Komplexitätstheorie

Kapitel 5: Zeitkomplexität, reloaded

Einleitung

Weitere Themen rund um Zeitkomplexität:

- Schaltkreiskomplexität und parallele Berechnungen (noch mehr Struktur innerhalb von P!)

- P-Härte

- Die Polynomielle Hierarchie: mehr Struktur zwischen P und PSpace

Kapitel 5

Schaltkreiskomplexität

Motivation

Schaltkreis ist Verschaltung von logischen Gattern, berechnet *Boolsche Funktion* $f : \{0,1\}^n \to \{0,1\}$

Das ist nichts weiter als Entscheidungsverfahren für Problem $L \subseteq \{0,1\}^*$, eingeschränkt auf Eingaben der Länge n

Familie von Schaltkreisen $(C_i)_{i\in\mathbb{N}}$ entscheidet also das gesamte L.

Schaltkreise interessant als Berechnungsmodell da:

- sie als abstraktes Modell für Hardware verstanden werden können
- man sie verwenden kann, um massiv parallele Berechnungen zu beschreiben

Schaltkreise

Definition (Boolscher) Schaltkreis

N-ärer *Boolscher Schaltkreis* C ist Tupel $(V, E, \omega, x_1, \dots, x_n, o)$ wobei

- (V, E) gerichteter azyklischer Graph,
- $x_1, \dots, x_n \in V$ *Eingabeknoten* mit Eingangsgrad 0
- $o \in V$ *Ausgabeknoten* mit Ausgangsgrad 0
- $\omega : V \setminus \{x_1, \dots, x_n\} \to \{\neg, \wedge, \vee, 0, 1\}$ Knotenbeschriftung so dass
  - $\omega(v) = \neg$ impliziert Eingangsgrad$(v) = 1$
  - $\omega(v) \in \{\wedge, \vee\}$ impliziert Eingangsgrad$(v) = 2$
  - $\omega(v) \in \{0, 1\}$ impliziert Eingangsgrad$(v) = 0$

Bei gegebener Belegung der Eingabeknoten mit 0,1 kann für jeden Knoten ein Wert berechnet werden. C berechnet Boolsche Funktion $f_C : \{0,1\}^* \to \{0,1\}$ wobei $f_C(b_1, \dots, b_n)$ der Wert des Knotens o bei Eingabebelegung $b_1, \dots, b_n$ ist

Komplexitätsmaße

Die Nicht-Eingabeknoten eines Schaltkreises werden *Gates* genannt

Sinnvolle Komplexitätsmaße für Schaltkreise:

- *Größe* $|C|$ von Schaltkreis C ist die Anzahl seiner Gates
- *Tiefe* $d(C)$ von Schaltkreis C ist die Länge des längsten Pfades in C

Schaltkreis als Modell für Hardware:

Größe beschreibt Anzahl benötigter Bauelemente

Schaltkreis als Modell für massiven parallele Berechnungen:

- Größe beschreibt Anzahl benötigter Prozessoren
- Tiefe beschreibt Dauer der Berechnung

Schaltkreise und Boolsche Funktionen

Definition Schaltkreiskomplexität

Schaltkreiskomplexität einer Boolschen Funktion f ist min$\{|C| \mid f_C = f\}$

Es gibt Boolsche Funktionen mit exponentieller Schaltkreiskomplexität:

Lemma

Für alle $n > 2$ gibt es Boolsche Funktion $f : \{0,1\}^n \to \{0,1\}$ mit Schaltkreiskomplexität $\geq \frac{2^n}{2n}$.

Interessanterweise kennt man keine "natürliche" Boolsche Funktion, die mehr als linear viele Gates benötigt!

Das ist noch erstaunlicher, da man zeigen kann, dass fast alle Boolschen Funktionen exponentielle Schaltkreiskomplexität haben.

Schaltkreise und Entscheidungsprobleme

Erkennen von Sprache:

- Wir beschränken uns o.B.d.A. auf Sprachen $L \subseteq \{0,1\}^*$
- Wir verwenden Familie $(C_n)_{n\in\mathbb{N}} = (C_1, C_2, C_3, \dots)$ wobei C_n Arität n hat und Eingaben der Länge n verarbeitet

Definition Erkennen von Sprachen

Familie $(C_n)_{n\in\mathbb{N}}$ von Schaltkreisen *erkennt* Sprache $L \subseteq \{0,1\}^*$ wenn jedes C_n die charakteristische Funktion $F_{L,n} : \{0,1\}^n \to \{0,1\}$ von $L \cap \{0,1\}^n$ berechnet, wobei

$$F_{L,n}(w) := \begin{cases} 1 & \text{wenn } w \in L \\ 0 & \text{wenn } w \notin L \end{cases}$$

Paritäts-Beispiel leicht zu generalisieren zu Schaltkreisfamilie für

$$\text{PARITY} := \{w \in \{0,1\}^* \mid w \text{ hat Parität } 1\}$$

Schaltkreise und Entscheidungsprobleme

Boolsche Funktion:

Ein einziger Schaltkreis, Schaltkreiskomplexität ist Zahl

Sprache:

Familie von Schaltkreisen, Schaltkreiskomplexität ist Funktion
(wie Zeit- und Platzkomplexität von TMs auch)

Definition Schaltkreiskomplexität von Sprachen

Schaltkreiskomplexität von L ist Funktion $f : \mathbb{N} \rightarrow \mathbb{N}$, so dass $f(n)$ die Schaltkreiskomplexität von $F_{L,n}$ ist.

Polynomielle Schaltkreiskomplexität

Theorem

Jedes $L \in P$ hat polynomielle Schaltkreiskomplexität.

Ideen:

- Fixiere p-zeitbeschränkte DTM M, nimm o.B.d.A. an, dass Eingabealphabet $\Sigma = \{0, 1\}$ ist.
- Für jede Eingabelänge n, konstruiere Schaltkreis C_n so dass: M akzeptiert w gdw. $f_{C_n}(w) = 1$
- Stelle Berechnung wieder als $(p(n) + 2) \times (p(n) + 1)$-Matrix dar:

$\triangleright$	q_0, a_0	a_1	$\cdots$	a_n	$\perp$	$\cdots$	$\perp$
$\triangleright$	b	q, a_1	$\cdots$	a_n	$\perp$	$\cdots$	$\perp$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

- Kodiere jeden möglichen Feldinhalt $(\Gamma \cup (Q \times \Gamma))$ als binäre Zahl

Uniformität

Die Umkehrung gilt nicht ohne weiteres.

Theorem

Es gibt unentscheidbare Probleme mit konstanter Schaltkreiskomplexität.

Problem: jeder Schaltkreis im Beweis ist sehr einfach, aber das Berechnen des Schaltkreises ist sehr schwer (bzw. unmöglich)

Definition Uniformität

Familie $(C_n)_{n\in\mathbb{N}}$ ist *uniform*, wenn es LogSpace-Transduktor gibt, der bei Eingabe n den Schaltkreis C_n ausgibt.

Bei Schaltkreisen als Modell für parallele Berechnungen ist uniformität offenbar eine sehr natürliche Annahme, Paritätsbeispiel ist uniform.

Uniformität

Uniformität stellt die gewünschte Äquivalenz her

Definition CVP

Das *Schaltkreisauswertungsproblem (Circuit Value Problem, CVP)*:

$$\text{CVP} := \{(C, w) \mid C\ n\text{-ärer Schaltkreis}, w \in \{0,1\}^n, f_C(w) = 1\}$$

Leicht zu sehen: CVP ist in P (Azyklizität ausnutzen)

Theorem

Jedes Problem, das durch eine uniforme Familie von polynomiell größenbeschränkten Schaltkreisen erkennbar ist, ist in P.

Interessanter: nicht nur Größe, sondern auch Tiefe betrachten

Kapitel 5

Die Klasse NC

NC

Massiv parallele Berechenbarkeit:

- Die Schaltkreis*tiefe* (Berechnungszeit) sollte nur logarithmisch sein
- Es ist akzeptabel, wenn die Schaltkreis*größe* (Anzahl Prozessoren) polynomiell ist

Definition NC

Problem L ist in NC^i, $i \geq 1$, wenn es uniforme Familie $(C_n)_{n\in\mathbb{N}}$ gibt so dass

- $(C_n)_{n\in\mathbb{N}}$ erkennt L
- es gibt $k \in \mathbb{N}$ mit $|C_n| \in \mathcal{O}(n^k)$
- $d(C_n) \in \mathcal{O}(\log^i(n))$

Nun ist NC $:= \bigcup_{i\geq 0}$ NC^i

Schon gesehen: PARITY $\in$ NC^1 $\subseteq$ NC

NC steht für Nick's Class, nach Nicolas Pippenger

NC

NC wird oft mit “effizient parallelisierbar” identifiziert, wobei aber:

- Bereits bei $i = 3$ ist $\log^i(n)$ nur bei *sehr* grossen Eingaben signifikant schneller als n (z.B. $\log^i(10000) \geq \frac{10000}{4}$)

  Dafür ist die ”poly viele Prozessoren” Annahme eher unrealistisch

- Entscheidungsprobleme und Optimierungsprobleme nicht mehr wechselseitig reduzierbar

Trotzdem ist NC wichtige Komplexitätsklasse!

Beachte: $\mathrm{NC}^1 \subseteq \mathrm{NC}^2 \subseteq \cdots$ NC ist unendliche Hierarchie in NC

Echtheit der Inklusionen unbekannt!

NC

Wir setzen nun NC in Beziehung zu unseren bisherigen Klassen

Schon gezeigt:

Theorem

$NC \subseteq P$

Theorem

$NC^1 \subseteq \text{LogSpace}$

NC

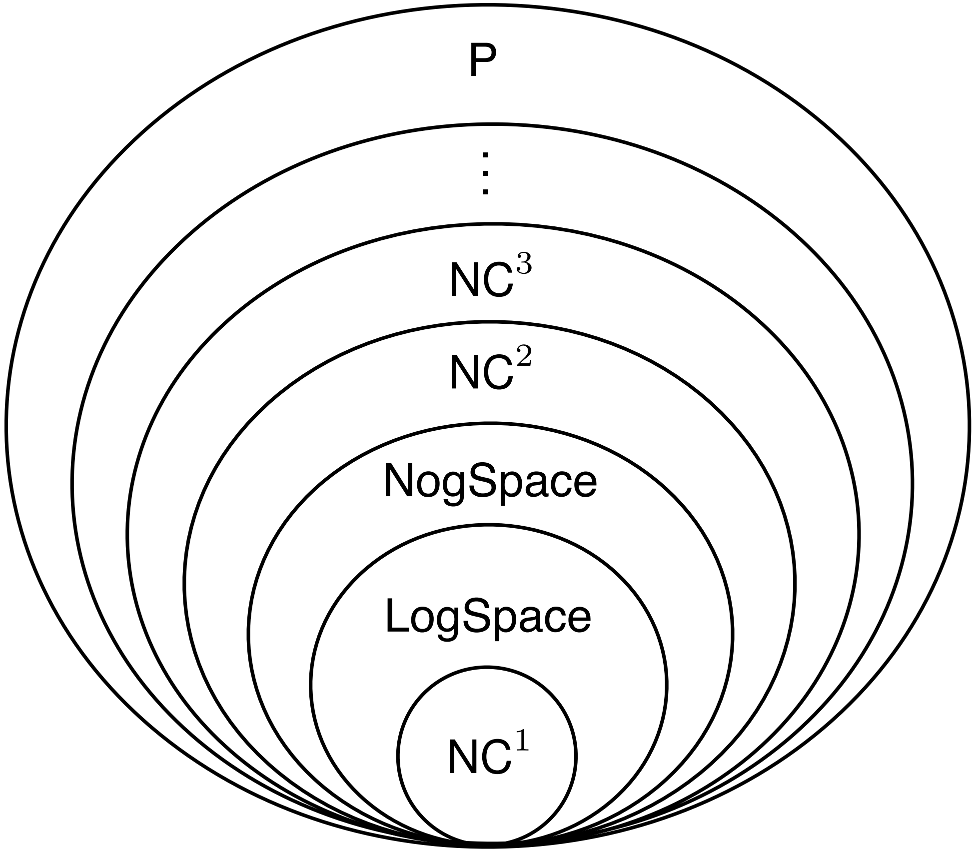
Theorem

NLogSpace $\subseteq$ NC^2 $\subseteq$ NC

Idee für Konstruktion von C_n:

- Wir repräsentieren Konfigurationen durch uqv für Arbeitsband plus Kopfposition auf Eingabeband (aber nicht dessen Inhalt)
- Dann ist Konfigurationsmenge nur von n abhängig, nicht von genauer Eingabe
- Der Schaltkreis berechnet den transitiven Abschluss von "$\vdash_M$" auf dieser Konfigurationsmenge
- Der Basisfall $\alpha \vdash_M \alpha'$ hängt dabei von Eingabe ab (da Eingabeband nicht repräsentiert)

NC



Kapitel 5

Die Klasse AC

AC

In den bisher betrachteten Schaltkreisen sind alle Gates binär

Definition Generalisierter Schaltkreis

Ein *generalisierter Schaltkreis* ist ein Schaltkreis, in dem es folgende Knotenarten gibt:

- Eingabeknoten
- 0- und 1-Knoten mit Eingangsgrad 0
- $\wedge$- und $\vee$-Knoten mit Eingangsgrad ≥ 1
- $\overline{\wedge}$- und $\overline{\vee}$-Knoten mit Eingangsgrad ≥ 1
  (wie $\wedge$ und $\vee$, jedoch mit negiertem Ergebnis)

Man nennt das auch “unbounded fan-in”

AC

Wir können nun Komplexitätsklassen analog zu NC definieren

Definition AC

Problem L ist in AC^i, $i \geq 0$, wenn es uniforme Familie von generalisierten Schaltkreisen $(C_n)_{n\in\mathbb{N}}$ gibt so dass

- $(C_n)_{n\in\mathbb{N}}$ erkennt L
- es gibt $k \in \mathbb{N}$ mit $|C_n| \in \mathcal{O}(n^k)$
- $d(C_n) \in \mathcal{O}(\log^i(n))$

Nun ist AC $:= \bigcup_{i\geq 0}$ AC^i

Theorem

Für alle $n \geq 0$: NC$^i \subseteq$ AC$^i \subseteq$ NC^{i+1}

Ob alle diese Inklusionen echt sind, ist offen.

AC

Interessanterweise konnte folgendes negatives Resultat bewiesen werden (ohne Beweis):

Theorem

PARITY $\notin$ AC^0

Beachte: AC^0 bezieht sich auf Schaltkreise von konstanter Tiefe und poly Größe (davon gibt es unendlich viele wegen unbounded fan-in)

Daraus folgt offensichtlich:

Theorem

AC^0 $\subsetneq$ NC^1, also auch AC^0 $\subsetneq$ P.

Kapitel 5

P-Härte

P-Härte

Wegen der identifizierten, reichen Struktur innerhalb von P macht es Sinn, Härte und Vollständigkeit für P zu betrachten.

Insbesondere NC vs. P: ist jedes Polynomialzeitproblem effizient parallelisierbar?

Vermutlich nicht, aber Beweis steht aus!

Polynomialzeit-Reduktionen sind hier nicht sinnvoll, da

für alle $L, L' \in$ P mit L' nicht-trivial: $L \leq_p L'$

(*nicht-trivial*: es gibt positive Instanzen und negative Instanzen)

P-Härte

Definition P-Härte, P-Vollständigkeit

Problem L ist

- *P-hart* wenn $L' \leq_{\text{log}} L$ für alle $L' \in$ NP;
- *P-vollständig* wenn L P-hart und in P.

Also: wenn Problem L P-vollständig, dann

1. L nicht mit logarithmischem Platz entscheidbar, außer wenn LogSpace = P
2. L nicht effizient parallelisierbar (in NC), außer wenn NC = P

Für 2. brauchen wir allerdings noch:

Theorem

Wenn $L \in$ NC und $L' \leq_{\text{log}} L$, dann $L' \in$ NC.

P-Härte

Theorem

CVP ist P-vollständig.

Weitere P-vollständige Probleme z.B.:

- das Leerheitsproblem für kontextfreie Grammatiken
- gegeben einen ungerichteten Graph und $n \geq 0$, gibt es Teilgraph (induziert durch Knotenmenge) in dem alle Knoten mindestens Grad n haben?

Kapitel 5

Die polynomielle Hierarchie

PH

(N)LogSpace, NC, AC, etc: reiche Struktur innerhalb von P

Die polynomielle Hierarchie liefert Struktur zwischen P und PSpace

Wichtiges Problem für Schaltkreisentwurf:

Definition Minimal Circuit (MC)

Schaltkreis C ist minimal wenn $|C'| \geq |C|$ für alle C' mit $f_C = f_{C'}$.
MC ist Menge aller minimalen Schaltkreise.

Was ist die "richtige" Komplexitätsklasse (Vollständigkeit!) für dieses Problem?

PH

Orakel: Unterprogramm, dessen Zeitverbrauch wir mit "1" bewerten

Wird als formale Sprache dargestellt

Definition Orakel-TM

Eine *Orakel-TM (OTM)* M^O ist eine (deterministische oder nicht-deterministische) TM M ausgestattet mit einem Orakel $O \subseteq \Sigma^*$. OTM hat

- ein zusätzliches *Orakelband*
- drei spezielle Zustände $q_?, q_+, q_-$.

Für q_+ und q_- sind normale Transitionen definiert. Der Folgezustand von $q_?$ ist q_+ wenn das momentane Wort auf dem Orakelband in O ist und q_- sonst. Kopfposition und Bandinhalte bleiben dabei unverändert.

Beachte: das Orakel kann sehr komplex sein, sogar unentscheidbar!

PH

Definition Orakel-Komplexitätsklassen

Sei $O \subseteq \Sigma^*$ ein Orakel. Dann:

- $\mathsf{P}^O := \{L \mid L$ wird von ODTM M^O in poly-Zeit entschieden $\}$
- $\mathsf{NP}^O := \{L \mid L$ wird von ONTM M^O in poly-Zeit entschieden $\}$

Sei $\mathcal{C}$ Komplexitätsklasse. Dann:

$$\mathsf{P}^{\mathcal{C}} := \bigcup_{O \in \mathcal{C}} \mathsf{P}^O \qquad \mathsf{NP}^{\mathcal{C}} := \bigcup_{O \in \mathcal{C}} \mathsf{NP}^O$$

Leicht zu sehen:

- $\mathsf{P}^{\mathsf{P}} = \mathsf{P}$, $\mathsf{NP}^{\mathsf{P}} = \mathsf{NP}$;
- $\mathsf{P}^{\mathsf{NP}} = \mathsf{P}^{\mathsf{SAT}}$ und ebenso für jedes andere NP-vollständige Problem

$\mathsf{NP}^{\mathsf{NP}} = \mathsf{NP}$ ist hingegen nicht klar, denn co-NP $\subseteq \mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{NP}^{\mathsf{NP}}$

PH

Wir werden im folgenden auch Komplemente von Orakelklassen verwenden.

Einige einfache Beobachtungen:

- $\mathsf{P}^{\mathcal{C}} = \mathsf{P}^{\text{co-}\mathcal{C}}$ und $\mathsf{NP}^{\mathcal{C}} = \mathsf{NP}^{\text{co-}\mathcal{C}}$ (vertausche q_+ und q_-)
- co-$\mathsf{NP}^{\mathcal{C}}$ bedeutet co-$(\mathsf{NP}^{\mathcal{C}})$, denn $(\text{co-}\mathsf{NP})^{\mathcal{C}}$ nicht definiert
- co-$(\mathsf{P}^{\mathcal{C}}) = P^{\mathcal{C}}$ analog zu $\mathsf{P} =$ co-P
- co-$\mathsf{NP}^{\mathcal{C}}$ vs. $\mathsf{NP}^{\mathcal{C}}$ analog zu co-NP vs. NP

PH

Die polynomielle Hierarchie entsteht nun durch wiederholtes Orakel-anwenden

Definition Polynomielle Hierarchie

- $\Sigma_1^p = \text{NP}$, $\Pi_1^p =$ co-NP, $\Delta_1^p = \text{P}$
- Für $k \geq 1$ sei
  - $\Sigma_{k+1}^p = \text{NP}^{\Sigma_k^p}$
  - $\Pi_{k+1}^p =$ co-Σ_{k+1}^p
  - $\Delta_{k+1}^p = \text{P}^{\Sigma_k^p}$

Also schon gezeigt: MC $\in \Pi_2^p$.

Lemma

Für alle $k \geq 1$ gilt: $\Delta_k^p \subseteq \Sigma_k^p \subseteq \Delta_{k+1}^p$ und $\Delta_k^p \subseteq \Pi_k^p \subseteq \Delta_{k+1}^p$

PH

$$
\begin{tikzcd}
 & \vdots & \\
 & \Delta_3^p & \\
\Sigma_2^p \arrow[ur, "\subseteq"] & & \Pi_2^p \arrow[ul, "\subseteq"'] \\
 & \Delta_2^p \arrow[ul, "\subseteq"] \arrow[ur, "\subseteq"'] & \\
\Sigma_1^p = \text{NP} \arrow[ur, "\subseteq"] & & \Pi_1^p = \text{co-NP} \arrow[ul, "\subseteq"'] \\
 & \Delta_1^p = \text{P} \arrow[ul, "\subseteq"] \arrow[ur, "\subseteq"'] &
\end{tikzcd}
$$

Echtheit der Inklusionen ist unbekannt.

PH

Es gibt auch eine Klasse für die gesamt polynomielle Hierarchie:

Definition Polynomielle Hierarchie

$\mathsf{PH} = \bigcup_{k \geq 1} \Sigma_k^p$

Die polynomielle Hierarchie liegt zwischen P und PSpace:

Theorem

$\mathsf{PH} \subseteq \textsc{PSpace}$

PH

Die polynomielle Hierarchie kollabiert wenn PH $= \Sigma_k^p$ für eine $k \geq 1$

Lemma

Wenn $\Sigma_k^p = \Sigma_{k+1}^p$, dann PH $= \Sigma_k^p$.

Also: $\Sigma_k^p \neq \Sigma_{k+1}^p$ stärkere Annahme als $\Sigma_{k-1}^p \neq \Sigma_k^p$

und P $\neq$ NP schwächste aller dieser Annahmen

Anders formuliert: PH kollabiert am ehesten weit oben!

Theorem

Wenn PH $=$ PSPACE, dann kollabiert PH.

Kapitel 5

Logische Charakterisierung der Polynomiellen Hierarchie

Charakterisierung PH

Charakterisierung generalisiert Definition von NP

Theorem

$L \in \Sigma_k^p$ gdw. es Polynom p und $L' \in \mathrm{P}$ gibt so dass

$$L = \{w \mid \exists u_1 \in A. \forall u_2 \in A. \exists u_3 \in A \ldots Q u_k \in A : (w, u_1, \ldots, u_k) \in L'\}.$$

wobei $A = \{0,1\}^{p(|w|)}$ und Q der sich durch Alternierung ergebende Quantor.

Die Klassen der polynomiellen Hierarchie werden also mittels logischer Ausdrückbarkeit beschrieben

Frage nach Echtheit der Inklusionen in PH: liefern zusätzliche Quantoren-alternierungen zusätzliche Ausdrucksstärke?

Charakterisierung PH

Lemma

Für $L \subseteq \Sigma^*$ gilt $L \in \Sigma_k^p$ gdw. es gibt Polynom p und Relation $R \subseteq \Sigma^* \times \Gamma^*$ so dass

- $(w, b) \in R$ impliziert $|b| \leq p(|w|)$
- $R \in \Pi_{k-1}^p$ (wobei $\Pi_0^p :=$ P)
- $L = \{w \mid \exists b : (w, b) \in R\}$

Idee:

- Induktion über k
- Der Fall $k = 1$ folgt direkt aus Definition NP
- In "$\Rightarrow$" ist der Beweis b eine Berechnung der NTM zusammen mit Beweisen für die "ja"-Antworten des Orakels (induktiv)

Charakterisierung PH

Korollar

Für $L \subseteq \Sigma^*$ gilt $L \in \Pi_k^p$ gdw. es gibt Polynom p und Relation $R \subseteq \Sigma^* \times \Gamma^*$ so dass

- $(w, b) \in R$ impliziert $|b| \leq p(|w|)$
- $R \in \Sigma_{k-1}^p$ (wobei $\Sigma_0^p :=$ P)
- $L = \{w \mid \forall b \in \Gamma^* \text{ mit } |b| \leq p(|w|) : (w, b) \in R\}$

Beweis: Für $\overline{L} \in \Sigma_k^p$ gibt es R wie in vorigem Lemma, verwende für L:

$$\widehat{R} := \{(w, b) \in \Sigma^* \times \Gamma^* \mid (w, b) \notin R \text{ und } |b| \leq p(|w|)\}$$

Aus Lemma + Korollar folgt nun das ursprüngliche Theorem:
Ersetze wiederholt Σ_i^p und Π_i^p durch ihre Beweissysteme

PH

Vollständigkeit

Vollständigkeit

Um Probleme korrekt in die polynomielle Hierarchie "einzuordnen", brauchen wir Vollständigkeitsbegriff

Definition

Für $k \geq 1$ ist Problem L

- Σ_k^p-*hart* wenn $L' \leq_p L$ für alle $L' \in \Sigma_k^p$;
- *NP-vollständig* wenn L sowohl Σ_k^p-hart als auch in Σ_k^p.

Für Π_k^p, Δ_k^p und PH analog (ausser für $\Delta_1^p =$ P)

Aber PH hat wahrscheinlich keine vollständigen Probleme:

Lemma

Wenn für PH vollständige Probleme existieren, kollabiert die Hierarchie

Vollständigkeit

QBF liefert uniforme Familie von "typischen" vollständigen Problemen

Für $\overline{V} = v_1, \ldots, v_n$ schreiben wir $\exists\overline{V}$ als Abkürzung für $\exists v_1 \cdots \exists v_n$

$\forall\overline{V}$ als Abkürzung für $\forall v_1 \cdots \forall v_n$

Definition k-QBF

QBF $Q_1\overline{V_1} \cdots Q_n\overline{V_n}\varphi$ heisst k-*QBF* wenn

- $n = k$
- $Q_1 = \exists$, $Q_2 = \forall$, $Q_3 = \exists$, etc. (Quantoren alternieren)

QBF$_k$ ist die Menge aller gültigen k-QBFs.

Beispiel für 3-QBF: $\exists v_1 \exists v_2 \forall v_3 \exists v_4 \exists v_5 . \varphi$

Vollständigkeit

Theorem

Für alle $k \geq 1$ ist QBF$_k$ Σ_k^p-vollständig.

Idee:

- "in Σ_k^p": benutze logische Charakterisierung
- Härte: benutze logische Charakterisierung und Übersetzung von TM in AL-Formel analog zum Beweis von Cook's Theorem

Beginnt man die Quantorenalternierung mit "$\forall$", so ist k-QBF Π_k^p-vollständig.

Vollständigkeit

In der Logik gibt es verschiedene natürliche Probleme, die vollständig für Klassen der polynomiellen Hierarchie sind.

Definition MINSAT

Für zwei WZen π und π' schreiben wir $\pi \leq \pi'$ gdw.

$$\pi'(v) = 1 \text{ impliziert } \pi(v) = 1 \text{ für alle Variablen } V$$

π ist *minimales Modell* von AL-Formel φ wenn

- π erfüllt φ
- für alle π', die φ erfüllen, gilt $\pi \leq \pi'$

MINSAT ist die Menge aller Tripel (φ, v) mit φ AL-Formel und v Variable so daß $\pi(v) = 0$ in allen minimalen Modellen von φ.

Theorem

MINSAT ist Π_2^p-vollständig.

Vollständigkeit

Weiteres natürliches vollständiges Problem z.B.:

Äquivalenzproblem für kontextfreie Grammatiken über
1-elementigen (Terminal-)Alphabeten ist Π_2^p-vollständig.

E s w i r d v e r m u t e t , Πd$_2^p$ -a sä s
i ä

Für Klassen weit oben in der polynomiellen Hierarchie scheint es
nur sehr wenig "natürliche" vollständige Probleme zu geben

Kapitel 5

Wie viele Klassen gibt es eigentlich?

Mehr Komplexitätsklassen

Abgesehen von den angegebenen Büchern:

http://qwiki.stanford.edu/wiki/Complexity_Zoo

http://www.math.ucdavis.edu/~greg/zoology/

Vollständigkeit für PH