Prof. Dr. Jasmin Blanchette
Jannis Limperg
Lydia Kondylidou

Ludwig-Maximilians-Universität München
Institut für Informatik
Abgabe bis 14. Juni 2023, 10:00 Uhr

Lösungsvorschlag zur Übung 8 zur Vorlesung

Formale Sprachen und Komplexität

FSK8-1 ***Sprachen einordnen*** (2 Punkte)

Für die i-fache Wiederholung des Worts w schreiben wir $(w)^i$, um Anfang und Ende von w zu markieren. Die Klammern sind daher *nicht* Teil des Alphabets der jeweiligen Sprachen. Die formalen Sprachen L_i, $i = 0, \ldots, 3$, seien definiert als

$$\begin{array}{lcll} L_0 & := & \{(a|a(a|b)^i a) \mid i \in \mathbb{N}\} & \subseteq \{a,b\}^* \\ L_1 & := & \{(a(b)^k a(b)^l)^i \mid i,k,l \in \mathbb{N}\} & \subseteq \{a,b\}^* \\ L_2 & := & \{a^i b^{i+1} \mid i \in \mathbb{N}\} & \subseteq \{a,b\}^* \\ L_3 & := & \{(ab)^i \$ (ab)^i \$ (ab)^i \mid i \in \mathbb{N}\} & \subseteq \{a,b,\$\}^* \end{array}$$

Bearbeiten Sie die folgenden Arbeitsaufträge für jede der Sprachen L_i.

a) Beweisen oder widerlegen Sie, dass L_i regulär ist.

b) Beweisen oder widerlegen Sie, dass L_i deterministisch kontextfrei ist.

c) Beweisen oder widerlegen Sie, dass L_i kontextfrei ist.

Hinweis: Nutzen Sie, dass manche Aussagen direkt aus anderen Aussagen folgen. Um zu beweisen, dass L_i regulär/deterministisch kontextfrei/kontextfrei ist, genügt es, ein geeignetes Konstrukt K_i (Grammatik, Automat oder regulärer Ausdruck) anzugeben und kurz zu begründen, warum $L(K_i) = L_i$ gilt.

LÖSUNGSVORSCHLAG:

- $L_0 = \{(a|a(a|b)^i a) \mid i \in \mathbb{N}\} = L((a|a(a|b)^* a)$ ist regulär (und damit auch kontextfrei und deterministisch kontextfrei), da die Sprache von einem regulären Ausdruck erzeugt wird. Jedes Wort der Sprache ist entweder ein einzelnes a oder es fängt mit a an, enthält dann beliebig viele a's oder b's und hört mit einem weiteren a auf. Der gegebene reguläre Ausdruck erzeugt genau solche Wörter.
- L_1 ist nicht kontextfrei und damit auch nicht regulär oder deterministisch kontextfrei. Beweis mit dem Pumping-Lemma für kontextfreie Sprachen. Gegeben sei eine Zahl $n \in \mathbb{N}$ und wir wählen das Wort $z = (ab^n ab^n)^2 = ab^n ab^n ab^n ab^n \in L_1$ mit $|z| \geq n$.

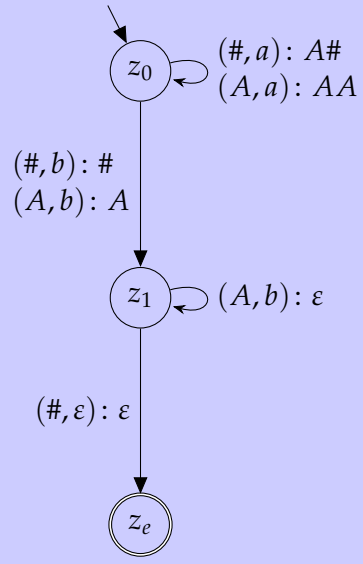
Betrachte nun Zerlegungen $z = uvwxy$ mit $|vwx| \leq n$ und $|vx| > 0$. Wir zeigen, dass es stets ein $i \in \mathbb{N}$ gibt sodass $uv^iwx^iy \notin L$ ist. Vollständige Fallunterscheidung:

- Das Teilwort vx enthält a. Dann enhält vx genau ein a, da $|vwx| \leq n$ und zwischen den a's jeweils b^n-Blöcke liegen. Das aufgepumpte Wort uv^2wx^2y enthält also 5 a's, aber Wörter in L_1 enthalten immer eine gerade Anzahl a's.
- Das Teilwort vx enthält kein a. Dann liegt vx in maximal zwei benachbarten b-Blöcken, da $|vwx| \leq n$. Liegt vx vollständig in einem b-Block, so ist uv^2wx^2y offensichtlich nicht in L_1. Liegt vx in zwei der b-Blöcke, so gibt es folgende Möglichkeiten:
  - vx liegt im ersten und zweiten Block.
  - vx liegt im zweiten und dritten Block.
  - vx liegt im dritten und vierten Block.

  In jedem dieser Fälle sind in uv^2wx^2y die zwei Hälften des Wortes nicht mehr gleich, sodass das aufgepumpte Wort nicht in L_1 liegen kann.

- L_2 ist nicht regulär. Beweis mit dem Pumping-Lemma für reguläre Sprachen. Sei $n \geq 1$ beliebig. Wir wählen $z = a^nb^{n+1}$ mit $z \in L_2$ und $|z| \geq n$. Sei $z = uvw$ eine beliebige Zerlegung von z mit $|uv| \leq n$ und $|v| \geq 1$. Aus diesen zwei Eigenschaften von uv folgt, dass uv nur aus a's besteht und dass v mindestens ein a enthält. Somit enthält uv^2w mehr als n a's, aber immer noch genau $n+1$ b's, also ist $uv^2w \notin L_2$. L_2 verletzt somit die Pumpingeigenschaft und ist nicht regulär.

  L_2 ist deterministisch kontextfrei (und damit auch kontextfrei), da der folgende DPDA M mit Startsymbol # die Sprache L_2 erkennt:



M schreibt zunächst für jedes gelesene a ein A in den Keller. Der Automat wechselt beim Lesen des ersten b in den Zustand z_1. Von dort aus liest er b-Eingaben, wobei für jede solche Eingabe genau ein A aus dem Keller entfernt wird. Wenn wir das Ende des Wortes erreichen und der Keller nur noch das Startsymbol # enthält, so haben wir n a's und $n+1$ b's gelesen und akzeptieren durch Übergang in z_e.

- L_3 ist nicht kontextfrei und damit auch nicht regulär oder deterministisch kontextfrei. Wir widerlegen die Kontextfreiheit mit dem Pumping-Lemma für kontextfreie Sprachen.

  Sei $n > 0$ beliebig. Wir wählen $z = (ab)^n\$(ab)^n\$(ab)^n$. Dann gilt $|z| \geq n$ und $z \in L_3$. Sei $uvwxy = z$ eine beliebige Zerlegung von z mit $|vwx| \leq n$ und $|vx| > 0$. Wenn vx das Symbol \$ enthält, dann gilt $uv^2wx^2y \notin L_3$, da $\#_\$(uv^2wx^2y) > 2$. Wenn vx das Symbol \$ nicht enthält, dann liegt vwx maximal in zwei der $(ab)^n$-Blöcke. Liegt vwx nicht im dritten Block, so ist $uv^0wx^0y \notin L_3$, da $\$(ab)^n$ ein Suffix ist, aber im vorderen Teil weniger als $2n$ $a's$ oder $2n$ $b's$ liegen. Analog: Liegt vwx nicht im ersten Block, so ist $uv^0wx^0y \notin L_3$, da $(ab)^n\$$ ein Präfix ist, aber im hinteren Teil weniger als $2n$ $a's$ oder $2n$ $b's$ liegen.

FSK8-2 ***Kontextsensitive Sprachen*** (0 Punkte)

Geben Sie eine Typ-1-Grammatik an, welche genau die Sprache

$$L = \{a^n \mid n \in \mathbb{N} \text{ und } n \text{ ist keine Primzahl}\}$$

erzeugt. Erläutern Sie die Funktionsweise Ihrer Grammatik: Wozu dienen die einzelnen Nichtterminale und Produktionen?

Demonstrieren Sie die Funktionsweise Ihrer Grammatik, indem Sie Ableitungen für die Wörter a^0, a^1, a^4 und a^6 angeben.

Hinweis: Wörter $a^n \in L$ mit Mindestlänge 4 können geschrieben werden als $a^{n \cdot m}$ mit $n, m \geq 2$. Eine Herangehensweise, diese Wörter zu erzeugen, ist zunächst a^n zu erzeugen und anschließend dieses Wort $m-1$-Mal zu kopieren. Diese Idee können Sie in eine Typ-1-Grammatik umsetzen.

LÖSUNGSVORSCHLAG:

Wir verwenden den Hinweis und erzeugen a^0 und a^1 direkt und ansonsten $a^{n \cdot m}$ mit $n, m \geq 2$, wobei die Idee ist, zunächst a^n zu erzeugen und anschließend $m-1$ Kopien davon zu machen.

Betrachte die Grammatik $G = (V, \Sigma, P, S)$ mit $V = \{S, N, A, \hat{A}, M, C, \hat{C}\}$, $\Sigma = \{a\}$ und den folgenden Produktionen P:

$$
\begin{aligned}
&S \to a \mid \varepsilon \mid NA\hat{A}\\
&N \to NA \mid M\\
&M \to \hat{C} \mid MC\\
&CA \to aAC\\
&C\hat{A} \to a\hat{A}\\
&Ca \to aC\\
&\hat{C}A \to aa\hat{C}\\
&\hat{C}a \to a\hat{C}\\
&\hat{C}\hat{A} \to aa
\end{aligned}
$$

Die ersten beiden Produktionen $S \to a$ und $S \to \varepsilon$ erzeugen die Wörter a^0 und a^1. Die Regel $S \to NA\hat{A}$ und die Produktion für Nichtterminal N erzeugen eine Satzform der Form $MA^i\hat{A}$ mit $i \geq 1$. Hierbei stellt $A^i\hat{A}$ quasi das spätere Wort a^n dar, welches noch $m-1$-Mal kopiert wird. Das letzte A ist dabei als $\hat{A}$ geschrieben, um es zu markieren. Offensichtlich gilt $n \geq 2$. Die Produktionen für M führen dann zu einer Satzform der Form $\hat{C}C^jA^i\hat{A}$ mit $j \geq 0$. Die Anzahl an C und $\hat{C}$ gibt an, wie viele zusätzliche Kopien von a^n erstellt werden (d.h. im Fall $j = 0$ wird a^n trotzdem einmal kopiert und damit ist $m \geq 2$). Die Verwendung von $\hat{C}$ ganz links dient dazu das linkeste C zu markieren.

Die Produktionen $CA \to aAC$ und $Ca \to aC$ schieben ein C von links nach rechts, wobei für jedes A ein a erzeugt wird, aber kleine a selbst gleich bleiben. Kommt ein C am Ende rechts an, so ist $C\hat{A}$ das Suffix. Mit $C\hat{A} \to a\hat{A}$ wird ein a für das letzte A erzeugt und C gelöscht. Das linkeste C ist durch $\hat{C}$ repräsentiert und wird genauso von links nach rechts geschoben, und erstellt für jedes A ein a, zudem werden alle vorhandenen A nun auch in a konvertiert (Regel $\hat{C}A \to aa\hat{C}$ und für das rechteste die Regel $\hat{C}\hat{A} \to aa$). Danach ist die Satzform von der Form $a^{n \cdot m}$. G ist wirklich Typ-1, da es keine verkürzenden Regeln gibt.

Beispielableitungen:

- $\underline{S} \Rightarrow \varepsilon = a^0$
- $\underline{S} \Rightarrow a = a^1$
- $\underline{S} \Rightarrow \underline{N}A\hat{A} \Rightarrow \underline{M}A\hat{A} \Rightarrow \underline{\hat{C}A}\hat{A} \Rightarrow aa\underline{\hat{C}\hat{A}} \Rightarrow aaaa$
- Variante 1 für a^6 ($6 = 3 \cdot 2$): $\underline{S} \Rightarrow \underline{N}A\hat{A} \Rightarrow \underline{N}AA\hat{A} \Rightarrow \underline{M}AA\hat{A} \Rightarrow \underline{\hat{C}A}A\hat{A} \Rightarrow aa\underline{\hat{C}A}\hat{A} \Rightarrow aaaa\underline{\hat{C}\hat{A}} \Rightarrow aaaaaa$
- Variante 2 für a^6 ($6 = 2 \cdot 3$): $\underline{S} \Rightarrow \underline{N}A\hat{A} \Rightarrow \underline{M}A\hat{A} \Rightarrow \underline{M}CA\hat{A} \Rightarrow \hat{C}\underline{CA}\hat{A} \Rightarrow \hat{C}aA\underline{C\hat{A}} \Rightarrow \underline{\hat{C}a}AaA\hat{A} \Rightarrow a\underline{\hat{C}A}aA\hat{A} \Rightarrow aaa\underline{\hat{C}a}A\hat{A} \Rightarrow aaaa\underline{\hat{C}\hat{A}} \Rightarrow aaaaaa$

FSK8-3 *LBAs* (2 Punkte)

Wir betrachten die Sprache $L = \{w \mid \#_a(w) = \#_e(w)\}$ über dem Alphabet $\{a, e\}$.

a) Erstellen Sie auf `https://turingmachinesimulator.com/` einen LBA, der L erkennt. Geben Sie sowohl einen Link zur Maschine an als auch den „Programmtext“ der Maschine.

Hinweis: Die Endzustände Ihrer Maschine dürfen keine ausgehenden Übergänge haben (d.h. $\delta(z, a) = \emptyset$ für alle Endzustände z). Ansonsten verhält sich die Maschine in der Simulation eventuell nicht so, wie wir LBAs definiert haben.

Wenn Sie Ihren LBA testen, verwenden Sie $\hat{a}$ oder $\hat{e}$, um das Ende eines Wortes zu markieren. Beachten Sie, dass Übergänge für a nicht automatisch auf $\hat{a}$ angewandt werden und dass $\hat{a}$ nicht die gleichen Übergänge wie a haben muss.

LÖSUNGSVORSCHLAG:

LBA: `http://turingmachinesimulator.com/shared/qnbapssmuc`

Idee: Wir gehen von links nach rechts über das Wort. Für jedes a, das wir dabei finden, suchen wir nach rechts ein entsprechendes e, und für jedes e suchen wir ein a. Die von links abgearbeiteten a's und e's ersetzen wir durch X und markieren so den Teil des Wortes, den wir nicht mehr besuchen müssen. Die entsprechenden e's und a's ersetzen wir durch E's und A's, um zu markieren, dass sie bereits verwendet wurden (und entsprechend für $\hat{e}$ und $\hat{a}$). Wenn am Ende nur noch X, A, E, $\hat{A}$ und $\hat{E}$ auf dem Band stehen, akzeptieren wir.

Genauer führen wir folgende Schritte in einer Schleife aus:

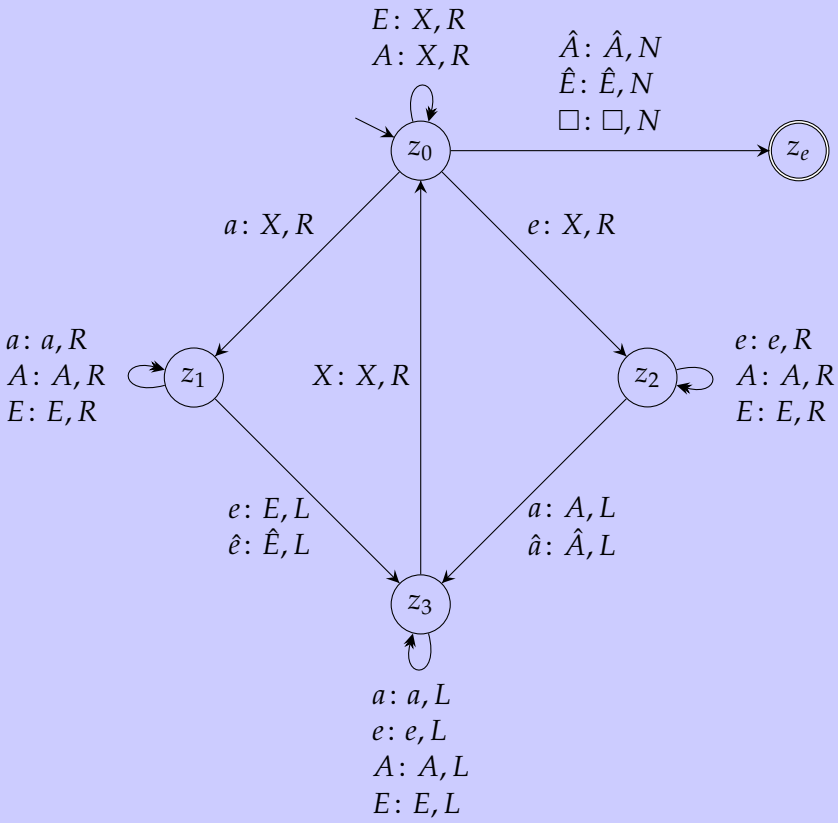
- (z_0, Startzustand) Laufe nach rechts über das Wort und halte nach dem ersten a oder e an. Das gefundene a oder e wird dabei durch X ersetzt. Ist das erste Zeichen bereits ein a oder e, halte auf dem zweiten Zeichen.
- (z_1) Ist das gefundene Zeichen ein a, laufe nach rechts und ersetze das erste e durch ein E. (z_2) Ist das gefundene Zeichen ein e, suche entsprechend nach einem a. Wir können auch $\hat{a}$ und $\hat{e}$ finden, die dann durch $\hat{A}$ und $\hat{E}$ ersetzt werden.
- (z_3) Laufe nach links über das Wort und halte rechts des ersten X.

Wir akzeptieren, wenn wir im ersten Schritt $\hat{A}$ oder $\hat{E}$ erreichen, es also keine a, e, $\hat{a}$ oder $\hat{e}$ mehr im Wort gibt. Als Spezialfall akzeptieren wir auch, wenn wir im Startzustand $\square$ lesen und das Wort also ε ist.

Als Optimierung ersetzen wir im ersten Schritt A und E durch X und verkürzen so den Teil des Wortes, den wir in den nächsten Durchläufen betrachten müssen. Das ist aber nicht nötig.

b) Geben Sie für Ihren LBA aus Teilaufgabe a) einen Zustandsgraphen an.

LÖSUNGSVORSCHLAG:



c) Geben Sie die Läufe der folgenden Wörter auf Ihrem LBA aus Teilaufgabe a) an: ε, a, $ae\hat{e}$, $eea\hat{a}$.

Hinweis: Wörter, die nicht in L liegen, erzeugen eventuell unendliche Läufe. Geben Sie in solchen Fällen ein Präfix an, aus dem ersichtlich wird, dass der Lauf unendlich ist.

LÖSUNGSVORSCHLAG:

Für ε:

$$z_0\square \vdash z_E\square$$

Für a:

$$z_0 a \vdash X z_1 \square$$

Für $ae\hat{e}$:

$$z_0 ae\hat{e} \vdash Xz_1 e\hat{e} \vdash z_3 XE\hat{e} \vdash Xz_0 E\hat{e} \vdash XXz_0\hat{e}$$

Für $eea\hat{a}$:

$$\begin{aligned} z_0 eea\hat{a} &\vdash Xz_2 eea\hat{a} \vdash Xez_2 ea\hat{a} \vdash Xeez_2 a\hat{a} \vdash Xez_3 eA\hat{a} \vdash Xz_3 eA\hat{a} \vdash z_3 XeA\hat{a} \\ &\vdash Xz_0 eA\hat{a} \vdash XEz_2 A\hat{a} \vdash XEAz_2\hat{a} \vdash XEz_3 A\hat{A} \vdash Xz_3 EA\hat{A} \\ &\vdash z_3 XEA\hat{A} \vdash Xz_0 EA\hat{A}XEz_0 A\hat{A} \vdash XEAz_0\hat{A} \vdash XEAz_E\hat{A} \end{aligned}$$

FSK8-4 ***Deterministisch kontextfrei*** (0 Punkte)

Sie dürfen als bekannt voraussetzen, dass die Sprache $L_{ww} = \{ww \mid w \in \{a,b\}^*\}$ nicht kontextfrei ist.

Betrachten Sie die folgende Grammatik G über $\Sigma = \{a,b\}$.

$$G = (\{S,A,B\}, \Sigma, \left\{\begin{array}{l} S \to A \mid B \mid AB \mid BA, \\ A \to aAa \mid aAb \mid bAa \mid bAb \mid a, \\ B \to aBa \mid aBb \mid bBa \mid bBb \mid b \end{array}\right\}, S)$$

a) Geben Sie 2 Wörter an, die in $L(G)$ liegen, und 2 Wörter, die nicht in $L(G)$ liegen. Jedes Wort soll mindestens 4 Buchstaben lang sein.

LÖSUNGSVORSCHLAG: $aaab, aabb \in L(G), abab, baba \notin L(G)$

b) Zeigen Sie, dass $L(G) = \{w \mid \forall v \in \Sigma^*.\ vv \neq w\}$.

LÖSUNGSVORSCHLAG:

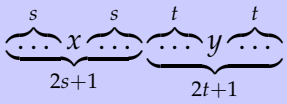
$\{w \mid \forall v \in \Sigma^*.\ vv \neq w\}$
$= \{w \mid |w| \text{ ist ungerade}\} \cup \{w \mid |w| \text{ ist gerade} \wedge \exists i \in \{1,\ldots,|w|/2\}.\ w[i] \neq w[i+|w|/2]\}$

A erzeugt die Wörter ungerader Länge mit einem a in der Mitte und B die Wörter ungerader Länge mit einem b in der Mitte.

Damit erzeugt $A \mid B$ genau die Wörter in $\{w \mid \forall v \in \Sigma^*.\ vv \neq w\} = \{w \mid |w| \text{ ist ungerade}\}$.

$AB \mid BA$ erzeugt genau die Wörter, die sich in zwei Blöcke ungerader Länge zerlegen lassen, deren Mittelbuchstaben ungleich sind. Wir wählen $s, t \in \mathbb{N}$ so, dass die Länge des ersten Blocks $2s+1$ und die Länge des zweiten Blocks $2t+1$ beträgt.

Es sind also Wörter der Form



mit $\{x, y\} = \{a, b\}$.

Die Länge des Gesamtwortes ist damit $2(s+t+1)$, der Index von x ist $s+1$, der Index von y ist $2(s+t+1) - t = s+1+(s+t+1)$.

Damit unterscheiden sich die Indizes von x und y genau um die halbe Wortlänge, also erzeugt $AB \mid BA$ die Wörter $\{w \mid |w| \text{ gerade} \wedge \exists i \in \{1, \ldots, |w|/2\}.w[i] \neq w[i + |w|/2]\}$.

$(A \mid B) \mid (AB \mid BA)$ erzeugt damit die Wörter $\{w \mid |w| \text{ ungerade}\} \cup$
$\{w \mid |w| \text{ gerade} \wedge \exists i \in \{1, \ldots, |w|/2\}.w[i] \neq w[i + |w|/2]\} = \{w \mid \forall v \in \Sigma^*.vv \neq w\}$

c) Zeigen Sie, dass $L(G)$ nicht deterministisch kontextfrei ist. (Achtung: Nicht verwechseln mit „nichtdeterministisch kontextfrei“.)

LÖSUNGSVORSCHLAG: Wäre $L(G)$ deterministisch kontextfrei, wäre auch $\overline{L(G)}$ deterministisch kontextfrei, da die deterministisch kontextfreien Sprachen unter Komplement abgeschlossen sind. Da aber $\overline{L(G)} = L_{ww}$ nicht kontextfrei ist, kann $L(G)$ auch nicht deterministisch kontextfrei sein.