ETH zürich
Departement Informatik

Theoretische Informatik

Prof. Dr. Juraj Hromkovič
Dr. Hans-Joachim Böckenhauer
`https://courses.ite.inf.ethz.ch/theo_inf_21`

Lösungsvorschläge – Blatt 11

Zürich, 17. Dezember 2021

Lösung zu Aufgabe 30

(a) Angenommen, L_1 sei kontextfrei. Wir betrachten dann die Zahl $n_1 = n_{L_1}$ gemäss dem Pumping-Lemma für kontextfreie Sprachen und das Wort $z = 10^{n_1}\#1^{n_1}\#1^{n_1+1} \in L_1$. Um Zweideutigkeiten zu vermeiden, sprechen wir jeweils vom ersten bzw. zweiten Summanden des Wortes, wenn wir 10^{n_1} bzw. 1^{n_1} meinen, und vom Ergebnis, wenn wir uns auf 1^{n_1+1} beziehen. Nach dem Pumping-Lemma existiert nun eine Zerlegung $z = uvwxy$, die die drei Bedingungen des Pumping-Lemmas erfüllt. Insbesondere gilt $|vx| \geq 1$ und $|vwx| \leq n_1$.

Wir unterscheiden nun drei Fälle bezüglich der Lage von v und x in z.

Falls vx ein #-Symbol enthält, so hat uwy die falsche Form. Wir nehmen also im Folgenden an, dass vx nur aus Nullen und Einsen besteht.

Falls vx keinen Teil des Ergebnisses enthält, so wird gegenüber z in uwy einer der beiden Summanden (oder beide) verkürzt, das Ergebnis jedoch nicht. Da beide Summanden mit einer 1 beginnen, stimmt nun also die Bedingung für Wörter in der Sprache nicht mehr, dies ist ein Widerspruch zu Bedingung (iii).

Falls vx hingegen mindestens einen Buchstaben des Ergebnisses enthält, ist wegen Bedingung (ii) ausgeschlossen, dass vx zusätzlich einen Buchstaben des ersten Summanden enthält. In uwy ist die binäre Länge des ersten Summanden also unverändert und die des Ergebnisses mindestens um 1 kürzer gegenüber z. Da alle drei Zahlen mit einer führenden 1 beginnen, kann die Bedingung auch hier nicht mehr erfüllt sein, unabhängig davon, ob vx einen Teil des zweiten Summanden enthält oder nicht.

Weitere Fälle können nicht auftreten und somit erhalten wir einen Widerspruch zum Pumping-Lemma, also war unsere Annahme falsch und L_1 ist nicht kontextfrei.

(b) Angenommen, L_2 sei kontextfrei. Wir betrachten dann die Zahl $n_2 = n_{L_2}$ gemäss dem Pumping-Lemma für kontextfreie Sprachen und das Wort $z = a^{n_2}b^{n_2}ca^{n_2}b^{n_2} \in L_2$. Nach dem Pumping-Lemma existiert dann eine Zerlegung $z = uvwxy$, die die drei Bedingungen des Pumping-Lemmas erfüllt. Insbesondere gilt $|vx| \geq 1$ und $|vwx| \leq n_2$.

Wir unterscheiden nun mehrere Fälle bezüglich der Lage von v und x in z.

Falls vx den Buchstaben c enthält, so enthält uwy kein c mehr, also ist $uwy \notin L_2$ im Widerspruch zur Bedingung (iii) des Pumping-Lemmas.

Falls vx nur Buchstaben des Präfixes $a^{n_2}b^{n_2}$ von z enthält, dann ist in uv^2wx^2y das Teilwort vor dem c länger als das Teilwort dahinter, also ist $uv^2wx^2y \notin L_2$ im Widerspruch zur Bedingung (iii) des Pumping-Lemmas.

Falls vx nur Buchstaben des Suffixes $a^{n_2}b^{n_2}$ enthält, dann ist in uwy das Teilwort vor dem c länger als das Teilwort dahinter, also ist $uwy \notin L_2$ im Widerspruch zur Bedingung (iii) des Pumping-Lemmas.

Es bleibt der Fall zu betrachten, dass vx sowohl Buchstaben vor dem c als auch nach dem c enthält. Wegen $|vwx| \leq n_2$ kann vx dann keine Symbole a aus dem Präfix a^{n_2} und keine Symbole b aus dem Suffix b^{n_2} enthalten. Damit enthält uv^2wx^2y vor dem c mehr bs als dahinter, also ist $uv^2wx^2y \notin L_2$ im Widerspruch zur Bedingung (iii) des Pumping-Lemmas.

In jedem der möglichen Fälle erhalten wir also einen Widerspruch zum Pumping-Lemma, also war unsere Annahme falsch und L_2 ist nicht kontextfrei.

Lösung zu Aufgabe 31

Weil L_1 kontextfrei ist und L_2 regulär, wird L_1 durch einen Kellerautomaten $M_1 = (Q_1, \{a,b\}, \Gamma, \delta_1, q_{0,1}, Z_0)$ akzeptiert und L_2 durch einen (nichtdeterministischen) endlichen Automaten $M_2 = (Q_2, \{a,b\}, \delta_2, q_{0,2}, F)$.

Wir können nun im Prinzip die beiden Automaten parallel laufen lassen, indem wir dieselbe Kreuzprodukt-Konstruktion verwenden wie bei der Kombination zweier endlicher Automaten. Das einzige zusätzliche Problem liegt darin, dass M_1 akzeptiert, sobald der Keller leer ist, und M_2 beim Erreichen eines akzeptierenden Zustands. Der neue Automat muss als Kellerautomat wieder durch das Leeren des Kellers akzeptieren.

Die Idee hierzu ist, das Leeren des Kellers nur zu gestatten, wenn M_2 gleichzeitig in einen akzeptierenden Zustand geht. Um aber erkennen zu können, wann der Keller geleert wird, müssen wir sicherstellen, dass Z_0 nur einmal benutzt wird, nämlich ganz unten im Keller. Wir verwandeln dazu zunächst M_1 in den äquivalenten Kellerautomaten $M_1' = (Q_1', \{a,b\}, \Gamma', \delta_1', q_{0,1}', Z_0')$ mit

$$\begin{aligned}
Q_1' &= Q_1 \cup \{q_{0,1}'\}, q_{0,1}' \notin Q_1,\\
\Gamma' &= \Gamma \cup \{Z_0'\}, Z_0' \notin \Gamma,\\
\delta_1'(q_{0,1}', \lambda, Z_0') &= \{(q_{0,1}, Z_0'Z_0)\},\\
\delta_1'(q, \lambda, Z_0') &= \{(q, \lambda)\} \text{ für alle } q \in Q_1,\\
\delta_1'(q, x, Z) &= \delta_1(q, x, Z) \text{ für alle } q \in Q_1, x \in \{a,b\} \cup \{\lambda\} \text{ und } Z \in \Gamma.
\end{aligned}$$

Der Kellerautomat M_1' hat also als unterstes Symbol Z_0'. Als einzig möglichen Anfangsschritt legt er darauf Z_0 und geht in $q_{0,1}$, stellt also die Anfangskonfiguration von M_1 her (mit zusätzlich Z_0' zuunterst im Keller). Dann verhält er sich genau wie M_1, solange Z_0' nicht wieder zum Vorschein kommt. In dem Fall, dass Z_0' zum Vorschein kommt (als oberstes auf dem Keller liegt), wird dieses vom Keller genommen, womit jener leer und die Berechnung beendet ist. Dies passiert also genau dann, wenn darüber alles aus dem Keller genommen wurde, also der Keller von M_1 geleert wurde. Die beiden Automaten sind folglich äquivalent.

Nun konstruieren wir als Kreuzprodukt aus M_1' und M_2 den Kellerautomaten $M = (Q, \{a,b\}, \Gamma', \delta, q_0, Z_0')$ mit

$$Q = Q_1' \times Q_2,$$

$$
\begin{aligned}
q_0 &= (q'_{0,1}, q_{0,2}),\\
\delta(q_0, \lambda, Z'_0) &= \{((q'_{0,1}, q_{0,2}), Z'_0 Z_0)\},\\
\delta((p,q), \lambda, Z'_0) &= \{((p,q), \lambda, \lambda)\} \text{ für alle } p \in Q_1, q \in F,
\end{aligned}
$$

und für alle $(p,q) \in Q$, $x \in \{a,b\}$ und $Z \in \Gamma$ setzen wir

$$
\begin{aligned}
\delta((p,q), x, Z) &= \{((p',q'), \beta) \mid (p',\beta) \in \delta'_1(p,x,Z), q' \in \delta_2(q,x)\} \text{ sowie}\\
\delta((p,q), \lambda, Z) &= \{((p',q), \beta) \mid (p',\beta) \in \delta'_1(p,\lambda,Z)\}.
\end{aligned}
$$

Aus dieser Konstruktion ergibt sich unmittelbar für jedes Wort $w \in \{a,b\}^*$, dass eine akzeptierende Berechnung

$$((q'_{0,1}, q_{0,2}), w, Z'_0) \vdash_M \ldots \vdash_M ((p,q), u, \alpha) \vdash_M \ldots \vdash_M ((p',q'), \lambda, \lambda)$$

von M auf w genau den akzeptierenden Berechnungen

$$(q'_{0,1}, w, Z'_0) \vdash_{M'_1} \ldots \vdash_{M'_1} (p, u, \alpha) \vdash_{M'_1} \ldots \vdash_{M'_1} (p', \lambda, \lambda)$$

und

$$(q_{0,2}, w) \vdash_{M_2} \ldots \vdash_{M_2} (q, u) \vdash_{M_2} \ldots \vdash_{M_2} (q', \lambda)$$

von M'_1 bzw. M_2 auf w entspricht und umgekehrt. Man muss dabei nur beachten, dass, wenn M'_1 und M Schritte machen, bei denen das leere Wort gelesen wird, im Fall von M_2 nichts geschieht, d. h. die Konfiguration von M_2 unverändert bleibt und kein Berechnungsschritt notwendig ist.

Insgesamt erhalten wir

$$L(M) = L(M'_1) \cap L(M_2) = L_1 \cap L_2,$$

also ist $L_1 \cap L_2$ kontextfrei.

Lösung zu Aufgabe 32

Wir zeigen im Folgenden, wie man mit einem 2-Keller-Automaten eine Turingmaschine simulieren kann. Die Idee dabei ist, dass der Schreibkopf der Turingmaschine ihr Arbeitsband in zwei Teile aufteilt, die mit den beiden Kellern simuliert werden, sodass eine der Kellerspitzen der Position des Schreibkopfs der Turingmaschine entspricht und ein Schritt nach links oder rechts durch Umkopieren von einem Keller auf den anderen simuliert wird.

Sei also M eine Turingmaschine. Dann simulieren wir die Arbeit von M auf einem 2-Keller-Automaten A wie folgt: Da die TM kein separates Eingabeband hat, müssen wir zunächst die gesamte Eingabe von A auf die Keller kopieren. Dieses Kopieren führen wir so durch, dass wir zunächst jedes gelesene Eingabesymbol auf den ersten Keller schieben. Weiterhin legen wir ein Leerzeichen ␣ auf den zweiten Keller. Damit zum Start der eigentlichen Simulation von M das erste Eingabesymbol lesbar ist, verschieben wir nun den Inhalt des ersten Kellers (bis auf das Kellerbodensymbol Z_0) Zeichen für Zeichen auf den zweiten Keller. Die hierfür nötigen Transitionen (und alle weiteren) sind natürlich λ-Transitionen, d. h. A beachtet von nun an seine eigene Eingabe nicht mehr. Dann schreiben wir die Endmarkierung ¢ des Arbeitsbandes auf den ersten Keller.

Wir können nun mit der eigentlichen Simulation beginnen. Hierfür interpretieren wir das oberste Symbol auf dem ersten Keller als das Feld das Arbeitsbandes, auf dem der

Schreibkopf gerade steht. Der restliche Inhalt des ersten Kellers beschreibt dann den Teil des Arbeitsbandes links vom Schreibkopf, der zweite Keller hingegen den Teil des Arbeitsbandes rechts vom Schreibkopf. Der Austausch des Symbols an der aktuellen Position des Arbeitsbandes kann nun einfach simuliert werden. Ein Schritt auf dem Band nach links entspricht dem Verschieben der Spitze des ersten Kellers auf den zweiten Keller, ein Schritt nach rechts wird simuliert durch Verschieben der zweiten Kellerspitze auf den ersten Keller. Falls dabei das ␣ auf dem zweiten Keller erreicht wird, wird dieses nicht verschoben, sondern auf den ersten Keller kopiert.

Da beide Keller noch ein Kellerbodensymbol besitzen, das während der Simulation nicht angerührt wird, kann diese nicht zu früh durch einen leeren Keller abgebrochen werden. Wenn die simulierte TM M in den Endzustand q_{accept} kommt, soll die Eingabe akzeptiert werden, A löscht dann also seine beiden Keller und hält. Endet hingegen die Berechnung von M in q_{reject}, dann beendet A die Simulation ohne die Keller zu leeren und akzeptiert dadurch nicht. Falls die Berechnung von M auf w nicht endet, dann endet auch die Simulation nicht und auch A arbeitet unendlich lange und akzeptiert die Eingabe w nicht.