Berechenbarkeit und Komplexität: Polynomielle Reduktion / NP-Vollständigkeit / Satz von Cook und Levin

Prof. Dr. Berthold Vöcking
Lehrstuhl Informatik 1
Algorithmen und Komplexität

11. Januar 2008

Wiederholung

Beispiele für Probleme in NP:

- CLIQUE
- KP-E
- BPP-E
- TSP-E
- Graphzusammenhang

Bekannt: $P \subseteq NP \subseteq EXPTIME$

Hypothese: $P \neq NP$

Polynomielle Reduktion

Definition (Polynomielle Reduktion)

L_1 und L_2 seien zwei Sprachen über Σ_1 bzw. Σ_2. L_1 ist polynomiell reduzierbar *auf L_2, wenn es eine Reduktion von L_1 nach L_2 gibt, die in polynomieller Zeit berechenbar ist. Wir schreiben $L_1 \leq_p L_2$.*

D.h. $L_1 \leq_p L_2$, genau dann, wenn es eine Funktion $f : \Sigma_1^* \to \Sigma_2^*$ mit folgenden Eigenschaften gibt:

- f ist in polynomieller Zeit berechenbar
- $\forall x \in \Sigma_1^* : x \in L_1 \Leftrightarrow f(x) \in L_2$

Polynomielle Reduktion

Lemma

$L_1 \leq_p L_2$, $L_2 \in P \Rightarrow L_1 \in P$.

Beweis: Die Reduktion f habe die polyn. Laufzeitschranke $p(\cdot)$. Sei B ein Algorithmus für L_2 mit polyn. Laufzeitschranke $q(\cdot)$.

Algorithmus A für L_1:

1. Berechne $f(x)$.
2. Starte Algorithmus B für L_2 auf $f(x)$.

Schritt 1 hat Laufzeit höchstens $p(|x|)$. Schritt 2 hat Laufzeit höchstens $q(|f(x)|) \leq q(p(|x|) + |x|)$. □

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

Die eigentliche Stärke des Reduktionsprinzips ist es, dass man Probleme unterschiedlichster Art aufeinander reduzieren kann.

Problem (Knotenfärbung – COLORING)

Eingabe: Graph $G = (V, E)$, Zahl $k \in \{1, \ldots, |V|\}$
Frage: Gibt es eine Färbung $c : V \to \{1, \ldots, k\}$ der Knoten von G mit k Farben, so dass benachbarte Knoten verschiedene Farben haben, d.h. $\forall \{u, v\} \in E : c(u) \neq c(v)$.

Problem (Erfüllbarkeitsproblem / Satisfiability — SAT)

Eingabe: Aussagenlogische Formel ϕ in KNF
Frage: Gibt es eine erfüllende Belegung für ϕ?

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

Satz

COLORING $\leq_p$ *SAT.*

Beweis:

Wir beschreiben eine polynomiell berechenbare Funktion f, die eine Eingabe (G, k) für das COLORING-Problem auf eine Formel ϕ für das SAT-Problem abbildet, mit der Eigenschaft

$$G \text{ hat eine } k\text{-Färbung} \quad \Leftrightarrow \quad \phi \text{ ist erfüllbar .}$$

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

Beschreibung der Funktion f :

Die Formel ϕ hat für jede Knoten-Farb-Kombination (v, i), $v \in V$, $i \in \{1, \ldots, k\}$, eine Variable x_v^i. Die Formel für (G, k) lautet

$$\phi = \bigwedge_{v \in V} \underbrace{(x_v^1 \vee x_v^2 \vee \ldots \vee x_v^k)}_{\text{Knotenbedingung}} \wedge \bigwedge_{\{u,v\} \in E} \bigwedge_{i \in \{1,\ldots,k\}} \underbrace{(\bar{x}_u^i \vee \bar{x}_v^i)}_{\text{Kantenbedingung}} .$$

Anzahl der Literale $= O(k \cdot |V| + k \cdot |E|) = O(|V|^3)$.

Die Länge der Formel ist somit polynomiell beschränkt und die Formel kann in polynomieller Zeit konstruiert werden.

Aber ist die Konstruktion auch korrekt?

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

Korrektheit:

zz: G hat eine k Färbung $\Rightarrow$ ϕ ist erfüllbar

- Sei c eine k-Färbung für G.
- Für jeden Knoten v mit $c(v) = i$ setzen wir $x_v^i = 1$ und alle anderen Variablen auf 0.
- Knotenbedingung: Offensichtlich erfüllt.
- Kantenbedingung: Für jede Farbe i und jede Kante $\{u, v\}$ gilt $\bar{x}_u^i \vee \bar{x}_v^i$, denn sonst hätten u und v beide die Farbe i.
- Damit erfüllt diese Belegung die Formel ϕ.

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

zz: ϕ ist erfüllbar $\Rightarrow$ G hat eine k Färbung

- Fixiere eine beliebige erfüllende Belegung für ϕ.
- Wegen der Knotenbedingung gibt es für jeden Knoten v mindestens eine Farbe mit $x_v^i = 1$.
- Für jeden Knoten wähle eine beliebige derartige Farbe aus.
- Sei $\{u, v\} \in E$. Wir behaupten $c(u) \neq c(v)$.
- Zum Widerspruch nehmen wir an, $c(u) = c(v) = i$. Dann wäre $x_u^i = x_v^i = 1$ und die Kantenbedingung $\bar{x}_u^i \vee \bar{x}_v^i$ wäre verletzt.

□

Beispiel einer polyn. Reduktion: COLORING $\leq_p$ SAT

COLORING $\leq_p$ SAT impliziert die folgenden beiden Aussagen.

Korollar

Wenn SAT einen Polynomialzeitalgorithmus hat, so hat auch COLORING einen Polynomialzeitalgorithmus.

Korollar

Wenn COLORING keinen Polynomialzeitalgorithmus hat, so hat auch SAT keinen Polynomialzeitalgorithmus.

NP-harte Probleme

Definition (NP-Härte)

Ein Problem L heißt NP*-hart, wenn* $\forall L' \in \mathrm{NP} : L' \leq_p L$.

Satz

L NP*-hart,* $L \in \mathrm{P} \Rightarrow \mathrm{P} = \mathrm{NP}$

Beweis: Polyzeitalgo für L liefert Polyzeitalgo für alle $L' \in \mathrm{NP}$. $\square$

Fazit: NP-harte Probleme haben keine Polyzeitalgo, es sei denn P$=$ NP.

NP-vollständige Probleme

Definition (NP-Vollständigkeit)

Ein Problem L heißt NP-vollständig, *falls gilt*

1. $L \in$ NP, *und*
2. L *ist* NP*-hart.*

Wir werden zeigen, dass SAT, CLIQUE, KP-E, BPP-E, TSP-E und viele weitere Probleme NP-vollständig sind.

Keines dieser Probleme hat somit einen Polynomialzeitalgorithmus; es sei denn P$=$ NP.

NP-Vollständigkeit des Erfüllbarkeitsproblems

Der Ausgangspunkt für unsere NP-Vollständigkeitsbeweise ist das Erfüllbarkeitsproblem.

Satz (Cook und Levin)

SAT ist NP*-vollständig.*

SAT hat somit keinen Polynomialzeitalgorithmus; es sei denn $P = NP$.

Beweis des Satzes von Cook und Levin

Offensichtlich gilt SAT $\in$ NP, denn die erfüllende Belegung kann als Zertifikat verwendet werden. Wir müssen also „nur“ noch zeigen, dass SAT NP-hart ist.

Sei $L \subseteq \Sigma^*$ ein Problem aus NP. Wir müssen zeigen $L \leq_p$ SAT.

Dazu konstruieren wir eine polynomiell berechenbare Funktion f, die jedes $x \in \Sigma^*$ auf eine Formel ϕ abbildet, so dass gilt

$$x \in L \Leftrightarrow \phi \in SAT \ .$$

Beweis des Satzes von Cook und Levin

M sei eine NTM, die L in polynomieller Zeit erkennt. Wir zeigen

$$M \text{ akzeptiert } x \Leftrightarrow \phi \in SAT \ .$$

Eigenschaften von M

- O.B.d.A. besuche M keine Bandpositionen links von der Startposition.
- Eine akzeptierende Rechnung von M gehe in den Zustand q_{accept} über und bleibt dort in einer Endlosschleife.
- Sei $p(x)$ ein Polynom, so dass M eine Eingabe x genau dann akzeptiert, wenn es einen Rechenweg gibt, der nach $p(|x|)$ Schritten im Zustand q_{accept} ist.

Beweis des Satzes von Cook und Levin

Beobachtung:

Sei $K_0 = q_0 x$ die Startkonfiguration von M. M akzeptiert genau dann, wenn es eine mögliche Konfigurationsfolge

$$K_0 \vdash K_1 \vdash \cdots \vdash K_{p(n)}$$

gibt, bei der $K_{p(n)}$ im Zustand q_{accept} ist.

Weiteres Vorgehen:

Wir konstruieren die Formel ϕ derart, dass ϕ genau dann erfüllbar ist, wenn es eine solche akzeptierende Konfigurationsfolge gibt.

Beweis des Satzes von Cook und Levin

Variablen in ϕ

- $Q(t,k)$ für $t \in \{0,\ldots,p(n)\}$ und $k \in Q$
- $H(t,j)$ für $t,j \in \{0,\ldots,p(n)\}$
- $S(t,j,a)$ für $t,j \in \{0,\ldots,p(n)\}$ und $a \in \Gamma$

Interpretation der Variablen:

- Die Belegung $Q(t,k) = 1$ soll besagen, dass sich die Rechnung zum Zeitpunkt t im Zustand k befindet.
- Die Belegung $H(t,j) = 1$ steht dafür, dass sich der Kopf zum Zeitpunkt t an Bandposition j befindet.
- die Belegung $S(t,j,a) = 1$ bedeutet, dass zum Zeitpunkt t an Bandposition j das Zeichen a geschrieben steht.

Beweis des Satzes von Cook und Levin

Kodierung einzelner Konfigurationen in der Teilformel ϕ_t:

Für jedes $t \in \{0, \ldots, p(n)\}$, benötigen wir eine Formel ϕ_t, die nur dann erfüllt ist, wenn es

1. genau einen Zustand $k \in Q$ mit $Q(t, k) = 1$ gibt,
2. genau eine Bandposition $j \in \{0, \ldots, p(n)\}$ mit $H(t, j) = 1$ gibt, und
3. für jedes $j \in \{0, \ldots, p(n)\}$ jeweils genau ein Zeichen $a \in \Gamma$ mit $S(t, j, a) = 1$ gibt.

Beweis des Satzes von Cook und Levin

Erläuterung zur Formel ϕ_t:

- Für eine beliebige Variablenmenge $\{y_1, \ldots, y_m\}$ besagt das folgende Prädikat, dass genau eine der Variablen y_i den Wert 1 annimmt:

$$(y_1 \vee \ldots \vee y_m) \wedge \bigwedge_{i \neq j} \neg(y_i \wedge y_j)$$

- Wie kann diese Formel in KNF gebracht werden?
- Die Anzahl der Literale in dieser Formel ist quadratisch in der Anzahl der Variablen.
- Die drei Anforderungen können also jeweils durch eine Formel in KNF in polynomiell beschränkter Länge kodiert werden.

Beweis des Satzes von Cook und Levin

Wir betrachten nun nur noch Belegungen, die die Teilformeln $\phi_0, \ldots, \phi_{p(n)}$ erfüllen und somit Konfigurationen $K_0, \ldots, K_{p(n)}$ beschreiben.

Als nächstes konstruieren wir eine Formel ϕ'_t für $1 \leq t \leq p(n)$, die nur für solche Belegungen erfüllt ist, bei denen K_t eine direkte Nachfolgekonfiguration von K_{t-1} ist.

Die Formel ϕ'_t besteht aus zwei Arten von Teilformeln ...

Beweis des Satzes von Cook und Levin

Zunächst beschreiben wir eine Teilformel, welche festlegt, dass die Bandinschrift von K_t an allen Positionen außer der Kopfposition (zum Zeitpunkt $t-1$) mit der Inschrift von K_{t-1} übereinstimmt.

Kodierung von Konfigurationsübergängen an Nicht-Kopfpositionen:

$$\bigwedge_{i=0}^{p(n)} \bigwedge_{z\in\Gamma} ((S(t-1,i,z) \wedge \neg H(t-1,i)) \Rightarrow S(t,i,z))$$

Wie kann diese Formel in KNF gebracht werden? ...

Beweis des Satzes von Cook und Levin

Für den Konfiguationsübergang müssen wir außerdem beschreiben, dass an der Kopfposition der richtige δ-Übergang realisiert wird.

Kodierung von Konfigurationsübergängen an der Kopfposition:

Die folgende Teilformel wird für alle $k \in Q$, $j \in \{0, \ldots, p(|x|) - 1\}$ und $a \in \Gamma$ benötigt:

$$(Q(t-1,k) \wedge H(t-1,j) \wedge S(t-1,j,a)) \quad \Rightarrow \bigvee_{(k,a,k',a',\kappa)\in\delta} (Q(t,k') \wedge H(t,j+\kappa) \wedge S(t,j,a')) \ ,$$

wobei κ die Werte $L = -1$, $N = 0$ und $R = 1$ annehmen kann.

Wie kann diese Formel in KNF gebracht werden? ...

Damit ist die Beschreibung von ϕ'_t abgeschlossen.

Beweis des Satzes von Cook und Levin

Die vollständige Formel ϕ lautet nun

$$Q(0,q_0) \wedge H(0,0) \wedge \bigwedge_{i=0}^{n} S(0,i,x_i) \wedge \bigwedge_{i=n+1}^{p(n)} S(0,i,B)$$
$$\wedge \bigwedge_{i=0}^{p(n)} \phi_i \wedge \bigwedge_{i=1}^{p(n)} \phi'_i \wedge Q(p(n),q_{\text{accept}}) \ .$$

Gemäß unserer Konstruktion ist ϕ genau dann erfüllbar, wenn es eine akzeptierende Konfigurationsfolge für M auf x der Länge $p(|x|)$ gibt. □

Kochrezept für NP-Vollständigkeitsbeweise

- Um Nachzuweisen, dass SAT NP-hart ist, haben wir in einer „Master-Reduktion“ alle Probleme aus NP auf SAT reduziert.
- Die NP-Vollständigkeit von SAT können wir jetzt verwenden, um nachzuweisen, dass weitere Probleme NP-hart sind.

Lemma

L^* NP-*hart*, $L^* \leq_p L \Rightarrow L$ NP-*hart*.

Beweis: Gemäß Voraussetzung gilt $\forall L' \in \text{NP} : L' \leq_p L^*$ und $L^* \leq_p L$. Aufgrund der Transitivität der polynomiellen Reduktion folgt somit $\forall L' \in \text{NP} : L' \leq_p L$. □

Karps Liste mit 21 NP-vollständigen Problemen

- SAT
  - CLIQUE
  - SET PACKING
  - VERTEX COVER
    - SET COVERING
    - FEEDBACK ARC SET
    - FEEDBACK NODE SET
    - DIRECTED HAMILTONIAN CIRCUIT
      - UNDIRECTED HAMILTONIAN CIRCUIT

Die Schachtelungstiefe beschreibt den Weg der Reduktionen, wie Karp sie in seinem Artikel geführt hat.

Karps Liste mit 21 NP-vollständigen Problemen

- SAT
  - 0-1 INTEGER PROGRAMMING
  - 3SAT
    - CHROMATIC NUMBER (COLORING)
      - CLIQUE COVER
      - EXACT COVER
        - 3-dimensional MATCHING
        - STEINER TREE
        - HITTING SET
        - KNAPSACK
          - JOB SEQUENCING
          - PARTITION
            - MAX-CUT

Es gibt noch tausende weitere bekannte NP-vollständige Probleme.