Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Moritz Maaß und Hanjo Täubig

Sommersemester 2005
Abschlussklausur
13. Juli 2005

Informatik IV

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

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

Allgemeine Hinweise zur Klausur

- Versehen Sie bitte alle von Ihnen genutzten Blätter mit Vornamen, Namen und Matrikelnummer.
- Bitte legen Sie Ihren Studentenausweis und einen amtlichen Lichtbildausweis auf die Hörsaalbank.
- Bitte schreiben Sie **nicht** in *roter* oder *grüner* Farbe bzw. mit Bleistift.
- Außer Ihrem Schreibgerät und einem handbeschriebenen DIN-A4-Blatt sind keine weiteren Hilfsmittel erlaubt.
- Begründen Sie alle Ihre Schritte.
- Die Bearbeitungszeit beträgt 90 Minuten.

Hörsaal verlassen von ...... bis ...... / von ...... bis ......

Vorzeitig abgegeben um ......

Besondere Bemerkungen:

	A1	A2	A3	A4	A5	A6	Σ	Korrektor
Erstkorrektur								
Zweitkorrektur								

Aufgabe 1 (14 Punkte)

Gelten folgende Aussagen? (Machen Sie ein Kreuz im Feld 'J', wenn die Aussage wahr ist, ansonsten bei 'N'.)
Für falsche Antworten werden keine Punkte abgezogen.

Es gibt eine Sprache L, die zwar von einem nichtdeterministischen Kellerautomaten (NPDA) akzeptiert wird, aber von keinem deterministischen Kellerautomaten (DPDA).	J | N
Eine Sprache L wird genau dann von einer kontextfreien Grammatik erzeugt, wenn L von einem nichtdeterministischen Kellerautomaten (NPDA) akzeptiert wird.	J | N
Wenn eine kontextfreie Sprache L auch von einer LR(k)-Grammatik erzeugt wird, dann ist L auf jeden Fall deterministisch kontextfrei.	J | N
Die Menge der deterministisch kontextfreien Sprachen (DCFL) ist für alle k identisch mit der Menge der LR(k)-Sprachen sowie mit der Menge der LR(1)-Sprachen.	J | N
Das Schnittproblem für kontextsensitive Sprachen ist nicht entscheidbar. ..	J | N
Es gibt Chomsky-0-Sprachen, die lediglich von nichtdeterministischen, nicht aber von deterministischen Turingmaschinen akzeptiert werden.	J | N
Wenn die Klasse der Chomsky-k-Sprachen unter einer bestimmten Operation abgeschlossen ist, dann ist auch die Klasse der Chomsky-($k+1$)-Sprachen unter dieser Operation abgeschlossen.	J | N
Wenn für die Klasse der Chomsky-k-Sprachen das Wort-, Leerheits-, Äquivalenz- oder Schnittproblem entscheidbar ist, dann ist dieses Problem auch für die Klasse der Chomsky-($k+1$)-Sprachen entscheidbar.	J | N
Eine Sprache $A \subseteq \Sigma^*$ ist genau dann entscheidbar, wenn ihre charakteristische Funktion χ_A berechenbar ist	J | N
Die Sprache $L = \{w \mid M_w$ akzeptiert alle Wörter ungerader Länge$\}$ ist rekursiv.	J | N
Jede berechenbare Funktion ist primitiv-rekursiv.	J | N
Alle totalen Funktionen sind LOOP-berechenbar.	J | N
Das spezielle Halteproblem H_s ist rekursiv aufzählbar, aber nicht rekursiv.	J | N
Die Worst-Case-Komplexität des QuickSort-Algorithmus ist $O(n \log n)$. .	J | N

Aufgabe 2 (9 Punkte)

Gegeben sei die reguläre Grammatik $G = (\{S, A, B\}, \{a, b\}, P, S)$ mit folgenden Produktionen P in EBNF:

$$\begin{array}{lcl} S & \rightarrow & aA \\ A & \rightarrow & aA|aB \\ B & \rightarrow & bS|b \end{array}$$

(i) Geben Sie einen nicht-deterministischen, endlichen Automaten an, der $L(G)$ erkennt (geben Sie den Zustandsgraph an).

(ii) Geben Sie eine äquivalente linkslineare Grammatik G' an.

(iii) Konstruieren Sie mittels der Potenzmengenkonstruktion einen deterministischen endlichen Automat, der $L(G)$ erkennt (geben Sie die Herleitung und den Zustandsgraph an).

Aufgabe 3 (8 Punkte)

Gegeben sei die Grammatik $G = (\{S, A, B\}, \{a, b\}, P, S)$ mit folgenden Produktionen P in EBNF:

$$\begin{array}{rcl} S & \rightarrow & SA|BA|b \\ A & \rightarrow & a \\ B & \rightarrow & SS|b \end{array}$$

Ordnen Sie die Grammatik G in die Chomsky-Hierarchie ein. Führen Sie den CYK-Algorithmus aus, um zu prüfen, ob das Wort *bababa* in $L(G)$ enthalten ist. Geben Sie die dabei entstehende Tabelle an. Geben Sie eine Linksableitung für *bababa* an, oder zeigen Sie, dass *bababa* nicht in $L(G)$ enthalten ist.

Aufgabe 4 (9 Punkte)

(i) Geben Sie LOOP-Programme an, welche die Variablen-Addition $x_i := x_j + x_k$ bzw. die bedingte Variablen-Subtraktion $x_i := x_j \dot{-} x_k$ ($a \dot{-} b$ bedeutet $max\{0, a - b\}$) simulieren.

Verwenden Sie nur die Basisanweisungen Zuweisung ($x_i := x_j$, $x_i := x_j + c$, $x_i := x_j \dot{-} c$), Sequenz ($P_1; P_2$) und LOOP-Schleife (**loop** x_i **do** P **end**).

(ii) Geben Sie ein LOOP-Programm an, welches folgenden Programmablauf simuliert:

for x_i **from** x_s **to** x_e **do**
  P
end

Hierbei werden x_i, x_s und x_e in P nicht verändert.

Verwenden Sie nur die Basisanweisungen wie in (i) gegeben. Zusätzlich dürfen Sie nur noch auf die oben definierten Konstrukte zurückgreifen.

(iii) Geben Sie ein LOOP-Programm an, welches den (binären) Logarithmus (abgerundet) eines in x_0 gegebenen Wertes berechnet, d.h. $\lfloor \log_2 x_0 \rfloor$.

Verwenden Sie nur die Basisanweisungen wie in (i) gegeben. Zusätzlich dürfen Sie nur noch auf die in der Vorlesung definierte If-Anweisung **if** $x_i = 0$ **then** P **end** und in vorangehenden Teilaufgaben definierte Konstrukte zurückgreifen.

Erläutern Sie Ihr Programm.

Aufgabe 5 (11 Punkte)

Geben Sie für die folgenden Funktionen jeweils eine primitiv rekursive Definition:

(i) $\min(n, m) = \begin{cases} n, & \text{falls } n \leq m, \\ m, & \text{sonst} \end{cases}$

(ii) $\text{twodiv}(n) = \lfloor \frac{n}{2} \rfloor$

Verwendenden Sie ausschließlich die nach den fünf Basisregeln primitiv rekursiven Funktion (konstante Funktionen, die Nachfolgerfunktion $n \to n + 1$, Projektionsfunktionen $\text{proj}_{k,i}$, Komposition, primitive Rekursion). Zusätzlich dürfen Sie nur noch annehmen, dass die Addition $add(n, m) = n+m$, und die bedingte Subtraktion $sub(n, m) = \max(n-m, 0)$ primitiv rekursiv sind. Erläutern Sie Ihre Definitionen.

Aufgabe 6 (9 Punkte)

Wir modifizieren das Modell des (nichtdet.) Kellerautomaten mit akzeptierenden Zuständen, indem wir ihm zusätzlich die Möglichkeit geben, den Keller umzudrehen. Der Wertebereich der ursprünglichen Übergangsfunktion $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Delta \to 2^{Q \times \Delta^*}$ wird dabei um die Flip-Option erweitert zu $\delta' : Q \times (\Sigma \cup \{\epsilon\}) \times \Delta \to 2^{Q \times \Delta^* \times \{\texttt{flip}, \texttt{noflip}\}}$. Für $(q', \alpha, \texttt{noflip}) \in \delta'(q, a, Z)$ liest der Flip-PDA im Zustand q mit oberstem Kellersymbol Z die Eingabe a (oder ϵ), ersetzt Z durch α und geht in den Zustand q' über. Für $(q', \alpha, \texttt{flip}) \in \delta'(q, a, Z)$ tut der Flip-PDA das gleiche, dreht danach aber den Keller um, so dass das oberste Element nun das unterste wird und umgekehrt. Der Flip-PDA geht also von der Konfiguration $(q, aw, ZZ_l \cdots Z_1 Z_0)$ nach $(q', w, Z_0 Z_1 \cdots Z_l \alpha^R)$ über (wobei α^R die gespiegelte Folge zu $\alpha \in \Delta^*$ ist).

Welche Sprachen kann ein Flip-PDA erkennen? Begründen Sie Ihre Aussage!