Kurs 01654 — Grundlagen der Theoretischen Informatik B

Lösungshinweise zur Klausur vom 17. 09. 2011

Aufgabe 1 (a) Es gilt für alle $n \in \mathbb{N}$:

$$f(n) + g(n) \leq \max\{f(n), g(n)\} + \max\{f(n), g(n)\} = 2 \cdot h(n).$$

Also ist $(f+g)(n) \leq 2 \cdot h(n) + 2$ für alle n, d. h. $f + g \in O(h)$ und damit $O(f+g) \subseteq O(h)$.
Andererseits gilt für alle $n \in \mathbb{N}$ offensichtlich auch

$$\max\{f(n), g(n)\} \leq f(n) + g(n).$$

D. h. $h \in O(f+g)$, also $O(h) \subseteq O(f+g)$. Es folgt $O(h) = O(f+g)$.

(b) Angenommen, es gälte $n \cdot \log n \cdot \log\log n \in O(n \cdot \log n)$. Dann gibt es ein $c \in \mathbb{N}$, $c \geq 1$, so dass mit für alle $n \in \mathbb{N}$ gilt:

$$n \cdot \log n \cdot \log\log n \leq c \cdot n \cdot \log n + c.$$

Damit gilt für alle n, für die $n \cdot \log n \geq c$ ist: $\log\log n \leq c + 1$.
Wir betrachten die Zahl $n_0 := 2^{2^{2c}}$. Dann ist $\log n_0 = 2^{2c}+1$, also $\log\log n_0 = 2c + 1$. Da $n_0 \cdot \log n_0 \geq c$ ist, folgt

$$2c + 1 \leq c + 1.$$

Wegen $c \geq 1$ ergibt sich daraus ein Widerspruch. Folglich gilt $n \cdot \log n \cdot \log\log n \notin O(n \cdot \log n)$.

Aufgabe 2 (a) Wir beschreiben zunächst die Arbeitsweise einer geeigneten Turingmaschine M. Es sei 0^n die Eingabe für M.

1. Berechne durch duales Inkrementieren $a = d(n)$ auf Band 2;
2. Berechne durch duales Inkrementieren $b = d(m)$, wobei $m = \lg(a)$, auf Band 3;
3. Positioniere den Kopf von Band 2 auf das (von links) erste Symbol $\neq$ B; schreibe ein B auf Band 4;

4. WHILE *aktuelles Symbol auf Band* $2 \neq \mathtt{B}$ DO
   Kopiere die Eingabe 0^n rechts an den aktuellen Inhalt von Band 4;
   Gehe auf Band 2 einen Schritt nach rechts
5. Positioniere den Kopf von Band 3 auf das (von links) erste Symbol $\neq \mathtt{B}$;
6. WHILE *aktuelles Symbol auf Band* $3 \neq \mathtt{B}$ DO
   Kopiere den Inhalt $\in \{0\}^*$ von Band 4 auf das Ausgabeband;
   Gehe auf Band 3 einen Schritt nach rechts.

Die Korrektheit der Maschine braucht hier nicht bewiesen zu werden. Zur Abschätzung des Zeitbedarfs geben wir für die einzelnen Befehle ein entsprechendes „$O(\cdots)$“ an:

1: $O(n)$ (nach Beispiel 2.2.2 des Kurstextes);

2: $O(\log n)$ (ebenfalls nach Beispiel 2.2.2 des Kurstextes);

3: $O(\log n)$;

4: $O(n \cdot \log n)$;

5: $O(\log \log n)$;

6: $O(n \cdot \log n \cdot \log \log n)$.

Der letzte Schritt dominiert die Rechenzeit. Es folgt

$$\tilde{t}_M \in O\left(n \cdot \log n \cdot \log \log n\right).$$

Damit ist f zeitkonstruierbar.

(b) Wir wollen den Zeithierarchiesatz (Satz 3.4.2 des Kurstextes) anwenden und überprüfen dazu dessen Voraussetzungen. Man beachte, dass gegenüber der Formulierung im Kurstext hier f und g vertauscht sind.

(1) Wie im Aufgabenteil (a) gezeigt, ist die Funktion f zeitkonstruierbar.

(2) Es gilt offensichtlich $n \in O\left(n \cdot \log n \cdot \log \log n\right)$.

(3) Dies zeigt auch $g \in O(f)$ für die durch $g(n) := n$ für alle $n \in \mathbb{N}$ definierte Funktion.

(4) Schließlich trifft $f \notin O\left(g(n) \cdot \log g(n)\right) = O\left(n \cdot \log n\right)$ zu, wie in Aufgabe 1 (b) gezeigt.

Damit sind alle Voraussetzungen des Zeithierarchiesatzes erfüllt. Es folgt die Behauptung.

(c) Solch eine Sprache L existiert, und deren Existenz folgt sofort aus dem Bandhierarchiesatz 3.4.5. Die Gültigkeit von dessen Voraussetzungen wird im Folgenden begründet.

(1) Offensichtlich ist $\log \in O(f)$.

(2) Nach Aufgabenteil (a) ist f zeitkonstruierbar, also laut Kurstext (s. KE 2, Seite 11, Bemerkung im Anschluss an Definition 3.3.1) auch bandkonstruierbar.

(3) Für die durch $h(n) := n \cdot \log n$ für alle $n \in \mathbb{N}$ definierte Funktion gilt offensichtlich $h \in O(f)$.

(4) $f \notin O(h)$ gilt gemäß Item (4) aus (b).

Damit sind alle Voraussetzungen des Bandhierarchiesatzes erfüllt, und es folgt $\mathsf{BAND}(h) \subsetneq \mathsf{BAND}(f)$.

Aufgabe 3 (a) (1) Die Komplexitätsklasse $\mathsf{NZEIT}(f)$ ist nur für berechenbare Funktionen $f : \mathbb{N} \to \mathbb{N}$ definiert.

Falsch. $f : \mathbb{N} \to \mathbb{N}$ kann nach Definition 4.1.2 eine beliebige (totale) Funktion sein.

(2) Jede Kontrollturingmaschine kann durch eine nichtdeterministische Turingmaschine ohne wesentlichen Mehrbedarf an Zeit und Band simuliert werden.

Richtig. Siehe Satz 4.1.4.

(3) Für jedes $f : \mathbb{N} \to \mathbb{N}$ besteht $\mathsf{NZEIT}(f)$ nur aus entscheidbaren Sprachen.

Richtig. Dies folgt aus Satz 4.2.1.

(b) (1) Die Komplexitätsklasse NP ist die Vereinigung aller $\mathsf{NZEIT}(f)$, wobei $f : \mathbb{N} \to \mathbb{N}$ total ist.

Falsch. Die Zeitschranken müssen Polynome sein.

(2) Die Menge Σ^* ist ein Element von NP.

Richtig. Es gilt sogar $\Sigma^* \in \mathsf{P}$.

(3) L ist polynomiell reduzierbar auf M ($L \leq_{pol} M$), wenn es eine Funktion $f \in \mathsf{FP}$ gibt, die genau die Elemente aus L nach M abbildet.

Richtig. Dies ist nur eine Umformulierung von Definition 4.3.2, Item 2.

(4) Wenn $L \leq_{pol} M$ und $L \in \mathsf{NP}$ gilt, dann ist $M \in \mathsf{NP}$.
Falsch. Dies folgt schnell aus der Tatsache, dass es Sprachen gibt, die nicht in NP liegen.

(5) Ist L NP-hart und gilt $L \in \mathsf{P}$, dann ist L NP-vollständig.
Richtig. Folgt aus $\mathsf{P} \subseteq \mathsf{NP}$; s. Definition 4.3.5.

(6) RUCKSACK ist polynomiell reduzierbar auf SAT.
Richtig. Beide Probleme sind NP-vollständig.

(c) (1) NBAND(n) besteht nur aus rekursiv-aufzählbaren Sprachen.
Richtig. Nach Satz 4.2.3 (1) sind alle Sprachen aus NBAND(n) entscheidbar, also erst recht rekursiv-aufzählbar.

(2) Es gilt PSPACE = NPSPACE.
Richtig. Siehe Satz 4.2.3 (2).

(3) Jede Sprache aus NLOGSPACE ist polynomiell reduzierbar auf GAP.
Richtig. Nach Satz 6.1.1 ist GAP NLOGSPACE-vollständig bez. log-Reduzierbarkeit, woraus mit Selbsttestaufgabe 4.3.4 (1) die Behauptung folgt.

Aufgabe 4 (a) Wir modifizieren die Kontrollturingmaschine T aus dem Kurstext, die die Menge RUCKSACK erkennt (s. KE 3, Seite 11), geeignet.
Es sei x die Eingabe auf Band 1, und $y = y_1 y_2 \ldots y_n \ldots$ sei die Hilfseingabe auf Band 0. Dann arbeite die Kontrollturingmaschine T' nach folgender Vorschrift:

1. Kopiere $d(N)$ auf Band 2; schreibe $d(0)$ auf Band 3;
2. Füge eine 0 rechts an das Wort auf Band 2;
3. Lies ein Zeichen $y_i \in \{0, 1\}$ der Hilfseingabe;
4. Falls $y_i = 1$, addiere die nächste Dualzahl $d(k_i)$ zum Inhalt von Band 3;
   Falls $y_i \neq 1$, überlies k_i;
   Falls die Eingabe zu Ende ist, gehe zu 5, sonst zu 3
5. Vergleiche die Inhalte von Band 2 und Band 3; falls gleich, HALT +, sonst HALT − .

Die Korrektheit der hierdurch gegebenen Maschine erhält man ebenso wie diejenige von T im Kurstext. Die Schleife 3 – 4 wird höchstens $n = \lg(x)$-mal durchlaufen, und jeder Durchlauf kostet höchstens $O(n)$ Rechenschritte. Die Schranke $O(n)$ wird auch bei der Ausführung der übrigen Befehle 1, 2 und 5 nicht überschritten. Damit folgt RR $\in$ NP.

(b) Es sei $\Sigma := \{0, 1, \#\}$. Wir definieren $f : \Sigma^* \to \Sigma^*$ für alle $x \in \Sigma^*$ durch

$$f(x) := \begin{cases} d(2N)\#d(k_1)\#\cdots\#d(k_n) & \text{falls } \begin{cases} \exists\, N, n, k_1, \ldots, k_n \in \mathbb{N},\ n \geq 1\,. \\ x = d(N)\#d(k_1)\#\cdots\#d(k_n) \end{cases} \\ \varepsilon & \text{sonst.} \end{cases}$$

Dann gilt für alle $x \in \Sigma^*$:

$$\begin{aligned} x \in \mathrm{RR} &\iff \begin{cases} \exists\, N, n, k_1, \ldots, k_n \in \mathbb{N},\ n \geq 1\,. \\ x = d(N)\#d(k_1)\ldots\#d(k_n) \wedge \\ \exists\, M \subseteq \{1, \ldots, n\}\,.\ \sum_{i \in M} k_i = 2 \cdot N \end{cases} \\ &\iff \begin{cases} \exists\, N, n, k_1, \ldots, k_n \in \mathbb{N},\ n \geq 1\,. \\ f(x) = d(2N)\#d(k_1)\ldots\#d(k_n) \wedge \\ \exists\, M \subseteq \{1, \ldots, n\}\,.\ \sum_{i \in M} k_i = 2 \cdot N \end{cases} \\ &\iff f(x) \in \mathrm{RUCKSACK}. \end{aligned}$$

Dies zeigt

$$\mathrm{RR} \leq_{pol} \mathrm{RUCKSACK},$$

denn offenbar ist f in Linearzeit berechenbar. (f testet die Gestalt der Eingabe und fügt ggf. lediglich eine 0 vor dem ersten # ein.) Damit folgt die NP-Härte von RUCKSACK aus derjenigen von RR (vergl. Lemma 4.3.3 und Definition 4.3.5). Da laut Kurstext RUCKSACK $\in$ NP gilt, erhalten wir auf diesem Wege die NP-Vollständigkeit dieser Sprache.

Aufgabe 5

Da die Grammatik klar ist, lassen wir den unteren Index 'G' bei der zugehörigen Ableitungsrelation hier weg.

(a) Induktionsanfang $k = 0$:

Es sei $w \in \Delta^*$, und es gelte $\mathtt{S} \xrightarrow{0} w$. Dann folgt $w = \mathtt{S}$. Natürlich ist $\#_{\mathtt{a}}(\mathtt{S}) = \#_{\mathtt{b}}(\mathtt{S})$. Damit gilt die Behauptung in diesem Falle.

Induktionsschluss $k \to k+1$:

Es sei wiederum $w \in \Delta^*$, und es gelte $\mathtt{S} \xrightarrow{k+1} w$. Dann gibt es nach Definition 7.1.6(2) ein Wort $v \in \Delta^*$ mit $\mathtt{S} \xrightarrow{k} v \xrightarrow{1} w$. Auf v ist die Induktionsvoraussetzung anwendbar. Danach gilt

$$\#_{\mathtt{a}}(v) = \#_{\mathtt{b}}(v).$$

Aus $v \xrightarrow{1} w$ folgt, dass w aus v durch Anwendung einer der drei Regeln der Grammatik entsteht. Da dabei entweder sowohl ein a als auch ein b oder weder ein a noch ein b hinzukommen, bleibt die behauptete Eigenschaft erhalten, d. h., es gilt auch $\#_{\mathtt{a}}(w) = \#_{\mathtt{b}}(w)$. Dies war zu zeigen.

(b) Es sei $w \in L(G)$. Dann gibt es ein $n \in \mathbb{N}$ mit $\mathtt{S} \xrightarrow{n} w$. Nach Aufgabenteil (a) folgt $\#_{\mathtt{a}}(w) = \#_{\mathtt{b}}(w)$. Da zudem $w \in \Sigma^*$ ist, ergibt sich $w \in L$, also, wie gewünscht, $L(G) \subseteq L$.

(c) Induktionsanfang $k = 0$:

Es sei $w \in L$, und es gelte $\lg(w) \leq 0$. Dann ist $\lg(w) = 0$, also $w = \varepsilon$. Für dieses Wort gilt sicherlich $\#_{\mathtt{a}}(w) = \#_{\mathtt{b}}(w)$. Also ist $w = \varepsilon \in L(G)$.

Induktionsschluss $k \to k+1$:

Wiederum sei $w \in L$, aber jetzt gelte $\lg(w) \leq k+1$. O. B. d. A. sei $\lg(w) > 0$ (andernfalls s. Induktionsanfang). Nach der in der Aufgabenstellung für L formulierten Eigenschaft gibt es dann Wörter $u, v \in L$, so dass $w = \mathtt{a}u\mathtt{b}v$ oder $w = \mathtt{b}u\mathtt{a}v$ zutrifft. Da $\lg(u) \leq k$ und $\lg(v) \leq k$ gilt, ist sowohl auf u als auch auf v die Induktionsvoraussetzung anwendbar. D. h., $u, v \in L(G)$, oder, m. a. W.,

$$\mathtt{S} \xrightarrow{*} u \quad \text{und} \quad \mathtt{S} \xrightarrow{*} v.$$

Es folgt

$$\mathtt{S} \to \mathtt{bSaS} \xrightarrow{*} \mathtt{b}u\mathtt{aS} \xrightarrow{*} \mathtt{b}u\mathtt{a}v \quad \text{und} \quad \mathtt{S} \to \mathtt{aSbS} \xrightarrow{*} \mathtt{a}u\mathtt{bS} \xrightarrow{*} \mathtt{a}u\mathtt{b}v.$$

Somit gilt $\mathtt{b}u\mathtt{a}v \in L(G)$ und $\mathtt{a}u\mathtt{b}v \in L(G)$, also $w \in L(G)$. Damit ist die Behauptung bewiesen.

(d) Sei $w \in L$ beliebig. Setze in (c) $k := \lg(w)$. Dann ist die Prämisse der dort behaupteten Implikation wahr. Es folgt $w \in L(G)$. Daraus erhalten wir $L \subseteq L(G)$.

(e) Nach (b) und (d) ist $L = L(G)$. Da die Grammatik G offensichtlich kontextfrei ist, ist L eine kontextfreie Sprache.

Aufgabe 6

Wir führen einen Widerspruchsbeweis und nehmen dazu an, L sei regulär. Dann gibt es nach dem Pumping-Lemma für reguläre Mengen (Satz 8.4.3) ein $p \in \mathbb{N}$, so dass für alle $t, z, \bar{t} \in \Sigma^*$ mit $tz\bar{t} \in L$ und $\lg(z) = p$ Wörter $u, v, w \in \Sigma^*$ derart existieren, dass $z = uvw$, $v \neq \varepsilon$ und $\forall i \geq 0 \,.\, tuv^i w\bar{t} \in L$ gilt.

Wir wählen nun ein $p \in \mathbb{N}$ mit dieser Eigenschaft und setzen $t := \mathtt{a}^p$, $z := \mathtt{b}^p$ und $\bar{t} := \varepsilon$. Dann ist offensichtlich $tz\bar{t} = \mathtt{a}^p\,\mathtt{b}^p \in L$ und $\lg(z) = p$.

Daher gibt es $u, v, w \in \Sigma^*$, so dass die obigen Bedingungen erfüllt sind. Da v ein Teilwort von $z = \mathtt{b}^p$ ist, können wir schreiben:

$$v = \mathtt{b}^{\lg(v)} \text{ und } tuv^0w\bar{t} = \mathtt{a}^p\,\mathtt{b}^{p-\lg(v)}.$$

Da $\lg(v) > 0$ (wegen $v \neq \varepsilon$) ist, treten in dem Wort $tuv^0w\bar{t}$ weniger b's als a's auf. D. h., dieses Wort ist *nicht* aus L. Dies widerspricht aber $\forall i \geq 0\,.\, tuv^iw\bar{t} \in L$. Also ist unsere Annahme falsch. M. a. W., L ist nicht regulär.

Aufgabe 7 (a) (1) Jede reguläre Menge über Σ ist eine Typ-0-Sprache.

Richtig. Jede reguläre Menge ist eine Typ-3-Sprache, und die Typ-3-Sprachen sind einer Teilklasse der Typ-0-Sprachen (s. Satz 9.4.8).

(2) Die Sprache $L = \{\mathtt{a}^l\mathtt{b}^m\mathtt{a}^n \mid l, m, n \in \mathbb{N}\}$ über dem Alphabet $\Gamma := \{\mathtt{a}, \mathtt{b}\}$ ist eine Typ-3-Sprache.

Richtig. Man kann leicht einen endlichen Automaten angeben, der die Sprache akzeptiert.

(3) Die endlichen Sprachen über Σ sind unter Komplementbildung abgeschlossen.

Falsch. Die Komplemente der endlichen Teilmengen von Σ^* sind unendliche Mengen.

(4) Die Typ-3-Sprachen über Σ sind unter Vereinigungsbildung abgeschlossen.

Richtig. Siehe Satz 8.4.1.

(5) Das Äquivalenzproblem für reguläre Mengen über Σ ist unentscheidbar.

Falsch. Siehe Satz 8.4.2.

(b) (1) Zu jeder kontextfreien Grammatik G gibt es eine Grammatik G' in Chomsky-Normalform mit $L(G) = L(G')$.

Falsch. Dies gilt nicht, falls $\varepsilon \in L(G)$.

(2) Gibt es zu einer Sprache $L \subseteq \Sigma^*$ ein $p \in \mathbb{N}$, so dass man jedes Wort $z \in L$ mit $\lg(z) \geq p$ so in Teilworte u, v, w, x, y zerlegen kann, dass (i) $z = uvwxy$, (ii) $vx \neq \varepsilon$, (iii) $\lg(vwx) \leq p$ und (iv) $\forall i \geq 0.\, uv^iwx^iy \in L$ gilt, dann ist L kontextfrei.

Falsch. Nur die umgekehrte Implikation gilt.

(3) Jede Sprache, die von einem determinierten Kellerautomaten erkannt wird, ist kontextfrei.

Richtig. Die deterministisch kontextfreien Sprachen sind eine Teilklasse aller kontextfreien Sprachen.

(4) Die deterministisch kontextfreien Sprachen sind unter Komplementbildung abgeschlossen.

Richtig. Siehe Satz 9.6.5.

(5) Das Äquivalenzproblem für kontextfreie Sprachen ist unentscheidbar.

Richtig. Siehe Satz 9.7.3.

(c) (1) GAP ist eine Typ-1-Sprache.

Richtig. Siehe Satz 7.4.1.

(2) Es sei $L \subseteq \Sigma^*$ eine kontextsensitive Sprache. Dann ist ihr Komplement $\Sigma^* \setminus L$ eine entscheidbare Menge.

Richtig. Nach Korollar 7.4.2 ist L entscheidbar. Damit ist auch $\Sigma^* \setminus L$ entscheidbar.

(3) Jede rekursiv-aufzählbare Menge $L \subseteq \Sigma^*$ ist eine Typ-0-Sprache.

Richtig. Siehe Satz 7.2.2.

(4) Das Wortproblem für Typ-0-Sprachen ist entscheidbar.

Falsch. Dies ist (im Wesentlichen) das Halteproblem für Turingmaschinen, und das ist bekanntermaßen unentscheidbar.