Karlsruher Institut für Technologie

aifb

Institut für Angewandte Informatik
und Formale Beschreibungsverfahren

15.02.2016

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

Name: ______________________ Vorname: ______________________

Matr.-Nr.: ______________________ Semester: ________ (WS 2015/16)

Ich bestätige, dass ich die folgenden Angaben gelesen und mich von der Vollständigkeit dieses Klausurexemplars überzeugt habe (Seiten 1-22).

____

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
(10)	(10)	(12)	(8)	(8)	(9)	(11)	(8)	(8)	(6)							(90)

Aufgabenübersicht

Aufgabe 1	**10 Punkte**

2016-H-01 **Endliche Automaten und reguläre Ausdrücke**

/ 10

Für Automaten dieser Aufgabe reicht jeweils die Angabe von δ.

(a) Geben Sie über dem Alphabet $E = \{a\}$ wie folgt einen **deterministischen** endlichen Automaten (DEA) an: Er soll möglichst wenige Zustände haben, aber trotzdem noch verkleinert werden können (welche Sprache er erkennt, ist unerheblich).

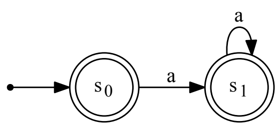
Sie sollen also den **kleinsten nicht minimalen (vereinfachten) DEA** angeben, der über einem **einelementigen** Alphabet gebildet werden kann. (Es gibt mehrere Lösungen.)

/ 3

Lösung:

$$A = (\{a\}, \{s_0, s_1\}, \delta, s_0, \{s_0, s_1\})$$

δ:



SKRIPT ID-2436

Eine weitere Alternative wäre, von s_1 zurück nach s_0 zu gehen.

Falls Sie (a) nicht lösen können, dürfen Sie für die folgenden Aufgabenteile einen beliebigen (nicht minimalen) DEA wählen. Zeichnen Sie diesen unter (a), bevor Sie fortfahren.

(b) Minimieren Sie Ihren DEA aus Aufgabenteil (a) nach dem Algorithmus aus der Vorlesung. Geben Sie insbesondere die (sehr einfache) Minimierungstabelle und den minimierten Automaten an.

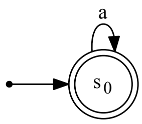
/ 2

Lösung:

s_1	$-$
	s_0

$$A_{min} = (\{a\}, \{s_0\}, \delta, s_0, \{s_0\})$$

δ:



Skript ID-2423

(c) Geben Sie einen regulären Ausdruck α an, der dieselbe Sprache definiert wie Ihr Automat aus Aufgabenteil (a) bzw. (b).

/ 1

Lösung:

$$L(A) = \{a\}^\star \implies \alpha = a^\star$$

bzw.

$$L(A) = \emptyset \implies \alpha = \emptyset$$

(d) Existiert für **jeden beliebigen** (vereinfachten) DEA mit n Zuständen über $E = \{a\}$ ein äquivalenter (vereinfachter) DEA mit $n + 1$ Zuständen? Begründen Sie oder finden Sie ein Gegenbeispiel. *Diese Aufgabe ist zum Knobeln, lösen Sie sie am besten zum Schluss.*

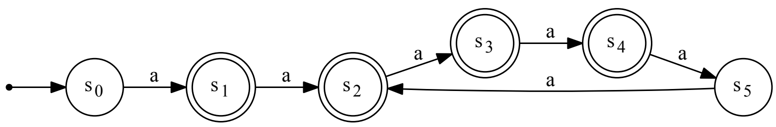
/ 4

Lösung:

Ein vereinfachter DEA über $E = \{a\}$ besteht immer aus einer Kette $(s_0, s_1, \ldots, s_{n-1})$ von durch a verbundenen Zuständen (End- oder Nichtendzustände), die nach dem letzten Zustand (s_{n-1}) eine Schleife zurück zu einem der vorherigen Zustände s_k bildet, siehe Beispielabbildung, wo $n = 6$ und $k = 2$ gilt.

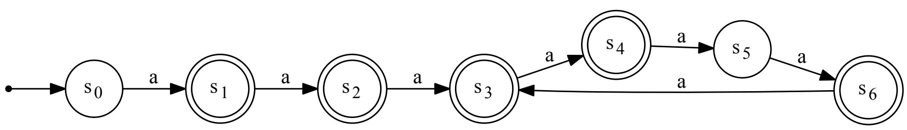
Nun können wir einen Zustand s_n einfügen, der genau dann Endzustand ist, wenn s_k Endzustand ist. Die Transition $s_{n-1} \to s_k$ biegen wir um zu $s_{n-1} \to s_n$ und zusätzlich fügen wir die neue Transition $s_n \to s_{k+1}$ ein. Der entstehende Automat erkennt offenbar dieselbe Sprache wie der Ursprungsautomat.

Beispiel-DEA mit $n = 6$ Zuständen:



Skript ID-11272

Äquivalenter DEA mit $n + 1 = 7$ Zuständen:



Skript ID-11273

Aufgabe 2	**10 Punkte**

2016-H-02 **Pumping-Lemma für EA-Sprachen**

/ 10

(a) Gegeben sei die Sprache L mit

$$L = \{w \in \{0, 1, 2\}^\star \mid |w|_0 = |w|_1 + |w|_2 \text{ oder } |w|_0 + |w|_1 = |w|_2\}$$

Es gilt beispielsweise:

$$\lambda, 01, 12, 2001, 1022, 21021000 \in L$$

$$0, 2, 22, 021, 2011, 11201, 0210120 \notin L$$

Zeigen Sie mit dem Pumping-Lemma für EA-Sprachen, dass L **nicht regulär** ist.

Hinweis: Achten Sie darauf, kein Testwort zu wählen, das nach dem Pumpen durch den jeweils anderen Fall abgedeckt ist. Wenn Ihr Testwort zum Beispiel

$$|w|_0 = |w|_1 + |w|_2$$

erfüllt, darf das gepumpte Wort auch den anderen Fall

$$|w|_0 + |w|_1 = |w|_2$$

nicht erfüllen.

/ 7

Lösung:

- Betrachte eine beliebige Partition von $w = xyz = 0^{2n}1^n2^n$ mit $|w| = 4n \geq n$ und
  (1) $|xy| \leq n$,
  (2) $|y| \geq 1$.
- Daraus folgt
  - $xy = 0^j$ mit $1 \leq j \leq n$, da $|xy| \leq n$ und $|y| \geq 1$,
  - $y = 0^k$ mit $1 \leq k \leq j$ und $x = 0^{j-k}$,
  - $z = 0^{2n-j}1^n2^n$.
- Wähle als Pumpvariable $i = 0$, dann ist $xy^iz = 0^{j-k}0^{2n-j}1^n2^n = 0^{2n-k}1^n2^n \notin L$, da $2n - k \neq 2n$ und gleichzeitig auch $2n - k + n = 3n - k \neq n$.

Folglich ist L nicht regulär.

Erklärung: Eine Fallunterscheidung ist nicht notwendig, da nur die Existenz eines einzigen Wortes $w \in L$, das die Pumpeigenschaft nicht erfüllt, ausreicht um zu zeigen, dass L nicht regulär ist.

(b) Angenommen, jemand beweist, dass für eine Sprache L' die Pump-Eigenschaft des Pumping-Lemmas für EA-Sprachen **erfüllt ist**. Argumentieren Sie, dass es dennoch sein kann, dass L' **nicht regulär** ist.

/ 3

Lösung:

Das Pumping-Lemma für EA-Sprachen ist lediglich eine notwendige, aber keine hinreichende Bedingung . Das bedeutet, dass eine Sprache durchaus das Pumping-Lemma für EA Sprachen erfüllen kann, selbst jedoch nicht regulär ist.

Aufgabe 3	**12 Punkte**

2016-H-03 **Kellerautomaten**

/ 12

Gegeben sei die Sprache L mit

$$L = \left\{0^i 1^j 2^k \mid i = j + k \text{ oder } i + j = k \text{ mit } i, j, k \in \mathbb{N}_0\right\}$$

Es gilt beispielsweise:

$$\lambda, 02, 0022, 000222, 01, 0012, 000122, 12, 111222, 00112222 \in L$$

$$0, 1, 2, 22, 012, 210, 0112, 01112, 11222 \notin L$$

Geben Sie einen **nichtdeterministischen** Kellerautomaten A an, der die Sprache L erkennt. Geben Sie A vollständig an.

Lösung:

$$A = (\{0, 1, 2\}, \{s_0, s_1, s_2, s_3, s_e\}, \{0, k_0\}, \delta, s_0, k_0, \{s_e\})$$

$$\begin{array}{rcl}
(s_0, 0, 0) & \Rightarrow & \{(s_0, 00)\} \\
(s_1, 1, 0) & \Rightarrow & \{(s_1, 00)\} \\
(s_2, 2, 0) & \Rightarrow & \{(s_3, \lambda)\} \\
(s_0, 1, 0) & \Rightarrow & \{(s_1, 00), (s_2, \lambda)\} \\
(s_1, 2, 0) & \Rightarrow & \{(s_3, \lambda)\} \\
(s_2, 1, 0) & \Rightarrow & \{(s_2, \lambda)\} \\
(s_3, 2, 0) & \Rightarrow & \{(s_3, \lambda)\} \\
(s_0, 2, 0) & \Rightarrow & \{(s_3, \lambda)\} \\
(s_3, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_2, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_0, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_0, 1, k_0) & \Rightarrow & \{(s_1, 0k_0)\} \\
(s_0, 0, k_0) & \Rightarrow & \{(s_0, 0k_0)\}
\end{array}$$

Skript ID-14150

Alternative mit nichtdet. Aufspaltung ganz zu Beginn:

$$A = (E, S, K, \delta, s_0, k_0, F)$$

$$\begin{array}{rcl}
(s_0, \lambda, k_0) & \Rightarrow & \{(s_{a0}, k_0), (s_{b0}, k_0), (s_e, k_0)\} \\
(s_{a0}, 0, 0) & \Rightarrow & \{(s_{a0}, 00)\} \\
(s_{a0}, 1, 0) & \Rightarrow & \{(s_{a1}, 00)\} \\
(s_{a0}, 2, 0) & \Rightarrow & \{(s_{a2}, \lambda)\} \\
(s_{a0}, 0, k_0) & \Rightarrow & \{(s_{a0}, 0k_0)\} \\
(s_{a0}, 1, k_0) & \Rightarrow & \{(s_{a1}, 0k_0)\} \\
(s_{a1}, 2, 0) & \Rightarrow & \{(s_{a2}, \lambda)\} \\
(s_{a1}, 1, 0) & \Rightarrow & \{(s_{a1}, 00)\} \\
(s_{a2}, 2, 0) & \Rightarrow & \{(s_{a2}, \lambda)\} \\
(s_{a2}, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_{b0}, 0, 0) & \Rightarrow & \{(s_{b0}, 00)\} \\
(s_{b0}, 1, 0) & \Rightarrow & \{(s_{b1}, \lambda)\} \\
(s_{b0}, 2, 0) & \Rightarrow & \{(s_{b2}, \lambda)\} \\
(s_{b0}, 0, k_0) & \Rightarrow & \{(s_{b0}, 0k_0)\} \\
(s_{b1}, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_{b1}, 1, 0) & \Rightarrow & \{(s_{b1}, \lambda)\} \\
(s_{b2}, \lambda, k_0) & \Rightarrow & \{(s_e, k_0)\} \\
(s_{b2}, 2, 0) & \Rightarrow & \{(s_{b2}, \lambda)\}
\end{array}$$

Skript ID-14157

Aufgabe 4 **8 Punkte**

2016-H-04 **Kontextfreie Grammatiken**

/ 8

Gegeben seien die kontextfreien Grammatiken G und G' mit $L(G) = L(G')$:

$$
\begin{aligned}
G &= (\{A, B, C, D, S\}, \{a, b\}, P, S),\\
P &= \{S \to aD \mid SAb,\\
&\quad A \to a \mid bbC,\\
&\quad B \to a \mid CBa,\\
&\quad C \to b \mid Bb,\\
&\quad D \to b \mid SAaC\}
\end{aligned}
$$

$$
\begin{aligned}
G' &= (\{A, B, C, C_{(a)}, C_{(b)}, D, D_{(0)}, D_{(1)}, D_{(2)}, D_{(3)}, D_{(4)}, S\}, \{a, b\}, P', S),\\
P' &= \{S \to C_{(a)}D \mid D_{(1)}C_{(b)},\\
&\quad A \to a \mid D_{(0)}C,\\
&\quad B \to a \mid D_{(2)}C_{(a)},\\
&\quad C \to b \mid BC_{(b)},\\
&\quad D \to b \mid D_{(4)}C,\\
&\quad D_{(0)} \to C_{(b)}C_{(b)},\\
&\quad D_{(1)} \to SA,\\
&\quad D_{(2)} \to CB,\\
&\quad D_{(3)} \to SA,\\
&\quad D_{(4)} \to D_{(3)}C_{(a)},\\
&\quad C_{(a)} \to a,\\
&\quad C_{(b)} \to b\}
\end{aligned}
$$

(a) In welcher Normalform befindet sich G'?

/ 1

(b) Geben Sie bezüglich G einen Ableitungsbaum für das Wort *abbbbb* an.

/ 3

(c) Geben Sie bezüglich G' einen Ableitungsbaum für das Wort *abbbbb* an.

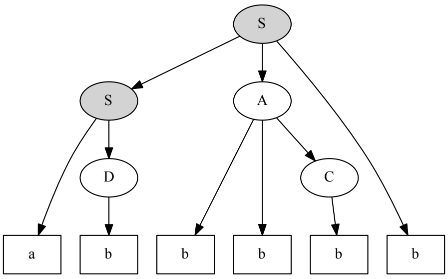
/ 4

Hinweise:

- Nutzen Sie Ihre Kenntnisse darüber, wie die Normalform von G' aus G konstruiert wird, um den Baum aus Teilaufgabe (b) einfach in eine Lösung für (c) abzuwandeln.
- Zeichnen Sie die Bäume auf die Rückseite der vorherigen Seite.

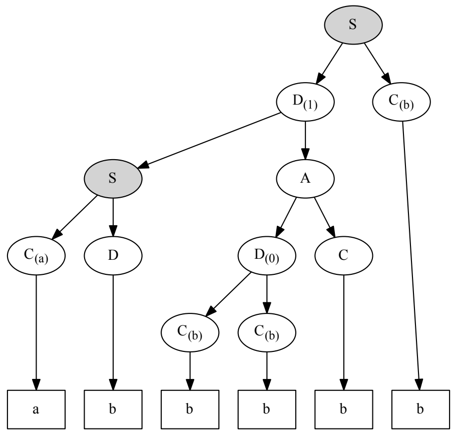
Lösung: G' ist in CNF.
Originalskript:

SKRIPT ID-9701



Skript in CNF:

SKRIPT ID-9713



Zur Erstellung des CNF-Baums müssen im Original-Baum einerseits diejenigen Blätter x betrachtet werden, die nicht von einem einzigen Nonterminalsymbol abgeleitet werden – hier muss ein Nonterminalsymbol $C_{(x)}$ als Zwischensymbol eingefügt werden. Andererseits müssen Abzweigungen von einem höheren Grad als 2 unter Zuhilfenahme von Nonterminalsymbolen $D_{(.)}$ durch mehrere Abzweigungen vom Grad 2 ersetzt werden. Da es nur Abzweigungen der Grade 1, 2 und 3 gibt, wobei die vom Grad 1 nur auf Terminale gehen, müssen nur die vom Grad 3 entsprechend der Produktionen von G' durch jeweils zwei Abzweigungen vom Grad 2 ersetzt werden.

Aufgabe 5	**8 Punkte**

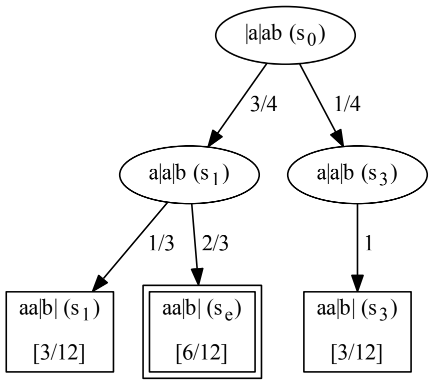
2016-H-05 **Komplexität**

/ 8

Eine **probabilistische Turingmaschine** ist eine Turingmaschine, die ihre Folgekonfiguration **zufällig** gemäß einer gegebenen Wahrscheinlichkeitsverteilung über die Menge der möglichen Übergänge in der Turingtafel bestimmt.

Das folgende Schaubild zeigt beispielhaft einen Berechnungsbaum einer probabilistischen Turingmaschine bei Eingabe des Wortes *aab*:

Hinweis: Die Schreibweise $u|a|v$ (s) für die Konfiguration ist äquivalent zur Schreibweise (u, s, av).



Die Zahl an den Kanten gibt an, mit welcher Wahrscheinlichkeit die Turingmaschine die entsprechende Abzweigung wählt, die Zahl an den Blättern, mit welcher Wahrscheinlichkeit die entsprechende Endkonfiguration erreicht wird. Die Turingmaschine akzeptiert also mit einer Wahrscheinlichkeit $6/12 = 1/2$.

Bezeichne pP die Klasse der Entscheidungsprobleme, die von einer probabilistischen Turingmaschine in **Polynomialzeit** mit einer vorgegebenen Wahrscheinlichkeit größer gleich p korrekt gelöst werden.

(a) Begründen Sie, warum $P \subseteq pP$ gilt (unabhängig vom Wert von p).

/ 4

Lösung:

Eine deterministische Turingmaschine kann als Spezialfall einer probabilistischen Turingmaschine betrachtet werden, die in jedem Zustand eine Folgekonfiguration mit der Wahrscheinlichkeit eins wählt . Falls ein Problem also von einer deterministischen Turingmaschine in Polynomialzeit gelöst werden kann, so kann es auch von einer probabilistischen Turingmaschine in Polynomialzeit mit einer Wahrscheinlichkeit von eins gelöst werden . Folglich muss P eine Teilmenge von pP sein.

(b) Begründen Sie, warum $pP \subseteq NP$ gilt (unabhängig vom Wert von p).

/ 4

Lösung:

Eine probabilistische Turingmaschine kann hingegen als Spezialfall einer nichtdeterministischen Turingmaschine betrachtet werden, die bei jeder Ausführung immer nur einen zufällig gewählten Ableitungsbaum durchläuft . Falls ein Problem also von einer probabilistischen Turingmaschine mit einer beliebigen Wahrscheinlichkeit in Polynomialzeit gelöst werden kann, so kann es auch von einer nichtdeterministischen Turingmaschine in Polynomialzeit gelöst werden . Folglich muss pP eine Teilmenge von NP sein.

Aufgabe 6	**9 Punkte**

2016-H-06 **Flip-Flops**

/ 9

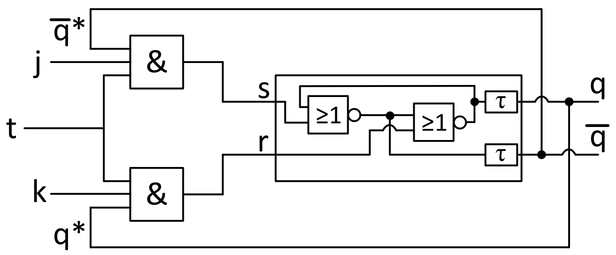
Bei dem gegebenen Schaltsymbol handelt es sich um ein asynchrones RS-Flip-Flop, das um zwei Verzögerungselemente an den beiden Ausgängen ergänzt wurde. Sie verzögern die Ausgabe der Signale q und $\overline{q}$ um $\tau = 0,5$ Nanosekunden (ns). Weitere Verzögerungen durch Gatter und Leitungen innerhalb der Schaltung können vernachlässigt werden.

(a) Ergänzen Sie die Skizze um alle notwendigen Leitungen und Gatter, um das asynchrone RS-Flip-Flop zu einem synchronen JK-Flip-Flop zu erweitern. Die Verzögerungselemente können Sie in dieser Teilaufgabe ignorieren.

Hinweis: Ein JK-Flip-Flop ist ein RS-Flip-Flop mit vorgeschalteten Gattern, die die für RS-Flip-Flops unzulässige Belegung $(r, s) = (1, 1)$ verhindern.

/ 3

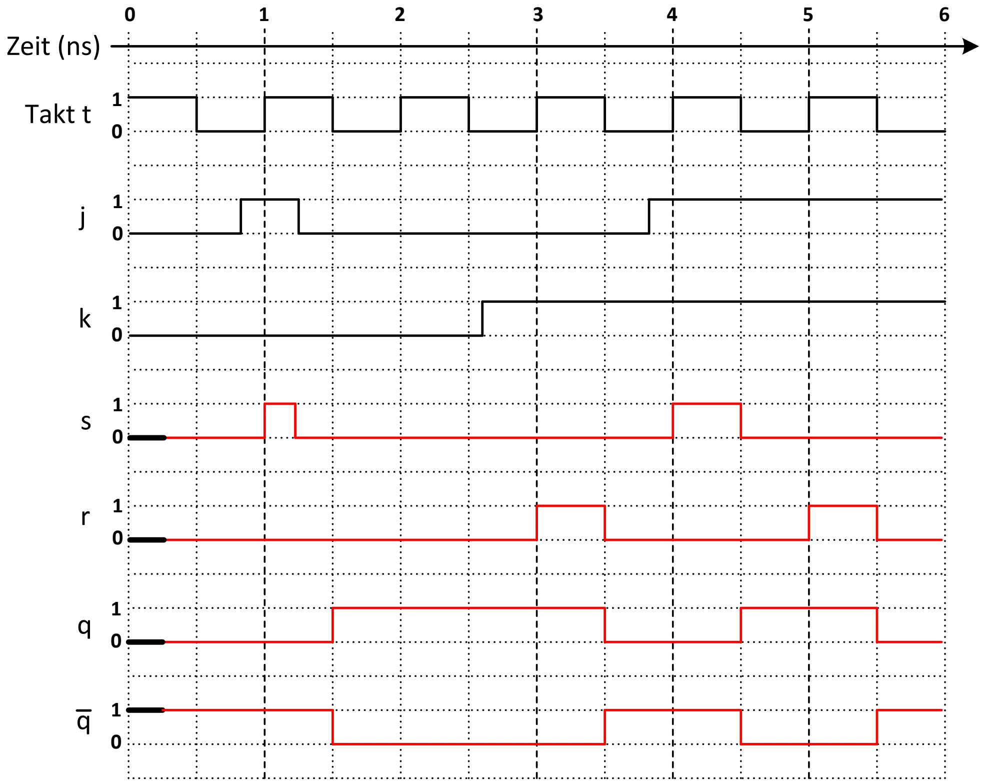
Lösung:



(b) Der gegebene Graph enthält die Signalverläufe für die drei Eingänge j, k und t (Takt) eines synchronen JK-Flip-Flops, sowie Anfangsbelegungen für die Ausgänge q und $\overline{q}$ und die internen Signale r und s (Eingangssignale des ursprünglichen asynchronen RS-Flip-Flops aus Aufgabenteil (a)). Vervollständigen Sie die Signale. Beachten Sie dabei, dass q und $\overline{q}$ **um** τ **verzögert** sind.

/ 6

Lösung:



Aufgabe 7	**11 Punkte**

2016-H-07 **Huffman-Kodierung**

/ 11

(a) Gegeben sei der folgende Text aus 29 Zeichen:

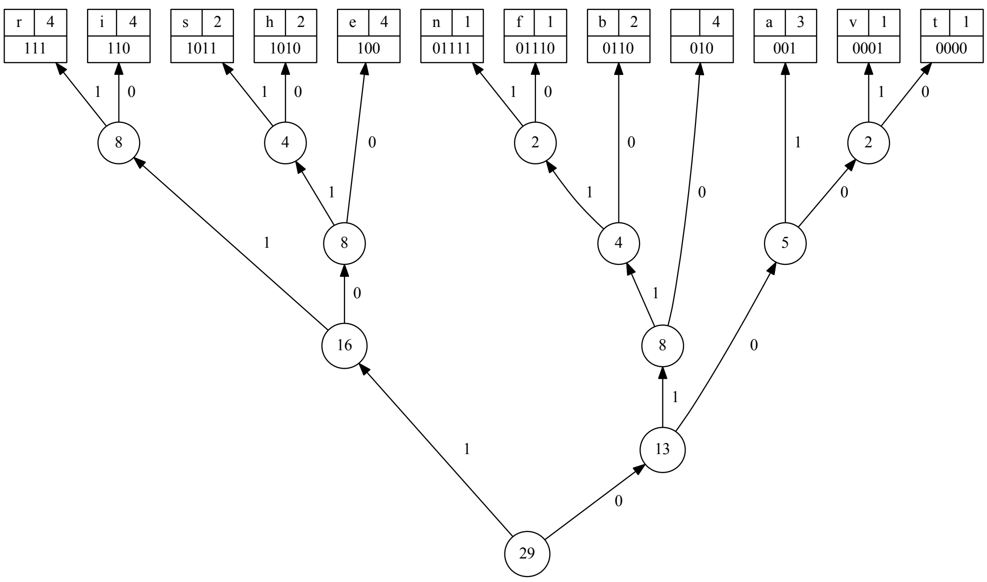
EIN_BRAVER_HAI_ISST_HAFERBREI

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.

/ 7

R	I	S	H	E	N	F	B	_	A	V	T
4	4	2	2	4	1	1	2	4	3	1	1

Lösung (Beispiel):



SKRIPT ID-13688

(b) Gegeben seien zwei gleich lange Texte A und B, die aus denselben Zeichen Z_1 bis Z_8 bestehen. Die Häufigkeitsverteilung aller Zeichen in beiden Texten kann der gegebenen Tabelle entnommen werden. Welcher der beiden Texte ist nach der Kodierung mit einer entsprechenden Huffman-Kodierung kürzer? Begründen Sie Ihre Antwort kurz.

/ 2

Zeichen:	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Z_8	gesamt
Text A	10	10	10	10	10	10	10	10	80
Text B	73	1	1	1	1	1	1	1	80

Lösung:
In Text A kommen alle n Zeichen gleich häufig vor. Da n eine Zweierpotenz ist, können sie also alle mit der gleichen Anzahl Bits (nämlich $\log_2 n$) kodiert werden, ohne die Fano-Bedingung zu verletzen.
Da in Text B die addierten Häufigkeiten von Z_2 bis Z_8 kleiner sind als die Häufigkeit von Z_1, wird dieses mit einem einzelnen Bit kodiert. Die anderen 7 Zeichen können mit maximal 4 Bit so kodiert werden, dass die Fano-Bedingung noch eingehalten wird (tatsächlich wird eines sogar mit nur 3 Bit kodiert). Da Z_1 73 mal vorkommt und die anderen Zeichen insgesamt 7 mal, ist Text B im kodierten Zustand kürzer.

SKRIPT ID-13692

(c) Wie lang ist Text A aus Teilaufgabe (b), wenn er mit einem Huffmancode kodiert ist?

/ 1

Lösung:
In Text A kommen alle n Zeichen gleich häufig vor. Da n eine Zweierpotenz ist, können

sie also alle mit der gleichen Anzahl Bits (nämlich $\log_2 n$) minimal kodiert werden, ohne die Fano-Bedingung zu verletzen.
Der Huffmankodierte Text A ist daher ($\log_2 8$) Bit/Zeichen $\times$ 80 Zeichen = 240 Bit lang.

SKRIPT ID-10805

(d) Wie lang wären die kodierten Texte A und B aus Teilaufgabe (b), wenn es sich bei den Zeichen Z_1 bis Z_8 um einzelne Ziffern handeln würde, die im BCD-Code kodiert sind?

/ 1

Lösung:
4 Bit/Zeichen $\times$ 80 Zeichen = 320 Bit

Aufgabe 8	**8 Punkte**

2016-H-8 **Rechnerarchitektur**

/ 8

(a) Nennen Sie die einzelnen Bestandteile des klassischen Konzepts des von Neumann-Rechners.

/ 2

Lösung:

- Speicherwerk
- Rechenwerk
- Steuerwerk (Leitwerk)
- Ein- und Ausgabewerk

(b) In heutigen Rechnern werden als Speicherelemente unter anderem CPU, Register, Cache-Speicher und Arbeitsspeicher eingesetzt. Grenzen Sie diese Elemente voneinander ab, indem Sie die folgenden Eigenschaften der einzelnen Elemente mit einem Begriff beschreiben bzw. bei Bedarf einen Verlauf markieren.

/ 3

	Preis pro Bit	Zugriffszeit	Kapazität
CPU	hoch	niedrig	niedrig
Register	$\updownarrow$	$\updownarrow$	$\updownarrow$
Cache-Speicher	$\updownarrow$	$\updownarrow$	$\updownarrow$
Arbeitsspeicher	niedrig	hoch	hoch

(c) Für ein erfolgreiches Zusammenwirken der einzelnen Funktionseinheiten sind spezielle Datensammelwege (Bus) von Bedeutung. Nennen Sie die drei typischen Datensammelwege heutiger Rechner und beschreiben Sie kurz deren Aufgabe.

/ 3

Lösung:

- Datenbus: bidirektionale Übertragung von Daten
- Adressbus: Übertragung der von der CPU berechneten Speicheradresse
- Kontrollbus / Steuerbus: Übermittlung von Steuersignalen zwischen der CPU und den anderen Funktionseinheiten

Aufgabe 9	**8 Punkte**

2016-H-9 **Assembler und Adressierung**

/ 8

Die Befehle einer Assembler-Sprache seien folgendermaßen aufgebaut:

`OpCode Q1, (Q2,) Z`

Dabei werde das Ergebnis der Operation `OpCode` basierend auf den durch `Q1` (und bei Rechenoperationen `Q2`) bezeichneten Operanden an der durch `Z` bezeichneten Adresse abgelegt.

Gegeben sei folgender Ausschnitt aus einem Assemblerprogramm.

```
STORE 1001, 1002
STORE #1002, 1001
STORE *1001, 1003
ADD 1003, 1002, *1003
SUBTRACT 40, #79, 1004
```

Dabei werden für die Adressierungsarten folgende Notationen verwendet:

- Unmittelbare Adressierung: Präfix '#'
- Direkte Adressierung: ohne Präfix
- Indirekte Adressierung: Präfix '*'

Gegeben sei weiter ein 4-zeiliger Assoziativ-Cache, der nach dem **Least-Recently-Used**-Prinzip arbeite und zu Beginn die eingetragenen Werte enthalte:

Cache-Zeile	Tag-Feld (Hauptspeicheradresse)	Datum	Zugriffs-Zeitpunkte
0	1001, 1001, 1004	40, 1002, 1	1, 3, 4, 12
1	1002	40	2, 5, 8
2	1003	40	6, 7, 9
3	40	80	10, 11

Bei jeder Assembler-Operation werden die beteiligten Operanden im Cache abgelegt.

Aktualisieren Sie die obige Tabelle mit den während des Programmlaufs anfallenden Werten. Schreiben Sie dabei die Zugriffszeitpunkte in die letzte Spalte.

Aufgabe 10	**6 Punkte**

2016-H-10 **Hauptspeicherzuweisung**

/ 6

Ein Prozessor benötigt Zugriff auf folgendes Speicherelement im virtuellen Hauptspeicher:

1 0 1 0 1 1	0 1 1 0

(a) Geben Sie sowohl die Adresse in der Seite als auch die dazugehörende Seitenadresse als Dezimalzahl an.

/ 1

Lösung:

Seitenadresse: 43

Adresse in der Seite: 6

(b) Auf welches Speicherelement muss zugegriffen werden, wenn die Zuordnungsfunktion für das gesuchte Speicherelement der Seite σ für den Seitenrahmen $F(\sigma) = 25$ berechnet? Geben Sie für das neue Speicherelement sowohl die Seitenrahmenadresse als auch die Adresse innerhalb des Seitenrahmens in Binärdarstellung an.

/ 2

Lösung:

0 1 1 0 0 1	0 1 1 0

(c) Was bedeutet es, wenn die Zuordnungsfunktion $F(\sigma)$ für die Seite σ den Ungültigkeitswert n liefert?

/ 1

Lösung:

Es tritt ein Seitenfehler auf, d.h. die Seite σ ist nicht im Hauptspeicher.

(d) Welche zwei Fälle können unterschieden werden, wenn $F(\sigma) = n$ ist? Wie ist das jeweilige Vorgehen?

/ 2

Lösung:

(1) es gibt einen freien Seitenrahmen: Seite σ wird in den Seitenrahmen kopiert

(2) es gibt keinen freien Seitenrahmen: befüllter Seitenrahmen ρ wird in den Hintergrundspeicher kopiert (Strategie zur Auswahl von ρ: least frequently used, least recently used, ...); Seite σ wird in Seitenrahmen ρ kopiert