Universität Karlsruhe
Fakultät für Informatik, Institut für Telematik
Prof. Dr. S. Abeck, R. Scholderer

14.04.2000, 9:00

Informatik 1 - Nachholklausur

Schreiben Sie recht oben Ihren
- Namen
- Vornamen
- Matrikelnummer
- Aufgabennummer

auf jedes ausgeteilte farbige Blatt.

Jede Aufgabe ist auf einem separaten Blatt zu lösen.
Die Klausuraufgaben sind am Ende der Klausur zurückzugeben.
Viel Erfolg!

Aufgabe 1: *Chomsky-Grammatik (12 Punkte)*

Gegeben ist folgende Grammatik
Grammatik G={N, ∑, P,S}
N = {S, A, B}
∑ = {0,1}
P = { $S \rightarrow 0M \mid 1M$
$M \rightarrow 0B \mid 1A$
$A \rightarrow 0 \mid 0M \mid 1AA$
$B \rightarrow 1 \mid 1M \mid 0BB$ }

a) Geben Sie den einschänkendsten Chomsky-Typ zu der Grammatik an und begründen Sie Ihre Antwort. (4 Punkte)
b) Überführen Sie die Grammatik in die Backus Naur Form (4 Punkte)
c) Erstellen Sie den Ableitungsbaum für das Wort 01110001 (4 Punkte)

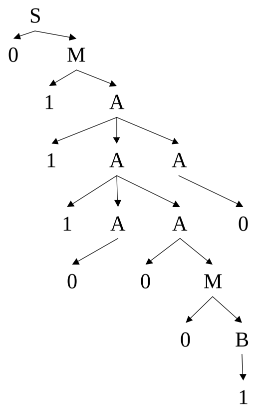
Lösung Aufgabe1: Chomsky

a)
Grammatik-Typ : CH-2 Grammatik (4 Punkte)
Begründung über Definition

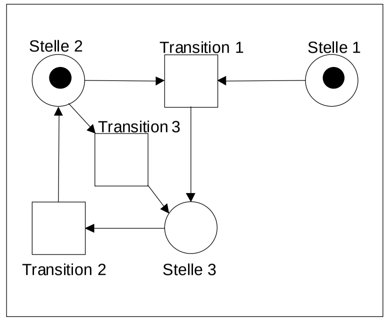
b) BNF

<*S*>::= 0<*M*> | 1<*M*>	*(1Punkt)*
<*M*> ::= 0<*B*> | 1<*A*>	*(1Punkt)*
<*A*> ::= 0 | 0<*M*> | 1<*AA*>	*(1Punkt)*
<*B*> ::= 1 | 1<*M*> | 0<*BB*>	*(1Punkt)*

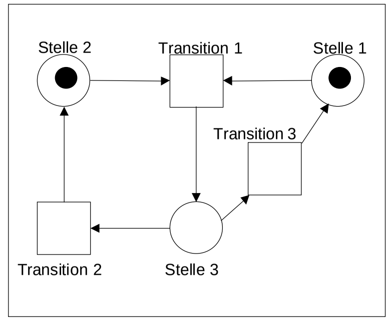
c) Ableitungsbaum
(4 Punkte)



Aufgabe 2: *Petrinetz (6 Punkte)*
Gegeben sind folgende Petrinetze



Petrinetz a

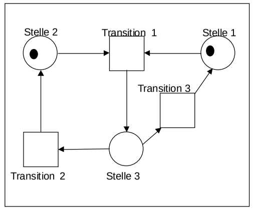


Petrinetz b

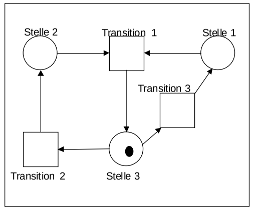
a) Geben Sie für beide Petrinetze alle Transitionen an, die als nächstes feuern können. (2 Punkte)
b) Welches der beiden Petrinetze kann nur zweimal feuern. Begründen Sie Ihre Aussage durch die Zustände, die das Petrinetz annimmt.

Lösung Aufgabe 2: *Petrinetz*

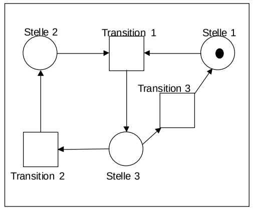
a) *Petrinetz a: Transition 1 und Transition 3 (2 Punkt)*
   Petrinetz b: Transition 1 (1 Punkt)
b) *Petrinetz b kann nur zweimal feuern (4 Punkte)*



Petrinetz b



Petrinetz b



Petrinetz b

Aufgabe 3: *Relationale Algebra (12 Punkte)*
Gegeben ist eine Bücherei, die zwei Buchtypen archiviert. Die vorrätigen Bücher sind in den Relationen Informatik und Mathematik beschrieben.

[Hinweis: Für die Teilaufgaben a,b und c gilt, dass nur die SQL Anfrage gewertet wird, aufgelistete Ergebnis-Tabellen werden in die Bewertung nicht aufgenommen!]

Relation Informatik:

Buchklasse	Autor	Titel	Stueckzahl
W1	Weber	Informatik I	0
K1	Kroeger	Infobasis	24
T1	Tichy	Informatikgrundlagen	0
T2	Turing	Intelligence Service	23
S1	Shannon	Informationstheorie	17
B1	Broy	Infogrundlagen	6

Relation Mathematik:

Buchklasse	Autor	Titel	Stueckzahl
S1	Shannon	Analysis I	8
N1	Neumann	Spieltheorie	31
H1	Hall	Analysis	0
T1	Teiher	Lineare Algebra	8
P1	Prent	Logik	15
T2	Thaler	Stochastik	11
T3	Turing	Morphogenesis	5

a) Geben Sie mittels einer SQL Datenbankoperation die Titel und Stückzahlen der Bücher aus der Relation Informatik aus, die momentan mindestens einmal vorhanden sind! (3 Punkte)
b) Geben Sie mittels einer SQL Datenbankoperation den Autor und Titel jener Mathematikbücher, von deren Autoren auch Informatikbücher vorhanden sind, aus! (3 Punkte)
c) Geben Sie mittels einer SQL Datenbankoperation alle Autoren der gesamten Bücherei aus! (3 Punkte)
d) Relationale Algebra:
   Geben Sie eine Projektion zu der Relation Informatik an, die die Attribute Titel und Stueckzahl beinhaltet. [Hinweis: Es ist u.a. ein Unterschema zu definieren!] (3 Punkte)

Lösung Aufgabe 3: *Relationale Algebra*
a) SELECT Titel, Stueckzahl FROM Informatik WHERE Stueckzahl > 0;

b)
SELECT inf.Autor, inf.Titel FROM Informatik AS inf, Mathematik AS mat
WHERE inf.Autor = mat.Autor;

c)
SELECT Autor FROM informatik UNION SELECT autor FROM mathematik;

d)
- ρ = Mathebuecher
- Mathe = {Buchklasse: CHAR(3), Autor: CHAR(20) Buchtitel: CHAR(20), Stueckzahl:Integer}
- Sub-Mathe = {Buchtitel: CHAR(20), Stueckzahl:Integer}

Sub-Mathe ist Unterschema von Mathe
Sub-Mathe $\subseteq$ Mathe

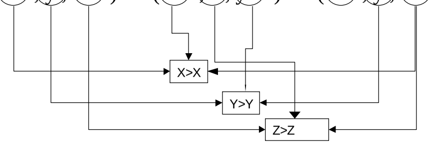
Projektion
$P_{\text{Sub-Mathe}}$: ρ(Mathe) $\rightarrow$ ρ (Sub-Mathe)

Aufgabe 4: *Modelle (8 Punkte)*
Gegeben ist die Formel $F = \exists x \; \exists y \; \exists z \; (P(x,y,z) \wedge P(x,z,y) \wedge P(z,y,x))$ und die Struktur A mit dem Paar (I_A, U_A), wobei
$I_A(P) = \{(m-1, m, m+1), m \in \mathbf{N} \}$
$U = \mathbf{N}$ ($\mathbf{N}$ = Natürliche Zahlen ohne Null)

Untersuchen Sie, ob die Struktur A ein Modell für die Formel F ist.

Lösung Aufgabe 4: *Modelle*

$\exists x\ \exists y\ \exists z\ (P(x\text{-}1\ ,\ y\ ,\ z\text{+}1\ ) \wedge P(x\text{-}1\ ,\ z\ ,\ y\text{+}1\ ) \wedge P(z\text{-}1\ ,\ y\ ,\ x\text{+}1\ ))$



$\exists x\ \exists y\ \exists z\ (P(0, 1 , 2) \qquad \wedge P(0, 1 , 2) \qquad \wedge P(0, 1, 2))$

x=0 und x=2 => Widerspruch

Widerspruch in den Aussagen.

Die Struktur ist kein Modell für die Formel.

Aufgabe 5: *Termersetzungssysteme (14 Punkte)*
Gegeben ist folgendes Termersetzungssystem:

(P) pred(succ(x))→x

(A0) add(zero,y)→y

(Al) add(succ(x),y)→succ(add(x,y))

(G0) zero $\stackrel{?}{=}$ zero → true

(Gl) zero $\stackrel{?}{=}$ succ(y) → false

(G2) succ(x) $\stackrel{?}{=}$ zero → false

(G3) succ(x) $\stackrel{?}{=}$ succ(y) → x $\stackrel{?}{=}$ y

a) Werten Sie den Ausdruck **add(pred(succ(zero)), add(zero, succ(zero))) $\stackrel{?}{=}$ zero** nach der Eager -Strategie aus. Eager: Es wird in Termen immer möglichst weit innen reduziert. D.h.: Auf eine Funktionsapplikation wird erst dann eine Regel angewandt, wenn sich kein Argument mehr reduzieren läßt. (7 Punkte)

b) Werten Sie den Ausdruck **pred(succ(succ(add(succ(zero), zero)))) $\stackrel{?}{=}$ succ(zero)** nach der Lazy-Strategie aus. Lazy: Es wird in Termen immer möglichst weit außen reduziert. D.h.: Auf Funktionsargumente werden nur dann Regeln angewandt, wenn weiter außen keine Regel anwendbar ist. (7 Punkte)

[Hinweis: Ein Ergebnis ohne Herleitung kann nicht in die Bewertung aufgenommen werden. Bei jedem Ersetzungsschritt ist auf die angewendeten Regeln zu verweisen]

Lösung Aufgabe 5: *Termersetzungssysteme*

a)

add(pred(succ(zero)), add(zero, succ(zero)) $\stackrel{?}{=}$ zero	**(P)**	
add(zero, add(zero, succ(zero)) $\stackrel{?}{=}$ zero	**(A0)**	**(2 Punkte)**

[oder alternativ: add(pred(succ(zero)), add(pred(succ(zero)), succ(zero)) mit A0]

add(zero, succ(zero)) $\stackrel{?}{=}$ zero	**(A0)**	**(2 Punkte)**
succ(zero) $\stackrel{?}{=}$ zero	**(G2)**	**(2 Punkte)**
false		**(1 Punkte)**

b)

pred(succ(succ(add(succ(zero), zero)))) $\stackrel{?}{=}$ succ(zero)	**(P)**	
succ(add(succ(zero), zero)) $\stackrel{?}{=}$ succ(zero)	**(A1)**	**(2 Punkte)**
add(succ(zero), zero) $\stackrel{?}{=}$ zero	**(A0)**	**(2 Punkte)**
succ(add(zero,zero)) $\stackrel{?}{=}$ zero	**(G3)**	**(1 Punkte)**
succ(zero) $\stackrel{?}{=}$ zero	**(G2)**	**(1 Punkte)**
false		**(1 Punkte)**

Aufgabe 6: *Funktional und Einbettung (10 Punkte)*
Gegeben ist folgende rekursive Funktionsdeklaration:

fct fac (**nat** x) **nat**:

if $x \stackrel{?}{=} 0$ **then** 1 **else** x * fac (x-1) **fi**

a) Geben Sie das zu fac gehörende Funktional τ an. (4 Punkte)

b) Geben Sie mittels Einbettung eine repetitiv rekursive Funktionsdeklaration für eine Verallgemeinerung fac_embed von fac mit der Funktionalität

fct fac_embed = **(nat, nat) nat:**

an, so daß gilt fac(x) = fac_embed(x, 1) für alle $x \in \mathbb{N}$.

(6 Punkte)

Lösung Aufgabe 6: *Einbettung*

a)
Funktional

$\boxed{\tau : (N^{\perp} \rightarrow N^{\perp}) \rightarrow (N^{\perp} \rightarrow N^{\perp}) \quad \text{(1 Punkt)}}$

$$\tau[f](x) = \begin{cases} \perp, \text{ falls } x = \perp & \text{(1 Punkt)} \\ 1, \text{ falls } x = 0 & \text{(1 Punkt)} \\ x * f(x), \text{ falls } x > 0 & \text{(1 Punkt)} \end{cases}$$

b)

fct fac (**nat** x, **nat** y) **nat:**

if x $\stackrel{?}{=}$ 0 **then** y*(2Pkt)* **else** fac (x-1, y*x*) (4Pkt)* **fi**

Aufgabe 7: *Gültigkeit und Lebensdauer (7 Punkte)*
Gegeben sei der folgende Programmabschnitt:

⌈ **nat** *x* = 2;
nat *y* = 3;

⌈ **fct** teilen = (**int** x, **nat** *y : y > 0)* **int**:
if *x< y* **then** 0 **else** 1 + teilen(*x – y, y)* **fi** ⌋

teilen(*x, y*) ⌋

a) Ordnen Sie den Konstanten und Variablen der auftretenden Ausdrücke durch Pfeile Bindungen zu (3 Punkte)
b) Geben Sie für die Identifikatoren x und y die Lebensdauer und deren Gültigkeit an. (4 Punkte)

Lösung Aufgabe 7: Gültigkeit und Lebensdauer

a)
Für jede Bindung werden 0,5 Punkte vergeben, bis auf den Sonderfall (s. Abbildung)

⌈ **nat** *x* = 2;
nat *y* = 3;
⌈ **fct** teilen = (**int** x, **nat** *y* : *y* > *0*) **int**:
if *x* < y **then** 0 **else** 1 + teilen(*x* – *y*, *y*) **fi** ⌋
teilen(*x*, *y*) ⌋

0,5

b)

	Lebens- x	dauer y	Gül- x	tig- y	ke- x	it y	Punkte
⌈ **nat** *x* = 2;	•		•				*0,5 Pkt*
nat *y* = 3;	•	•	•	•			*0,5 Pkt*
⌈ **fct** teilen = (**int** x, **nat** *y : y > 0)* **int**:	•	•			•	•	*1 Pkt*
if *x*<*y* **then** 0 **else** 1 + teilen(*x* – *y*, *y)* **fi**	•	•			•	•	*1 Pkt*
⌋	•	•			•	•	*0,5 Pkt*
teilen(*x*, *y*) ⌋	•	•	•	•			*0,5 Pkt*