ETH zürich
Departement Informatik

Theoretische Informatik

Prof. Dr. Juraj Hromkovič
http://www.ita.inf.ethz.ch/theoInf16

Lösungsvorschläge – Blatt Z1

Zürich, 2. Dezember 2016

Lösung zu Aufgabe Z1

Wir zeigen $L_{q_i} \notin \mathcal{L}_{\mathrm{R}}$, indem wir beweisen, dass $L_{\mathrm{U}} \leq_{\mathrm{EE}} L_{q_i}$. Wir konstruieren also eine TM A, die eine Eingabe für L_{U} in eine Eingabe für L_{q_i} transformiert.

Dabei arbeitet A wie folgt:

- A prüft, ob die Eingabe x die Form $\mathrm{Kod}(M)\#w$ hat.
- Wenn die Eingabe nicht diese Form hat, dann gibt A ein Wort aus, das nicht in L_{q_i} liegt, also zum Beispiel das leere Wort λ.
- Wenn die Eingabe die Form $\mathrm{Kod}(M)\#w$ hat, dann bestimmt A die Nummer i von q_{accept} in $\mathrm{Kod}(M)$ und gibt $\mathrm{Kod}(M)\#w\#0^i$ aus.

Wir zeigen nun, dass $x \in L_{\mathrm{U}} \iff A(x) \in L_{q_i}$.

Sei $x \in L_{\mathrm{U}}$. Dann ist die Eingabe $\mathrm{Kod}(M)\#w$ für ein $w \in L(M)$, d. h. M beendet ihre Berechnung auf w in q_{accept}, der in $\mathrm{Kod}(M)$ die Nummer i trägt. Da M mehr als i Zustände hat (q_{reject} ist Zustand $i+1$ gemäss der TM-Kodierung aus Abschnitt 4.6 im Buch) und die Berechnung in Zustand i endet, ist $A(x) = \mathrm{Kod}(M)\#w\#0^i \in L_{q_i}$.

Sei $x \notin L_{\mathrm{U}}$. Dann ist entweder x eine ungültige Eingabe, oder $w \notin L(M)$. Wenn x eine ungültige Eingabe ist, so ist $A(x) = \lambda \notin L_{q_i}$.

Andernfalls gilt $A(x) = \mathrm{Kod}(M)\#w\#0^i \notin L_{q_i}$, da die Berechnung von M auf w nie q_{accept} (also den i-ten Zustand) erreicht.

Da die EE-Reduktion ein Spezialfall der R-Reduktion ist, folgt aus $L_{\mathrm{U}} \notin \mathcal{L}_{\mathrm{R}}$ und $L_{\mathrm{U}} \leq_{\mathrm{EE}} L_{q_i}$ sofort $L_{q_i} \notin \mathcal{L}_{\mathrm{R}}$.

Lösung zu Aufgabe Z2

Wir zeigen $L'_{q_i} \notin \mathcal{L}_{\mathrm{R}}$, indem wir beweisen, dass $L_{\mathrm{U}} \leq_{\mathrm{EE}} L'_{q_i}$. Wir konstruieren also eine TM A, die eine Eingabe für L_{U} in eine Eingabe für L'_{q_i} transformiert.

Dabei arbeitet A wie folgt:

- A prüft, ob die Eingabe x die Form $\mathrm{Kod}(M)\#w$ hat.
- Wenn die Eingabe nicht diese Form hat, dann gibt A ein Wort aus, das nicht in L_{q_i} liegt, also zum Beispiel das leere Wort λ.

- Wenn die Eingabe die Form $\text{Kod}(M)\#w$ hat, dann konstruiert A eine TM $N_{M,w}$, die ihre eigene Eingabe ignoriert und für jede Eingabe die TM M auf w simuliert.
- A bestimmt die Nummer i von q_{accept} in $\text{Kod}(N_{M,w})$ und gibt $\text{Kod}(N_{M,w})\#0^i$ aus.

Wir zeigen nun, dass $x \in L_{\text{U}} \iff A(x) \in L'_{q_i}$.
Sei $x \in L_{\text{U}}$. Dann ist die Eingabe $\text{Kod}(M)\#w$ für ein $w \in L(M)$, d. h. M beendet ihre Berechnung auf w in q_{accept}. Weil $N_{M,w}$ für jede eigene Eingabe M auf w simuliert, akzeptiert $N_{M,w}$ jede Eingabe, endet also immer in $q_{\text{accept}} = q_i$, und $N_{M,w}$ hat nach Definition der Turingmaschinen-Kodierung aus Abschnitt 4.6 im Buch mindestens $i+1$ Zustände. Damit ist $\text{Kod}(N_{M,w})\#0^i \in L'_{q_i}$.
Sei $x \notin L_{\text{U}}$. Dann ist entweder x eine ungültige Eingabe, oder $w \notin L(M)$. Wenn x eine ungültige Eingabe ist, so ist $A(x) = \lambda \notin L'_{q_i}$.
Andernfalls gilt $A(x) = \text{Kod}(N_{M,w})\#0^i \notin L'_{q_i}$, da die Berechnung von $N_{M,w}$ auf keiner Eingabe q_{accept} (also den i-ten Zustand) erreicht.
Da die EE-Reduktion ein Spezialfall der R-Reduktion ist, folgt aus $L_{\text{U}} \notin \mathcal{L}_{\text{R}}$ und $L_{\text{U}} \leq_{\text{EE}} L'_{q_i}$ sofort $L'_{q_i} \notin \mathcal{L}_{\text{R}}$.

Lösung zu Aufgabe Z3

Wir zeigen $L_{\text{EQ},\lambda} \notin \mathcal{L}_{\text{R}}$, indem wir beweisen, dass $L_{\text{H},\lambda} \leq_{\text{EE}} L_{\text{EQ},\lambda}$. Wir konstruieren also eine TM A, die eine Eingabe für $L_{\text{H},\lambda}$ in eine Eingabe für $L_{\text{EQ},\lambda}$ transformiert.
Dabei arbeitet A wie folgt:

- A prüft, ob die Eingabe x die Form $\text{Kod}(M)$ für eine TM M hat.
- Wenn die Eingabe nicht diese Form hat, dann gibt A ein Wort aus, das nicht in $L_{\text{EQ},\lambda}$ liegt, also zum Beispiel das leere Wort λ.
- Wenn die Eingabe die Form $\text{Kod}(M)$ hat, dann konstruiert A eine TM M', in der alle Transitionen zu q_{reject} umgeleitet werden zu q_{accept}. Die TM M' akzeptiert also genau die Wörter, auf denen M hält.
- Ausserdem konstruiert A eine TM $\overline{M}$, die jede Eingabe akzeptiert.
- A produziert die Ausgabe $\text{Kod}(M')\#\text{Kod}(\overline{M})$.

Wir zeigen nun, dass $x \in L_{\text{H},\lambda} \iff A(x) \in L_{\text{EQ},\lambda}$.
Sei $x \in L_{\text{H},\lambda}$. Dann ist die Eingabe $\text{Kod}(M)$ für eine TM M, so dass M auf λ hält, also ihre Berechnung auf λ in q_{accept} oder q_{reject} beendet. Damit akzeptiert M' das Wort λ. Weil $\overline{M}$ jede Eingabe akzeptiert, also auch λ, ist also $\text{Kod}(M')\#\text{Kod}(\overline{M}) \in L_{\text{EQ},\lambda}$.
Sei $x \notin L_{\text{H},\lambda}$. Dann ist entweder x eine ungültige Eingabe, oder $x = \text{Kod}(M)$ und M hält nicht auf λ. Wenn x eine ungültige Eingabe ist, so ist $A(x) = \lambda \notin L_{\text{EQ},\lambda}$.
Andernfalls gilt $A(x) = \text{Kod}(M')\#\text{Kod}(\overline{M})$. Weil M nicht auf λ hält, gilt $\lambda \notin L(M')$. Da aber $\lambda \in L(\overline{M})$ gilt, ist $\text{Kod}(M')\#\text{Kod}(\overline{M}) \notin L_{\text{EQ},\lambda}$.
Da die EE-Reduktion ein Spezialfall der R-Reduktion ist, folgt aus $L_{\text{H},\lambda} \notin \mathcal{L}_{\text{R}}$ und $L_{\text{H},\lambda} \leq_{\text{EE}} L_{\text{EQ},\lambda}$ sofort $L_{\text{EQ},\lambda} \notin \mathcal{L}_{\text{R}}$.

ETH zürich
Departement Informatik

Theoretische Informatik

Prof. Dr. Juraj Hromkovič
http://www.ita.inf.ethz.ch/theoInf16

Lösungsvorschläge – Blatt Z2

Zürich, 2. Dezember 2016

Lösung zu Aufgabe Z4

Sei Σ ein Alphabet mit $|\Sigma| = k \geq 2$ und sei $w = a_1 a_2 \dots a_m$ ein Wort der Länge m über Σ. Um eine KNF-Formel anzugeben, die genau dann erfüllbar ist, wenn w eine Davenport-Schinzel-Sequenz der Ordnung 2 über Σ ist, verwenden wir Variablen $X_{a,j}$ für alle $a \in \Sigma$ und alle $1 \leq j \leq m$. Die Belegung der Variablen $X_{a,j}$ mit 1 soll bedeuten, dass an der Position j in w das Zeichen a steht, dass also $a_j = a$ gilt. Analog soll die Belegung mit 0 bedeuten, dass $a_j \neq a$.

Dann lässt sich die Bedingung (i) beschreiben durch die Teilformel

$$F_{\text{(i)}} = \bigwedge_{a \in \Sigma} \bigwedge_{1 \leq j \leq m-1} (\overline{X_{a,j}} \vee \overline{X_{a,j+1}}) \,.$$

Für alle möglichen Buchstaben und alle zwei aufeinanderfolgenden Positionen in w verbietet $F_{\text{(i)}}$, dass dort zweimal das gleiche Zeichen steht.

Um die Bedingung (ii) mit einer Teilformel zu beschreiben, müssen wir jeweils vier Positionen $1 \leq j_1 < j_2 < j_3 < j_4 \leq m$ in w und zwei beliebige verschiedene Zeichen a und b aus Σ betrachten und auf diesen Positionen das Muster *abab* ausschliessen. Für konkrete Positionen muss also gelten

$$\neg(X_{a,j_1} \wedge X_{b,j_2} \wedge X_{a,j_3} \wedge X_{b,j_4}) \,.$$

Dies ist äquivalent zu der Klausel

$$Z_{a,b,j_1,j_2,j_3,j_4} = (\overline{X_{a,j_1}} \vee \overline{X_{b,j_2}} \vee \overline{X_{a,j_3}} \vee \overline{X_{b,j_4}}) \,.$$

Damit ergibt sich für die Bedingung (ii) die Teilformel

$$F_{\text{(ii)}} = \bigwedge_{a,b \in \Sigma, a \neq b} \bigwedge_{1 \leq j_1 < j_2 < j_3 < j_4 \leq m} Z_{a,b,j_1,j_2,j_3,j_4}$$

und insgesamt zur Beschreibung der Davenport-Schinzel-Bedingung der Ordnung 2 für das Wort w die KNF-Formel

$$F_{\text{(i)}} \wedge F_{\text{(ii)}} \,.$$

Weil unsere Formel aber nicht nur für ein konkretes Wort w, sondern für eine gegebene Wortlänge m beschreiben soll, ob es eine Davenport-Schinzel-Sequenz dieser Länge über Σ gibt, müssen wir in der Formel noch die Eindeutigkeit des Worts beschreiben.

Die Teilformel

$$F_{\text{Wort},1} = \bigwedge_{1 \le i \le m} \bigwedge_{a,b \in \Sigma, a \neq b} (\overline{X_{a,i}} \vee \overline{X_{b,i}})$$

beschreibt, dass an einer Position des Wortes nicht zwei verschiedene Buchstaben stehen können. Weiterhin beschreibt die Teilformel

$$F_{\text{Wort},2} = \bigwedge_{1 \le i \le m} \bigvee_{a \in \Sigma} X_{a,i}\,,$$

dass an jeder Position des Wortes mindestens ein Buchstabe steht.
Damit ergibt sich insgesamt die Formel

$$F_{\text{(i)}} \wedge F_{\text{(ii)}} \wedge F_{\text{Wort},2} \wedge F_{\text{Wort},1}\,.$$

Lösung zu Aufgabe Z5

Um VC $\leq_p$ DS zu zeigen, geben wir eine Polynomialzeit-Reduktion an, die eine beliebige VC-Instanz in eine DS-Instanz umwandelt. Wir gehen zunächst davon aus, dass die VC-Instanz keine isolierten Knoten hat.

Sei (G, k) eine Instanz von VC, wobei $G = (V, E)$ ein ungerichteter Graph ohne isolierte Knoten ist und $k \in \mathbb{N}$. Wir konstruieren daraus die DS-Instanz (G', k), wobei der ungerichtete Graph $G' = (V', E')$ wie folgt definiert ist: Wir fügen für jede Kante $e = \{x, y\} \in E$ von G einen neuen Knoten z_e und die beiden Kanten $\{x, z_e\}$ und $\{z_e, y\}$ hinzu. Formal gilt also

$$\begin{aligned} V' &= V \cup \{z_e \mid e \in E\} \quad \text{und} \\ E' &= E \cup \{\{x, z_e\} \mid e \in E \text{ und } x \in e\}. \end{aligned}$$

Nun wollen wir zeigen, dass G ein Vertex-Cover der Grösse k hat genau dann, wenn G' eine dominierende Menge der Grösse k hat.

Sei C ein Vertex-Cover von G mit $|C| = k$. Da jedes Vertex-Cover eine dominierende Menge ist, werden auch in G' alle Knoten aus V von C dominiert. Weil C ein Vertex-Cover für G ist, ist für jede Kante $e = \{x, y\}$ aus E mindestens ein Endpunkt x in C enthalten. Dieser ist dann in G' zu dem Knoten z_e benachbart, also werden in G' auch alle Knoten z_e von C dominiert. Damit ist C eine dominierende Menge von G'.

Sei D' eine dominierende Menge von G' mit $|D'| = k$. Dann können wir D' in eine höchstens gleich grosse dominierende Menge D umwandeln, die keinen der Knoten z_e enthält: Für $e = \{x, y\}$ dominiert z_e nur die Knoten x, y und sich selbst. Diese drei Knoten werden aber auch von x (oder auch von y) dominiert, also können wir z_e durch x ersetzen.

Wir gehen also im Folgenden davon aus, dass D eine dominierende Menge von G' mit $|D| \leq k$ ist, für die $D \subseteq V$ gilt. Wir wollen zeigen, dass D dann ein Vertex-Cover für G ist. Weil D eine dominierende Menge von G' ist, die keinen der Knoten z_e enthält, ist jeder Knoten z_e zu einem Knoten aus D benachbart. Damit ist aber für jede Kante $e \in E$ mindestens einer der Endpunkte von e in D enthalten. Also ist D ein Vertex-Cover für G.

Falls die VC-Instanz m isolierte Knoten enthält, können wir auf analoge Weise zeigen, dass G genau dann ein Vertex-Cover der Grösse k enthält, wenn G' eine dominierende Menge der Grösse $k + m$ hat. Um eine dominierende Menge zu erhalten, fügen wir alle isolierten Knoten zum Vertex-Cover hinzu.

Lösung zu Aufgabe Z6

Wir betrachten die in der Aufgabenstellung beschriebene Reduktion und zeigen, dass die Formel Φ genau dann erfüllbar ist, wenn es eine Teilmenge $U \subseteq S$ gibt mit $\sum_{x \in U} x = t$.

Sei α eine erfüllende Belegung von Φ. Wir konstruieren hieraus eine Teilmenge U_α wie folgt: Für jede Variable x_i mit $\alpha(x_i) = 1$ wählen wir $r_i \in U_\alpha$, für jede Variable x_i mit $\alpha(x_i) = 0$ wählen wir $r'_i \in U_\alpha$. Für jede Klausel C_j, in der α alle drei Literale erfüllt, nehmen wir zusätzlich noch s_j in U_α auf. Analog nehmen wir s'_j in U_α auf, falls α in C_j genau zwei Literale erfüllt und wir fügen s_j und s'_j zu U_α hinzu, falls α in C_j genau ein Literal erfüllt. Es ist zu zeigen, dass für die so definierte Menge U_α gilt, dass $\sum_{u \in U_\alpha} u = t$.

Wir bemerken zunächst, dass an keiner Stelle in der Addition ein Übertrag auftreten kann. Die Stelle i ist für alle $1 \leq i \leq n$ nur in den beiden Zahlen r_i und r'_i gleich 1 und in allen anderen Zahlen gleich 0. Für alle $1 \leq j \leq k$ ist die Stelle $n + j$ nur in fünf Zahlen ungleich Null, nämlich 1 in den drei Zahlen r_i bzw. r'_i, so dass x_i in C_j positiv bzw. negativ vorkommt, und 1 bzw. 2 in den beiden Zahlen s_j und s'_j. Die Summe an dieser Stelle kann also den Wert 6 nicht übersteigen. Damit können wir für den Beweis die einzelnen Stellen der Zahlen unabhängig voneinander betrachten.

Wir analysieren zunächst die Stelle i für $1 \leq i \leq n$. Da α die Variable x_i nicht gleichzeitig positiv und negativ belegen kann, wurde genau eine der beiden Zahlen r_i und r'_i in U_α aufgenommen. Da die Stelle i in allen anderen Zahlen eine Null enthält, ist die Summe an dieser Stelle 1, wie von t gefordert. Wenn α in der Klausel C_j genau l Literale erfüllt, für ein $l \in \{1, 2, 3\}$, dann tragen die entsprechend ausgewählten Zeilen für diese Literale einen Betrag von l zu der Summe an der Stelle $n + j$ bei. Zusammen mit den nach der oben beschriebenen Vorschrift gegebenenfalls hinzugefügten Zahlen s_j und s'_j ergibt sich hier immer eine Summe von $t[n + j] = 4$. Damit erfüllt U_α die Bedingung, dass $\sum_{u \in U_\alpha} u = t$.

Nehmen wir nun umgekehrt an, dass U eine Teilmenge von S ist, die $\sum_{u \in U_\alpha} u = t$ erfüllt. Wir konstruieren hieraus eine Belegung α_U für X wie folgt: Falls $r_i \in U$, dann setzen wir $\alpha_U(x_i) = 1$, falls $r'_i \in U$, dann setzen wir $\alpha_U(x_i) = 0$. Für jedes $i \in \{1, \ldots, n\}$ muss wegen $t[i] = 1$ einer dieser beiden Fälle auftreten, damit ist α_U wohldefiniert. Wir zeigen nun, dass α_U die Formel Φ erfüllt. Da $t[n + j] = 4$, sich die Zahlen s_j und s'_j an der Stelle $n + j$ aber nur zu höchstens 3 aufsummieren, muss für jede Stelle $n + j$ mit $1 \leq j \leq k$ eine der Zahlen r_i oder r'_i mit $r_i[n + j] = 1$ bzw. $r'_i[n + j] = 1$ in U enthalten sein. Falls dies eine Zahl r_i ist, dann wurde $\alpha_U(x_i) = 1$ gesetzt. Nach der Konstruktion tritt die Variable x_i in C_j positiv auf (weil $r_i[n + j] = 1$) und somit wird die Klausel C_j von α_U erfüllt. Falls die Zahl r'_i mit $r'_i[n + j] = 1$ in U enthalten ist, dann wurde $\alpha_U(x_i) = 0$ gesetzt. Nach der Konstruktion tritt die Variable x_i in C_j negativ auf (weil $r'_i[n + j] = 1$) und somit wird die Klausel C_j ebenfalls von α_U erfüllt. Damit ist α_U eine erfüllende Belegung für Φ.

http://www.springer.com/978-3-658-06432-7

Theoretische Informatik
Formale Sprachen, Berechenbarkeit, Komplexitätstheorie, Algorithmik, Kommunikation und Kryptographie
Hromkovič, J.
2014, XVIII, 349 S. 87 Abb., Softcover
ISBN: 978-3-658-06432-7