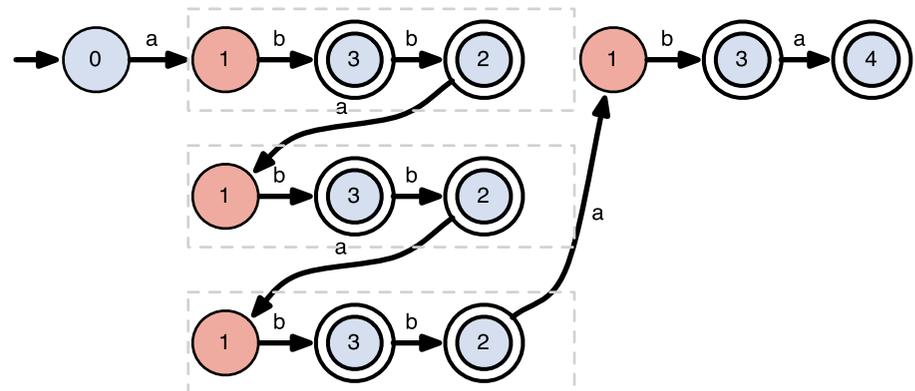


Berechenbarkeitstheorie

21. Vorlesung



Dr. Franziska Jahnke

Institut für Mathematische Logik und Grundlagenforschung

WWU Münster

Die Klassen P und NP

Definition

Die Komplexitätsklasse P bezeichnet alle Probleme, die in Polynomialzeit von einer deterministischen TM entschieden werden können. Das heißt:

$$P := \bigcup_{k=1}^{\infty} \text{TIME}(n^k)$$

- **Def.:** Ein **Verifizierer für eine Sprache L** ist eine TM für die gilt:
 $L = \{w \mid \exists z \in \Sigma^* : \langle w, z \rangle \text{ wird von } V \text{ akzeptiert}\}$
 z heißt **Zeuge** oder **Zertifikat** Laufzeit von V wird bzgl. $\langle w, z \rangle$ gemessen
- **Def.:** Ein Verifizierer V für L heißt **polynomiell**, gdw.
 - (1) $\exists k$, sodass $L = \{w \mid \exists z \in \Sigma^* : |z| \leq |w|^k \text{ und } \langle w, z \rangle \in L(V)\}$
 - (2) V hat polynomielle Laufzeit

Definition

Die Komplexitätsklasse NP umfasst alle Sprachen, für die es einen polynomiellen Verifizierer gibt.

SAT (Erfüllbarkeit)

Eingabe: Boolesche Formel ϕ in den Variablen x_i

Frage: Gibt es eine ϕ erfüllende Belegung der Variablen x_i ?

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

→ erfüllbar mit z.B. $x_1 = 1, x_2 = 1$ und $x_3 = 1$

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

→ nicht erfüllbar (Wahrheitstabelle ausfüllen)

- Zeuge ist der Vektor der Variablenbelegung

Verifizierer zu SAT

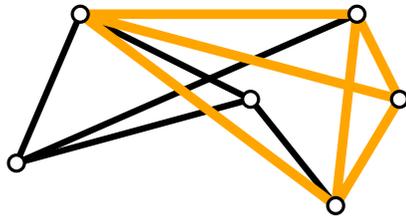
1. Ersetze alle Variablen durch ihre Belegung
2. Suche eine atomare Verbindung mit \wedge, \vee, \neg und ersetze sie (zum Beispiel ersetze $1 \vee 0$ durch 1)
3. Wiederhole bis Ausdruck ausgewertet und akzeptiere, wenn Ergebnis 1

CLIQUE

Eingabe: Graph G und natürliche Zahl k

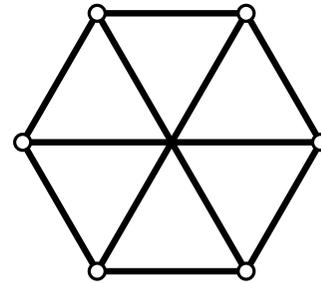
Frage: Hat G eine k -Clique (d.h., K_k als Teilgraph)?

Bsp.



$k = 4$

Graph besitzt 4-Clique



$k = 3$

Graph besitzt keine 3-Clique

- Zeuge ist Auswahl der Knoten der Clique

Verifizierer zu CLIQUE

1. Prüfe für jedes Paar von ausgewählten Knoten (des Zeugen), ob zwischen diesen Knoten eine Kante existiert
2. Ist jedes solche Paar durch eine Kante verbunden, akzeptiere

SUBSET-SUM

Eingabe: Multimenge $S = \{X_1, X_2, \dots, X_n\}$, $X_i \in \mathbf{N}$ und $B \in \mathbf{N}$

Frage: Existiert $S' \subseteq S$ mit $\sum_{X \in S'} X = B$

Bsp.

$S = \{2, 3, 4, 8, 19\}$ $B = 25$ $2 + 4 + 19 = 25$, also ja

$S = \{1, 1, 4, 8, 19\}$ $B = 30$ nicht möglich

- Zeuge ist Auswahl S'

Verifizierer zu SUBSET-SUM

1. Prüfe ob $S' \subseteq S$
2. Prüfe, ob $\sum_{X \in S'} X = B$
3. Wenn beides erfolgreich, akzeptiere

Satz 38

$$\text{NP} \subseteq \text{EXPTIME} = \bigcup_{k=1}^{\infty} \text{TIME} \left(2^{(n^k)} \right)$$

Beweis

- wir müssen zeigen, dass jede Sprache aus NP in $\text{TIME}(2^{(n^k)})$, für ein $k \in \mathbf{N}$
- sei $L \in \text{NP}$ mit polynomiellen Verifizierer V
- wir konstruieren Entscheider M für L wie folgt

$M(w)$

1. Zähle alle möglichen Zeugen auf (d.h. alle Worte aus Σ^*)
 \rightarrow für jeden Zeugen z simuliere $V(\langle w, z \rangle)$
2. akzeptiere, sobald V akzeptiert, ansonsten prüfe nächstes z

- es gibt $2^{O(p(n))}$ Zeugen, wobei p ein Polynom
- pro Zeugen z benötigt Simulation $p'(|\langle w, z \rangle|)$ Zeit, wobei p' Polynom
- L ist aus $\text{TIME}(2^{O(p(n))} p'(n + p(n))) \subset \text{EXPTIME}$



NP-Schwerheit

Definition

Seien $L_1 \subseteq \Sigma^*$ und $L_2 \subseteq \Sigma'^*$ Sprachen. Die Funktion f heißt **polyzeit Reduktion von L_1 auf L_2** , gdw.

1. f ist berechenbar und total,
2. $\forall w \in \Sigma^* : w \in L_1 \iff f(w) \in L_2$,
3. f wird durch eine TM berechnet deren Laufzeit ein Polynom in der Eingabelänge ist

Definition

Eine Sprache L_1 ist **polyzeit reduzierbar** auf L_2 , gdw. es eine polyzeit Reduktion von L_1 auf L_2 gibt.

Schreibweise: $L_1 \leq_p L_2$

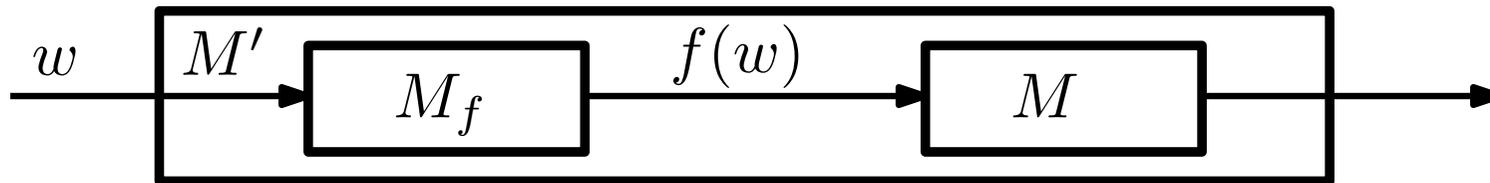
- modifizierter Reduktionsbegriff erlaubt es uns Probleme zu "übersetzen" unter Beibehaltung von polynomiellen Laufzeiten ihrer Entscheider

Satz 39

Wenn $B \in P$ und $A \leq_p B$, dann auch $A \in P$.

Beweis

- Sei M der polynomieller Entscheider für B und f eine polyzeit Reduktion von A auf B .
- Sei M_f TM, die f in polyzeit berechnet.
- Konstruiere TM M' , als Hintereinanderschaltung von M und M_f



$$M'(w) \text{ akz.} \iff M(f(w)) \text{ akz.} \iff f(w) \in B \iff w \in A$$

- Laufzeit von $M'(w)$ entspricht $t_f(|w|) + t_M(|f(w)|)$

\swarrow Laufzeit M_f \swarrow Laufzeit M
- $|f(w)|$ ist beschränkt durch ein Polynom p und t_M durch ein Polynom p'
 $\rightarrow t_M(|f(w)|) \leq \underline{p'(p(n))}$ \blacktriangleleft Polynom
- Laufzeit von M ist beschränkt durch Polynom $\rightarrow A \in P$ □

Definition

Eine Sprache L heißt **NP-vollständig**, gdw.

1. $L \in \text{NP}$,
2. $\forall L' \in \text{NP}$ gilt $L' \leq_p L$.

- NP-vollständige Sprachen kann man als die schwersten Sprachen in der Klasse NP verstehen

$$\text{NPC} := \{L \mid L \text{ ist NP-vollständig}\}$$

- Gilt in der obigen Definition nur der Punkt 2., heißt L **NP-schwer**.

Satz 40

Wenn $B \in P$ und $B \in \text{NPC}$, dann gilt $P = \text{NP}$.

Beweis

- Für jedes $A \in \text{NP}$ gilt nach Definition $A \leq_p B$
- Aus Satz 39 folgt dann aber, dass $A \in P$
- Also gilt $\text{NP} \subseteq P$, und $P \subseteq \text{NP}$ wurde bereits gezeigt $\longrightarrow P = \text{NP}$



Satz von Cook-Levin

SAT ist NP-vollständig.

Beweisidee

- Wir haben bereits $\text{SAT} \in \text{NP}$ gezeigt, wir müssen noch zeigen:
 $\forall L \in \text{NP}: L \leq_p \text{SAT}$
- Wir betrachten ein $L \in \text{NP}$ (fest für den Rest des Beweises)
- Sei N eine nicht-deterministische (polynomialzeit) TM die L erkennt
- gesucht ist polyzeit Reduktion f_N von L auf SAT

$$w \in \Sigma^* \xrightarrow{f_N} \langle \phi \rangle$$

- Die Formel ϕ ist definiert in Abhängigkeit von N und w .
- **Hauptidee:** Die Variablenbelegung (von ϕ) kodiert einen Lauf der NTM N , hierbei stellt die Erfüllbarkeit von ϕ sicher, dass ein korrekter akzeptierender Lauf von $N(w)$ kodiert wurde.

Es existiert erfüllende
Variablenbelegung für ϕ



Es existiert ein akzeptierender
Lauf für $N(w)$

Beweis

- N ist polyzeit Halbband NTM für L mit Laufzeit $t_N(n) \leq n^k - 3$
- Wir notieren den Lauf (Pfad im Berechnungsbaum) von $N(w)$ als

Tableau

Tableau:

- $n^k \times n^k$ Tabelle
- Zellen der 1. und letzte Spalte enthalten #
- 1. Zeile enthält # Startkonfiguration + Blanks + #
- Zeile = Konfiguration des kodierten Laufes
- i te Zeile ist eine Folgekonfiguration der $(i - 1)$ en Zeile
- tritt eine akzeptierende Konfiguration auf, enthalten alle folgenden Zeilen eine Kopie dieser Zeile

Bsp.

| | | | | | | | | | | | |
|---|-------|-------|---|---|---|-------|---|---|---|---|---|
| # | q_0 | a | b | b | b | □ | □ | □ | □ | □ | # |
| # | b | q_3 | b | b | b | □ | □ | □ | □ | □ | # |
| | | | | | | | | | | | |
| | | | | | | | | | | | |
| # | a | a | b | b | b | q_A | a | □ | □ | □ | # |
| # | a | a | b | b | b | q_A | a | □ | □ | □ | # |

↘ $(q_3, b, R) \in \delta(q_0, a)$

- Tableau existiert mit $q_A \Leftrightarrow$ es gibt für $N(w)$ akz. Lauf $\Leftrightarrow w \in L$
- Variablen von ϕ kodieren die Zellenbelegung des Tableaus

$$x_{i,j,s} = 1 \iff \text{Tabellenzelle } (i, j) \text{ enthält Symbol } s$$

- $s \in C := Q \cup \Gamma \cup \{\#, \square\}$, C hängt von N ab, aber nicht von w
- Formel ϕ besteht aus 4 Teilen:

$$\phi = \phi_{\text{zelle}} \wedge \phi_{\text{start}} \wedge \phi_{\text{akz}} \wedge \phi_{\text{trans}}$$

- ϕ_{zelle} sichert die korrekte Verwendung der Variablen x für das Tableau
- ϕ_{start} sichert dass erste Zeile Startkonfiguration enthält
- ϕ_{akz} sichert dass q_A im Tableau auftritt
- ϕ_{trans} sichert Zeile i eine Nachfolgerkonfiguration der Zeile $(i - 1)$ ist



Alles zusammen erzwingt, dass die Variablen ein (korrektes) Tableau mit q_A beschreiben