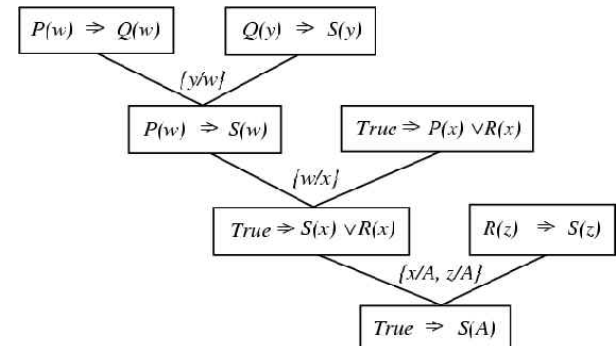


Beweissysteme

Hartmut Klauck
Universität Frankfurt
WS 06/07
13.12.



Runden

- Wir wollen nun zeigen, dass alle Probleme in $PCP(\text{poly}, \text{poly})$ ein $MIP(2)$ System mit nur einer Runde haben.
- **Theorem 9.1**
Sei L in $PCP(\text{poly}, \text{poly})$, dann gibt es ein MIP System für L mit 2 Beweisern, bei dem die Kommunikation in einer Runde verläuft
- Insbesondere gilt dies für TAUT

Beweis

- Wir werden später sehen, dass $PCP(\text{poly}, \text{poly}) = PCP(\text{poly}(n), O(1))$ gilt, wobei die Vollständigkeit des PCP mit $O(1)$ Fragen 1 ist.
- **Definition:** Ein PCP-Verifizierer ist nichtadaptiv, wenn die von ihm betrachteten Positionen des PCP-Beweises nur von der Eingabe (und den Zufallsbits des Verifizierers), nicht aber vom Inhalt des Beweises abhängen
- **Lemma 9.2**
Jeder PCP-Verifizierer mit q Fragen für Beweise über $\{0,1\}$ kann durch einen nichtadaptiven PCP-Verifizierer mit $q \cdot 2^q$ Fragen ersetzt werden
 - Beweis:
Es gibt nur 2^q mögliche Antwortsequenzen, für jede gibt es q Fragen. Frage einfach alle diese.
- Damit erhalten wir einen nichtadaptiven PCP-Verifizierer mit $O(1)$ Fragen und $\text{poly}(n)$ Zufall für alle Sprachen in NEXP

Beweis

- Wir folgen wieder dem Ansatz, beide MIP Beweiser als PCP zu betrachten, und dem zweiten nur eine Frage zu stellen
- Wir nehmen an, dass Vollständigkeit 1 und Korrektheit $1-1/(100t)$ sind
- „Ehrliche“ Beweiser P_1, P_2 antworten auf Fragen nach Position i mit $p(i)$
- Der MIP-Verifizierer simuliert den PCP-Verifizierer. Dabei stellt er die Fragen an P_1 , und zwar gleichzeitig
- P_1 kann nun mogeln, indem er adaptiv antwortet!
- Angenommen der PCP-Verifizierer stellt $t=O(1)$ Fragen an den PCP-Beweis
- Der MIP-Beweiser stellt t Fragen an P_1 und eine Frage an P_2 , und zwar eine zufällige der t Fragen
- Der MIP-Verifizierer verwirft, wenn
 - der PCP-Verifizierer verwerfen würde
 - P_1 und P_2 nicht übereinstimmen
- Sonst wird akzeptiert

Beweis

- Wieder wird für $x \in L$ der korrekte Beweis mit $Ws. 1$ akzeptiert
- Für x nicht $\in L$ ergeben die Antworten von P_2 einen string p , der mit $Ws. 1 - 1/(100t)$ verworfen wird (vom PCP-Verifizierer).
- Wenn P_1 nicht konsistent mit p ist, wird dies wie zuvor mit $Ws. 1/(3t)$ entdeckt.
- Wir erhalten ein Protokoll mit einer Runde, Korrektheit $1/(3t)$ und Vollständigkeit 1
- Wir würden nun gerne die Korrektheit erhöhen, aber dies würde die Anzahl der Runden wieder erhöhen
- Wir benötigen paralleles Boosten

Parallel Repetition Theorem

- **Theorem 9.3**

- Gegeben sei in MIP System mit 2 Beweisern und einer Runde, Vollständigkeit 1 und Korrektheit $1-\alpha$, und einer maximalen Länge m der Nachrichten der Beweiser

Wir wiederholen das System k mal:

- Der Verifizierer sendet k Fragen zugleich
- Die Beweiser senden k Antworten
- Der Verifizierer akzeptiert, wenn alle k Läufe zu akzeptieren sind

- Dann hat das neue System Vollständigkeit 1 und Korrektheit: $1-f(\alpha)^{k/m}$

- für eine Funktion $f(u)$, die für $u \rightarrow 0$ gegen eine Konstante c geht, und sich für $u=1-t$ wie $1-t^{O(1)}$ verhält

- Für uns ist $\alpha=1-1/(3t)$ und wir können mit $k=(3tmn)^{O(1)}=\text{poly}(n)$ die Korrektheit auf $1-1/2^n$ verbessern

Parallel Repetition Theorem

- Bemerkung: Damit können wir jedes MIP System mittels der Wahl $k = \text{poly}(n)$ auf exponentiell kleinen Fehler Boosten.
- Die (durch sequentielles Boosten erreichbare) Fehlerwahrscheinlichkeit α^k kann nicht erreicht werden
- Der Beweis ist sehr aufwendig

Bemerkung

- Wir haben zu Beginn verwendet, dass $MIP=NEXP=PCP(\text{poly},\text{poly})=PCP(\text{poly},O(1))$
- Unser Ziel ist es nun $NEXP=PCP(\text{poly},\text{poly})$ zu zeigen
- Dabei wird unser Beweissystem nichtadaptiv sein, und wie können auf dieselbe Art wie vorher zeigen, dass MIP mit einer Runde alle Sprachen in $NEXP$ beweist (wir haben nur die Nichtadaptivität verwendet, nicht, dass die Anzahl der Fragen nur konstant ist).

PCP vs. NEXP

- **Theorem 9.4**
 - $PCP(\text{poly}, \text{poly}) = NEXP$
- $PCP(\text{poly}, \text{poly}) \subseteq NEXP$ folgt mit 8.4
- Wir definieren für die andere Inklusion zunächst ein vollständiges Problem für NEXP und konstruieren für dieses Problem ein PCP mit $\text{poly}(n)$ Zufall und Fragen.

Ein NEXP-vollständiges Problem

- Betrachten wir den Beweis der NP-Vollständigkeit von SAT
- Wir können genauso für NEXP vorgehen
- Dabei erhalten wir zu einer Eingabe x für eine Sprache L aus NEXP eine KNF-Formeln mit exponentieller Länge und exponentiell vielen Variablen, die genau dann erfüllbar ist, wenn $x \in L$
- Wir benötigen nun eine „kompaktere“ Version des SAT-Problems
- Die einzelnen Klauseln der KNF sind leicht über die Turingmaschine für L und die Eingabe x zu beschreiben

Orakel SAT

- Wir bezeichnen durch Grossbuchstaben strings von polynomiell vielen Variablen
- B_1, B_2, B_3 seien strings von s Variablen
- Z sei ein string von r Variablen
- T habe 3 Variablen
- $\Phi(Z, B_1, B_2, B_3, T)$ sei eine Boolesche Formel in $r+3s+3$ Variablen
- Eine Boolesche Funktion $f: \{0,1\}^s \rightarrow \{0,1\}$ ist ein 3-erfüllendes Orakel für Φ , wenn für alle Belegungen der Variablen in Z, B_1, B_2, B_3, T gilt, dass $\Phi(Z, B_1, B_2, B_3, f(B_1), f(B_2), f(B_3))$ wahr ist.
- Eine Formel Φ heisst Orakel-erfüllbar, wenn ein solches f existiert
- Das Problem Orakel-3-SAT besteht darin, zu gegebenem Φ zu entscheiden, ob Φ Orakel-erfüllbar ist.

Orakel SAT

- **Theorem**
Orakel 3-SAT ist NEXP-vollständig.
- **Beweis:**
 - Offensichtlich liegt Orakel 3-SAT in NEXP
 - Es ist weiterhin eine (in polynomieller Zeit berechenbare) Reduktion von jeder Sprache L in NEXP auf Orakel 3-SAT zu zeigen.