Hochschule Karlsruhe
University of Applied Sciences

Fakultät für Informatik
und Wirtschaftsinformatik

Lehrgebiet für Grundlagen der Informatik
Prof. Dr. Heiko Körner

11. Übung zur Vorlesung Theoretische Informatik I

Musterlösungen

Aufgabe 1: L ist regulär, also existiert ein DEA, der genau die Wörter aus L erkennt. Wir können annehmen, dass keine „nutzlosen“ Zustände vorhanden sind, also irgendwelche Zustände, von denen man aus gar keinen Endzustand mehr erreichen kann.

a) Wenn man nun alle verbliebenen Zustände zu Endzuständen ernennt, so erhält man einen DEA, der nicht nur jedes Wort, sondern auch auf dem Weg dahin jedes bislang verarbeitete Präfix akzeptiert. Also erkennt der DEA dann genau die Sprache L_P.

b) Wenn man analog jeden Zustand zu einem Startzustand erklärt, so erhält man entsprechend einen NEA, der alle Suffixe von allen Wörtern aus L akzeptiert. Ein solcher NEA erkennt also die Sprache L_S.

Egal ob DEA oder NEA — die akzeptierte Sprache ist in jedem Fall regulär. □

Aufgabe 2: Da reguläre Sprachen unter Schnittbildung abgeschlossen sind, lässt sich leicht zeigen, dass L nicht regulär ist:

a) Natürlich ist L' regulär. Z.B. wird L' durch den regulären Ausdruck 0^+1^+ beschrieben.

b) L enthält nur Wörter, die eine gleiche Anzahl von Nullen und Einsen besitzen. L' enthält nur Wörter, die zuerst mit einer Sequenz von Nullen beginnen und dann mit einer Sequenz von Einsen enden. Der Schnitt von L und L' enthält demnach genau die Wörter, die mit einer bestimmten Anzahl von Nullen beginnen und mit der gleichen Anzahl von Einsen enden. Es gilt also $L \cap L' = \{0^n 1^n \mid n \in \mathbb{N}\}$. Von dieser Sprache ist uns bereits bekannt, dass sie nicht regulär ist (siehe erstes Beispiel zur Anwendung des Pumping–Lemmas in der Vorlesung).

c) Wäre L regulär, so wären beide Sprachen L und L' regulär. Also müsste auch $L \cap L'$ regulär sein, was nicht der Fall ist. Also kann L nicht regulär sein.

Aufgabe 3: Mögliche Lösungen lauten wie folgt:

a) Ein NKA M kann die Sprache L wie folgt erkennen. Es bezeichne dazu $|w|_0$ die Anzahl der Nullen und $|w|_1$ die Anzahl der Einsen in einem Wort $w \in \{0,1\}^*$. Sei jetzt x das aktuell zu überprüfende Wort. Wenn bislang ein Präfix w von x verarbeitet wurde und in w die Differenz der Anzahl der Nullen und der Anzahl der Einsen positiv ist (also $|w|_0 > |w|_1$ gilt), so soll der Keller aus einer Folge von $|w|_0 - |w|_1$ Nullen bestehen. Ist dagegen die Differenz negativ, so soll der Keller eine Folge von $|w|_1 - |w|_0$ Einsen enthalten. Dies lässt sich wie folgt erreichen. Immer wenn M eine Null liest und auch eine Null oben auf dem Keller steht, wird eine weitere Null auf den Keller gelegt. Falls sich dagegen eine Eins oben auf dem Keller befindet, wird diese gelöscht. Beim Lesen einer Eins ist es umgekehrt: eine Null oben auf dem Keller wird entfernt, bei einer Eins wird noch eine weitere Eins hinzugefügt. M kann nichtdeterministisch jederzeit annehmen, dass das Ende der Eingabe erreicht wurde. M räumt dann den Keller leer und verifiziert, dass sich nur noch eine (nichtleere) Nullfolge darin befindet. Zum Schluss wird noch das Startsymbol # aus dem Keller entfernt.

b) Sei $M = (\{z_0, z_1\}, \{0,1\}, \{\#, 0, 1\}, \delta, z_0, \#)$. Die folgende Übergangsfunktion implementiert die obige Idee:

$$\begin{array}{lll} z_0 0\# \to z_0 0\#, & z_0 00 \to z_0 00, & z_0 01 \to z_0 \varepsilon, \\ z_0 1\# \to z_0 1\#, & z_0 11 \to z_0 11, & z_0 10 \to z_0 \varepsilon, \\ z_0 \varepsilon 0 \to z_1 \varepsilon, & z_1 \varepsilon 0 \to z_1 \varepsilon, & z_1 \varepsilon \# \to z_1 \varepsilon \ . \end{array}$$

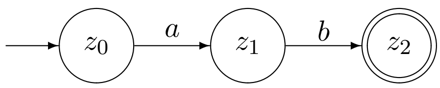
c) M akzeptiert wie geplant $01001 \in L$ wegen

$$\begin{aligned} &(z_0, 01001, \#) \vdash (z_0, 1001, 0\#) \vdash (z_0, 001, \#) \vdash (z_0, 01, 0\#) \\ &\vdash (z_0, 1, 00\#) \vdash (z_0, \varepsilon, 0\#) \vdash (z_1, \varepsilon, \#) \vdash (z_1, \varepsilon, \varepsilon) \ . \end{aligned}$$

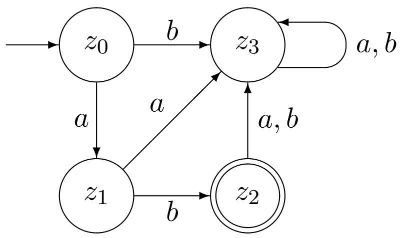
d) M ist nicht deterministisch, denn bei noch nicht vollständigem Verbrauch der Eingabe und gleichzeitigem Vorliegen einer Null auf dem Keller kann M im Zustand z_0 entweder das nächste Zeichen verarbeiten ($z_0 00 \to z_0 00$, $z_0 10 \to z_0 \varepsilon$) oder aber in den Zustand z_1 wechseln ($z_0 \varepsilon 0 \to z_1 \varepsilon$).

Aufgabe 4: Ein DEA für L ergibt sich aus den folgenden drei Schritten:

a) Ein DEA, der nur das Wort ab akzeptiert, ist denkbar einfach:



b) Mit totaler Übergangsfunktion δ sieht der DEA wie folgt aus:



c) Gemäß der zu verwendenden Konstruktion werden nun noch lediglich die Endzustände ausgetauscht, was zu diesem DEA führt: