Institut für Theoretische Informatik
Lehrstuhl Prof. Dr. D. Wagner

Übungsblatt 7

Vorlesung Theoretische Grundlagen der Informatik im WS 19/20

Ausgabe: 21. Januar 2020
Abgabe: 4. Februar 2020, 11:00 Uhr (im Kasten im UG von Gebäude 50.34)

Aufgabe 1

(3 + 1 + 1 = 5 Punkte)

Sei $G = (\Sigma, V, S, R)$ mit $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$ und $V = \{S, A, B, C, D, E, F\}$ die durch folgende Regelmenge gegebene Grammatik:

$$\begin{aligned}
S &\to C \mid FA \mid D \\
A &\to F\mathsf{c}A \mid \mathsf{d} \mid C\mathsf{b} \\
B &\to \mathsf{a} \mid C\mathsf{b}A \mid \mathsf{a}AE \\
C &\to \mathsf{c}DA \mid \mathsf{a}A \mid F\mathsf{cb} \\
D &\to \mathsf{c}F\mathsf{d} \mid FDA \\
E &\to \mathsf{a}F\mathsf{a} \mid CE\mathsf{d} \mid \mathsf{a} \\
F &\to \mathsf{b}D \mid \mathsf{b}FC
\end{aligned}$$

(a) Identifizieren Sie alle nutzlosen Variablen in G mit dem Verfahren aus der Vorlesung. Geben Sie die Grammatik G' an, die durch Entfernen der nutzlosen Variablen entsteht.

(b) Ist G' minimal in dem Sinne, dass es keine Grammatik mit weniger Variablen gibt, die $L(G')$ erzeugt? Begründen Sie Ihre Antwort.

(c) Ist $L(G)$ endlich? Begründen Sie Ihre Antwort.

Hinweis: Die Grammatik muss dazu nicht notwendigerweise in Chomsky-Normalform gebracht werden.

Lösung:

(a) **Schritt 1:** Berechne die Menge V' der Variablen, die ein Wort erzeugen können.

I. Initialisiere $Q = V' = \{A, B, E\}$.

II. Entnehme A aus Q. Ersetze in allen Regeln A durch d:

$$\begin{array}{lcl} S & \to & C \mid F\mathsf{d} \mid D \\ A & \to & F\mathsf{cd} \mid \mathsf{d} \mid C\mathsf{b} \\ B & \to & \mathsf{a} \mid C\mathsf{bd} \mid \mathsf{ad}E \\ C & \to & \mathsf{c}D\mathsf{d} \mid \mathsf{ad} \mid F\mathsf{cb} \\ D & \to & \mathsf{c}F\mathsf{d} \mid FD\mathsf{d} \\ E & \to & \mathsf{a}F\mathsf{a} \mid CE\mathsf{d} \mid \mathsf{a} \\ F & \to & \mathsf{b}D \mid \mathsf{b}FC \end{array}$$

Nun ist $V' = \{A, B, C, E\}$ und $Q = \{B, E, C\}$.

III. Entnehme B aus Q. B kommt nicht auf der rechten Seite einer Regel vor, also ergibt sich keine Änderung. Nun ist $Q = \{E, C\}$.

IV. Entnehme E aus Q. Ersetze in allen Regeln E durch a:

$$\begin{array}{lcl} S & \to & C \mid F\mathsf{d} \mid D \\ A & \to & F\mathsf{cd} \mid \mathsf{d} \mid C\mathsf{b} \\ B & \to & \mathsf{a} \mid C\mathsf{bd} \mid \mathsf{ada} \\ C & \to & \mathsf{c}D\mathsf{d} \mid \mathsf{ad} \mid F\mathsf{cb} \\ D & \to & \mathsf{c}F\mathsf{d} \mid FD\mathsf{d} \\ E & \to & \mathsf{a}F\mathsf{a} \mid C\mathsf{ad} \mid \mathsf{a} \\ F & \to & \mathsf{b}D \mid \mathsf{b}FC \end{array}$$

Nun ist $Q = \{C\}$.

V. Entnehme C aus Q. Ersetze in allen Regeln C durch ad:

$$\begin{array}{lcl} S & \to & \mathsf{ad} \mid F\mathsf{d} \mid D \\ A & \to & F\mathsf{cd} \mid \mathsf{d} \mid \mathsf{adb} \\ B & \to & \mathsf{a} \mid \mathsf{adbd} \mid \mathsf{ada} \\ C & \to & \mathsf{c}D\mathsf{d} \mid \mathsf{ad} \mid F\mathsf{cb} \\ D & \to & \mathsf{c}F\mathsf{d} \mid FD\mathsf{d} \\ E & \to & \mathsf{a}F\mathsf{a} \mid \mathsf{adad} \mid \mathsf{a} \\ F & \to & \mathsf{b}D \mid \mathsf{b}F\mathsf{ad} \end{array}$$

Nun ist Q leer und Schritt 1 endet mit $V' = \{S, A, B, C, E\}$.

D und F sind nutzlos und können entfernt werden:

$$\begin{array}{lcl} S & \to & C \\ A & \to & \mathsf{d} \mid C\mathsf{b} \\ B & \to & \mathsf{a} \mid C\mathsf{b}A \mid \mathsf{a}AE \\ C & \to & \mathsf{a}A \\ E & \to & CE\mathsf{d} \mid \mathsf{a} \end{array}$$

Schritt 2: Berechne die Menge V'' der Variablen, die von S erreichbar sind.

I. Initialisiere $V'' = \{S\}$.
II. Über S lässt sich C erreichen: $V'' = \{S, C\}$.
III. Über C lässt sich A erreichen: $V'' = \{S, A, C\}$.
IV. Über A lassen sich keine neuen Variablen erreichen. B und E sind also nutzlos.

Durch Entfernen der nutzlosen Variablen entsteht die Grammatik $G' = (\Sigma, V', S, R')$ mit $V' = \{S, A, C\}$ und Regelmenge R':

$$\begin{aligned} S &\rightarrow C \\ A &\rightarrow \mathsf{d} \mid C\mathsf{b} \\ C &\rightarrow \mathsf{a}A \end{aligned}$$

(b) Nein, $L(G')$ kann auch mit nur zwei Variablen erzeugt werden, z.B. durch $G'' = (\Sigma, V'', C, R'')$ mit $V'' = \{A, C\}$ und folgender Regelmenge R'':

$$\begin{aligned} A &\rightarrow \mathsf{d} \mid C\mathsf{b} \\ C &\rightarrow \mathsf{a}A \end{aligned}$$

(c) $L(G)$ ist nicht endlich. Betrachte dazu G''. Durch alternierendes Anwenden der Ableitungen $C \rightarrow \mathsf{a}A$ und $A \rightarrow C\mathsf{b}$ lassen sich beliebig lange Terme der Form $\mathsf{a}^i C \mathsf{b}^i$ konstruieren. Abschließendes Anwenden von $C \rightarrow \mathsf{a}A \rightarrow \mathsf{ad}$ generiert ein Wort der Form $\mathsf{a}^{i+1}\mathsf{db}^i$, also gilt $L(G) = \{\mathsf{a}^{i+1}\mathsf{db}^i \mid i \in \mathbb{N}_0\}$.

Aufgabe 2 (3 Punkte)

Sei $L = \{x \, \# \, y^{\mathsf{R}} \mid x, y \text{ sind binär kodierte Zahlen mit } x + 1 = y\}$. Geben Sie formal einen deterministischen Kellerautomaten an, der L erkennt.

In einer früheren Version des Übungsblatts wurde gefordert, dass x und y keine führenden Nullen enthalten. Diese Forderung wurde gestrichen. Für die volle Punktzahl reicht es, wenn Ihr Automat auf solchen Eingaben korrekt arbeitet, in denen keine führenden Nullen vorkommen. Eingaben mit führenden Nullen müssen nicht korrekt behandelt werden.

Lösung:
Sei $(Q = \{q_1, q_2, q_3, q_4, q_5, f\}, \Sigma = \{\mathtt{0}, \mathtt{1}, \#\}, \Gamma = \Sigma \cup \{\perp\}, q_1, \perp, \delta)$. Die Übergangsfunktion δ ist folgendermaßen definiert. Lese zunächst x und lege es auf den Stack. Ignoriere dabei führende Nullen:

$$\begin{aligned} &\delta(q_1, 0, Z) = (q_1, Z) \\ &\delta(q_1, 1, Z) = (q_2, 1Z) \\ &\delta(q_2, a, Z) = (q_2, aZ) \qquad \text{für } a = \mathtt{0}, \mathtt{1} \end{aligned}$$

Gehe in den Zustand q_3, sobald das Trennsymbol gelesen wurde:

$$\begin{aligned} &\delta(q_1, \#, Z) = (q_3, Z) \\ &\delta(q_2, \#, Z) = (q_3, Z) \end{aligned}$$

Das Suffix von x, das nur aus Einsen besteht, muss in y durch Nullen ersetzt worden sein:

$$\delta(q_3, \mathtt{0}, \mathtt{1}) = (q_3, \varepsilon)$$

Falls x nur aus Einsen besteht, muss y mit einer Eins beginnen. Danach dürfen noch führende Nullen kommen, die in Zustand q_5 überprüft werden (siehe unten).

$$\delta(q_3, \mathtt{1}, \perp) = (q_5, \perp)$$

Ansonsten wird die letzte Null in x durch eine Eins in y ersetzt:

$$\delta(q_3, \mathtt{1}, \mathtt{0}) = (q_4, \varepsilon)$$

Lese nun das restliche Wort. Ab jetzt müssen alle Symbole übereinstimmen. Wenn nach Abarbeiten des Stacks y schon komplett gelesen wurde, akzeptiere durch leeren Stack.

$$\begin{aligned}
\delta(q_4, \mathtt{0}, \mathtt{0}) &= (q_4, \varepsilon) \\
\delta(q_4, \mathtt{1}, \mathtt{1}) &= (q_4, \varepsilon) \\
\delta(q_4, \varepsilon, \perp) &= (q_4, \varepsilon)
\end{aligned}$$

Ansonsten können noch führende Nullen kommen, aber keine Eins mehr. Überprüfe dies im Zustand q_5.

$$\begin{aligned}
\delta(q_4, \mathtt{0}, \perp) &= (q_5, \perp) \\
\delta(q_5, \mathtt{0}, \perp) &= (q_5, \perp) \\
\delta(q_5, \varepsilon, \perp) &= (q_5, \varepsilon)
\end{aligned}$$

Alle nicht explizit angegebenen Übergänge sollen ohne Veränderungen des Stacks in den Fehlerzustand f gehen.

Aufgabe 3

(2 + 2 + (2) + 2 + 1 = 7 + (2) Punkte)

Aus der Vorlesung wissen Sie, dass eine Linksableitung eine Ableitung ist, bei der in jedem Schritt das linkeste Nichtterminalsymbol des aktuellen Wortes aus $(\Sigma \cup V)^*$ abgeleitet wird. Sei $G = (\Sigma, V, S, R)$ eine kontextfreie Grammatik. Zu jedem Wort $w \in L(G)$ gibt es eine Linksableitung bezüglich G.

(a) Geben Sie einen nichtdeterministischen Kellerautomaten an, der $L(G)$ erkennt.

Anmerkung: Konstruieren Sie keine zu G äquivalenten Grammatiken, sondern denken Sie an das Konzept der Linksableitung.

Lösung:
Wähle $Q = \{q, f\}$ als Zustandsmenge, q als Startzustand, Σ als Eingabealphabet, $\Gamma = \Sigma \cup V$ als Stackalphabet und S als Stackinitialisierung.

Die Zustandsüberführungsfunktion δ sei folgendermaßen definiert. Für jede Ableitungsregel $A \to v$ gilt $\delta(q, \varepsilon, A) \to (q, v)$. Außerdem sei $\delta(q, a, a) \to (q, \varepsilon)$ für jedes $a \in \Sigma$. Alle anderen Übergänge gehen ohne Veränderung des Stacks in den Fehlerzustand f über.

Der Automat soll durch leeren Stack akzeptieren.

(b) Beweisen Sie, dass Ihr Kellerautomat genau $L(G)$ erkennt.

Lösung:
Sei $w \in L(G)$. Man orientiere sich an einer Linksableitung von w. Der Inhalt des Stacks soll stets auf das noch zu lesende Teilwort abzuleiten sein. Ist das oberste Symbol auf dem Stack ein Terminalsymbol a, so kann der Übergang $\delta : (q, a, a) \mapsto (q, \varepsilon)$ angewendet werden, um das Symbol a zu lesen und vom Stack zu entfernen. Ansonsten ist das oberste Symbol auf dem Stack ein Nichtterminalsymbol. Nach Konstruktion ist dies das linkeste Nichtterminalsymbol in der Linksableitung. Dieses Symbol kann dann mit ε-Übergängen weiter abgeleitet werden.

Dieselbe Idee „rückwärts gedacht" können wir anwenden, um aus einer akzeptierenden Abarbeitung eines Wortes w durch den Automaten eine Ableitung von w bezüglich G zu erzeugen.

In Anbetracht von (a) und (b) stellt man sich die Frage, wozu die Greibach-Normalform überhaupt nützlich ist. Das TGI-Team hat leider keine überzeugende Antwort. Und Sie?

(c) *Bonusaufgabe:* Erklären Sie möglichst überzeugend, was der Nutzen der Greibach-Normalform ist.

Diese Bonusaufgabe wird mit maximal 2 Punkten bewertet.

Wir versuchen nun, den Begriff der Linksableitung sinnvoll für kontextsensitive Sprachen zu erweitern: Eine *Präfixableitung* ist eine Ableitung, bei der in jedem Schritt ein Präfix des aktuellen Wortes aus $(\Sigma \cup V)^*$ abgeleitet wird. Eine Linksableitung ist also ein Spezialfall der Präfixableitung, bei der jedes Präfix Länge Eins hat.

(d) Sei G eine kontextsensitive Grammatik, sodass zu jedem Wort $w \in L(G)$ eine Präfixableitung existiert. Erklären Sie, wie Sie Ihren Ansatz aus Teilaufgabe (a) erweitern können, um einen Kellerautomaten zu konstruieren, der $L(G)$ erkennt.

Lösung:
Sei k die maximale Anzahl von Symbolen, die auf der linken Seite einer Ableitungsregel in G stehen. Anstatt nur das oberste Symbol auf dem Stack für die Ableitung in Betrachtung zu ziehen, müssen ggf. die obersten k Symbole auf dem Stack in Betrachtung gezogen werden. Um dies zu ermöglichen, werden $|\Gamma|^k$ neue Zustände verwendet. Durch Lesen und Entfernen der obersten k Symbole können diese im Zustand gespeichert werden. Dann können Übergänge ausgeführt werden, die Ableitungen entsprechen, bei denen auf der linken Seite bis zu k Symbole stehen.

(e) Geben Sie eine kontextsensitive Grammatik G und ein Wort $w \in L(G)$ an, sodass w nicht durch eine Präfixableitung abgeleitet werden kann.

Lösung:
Wähle $G = (\Sigma = \{\mathtt{a}, \mathtt{c}\}, V = \{S, A, B, C\}, S, R = \{S \to AB, B \to C, AC \to \mathtt{ac}\})$. Das Wort $\mathtt{ac}$ hat genau eine Ableitung, nämlich $S \to AB \to AC \to \mathtt{ac}$, was im zweiten Schritt nicht einer Präfixableitung entspricht.

Aufgabe 4

(1 + 2 + 2 + 1 = 6 Punkte)

In Vorlesung und Übung wurden bisher *nichtdeterministische* Kellerautomaten betrachtet. Sei $\mathcal{A}$ nun ein *deterministischer* Kellerautomat, d.h. es gibt keine ε-Übergänge und für jedes Eingabesymbol ist der Übergang von $\mathcal{A}$ eindeutig definiert. Gehen Sie davon aus, dass das Eingabealphabet

Σ mindestens zwei Symbole enthält, dass $\mathcal{A}$ durch akzeptierende Endzustände akzeptiert und dass der Stack von $\mathcal{A}$ niemals leer ist.

Definiere für solche deterministischen Kellerautomaten das Konzept der *Essenz* folgendermaßen. Für jedes Wort w existiert ein[1] Wort w', sodass die Höhe des Stacks nach Verarbeitung von ww' minimal unter allen möglichen w' ist. Ist nach Abarbeitung von ww' die Höhe des Stacks h, so ist also für jedes Wort v die Höhe des Stacks nach Abarbeitung von $ww'v$ mindestens h. Sei $\alpha_1\alpha_2\ldots\alpha_h$ der Inhalt des Stacks nach Abarbeitung von ww'. Das oberste Symbol auf dem Stack ist $\hat{\alpha} := \alpha_h$. Der aktuelle Zustand sei q. Bezeichne $(q, \hat{\alpha})$ als die *Essenz* von w.

(a) Zeigen Sie, dass es mindestens zwei unterschiedliche Wörter $w_1 \neq w_2$ mit der gleichen Länge und derselben Essenz gibt.

Hinweis: Wie viele unterschiedliche Essenzen kann es höchstens geben?

(b) Seien w_1, w_2 Wörter mit derselben Essenz $(q, \hat{\alpha})$. Zeigen Sie, dass dann für jedes Wort $v \in \Sigma^\star$ gilt:

$$w_1 w_1' v \in L(\mathcal{A}) \iff w_2 w_2' v \in L(\mathcal{A})$$

Hinweis: Erinnern Sie sich daran, dass die Konfiguration (q, v, α) eines Kellerautomaten aus drei Teilen besteht: dem aktuellen Zustand q, dem Teil der Eingabe $v \in \Sigma^\star$, der noch nicht gelesen wurde, und dem Stackinhalt $\alpha \in \Gamma^\star$.

(c) Definieren Sie die Sprache, die genau aus allen Palindromen besteht. Nutzen Sie das Ergebnis der letzten Teilaufgabe, um zu zeigen, dass $\mathcal{A}$ die Palindromsprache nicht erkennen kann.

(d) Bestimmen Sie das minimale k, sodass deterministische Kellerautomaten alle Sprachen von Typ-k erkennen können. Begründen Sie kurz.

Lösung:

(a) Es gilt $\hat{\alpha} \in \Gamma$ und $q \in Q$. Damit kann es höchstens $|\Gamma \times Q|$, also endlich viele Essenzen geben. Da $|\Sigma^n| \geq 2^n$ ist, gibt es ein $n_0 > \lceil \log(|\Gamma \times Q|) \rceil$, sodass mehr Wörter der Länge n_0 als Essenzen existieren. Also muss es mindestens zwei unterschiedliche Wörter mit gleicher Länge und derselben Essenz geben.

(b) Betrachte für $i \in \{1, 2\}$ die Konfigurationen (q_i, v_i, α_i) von $\mathcal{A}$ nach der Abarbeitung von $w_i w_i'$. Da w_1 und w_2 dieselbe Essenz haben, gilt $q_1 = q_2$. Außerdem ist das oberste Symbol auf dem Stack $\hat{\alpha}$. Der Teil v_i der Eingabe, der noch nicht gelesen wurde, ist in beiden Fällen v. Der einzige Unterschied zwischen den Konfigurationen kann also im Stackinhalt außer dem obersten Stacksymbol $\hat{\alpha}$ liegen. Da der Stack aber nach Definition von w_i' bei der weiteren Abarbeitung nicht schrumpfen kann, kann der Inhalt des Stacks bis auf das oberste Symbol nicht gelesen, geschrieben oder geändert werden.

Damit ist das Verhalten (insbesondere das Akzeptanzverhalten) von $\mathcal{A}$ nach dem Lesen von $w_i w_i'$ identisch für jedes $v \in \Sigma^\star$.

(c) Die Palindromsprache ist $L_P = \{w \in \Sigma^\star \mid w = w^{\mathsf{R}}\}$. Nach Aufgabenteil (a) gibt es Wörter $w_1 \neq w_2$ mit gleicher Länge und derselben Essenz. Nach Aufgabenteil (b) gilt dann

$$w_1 w_1' (w_1 w_1')^{\mathsf{R}} \in L(\mathcal{A}) \iff w_2 w_2' (w_1 w_1')^{\mathsf{R}} \in L(\mathcal{A}).$$

[1] Das Wort w' ist nicht notwendigerweise eindeutig.

Da $w_1w_1'(w_1w_1')^{\mathsf{R}}$ ein Palindrom ist, wegen $w_1 \neq w_2$ das Wort $w_2w_2'(w_1w_1')^{\mathsf{R}}$ jedoch nicht, kann $\mathcal{A}$ nicht die Palindromsprache erkennen.

(d) Die Palindromsprache ist kontextfrei. Deterministische Kellerautomaten können also nicht die Typ-2-Sprachen erkennen. Deterministische Kellerautomaten können aber trivialerweise deterministische endliche Automaten simulieren. Deterministische Kellerautomaten können also die Typ-3-Sprachen erkennen.

Aufgabe 5

(1 + 2 + 3 + 1 = 7 Punkte)

Sie und Ihre Kommilitonen Lena und Leopold haben jeweils einen nichtdeterministischen Kellerautomaten erworben. Lena klaut nun den Stack von Leopolds Kellerautomat und baut ihn in ihrem eigenen Kellerautomat als Zweitstack ein, den sie unabhängig vom Erststack benutzen kann. Wir betrachten nun also drei verschiedene Maschinenmodelle: Leopolds Kellerautomat NPDA_0 hat überhaupt keinen Stack mehr, Ihr Kellerautomat NPDA_1 ist ein ganz normaler Kellerautomat mit einem Stack, und Lenas Kellerautomat NPDA_2 hat zwei Stacks.

(a) Leopold fragt sich, ob sein Kellerautomat NPDA_0 jetzt weniger Sprachen erkennen kann als Ihr Kellerautomat NPDA_1. Beantworten Sie diese Frage, indem Sie begründen, welche Klasse von Sprachen von NPDA_0 erkannt werden können.

Lena hofft, dass sie mit ihrem Kellerautomat NPDA_2 mehr Sprachen erkennen kann als Ihr Kellerautomat NPDA_1.

(b) Zeigen Sie, dass Lena recht hat, indem Sie eine Sprache angeben, die von NPDA_2 erkannt wird, von NPDA_1 aber nicht erkannt werden kann. Begründen Sie beides!

(c) Welche Klasse von Sprachen kann NPDA_2 erkennen?
Hinweis: Kann NPDA_2 ein mächtigeres Berechnungsmodell simulieren?

(d) Kann Lena ihren Kellerautomaten noch mächtiger machen, indem sie einen dritten Stack einbaut? Begründen Sie!

Lösung:

(a) Aus der Vorlesung ist bekannt, dass NPDA_1 genau die kontextfreien Sprachen erkennen kann. Ohne einen Stack ist Leopolds Kellerautomat NPDA_0 ein nichtdeterministischer endlicher Automat, der bekannterweise genau die regulären Sprachen erkennen kann. Da die Menge der regulären Sprachen eine echte Teilmenge der kontextfreien Sprachen ist, kann NPDA_0 tatsächlich weniger Sprachen erkennen als NPDA_1.

(b) In Aufgabe 6 von Übungsblatt 6 wurde gezeigt, dass die Sprache

$$L = \{\mathsf{0}^n\mathsf{1}^n\mathsf{2}^n \mid n \in \mathbb{N}\}$$

nicht kontextfrei ist. Da NPDA_1 genau die kontextfreien Sprachen erkennen kann, kann er insbesondere L nicht erkennen. Lenas Kellerautomat NPDA_2 kann L allerdings erkennen. Sie kann z.B. so vorgehen, dass sie für jede $\mathsf{0}$ je ein Zeichen auf beide Stacks legt, für jede $\mathsf{1}$ ein Zeichen vom ersten Stack löscht und für jede $\mathsf{2}$ ein Zeichen vom zweiten Stack löscht.

(c) Lenas Kellerautomat NPDA_2 mit zwei Stacks L, R ist genauso mächtig wie eine Turingmaschine, kann also genau die semi-entscheidbaren Sprachen erkennen. Einerseits kann eine Turingmaschine mit drei Bändern einen Kellerautomaten mit zwei Stacks simulieren, indem sie jeden Stack auf einem eigenen Band speichert. Andererseits kann auch ein Kellerautomat $\mathcal{P}$ mit zwei Stacks eine Turingmaschine $\mathcal{M}$ wie folgt simulieren. Zuerst schreibt $\mathcal{P}$ die Eingabe w auf den Stack R. Anschließend werden alle Symbole von R nach L verschoben. Somit steht das erste Zeichen von w an erster Stelle von L. Diese Position stellt nun den Kopf der Turingmaschine dar. Der Stack L kodiert somit das Wort links vom Kopf und das Zeichen auf dem Lesekopf und R das Wort rechts vom Lesekopf. Eine Schreiboperation ist also eine Folge von einer `pop`- und einer `push`-Operation auf L. Ein Schritt nach links ist eine `pop`-Operation auf L und eine `push`-Operation auf R, entsprechend umgekehrt wird eine Bewegung nach rechts simuliert.

(d) Nein, denn ein Kellerautomat mit drei Stacks lässt sich ebenfalls mit einer Turingmaschine simulieren.

Aufgabe 6 (1 + 1 + 1 = 3 Punkte)

Hinweis: Die für diese Aufgabe notwendigen Grundlagen zur Huffman-Kodierung werden in der Vorlesung vom 28.1. vorgestellt.

Die folgende Tabelle gibt eine Häufigkeitsverteilung für die Zeichen des Alphabets $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{e}, \mathsf{f}, \mathsf{g}, \mathsf{h}\}$ an.

a	b	c	d	e	f	g	h
17%	5%	12%	23%	8%	11%	18%	6%

(a) Nehmen Sie an, Sie möchten Wörter über dem Alphabet Σ binär kodieren. Was ist die erwartete Länge der Kodierung eines Wortes in Abhängigkeit von dessen Länge n, wenn Sie die obige Verteilung und folgende naive Binärkodierung der Zeichen zugrunde legen?

a	b	c	d	e	f	g	h
000	001	010	011	100	101	110	111

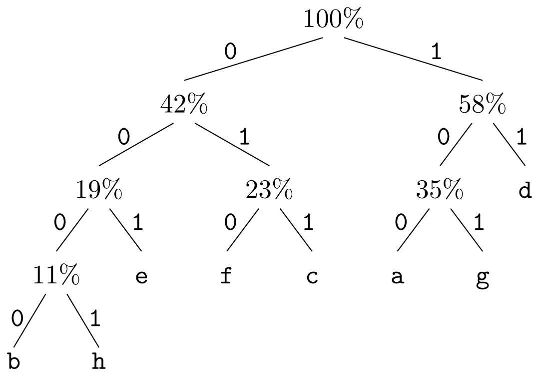
(b) Konstruieren Sie den Huffman-Kodierungsbaum zu der obigen Häufigkeitsverteilung.

(c) Was ist die erwartete Länge der Kodierung eines Wortes in Abhängigkeit von dessen Länge n, wenn sie die obige Verteilung und Ihre Huffman-Kodierung zugrunde legen?

Lösung:

(a) Die erwartete Länge ist $3n$.

(b) Siehe folgenden Baum:



(c) Die erwartete Länge ist

$$\left(11\% \cdot 4 + 23\% \cdot 2 + (100\% - 11\% - 23\%) \cdot 3\right) n = 2,88n.$$