Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

Übungsblatt 5

Vorlesung Theoretische Grundlagen der Informatik im WS 16/17

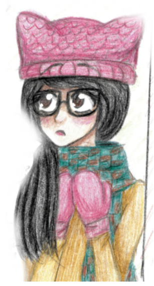
Ausgabe 8. Dezember 2016
Abgabe 20. Dezember 2016, 11:00 Uhr (im Kasten im UG von Gebäude 50.34)

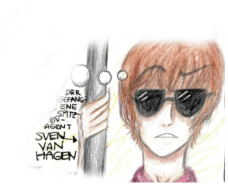
Bitte nutzen Sie den WebInScribe Deckblattgenerator
und heften Sie das Deckblatt an Ihr Übungsblatt.
`https://webinscribe.ira.uka.de/deckblatt/index.php?course=10588`.

Aufgabe 1

(2 + 3 = 5 Punkte)

Der ebenso geniale, wie auch frustrierte Wissenschaftler und Superbösewicht Doktor Meta ist in Geldnot. Es ist zum aus der Haut fahren! Sven van Hagen befindet sich in seiner Gewalt, Turing-Man ist in einer Endlosschleife gefangen und die schöne Elsa unterstützt Metas Pläne. Doch nach dem Bau seines neuen Hauptquartiers ist der Doktor finanziell in Schwierigkeiten. Schon seit langem hofft Doktor Meta deswegen auf die Unterstützung der Allianz für Diktatoren, die unter Superbösewichten sehr bekannt ist. Dort wird eine Liste von Bösewichten geführt, die einer (sehr größzügigen) Spende würdig sind. Offensichtlich steht Metas Name nicht darauf. Um sich in die Liste eintragen zu lassen muss man dieses Jahr allerdings *nur* ein dort noch nicht bekanntes co-$\mathcal{NP}$-Vollständiges Problem einschicken. Das ist seine Chance. Elsa erinnert sich, dass das Problem $\overline{\text{SAT}}$ co-$\mathcal{NP}$-vollständig ist.
Das Problem $\overline{\text{SAT}}$ ist folgendermaßen definiert:



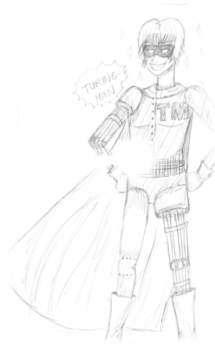


> **Gegeben:** Menge U von Variablen, Menge C von Klauseln.
>
> **Frage:** Existiert *keine* Wahrheitsbelegung von U, so dass C erfüllt wird? Das heißt, nimmt für jede Wahrheitsbelegung von U mindestens eine Klausel aus C den Wahrheitswert `falsch` an?

Da Doktor Meta aufgrund der Bekanntheit von SAT befürchtet, dass $\overline{\text{SAT}}$ bereits bekannt ist, denkt er sich ein neues Problem aus. Die Sprache L_{reg} ist über einem endlichen Alphabet Σ folgendermaßen definiert:

$$L_{reg} = \{(R, S) \mid R, S \text{ sind reguläre Ausdrücke mit } L(R) = L(S)\}$$

Helfen Sie Dr. Meta und beweisen Sie, dass L_{reg} co-$\mathcal{NP}$-vollständig ist, indem Sie wie folgt vorgehen:



(a) Zeigen Sie $L_{reg} \in$ co-$\mathcal{NP}$. Sie dürfen davon ausgehen, dass ein Algorithmus existiert, der zu einem regulären Ausdruck R der Länge m und einem Wort w der Länge n in $\mathcal{O}(mn)$ Zeit entscheidet, ob $w \in L(R)$.

(b) Zeigen Sie, dass L_{reg} co-$\mathcal{NP}$-schwer ist, indem Sie $\overline{\text{SAT}} \propto L_{reg}$ zeigen. *Hinweis*: Wählen Sie $S = \Sigma^\star$ fest.

Aufgabe 2

(3 + 1 = 4 Punkte)

Zeigen Sie, dass aus $\mathcal{P} = \mathcal{NP}$ folgt, dass alle Sprachen in $\mathcal{P}$, bis auf $\emptyset$ und $\Sigma^\star$, $\mathcal{NP}$-vollständig sind. Wieso gilt die Aussage nicht für $\emptyset$ und $\Sigma^\star$?

Aufgabe 3

(3 + 3 = 6 Punkte)

Sei L eine Sprache in $\mathcal{NP}$ über einem endlichen Alphabet Σ, d.h. es existiert eine deterministische Turingmaschine $\mathcal{M}$, so dass gilt

$$w \in L \iff \exists x \in \Sigma^\star : \mathcal{M} \text{ akzeptiert } \langle w, x \rangle,$$

wobei die Laufzeit von $\mathcal{M}$ polynomial in $n = |w|$ ist.

Hinweis: dies ist die „Zeugendefinition“ der Klasse $\mathcal{NP}$, da x die Zugehörigkeit von w zu L bezeugt.

(a) Beweisen Sie, dass jede Sprache $L \in \mathcal{NP}$ von einer deterministischen Turingmaschine in $\mathcal{O}(2^{p(n)})$ entschieden werden kann, wobei $p(n)$ ein Polynom über n ist.

(b) Nehmen Sie nun an, dass sie für eine $\mathcal{NP}$-vollständige Sprache $\mathcal{L}$ die Turingmaschine $\mathcal{M}$ so modifizieren können, dass $\mathcal{M}$ für jedes $w \in L$ höchstens 42 Zeugen x ablehnt. Zeigen Sie, dass dann $\mathcal{P} = \mathcal{NP}$ gilt.

Aufgabe 4

(2 + 2 + 3 + 1 + 1 = 9 Punkte)

Das Problem 3-Sum ist wie folgt definiert.

> **Gegeben:** Eine endliche Menge $S \subset \mathbb{R}$.
>
> **Frage:** Existieren drei nicht notwendigerweise unterschiedliche Zahlen $a, b, c \in S$, so dass $a + b + c = 0$?

Es ist offen, ob ein Algorithmus existiert der das Problem 3-Sum in der Laufzeit $O(n^{2-\epsilon}), \epsilon > 0$ löst. Ein Problem Π heißt 3-Sum-*schwer*, wenn ein subquadratischer Algorithmus für Π impliziert, dass 3-Sum in subquadratischer Zeit lösbar ist.

Das Problem 3-Linienschnitt ist wie folgt definiert:

Gegeben: Eine endliche Menge $\mathcal{L}$ von Linien in der euklidischen Ebene.

Frage: Existieren drei Linien aus $\mathcal{L}$ die sich in genau einem Punkt schneiden?

Im Folgenden sollen Sie zeigen, dass 3-LINIENSCHNITT 3-SUM-schwer ist. Gehen Sie dafür davon aus, das Sie eine Linie $y = mx + b$ als Paar (m, b) kodieren. Sie wissen außerdem, dass die Steigung m nur die Werte $\{-1, 0, 1\}$ annehmen kann.

(a) Geben Sie für jede Zahl $x \in S$ eine Menge von Linien $\mathcal{L}_x$ an, so dass für drei Zahlen $a+b+c = 0$ genau dann gilt, wenn drei Linien $l_a \in L_a$, $l_b \in L_b$ und $l_c \in L_c$ existieren, die sich in genau einem Punkt schneiden.

(b) Konstruieren Sie zu der Instanz $S = \{1, 2, -3, -4\}$ die Mengen $L_a, a \in S$ graphisch. Kennzeichnen Sie die Mengen L_a eindeutig. Geben Sie zwei Lösungen der Form $\{(m_1, b_1), (m_2, b_2), (m_3, b_3)\}$ für die 3-LINIENSCHNITT Instanz an. Entspricht Ihre Lösung der 3-LINIENSCHNITT Instanz einer Lösung der 3-SUM Instanz?

(c) Zeigen Sie die Korrektheit Ihrer Transformation aus (a).

(d) Benötigt Ihre Transformation subquadratische Zeit?

(e) Der sogenannte Sweep-Line Algorithmus zum Schnitt von Linien benötigt $O(n \log n + k)$ Laufzeit, wobei $k \in \mathbb{N}$ die Anzahl der Schnittpunkte ist. Warum impliziert dies noch keinen subquadratischen Algorithmus für das 3-SUM Problem? Welche Voraussetzungen muss die Transformation von S nach $\mathcal{L}$ erfüllen, damit Sie mit Hilfe des Sweep-Line Algorithmus einen subquadratischen Algorithmus für 3-SUM erhalten?

Aufgabe 5

(3 + 3 + 3 = 9 Punkte)

Beweisen Sie die $\mathcal{NP}$-vollständigkeit der folgenden Probleme.

- **Gegeben:** Zwei Graphen $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ und eine Zahl $K \in \mathbb{N}$.

  Frage: Gibt es zwei Teilmengen $E_1' \subset E_1, E_2' \subset E_2$, mit $|E_1'| = |E_2'| \geq K$, so dass die Graphen $G_1' = (V(E_1'), E_1'), G_2' = (V(E_2'), E_2')$ isomorph sind?

  Nutzen Sie, dass das Problem HAMILTONISCHER KREIS $\mathcal{NP}$-vollständig ist.

- **Gegeben:** Ein gerichteter Graph $G = (V, E)$, eine positive Zahl $K \in \mathbb{N}$.

  Frage: Gibt es eine Teilmenge $V' \subset V$, mit $|V'| \leq K$, so dass für jeden gerichteten Zyklus C in G, $V(C) \cap V' \neq \emptyset$ gilt.

  Nutzen Sie das Problem VERTEX COVER für die Reduktion.

  Hinweis: $V(C)$ bezeichnet die Knotenmenge des Zyklus C.

- **Gegeben:** Eine Menge S und eine Menge $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ von Teilmengen A_i von S.

  Frage: Gibt es eine Partition $S_0 \dot\cup S_1 = S$, so dass kein A_i vollständig in S_0 oder S_1 enthalten ist?

  Nutzen Sie das Problem 3-SAT zur Reduktion.