Technische Universität München
Fakultät für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Werner Meixner, Alexander Krauss

Sommersemester 2009
Lösungsblatt Endklausur
7. August 2009

Einführung in die Theoretische Informatik

Name	Vorname	Studiengang	Matrikelnummer
....	....	☐ Diplom ☐ Inform. ☐ Bachelor ☐ BioInf. ☐ Lehramt ☐ Mathe.	....

Hörsaal	Reihe	Sitzplatz	Unterschrift
....	....	....	....

Code:

Allgemeine Hinweise

- Bitte füllen Sie obige Felder in Druckbuchstaben aus und unterschreiben Sie!
- Bitte schreiben Sie nicht mit Bleistift oder in roter/grüner Farbe!
- Die Arbeitszeit beträgt 120 Minuten.
- Alle Antworten sind in die geheftete Angabe auf den jeweiligen Seiten (bzw. Rückseiten) der betreffenden Aufgaben einzutragen. Auf dem Schmierblattbogen können Sie Nebenrechnungen machen. Der Schmierblattbogen muss ebenfalls abgegeben werden, wird aber in der Regel nicht bewertet.
- Es sind keine Hilfsmittel außer einem handbeschriebenen DIN-A4-Blatt zugelassen.

Hörsaal verlassen von ...... bis ...... / von ...... bis ......
Vorzeitig abgegeben um ......
Besondere Bemerkungen:

	A1	A2	A3	A4	A5	Σ	Korrektor
Erstkorrektur							
Zweitkorrektur							

Aufgabe 1 (10 Punkte)

Wahr oder falsch? Begründen Sie im Folgenden Ihre Antworten möglichst knapp!

1. Das PCP $((01, 0), (10, 01), (0, 01))$ besitzt eine Lösung.
2. Das Halteproblem auf leerem Band H_0 ist NP-hart.
3. Wenn f berechenbar ist, dann ist $A_f := \{w \in \Sigma^* \mid f(w) \neq \perp\}$ semi-entscheidbar.
4. Wenn A NP-vollständig ist, dann ist χ_A μ-rekursiv.
5. Für das spezielle Halteproblem $K = \{w \in \{0, 1\}^* \mid M_w[w] \downarrow\}$ und eine beliebige Sprache A gilt: Wenn $K \cap A$ entscheidbar ist, dann ist A endlich.
6. Jedes Problem ist entweder in P oder in NP
7. Das Postsche Korrespondenzproblem ist semi-entscheidbar.
8. Für jede Turingmaschine M ist die Funktion
$$f_M(x) = \begin{cases} 1 & \text{falls } M \text{ auf allen Eingaben hält} \\ 0 & \text{sonst} \end{cases}$$
berechenbar.
9. Wenn f und g primitiv rekursiv sind, und $f(x) = g(h(x))$, dann ist auch h primitiv rekursiv.
10. Sei $A \subseteq \Sigma^*$. Wenn χ_A total ist, dann ist A entscheidbar.

Lösungsvorschlag

1. (f): Keine Berechnung kann mit einem der Paare enden.
2. (w): Es gibt eine Turingmaschine M, die das SAT Problem entscheidet und insbesondere stets hält. Falls M in einem nichtakzeptierenden Zustand hält, dann kann man eine nicht haltende Turingmaschine starten. Damit wird SAT polynomiell auf H_0 reduziert.
3. (w): $\chi'_{A_f}(x) = \begin{cases} 1 & \text{falls } f(x) \neq \perp \\ \perp & \text{sonst} \end{cases}$ ist berechenbar, da f berechenbar ist.
4. (w): $A \in NP \implies A$ entscheidbar $\implies \chi_A$ berechenb. $\implies \chi_A$ μ-rek.
5. (f): Gegenbsp: $A = \overline{K}$. Dann ist $K \cap A =$.
6. (f): (z.B.) unentscheidbare Mengen liegen weder in P noch NP.
7. (w): Systematisch alle Kombinationen aufzählen und prüfen.

8. (w): Für jede TM M ist f_M eine konstante Funktion.

9. (f): Ggbsp: $f(x) = 0$, $g(x) = 0$, h beliebig und nicht PR.

10. (f): χ_A ist nach Def. immer total. Entscheidend ist, dass sie *berechenbar* ist.

Richtige Antwort: 0,5 Punkte
Begründung auch richtig/sinnvoll: 0,5 Punkte

Aufgabe 2 (8 Punkte)

Gegeben sei der Kellerautomat $K=(\{q\},\Sigma,\Gamma,q,Z_0,\delta)$ mit $\Sigma=\{a,b,\#\}$, $\Gamma=\{Z_0,X,Y,Z\}$ und der Übergangsfunktion

$$\begin{array}{lclclcl} \delta(q,\epsilon,Z_0) & = & \{(q,XZ)\}\,, & \quad & \delta(q,a,X) & = & \{(q,XY)\}\,, \\ \delta(q,\#,X) & = & \{(q,\epsilon)\}\,, & & \delta(q,b,Y) & = & \{(q,\epsilon)\}\,, \\ \delta(q,a,Z) & = & \{(q,\epsilon)\}\,. & & & & \end{array}$$

1. Geben Sie eine Berechnung als Konfigurationsfolge an, die zeigt, dass K das Wort $a\#ba$ mit leerem Keller akzeptiert, d. h., dass $a\#ba \in L_\epsilon(K)$ gilt.

2. Modifizieren Sie die Übergangsfunktion δ so zu einer Funktion δ', dass der Kellerautomat
$$K' = (\{q\},\Sigma,\Gamma,q,Z_0,\delta')$$
die Sprache $L_\epsilon(K)^*$ mit leerem Keller akzeptiert.

3. Leiten Sie eine Grammatik G her, die $L_\epsilon(K)$ erzeugt. Die Korrektheit von G muss durch systematische Anwendung einer geeigneten Methode sichergestellt werden.

Lösungsvorschlag

1. $$\begin{array}{lr} (q,a\#ba,Z_0) \rightarrow_K (q,a\#ba,XZ) & (\tfrac{1}{2}\text{ P.}) \\ \rightarrow_K (q,\#ba,XYZ) & (\tfrac{1}{2}\text{ P.}) \\ \rightarrow_K (q,ba,YZ) \rightarrow_K (q,a,Z) \rightarrow_K (q,\epsilon,\epsilon) & (1\text{ P.}) \end{array}$$

2. Hinzufügen: $\delta(q,\epsilon,Z_0)\cup\{(q,\epsilon)\}$ und $\delta(q,a,Z)\cup\{(q,Z_0)\}$. (2 P.)

3. Anwendung der Methode aus der Vorlesung (erforderlich!), allerdings können in der Notation alle Tripel $[p,A,q]$ vereinfacht werden zu A, weil es nur einen Zustand gibt.

$$\begin{array}{lllllllllr} S & \rightarrow & Z_0\,, & & & & & & & (\tfrac{1}{2}\text{ P.}) \\ Z_0 & \rightarrow & XZ\,, & X & \rightarrow & aXY\,, & & & & (2\text{ P.}) \\ X & \rightarrow & \#\,, & Y & \rightarrow & b\,, & Z & \rightarrow & a\,. & (1\tfrac{1}{2}\text{ P.}) \end{array}$$

Aufgabe 3 (6 Punkte)

Geben Sie für jede der folgenden Mengen an, ob sie entscheidbar ist oder nicht. Beweisen Sie ihre Behauptungen.

1. $L_1 = \{w \in \Sigma^* \mid \varphi_w(0) = 0\}$
2. $L_2 = \{w \in \Sigma^* \mid \varphi_w(w) = w\}$
3. $L_3 = \{w \in \Sigma^* \mid \varphi_0(0) = w\}$

Lösungsvorschlag

1. Unentscheidbar nach Rice. Sei $F = \{f \mid f(0) = 0, f \text{ berechenbar}\}$. Dann ist die Identitätsfunktion $i \in F$, aber die überall undefinierte Funktion $\Omega \notin F$, und $L_1 = \{w \mid \varphi_w \in F\}$. Damit sind die Bedingungen des Satzes erfüllt. (2P.)
2. Unentscheidbar, aber der Satz von Rice ist nicht anwendbar. Reduktion z.B. von H_0: Für w, konstruiere TM, die $M_w[\epsilon]$ simuliert und bei Halten ihre ursprüngliche Eingabe zurückgibt. Reduktionen von K funktioniert auch. (2P.)
3. Trivial entscheidbar, da einelementige Menge:
   $L_3 = \{\varphi_0(0)\}$ (bzw. $L_3 = \emptyset$, falls $\varphi_0(0) = \perp$). (2P.)

Richtige Behauptung (entscheidbar oder nicht): 0,5P.
Beweis: 1,5P.

Aufgabe 4 (8 Punkte)

Zeigen Sie die primitive Rekursivität der folgenden Funktionen *eq*, *step* und *mod* unter Beachtung des anschließenden Hinweises:

1. Das 2-stellige Prädikat $eq(m, n)$, das einem Paar $(m, n) \in \mathbb{N} \times \mathbb{N}$ den Wert 1 zuordnet, falls $m = n$ gilt, und andernfalls den Wert 0 besitzt.

2. Eine Funktion $step(x, n)$, die für $x < n$ den Wert $x + 1 \bmod n$ liefert.
   Andernfalls darf der Funktionswert beliebig sein.

3. Die Funktion $mod(a, b)$ für natürliche Zahlen $a, b \in \mathbb{N}$, die für $b = 0$ den Wert 0 hat und andernfalls $a \bmod b$ berechnet.

<u>*Hinweis*</u>:
Sie dürfen zusätzlich zu den Basisfunktionen der primitiven Rekursion die folgenden Funktionen als primitiv rekursiv annehmen: $plus(m, n)$ $(+)$, $times(m, n)$ $(\cdot)$, $dotminus(m, n)$ $(\dot{-})$, $pred(n)$, $c(m, n)$, $p_1(n)$, $p_2(n)$, $ifthen(n, a, b)$ und c_n^k (konstante k-stellige Funktion). Sie dürfen die erweiterte Komposition und das erweiterte rekursive Definitionsschema benützen. LOOP-Programme sind nicht erlaubt.

Lösungsvorschlag

1. $eq(n, m) = 1 \dot{-} ((n \dot{-} m) + (m \dot{-} n))\,.$ (3 P.)

2. $step(x, n) = ifthen(eq(x + 1, n), 0, x + 1)\,.$ (2 P.)

3. $mod(0, b) = 0,$
   $mod(n + 1, b) = ifthen(eq(b, 0), 0, step(mod(n, b), b))\,.$ (3 P.)

Andere Varianten sind selbstverständlich möglich, müssen aber dem (erweiterten) PR-Schema entsprechen.
Wenn *mod* für die Definition von *step* verwendet wurde, und *mod* aber falsch ist (z.B. nicht PR), dann kann das leider nicht als Folgefehler gewertet werden, da Aufgabenteil 2 sonst trivial würde.

Aufgabe 5 (8 Punkte)

Wir betrachten einen PC-Konfigurator, der PCs aus verschiedenen Komponenten zusammenstellt. Dafür sei A eine Vereinigung von paarweise disjunkten, endlichen Mengen $A_1, \ldots, A_n$ von möglichen Komponenten. Intuitiv enthalten die A_i Komponenten gleichen Typs (z.B. "Prozessoren", "Grafikkarten", "Festplatten").
Eine *Konfiguration* ist eine Menge $K \subseteq A$, die aus jedem A_i genau ein Element enthält (damit ist $|K| = n$). Einige Komponentenkombinationen sind aber inkompatibel. Das beschreibt man durch Inkompatibilitätsmengen $I_1, \ldots, I_k \subseteq A$. Eine Konfiguration K heißt *lauffähig*, wenn es kein $j \in \{1 \ldots k\}$ gibt mit $I_j \subseteq K$.

1. Wir betrachten folgendes Problem:

   KONFIGURATOR:
   Gegeben: Konstanten n und k, endliche Mengen $A, A_1, \ldots, A_n$ und $I_1, \ldots, I_k \subseteq A$, so dass $A = A_1 \cup \cdots \cup A_n$ und $A_i \cap A_j = \emptyset$ für $i \neq j$.

   Problem: Gibt es eine lauffähige Konfiguration $K \subseteq A$?

   Geben Sie eine polynomielle Reduktion von KONFIGURATOR auf SAT an.

2. Das folgende modifizierte Problem beachtet auch Kundenwünsche:

   KONFIGURATOR+ (Änderungen unterstrichen):
   Gegeben: Konstanten n und k, endliche Mengen $A, A_1, \ldots, A_n$ und $I_1, \ldots, I_k \subseteq A$ $\underline{\text{und } W \subseteq A}$, so dass $A = A_1 \cup \cdots \cup A_n$ und $A_i \cap A_j = \emptyset$ für $i \neq j$.

   Problem: Gibt es eine lauffähige Konfiguration $K \subseteq A$ $\underline{\text{mit } W \subseteq K}$?

   Intuitiv bezeichnet die Wunschmenge W dabei die Menge der Komponenten, die der Kunde auf jeden Fall in seinem System haben möchte.

   Zeigen Sie:

   (a) KONFIGURATOR $\leq_p$ KONFIGURATOR+
   (b) KONFIGURATOR+ $\leq_p$ KONFIGURATOR

   Hinweis: Beachten Sie, dass eine Wunschmenge W unerfüllbar sein kann, wenn z.B. $|W \cap A_i| > 1$ für ein $i \in \{1 \ldots n\}$.

Lösungsvorschlag

1. Wir verwenden die Variablen v^a, die besagt ob $a \in A$ in der Konfiguration vorkommt. (1P.)

   Es wird mindestens eine Komponente ausgewählt:

   $$\bigwedge_{1 \le i \le n} \bigvee_{a \in A_i} v^a$$

   (1P.)

   Es wird maximal eine Komponente ausgewählt:

   $$\bigwedge_{1 \le i \le n} \bigwedge_{\substack{a,a' \in A_i \\ a \neq a'}} \neg v^a \vee \neg v^{a'}$$

   (1P.)

   Es werden keine inkompatiblen Komponenten ausgewählt:

   $$\bigwedge_{1 \le j \le k} \bigvee_{a \in I_j} \neg v^a$$

   (1P.)

2. (a) Mit leerer Wunschmenge. (1P.)

   (b) Die Wünsche $w \in W$ werden durch neue, einelementige Inkompatibilitätsmengen $J_{i,w'} = \{w'\}$ für alle $w, w' \in A_i$, $w' \neq w$, $i \in [n]$ modelliert.
   Alternativ kann man die A_i entsprechend einschränken. (3P.)