Harald Ganzinger
Jürgen Stuber Saarbrücken, 12.3.97

Bemerkungen zur Klausur Informatik III

Aufgabe 1

Das war zwar etwas komplizierter als ich mir eigentlich gewünscht hätte, es haben aber doch sehr viele diese Aufgabe richtig gelöst. Es war nicht nötig, bei (b) alle 16 Zustände anzugeben, es waren nur 7 erreichbar.

Häufige Fehler: Bei (b) nicht mit $\{z_0, z_1\}$ angefangen.

Aufgabe 2

Das Komplement besteht aus den drei Teilen $\{ubav \mid u, v \in \Sigma^*\}$, $\{a^n b^{2n} \mid n \geq 0\}$ und $\{a^m b^n \mid n \text{ ungerade}\}$. Das ist die einfachste Lösung, es gab auch andere.

Häufige Fehler: $\{a^m b^n \mid n \text{ ungerade}\}$ fehlt, Kettenregeln oder ε-Regeln in der Chomsky-Normalform, Ableitungsbäume fehlen (was war daran eigentlich so schwer?).

Die natürlichen Zahlen fangen in der Informatik bei 0 an, weil man fast in jeder Situation auch die 0 braucht. Nebenbei, auch Induktionsbeweise werden oft einfacher, wenn man als Basisfall $n = 0$ wählt. Wir haben hier aber keine Punkte abgezogen, wenn die Lösung unter der Annahme $\mathbb{N} = \{1, 2, 3, dots\}$ korrekt war. In dem Fall sind auch noch die Teilsprachen a^* und b^* im Komplement.

Aufgabe 3

Hier kann man entweder dreimal eine eins addieren, oder in einem Duchlauf eine 'schriftliche Addition' mit Übertrag und so weiter machen.

Häufige Fehler: Multiplikation mit drei statt Addition (??)

Aufgabe 4a

Regulär. Man kann einen Automaten mit drei Zuständen bauen, der L_1 akzeptiert (führende Nullen lassen wir zu). Dazu hilft es zu wissen, daß eine Dezimalzahl genau dann durch drei teilbar ist, wenn ihre Quersumme durch drei teilbar ist. Man addiert dann die Ziffern modulo drei, und der Zustand, der der Restklasse [0] entspricht, ist Endzustand.

Intuitiv ist eine Sprache regulär, wenn man die Worte mit konstantem endlichem Speicher erkennen kann, unabhängig von der Wortlänge. Hier kann man über das Wort laufen und braucht sich nur zu merken, in welcher von drei Restklassen das schon gelesene Teilwort liegt.

Wer Lust hat, kann sich ja mal überlegen, daß für alle $b, n \geq 2$ die Sprache $\{w \mid w \in \{0, \ldots, b-1\}^*, w \text{ gelesen als Zahl in } b\text{-närer Darstellung ist durch } n \text{ teilbar}\}$ regulär ist, und das es einen deterministischen Automaten mit höchstens n Zuständen dafür gibt.

Häufige Fehler: Regularität nicht gesehen. Mehrere Leute meinten sogar, daß L_1 unentscheidbar wäre!

Aufgabe 4b

Deterministisch kontextfrei, nicht regulär.

Wenn man aus Formeln alles bis auf die Klammern wegläßt, erhält man korrekte Klammerungen. Z.B. sind ε, $()()()$, $(()())$ korrekte Klammerungen.

Deterministisch kontextfrei: Indem man einen DPDA angibt. Daß L_2 *deterministisch* kontextfrei ist, sieht man intuitiv daran, daß man jederzeit genau weiß, was zu tun ist; man muß nicht raten.

Nicht regulär: Pumpinglemma, wähle $w = (^n)^n$.

Alternativ kann man auch die Abschlußeigenschaften nutzen: $L_2 \cap \{(^*)^*\} = \{(^n)^n \mid n \geq 0\}$, und das ist das gleiche wie $\{a^n b^n \mid n \geq 0\}$ und nicht regulär. Da reguläre Sprachen abgeschlossen unter Schnitt sind, und $\{(^*)^*\}$ nicht regulär ist, kann L_2 nicht regulär sein.

Häufige Fehler: nur kontextfrei gezeigt (gab aber auch einige Punkte), beim Pumpinglemma Aufteilung gewählt.

Aufgabe 4c

Semi-entscheidbar, nicht entscheidbar.

Semi-entscheidbar: Simuliere die Maschine und teste nach jedem Schritt, ob am Kopf zwei gleiche Zeichen nebeneinander. Falls ja, akzeptiere.

Nicht entscheidbar: Reduziere das Halteproblem auf leerem Band H_0 auf L_3, d.h. zeige $H_0 \leq L_3$. Dazu muß man die Maschine M_w so umbauen, daß sie niemals zwei gleiche Zeichen nebeneinander schreibt, und erst wenn sie hält zwei gleiche Zeichen schreiben.

Allgemein geht es hier um eine Eigenschaft der Maschine, die sich nicht nur auf die Sprache bezieht, deshalb ist der Satz von Rice nicht anwendbar. Wesentlich für die Reduktion $H_0 \leq L_3$ und damit die Unentscheidbarkeit ist, daß man jede Maschine so umbauen kann, daß die Eigenschaft erfüllt ist.

Häufige Fehler: Maschine nicht umgebaut, Reduktion in falscher Richtung.

Aufgabe 4d

Nicht semi-entscheidbar.

Man zeigt zuerst mit dem Satz von Rice, daß L_4 nicht entscheidbar ist. Mit dove-tailing zählt man parallel Worte x in L und Zahlen $n \in \mathbb{N}$ auf, und simuliert M_w gestartet auf x für n Schritte. Findet man ein $x \in T(M_w) \cap L$, ist $w \in \overline{L_4}$. Also ist $\overline{L_4}$ semi-entscheidbar, damit kann L_4 nicht semi-entscheidbar sein.

Nur sehr wenige haben hier den Satz von Rice richtig angewendet. Der Satz von Rice sagt intuitiv: Jede nichttriviale Eigenschaft der von einer Turingmaschine akzeptierten Sprache ($\hat{=}$ berechneten Funktion) ist unentscheidbar. Also: Eine Sprache $\{w \mid \ldots T(M_w) \ldots\}$ oder $\{w \mid$ die von M_w berechnete Funktion $\ldots\}$ ruft nach dem Satz von Rice.

Häufige Fehler: Satz von Rice nicht angewendet. Nur für spezielles $L = \Sigma^*$ Unentscheidbarkeit mit Leerheitsproblem gezeigt (gab aber auch ein paar Punkte).

Aufgabe 5

Man benutzt dynamisches Programmieren ähnlich wie beim CYK-Algorithmus.

D.h. man legt für alle Teilworte eine Tabelle an, in der man zunächst die Teilworte markiert, die in P liegen. Es gibt $O(n^2)$ Teilworte, und für jedes Teilwort geht das wegen $L \in P$ in Zeit $\leq p(|x|)$, also ist das polynomiell.

Dann geht man die Tabelle der Länge der Teilworte nach durch und markiert Teilworte, die sich aus zwei kürzeren Teilworten zusammensetzen lassen. Das Geht in $O(n^3)$.

Häufige Fehler: $\forall n L^n \in P$ gezeigt. Man hat dann für jedes n ein anderes Polynom, deshalb ist das zu schwach.

Alle Aufteilungen betrachtet: Es gibt 2^{n-1}, das sind zu viele.

$L^* = \{w_1 \ldots w_n \mid w_1 \ldots w_n \in L, n \geq 0\} \supsetneq \{w^n \mid w \in L, n \geq 0\}$!

Diese Aufgabe war ziemlich schwer, das war aber beabsichtigt.

Aufgabe 6

Reduziere CLIQUE auf TEILGRAPH-ISOMORPHISMUS. Dazu gibt man die k-Clique explizit als Teilgraph G_1 an. Alle HAMILTON-KREIS/PFAD-Probleme gingen auch.

Häufige Fehler: Reduktion in falscher Richtung, Funktion nur für die lösbaren Instanzen des Problems definiert, $TI \in NP$ fehlt.

Allgemein

Don't panic.

Entscheidbarkeit, Semi-Entscheidbarkeit, Reduktion sind wichtig! Schaut Euch im Schöning Kapitel 2.6 an!

Lernt NP-Vollständigkeit beweisen, wenn man begriffen hat wie's geht, sind es leicht verdiente Punkte. Wir machen hier keine schweren Reduktionen. Und für später wichtig ist es auch.