Prof. T. Nipkow
M. Eberl, J. Rädle, S. Wimmer
Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2019 – Übungsblatt Lösungsskizze 12

AUFGABE 12.1. (*Wichtige Begriffe*) Stufe A

Überprüfen Sie, dass Sie die folgenden Begriffe korrekt definieren können.

- Polynomielle Reduktion $\leq_p$
- SAT
- 3-KNF-SAT

AUFGABE 12.2. (*Quiz Komplexität*) Stufe B

Nehmen Sie an, dass $\mathbf{P} \neq \mathbf{NP}$.

(a) Wenn $\mathsf{A} \leq_p$ 2-KNF-SAT, dann ist A **NP**-vollständig.
(b) Sei $\mathsf{B} \in \mathbf{P}$ und $\mathsf{A} \leq \mathsf{B}$. Dann ist $\mathsf{A} \in \mathbf{P}$.
(c) Sei $\overline{\mathsf{A}} \in \mathbf{P}$. Dann ist $\mathsf{A} \in \mathbf{NP}$. (Hinweis: $\overline{\mathsf{A}}$ ist das Komplement von A.)
(d) Es gilt: ($\mathsf{A} \leq_p$ SAT und 3COL $\leq_p \mathsf{A}$) gdw. A **NP**-vollständig.
(e) Sei der Graph $\mathcal{G} = (V, E)$ gegeben. Folgendes Problem ist **NP**-schwer:
`INPUT:` Eine natürliche Zahl k. `OUTPUT:` „Ja“, falls in $\mathcal{G}$ eine Rundreise der Länge $\leq k$ existiert, die alle Knoten genau einmal besucht, sonst „Nein“.

Lösungsskizze

(a) Falsch. Sei $\mathsf{A} =$ 2-KNF-SAT, dann gibt es die Reduktion trivialerweise (Identität), aber 2-KNF-SAT $\in \mathbf{P}$, deshalb kann 2-KNF-SAT unter der Annahme $\mathbf{P} \neq \mathbf{NP}$ nicht **NP**-schwer sein, und folglich auch nicht **NP**-vollständig.
(b) Falsch. $\leq$ sagt nichts über die Komplexität der Reduktion aus. Wähle z.B. $\mathsf{A} =$ 3-KNF-SAT und $\mathsf{B} = \{1\}$. Eine korrekte Reduktion kann zuerst berechnen ob eine gegebene Formel erfüllbar ist und dann entsprechend 0 oder 1 ausgeben.
(c) Wahr. $\overline{A} \in \mathbf{P}$, also $A \in \mathbf{P}$, da $\overline{A}$ in polynomieller Zeit zu entscheiden auch bedeutet, A in polynomieller Zeit entscheiden zu können. Da $\mathbf{P} \subseteq \mathbf{NP}$, gilt $A \in \mathbf{NP}$.
(d) Wahr. Nach VL SAT $\leq_p$ 3COL und somit SAT $\leq_p \mathsf{A} \leq_p$ SAT gdw. A **NP**-vollständig.
(e) Falsch. Entweder es existiert keine solche Rundreise, dann kann sofort 0 ausgegeben werden, oder es existiert eine kürzeste Rundreise der Länge n, dann ist die Antwort 1 gdw. $k \geq n$.

AUFGABE 12.3. (*SAT-Varianten*) Stufe C

Wir betrachten verschiedene Varianten von SAT, die auch NP-vollständig sind. Zeigen Sie diese NP-Vollständigkeit, indem Sie für jede Variante X eine Reduktion 3-KNF-SAT $\leq_p X$ angeben und $X \in$ NP zeigen.

(a) 3-OCC-KNF-SAT:
- Eingabe: Eine Formel F in KNF, bei der jede Variable höchstens dreimal auftritt.
- Frage: Ist F erfüllbar?

(b) Wir betrachten den `ITE`-Operator mit der Semantik $\texttt{ITE}(x, y, z) := (x \to y) \wedge (\neg x \to z)$. Eine `ITE`-Formel genügt der folgenden Grammatik:

$$F \to \texttt{ITE}(F, F, F) \mid x \mid \mathsf{true} \mid \mathsf{false} \qquad \text{für Variablen } x \in \mathcal{V}$$

`ITE`-SAT:
- Eingabe: Eine `ITE`-Formel F.
- Frage: Ist F erfüllbar?

(c) 1-Pro-Klausel-KNF-SAT:
- Eingabe: Eine Formel F in KNF.
- Frage: Gibt es eine erfüllende Belegung für F, so dass genau ein Literal pro Klausel auf *wahr* gesetzt wird?

Lösungsskizze

(a)
- **NP-schwer:** Sei F eine Formel in 3-KNF. Somit $F = \bigwedge_{i=1}^{n} K_i$ und $K_i = \bigvee_{j=1}^{m} L_{ij}$ mit $m \leq 3$. O.b.d.A. kommen keine Variablen doppelt in einer Klausel vor. Sei v die Anzahl der in F verwendeten Variablen. Dann ersetzten wir in jeder Klausel die Variablen mit für diese Klausel spezifische Variablen und fügen eine Bedingung ein, die erzwingt, dass alle Kopien einer Variable in der Eingabe den gleichen Wert zugewiesen bekommen. Diese Bedingung hat eine Ringstruktur, um Variablenverwendungen zu sparen: $(x_{k,1} \to x_{k,2}) \wedge (x_{k,2} \to x_{k,3}) \wedge \ldots (x_{k,n} \to x_{k,1})$.

$$f(F) = \bigwedge_{i=1}^{n} f(K_i) \wedge \bigwedge_{k=1}^{v} \bigwedge_{i=1}^{n} (\neg x_{k,i} \vee x_{k,(i \bmod n)+1})$$
$$f(K_i) = \bigvee_{j=1}^{m} f(L_{ij})$$
$$f(L_{ij}) = \begin{cases} x_{k,i} & \text{falls } L_{ij} = x_k \\ \neg x_{k,i} & \text{falls } L_{ij} = \neg x_k \end{cases}$$

  Diese Reduktion ist in polynomieller Zeit berechenbar, da nur die Eingabe einmal kopiert werden muss und ein zusätzlicher Term der Größe $m \cdot v$ erzeugt wird.
  Korrektheit: Aus jeder erfüllenden Belegung σ für F können wir eine erfüllende Belegung σ' für $f(F)$ konstruieren mit $\sigma'(x_{k,i}) = \sigma(x_k)$. Auch können wir aus jeder erfüllenden Belegung σ für $f(F)$ ein Belegung für F konstruieren. Da $\sigma(x_{k,i}) = \sigma(x_{k,j})$ für alle i, j gelten muss, können wir einfach σ' über $\sigma'(x_k) = \sigma(x_{k,1})$ definieren. Somit ist die Konstruktion erfüllbarkeitserhaltend und die Reduktion korrekt.
- $\in$ **NP**: Eine erfüllende Belegung ist ein geeignetes Zertifikat, das es maximal genauso groß ist wie die Formel F und in polynomieller Zeit geprüft werden kann.

(b)
- **NP-schwer:** Sei F eine Formel in 3-KNF. Wir wandeln F in polynomieller Zeit unter der Verwendung der folgenden semantischen Äquivalenzen in eine ITE-Formel:

$$\neg x \equiv \texttt{ITE}(x, \mathsf{false}, \mathsf{true}) \qquad x \wedge y \equiv \texttt{ITE}(x, y, \mathsf{false}) \qquad x \vee y \equiv \texttt{ITE}(x, \mathsf{true}, y)$$

  Eine Klausel mit drei Literalen, z.B. $\neg x \vee \neg y \vee \neg z$ lässt sich damit umformen zu

$$\neg x \vee \neg y \vee \neg z \equiv \texttt{ITE}(\texttt{ITE}(x, \mathsf{false}, \mathsf{true}), \mathsf{true}, \texttt{ITE}(\texttt{ITE}(y, \mathsf{false}, \mathsf{true}), \mathsf{true}, \texttt{ITE}(z, \mathsf{false}, \mathsf{true})))$$

  Jede Klausel wird in eine ITE-Formel übersetzt, welche nur um einen konstanten Faktor größer ist. Entsprechend übersetzt sich jede 3KNF-Formel in eine semantisch äquivalente ITE-Formel, die nur um einen konstanten Faktor größer ist. Die Korrektheit folgt aus der Verwendung semantisch-äquivalenten Umformungen.
- $\in$ **NP**: Eine erfüllende Belegung ist ein geeignetes Zertifikat, das es maximal genauso groß ist wie die Formel F und in polynomieller Zeit geprüft werden kann.

(c) Eine ausführliche Lösung finden Sie auf: `http://cs.nyu.edu/courses/fall14/CSCI-UA.0310-001/1in3sat.pdf`

AUFGABE 12.4. (*Nullstellen Problem*) Stufe C

Wir betrachten das NULLSTELLEN-Problem:

Gegeben Ein Polynom p über Variablen $x_1, \ldots, x_n$ mit ganzzahligen Koeffizienten.

Problem Gibt es eine ganzzahlige Nullstelle, also $v_1, \ldots, v_n \in \mathsf{Z}$ mit $p(v_1, \ldots, v_n) = 0$?

(a) Zeigen Sie, dass das NULLSTELLEN-Problem NP-hart ist, indem Sie eine Reduktion von SAT angeben. Hinweis: Die folgenden Beziehungen könnten dabei nützlich sein:

$$f(x)g(x) = 0 \iff f(x) = 0 \vee g(x) = 0$$
$$f(x)^2 + g(x)^2 = 0 \iff f(x) = 0 \wedge g(x) = 0$$

(b) Wo liegt das Problem bei folgendem "Beweis", dass NULLSTELLEN in NP ist:

> Ein Zertifikat ist genau eine Nullstelle von p, also ein Vektor von Zahlen $v_1, \ldots, v_n \in \mathsf{Z}$. Ein Verifikator muss nun lediglich überprüfen, ob $p(v_1, \ldots, v_n) = 0$. Nach Satz 5.10 ist das Problem damit in NP.

Lösungsskizze

(a) Ausgehend von einer Booleschen Formel F konstruieren wir ein Polynom, welches genau dann eine Nullstelle hat, wenn F erfüllbar ist. Wie man $\wedge$ und $\vee$ mit Polynomen darstellt, ist schon vorgegeben, es fehlt lediglich die Negation. Wir können aber annehmen, dass die Negation in F lediglich auf Variablen vorkommt, da man jede Formel mit den Gesetzen von deMorgan leicht in diese Form bringen kann.
Aus jeder aussagenlogischen Variablen x_i wird nun eine ganzzahlige Variable X_i im Polynom. X_i soll genau dann null sein, wenn x_i false ist.
Wir führen einen Term ein, der dafür sorgt, dass jede Variable nur den Wert 0 oder 1 haben kann:

$$\sum_{1 \leq i \leq n} (X_i(1 - X_i))^2$$

Der Rest des Polynoms hat nun dieselbe Struktur wie die Formel. Wir stellen ein positives Literal durch $(1 - X_i)$ dar und ein negatives durch X_i. Eine Disjunktion wird zu einem Produkt, und die Konjunktion zu einer Summe der Quadrate.
Beispielsweise wird die Formel

$$(x_1 \vee \overline{x_2}) \wedge (x_2 \vee x_3)$$

durch das Polynom

$$\sum_{1 \leq i \leq 3} (X_i(1 - X_i))^2 + ((1 - X_1)X_2)^2 + ((1 - X_2)(1 - X_3))^2$$

dargestellt. Die Nullstellen des Polynoms entsprechen nun genau den erfüllenden Belegungen der Formel. Offensichtlich ist die Reduktion polynomiell.

(b) Das Problem ist, dass die ganzzahlige Nullstelle beliebig groß sein kann, und nicht polynomiell beschränkt in der Größe der Eingabe (des Polynoms). Damit kann der Verifikator nicht in Polynomialzeit laufen.
Anmerkung: Tatsächlich ist NULLSTELLEN nicht in NP, denn es handelt sich hier um Hilberts 10. Problem (vgl. Satz 4.68), welches unentscheidbar ist.

AUFGABE 12.5. (*co-NP*) Stufe E

Auf dem vorherigen Übungsblatt haben wir uns mit der Komplexitätsklasse co-NP beschäftigt:

$$\text{co-NP} = \{\mathsf{L} \mid \overline{\mathsf{L}} \in \text{NP}\}$$

Dort haben wir gezeigt dass $\mathsf{L} \in$ co-NP gdw. es eine DTM M und ein Polynom p gibt, so dass
$\{w \in \Sigma^* \mid \forall c \in \Delta^{p(|w|)}.\ w\#c \in \mathsf{L}(\mathsf{M})\} = \mathsf{L}$ und $\mathsf{time}_{\mathsf{M}}(w\#c) \leq p(|w|)$.
In dieser Aufgabe untersuchen wir die Beziehungen zwischen P, NP und co-NP.

(a) Zeigen Sie $\text{P} \subseteq \text{NP} \cap \text{co-NP}$
(b) Zeigen Sie unter der Annahme $\text{P} = \text{NP}$, dass $\text{NP} = \text{co-NP}$ gilt.
(c) Angenommen es gäbe eine co-NP vollständige Sprache L mit $\mathsf{L} \in$ NP. Zeigen Sie dass dann $\text{NP} = \text{co-NP}$ gilt

Lösungsskizze

(a) Da P unter Komplement abgeschlossen ist gilt $\text{co-P} = \text{P}$. Mit $\text{co-P} \subseteq \text{co-NP}$ folgt $\text{P} \subseteq \text{NP} \cap \text{co-NP}$.
(b) Hier verwenden wir wiederum die Abgeschlossenheit von P unter Komplement. Wenn $\text{P} = \text{NP}$, muss auch NP unter Komplement abgeschlossen sein, woraus direkt $\text{NP} = \text{co-NP}$ folgt.
(c) Wir zeigen $\text{NP} \subseteq \text{co-NP}$:
Wegen $\mathsf{L} \in \text{NP}$ gilt $\overline{\mathsf{L}} \in \text{co-NP}$. Außerdem ist $\overline{\mathsf{L}}$ mit dem Ergebnis aus Aufgabe 11.6 (a) NP-vollständig. Für ein beliebiges $\mathsf{L}' \in \text{NP}$ gilt daher $\mathsf{L}' \leq_p \overline{\mathsf{L}}$. Da $\overline{\mathsf{L}} \in \text{co-NP}$ folgt $\mathsf{L}' \in \text{co-NP}$ und damit die Behauptung.

Für die Gegenrichtung $\text{co-NP} \subseteq \text{NP}$ zeigen wir dass für ein beliebiges $\mathsf{L}' \in \text{co-NP}$ gilt:

$$\mathsf{L}' \in \text{co-NP} \Longrightarrow \overline{\mathsf{L}'} \in \text{NP} \Longrightarrow \overline{\mathsf{L}'} \in \text{co-NP} \Longrightarrow \mathsf{L}' \in \text{NP}$$

Hierbei folgt die zweite Implikation aus dem Ergebnis $\text{NP} \subseteq \text{co-NP}$; die anderen Schritte folgen aus der Definition von co-NP.