Probeklausur zur Vorlesung
Grundbegriffe der Informatik
22. Januar 2016

Hinweis: Diese Probeklausur wurde von Tutoren erstellt. Die An-/Abwesenheit bestimmter Aufgabentypen oder auch deren Schwierigkeit in der Probeklausur sagt nichts über die richtige Klausur aus. Diese Probeklausur wurde vor allem weder vom Übungsleiter noch vom Professor konzipiert. Sie dient nur Übungszwecken.

Name:
Vorname:
Matr.-Nr.:
Tut.-Nr.:

Aufgabe	1	2	3	4	5	6
max. Punkte	8	7	5	2	9	6
tats. Punkte						

Gesamtpunktzahl: / 37	Note:

Aufgabe 1 (8 Punkte)

Kreuzen Sie für die folgenden Aussagen an, ob sie wahr oder falsch sind. Hinweis: Für jede richtige Antwort gibt es einen Punkt, für jede falsche Antwort wird ein Punkt abgezogen. Wenn Sie kein Kreuz setzen, bekommen Sie weder Plus- noch Minuspunkt, für das Ankreuzen beider Möglichkeiten wird ein Punkt abgezogen. Die gesamte Aufgabe wird mit mindestens 0 Punkten bewertet.

(a) Es gilt $\log(n) \in \mathcal{O}(\sqrt{n})$.

☒ wahr ☐ falsch

(b) Sei

$$L = \{w \in \{a,b\}^* \mid N_a(w) = N_b(w)\}$$

und

$$G = (\{S\}, \{\mathbf{a}, \mathbf{b}\}, S, \{S \to \mathbf{a}S\mathbf{b} \mid \mathbf{b}S\mathbf{a} \mid \varepsilon\})$$

gegeben. Dann gilt $L(G) = L$.

☐ wahr ☒ falsch

(c) Das leere Wort ϵ ist definiert als die Abbildung

$$\epsilon : \{\} \to \{\} \ .$$

☒ wahr ☐ falsch

(d) Seien L_1 und L_2 formale Sprachen. Dann gilt

$$L_1^* = L_2^* \to L_1 = L_2 \ .$$

☐ wahr ☒ falsch

(e) $\exists x \in \mathbb{N}_0 : \forall y \in \mathbb{N}_0 : x = y$

☐ wahr ☒ falsch

(f) Der Graph mit der Adjazenzmatrix

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

ist schlingenfrei.

☒ wahr ☐ falsch

(g) Eine Relation $R \subseteq A \times B$ ist injektiv, wenn gilt:

$$\forall (a_1, b_1), (a_2, b_2) \in R : a_1 = a_2 \to b_1 = b_2$$

☒ wahr ☐ falsch

(h) In ungerichteten Bäumen ist die Wurzel immer eindeutig

☐ wahr ☒ falsch

Aufgabe 2 (1.5+1.5+4 Punkte)

Es sei $G = (N, T, S, P)$ eine Grammatik mit den Nichtterminalsymbolen $N = \{S, A\}$, den Terminalsymbolen $T = \{a, b, c\}$ und den Produktionen

$$P = \{S \to AbS \mid c, A \to a \mid ac\}$$

a) Geben Sie für jedes der Wörter *abc*, *ababab* und *acbc* an, ob es in $L(G)$ enthalten ist.

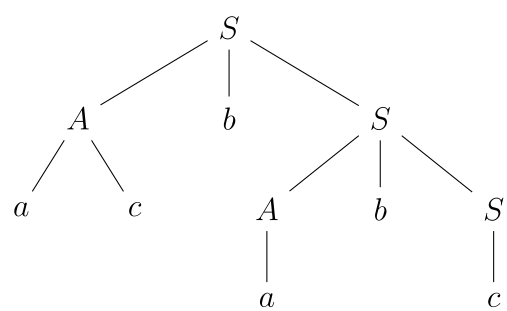
b) Leiten Sie das Wort $w = acbabc$ mithilfe von G ab und zeichnen Sie den Ableitungsbaum.

c) Es sei $U := T \cup N$. Beweisen Sie mittels vollständiger Induktion über die Anzahl der Ableitungsschritte, dass für jedes $n \in \mathbb{N}_0$ und jedes Wort $w \in U^*$ mit $S \Rightarrow^n w$ gilt, dass $N_a(w) + N_A(w) = N_b(w)$ ist.

Lösung

a) *abc* ist enthalten.
ababab ist NICHT enthalten.
acbc ist enthalten.

b) $S \Rightarrow AbS \Rightarrow acbS \Rightarrow acbAbS \Rightarrow acbabS \Rightarrow acbabc$



c) Setze $U := T \cup N$.
Wir zeigen zunächst mittels vollständiger Induktion, dass für jedes $n \in \mathbb{N}_0$ und jedes Wort $w \in U^*$ mit $S \Rightarrow^n w$ gilt, dass $N_a(w) + N_A(w) = N_b(w)$.

Induktionsanfang: $(n = 0)$
Sei $w \in U^*$ mit $S \Rightarrow^0 w$. Dann ist $w = S$ und $N_a(w) + N_A(w) = N_a(S) + N_A(S) = 0 + 0 = 0 = N_b(S) = N_b(w)$.

Induktionsschritt: Sei $n \in \mathbb{N}_0$ so, dass für alle $w \in U^*$ mit $S \Rightarrow^n w$ gilt, dass $N_a(w) + N_A(w) = N_b(w)$ (Induktionsvoraussetzung).
Sei außerdem $w' \in U^*$ ein beliebiges Wort, für das $S \Rightarrow^{n+1} w'$ gilt. Nach Definition von $\Rightarrow^{n+1}$ gibt es dann ein Wort $\bar{w} \in U^*$ mit $S \Rightarrow^n \bar{w}$, aus dem man w' durch eine Produktionsanwendung $\bar{w} \Rightarrow w'$ erhält.

Fall 1: Diese letzte Produktion ist $S \to AbS$
Dann ist

$$\begin{aligned} N_A(w') + N_a(w') &= N_A(\bar{w}) + 1 + N_a(\bar{w}) \\ &\overset{\text{IV}}{=} N_b(\bar{w}) + 1 \\ &= N_b(w'). \end{aligned}$$

Fall 2: Diese letzte Produktion ist $S \to c$
Dann ist

$$\begin{aligned} N_A(w') + N_a(w') &= N_A(\bar{w}) + N_a(\bar{w}) \\ &\overset{\text{IV}}{=} N_b(\bar{w}) \\ &= N_b(w'). \end{aligned}$$

Fall 3: Diese letzte Produktion ist $A \to a$ oder $A \to ac$
Dann ist

$$\begin{aligned} N_A(w') + N_a(w') &= N_A(\bar{w}) - 1 + N_a(\bar{w}) + 1 \\ &= N_A(\bar{w}) + N_a(\bar{w}) \\ &\overset{\text{IV}}{=} N_b(\bar{w}) \\ &= N_b(w'). \end{aligned}$$

In jedem Fall ist $N_A(w') + N_a(w') = N_b(w')$.

Schlussworte: Nach dem Prinzip der vollständigen Induktion gilt, dass für jedes $n \in \mathbb{N}_0$ und jedes Wort $w \in U^*$ mit $S \Rightarrow^n w$ gilt, dass $N_a(w) + N_A(w) = N_b(w)$.

Aufgabe 3 (1+2+2 Punkte)

Gegeben sei folgende Adjazenzmatrix

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

eines Graphen G.

(a) Ist der Graph gerichtet oder ungerichtet?
Wie kann man das an der Adjazenzmatrix ablesen?

(b) Zeichnen Sie G.

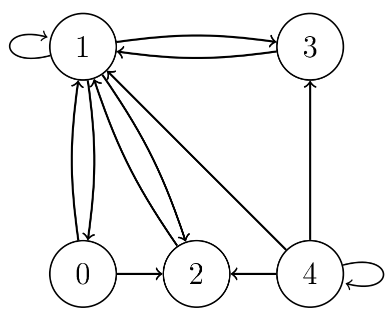
(c) Geben Sie die Wegematrix für G an.

Lösung

(a) Gerichtet, da

$$A_{02} = 1 \neq 0 = A_{20}$$

(b)



(c) Wegematrix ergibt sich als

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Aufgabe 4 (2 Punkte)

Mnemonik	Beschreibung
LDC a	$c \to Akku$
LDV a	$\langle a \rangle \to Akku$
STV a	$Akku \to \langle a \rangle$
ADD a	$Akku + \langle a \rangle \to Akku$
AND a	$Akku$ AND $\langle a \rangle \to Akku$
OR a	$Akku$ OR $\langle a \rangle \to Akku$
XOR a	$Akku$ OR $\langle a \rangle \to Akku$
EQL a	$\left.\begin{array}{ll}\text{falls } Akku = \langle a \rangle, & -1 \\ \text{sonst} & 0\end{array}\right\} \to Akku$
JMP a	$a \to$ IAR
JMN a	wenn $Akku < 0$, dann $a \to$ IAR
HALT	stoppt die MIMA
NOT	bilde Eins-Komplement von $Akku \to Akku$
RAR	rotiere $Akku$ eins nach rechts $\to Akku$

Tabelle 1: Alle benötigten MIMA-Befehle

Benutzen sie für die Bearbeitung der Aufgabe nur die oben angegeben Befehle und gehen sie von der in der Vorlesung vorgestellten Standard-Version der MIMA aus(Keine MIMA-X).
Es seien a_1 und a_2 gültige 20-bit Speicheradressen mit Werten > 0. Ebenso sei der folgende MIMA-Code gegeben:

```
START:  LDC   0x000F
        AND   a1
        STV   a2
        HALT
```

Geben sie an, was an den Speicheradressen a_1 und a_2 nach Ausführung des Codes steht und was dieser MIMA-Code im allgemeinen tut.

Lösung

Der Code rechnet $a_1 \quad '\texttt{mod}' \quad 16$ und $a_1 = a_1$, $a_2 = a_1 \quad '\texttt{mod}' \quad 16$

Aufgabe 5 (3+4+2 Punkte)

a) Welche der folgenden Formeln sind Tautologien?

1. $(A \to B) \to (\neg B \to \neg A)$
2. $(A \to B) \to (\neg A \to \neg B)$
3. $(\neg A \vee B) \vee (A \wedge \neg B)$

b) Sind die folgenden Formeln erfüllbar? Wenn ja, geben Sie eine passende Variablenbelegung an. Wenn nein, begründen Sie dies mit aussagenlogischen Umformungen.

1. $(((A \to (A \wedge \neg A)) \vee (A \leftrightarrow B)) \to B)$
2. $(\neg A \wedge (A \vee \neg A)) \wedge (\neg(A \leftrightarrow B) \wedge \neg B)$

c) Können die folgenden Symbolen in Formeln der Prädikatenlogik bzw. der Aussagenlogik enthalten sein?

Tragen Sie in der folgenden Tabelle ein **J** für „ja“ und **N** für „nein“ ein.

	$\to$	$\forall$	$\neg$	$\wedge$	,	$\doteq$	$\exists$	(
Aussagenlogik								
Prädikatenlogik								

Lösung

a) 1. ja
   2. nein
   3. ja

b) 1. Wir suchen eine Interpretation, die die Formel wahr macht: Die Implikation wird unter anderem dann wahr, wenn B wahr ist, d.h. wenn $I(B) = W$ gilt. In diesem Fall beeinflusst die Wahl von A die Auswertung der Formel nicht mehr, d.h. z.B. ist $I(A) = F$ und $I(B) = W$ ein Modell der Formel.
   2. $(\neg A \wedge (A \vee \neg A)) \wedge (\neg(A \leftrightarrow B) \wedge \neg B) \equiv$
   $(\neg A \wedge 1) \wedge (\neg(A \leftrightarrow B) \wedge \neg B) \equiv$
   $\neg A \wedge (((A \wedge \neg B) \vee (B \wedge \neg A)) \wedge \neg B) \equiv$
   $\neg A \wedge ((A \wedge \neg B) \vee (B \wedge \neg A \wedge \neg B)) \equiv$
   $\neg A \wedge (A \wedge \neg B) \equiv$
   0

c) Tabelle:

	$\rightarrow$	$\forall$	$\neg$	$\wedge$	,	$\doteq$	$\exists$	(
Aussagenlogik	ja	nein	ja	ja	nein	nein	nein	ja
Prädikatenlogik	ja	ja	ja	ja	ja	ja	ja	ja

Hinweis zum Korrigieren: 0.25 Punkte pro Spalte, runden auf halbe Punkte

Aufgabe 6 (2.5+1+1+1.5 Punkte)
Gegeben seien ein Alphabet $A = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}, \mathrm{e}, \mathrm{f}\}$ und ein Wort

$$w = \mathrm{eaadefadadaddeeabeeaefdeefadadeaefadeaea}$$

Es soll eine blockweise Huffman-Kodierung anhand dieses Wortes gefunden werden.

a) Die Häufigkeitsverteilung der enthaltenen Zweierblöcke sieht wie folgt aus:

Block	be	ea	de	ad	ef	Summe
Anzahl	1	6	2	7	4	20
Code						

Konstruieren Sie den dazugehörigen Huffman-Baum und geben Sie die Codewörter der einzelnen Blöcke an.

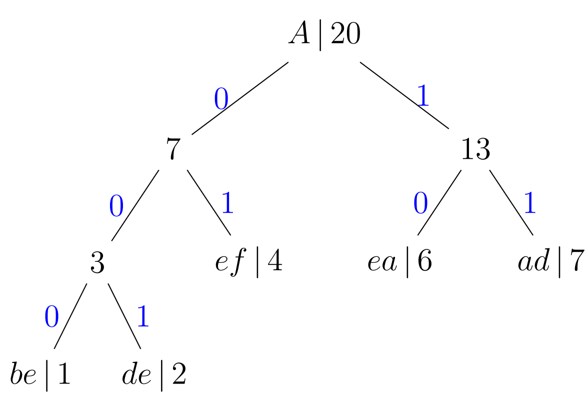
b) Kodieren Sie das Wort $u = ea\ ad\ ef\ ad\ ad\ ad\ de\ ea\ be\ ea$

c) Dekodieren Sie das Wort $v = 0011100001$.

d) Angenommen, das Speichern eines Zeichens von A kostet 4 bit und das Speichern eines Zeichens des Codewortes hingegen nur 1 bit. Um wieviel Prozent hat sich der benötigte Speicherplatz des Wortes u durch die Benutzung des Codes verändert?

Lösung

a) Baum:



Code:

Block	be	ea	de	ad	ef	Summe
#	1	6	2	7	4	20
Code	000	10	001	11	01	

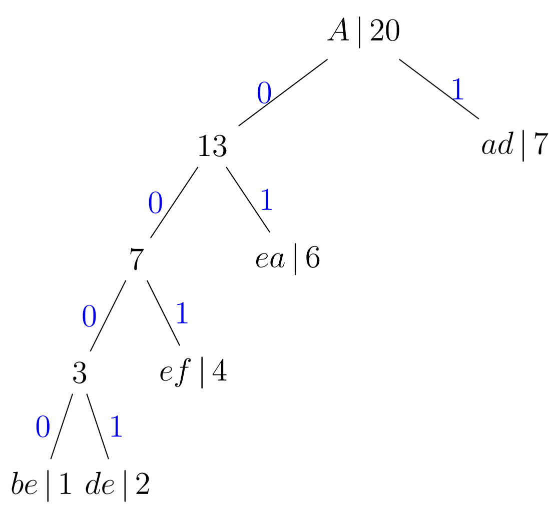
b) $u = ea\ ad\ ef\ ad\ ad\ ad\ de\ ea\ be\ ea$
$h(u) = 10\ 11\ 01\ 11\ 11\ 11\ 001\ 10\ 000\ 10$

c) $v = 001\ 11\ 000\ 01$
$h^{-1}(v) = de\ ad\ be\ ef$

d) Speicherplatz von u: 80 bit
Speicherplatz von $h(u)$: 22 bit
$\Rightarrow 72,5\%$ weniger Platz nötig
Hinweis: Diese Lösung ist exakt und keine Rundung.

ALTERNATIVE LÖSUNG:

a) Baum:



Code:

Block	be	ea	de	ad	ef	Summe
#	1	6	2	7	4	20
Code	0000	01	0001	1	001	

b) $u = ea\ ad\ ef\ ad\ ad\ ad\ de\ ea\ be\ ea$
$h(u) = 01\ 1\ 001\ 1\ 1\ 1\ 0001\ 01\ 0000\ 01$

c) $v = 001\ 11\ 000\ 01$
$h^{-1}(v) = ef\ ad\ ad\ be\ ad$

d) Speicherplatz von u: 80 bit
Speicherplatz von $h(u)$: 21 bit
$\Rightarrow 73,75\%$ weniger Platz nötig
Hinweis: Diese Lösung ist exakt und keine Rundung.