, W, C)

$$\begin{aligned}w' &:= 2w \\c &:= 2w \\W = C &:= \sum_{a \in M} w(a)\end{aligned}$$

- Die Konstruktion benötigt nur Polynomialzeit.
- Es ist (M, w) genau dann eine Ja-Instanz, wenn (M, w', c, W, C) eine Ja-Instanz ist (ohne Beweis).

Auswirkung auf die Frage $\mathcal{P} = \mathcal{NP}$

- Wir haben gesehen, dass es für je zwei \mathcal{NP} -vollständige Probleme eine polynomiale Transformation von einem zum anderen Problem gibt.
- Deshalb sind alle \mathcal{NP} -vollständigen Probleme im wesentlichen gleich. schwer
- Dies hat Auswirkungen auf die Frage, ob $\mathcal{P} = \mathcal{NP}$ ist.

Satz:

Sei L \mathcal{NP} -vollständig, dann gilt:

- $L \in \mathcal{P} \implies \mathcal{P} = \mathcal{NP}$
- $L \notin \mathcal{P} \implies$ für jede \mathcal{NP} -vollständigen Sprache L' gilt $L' \notin \mathcal{P}$

Auswirkung auf die Frage $\mathcal{P} = \mathcal{NP}$

Beweis: L ist \mathcal{NP} -vollständig, $L \in \mathcal{P} \implies \mathcal{P} = \mathcal{NP}$

- Sei $L \in \mathcal{P}$ und L \mathcal{NP} -vollständig.
- Dann existiert eine polynomiale deterministische TM M für L .
- Sei $L' \in \mathcal{NP}$:
- Es gibt polynomiale Transformation $L' \propto L$.
- Hintereinanderausführung von $L' \propto L$ und M liefert deterministische polynomielle TM-Berechnung für L' .
- Damit ist $L' \in \mathcal{P}$.

Auswirkung auf die Frage $\mathcal{P}=\mathcal{NP}$

Beweis: L ist \mathcal{NP} -vollständig, $L \notin \mathcal{P}$

\implies für jede \mathcal{NP} -vollständigen Sprache L' gilt $L' \notin \mathcal{P}$

- Sei $L \notin \mathcal{P}$ und L \mathcal{NP} -vollständig.
- Angenommen für eine \mathcal{NP} -vollständige Sprache L' gilt: $L' \in \mathcal{P}$
- Dann folgt aus Teil 1 des Satzes $\mathcal{P} = \mathcal{NP}$.
- Dies ist aber ein Widerspruch zur Voraussetzung $L \notin \mathcal{P}$.

- Die Klasse \mathcal{P} ist die Klasse aller Entscheidungsprobleme/Sprachen die mit einer **deterministischen** Turingmaschine in polynomieller Zeit gelöst werden können.
- Die Klasse \mathcal{NP} ist die Klasse aller Entscheidungsprobleme/Sprachen die mit einer **nicht-deterministischen** Turingmaschine in polynomieller Zeit gelöst werden können.
- Informell ausgedrückt: Π gehört zu \mathcal{NP} , falls Π folgende Eigenschaft hat: Ist die Antwort bei Eingabe eines Beispiels I von Π Ja, dann kann die Korrektheit eines Beweises (Zeugen) dafür in polynomialer Zeit überprüft werden.

- Eine **polynomiale Transformation** einer Sprache $L_1 \subseteq \Sigma_1^*$ in eine Sprache $L_2 \subseteq \Sigma_2^*$ ist eine Funktion $f: \Sigma_1^* \rightarrow \Sigma_2^*$ mit den Eigenschaften:
 - es existiert eine polynomiale deterministische Turing-Maschine, die f berechnet;
 - für alle $x \in \Sigma_1^*$ gilt: $x \in L_1 \Leftrightarrow f(x) \in L_2$.
- Eine Sprache L heißt **\mathcal{NP} -vollständig**, falls gilt:
 - $L \in \mathcal{NP}$ und
 - für alle $L' \in \mathcal{NP}$ gilt $L' \leq L$ (\mathcal{NP} -Schwere).
- **Bedeutung:** Unter der Annahme $\mathcal{P} \neq \mathcal{NP}$ gibt es kein polynomielles Lösungsverfahren für ein \mathcal{NP} -vollständiges Problem.

- Mit dem Satz von Cook haben wir direkt gezeigt, dass Problem SAT \mathcal{NP} -schwer ist.
- Bei allen anderen Problemen haben wir polynomielle Transformationen (Reduktionen) benutzt um die \mathcal{NP} -Schwere nachzuweisen:

$\text{SAT} \propto 3\text{SAT} \propto 3\text{COLOR} \propto \text{EXACT COVER} \propto \text{SUBSET SUM} \propto$
 $\text{PARTITION} \propto \text{KNAPSACK}$

- Die Klasse \mathcal{NPC} (\mathcal{NP} -complete) sei die Klasse der \mathcal{NP} -vollständigen Sprachen/Probleme.
- Die Klasse \mathcal{NPI} (\mathcal{NP} -intermediate) ist definiert durch $\mathcal{NPI} := \mathcal{NP} \setminus (\mathcal{P} \cup \mathcal{NPC})$.

Klasse der Komplementsprachen

- Die Klasse $\mathbf{co} - \mathcal{P}$ ist die Klasse aller Sprachen $\Sigma^* \setminus L$ für $L \subseteq \Sigma^*$ und $L \in \mathcal{P}$.
- Die Klasse $\mathbf{co} - \mathcal{NP}$ ist die Klasse aller Sprachen $\Sigma^* \setminus L$ für $L \subseteq \Sigma^*$ und $L \in \mathcal{NP}$.

Die Klassen \mathcal{NPI} , co-P und co-NP

- Die Klasse \mathcal{NPC} (\mathcal{NP} -complete) sei die Klasse der \mathcal{NP} -vollständigen Sprachen/Probleme.
- Die Klasse \mathcal{NPI} (\mathcal{NP} -intermediate) ist definiert durch $\mathcal{NPI} := \mathcal{NP} \setminus (\mathcal{P} \cup \mathcal{NPC})$.

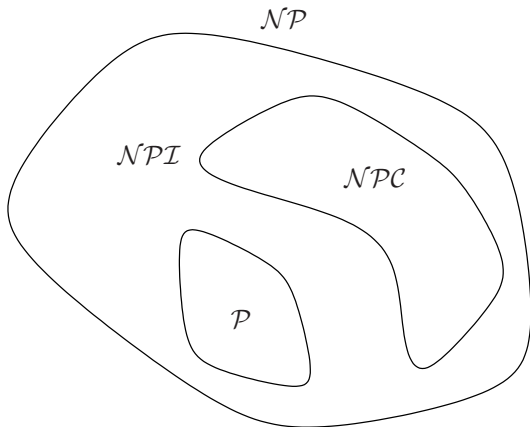
Klasse der Komplementsprachen

- Die Klasse co-P ist die Klasse aller Sprachen $\Sigma^* \setminus L$ für $L \subseteq \Sigma^*$ und $L \in \mathcal{P}$.
- Die Klasse co-NP ist die Klasse aller Sprachen $\Sigma^* \setminus L$ für $L \subseteq \Sigma^*$ und $L \in \mathcal{NP}$.

Satz (Ladner (1975)):

Falls $\mathcal{P} \neq \mathcal{NP}$, so folgt $\mathcal{NPI} \neq \emptyset$.

Vermutete Situation



Offensichtlich: $\mathcal{P} = \text{co} - \mathcal{P}$.

Frage: Gilt auch $\mathcal{NP} = \text{co} - \mathcal{NP}$?

- Natürlich folgt aus $\mathcal{NP} \neq \text{co} - \mathcal{NP}$, dass $\mathcal{P} \neq \mathcal{NP}$ gilt.
- Aber was folgt aus $\mathcal{NP} = \text{co} - \mathcal{NP}$?
- Vermutlich ist $\mathcal{NP} \neq \text{co} - \mathcal{NP}$
(Verschärfung der $\mathcal{P} \neq \mathcal{NP}$ -Vermutung).

Problem co-TSP

Gegeben: Graph $G = (V, E)$, $c: E \rightarrow \mathbb{Z}^+$ und ein Parameter K .

Aufgabe: Gibt es *keine* Tour der Länge $\leq K$?

- **Bemerkung:** Für ein vernünftiges Kodierungsschema von TSP ist es leicht nachzuweisen, ob ein gegebener String eine gültige TSP-Instanz repräsentiert.
- co-TSP in co- \mathcal{NP} , denn TSP in \mathcal{NP} .
- Frage: Ist co-TSP in \mathcal{NP} ?
- Vermutung: Nein.

Satz (Lemma):

Falls L \mathcal{NP} -vollständig ist und $L \in \text{co} - \mathcal{NP}$, so ist $\mathcal{NP} = \text{co} - \mathcal{NP}$.

Satz (Lemma):

Falls L \mathcal{NP} -vollständig ist und $L \in \text{co} - \mathcal{NP}$, so ist $\mathcal{NP} = \text{co} - \mathcal{NP}$.

Beweis:

- Sei $L \in \text{co} - \mathcal{NP}$ und L \mathcal{NP} -vollständig.
- Dann existiert eine polynomiale nichtdet. Berechnung für L^c .
- Für alle $L' \in \mathcal{NP}$ gilt: $L' \leq L$.
- Also existiert eine det. poly. Transformation $L'^c \leq L^c$.
- Deshalb existiert eine poly. nichtdet. Berechnung für L'^c .
- Also $L' \in \text{co} - \mathcal{NP}$.

Bemerkung

- Mit der Vermutung $\mathcal{NP} \neq \text{co} - \mathcal{NP}$ folgt auch $\mathcal{NPC} \cap \text{co} - \mathcal{NP} = \emptyset$.
- Unter dieser Annahme: Wenn ein Problem in \mathcal{NP} und $\text{co} - \mathcal{NP}$ ist, aber nicht in \mathcal{P} , so ist es in \mathcal{NPI} .

Problem Subgraphisomorphie

Gegeben: Graphen $G = (V, E)$ und $H = (V', E')$ mit $|V'| < |V|$

Frage: Gibt es eine Menge $U \subseteq V$ mit $|U| = |V'|$ und eine bijektive Abbildung $\text{Iso}: V' \rightarrow U$, so dass für alle $x, y \in V'$ gilt:
 $\{x, y\} \in E' \iff \{\text{Iso}(x), \text{Iso}(y)\} \in E$

Frage anschaulich: Ist H isomorph zu einem Subgraphen von G ?

Problem Subgraphisomorphie

Gegeben: Graphen $G = (V, E)$ und $H = (V', E')$ mit $|V'| < |V|$

Frage: Gibt es eine Menge $U \subseteq V$ mit $|U| = |V'|$ und eine bijektive Abbildung $\text{Iso}: V' \rightarrow U$, so dass für alle $x, y \in V'$ gilt:
 $\{x, y\} \in E' \iff \{\text{Iso}(x), \text{Iso}(y)\} \in E$

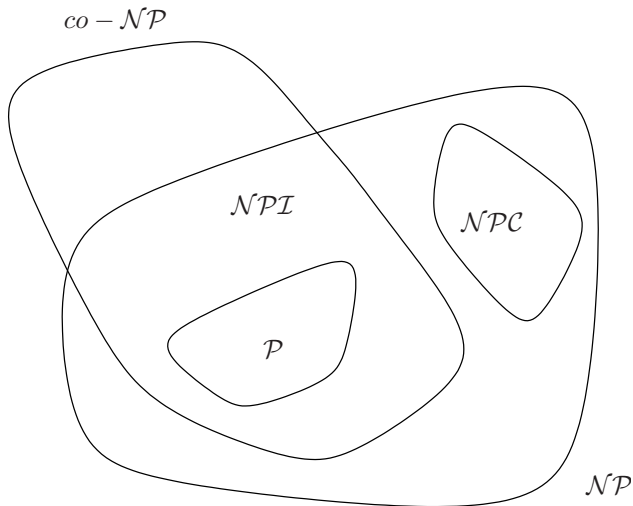
Problem Subgraphisomorphie ist \mathcal{NP} -vollständig (ohne Beweis).

Problem Graphisomorphie

Gegeben: Graphen $G = (V, E)$ und $H = (V', E')$ mit $|V| = |V'|$.

Frage: Existiert eine bijektive Abbildung $\text{Iso}: V' \rightarrow V$ mit
 $\{x, y\} \in E' \iff \{\text{Iso}(x), \text{Iso}(y)\} \in E$?

- Frage anschaulich: Sind G und H isomorph?
- Graphisomorphie ist ein Kandidat für ein Problem aus \mathcal{NP}^I .
- Graphisomorphie liegt in \mathcal{NP} und $\text{co-}\mathcal{NP}$.



Ein **Suchproblem** Π wird beschrieben durch

- die Menge der Problembeispiele / Instanzen D_{Π} und
- für $I \in D_{\Pi}$ die Menge $S_{\Pi}(I)$ *aller* Lösungen von I .

Die **Lösung** eines Suchproblems für eine Instanz D_{Π} ist

- ein beliebiges Element aus $S_{\Pi}(I)$ falls $S_{\Pi}(I) \neq \emptyset$
- \emptyset sonst

TSP-Suchproblem (Variante 1)

Gegeben: Graph $G = (V, E)$ vollständig und gewichtet mit Gewichtsfunktion $c: E \rightarrow \mathbb{Q}$.

Aufgabe: Gib eine optimale Tour zu G bezüglich c an.

- Bemerkung: $S_{\Pi}(G)$ ist die Menge aller optimalen Touren zu G .

TSP-Suchproblem (Variante 2)

Gegeben: Graph $G = (V, E)$ vollständig und gewichtet mit Gewichtsfunktion $c: E \rightarrow \mathbb{Q}$, Parameter $k \in \mathbb{Q}$.

Aufgabe: Gib eine Tour zu G bezüglich c mit Maximallänge k an, falls eine existiert.

Beispiel: Hamilton–Kreis Suchproblem

Gegeben ist ein Graph $G = (V, E)$.

Ein Hamilton–Kreis in G ist eine Permutation π auf V , so dass

$$\{\pi(n), \pi(1)\} \in E \text{ und } \{\pi(i), \pi(i+1)\} \in E \text{ für } 1 \leq i \leq n-1 \text{ ist.}$$

Hamilton–Kreis Suchproblem

Gegeben: Ein ungerichteter, ungewichteter Graph $G = (V, E)$.

Aufgabe: Gib einen Hamilton–Kreis in G an, falls einer existiert.

- Bemerkung: $S_{\Pi}(G)$ ist die Menge aller Hamilton-Kreise in G .

Ein **Aufzählungsproblem** Π ist gegeben durch

- die Menge der Problembeispiele D_{Π} und
- für $I \in D_{\Pi}$ die Menge $S_{\Pi}(I)$ aller Lösungen von I .

Die **Lösung** der Instanz I eines Aufzählungsproblem Π besteht in der Angabe der Kardinalität von $S_{\Pi}(I)$, d.h. von $|S_{\Pi}(I)|$.

Hamilton–Kreis Aufzählungsproblem

Gegeben: Ein ungerichteter, ungewichteter Graph $G = (V, E)$.

Aufgabe: Wieviele Hamilton–Kreise gibt es in G ?

Zu einem Suchproblem Π sei R_Π folgende Relation:

$$R_\Pi := \{(x, s) \mid x \in D_\Pi, s \in S_\Pi(x)\}$$

Eine Funktion $f: \Sigma^* \rightarrow \Sigma^*$ **realisiert** eine Relation R , wenn für alle $x \in \Sigma^*$ gilt:

$$f(x) = \begin{cases} \varepsilon & \nexists y \in \Sigma^* \setminus \{\varepsilon\} : (x, y) \in R \\ y & \text{sonst, mit beliebigem } y : (x, y) \in R \end{cases}$$

Ein Algorithmus **löst** das durch R_Π beschriebene Suchproblem Π , wenn er eine Funktion berechnet, die R_Π realisiert.

Eine **Orakel-Turing-Maschine** zum Orakel $G: \Sigma^* \rightarrow \Sigma^*$ ist eine deterministische Turing-Maschine mit

- einem ausgezeichnetem **Orakelband** und
- zwei zusätzlichen Zuständen q_f und q_a .

Dabei ist

- q_f der **Fragezustand** und
- q_a der **Antwortzustand**

des Orakels.

- Die Arbeitsweise ist in allen Zuständen $q \neq q_f$ wie bei der normalen Turing-Maschine.

Wenn der

- Zustand q_f angenommen wird,
- Kopf sich auf Position i des Orakelbandes befindet
- Inhalt des Orakelbandes auf Position $1, \dots, i$ das Wort $y = y_1 \dots y_i$ ist,

dann verhält sich die Orakel-TM wie folgt:

- falls $y \notin \Sigma^*$: Fehlermeldung und die Orakel-TM hält.
- In einem Schritt wird y auf dem Orakelband gelöscht,
- $G(y)$ wird auf Positionen $1, \dots, |G(y)|$ des Orakelbandes geschrieben
- Der Kopf des Orakelbandes springt auf Position 1
- Folgezustand ist q_a .

Bemerkung

- Orakel-TM und Nichtdeterministische TM sind verschiedene Konzepte.

Turing-Reduktion

Seien R, R' Relationen über Σ^* . Eine **Turing-Reduktion** α_T von R auf R' ($R \alpha_T R'$), ist eine Orakel-Turing-Maschine \mathcal{M} ,

- deren Orakel die Relation R' realisiert
- die selbst in polynomialer Zeit die Funktion f berechnet, die R realisiert.

Bemerkung:

- Falls R' in polynomialer Zeit realisierbar ist und $R \alpha_T R'$, so ist auch R in polynomialer Zeit realisierbar.
- Falls $R \alpha_T R'$ und $R' \alpha_T R''$ so auch $R \alpha_T R''$.

Ein Suchproblem Π heißt \mathcal{NP} -schwer, falls es eine \mathcal{NP} -vollständige Sprache L gibt mit $L \leq_T \Pi$.

Bemerkung

- Ein Problem das \mathcal{NP} -schwer ist, muss nicht notwendigerweise in \mathcal{NP} sein.

TSP-Suchproblem (Variante 1)

Gegeben: Graph $G = (V, E)$ vollständig und gewichtet mit Gewichtsfunktion $c: E \rightarrow \mathbb{Q}$.

Aufgabe: Gib eine optimale Tour zu G bezüglich c an.

TSP-Entscheidungsproblem

Gegeben: Graph $G = (V, E)$ vollständig und gewichtet mit Gewichtsfunktion $c: E \rightarrow \mathbb{Q}$, Parameter $k \in \mathbb{Q}$.

Aufgabe: Gibt es eine Tour der Länge höchstens k ?

Satz:

Das TSP-Suchproblem ist NP-schwer.

- Bezeichne TSP_E das Entscheidungsproblem.
- Bezeichne TSP_S das Suchproblem.

Die zu TSP_E bzw. TSP_S gehörenden Relationen R_E und R_S sind gegeben durch

$$R_E := \{(x, J) \mid x \in J_{TSP_E}\},$$
$$R_S := \{(x, y) \mid x \in D_{TSP_O}, y \in S_{TSP_O}(x)\}.$$

$$R_E := \{(x, J) \mid x \in J_{TSP_E}\},$$
$$R_S := \{(x, y) \mid x \in D_{TSP_O}, y \in S_{TSP_O}(x)\}.$$

Wir zeigen $R_E \propto_T R_S$:

Dazu geben wir eine OTM (Orakel-Turing-Maschine) mit Orakel $\Omega : \Sigma^* \rightarrow \Sigma^*$ an. Ω realisiert R_S .

Die OTM arbeitet wie folgt für eine Eingabe w :

- Schreibe die Eingabe auf das Orakelband und gehe in Zustand q_f .
- Weise das Orakel an, in einem Schritt $\Omega(w)$ auf das Orakelband zu schreiben und anschließend in den Zustand q_a zu wechseln.
- Prüfe, ob $\Omega(w)$ eine Tour der Länge $\leq k$ kodiert. Falls ja, lösche das Band und schreibe J , andernfalls lösche das Band.

Die gegebene OTM realisiert R_E und hat polynomial beschränkte Laufzeit.