technische universität
dortmund

Beate Bollig, Marc Gillé, Jens Lechner
Lehrstuhl 2, Fakultät für Informatik
Sommersemester 2011

TIfAI Übung – Blatt 7

Ausgabedatum: 17.5.2011 — Abgabedatum: 23.5.2011, 14:00 Uhr

Aufgabe 7.1: Reduktionen

Kurzaufgabe (1 Punkt):

Welchen Reduktionsbegriff verwenden wir in der Entscheidbarkeitstheorie? Warum benutzen wir hier weder Turing-Reduktionen noch polynomielle Reduktionen?

Hauptaufgabe (4 Punkte):

Beweise oder widerlege die folgenden Behauptungen:

1. $H_\varepsilon \leq$ SAT
2. 2-SAT $\leq_p$ CLIQUE
3. TSP $\leq_p$ U
4. HC $\leq$ 2-SAT
5. $\emptyset \leq_p H$
6. D $\leq_T$ 2-SAT
7. 2-SAT $\leq \Sigma^*$

Aufgabe 7.2: Diagonalisierung

Kurzaufgabe (1 Punkt):

Erkläre kurz das Prinzip der Diagonalisierung anhand des Beweises, dass die Diagonalsprache nicht rekursiv ist.

Hauptaufgabe (4 Punkte):

Wir wollen jedem Element aus $\{0,1\}^*$ eine deterministische Turingmaschine zuordnen. Aus der Vorlesung ist bekannt, dass wir mit Hilfe von Gödelnummern Turingmaschinen mit Zeichenketten aus $\{0,1\}^*$ codieren können. Wir modifizieren die Codierung der Turingmaschine mit Gödelnummern, so dass für eine Turingmaschine M unendlich viele Codierungen existieren, indem wir beliebig viele Einsen am Ende der Gödelnummer hinzufügen dürfen. Trotzdem codiert nicht jedes Element aus $\{0,1\}^*$ eine Turingmaschine. Dieses Problem beheben wir, indem wir für alle $x \in \{0,1\}^*$, die keine gültige Codierung für eine Turingmaschine darstellen, festlegen, dass diese Elemente die Turingmaschine codieren, die nur aus einem Zustand besteht, sich nicht bewegt und direkt verwirft. Somit codiert nun jedes Element aus $x \in \{0,1\}^*$ eine deterministische Turingmaschine M_x.

Wir können annehmen, dass wir bei gegebenem x M_x effizient simulieren können, d.h. eine andere Turingmaschine kann jeden Schritt der Turingmaschine M_x in Zeit $O(q(n))$ simulieren, wobei n die Eingabelänge und $q(n)$ ein Polynom in n ist.

Betrachte die folgende Turingmaschine D, die für eine Eingabe $x \in \{0,1\}^*$ wie folgt arbeitet:

1. Simuliere M_x auf Eingabe x für höchstens $2^{|x|}$ Schritte:
2. Falls M_x innerhalb dieser Zeit die Eingabe akzeptiert oder verwirft, gib das Gegenteil von M_x aus. Sonst verwirf.

Sei $L(D)$ die Sprache, die diese Turingmaschine entscheidet.

1. Zeige, dass $L(D) \in \text{EXP}$, d.h. die Laufzeit ist $O(2^{p(n)})$ für ein Polynom p.
2. Zeige, dass es keine polynomielle deterministische Turingmaschine gibt, die die Sprache $L(D)$ entscheidet.

   Hinweis: Nimm an das es eine polynomielle Turingmaschine M für $L(D)$ gibt und zeige $L(M) \neq L(D)$, wobei $L(M)$ die Sprache ist, die von M entschieden wird.

Aufgabe 7.3: Berechenbarkeit

Kurzaufgabe (1 Punkt):

Erkläre die Begriffe *rekursiv* und *rekursiv aufzählbar* und beschreibe in eigenen Worten, was der Satz von Rice aussagt.

Hauptaufgabe (4 Punkte):

1. Zeige für jede folgende Sprache, ob sie rekursiv, rekursiv aufzählbar oder nicht rekursiv ist:
   (a) $L_1 = \{< M >\mid M \text{ berechnet eine total rekursive Funktion}\}$.
   (b) $L_2 = \{w_1\#w_2 \mid w_1 \in L' \text{ und } w_2 \in L''\}$, wobei $L', L'' \subseteq \{0,1\}^*$ rekursiv aufzählbar sind.
   (c) Eine beliebige Sprache L_3 aus PSPACE.
2. Zeige, dass eine Sprache L genau dann rekursiv ist, wenn es eine Turingmaschine gibt, die auf einem Ausgabeband alle Wörter $w \in L$, jeweils durch ein Sonderzeichen # getrennt, in kanonischer Reihenfolge ausgibt.

Testfragen:

1. Wie ist der Aufbau der Gödelnummer einer Turingmaschine?
2. Was ist eine universelle Turingmaschine?
3. Wie funktioniert der Beweis des Satzes von Rice?