Karlsruher Institut für Technologie

aifb
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren

Lösung zur Klausur über den Stoff der Vorlesung
„Grundlagen der Informatik II“
(90 Minuten)

Name: ____________________ Vorname: ____________________

Matr.-Nr.: ____________________ Semester: __________ (WS 2018/19)

Ich bestätige, dass ich die folgenden Angaben gelesen und mich von der Vollständigkeit dieses Klausurexemplars überzeugt habe (Seiten 1-18).

____
Unterschrift des o. g. Klausurteilnehmers bzw. der o. g. Klausurteilnehmerin

Anmerkungen:

1. Legen Sie bitte Ihren Studierendenausweis bereit.
2. Bitte tragen Sie **Name**, **Vorname** und **Matr.-Nr.** deutlich lesbar ein.
3. Die folgenden **10 Aufgaben** sind vollständig zu bearbeiten.
4. Folgende Hilfsmittel sind zugelassen: **keine.**
5. Täuschungsversuche führen zum Ausschluss von der Klausur.
6. Unleserliche oder mit Bleistift geschriebene Lösungen können von der Klausur bzw. Wertung ausgeschlossen werden.
7. Die Bearbeitungszeit beträgt 90 Minuten.

Nur für den Prüfer :

1	2	3	4	5	6	7	8	9	10	-	-	-	-	-	-	gesamt
(13)	(11)	(10)	(10)	(7)	(8)	(8)	(11)	(9)	(3)							(90)

Aufgabenübersicht

Aufgabe 1	**13 Punkte**

2019-H-01 **Kellerautomaten**

/ 13

Gegeben sei die folgende Sprache L:

$$L = \left\{ w \in \{a, b, c\}^\star \mid w = a^i b^j c^k \quad \text{mit} \quad i, j, k \in \mathbb{N}_0, i > 0 \quad \text{und} \quad i = j + k \right\}$$

L enthält also alle Wörter über $\{a, b, c\}$, mit folgenden Eigenschaften:

- es gibt **genau** so viele a's wie b's und c's zusammen

(a) Geben Sie einen **nichtdeterministischen** Kellerautomaten $A = (E, S, K, \delta, s_0, k_0, F)$ mit $L(A) = L^\star$ an. Geben Sie A vollständig an.

Hinweis: Bspw. gilt:

$$\lambda, aabbac, abaaabcc, abac, abacab \in L^\star;$$
$$a, b, bbbaa, ababa, cbbbaa, aacb \notin L^\star.$$

Lösung:

$A = (\{a, b, c\}, \{s_0, s_1, s_2, s_3, s_e\}, \{a, b, c, k_0\}, \delta, s_0, k_0, \{s_e\})$

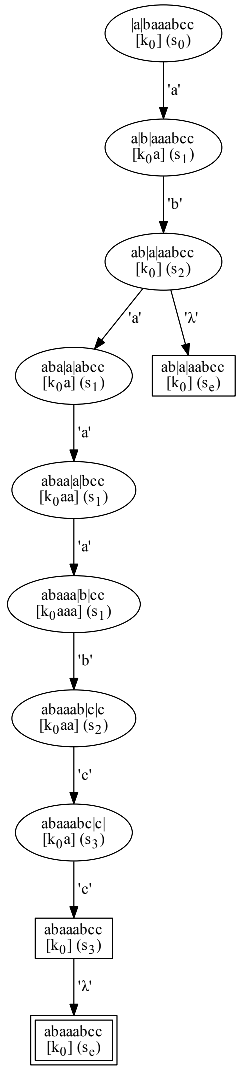
$(s_0, \lambda, k_0) \to (s_e, \lambda)$ // leeres Wort wird angenommen
$(s_0, a, k_0) \to (s_1, ak_0)$ // schreibe ein a
$(s_1, a, a) \to (s_1, aa)$
$(s_1, b, a) \to (s_2, \lambda)$ // lösche a, Wechsel des Zustands wegen der Reihenfolge
$(s_1, c, a) \to (s_3, \lambda)$ // lösche a, Wechsel des Zustands wegen der Reihenfolge
$(s_2, a, k_0) \to (s_1, ak_0)$ // neuer Zyklus
$(s_2, b, a) \to (s_2, \lambda)$
$(s_2, c, a) \to (s_3, \lambda)$ // lösche a, Wechsel des Zustands wegen der Reihenfolge
$(s_2, \lambda, k_0) \to (s_e, k_0)$ // Wort abgearbeitet
$(s_3, a, k_0) \to (s_1, ak_0)$ // neuer Zyklus
$(s_3, c, a) \to (s_3, \lambda)$
$(s_3, \lambda, k_0) \to (s_e, k_0)$ // Wort abgearbeitet

/ 9

(b) Berechnen Sie für das Wort *abaaabcc* den Ablauf.

/ 4

Lösung: Es ergibt sich für das Wort *abaaabcc* folgender Berechnungsablauf:



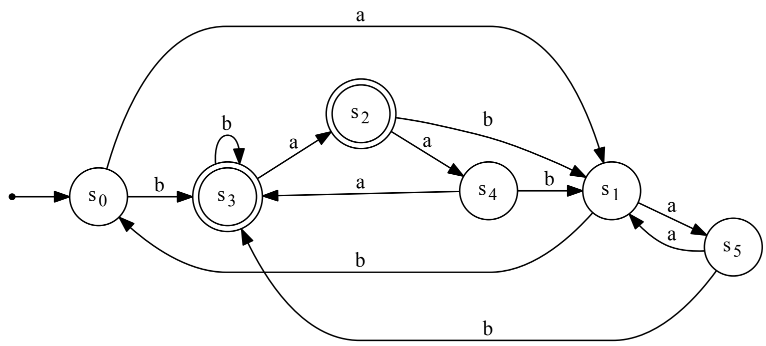
Aufgabe 2	**11 Punkte**

2019-H-02 **Minimierung endlicher Automaten**

/ 11

Gegeben sei folgender deterministischer endlicher Automat $A = (E, S, \delta, s_0, F)$. Durch das abgebildete Zustandsüberführungsdiagramm sei δ definiert.

δ :



(a) Minimieren Sie A und geben Sie den minimierten Automaten A' vollständig an. Geben Sie insbesondere die Übergangstabellen für A, A' und ein Zustandsüberführungsdiagramm δ' an.

/ 7

Lösung:

- Übergangstabelle für A:

	a	b
s_0	s_1	s_3
s_1	s_5	s_0
s_2	s_4	s_1
s_3	s_2	s_3
s_4	s_3	s_1
s_5	s_1	s_3

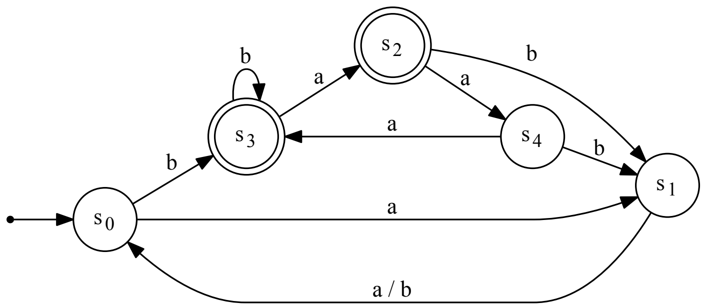
- Dreieckstabelle:

s_1	$\times_1$				
s_2	$\times_0$	$\times_0$			
s_3	$\times_0$	$\times_0$	$\times_1$		
s_4	$\times_1$	$\times_1$	$\times_0$	$\times_0$	
s_5	$-$	$\times_1$	$\times_0$	$\times_0$	$\times_1$
	s_0	s_1	s_2	s_3	s_4

- Übergangstabelle für A':

	a	b
s_0	s_1	s_3
s_1	s_0	s_0
s_2	s_4	s_1
s_3	s_2	s_3
s_4	s_3	s_1

- Der minimierte Automat A' lautet wie folgt:
  $A' = (\{a, b\}, \{s_0, s_1, s_2, s_3, s_4\}, \delta', s_0, \{s_2, s_3\})$



(b) Geben Sie die Mengen aller zueinander k-äquivalenten Zustände für $k \in \{0, 1, 2\}$ und die Mengen äquivalenter Zustände des endlichen Automaten A an. Verwenden Sie hierfür die nachfolgende Tabelle. Geben Sie auch einelementige Mengen an.

/ 4

Lösung:

0−Äquivalenz	$\{s_2, s_3\}, \{s_0, s_1, s_4, s_5\}$
1−Äquivalenz	$\{s_2\}, \{s_3\}, \{s_1\}, \{s_0, s_5\}, \{s_4\}$
2−Äquivalenz	$\{s_2\}, \{s_3\}, \{s_1\}, \{s_0, s_5\}, \{s_4\}$
Äquivalenz	$\{s_2\}, \{s_3\}, \{s_1\}, \{s_0, s_5\}, \{s_4\}$

Aufgabe 3	**10 Punkte**
2019-H-03	**Pumping Lemma für kontextfreie Sprachen**

/ 10

Gegeben sei die Sprache L:

$$L = \{a^m b^m c^m \mid m \geq 1\}$$

Zeigen Sie mit dem Pumping-Lemma für kontextfreie Sprachen, dass L nicht kontextfrei ist.

Hinweis: Betrachten Sie gegebenenfalls verschiedene Fälle.

Lösung:
Angenommen, L ist kontextfrei. Dann muss L das Pumping Lemma (PL) erfüllen.

(a) da L das PL erfüllt, muss es also die Konstante n ≥ 0 geben

(b) Wort $z \in L$ in Abhängigkeit von n wählen: $z = a^n b^n c^n$, es gilt $|\ z\ | \geq n$

(c) nun gilt laut PL, dass es eine Zerlegung des Wortes $z = uvwxy$ gibt, die die Eigenschaften (a) und (b) erfüllt
da $|\ vwx\ | \leq n$ können wir alle Zerlegungen auf die Folgenden Fälle reduzieren:
  (1) vwx enthält nur a's
  (2) vwx enthält a's und b's
  (3) vwx enthält nur b's
  (4) vwx enthält b's und c's
  (5) vwx enthält nur c's

(d) um das PL zu widerlegen, genügt es nun für jede Zerlegung ein $i \in N$ zu finden, sodass $uv^i wx^i y \notin L$
  - wähle für jeden obigen Fall i = 0
  - dann gilt für Fall (1), dass $uv^0 wx^0 w \notin L$, da $|vx|a \neq 0$ und das Wort somit weniger a's als c's enthält
  - Fall (2): es gilt, dass $|vx|a \neq 0$ oder $|vx|b \neq 0$ oder beide und damit $uv^0 wx^0 w \notin L$, da die Anzahl der a's oder b's nicht mehr mit denen der c's übereinstimmt
  - die restlichen Fälle sind analog

Dies ist ein Widerspruch zur Annahme, dass L das PL erfüllt. Damit ist L nicht kontextfrei.

Aufgabe 4	**10 Punkte**

2019-H-04 **Turingmaschine**

/ 10

Gegeben sei die Sprache $L = \{w \in \{0,1\}^* \mid \exists n \in N_0 : w = (001)^n\}$. Geben Sie eine Turingmaschine M an, die für Wörter $w \in L$ die gesamte Eingabe mit Einsen überschreibt und für alle anderen Wörter $w' \notin L$ die gesamte Eingabe mit Nullen überschreibt. In beiden Fällen soll M nach der Bearbeitung in einem Endzustand stoppen. Definieren Sie M vollständig.

Hinweis: Wenn die Eingabe das leere Wort ist, muss auch nichts überschrieben werden; die Maschine soll in diesem Fall nur in einem Endzustand halten.

Lösung:

$M = \{S, E, B, \delta, s_0, F\}$

$S = \{s_0, s_1, s_2, s_N, s_{NN}, s_{EE}\}$

$E = \{0, 1\}$

$B = \{0, 1, *\}$

$F = \{s_{EE}, s_{NN}\}$

Zustand	0	1	*
s_0	(s_1,1,R)	(s_N,0,N)	(s_{EE},*,N)
s_1	(s_2,1,R)	(s_N,0,N)	(s_N,*,L)
s_2	(s_N,0,N)	(s_0,1,R)	(s_N,*,L)
s_N	(s_N,0,R)	(s_N,1,R)	(s_{NN},*,L)
s_{NN}	(s_{NN},0,L)	(s_{NN},0,L)	
s_{EE}			

Aufgabe 5	**7 Punkte**

2019-H-05 **Komplexität**

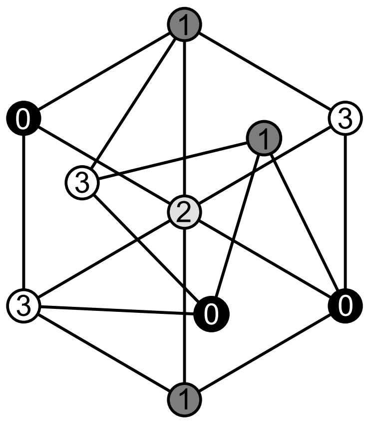
/ 7

k-COLOR bezeichnet das Entscheidungsproblem, ob die Knoten eines Graphen mit $k \geq 3$ Farben so eingefärbt werden können, dass benachbarte Knoten jeweils unterschiedliche Farben besitzen.

Formal wird dies wie folgt ausgedrückt: Sei $G = (V, E)$ ein ungerichteter Graph mit Knotenmenge V und Kantenmenge E. Die Funktion $f : V \to C \subseteq \mathbb{N}_0$ mit $|C| = k$ heißt Knotenfärbung von G. (C beschreibt hierbei eine Menge an Farben, die durch natürliche Zahlen kodiert sind.) f heißt **zulässig**, falls

$$\forall (v, w) \in E : f(v) \neq f(w).$$

Beispiel einer Graphfärbung für $k = 4$:



(a) Nehmen Sie an, Sie wüssten, dass k-COLOR NP-vollständig ist, und wollten zeigen, dass SAT (Entscheidungsproblem der Aussagenlogik) NP-schwer ist. Beschreiben Sie konkret, welche Schritte erforderlich sind, um k-COLOR auf SAT zu reduzieren.

/ 3

Lösung:

Es muss gezeigt werden, dass jede k-COLOR Probleminstanz als aussagenlogische Formel in konjunktiver Normalform dargestellt werden kann und die Transformation in polynomieller Zeit möglich ist. Für einen gegebenen Graphen existiert genau dann eine zulässige k-Färbung, wenn diese aussagenlogische Formel erfüllbar ist.

Nehmen Sie im Folgenden an, dass es für jeden Knoten v und jede Farbe $c \in C$ eine Boolesche Variable v_c gibt, die genau dann wahr ist, wenn v die Farbe c annimmt.

(b) Wie können Sie durch eine aussagenlogische Formel in konjunktiver Normalform darstellen, dass jeder Knoten mindestens einen Farbwert annehmen muss? Geben Sie diese Normalform formal oder umgangssprachlich präzise an.

Hinweis: Ignorieren Sie hierbei, ob die resultierende Färbung des Graphen zulässig ist.

/ 4

Lösung:

Sei $C = \{1, \ldots, k\}$. Für jeden Knoten v wird eine Klausel $K_v := (v_1 \vee \ldots \vee v_k)$ erzeugt. Die einzelnen Klauseln K_v werden für alle $v \in V$ durch logische UND-Operatoren miteinander verknüpft.

Aufgabe 6	**8 Punkte**

2019-H-06 **Mealy-/Moore-Automat**

/ 8

In einem Parkhaus kostet ein Parkticket €2. Der Parkscheinautomat akzeptiert nur €1 und €2 Münzen. Nachdem in den Parkscheinautomaten €2 eingeworfen wurden, kann der Kunde ein Ticket lösen, indem er auf den entsprechenden Knopf am Parkscheinautomaten drückt. In allen anderen Fällen wird kein Ticket ausgegeben. Nachdem der Kunde Geld in den Parkscheinautomaten geworfen hat, kann der Vorgang durch das Drücken eines zweiten Knopfes abgebrochen werden. Der Kunde bekommt das eingezahlte Geld danach wieder ausbezahlt, allerdings kein Ticket.

(a) Modellieren Sie den Parkscheinautomaten in einem Mealy-Automaten. Definieren Sie den Automaten A vollständig.

/ 7

Hinweis: Beachten Sie, dass der Parkautomat manchmal auch nicht bedient wird und in dieser Zeit nichts tut. Der Parkscheinautomat gibt kein Wechselgeld.

Lösung:

$M = (\Sigma, S, A, \delta, \gamma, s_0)$

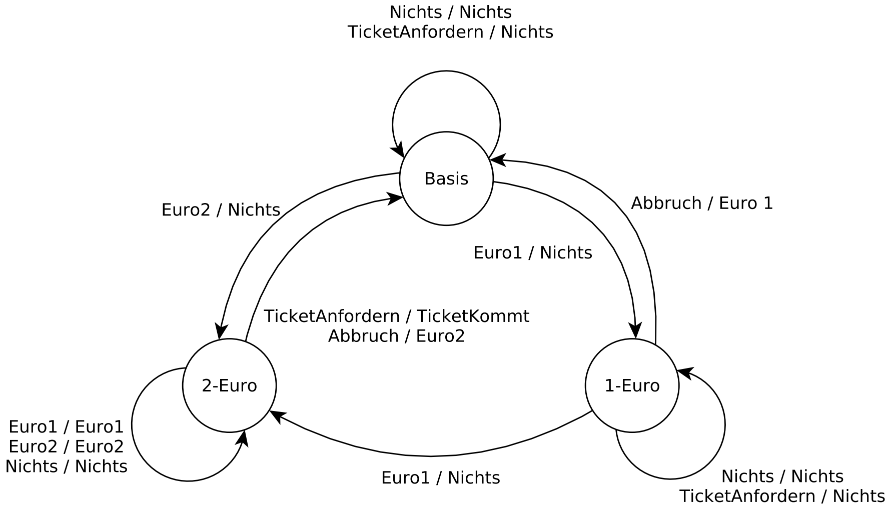
$\Sigma = \{Nichts, Euro1, Euro2, TicketAnfordern, Abbruch\}$

$S = \{Basis, 1 - Euro, 2 - Euro\}$

$A = \{Nichts, Euro1, Euro2, TicketAusgeben\}$

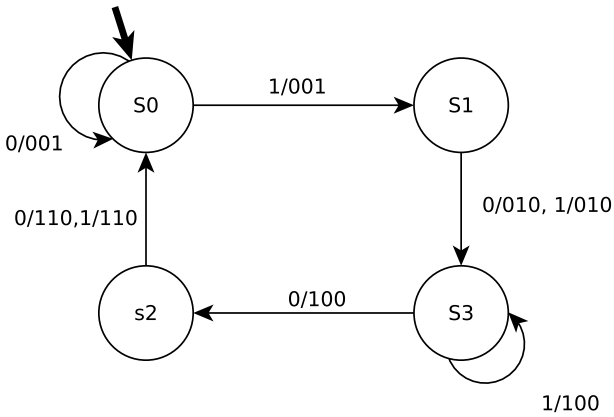
$s_0 = Basis$

δ und γ können als Graph modelliert werden:



(b) Ist der nachfolgende Automat ein Moore- oder Mealy-Automat? Begründen Sie.

/ 1



Lösung:

Es ist ein Mealy-Automat, da die Ausgabefunktion abhängig vom aktuellen Zustand ist und der Eingabe.

Aufgabe 7	**8 Punkte**
2019-H-07	**Huffman**

/ 8

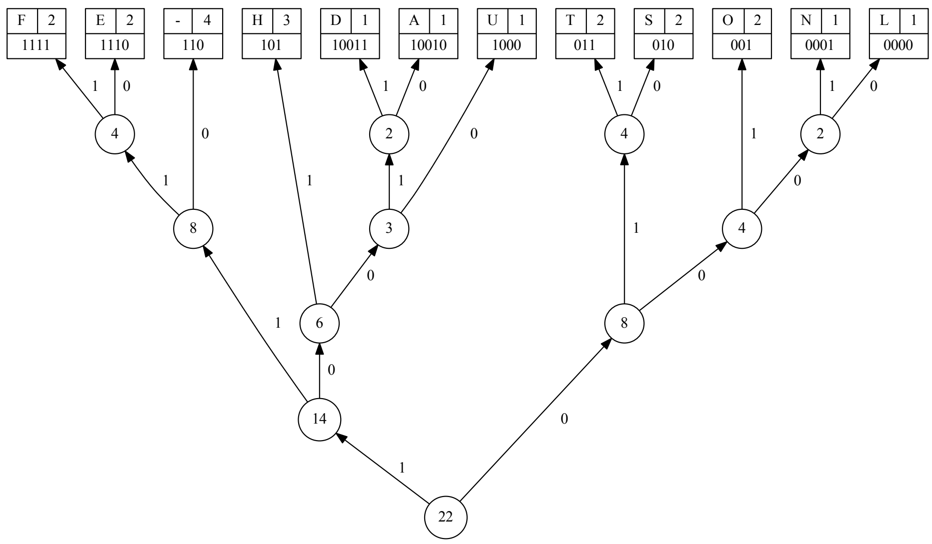
Gegeben sei der folgende Text aus 22 Zeichen:

DO-NOT-HASSLE-THE-HUFF

Erzeugen Sie zu der durch den Text gegebenen Wahrscheinlichkeitsverteilung eine Huffman-Kodierung. Tragen Sie dazu die Häufigkeiten der Zeichen in die untere Zeile der ersten Tabelle ein, erstellen Sie einen Huffman-Baum mit Angabe der Häufigkeiten an den Knoten und geben Sie in der zweiten Tabelle für jedes Zeichen eine dem Baum entsprechende Kodierung an.

F	E	-	H	D	A	U	T	S	O	N	L
2	2	4	3	1	1	1	2	2	2	1	1

Lösung (Beispiel):



Aufgabe 8	**11 Punkte**
2019-H-08	**Binary Decision Diagram**

/ 11

Gegeben sei die Boolesche Funktion $f : \mathbb{B}^3 \to \mathbb{B}$ mit

$$f(a,b,c) = \overline{(a \oplus b) \vee c}$$

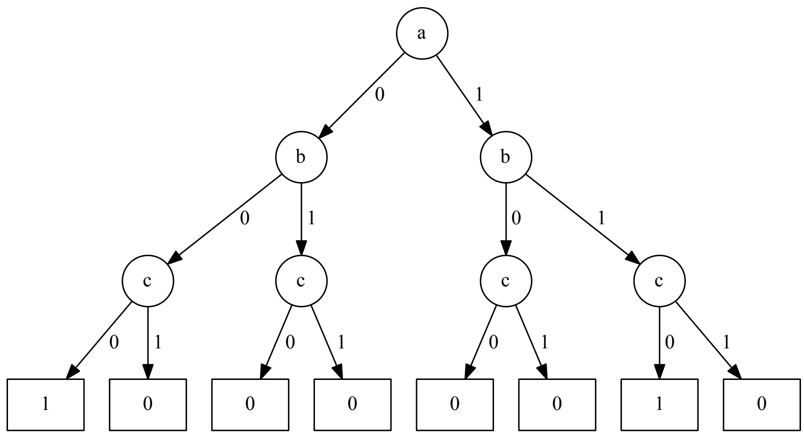
(a) Tragen Sie in die folgende Tabelle die Funktionswerte der Funktion f ein.

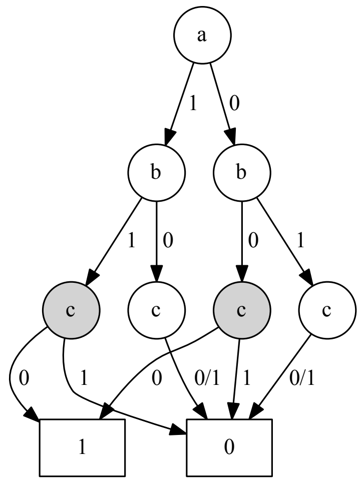
/ 3

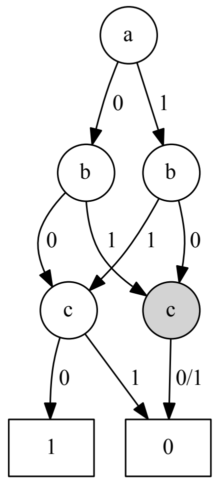
a	b	c	$a \oplus b$	$(a \oplus b) \vee c$	f
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	1	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	0	1	0

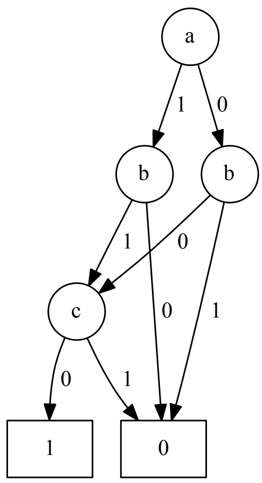
(b) Berechnen Sie das zu f gehörende BDD mit der Variablenreihenfolge $a \to b \to c$, indem Sie mit einer Baumdarstellung der Wertetabelle beginnen und den Algorithmus aus der Vorlesung anwenden.

/ 7




a
1
0
b
b
1
0
0
1
c
c
c
c
0
1
0
0/1
1
0/1
1
0


a
0
1
b
b
0
1
1
0
c
c
0
1
0/1
1
0


a
1
0
b
b
1
0
c
0
1
0
1
1
0

(c) Geben Sie die disjunktive Normalform (DNF): f (a, b, c) an.
f(a,b,c)= abc’ + a’b’c’

/ 1

Aufgabe 9	**9 Punkte**

2019-H-09 **Programmiersprachen**

/ 9

In der Vorlesung haben wir gesehen, dass alle gängigen Programmiersprachen dieselbe Ausdrucksmächtigkeit haben, also einander simulieren können. Im Folgenden seien gegeben:

- eine **GOTO-Sprache** (das sei eine übliche Assemblersprache, die Sprünge in Abhängigkeit des aktuellen Akkumulator-Werts erlaubt: `JNZ label`) und
- eine **WHILE-Sprache** (das sei eine höhere Sprache vergleichbar zu Java, die allgemeine WHILE-Schleifen unterstützt: `while (b) { * do something * }`).

(a) Gegeben sei ein Programmausschnitt in der GOTO-Sprache:

/ 4

```
label:  LOAD #1
        STORE R1
        MULT R2
        STORE R1
        LOAD R2
        SUB #1
        STORE R2
        JNZ label  // „JUMP NOT ZERO“
```

Simulieren Sie das Programm durch die WHILE-Sprache, d. h. geben Sie ein äquivalentes WHILE-Programm an.

Hinweise:

- Nutzen Sie möglichst Java-Notation oder einen verständlichen Pseudocode.
- Der Akkumulator sei über `ACC` erreichbar, die Register über ihre Namen `Ri`.

Lösung:

```
while (R2 != 0) {
    R1 = 1;
    R1 *= R2;
    R2 -= 1;
}
```

(b) Was müsste im gegebenen Programmausschnitt der **GOTO-Sprache** verändert werden, damit bei Terminierung die Summe $\sum_{x=2}^{R2} x$ in `R1` steht? (Annahme: `R2 > 2`)

/ 5

Lösung:

```
        LOAD #0
        STORE R1
label:  LOAD R1
        ADD R2
        STORE R1
        LOAD R2
        SUB #1
        STORE R2
        JNZ label  // „JUMP NOT ZERO“
        LOAD R1
        SUB #1
        STORE R1
```

Aufgabe 10	**3 Punkte**

2019-H-10 **Betriebssysteme**

/ 3

Betrachten Sie die Prozesse P1 bis P4, die in die Warteschlange eines Prozessors zur Bearbeitung eingereiht werden.

Prozesse	CPU-Zeit in ms
P1	20
P2	30
P3	25
P4	5

Teilen Sie den Prozessen Rechenzeit gemäß dem Round-Robin Verfahren zu. Die Zeitscheibe sei dabei in feste Zeitspannen der Länge $Z = 10\,\text{ms}$ unterteilt. Veranschaulichen Sie Ihr Ergebnis auf dem gegebenen Zeitstrahl.

Lösung: