Prof. Dr. Javier Esparza Garching b. München, den 14.10.17

Wiederholungsklausur

Einführung in die theoretische Informatik

Sommer-Semester 2017

Beachten Sie:

- Soweit nicht anders angegeben, ist stets eine Begründung bzw. der Rechenweg anzugeben!
- Die Klausur besteht aus 8 Aufgaben auf 9 Seiten. Insgesamt können Sie 40 Punkte erreichen.
- Nichtbeachtung der aufgabenspezifischen Hinweise kann zu Punktabzug führen.

Viel Erfolg!

Aufgabe 1 (Quiz zu Regulären und Kontextfreien Sprachen) (5 Punkte)

Geben Sie an, ob die folgenden Aussagen jeweils korrekt oder inkorrekt sind, **und begründen Sie Ihre Antwort** (z.B. Argumentation anhand von Ergebnissen der Vorlesung bzw. Übung, passendes Gegenbeispiel).

(a) Sei L regulär. Dann gilt $\Sigma^* \setminus \mathsf{L}^* = (\Sigma^* \setminus \mathsf{L})^*$. 1

Lösungsskizze: Inkorrekt. Für jedes L gilt: $\varepsilon \notin \Sigma^* \setminus \mathsf{L}^*$, aber $\varepsilon \in (\Sigma^* \setminus \mathsf{L})^*$.

(b) Typ-2 Sprachen der Chomsky-Hierarchie sind unter Schnitt und Vereinigung abgeschlossen. 1

Lösungsskizze: Inkorrekt. Typ-2 Sprachen sind genau die kontextfreien Sprachen, die aber nicht unter Schnitt abgeschlossen sind.

(c) **Definition:** Eine Grammatik wird als *schleifenfrei* bezeichnet, wenn es keine Ableitung $X \to^+ \alpha X \beta$ für $X \in V$ und $\alpha, \beta \in (\Sigma \cup V)^*$ gibt. 1

Aussage: Schleifenfreie kontextfreie Grammatiken erzeugen reguläre Sprachen.

Lösungsskizze: Korrekt. Es gibt in diesem Fall nur endlich viele Ableitungen, da wir durch wiederholtes Substituieren von Variablen mit den rechten Seiten der Produktionsregeln (terminiert, da schleifenfrei!) man eine Grammatik der Form $\mathsf{S} \to w_1 \mid w_2 \mid \cdots \mid w_n$ erhält. Somit erzeugen diese Grammatiken nur endliche Sprachen und alle endliche Sprachen sind auch reguläre Sprachen.

(d) Für jeden PDA P gibt es einen PDA P' mit $\mathsf{L}_F(\mathsf{P}) = \mathsf{L}_F(\mathsf{P}')$, so dass P' genau zwei Zustände hat: einen Anfangszustand und einen Endzustand. 1

Lösungsskizze: Korrekt. Nach Vorlesung gibt es zu jedem PDA P eine Grammatik G, die die gleiche Sprache beschreibt. G wird mit dem Verfahren der Vorlesung in einen PDA P', der die gleiche Sprache akzeptiert, nur einen Zustand hat und mit leerem Stack akzeptiert, übersetzt. Wandle diesen PDA zu einem PDA mit Endzuständen um.

(e) Sei $\mathsf{L} = \{\mathsf{a}^p \mid p \text{ ist prim}\}$ eine Sprache über dem Alphabet $\Sigma = \{\mathsf{a}\}$. Dann ist L^* eine reguläre Sprache. 1

Lösungsskizze: Korrekt. Für jedes $n \geq 2$ gibt es eine Zerlegung $n = kp$ wobei p eine Primzahl ist. Somit gilt $\mathsf{a}^n \in \mathsf{L}^*$. Somit gilt $\mathsf{L}^* = \Sigma^* \setminus \{\mathsf{a}\}$ und L^* ist eine reguläre Sprache.

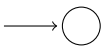
Aufgabe 2 (Reguläre Sprachen) (5 Punkte)

Geben Sie für jeden Teilausdruck des regulären Ausdrucks $((\mathsf{a}\emptyset)^*\mathsf{b} \mid \mathsf{ab})^*$ einen ε-NFA an. Verwenden Sie hierbei das Verfahren aus der Vorlesung ohne Abänderung, d.h. Sie sollen weder die regulären Ausdrücke noch die Automaten, die Sie konstruieren, vereinfachen.

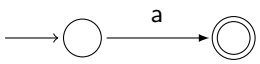
Hinweis: Es gibt 9 Teilausdrücke. Der Ausdruck selbst ist auch ein Teilausdruck.

Lösungsskizze:

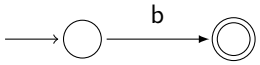
- $\emptyset$:



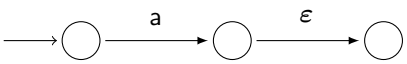
- a:



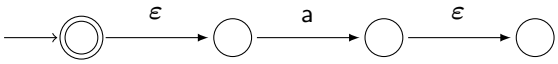
- b:



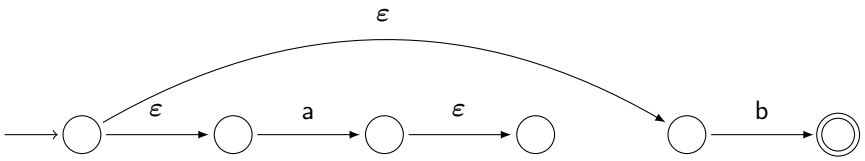
- $\mathsf{a}\emptyset$:



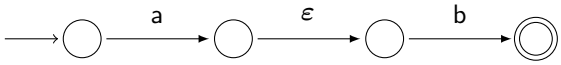
- $(\mathsf{a}\emptyset)^*$:



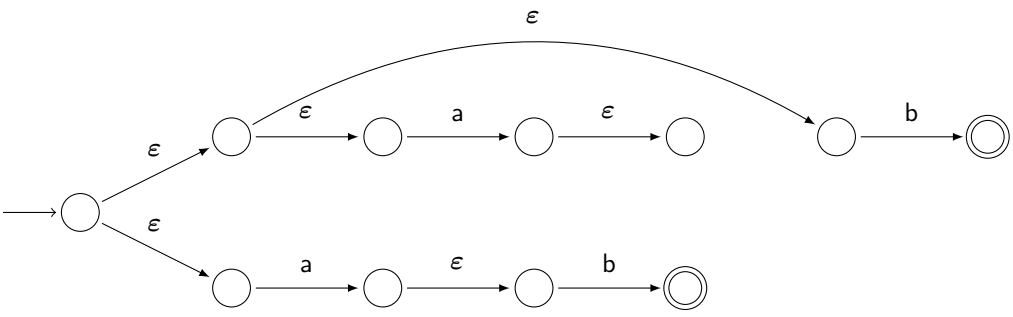
- $(\mathsf{a}\emptyset)^*\mathsf{b}$:



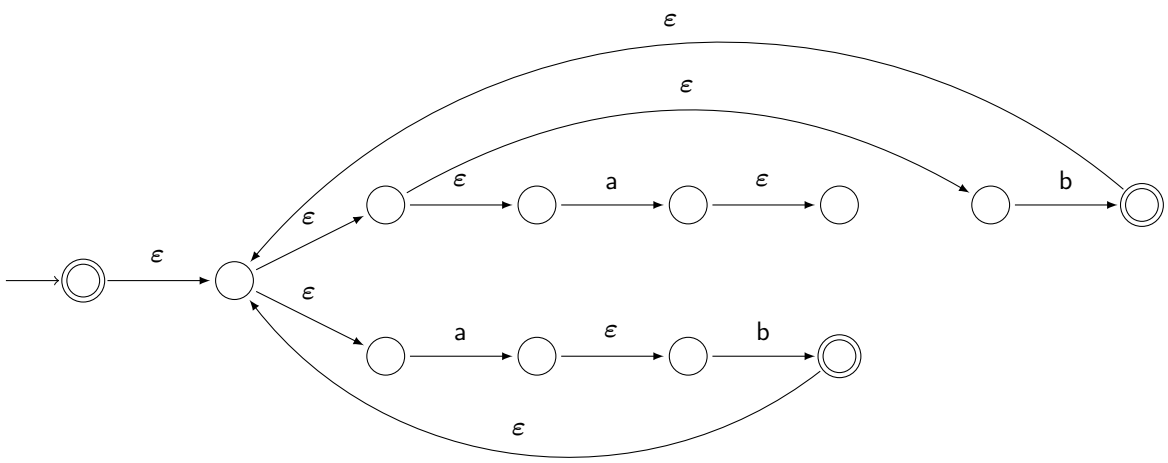
- ab:



- $(\mathsf{a}\emptyset)^*\mathsf{b} \mid \mathsf{ab}$:

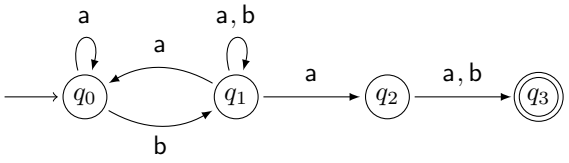


- $((a\emptyset)^*b \mid ab)^*$:

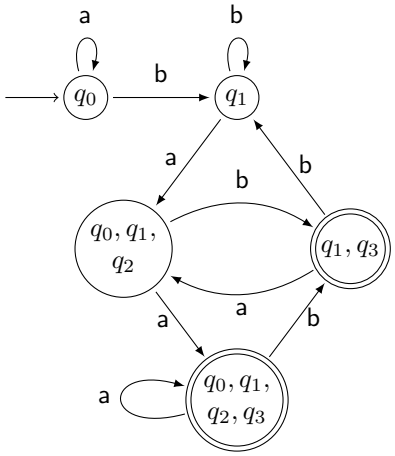


Aufgabe 3 (Determinisierung) (3 Punkte)

Determinisieren Sie unter Verwendung des Potenzmengenverfahrens aus der Vorlesung den folgenden NFA. Beschriften Sie hierbei die Zustände des konstruierten DFAs mit den korrespondierenden Zustandsmengen des NFAs.



Lösungsskizze:

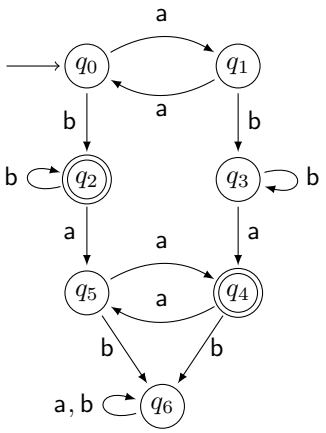


Aufgabe 4 (Reguläre Sprachen) (5 Punkte)

Betrachten Sie die Sprache $\mathsf{L} = \{\mathsf{a}^i\mathsf{b}^n\mathsf{a}^j \mid i + j \text{ ist gerade} \wedge n > 0\}$ über dem Alphabet $\Sigma = \{\mathsf{a}, \mathsf{b}\}$.

(a) Konstruieren Sie einen DFA D mit $\mathsf{L} = \mathsf{L}(\mathsf{D})$. 3

Lösungsskizze:



(b) Konstruieren Sie einen regulären Ausdruck r mit $\mathsf{L} = \mathsf{L}(r)$. 2

Lösungsskizze:

$$r = (\mathsf{aa})^*\mathsf{bb}^*(\mathsf{aa})^* \mid \mathsf{a}(\mathsf{aa})^*\mathsf{bb}^*\mathsf{a}(\mathsf{aa})^*$$

Aufgabe 5 (Lineare Grammatiken) (5 Punkte)

Betrachten Sie die rechtslineare Grammatik $\mathsf{G} = (\{S, T\}, \{\mathsf{a}, \mathsf{b}\}, P, S)$ mit P:

$$S \to \mathsf{a}S \mid T \qquad T \to \varepsilon \mid \mathsf{b} \mid \mathsf{b}T$$

(a) Geben Sie eine linkslineare Grammatik G' mit $\mathsf{L}(\mathsf{G}) = \mathsf{L}(\mathsf{G}')$ an. 1

Lösungsskizze:

$$S' \to T \mid T\mathsf{b}$$
$$T \to T\mathsf{b} \mid S$$
$$S \to S\mathsf{a} \mid \varepsilon$$

(b) Beschreiben Sie ein allgemeines Verfahren, das eine linkslineare Grammatik G_L mit $\mathsf{L}(\mathsf{G}_L) = \mathsf{L}(\mathsf{G}_R)$ aus einer rechtslinearen Grammatik G_R konstruiert. Begründen Sie hierbei die Korrektheit Ihres Verfahrens. 4

Lösungsskizze: Sei $\mathsf{G}_R = (V, \Sigma, S_R, P_R)$. Dann ist $\mathsf{G}_L = (V \cup \{S_L\}, \Sigma, S_L, P_L)$ mit P_L:

$$\begin{aligned} P_L = &\{S_R \to \varepsilon\} \\ &\cup \{S_L \to X \mid X \to \varepsilon \in P_R\} \cup \{S_L \to X\mathsf{a} \mid X \to \mathsf{a} \in P_R\} \\ &\cup \{Y \to X \mid X \to Y \in P_R\} \cup \{Y \to X\mathsf{a} \mid X \to \mathsf{a}Y \in P_R\} \end{aligned}$$

Die linkslineare Grammatik rät beginnend mit dem letzten Zeichen eine Ableitung und läuft sie rückwärts bis zum Startsymbol S_R ab.
Alternative: Übersetze Grammatik in NFA, invertiere NFA, übersetze in eine rechtslineare Grammatik, invertiere nochmal.

Aufgabe 6 (Kontextfreie Sprachen) (6 Punkte)

Betrachten Sie die kontextfreie Grammatik $\mathsf{G} = (\{S, T, U, X, Y, Z, A, B\}, \{\mathsf{a}, \mathsf{b}\}, P, S)$ in CNF mit P :

$$\begin{array}{lll} S \to AB \mid AX \mid AY \mid AZ \mid SS & X \to TB & A \to \mathsf{a} \\ T \to AB \mid AX & Y \to UZ & B \to \mathsf{b} \\ U \to AY \mid AZ & Z \to BB & \end{array}$$

(a) Bestimmen Sie mithilfe des CYK-Algorithmus für die Wörter $w_1 = \mathsf{aabbbb}$ und $w_2 = \mathsf{bababb}$, ob sie in $\mathsf{L}(\mathsf{G})$ liegen. Geben Sie die vollständige CYK-Tabellen an und halten Sie sich an die Darstellung aus den Übungen. 4

Lösungsskizze:

${}_{16}\ S, U$					
${}_{15}\ X$	${}_{26}\ Y$				
${}_{14}\ S, T$	${}_{25}$	${}_{36}$			
${}_{13}$	${}_{24}\ S, X, U$	${}_{35}$	${}_{46}$		
${}_{12}$	${}_{23}\ S, T$	${}_{34}\ Z$	${}_{45}\ Z$	${}_{56}\ Z$	
${}_{11}\ A$	${}_{22}\ A$	${}_{33}\ B$	${}_{44}\ B$	${}_{55}\ B$	${}_{66}\ B$
a	a	b	b	b	b

${}_{16}$					
${}_{15}$	${}_{26}\ S$				
${}_{14}$	${}_{25}\ S$	${}_{36}$			
${}_{13}$	${}_{24}$	${}_{35}$	${}_{46}\ S, X, U$		
${}_{12}$	${}_{23}\ S, T$	${}_{34}$	${}_{45}\ S, T$	${}_{56}\ Z$	
${}_{11}\ B$	${}_{22}\ A$	${}_{33}\ B$	${}_{44}\ A$	${}_{55}\ B$	${}_{66}\ B$
b	a	b	a	b	b

Somit ist $w_1 \in \mathsf{L}(\mathsf{G})$ und $w_2 \notin \mathsf{L}(\mathsf{G})$.

(b) Bestimmen Sie unter Verwendung Ihrer Tabellen jeweils alle Infixe von w_1 und w_2 die in $\mathsf{L}(\mathsf{G})$ liegen. 2

Lösungsskizze:

- $w_1 : \{\mathsf{aabb}, \mathsf{aabbbb}, \mathsf{abb}, \mathsf{ab}\}$
- $w_2 : \{\mathsf{ababb}, \mathsf{abab}, \mathsf{abb}, \mathsf{ab}\}$

Aufgabe 7 (Quiz zu Entscheidbarkeit und Turingmaschinen) (5 Punkte)

Geben Sie an, ob die folgenden Aussagen jeweils korrekt oder inkorrekt sind, **und begründen Sie Ihre Antwort** (z.B. Argumentation anhand von Ergebnissen der Vorlesung bzw. Übung, passendes Gegenbeispiel).

(a) Eine PCP-Instanz hat entweder keine oder unendlich viele Lösungen. 1

Lösungsskizze: Korrekt. Wir nehmen an es gibt nur endlich viele, aber mindestens eine Lösung. Sei $w = i_1 i_2 \dots i_n \neq \varepsilon$ eine Lösung und somit gilt $v_{i_1} v_{i_2} \dots v_{i_n} = u_{i_1} u_{i_2} \dots u_{i_n}$ (oben und unten sind gleich). Dann gilt aber auch $(v_{i_1} v_{i_2} \dots v_{i_n})^j = (u_{i_1} u_{i_2} \dots u_{i_n})^j$ für alle j. Somit ist die unendliche Menge $\{w\}^+$ auch alles Lösungen.

(b) Sei G_1 eine kontextfreie und G_2 eine rechtslineare Grammatik. Dann ist $\mathsf{L}(\mathsf{G}_1) \cap \mathsf{L}(\mathsf{G}_2) \stackrel{?}{=} \emptyset$ entscheidbar. 1

Lösungsskizze: Korrekt. Konstruiere PDA P für G_1 und DFA D für G_2. Konstruiere Produkt-PDA und übersetzte in zurück in eine CFG $\mathsf{G}_\times$ und prüfe, ob die Startvariable S produktiv ist.

Verkürzte Argumentation: eine rechtslineare Grammatik beschreibt eine reguläre Sprache. Laut Übung ist dieser Schnitt wieder kontextfrei und man kann somit anhand der Grammatik prüfen, ob die Startvariable produktiv ist, um auf Sprachleerheit zu testen.

(c) Die Sprache $\mathsf{L} = \{w \mid M_w \text{ akzeptiert mindestens ein Wort}\}$ ist rekursiv aufzählbar. 1

Lösungsskizze: Korrekt, da χ_{L} semi-entscheidbar ist und somit ist die Menge rekursiv aufzählbar. χ_{L} kann wie folgt berechnet werden: Für eine gegebene Schranke c: enumeriere alle Wörter $|w'| \leq c$ und simuliere $M_w[w']$ für maximal c Schritte. Falls die TM ein Wort akzeptiert, terminiere und akzeptiere w. Führe diesen Algorithmus für $c = 1, 2, \dots$ aus.

(d) **Definition:** Eine Single-Direction-Maschine ist eine DTM, die während eines Laufes Ihren Kopf entweder immer nur nach links oder immer nur nach rechts bewegt. Sie können davon ausgehen, dass es keine Transitionen mit N gibt, d.h. der Kopf wird immer bewegt. 1

Aussage: Das Wortproblem ist für Single-Direction-Maschinen entscheidbar.

Lösungsskizze: Korrekt. Da die TM den Kopf immer nur nach rechts oder links bewegt, verlässt der Kopf nach spätestens $|w|$ Schritten das beschriebene Band und wird dann nur noch $\square$ lesen. Dann dauert es maximal $|Q| + 1$ Schritte bis ein Zustand sich wiederholt und von diesem Zeitpunkt an wird die DTM sich in einer Schleife befinden. Somit simuliere TM für $|Q| + |w| + 1$ Schritte. Falls das Wort innerhalb dieser Zeit nicht akzeptiert wird, wird es nie akzeptiert.

(e) Sei $\mathsf{L} \in \mathsf{NP}$. Dann gilt $\mathsf{L}^* \in \mathsf{NP}$. 1

Lösungsskizze: Korrekt. Ein geeignetes Zertifikat für $w \in \mathsf{L}^*$ ist $c = w_1 \# c_1 \# \dots w_n \# c_n$. Der deterministische Verifikator prüft dann $w = w_1 \dots w_n$ und führt dann für jedes $w_i \# c_i$ den Verifikator für L aus.

Aufgabe 8 (Reduktionen) (6 Punkte)

Mit NOT-k-UNIV bezeichnen wir das Problem:

Gegeben: Ein ε-NFA N mit n Zuständen über dem Alphabet $\Sigma = \{0, 1\}$ und eine Zahl $m \leq n$.

Frage: Gibt es ein Wort w der Länge m, so dass N dieses Wort nicht akzeptiert?

...und in Mengenschreibweise:

$$\text{NOT-k-UNIV} := \{(\mathsf{N}, m) \mid \mathsf{N} = (Q, \Sigma, \delta, q_0, F) \text{ ist } \varepsilon\text{-NFA} \wedge m \leq |Q| \wedge \mathsf{L}(\mathsf{N}) \cap \Sigma^m \neq \Sigma^m\}$$

(a) Zeigen Sie, dass NOT-k-UNIV NP-vollständig ist. Verwenden Sie hierbei eine geeignete Reduktion von 3-KNF-SAT auf NOT-k-UNIV. [4]

(b) Wenden Sie Ihre Reduktion auf $(x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee x_3)$ an. Geben Sie den konstruierten ε-NFA graphisch an. [2]

Hinweis: Betrachten Sie die Menge aller nichterfüllenden Belegungen für die Formel F, die genau m Variablen hat, als Wörter der Länge m.

Lösungsskizze:

(a)

- Idee:
  Der ε-NFA N erkennt alle nicht erfüllenden Belegungen von F.
- Reduktion:
  Sei $F = \bigwedge_{i=1}^{k} C_i$ eine Formel in 3-KNF mit m Variablen $(x_1, x_2, \ldots x_m)$ und mit $C_i = \bigvee_{j=1}^{3} L_{ij}$ für jedes i.
  Der ε-NFA wird dann wie folgt definiert $\mathsf{N} = (\{q_0\} \cup \{q_{ij} \mid i \in [1;k] \wedge j \in [0;m]\}, \{0,1\}, \delta, q_0, \{q_{im} \mid i \in [1;k]\})$ mit δ:

$$\begin{aligned}\delta = &\{(q_0, \varepsilon, q_{i0}) \mid i \in [1;k]\} \\ &\cup \{(q_{i(k-1)}, 0, q_k) \mid k \in [1,m] \wedge \neg x_k \notin \{L_{i1}, L_{i2}, L_{i3}\}\} \\ &\cup \{(q_{i(k-1)}, 1, q_k) \mid k \in [1,m] \wedge x_k \notin \{L_{i1}, L_{i2}, L_{i3}\}\}\end{aligned}$$

  Die Reduktion bildet dann F auf (N, m) ab.
- Korrektheit:
  F ist erfüllbar gdw. nicht alle Belegungen nicht erfüllend sind. Sei $\sigma(F) = 0$ eine nicht erfüllende Belegung. Dann existiert insbesondere eine Klausel C_i mit $\sigma(C_i) = \sigma(L_{i1}) + \sigma(L_{i2}) + \sigma(L_{i3}) = 0$. Dass Wort $w = \sigma(x_1)\sigma(x_2)\ldots\sigma(x_m)$ wird dann von q_{i0} aus akzeptiert und es gilt somit $w \in \mathsf{L}(\mathsf{N})$. Der Beweis ist analog für $\sigma(F) = 1$
- Polynomielle Reduzierbarkeit:
  Der konstruierte ε-NFA hat $\mathcal{O}(km)$ Zuständen und ein konstantes Alphabet und hat somit maximal Größe polynomiell in der Größe von F. Er kann auch in polynomieller Zeit berechnet werden, da jede Komponente während des Lesens einer Klausel C direkt konstruiert werden kann.
- NOT-k-UNIV $\in$ NP:
  Rate $w \in \Sigma^m$ (dies ist möglich, da m kleiner als N ist). Übersetze den ε-NFA in einen NFA N' und prüfe $w \notin \mathsf{L}(\mathsf{N}')$ mithilfe des polynomiellen Algorithmus für das Wortproblem (auf NFAs) der Vorlesung.

(b) $(\mathsf{N}, 4)$ mit N: