Informatik III:
Theoretische Informatik

Prof. Bernhard Nebel
Dr. Stefan Wölfl

Wintersemester 2011/12
Universität Freiburg
Institut für Informatik

Übungsblatt 13

Abgabe: 13. Februar 2012

Hinweis: Anstelle der Vorlesung findet am 6. Februar eine Probeklausur statt.

Aufgabe 13.1 (Universelle Turingmaschine; 2+2+2 Punkte)

Eine Turingmaschine U mit Eingabealphabet $\{0, 1, \#\}$ heißt *universell*, falls U angesetzt auf ein beliebiges Wort $w\#x \in \{0, 1, \#\}^*$ genau dasselbe Wort berechnet wie M_w angesetzt auf das Wort x.

(a) Beschreiben Sie die Arbeitsweise einer universellen Turingmaschine in Form eines Algorithmus. Erläutern Sie kurz, warum die von Ihnen beschriebene Maschine universell ist.

(b) Die *universelle Sprache* L_U ist definiert als die Sprache

$$L_U = \{\, w\#v \in \{0, 1, \#\}^* \;:\; M_w \text{ akzeptiert } v\}.$$

Beweisen Sie, dass L_U rekursiv aufzählbar ist, indem Sie zeigen, dass die von einer universellen Turing-Maschine akzeptierte Sprache gerade die universelle Sprache L_U ist.

(c) Ist L_U entscheidbar? Beweisen Sie Ihre Behauptung.

Aufgabe 13.2 (Entscheidungsprobleme für Graphen; 3 + 3 Punkte)

Ein *ungerichteter* (einfacher) Graph $G = \langle V, E\rangle$ ist gegeben durch eine nicht leere Menge V (*Knoten*) und einer Menge E von zweielementigen Teilmengen von V (*Kanten*). Ein *Pfad* (der Länge k) ist eine Folge von Kanten $\{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{k-1}, v_k\} \in E$. Man bezeichnet dann v_0 als Start- und v_k als Endknoten des Pfades. Ein Graph heißt *zusammenhängend*, falls zwischen je zwei verschiedenen Knoten ein Pfad in G existiert. Eine *Clique* ist eine Teilmenge $C \subseteq V$, derart dass für je zwei verschiedene $v, v' \in C$ gilt $\{v, v'\} \in E$.

Wir betrachten die folgenden beiden Entscheidungsprobleme für Graphen.

Zusammenhang: Gegeben ein ungerichteter Graph $G = \langle V, E\rangle$. Ist G zusammenhängend?

Clique: Gegeben ein ungerichteter Graph $G = \langle V, E\rangle$ sowie $k \in \mathbb{N}$. Besitzt G eine Clique C mit $|C| \geq k$?

Geben Sie für jedes Problem eine geeignete Codierung an. Formulieren Sie jeweils einen deterministischen Algorithmus, der das Problem löst, und geben Sie eine obere Abschätzung der Laufzeit Ihres Algorithmus in Abhängigkeit von der Eingabe an. Welche Probleme sind in P, welche in NP (Begründung)?

Bonusaufgabe 13.3 (Alternative Definition von NP; 4 Punkte)

Zeigen Sie: Eine Sprache $L \subseteq \Sigma^*$ ist in NP genau dann, wenn es eine Sprache $L' \in \mathsf{P}$ über einem Alphabet $\Sigma' \supseteq \Sigma \cup \{\#\}$ sowie ein Polynom $p(n)$ gibt, derart dass gilt:

$$L = \{\, x \in \Sigma^* \,:\, \text{es gibt ein } z \in \Sigma'^* \text{ mit } |z| \leq p(|x|) \text{ und } x\#z \in L'\}.$$

Hinweis: Diese Definition formuliert man häufig so: Unter einem *Verifizierer* für eine Sprache $L \subseteq \Sigma^*$ versteht man eine deterministische Turingmaschine M mit Eingabealphabet Σ', für die gilt:

$$L = \{\, x \in \Sigma^* \,:\, M \text{ akzeptiert } x\#z, \text{ für ein } z \in \Sigma'^*\}.$$

In dieser Situation bezeichnet man das Wort z als ein *Zertifikat* für x. Die Aussage obiger Behauptung ist also, dass eine Sprache L genau dann in NP ist, falls es einen polynomiellen Verifizierer für L gibt (hier: eine Turingmaschine, die L' akzeptiert).

Bonusaufgabe 13.4 (Reduktion II; 3 Punkte)

Zeigen Sie, dass das Problem zu entscheiden, ob zwei natürliche Zahlen teilerfremd sind, reduzierbar ist auf das Problem für natürliche Zahlen a, b, c mit $c > 0$ zu entscheiden, ob es ganze Zahlen x und y mit $ax + yb = c$ gibt.

Hinweis: Zwei natürliche Zahlen sind teilerfremd, wenn 1 der einzige gemeinsame Teiler (> 0) beider Zahlen ist.

Bonusaufgabe 13.5 (Reduktion III; 4 Punkte)

Das Ableitungsproblem für Typ-0 Grammatiken DER_0 ist wie folgt definiert: Gegeben eine Typ-0 Grammatik $G = \langle V, \Sigma, P, S\rangle$ und Wörter $u, v \in (V \cup \Sigma)^*$, gibt es eine Ableitung von v aus u (symbolisch: $u \Rightarrow^* v$)?

Zeigen Sie, dass das Problem DER_0 auf das modifizierte Postsche Korrespondenzproblem reduzierbar ist.

Hinweis: Konstruieren Sie aus einer DER_0-Instanz eine MPCP-Instanz über einem Alphabet $\Sigma' := V \cup \Sigma \cup \{\#\}$, das einen "Spielstein" (p, q) für jede Produktionsregel $p \to q \in P$, "Kopiersteine" (x, x) für $x \in \Sigma'$, sowie spezielle Anfangs- und Endsteine hat.