Lehrstuhl für Theoretische Informatik
Fakultät für Informatik
Technische Universität München

Esolution
Sticker mit SRID hier einkleben

Hinweise zur Personalisierung:

- Ihre Prüfung wird bei der Anwesenheitskontrolle durch Aufkleben eines Codes personalisiert.
- Dieser enthält lediglich eine fortlaufende Nummer, welche auch auf der Anwesenheitsliste neben dem Unterschriftenfeld vermerkt ist.
- Diese wird als Pseudonym verwendet, um eine eindeutige Zuordnung Ihrer Prüfung zu ermöglichen.

Einführung in die Theoretische Informatik

Klausur:	IN0011 / Endterm	**Datum:**	Mittwoch, 3. August 2022
Prüfer:	Prof. Dr. Dr. h.c. Javier Esparza	**Uhrzeit:**	08:15 – 11:15

	A 1	A 2	A 3	A 4	A 5	A 6	A 7	A 8	A 9
I									

Bearbeitungshinweise

- Diese Klausur umfasst **16 Seiten** mit insgesamt **9 Aufgaben**.
  Bitte kontrollieren Sie jetzt, dass Sie eine vollständige Angabe erhalten haben.
- Die Gesamtpunktzahl in dieser Klausur beträgt 100 Punkte.
- Das Heraustrennen von Seiten aus der Prüfung ist untersagt.
- Als Hilfsmittel sind zugelassen:
  - ein **beidseitig handschriftlich beschriebenes DIN A4 Blatt**
  - ein **analoges Wörterbuch** Deutsch $\leftrightarrow$ Muttersprache **ohne Anmerkungen**
- Mit * gekennzeichnete Teilaufgaben sind ohne Kenntnis der Ergebnisse vorheriger Teilaufgaben lösbar.
- **Es werden nur solche Ergebnisse gewertet, bei denen der Lösungsweg erkennbar ist.**
- Schreiben Sie weder mit roter / grüner Farbe noch mit Bleistift.
- Schalten Sie alle mitgeführten elektronischen Geräte vollständig aus, verstauen Sie diese in Ihrer Tasche und verschließen Sie diese.
- $0 \in \mathbb{N}$

Hörsaal verlassen von _____ bis _____ / Vorzeitige Abgabe um _____

Aufgabe 1 Reguläre und kontextfreie Sprachen (10 Punkte)

Für die folgenden Fragen ist eine Begründung nicht gefordert. Sie erhalten die Punkte auf eine Teilaufgabe genau dann, wenn Sie alle Antwortmöglichkeiten korrekt angekreuzt haben. Es ist immer mindestens eine Antwortmöglichkeit richtig. Jede Teilaufgabe bringt 2 Punkte.

Kreuzen Sie richtige Antworten an ☒
Kreuze können durch vollständiges Ausfüllen gestrichen werden ■
Gestrichene Antworten können durch nebenstehende Markierung erneut angekreuzt werden ×■

In dieser Aufgabe verwenden wir durchgehend das Alphabet $\Sigma := \{a, b\}$.

a) Seien $A, B, C \subseteq \Sigma^*$ Sprachen. Welche der folgenden Aussagen sind wahr?

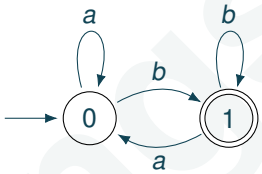
- ☐ $A \neq B \Rightarrow A^* \neq B^*$ Gegenbeispiel: $A = \{a, b\}, B = \{a, b, ab\}$
- ☒ $A(B \cup C) = AB \cup AC$
- ☐ $|A||B| \leq |AB|$ Gegenbeispiel: $A = B = \{a, aa\}$

b) Sei r ein regulärer Ausdruck und M ein NFA mit n Zuständen und $L(M) = L(r)$. Welche Aussagen sind wahr?

- ☒ Es gibt einen NFA für $L(ar)$ mit höchstens $n+1$ Zuständen. Wir fügen einen neuen Initialzustand ein, mit einer a-Transition zum alten Initialzustand.
- ☐ Es gibt einen NFA für $L(rar)$ mit höchstens $n+1$ Zuständen. Gegenbeispiel: $r = a$. Es gibt einen NFA M mit 2 Zuständen, aber für $L(aaa)$ hat jeder NFA mindestens 4 Zustände.
- ☒ Es gibt einen NFA für $L(ra)$ mit höchstens $n+1$ Zuständen. Wir fügen einen neuen Finalzustand f ein. Die alten Finalzustände sind nicht mehr final, und erhalten eine a-Transition zu f.

c) Wieviele Zustände hat der minimale DFA für die Sprache $L((a^*b)^+)$? (Beachten Sie $r^+ = rr^*$.)

- ☐ 1
- ☒ 2
- ☐ 3
- ☐ ≥ 4



d) Sei G die Grammatik mit Produktionen $S \to aSb \mid Sb \mid b$. Wie viele Zustände hat der kleinste PDA M mit $L_\epsilon(M) = L(G)$? (Beachten Sie, dass $L_\epsilon(M)$ die Wörter sind, die M über leeren Keller akzeptiert.)

- ☒ 1 Wir wissen, dass es für jede CFL einen PDA mit einem Zustand gibt.
- ☐ 2
- ☐ 3
- ☐ ≥ 4

e) Sei G eine kontextfreie Grammatik mit 7 Produktionen. Welche Aussagen sind wahr?

- ☐ G hat höchstens 7 erreichbare Nichtterminale.
- ☒ G hat höchstens 7 nützliche Nichtterminale.
- ☒ G hat höchstens 7 erzeugende Nichtterminale.

Siehe Q7.5.

Aufgabe 2 **Rekursive Prozeduren (12 Punkte)**

Sei $\Sigma = \{a, b\}$ das Alphabet.

a)* Geben Sie eine rekursive Prozedur $\text{start}_a(r)$ an, die für einen regulären Ausdruck r ausgibt, ob es ein Wort $w \in L(r)$ gibt, das mit a anfängt. Formal gilt also $\text{start}_a(r) := \big(L(r) \cap \{aw \mid w \in \Sigma^*\} \neq \emptyset\big)$.
Sie dürfen hierfür die Hilfsprozeduren $\text{empty}(r) := \big(L(r) = \emptyset\big)$ und $\text{contains-}\varepsilon(r) := (\varepsilon \in L(r))$ verwenden.

0 1 2 3 4

- $\text{start}_a(\emptyset) = \text{false}$
- $\text{start}_a(\varepsilon) = \text{false}$
- $\text{start}_a(x) = \begin{cases} \text{true} & \text{wenn } x = a \\ \text{false} & \text{sonst} \end{cases}$
- $\text{start}_a(\alpha\beta) = \begin{array}{l}(\text{start}_a(\alpha) \wedge \neg\,\text{empty}(\beta)) \\ \vee\ (\text{contains-}\varepsilon(\alpha) \wedge \text{start}_a(\beta))\end{array}$
- $\text{start}_a(\alpha \mid \beta) = \text{start}_a(\alpha) \vee \text{start}_a(\beta)$
- $\text{start}_a(\alpha) = \text{start}_a(\alpha)$

b)* Für einen regulären Ausdruck r sei $\text{contains}_{ab}(r)$ die Aussage, dass ab ein Teilwort von **jedem** Wort $w \in L(r)$ ist. Formal soll also $\text{contains}_{ab}(r) := \big(L(r) \subseteq \Sigma^*\{ab\}\Sigma^*\big)$ gelten. Anhand folgender Äquivalenzen könnte man eine rekursive Prozedur für $\text{contains}_{ab}(r)$ konstruieren – allerdings sind zwei der Fälle falsch. Sei $x \in \Sigma$, und seien α, β reguläre Ausdrücke:

0 1 2 3 4

(1) $\text{contains}_{ab}(\emptyset) \Leftrightarrow \text{true}$

(2) $\text{contains}_{ab}(\epsilon) \Leftrightarrow \text{false}$

(3) $\text{contains}_{ab}(x) \Leftrightarrow \text{false}$

(4) $\text{contains}_{ab}(\alpha^*) \Leftrightarrow \text{contains}_{ab}(\alpha)$

(5) $\text{contains}_{ab}(\alpha\beta) \Leftrightarrow \text{contains}_{ab}(\alpha) \vee \text{contains}_{ab}(\beta)$

(6) $\text{contains}_{ab}(\alpha \mid \beta) \Leftrightarrow \text{contains}_{ab}(\alpha) \wedge \text{contains}_{ab}(\beta)$

Welche zwei Fälle sind falsch? Begründen Sie dies jeweils anhand eines passenden Gegenbeispiels.

(4) ist falsch: Sei $\alpha = ab$. Dann gilt $\text{contains}_{ab}(ab)$ und es folgt $\text{contains}_{ab}((ab)^*) \Leftrightarrow \text{contains}_{ab}(ab) \Leftrightarrow \text{true}$. Aber $\varepsilon \in L((ab)^*)$.
(5) ist falsch: Sei $\alpha = a$ und $\beta = b$. Dann gilt $\text{contains}_{ab}(a) \Leftrightarrow \text{false} \Leftrightarrow \text{contains}_{ab}(b)$ und es folgt $\text{contains}_{ab}(ab) \Leftrightarrow \text{contains}_{ab}(a) \wedge \text{contains}_{ab}(b) \Leftrightarrow \text{false}$. Aber jedes Wort in $L(ab)$ enthält das Teilwort ab.

c)* Seien $\text{AG}(r)$ und $\text{AU}(r)$ die Aussagen, dass alle Wörter eines regulären Ausdrucks r gerade bzw. ungerade Länge haben. Formal gilt $\text{AG}(r) := (L(r) \subseteq (\Sigma\Sigma)^*)$ und $\text{AU}(r) := (L(r) \subseteq \Sigma(\Sigma\Sigma)^*)$. Beweisen Sie, dass $\text{AG}(rs) \Leftrightarrow (\text{AG}(r) \wedge \text{AG}(s)) \vee (\text{AU}(r) \wedge \text{AU}(s))$ gilt, für reguläre Ausdrücke r, s, die das Symbol $\emptyset$ **nicht enthalten**.

0 1 2 3 4

„$\Leftarrow$“: $\text{AG}(r) \wedge \text{AG}(s)$ impliziert $L(rs) = L(r)L(s) \subseteq (\Sigma\Sigma)^*(\Sigma\Sigma)^* = (\Sigma\Sigma)^*$, und aus $\text{AU}(r) \wedge \text{AU}(s)$ erhalten wir $L(rs) = L(r)L(s) \subseteq \Sigma(\Sigma\Sigma)^*\Sigma(\Sigma\Sigma)^* = (\Sigma\Sigma)^+ \subseteq (\Sigma\Sigma)^*$.

„$\Rightarrow$“: Da r, s das Symbol $\emptyset$ nicht enthalten, sind $L(r), L(s)$ nicht leer. Wir zeigen die Aussage über Kontraposition, es gilt also weder $\text{AG}(r) \wedge \text{AG}(s)$ noch $\text{AU}(r) \wedge \text{AU}(s)$. Nun unterscheiden wir drei Fälle.
Fall 1: $\text{AG}(r)$ gilt. Dann kann $\text{AG}(s)$ nicht gelten, und es gibt $u \in L(r), v \in L(s)$ mit $|u|$ gerade, $|v|$ ungerade. (Beachte $L(r) \neq \emptyset$).
Fall 2: $\text{AU}(r)$ gilt. Dann kann $\text{AU}(s)$ nicht gelten, und es gibt $u \in L(r), v \in L(s)$ mit $|u|$ ungerade, $|v|$ gerade. (Beachte $L(r) \neq \emptyset$).
Fall 3: Es gibt $u_1, u_2 \in L(r)$ mit $|u_1|$ gerade, $|u_2|$ ungerade. Da $L(s) \neq \emptyset$ gibt es ein $v \in L(s)$, und entweder $u_1 v$ oder $u_2 v$ hat ungerade Länge.

Aufgabe 3 Chomsky-Normalform (10 Punkte)

In dieser Aufgabe geht es darum, eine kontextfreie Grammatik (CFG) in Chomsky-Normalform (CNF) zu konvertieren. Wir führen allerdings jeden Schritt einzeln aus, und jeweils auf einer anderen Grammatik – Sie können also die Aufgabenteile unabhängig voneinander bearbeiten.
Eine Grammatik $G = (V, \Sigma, P, S)$ ist in CNF, wenn jede Produktion $(X \to \alpha) \in P$, mit $X \in V$ und $\alpha \in (\Sigma \cup V)^*$, folgende Bedingungen erfüllt:

(1) $\alpha \in \Sigma \cup V^*$; Terminale dürfen nur in Produktionen der Länge 1 erzeugt werden.

(2) $|\alpha| \leq 2$; jede Produktion hat höchstens Länge 2.

(3) $\alpha \neq \varepsilon$; es gibt keine ε-Produktionen.

(4) $\alpha \notin V$; es gibt keine Kettenproduktionen.

Achtung: Die Teilaufgaben fragen jeweils spezifisch nach dem Ergebnis, das sich durch die Ausführung des Algorithmus aus der Vorlesung ergibt, nicht nach einer beliebigen äquivalenten CFG, die den Bedingungen genügt. Details, wie etwa die Namen der Variablen oder die Reihenfolge, in der Produktionen betrachtet werden, können Sie frei wählen.

0 1 2

a)* *Entfernen von Terminalen in langen Produktionen.* Die CFG G_a ist gegeben durch folgende Produktionen:

$$S \to c \mid X \mid aSX \mid SdS \mid dd \qquad X \to a \mid aX$$

Führen Sie den ersten Schritt des Algorithmus zur Überführung in CNF aus und geben Sie die Produktionen einer CFG G'_a an, so dass $L(G_a) = L(G'_a)$ gilt und G'_a Bedingung (1) erfüllt.

(Platz für Zwischenschritte / Erklärungen)

Fertige Grammatik G'_a:

$S \to c \mid X \mid ASX \mid SDS \mid DD$
$X \to a \mid AX$
$A \to a$
$D \to d$

0 1 2

b)* *Entfernen langer Produktionen.* Die CFG G_b ist gegeben durch die folgenden Produktionen:

$$S \to c \mid ABSA \qquad A \to a \mid AAA \qquad B \to d$$

Führen Sie den zweiten Schritt des Algorithmus zur Überführung in CNF aus und geben Sie die Produktionen einer CFG G'_b an, so dass $L(G_b) = L(G'_b)$ und G'_b Bedingung (1) und (2) erfüllt.

(Platz für Zwischenschritte / Erklärungen)

Fertige Grammatik G'_b:

$S \to c \mid AX_{BSA}$ $\quad X_{BSA} \to BX_{SA}$
$A \to a \mid AX_{AA}$ $\quad X_{SA} \to SA$
$B \to d$ $\quad X_{AA} \to AA$

c)* *Entfernen von ε-Produktionen.* Die CFG G_c ist gegeben durch folgende Produktionen:

0 1 2 3

$$S \to AD \mid D \qquad C \to BA$$
$$A \to a \mid BB \qquad D \to d \mid SS \mid DB$$
$$B \to b \mid \varepsilon \qquad E \to e \mid CA$$

Führen Sie den dritten Schritt des Algorithmus zur Überführung in CNF aus. Geben Sie die Produktionen einer CFG G_c' an, so dass $L(G_c) = L(G_c')$ und G_c' Bedingungen (1), (2) und (3) erfüllt.

(Platz für Zwischenschritte / Erklärungen)

Die Nichtterminale, die ε erzeugen, berechnen wir iterativ:
1. $\{B\}$, 2. $\{B, A\}$, 3. $\{B, A, C\}$, 4. $\{B, A, C, E\}$.
Dann fügen wir die entsprechenden neuen Produktionen ein und entfernen die ε-Produktionen:

Fertige Grammatik G_c':

$S \to AD \mid D$
$A \to a \mid BB \mid B$
$B \to b$
$C \to BA \mid B \mid A$
$D \to d \mid SS \mid DB \mid D$
$E \to e \mid CA \mid C \mid A$

d)* *Entfernen von Kettenproduktionen.* Die CFG G_d ist gegeben durch die Produktionen:

0 1 2 3

$$S \to A \mid D \qquad C \to c \mid B \mid D$$
$$A \to a \mid S \mid AA \qquad D \to d$$
$$B \to b \mid B \mid BC$$

Führen Sie den vierten Schritt des Algorithmus zur Überführung in CNF aus und geben Sie die Produktionen einer CFG G_d' in CNF an, so dass $L(G_d) = L(G_d')$ gilt.

(Platz für Zwischenschritte / Erklärungen)

Die Variablen, die wir mit Kettenproduktionen ineinander umwandeln können, berechnen wir iterativ:
1. $S \to A, S \to D, A \to S, B \to B, C \to B, C \to D$
2. zusätzlich $S \to S, A \to A, A \to D$
Dann entfernen wir die Kettenproduktionen und fügen stattdessen entsprechende neue Produktionen ein:

Fertige Grammatik G_d':

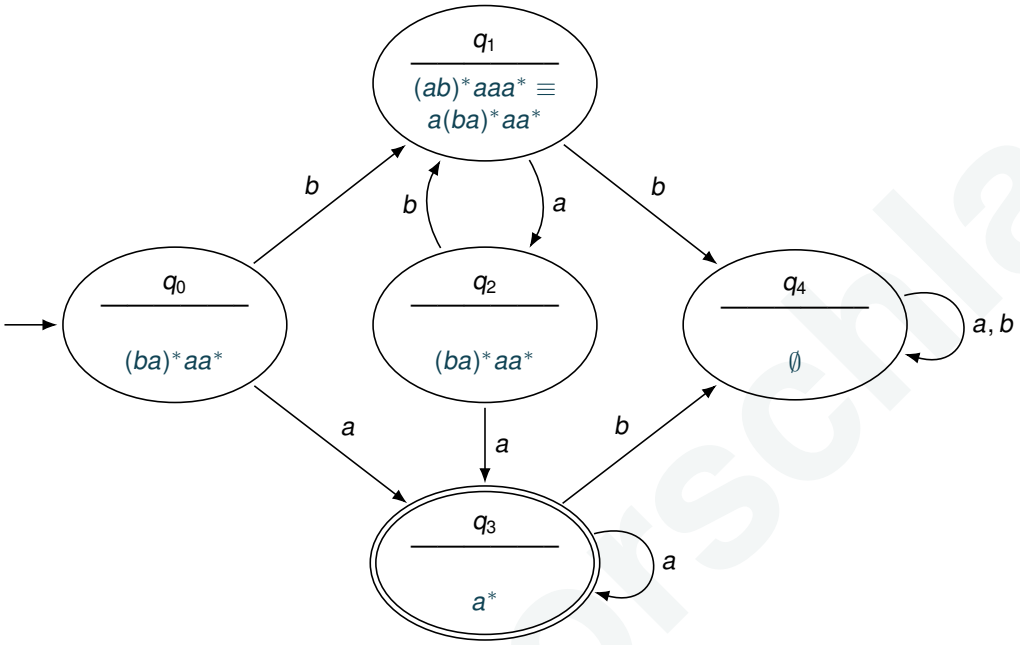
$S \to a \mid d \mid AA$
$A \to a \mid d \mid AA$
$B \to b \mid BC$
$C \to b \mid c \mid d \mid BC$
$D \to d$

Aufgabe 4 Residualsprachen (10 Punkte)

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA und $q \in Q$ ein Zustand von M. Die Menge $L_M(q) := \{w \in \Sigma^* : \hat{\delta}(q, w) \in F\}$ sind genau die Wörter, die M akzeptiert, wenn M in Zustand $q \in Q$ startet. Es gilt, dass $L_M(q)$ immer eine Residualsprache von $L(M)$ ist.

0 1 2 3 4 5

a)* Der folgende DFA M akzeptiert die Sprache $L((ba)^*aa^*)$. Beschriften Sie jeden Zustand q mit einem regulären Ausdruck für die Sprache $L_M(q)$.



0 1 2 3

b)* Sei N ein DFA mit Zuständen $\{p_1, p_2, p_3, p_4\}$. Es gilt $L_N(p_i) = L(r_i)$, für $i \in \{1, 2, 3, 4\}$ und reguläre Ausdrücke $r_1 = a(ba)^*$, $r_2 = (ba)^*$, $r_3 = (ab)^*a$, $r_4 = \emptyset$. Alle Zustände von N sind erreichbar. Ist N minimal? Begründen Sie Ihre Antwort.

☐ ja ☒ nein

Begründung: Ein DFA ist nur dann minimal, wenn jeder Zustand eine unterschiedliche Residualsprache hat. Da $L(r_1) = L(r_3)$, ist N nicht minimal.

0 1 2

c)* Ist der DFA M aus der Teilaufgabe a) minimal? Begründen Sie ihre Antwort.

☐ ja ☒ nein

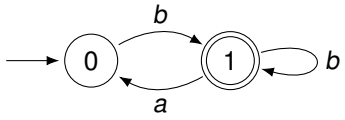
Begründung: Der DFA aus Teilaufgabe a) ist nicht minimal, da die regulären Ausdrücke von q_0 und q_2 äquivalent sind.
Alternative Lösung: M ist nicht minimal, da die Zustände q_0 und q_2 äquivalent sind: sowohl ihre a- als auch ihre b-Transition führen zum selben Zustand.

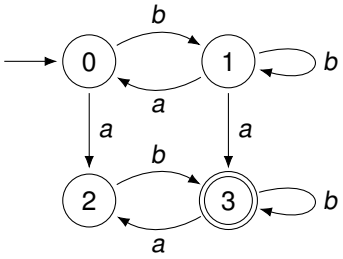
Aufgabe 5 Ein a geht noch (10 Punkte)

Sei $\Sigma := \{a, b\}$. Für eine Sprache $L \subseteq \Sigma^*$ bezeichnet $L^\curlyvee$ die Sprache, die man erhält, wenn man den Wörtern aus L an einer beliebigen Stelle ein a einfügt. Formal gilt also $L^\curlyvee = \{uav : uv \in L,\ u, v \in \Sigma^*\}$. Wir erhalten z.B. $\{a, ab\}^\curlyvee = \{aa, aab, aba\}$ und $(\{ab\}^*)^\curlyvee = L((ab)^*(a \mid aab)(ab)^*)$. Beachten Sie insbesondere, dass das Einfügen nicht optional ist.

a)* Sei M der folgende NFA. Geben Sie einen NFA M' mit $L(M') = L(M)^\curlyvee$ und höchstens 4 Zuständen an.

0 1 2





b)* Sei $M = (Q, \Sigma, \delta, q_0, F)$ nun ein beliebiger NFA. Konstruieren Sie einen NFA $M' = (Q', \Sigma, \delta', q_0', F')$ mit $L(M') = L(M)^\curlyvee$ und $|Q'| \leq 2|Q|$. Schildern Sie insbesondere Ihre Idee in natürlicher Sprache und geben Sie Q', δ', q_0', und F' präzise an.

0 1 2 3 4 5

Die Idee ist, dass wir zwei Kopien M_1, M_2 von M erzeugen. Der NFA startet in M_1. Zu jedem Zeitpunkt ist es möglich, in M_1 ein a zu lesen, und so nach M_2 in den gleichen Zustand zu wechseln. In M_2 kann dann das Wort akzeptiert werden.
Formal wählen wir also $Q' := Q \cup \{q_a : q \in Q\}$, $q_0' := q_0$, $F' := \{q_a : q \in F\}$ und Transitionen $\delta' := \delta \cup \{(q, a, q_a) : q \in Q\} \cup \{(q_a, x, r_a) : (q, x, r) \in \delta\}$.

c)* Geben Sie $L_1, L_2 \subseteq \Sigma^*$ mit $L_1 \neq L_2$ und $L_1^\curlyvee = L_2^\curlyvee$ an. **Hinweis:** Es gibt solche L_1, L_2 mit $|L_1|, |L_2| \leq 3$.

0 1 2 3

$L_1 := \{aab, baa\}$, $L_2 := \{aab, aba, baa\}$. Dann gilt $L_1^\curlyvee = L_2^\curlyvee = \{aaab, aaba, abaa, baaa\}$.

Aufgabe 6 **Sensitivität (18 Punkte)**

Sei $\Sigma = \{a, b\}$. Gegeben sind die folgenden vier Sprachen:

$$L_1 = \{a^n b^n a^m b^m \mid n, m \in \mathbb{N}\}$$
$$L_2 = \{a^n b^m a^m b^n \mid n, m \in \mathbb{N}\}$$
$$L_3 = \{a^n b^m a^n b^m \mid n, m \in \mathbb{N}\}$$
$$L_4 = \{a^n b^m b^n a^m \mid n, m \in \mathbb{N}\}$$

0 1 2

a)* Genau eine dieser Sprachen ist nicht kontextfrei. Welche? (Ohne Begründung.)

Die nicht kontextfreie Sprache ist L_3 .

0 1 2 3 4 5 6

b)* Geben Sie für die drei anderen Sprachen jeweils eine kontextfreie Grammatik (CFG) mit höchstens 5 Produktionen an.

Die Produktionen der CFG für L_1 sind:

$S \to XX$
$X \to aXb \mid \varepsilon$

Die Produktionen der CFG für L_2 sind:

$S \to aSb \mid X$
$X \to bXa \mid \varepsilon$

Die Produktionen der CFG für L_4 sind:

$S \to XY$
$X \to aXb \mid \varepsilon$
$Y \to bYa \mid \varepsilon$

(Der Trick ist, dass $b^m b^n = b^{m+n} = b^n b^m$.)

0 1 2

c)* Welche der Sprachen L_1, L_2, L_3, L_4 sind regulär? (Ohne Begründung.)

Keine der vier Sprachen ist regulär.

↪

d)* Zeigen Sie von einer der Sprachen L_1, L_2, L_3, L_4, dass sie nicht regulär ist, indem Sie das Pumping-Lemma für reguläre Sprachen verwenden.

0 1 2 3 4

Wir wählen L_1. (Der Beweis funktioniert analog für alle vier Sprachen.) Wir führen einen Widerspruchsbeweis und nehmen an, dass L_1 regulär wäre. Dann gilt die Aussage des Pumping-Lemma (PL) für reguläre Sprachen. Sei $n \in \mathbb{N}$ die PL-Zahl. Nun wählen wir das Wort $z := a^n b^n a^n b^n$. Nach PL existiert eine Zerlegung $z = uvw$. Es gilt $|uv| \leq n$ und $v \neq \varepsilon$, wir können also $v = a^i$ schreiben, für ein $i > 0$. Nun gilt auch $uv^0w \in L_1$, aber $uw = a^{n-i}b^n a^n b^n \notin L$, da $n - i \neq n$. Dies ist ein Widerspruch, L_1 kann somit nicht regulär sein.

e)* Wählen Sie eine Sprache $L \in \{L_1, L_2, L_3, L_4\}$, die nicht regulär ist, und die Sie in Teilaufgabe d) nicht gewählt haben. Zeigen Sie, dass L unendlich viele Residualsprachen hat und somit nicht regulär ist. Geben Sie insbesondere Wörter $w_1, w_2, ... \in \Sigma^*$ an, sodass $L^{w_i} \neq L^{w_j}$ gilt, für alle $i, j \in \mathbb{N}$ mit $i \neq j$.

0 1 2 3 4

Wir wählen $L := L_2$. (Ähnliche Beweise funktionieren für alle vier Sprachen.) Sei $w_i := a^i ba$ für alle $i \in \mathbb{N}$. Dann ist $L^{w_i} = \{b^i\}$, diese sind offensichtlich paarweise verschieden.

Aufgabe 7 **Berechenbarkeit und Komplexität (10 Punkte)**

Für die folgenden Fragen ist eine Begründung nicht gefordert. Sie erhalten die Punkte auf eine Teilaufgabe genau dann, wenn Sie alle Antwortmöglichkeiten korrekt angekreuzt haben. Es ist immer mindestens eine Antwortmöglichkeit richtig. Jede Teilaufgabe bringt 2 Punkte.

Kreuzen Sie richtige Antworten an ☒
Kreuze können durch vollständiges Ausfüllen gestrichen werden ■
Gestrichene Antworten können durch nebenstehende Markierung erneut angekreuzt werden ✕■

In dieser Aufgabe verwenden wir durchgehend das Alphabet $\Sigma := \{0, 1\}$.

a) Welche der folgenden Sprachen sind entscheidbar?

- ☐ $\{w \in \Sigma^* : L(M_w) = \{vv : v \in \Sigma^*\}\}$ Satz von Rice.
- ☒ $\{ww : w \in \Sigma^*\}$

b) Für welche der folgenden Sprachen gibt es ein $v \in \Sigma^*$, sodass sie entscheidbar sind?
Beachten Sie, dass $\varphi_u : \Sigma^* \to \Sigma^*$ die Funktion ist, die von M_u berechnet wird.

- ☒ $\{w \in \Sigma^* : \varphi_v(v) = 0\}$ Die Bedingung hängt nicht von w ab.
- ☒ $\{w \in \Sigma^* : \varphi_v(w) = 0\}$ Wähle z.B. v als Kodierung einer TM, die nie hält.
- ☐ $\{w \in \Sigma^* : \varphi_w(v) = 0\}$ Satz von Rice.

c) Sei $L \subseteq \Sigma^*$ endlich. Welche der folgenden Sprachen sind semi-entscheidbar?

- ☒ $\{w \in \Sigma^* : L(M_w) = \Sigma^*\} \cap L$ Die resultierende Menge ist endlich.
- ☒ $\{w \in \Sigma^* : L(M_w) \cap L = L\}$ Wir können $M_w[x]\downarrow$ für jedes $x \in L$ überprüfen.
- ☒ $\{w \in \Sigma^* : L(M_w) \cap L = \Sigma^*\}$ Die Bedingung ist trivialerweise falsch.

d) Gibt es eine reguläre Sprache $L \in$ NP?

- ☒ Ja Tatsächlich sind alle regulären Sprachen in NP.
- ☐ Nein
- ☐ Unbekannt (hängt von P $\stackrel{?}{=}$ NP ab)

e) Bei dem NP-vollständigen Problem SAT geht es darum, von einer aussagenlogische Formel F zu überprüfen, ob sie erfüllbar ist. Die Formel F besteht dabei aus Variablen $x_1, ..., x_k$, die beliebig mit $\wedge, \vee, \neg$ (also logischer Konjunktion, Disjunktion und Negierung) verknüpft werden. Ein Beispiel für eine solche Formel ist $(x_1 \wedge x_2) \vee \neg(x_2 \vee \neg x_3)$.
Welche der folgenden Varianten von SAT sind NP-vollständig, unter der Annahme P $\neq$ NP?

- ☐ Die Formel F enhält $\neg$ nicht. Die Formel ist immer erfüllt, wenn alle Variablen wahr sind.
- ☒ Die Formel F enthält $\wedge$ nicht. Man kann $x \wedge y$ auch als $\neg(\neg x \vee \neg y)$ schreiben.
- ☐ Jede Variable x_i kommt nur einmal in F vor. Solange möglich, ersetzen wir $\neg(x \wedge y)$ durch $\neg x \vee \neg y$, $\neg(x \vee y)$ durch $\neg x \wedge \neg y$, und $\neg\neg x$ durch x. Anschließend wird jede Variable immer noch nur einmal verwendet, und Negationen können nur unmittelbar vor Variablen auftreten. Wenn $\neg x$ enthalten ist, setzen wir x auf falsch, sonst auf wahr. Dies ist immer eine erfüllende Belegung.

Aufgabe 8 **Reduktion (12 Punkte)**

Sei $\Sigma := \{a, b\}$. Für ein Wort $w \in \Sigma^*$ bezeichnen wir mit w^S das Wort, das man erhält, wenn man in w jedes a durch ein b, und jedes b durch ein a ersetzt. Z.B. gilt also $\varepsilon^S = \varepsilon$, $(baa)^S = abb$ und $(abbab)^S = baaba$. Für eine Sprache $L \subseteq \Sigma^*$ definieren wir dann $L^S := \{w^S : w \in L\}$.

a)* Sei H die Grammatik über dem Alphabet Σ mit den Produktionen $S \to aSbb \mid SS \mid ba$. Geben Sie eine kontextfreie Grammatik (CFG) H' an, sodass $L(H') = L(H)^S$ und H' höchstens 3 Produktionen hat. Eine Begründung ist nicht erforderlich. ☐ 0 ☐ 1

$S \to bSaa \mid SS \mid ab$

Wir betrachten nun die beiden folgenden Probleme, jeweils für eine CFG G über Σ.

$\langle 1 \rangle$ Ist $L(G) = \Sigma^*$? $\langle 2 \rangle$ Ist $L(G) = L(G)^S$?

b)* Ist $\overline{\langle 2 \rangle}$ semi-entscheidbar (also $\langle 2 \rangle$ co-semi-entscheidbar)? Begründen Sie Ihre Antwort. ☐ 0 ☐ 1 ☐ 2

☒ ja ☐ nein

Begründung: Man kann G zu einer Grammatik G' mit $L(G') = L(G)^S$ konvertieren, indem man alle a durch b ersetzt und umgekehrt. Danach semi-entscheiden wir $\overline{\langle 2 \rangle}$, indem wir alle Wörter $w \in \Sigma^*$ durchgehen, und $w \in L(G)$ und $w \in L(G')$ entscheiden (das Wortproblem für CFG ist entscheidbar). Falls $L(G) \neq L(G)^S$, gibt es ein Wort, sodass genau eine dieser Aussagen stimmt, und wir terminieren.

Wir wissen bereits, dass $\langle 1 \rangle$ unentscheidbar ist. Unser Ziel ist es, zu zeigen, dass auch $\langle 2 \rangle$ unentscheidbar ist, indem wir $\langle 1 \rangle$ auf $\langle 2 \rangle$ reduzieren, also $\langle 1 \rangle \leq \langle 2 \rangle$ zeigen.

c)* Geben Sie eine Reduktionsfunktion für $\langle 1 \rangle \leq \langle 2 \rangle$ an. Beschreiben Sie also ein Verfahren, das eine CFG $G = (V, \Sigma, P, S)$ zu einer CFG $G' = (V', \Sigma, P', S')$ konvertiert, sodass $L(G) = \Sigma^*$ gilt, genau dann wenn $L(G') = L(G')^S$ gilt. ☐ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Wir wählen G' so, dass $L(G') = \{a\}L(G) \cup \{b\}\Sigma^*$ gilt. Genauer wählen wir Produktionen $S' \to aS \mid bR$ und $R \to aR \mid bR \mid \varepsilon$, sowie die Produktionen aus P.

Führen Sie Ihr Verfahren zur Veranschaulichung auf der Grammatik G mit Produktionen $S \to aSbS \mid ba$ aus, und geben Sie die resultierende Grammatik G' an.

$S' \to aS \mid bR$
$R \to aR \mid bR \mid \varepsilon$
$S \to aSbS \mid ba$

0 1 2 3 4

d) Beweisen Sie, dass ihr Verfahren aus Teilaufgabe c) korrekt ist, dass also $L(G) = \Sigma^* \Leftrightarrow L(G') = L(G')^S$ gilt.

„$\Rightarrow$“: Wenn $L(G) = \Sigma^*$ gilt, dann folgt $L(G') = \{a\}L(G) \cup \{b\}\Sigma^* = \Sigma^+ = (\Sigma^+)^S$.

„$\Leftarrow$“: Wir zeigen die Implikation über Kontraposition. Wir nehmen also $L(G) \neq \Sigma^*$ an, somit existiert ein $w \in \Sigma^*$ mit $w \notin L(G)$. Es folgt $aw \notin \{a\}L(G)$, und offensichtlich gilt $aw \notin \{b\}\Sigma^*$. Insgesamt erhalten wir $aw \notin L(G')$.
Weiterhin gilt $L(G')^S = \{b\}L(G)^S \cup \{a\}\Sigma^*$, und somit $aw \in \{a\}\Sigma^* \subseteq L(G')^S$. Da das Wort aw in genau einer der Sprachen $L(G'), L(G')^S$ enthalten ist, folgt $L(G') \neq L(G')^S$.

Alternative Lösung: $L(G') = L(G')^S \Rightarrow \{a\}L(G) \cup \{b\}\Sigma^* = \{b\}L(G)^S \cup \{a\}\Sigma^* \Rightarrow L(G) = \Sigma^*$.

Aufgabe 9 **Sequenz (8 Punkte)**

Sei $\Sigma := \{0,1\}$ und $L \subseteq \Sigma^*$ eine Sprache. Wir definieren eine Folge an Wörtern $w_0, w_1, w_2, ... \in \Sigma^*$ wie folgt:

$$w_0 := \varepsilon \qquad w_{n+1} := \begin{cases} w_n 1 & \text{wenn } w_n \in L \\ w_n 0 & \text{wenn } w_n \notin L \end{cases} \qquad \text{für } n \in \mathbb{N}$$

Wir schreiben $\sigma(L) := \{w_i : i \in \mathbb{N}\}$ für die Sprache, die genau die Wörter der Folge enthält.

a)* Geben Sie einen regulären Ausdruck für die Sprache $\sigma(\Sigma^* 0)$ an. (Ohne Begründung.)

0
1
2

$(01)^*(0 \mid \epsilon)$

b)* Zeigen Sie: Wenn L regulär ist, dann ist auch $\sigma(L)$ regulär.
Hinweis: Für eine Aussage A können Sie die Notation $I(A)$ verwenden; es sei $I(A) := 1$, falls A gilt, und $I(A) := 0$ sonst. Obige Definition lässt sich z.B. als $w_{n+1} = w_n I(w_n \in L)$ schreiben.

0
1
2
3
4
5
6

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA für L. Wir modifizieren M, indem wir alle 0-Transitionen aus akzeptierenden Zuständen löschen, ebenso wie alle 1-Transitionen aus nicht-akzeptierenden Zuständen. Weiterhin machen wir alle Zustände final. Wir erhalten also einen DFA $M' = (Q \cup \{q_\emptyset\}, \Sigma, \delta', q_0, Q)$ mit folgenden Transitionen δ':

$$\delta'(q, I(q \in F)) := \delta(q, I(q \in F)) \qquad \text{für } q \in Q$$

Alle anderen Transitionen führen zum Fangzustand $q_\emptyset$. Insbesondere beobachten wir $L(M') = \{w \in \Sigma^* : \hat{\delta}'(q_0, w) = \hat{\delta}(q_0, w)\}$, da jeder Zustand in Q in M' akzeptierend ist und M' nur Transitionen aus M enthält.
Sei $w_0, w_1, ...$ die Folge aus der Definition von $\sigma(L(M))$, mit $\sigma(L(M)) = \{w_i : i \in \mathbb{N}\}$. Wir zeigen nun $L(M') = \sigma(L(M))$. Hierzu genügt es, $L(M') \cap \Sigma^n = \{w_n\}$ für alle $n \in \mathbb{N}$ zu zeigen. Dies machen wir per Induktion über n.
Für die Basis müssen wir $w_0 = \varepsilon \in L(M')$ zeigen, was aus $q_0 \in Q = F$ folgt. Für den Schritt sei n beliebig, und wir nehmen $L(M') \cap \Sigma^n = \{w_n\}$ an. Sei $w \in \Sigma^n, c \in \Sigma$, und $q := \delta(q_0, w)$. Es gilt

$$\begin{aligned} wc \in L(M') &\Leftrightarrow \hat{\delta}'(q_0, wc) = \hat{\delta}(q_0, wc) \\ &\Leftrightarrow \hat{\delta}'(q_0, w) = q \wedge \delta'(q, c) = \delta(q, c) \\ &\Leftrightarrow w \in L(M') \wedge c = I(q \in F) \\ &\Leftrightarrow w = w_n \wedge c = I(w \in L(M)) \\ &\Leftrightarrow wc = w_{n+1} \end{aligned}$$

Alternative Lösung: Sei $q_n := \hat{\delta}(q_0, w_n)$ (man beachte $w_0 = \varepsilon$), und $x_n := I(w_n \in L(M))$, für alle $n \in \mathbb{N}$. Dann gilt $w_{n+1} = w_n x_n$, sowie $x_n = I(q_n \in F)$. Es folgt $q_{n+1} = \hat{\delta}(q_0, w_{n+1}) = \delta(q_n, x_n) = \delta(q_n, I(q_n \in F))$. Insbesondere hängt q_{n+1} nur von q_n ab.
Da $q_n \in Q$ und Q endlich ist, gibt es $i < j$ mit $q_i = q_j$. Also gilt

$$L = \{w \in L : |w| < i\} \cup L(w_{i-1}(x_i ... x_{j-1})^*)\{x_i ... x_k : i \leq k < j\}$$

Zusätzlicher Platz für Lösungen. Markieren Sie deutlich die Zuordnung zur jeweiligen Teilaufgabe. Vergessen Sie nicht, ungültige Lösungen zu streichen.