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:**	Freitag, 6. August 2021
Prüfer:	Prof. Dr. Dr. h.c. Javier Esparza	**Uhrzeit:**	17:00 – 20:00

Bearbeitungshinweise

- Diese Klausur umfasst **18 Seiten** mit insgesamt **9 Aufgaben**.
  Bitte kontrollieren Sie jetzt, dass Sie eine vollständige Angabe erhalten haben.
- Die Gesamtpunktzahl in dieser Prüfung beträgt 100 Punkte.
- 45 Punkte sind *hinreichend* zum Bestehen.
- Sie müssen Ihre Klausur eingescannt bis 20:15:00 online auf TUMExam einreichen:
  - Sie müssen nur die von Ihnen bearbeiteten Seiten und das **unterschriebene** Deckblatt hochladen.
  - Achten Sie darauf, dass sowohl Ihre **Lösungen als auch die Barcodes klar lesbar** sind.
  - In begründeten Fällen (wie z.B. bei technischen Problemen) können Sie der Übungsleitung bis 20:15:00 per Email (theoleitung@in.tum.de) ihre Klausur zukommen lassen, als PDF oder SHA256-Prüfsumme.
- Sie müssen die Klausur alleine bearbeiten. Die Klausur ist open-book (Kofferklausur), allerdings dürfen Sie in keinster Weise Unterstützung von anderen Personen **erhalten** oder diesen **geben** (in Person, Chat, Foren, Diskussiongruppen, etc.). Eine solche Unterstützung wird als Unterschleif bewertet und mit den Konsequenzen, wie in der APSO beschrieben, geahndet.
- Sie dürfen jegliche Art von Literatur (auch im Internet) benutzen. Sollten Sie dabei auf Lösungsansätze stoßen, die Sie für die Klausur verwenden möchten, so müssen Sie diese Teile klar und deutlich zitieren (Literaturverweis bzw. Link). Die Lösung selber müssen Sie dennoch weiterhin selbstständig in die Klausur übertragen. Ihnen ensteht durch eine Zitation kein Nachteil.
- Sie können uns Fragen zu technischen Problemen via Zulip oder E-Mail (theoleitung@in.tum.de) stellen. Inhaltliche Fragen werden wir nicht beantworten; falls Ihnen eine Aufgabenstellung mehrdeutig erscheint, notieren Sie bitte Ihre Interpretation der Aufgabe.
- Mit * gekennzeichnete Teilaufgaben sind ohne Kenntnis der Lösung vorheriger Teilaufgaben lösbar.
- Sie dürfen Ergebnisse vorheriger Teilaufgaben auch dann verwenden, wenn Sie diese nicht lösen konnten.
- **Es werden nur solche Ergebnisse gewertet, bei denen der Lösungsweg erkennbar ist**. Alle Aufgaben sind **grundsätzlich zu begründen**, sofern es nicht ausdrücklich anders vermerkt ist.
- Schreiben Sie weder mit roter noch grüner Farbe.
- Ihre Lösungen müssen **handschriftlich** verfasst sein (digital oder auf Papier)!
- $0 \in \mathbb{N}$.

Aufgabe 1 Reguläre und kontextfreie Sprachen (12 Punkte)

Für die folgenden Fragen ist eine Begründung nur erforderlich, wenn explizit danach gefragt wird.

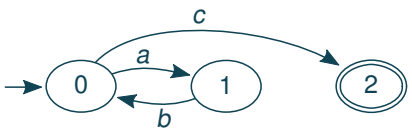
0 1 2

a)* Geben Sie Sprachen $A, B, C \subseteq \{a, b, c\}^*$ mit $B \neq C$ und $AB = CA$ an.

$A := \Sigma^*, B := \{\varepsilon, b\}, C := \{\varepsilon, c\}$
Alternativ: $A := \{a\}, B := \{ab\}, C := \{ba\}$

0 1 2

b)* Geben Sie einen NFA M mit höchstens drei Zuständen an, der eine Sprache L mit $L = ab\,L \cup \{c\}$ akzeptiert.



0 1 2

c)* Geben Sie eine kontextfreie Grammatik für eine Sprache L mit $L = \{a\}L^*\{b\} \cup \{bb\}$ an, mit höchstens 4 Produktionen. Beachten Sie, dass z.B. $S \to SS \mid \varepsilon$ zwei Produktionen sind.

$S \to aTb \mid bb \qquad T \to ST \mid \varepsilon$

0 1 2 3

d)* Welche Sprachen können von kontextfreie Grammatiken über einem Alphabet Σ erzeugt werden, bei denen die rechte Seite jeder Produktion Länge 1 hat? Beachten Sie $|\varepsilon| = 0$.

Genau die Sprachen in $\mathcal{P}(\Sigma) = \{S : S \subseteq \Sigma\}$.

0 1 2 3

e)* Seien $M_1 = (Q, \Sigma, \delta, q_0, F_1)$, $M_2 = (Q, \Sigma, \delta, q_0, F_2)$ zwei beliebige DFAs, die sich nur in ihren Finalzuständen unterscheiden, mit $F_1 \neq F_2$ und $L(M_1) = L(M_2)$. Gibt es solche M_1, M_2, in denen alle Zustände erreichbar sind? Wenn ja, geben Sie ein Beispiel an, wenn nein, begründen Sie dies kurz.

Nein, alle Zustände in $(F_1 \cup F_2) \setminus (F_1 \cap F_2)$ sind nicht erreichbar, denn falls $q = \delta(q_0, w)$ für einen solchen Zustand q und ein Wort w, wäre w nur in genau einer der Sprachen enthalten und diese damit unterschiedlich.

Aufgabe 2 CYK-Algorithmus (10 Punkte)

Die Grammatik G ist eine kontextfreie Grammatik in CNF mit 2 Terminalen (f, b) und 2 Nichtterminalen (F, B) **wobei F das Startsymbol ist**.

In der folgenden CYK-Tabelle für das Wort *fbbff* und die Grammatik G sind die unteren beiden Zeilen bereits ausgefüllt.

1,5 F				
1,4 B	2,5 B			
1,3 F	2,4 F	3,5 B		
1,2 *F*	2,3 –	3,4 *F*	4,5 *B*	
1,1 *F*	2,2 *B*	3,3 *B*	4,4 *F*	5,5 *F*
f	*b*	*b*	*f*	*f*

a)* Geben Sie die Grammatik G an.

0 1 2 3

$F \to FB \mid BF \mid f$
$B \to FF \mid b$

b) Vervollständigen Sie die CYK-Tabelle am Anfang der Aufgabe. Sie können entweder die gesamte Tabelle in dem nachfolgenden Antwortbereich angeben oder direkt in die Tabelle oben schreiben.

0 1 2 3

1,5 F				
1,4 B	2,5 B			
1,3 F	2,4 F	3,5 B		
1,2 *F*	2,3 –	3,4 *F*	4,5 *B*	
1,1 *F*	2,2 *B*	3,3 *B*	4,4 *F*	5,5 *F*
f	*b*	*b*	*f*	*f*

c)* Gilt $fbbff \in L(G)$? Begründen Sie Ihre Antwort *kurz* mit Hilfe der ausgefüllten Tabelle aus Teilaufgabe b). Falls Sie Teilaufgabe b) nicht bearbeitet haben, dürfen Sie annehmen, dass in jeder Zelle in den oberen 3 Zeilen der Tabelle exakt das Nichtterminal F steht.

0 1

Ja, weil F in Zelle 1,5 steht.

0 1 d)* Gilt *bbf* $\in L(G)$? Begründen Sie Ihre Antwort *kurz* mit Hilfe der ausgefüllten Tabelle aus Teilaufgabe b). Falls Sie Teilaufgabe b) nicht bearbeitet haben, dürfen Sie annehmen, dass in jeder Zelle in den oberen 3 Zeilen der Tabelle exakt das Nichtterminal *F* steht.

Ja, weil *F* in Zelle 2,4 steht.

0 1 2 e)* Aus einer ausgefüllten CYK-Tabelle kann man den Syntaxbaum für ein Wort gewinnen. Wie würden Sie den CYK-Algorithmus abändern, um den Syntaxbaum für ein Wort effizienter ablesen zu können?

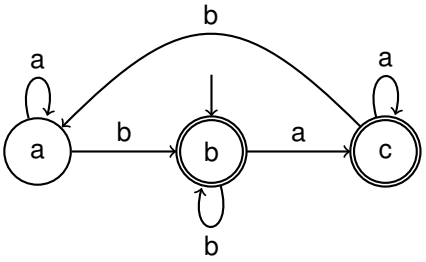
Das Problem ist, dass man für die Nichtterminale in einer Zelle nicht weiß, aus welcher Kombination aus Zellen sie entstanden sind. Das heißt, man muss, um den Syntaxbaum zu gewinnen, alle möglichen Kombinationen überprüfen.
Statt nur die Nichttermine in der Tabelle zu speichern, kann man sich zusätzlich auch speichern, wegen welcher Kombination aus Zellen und Produktion das Nichtterminal erzeugt wurde. Dann kann man direkt den Syntaxbaum aus der Tabelle ablesen.

Aufgabe 3 **Produktkonstruktion (8 Punkte)**

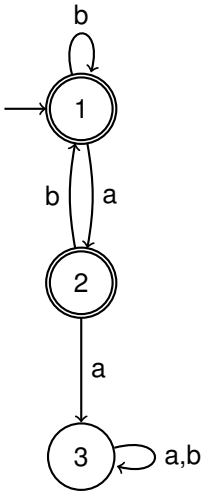
a)* Berechnen Sie mit Hilfe der Produktkonstruktion einen DFA für $\{w \in \{a,b\}^* \mid w \in L(D_1) \wedge w \in L(D_2)\}$. Zeichnen Sie direkt in die Aufgabenstellung. Wenn Sie zusätzlichen Platz benötigen, finden Sie leere Seiten am Ende der Klausur.

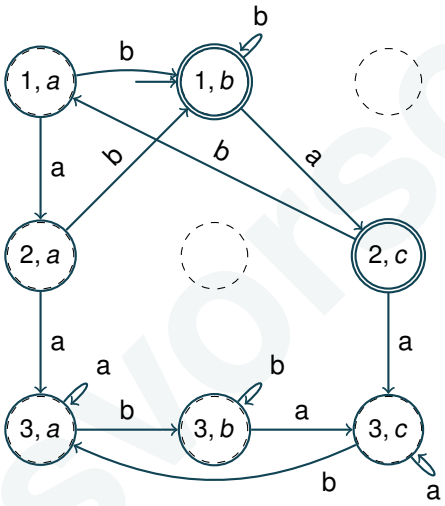
0 1 2 3 4 5 6

DFA D_2



DFA D_1





b)* Beschreiben Sie, wie man die Produktkonstruktion abändern kann, um aus 2 DFAs $D_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ und $D_2 = (Q_2, \Sigma, \delta_2, q_0', F_2)$ einen DFA zu gewinnen, der die Sprache $\{w \in \Sigma^* \mid w \notin L(D_1) \vee w \in L(D_2)\}$ akzeptiert.

0 1 2

Die akzeptierenden Zustände müssen geändert werden, sodass $F = \{(q_1, q_2) \in Q_1 \times Q_2 \mid q_1 \notin F_1 \vee q_2 \in F_2\}$.

Aufgabe 4 **Kontextfreie Sprachen (10 Punkte)**

0 1 2 3 4 5 6 7 8 9 10

Ist L kontextfrei? Wenn ja, geben Sie eine kontextfreie Grammatik an, wenn nein, widerlegen Sie es.

a)* $L := M \cap M^R$, wobei $M := \{b^i a^j b^k : 0 \leq i < j\}$.

Es gilt $L = \{b^i a^j b^k : i < j \wedge j > k\}$, die Sprache ist nicht kontextfrei. Beweis über Pumping Lemma für kontextfreie Sprachen:
Wir nehmen an, L wäre kontextfrei. Dann gilt das PL für kontextfreie Sprachen.
Sei $n \in \mathbb{N}$ beliebig. Wir wählen das Wort $z := b^n a^{n+1} b^n$. Sei $uvwxy = z$ eine beliebige Zerlegung, die die Eigenschaften des Pumping-Lemmas erfüllt. Wenn v oder x sowohl a als auch b beinhalten würde, wäre $uv^2wx^2y \notin L$, da dies die Struktur des Wortes verändert.
Fall 1, $a \in vx$: Hier pumpen wir ab, und das resultierende Wort uv^0wx^0y hat Form $b^i a^j b^n$ oder $b^n a^i b^j$, mit $i, j \leq n$. Somit gilt $uv^0wx^0y \notin L$.
Fall 2, $a \notin vx$: Das Wort uv^2wx^2y hat Form $b^i a^{n+1} b^j$ mit $i > n$ oder $j > n$, und es gilt $uv^2wx^2z \notin L$.
In beiden Fällen haben wir gezeigt, dass die Zerlegung die erforderlichen Eigenschaften nicht erfüllt. Dies ist ein Widerspruch, und somit kann die Sprache nicht kontextfrei sein.

b)* $L := \{b^i a^j b^k : 0 \leq i < j \vee 0 \leq k < j\}$

Die Sprache ist kontextfrei.

$$S \to TA \mid AR$$
$$T \to bTa \mid Ta \mid a$$
$$R \to aRb \mid aR \mid a$$
$$A \to bA \mid \varepsilon$$

Aufgabe 5 PDA Konstruktion (14 Punkte)

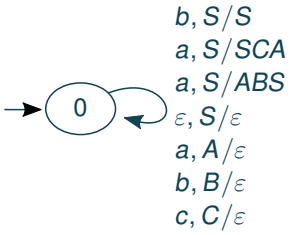
Sei G die Grammatik mit Produktionen

$$S \to bS \mid aSca \mid aabS \mid \varepsilon$$

a)* Konstruieren Sie einen zu G äquivalenten Kellerautomaten mit einem Zustand. Verwenden Sie dazu eine aus der Vorlesung oder den Übungen bekannte Notation. Ihr PDA soll über leeren Keller akzeptieren.

0 1 2 3 4 5 6

Wir konvertieren G zunächst zu $S \to bS \mid aSCA \mid aABS \mid \varepsilon,\ A \to a,\ B \to b,\ C \to c$.
Der PDA verwendet S als initiales Kellersymbol.



b)* Finden Sie das lexikographisch erste Wort $w \in L(G)$ mit Länge 58. Begründen Sie Ihre Antwort kurz. Sie können w^k als Abkürzung für die k-fache Konkatenation verwenden, für ein Wort w und $k \in \mathbb{N}$.
Zur Erinnerung: $a <_{\text{lex}} b <_{\text{lex}} c$, und für zwei unterschiedliche Wörter u, v mit gleicher Länge gilt $u <_{\text{lex}} v$ genau dann, wenn $u_i <_{\text{lex}} v_i$ an der ersten Stelle i, an der sie sich unterscheiden. Z.B. gilt also $abac <_{\text{lex}} abba$.

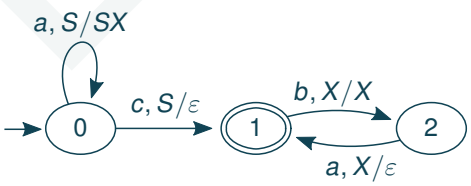
0 1 2

Wir wählen Produktion $S \to aSca$ so oft wie möglich, und danach führen wir einmal $S \to aabS$, einmal $S \to bS$, und einmal $S \to \varepsilon$ aus, in dieser Reihenfolge. Letztere ergeben das Wort $aabb$, wir erhalten also $a^{18}aabb(ca)^{18}$.

c)* Konstruieren Sie einen **deterministischen** Kellerautomaten M, mit $L(M) = L(G')$, wobei G' die Grammatik mit den Produktionen $S \to aSba \mid aS \mid c$ ist. Ihr Automat soll über Finalzustände akzeptieren und höchstens 5 Zustände und 4 Kellersymbole haben.

0 1 2 3 4 5 6

Der PDA verwendet S als initiales Kellersymbol.



Aufgabe 6 Entscheidbarkeit und Komplexität (12 Punkte)

Entscheiden Sie, ob die folgenden Aussagen jeweils wahr sind, **unter der Annahme P $\neq$ NP**. Falls ja, geben Sie eine *kurze* Begründung an, falls nein, ein Gegenbeispiel.
Achtung: Wenn die Aussage falsch ist, müssen Sie ein konkretes Gegenbeispiel angeben, eine Begründung genügt nicht.

0 1 2

a)* Wenn $L \subseteq \{a, b\}^*$ entscheidbar ist, dann ist L^a entscheidbar. Hier bezeichnet $L^a := \{w : aw \in L\}$ die Residualsprache bezüglich a von L.

Wahr, um $w \in L^a$ für ein Wort w zu entscheiden, müssen wir nur überprüfen, ob aw in L enthalten ist.

0 1 2

b)* Seien M_1, M_2 beliebige Turingmaschinen, und L eine Sprache mit $L(M_1) \subseteq L \subseteq L(M_2)$. Dann ist L semi-entscheidbar.

Falsch, sei z.B. M_1 eine TM, die nie akzeptiert, M_2 eine TM, die immer akzeptiert, und L eine beliebige nicht semi-entscheidbare Sprache, beispielsweise $L := \mathcal{H}_{\Sigma^*}$. Dann gilt $L(M_1) = \emptyset$ und $L(M_2) = \Sigma^*$, aber natürlich $\emptyset \subseteq L \subseteq \Sigma^*$.

0 1 2

c)* Es gibt keine reguläre Sprache $L \in$ NP.

Falsch, z.B. $\emptyset \in$ P $\subseteq$ NP.

0 1 2 3

d)* Sei L entscheidbar und M eine NTM. Dann ist $L(M) \setminus L$ rekursiv aufzählbar.

Wahr. Rekursiv aufzählbar ist äquivalent zu semi-entscheidbar. Für eine Eingabe w überprüfen wir, ob $w \in L$; falls ja, gehen wir in eine Endlosschleife, falls nein, lassen wir M laufen.

0 1 2 3

e)* Für NP-vollständige Sprachen $L_1, L_2 \subseteq \{0, 1\}^*$ ist $\{w_1\$w_2 : w_1 \in L_1, w_2 \in L_2\}$ stets NP-hart/NP-schwierig.

Wahr. Wir reduzieren $L_1 \leq_p L_1\{\$\}L_2$, und fixieren dazu ein beliebiges $x \in L_2$. ($L_2 \neq \emptyset$, da die leere Menge nicht NP-vollständig ist.) Unsere Reduktionsfunktion ist $f(w) := w\$x$.

Aufgabe 7 Reduktion (12 Punkte)

In dieser Aufgabe betrachten wir reguläre Ausdrücke über einem Alphabet $\Sigma = \{0, 1\}$. Die Länge $|r|$ eines regulären Ausdrucks r ist die Länge der üblichen Repräsentation über dem Alphabet $\Sigma \cup \{(,), *, |, \epsilon, \emptyset\}$. Z.B. hat der Ausdruck $(0|1)^*$ Länge 6. Wir verwenden r^k als Makro für die k-fache Konkatenation von r. Beachten Sie, dass $|r^k| = k\,|r|$, so hat etwa der Ausdruck $(0|1)^{10}$ Länge 50, **nicht** Länge 7.

a)* Sei $V = \{x_1, x_2, \ldots, x_9\}$ eine Menge von booleschen Variablen. Wir identifizieren eine Belegung von V mit einem Wort $w \in \{0,1\}^9$ der Länge 9. Hierfür legen wir also die Reihenfolge $x_1, \ldots, x_9$ fest, sodass w_i mit der Belegung von x_i korrespondiert. Sei F die aussagenlogische Formel

0 1 2

$$F = (x_2 \vee x_5 \vee \neg x_7) \wedge (x_3 \vee \neg x_4 \vee \neg x_7)$$

Geben Sie einen regulären Ausdruck r_F mit $|r_F| \leq 150$ an, sodass $L(r_F)$ genau die Belegungen enthält, die F **nicht** erfüllen. Sie können Σ als Makro für $(0 \mid 1)$ verwenden (das Makro hat Länge 5).

$r_F := \Sigma\ 0\ \Sigma^2\ 0\ \Sigma\ 1\ \Sigma^2 \mid \Sigma^2\ 01\ \Sigma^2\ 1\ \Sigma^2$
$|r_F| = 7 + 12 \cdot 5 = 67$

b)* Geben Sie einen regulären Ausdruck r_V mit $|r_V| \leq 150$ für die Sprache $\{w \in \{0,1\}^* : |w| \neq 9\}$ an.

0 1 2

$r_V := (\Sigma \mid \epsilon)^8 \mid \Sigma^{10}\Sigma^*$
$|r_V| = 2 + 8 \cdot 9 + 11 \cdot 5 = 129$

c)* Wir betrachten das Nicht-Universalitätsproblem für reguläre Ausdrücke, was wir als RE-NONUNI bezeichnen. Es ist folgendermaßen definiert:

0 1 2 3 4 5 6 7 8

Eingabe: Ein regulärer Ausdruck r über $\Sigma = \{0, 1\}$.
Ausgabe: Ist $L(r) \neq \Sigma^*$?

Zeigen Sie 3KNF-SAT $\leq_p$ RE-NONUNI, indem Sie eine entsprechende Reduktionsfunktion beschreiben und argumentieren, dass diese die notwendigen Bedingungen erfüllt.

Sei F eine beliebige Instanz von 3KNF-SAT mit k Variablen, also eine Konjunktion von Klauseln $F_1 \wedge \ldots \wedge F_n$. Wir erzeugen einen regulären Ausdruck r_F, der genau die Belegungen erzeugt, die F nicht erfüllen. Den Ausdruck r_F erzeugen wir wie in Teilaufgabe a), er hat also die Form $r_F = r_1 \mid \ldots \mid r_n$, wobei r_i ein regulärer Ausdruck ist, der die Belegungen erzeugt, die Klausel F_i nicht erfüllen. r_i hat also die Form $r_i = t_1 \ldots t_k$ mit

$$t_i := \begin{cases} \Sigma & \text{falls } x_i, \neg x_i \notin F_i \\ 0 & \text{falls } x_i \in F_i \\ 1 & \text{falls } \neg x_i \notin F_i \end{cases}$$

Zusätzlich erzeugen wir einen regulären Ausdruck r_V für alle Binärwörter, die nicht die richtige Länge haben, wie in Teilaufgabe b). Es gilt also $r_V := (\Sigma \mid \epsilon)^{k-1} \mid \Sigma^{k+1}\Sigma^*$.
Die Reduktionsfunktion liefert nun den regulären Ausdruck $r := r_F \mid r_V$, dessen Länge polynomiell in k ist: $|r_i| \leq |\Sigma^k| = 5k$, somit folgt $|r_F| \leq (5k+1)n \leq 6kn$, und mit $|r_V| \leq 18k + 7$ erhalten wir $|r| \leq 6kn + 18k + 7$.
Wenn F erfüllbar ist, gibt es eine erfüllende Belegung $\sigma : \{x_1, \ldots, x_k\} \to \{0,1\}$. Sei $w \in \Sigma^k$ das entsprechende Wort (also $w_i := \sigma(x_i)$). Dann gilt offensichtlich $w \notin L(r_V)$. Da r genau die Belegungen erzeugt, die F nicht erfüllen, gilt $w \notin L(r) \Rightarrow L(r) \neq \Sigma^*$.
Falls F nicht erfüllbar ist, sei $w \in \Sigma^*$ beliebig. Für $|w| \neq k$ gilt $w \in L(r_V)$, ansonsten ist w eine gültige Belegung. Diese kann nicht erfüllend sein, und muss somit eine Klausel F_i verletzen. Also $w \in L(r_i) \subseteq L(r_F)$. Da w beliebig ist, folgt $L(r) = \Sigma^*$.

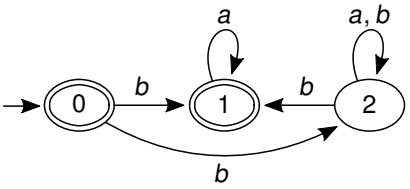
Lösungbox Aufgabe 7 (fortlaufend)

Aufgabe 8 Fast Azyklisch (10 Punkte)

Ein NFA ist *fast azyklisch*, wenn alle Zyklen Schlingen sind. Formal: Ein NFA $A = (Q, \Sigma, \delta, q_0, F)$ ist fast azyklisch wenn es keine Zustände q, q' mit $q \neq q'$ gibt, sodass man von q aus q' erreichen kann, und von q' aus q erreichen kann.
Ein regulärer Ausdruck r ist *fast azyklisch*, wenn er in der Form $r = t_1 \mid t_2 \mid ... \mid t_k$ ist, wobei $t_1, ..., t_k$ reguläre Ausdrücke sind, die das Symbol $\mid$ nicht beinhalten. Beispielsweise ist $a^*b(ab^*a)^* \mid ba^*b \mid \epsilon$ fast azyklisch, und $a^*b(a \mid b)^*$ nicht.

a)* Geben Sie einen einen fast azyklischen regulären Ausdruck r an, sodass $L(r)$ die von diesem NFA akzeptierten Sprache ist:

0 1 2 3 4



$\epsilon \mid b(b^*a^*)^*ba^* \mid ba^*$

b)* Sei M ein beliebiger, fast azyklischer NFA. Konstruieren Sie einen fast azyklischen regulären Ausdruck r mit $L(r) = L(M)$ und begründen Sie, wieso Ihre Konstruktion korrekt ist.

0 1 2 3 4 5 6

Sei $M = (Q, \Sigma, \delta, q_0, F)$. Wir betrachten den azyklischen NFA M', der aus M entsteht, indem wir alle Schlingen entfernen. Also $M' = (Q, \Sigma, \delta', q_0, F)$ mit $\delta'(q, a) := \delta(q, a) \setminus \{q\}$.
Da M' azyklisch ist, gibt es nur endlich viele akzeptierende Pfade.
Für jeden solchen Pfad $\pi = (s_1, c_1, ..., s_{n-1}, c_{n-1}, s_n)$ erzeugen wir einen regulären Ausdruck r_π, der die Wörter erzeugt, die M über diesen Pfad akzeptieren kann. Hierbei sind $s_1, ..., s_n \in Q$ die Zustände, die der Pfad durchläuft, $c_i \in \Sigma$ das Zeichen der Transition von s_i zu s_{i+1} ist. (Man beachte $s_1 = q_0$ und $s_n \in F$.) Es gilt also

$$s_1 \xrightarrow{c_1} s_2 \xrightarrow{c_2} ... \xrightarrow{c_{n-1}} s_n$$

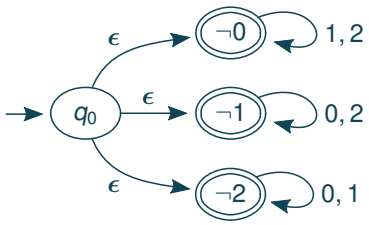
Wir wählen r_π als $r_\pi = t_0 c_0 t_1 c_1 ... c_{n-1} t_n$, wobei t_i genau die Schlingen von q_i erzeugt. Wenn q_i also Schlingen mit Zeichen $a_1, ..., a_k \in \Sigma$ hat, setzen wir $t_i := (a_1^* ... a_k^*)^*$, da $(a_1^* ... a_k^*)^* \equiv (a_1 | ... | a_k)^*$. Offensichtlich enthält r_π kein $\mid$.
Es gibt nur endlich viele akzeptierenden Pfade $\pi_1, ..., \pi_l$, und wir setzten nun $r = r_{\pi_1} \mid ... \mid r_{\pi_k}$.
Wie bereits argumentiert wurde, enthält r_{π_i} kein $\mid$, der Ausdruck r ist also fast azyklisch.
Schließlich müssen wir noch $L(M) = L(r)$ begründen. Ein Wort w wird von einem NFA genau dann akzeptiert, wenn es einen akzeptierenden Pfad für w gibt. Die Aussage folgt dann daraus, dass wir für jeden akzeptierenden Pfad einen regulären Ausdruck konstruiert haben, der genau die Wörter erzeugt, die auf diesem Pfad liegen.

Aufgabe 9 **Alle kommen vor (12 Punkte)**

Sei $\Sigma_n = \{1, 2, \ldots, n\}$. Sei $L_n \subseteq \Sigma_n^*$ die Menge der Wörter, in denen alle Buchstaben von Σ_n mindestens einmal vorkommen. Z.B. gilt $123, 2213, 333312 \in L_3$ und $11122, \varepsilon, 23, 33232323 \notin L_3$.

0 1 2

a)* Konstruieren Sie einen ϵ-NFA für $\overline{L_3}$ mit höchstens 4 Zuständen.



0 1 2 3 4 5

b)* Zeigen Sie: Jeder DFA für L_n hat mindestens 2^n Zustände.
Hinweis: Wenn Sie Teilaufgabe c) lösen, erhalten Sie die Punkte für b) automatisch.

Wir zeigen, dass es mindestens 2^n Äquivalenzklassen für die Relation $\equiv_{L_n}$ gibt, und somit der minimale DFA für L_n auch mindestens 2^n Zustände hat.
Für eine beliebige Teilmenge $M \subseteq \Sigma_n$ sei w_M das Wort, in dem jedes Zeichen in M genau einmal vorkommt, in aufsteigender Reihenfolge.
Wir behaupten nun $w_A \not\equiv_{L_n} w_B$ für alle $A, B \subseteq \Sigma_n$ mit $A \neq B$. Seien also solche A, B beliebig. Da $A \neq B$, gibt es ein $c \in \Sigma_n$ mit $c \in A \wedge c \notin B$ oder $c \notin A \wedge c \in B$. Wir nehmen (ohne Einschränkung der Allgemeingültigkeit) ersteres an, also $c \in A \wedge c \notin B$.
Sei nun $v := w_{\Sigma_n \setminus \{c\}}$ ein Wort, das alle Zeichen außer c enthält. Dann ist $w_A w \in L$, aber $w_B w \notin L$. Somit folgt $w_A \not\equiv_{L_n} w_B$.
Wir haben also für jedes $M \subseteq \Sigma_n$ eine unterschiedliche Äquivalenzklasse gefunden, und somit gibt es mindestens 2^n.

c)* Zeigen Sie: Jeder NFA für L_n hat mindestens 2^n Zustände.
Hinweis: Sie dürfen folgende Aussage ohne Beweis verwenden. Für zwei beliebige Wörter $u, v \in \Sigma_n^*$, in denen jedes Zeichen genau einmal vorkommt, und beliebige Zerlegungen $u = u_1u_2$, $v = v_1v_2$ mit $u_1 \neq v_1$, gilt $u_1v_2 \notin L_n$ oder $v_1u_2 \notin L_n$.

0 1 2 3 4 5

Für eine beliebige Teilmenge $M \subseteq \Sigma_n$ sei w_M das Wort, in dem jedes Zeichen in M genau einmal vorkommt, in aufsteigender Reihenfolge, und $v_M := w_{\Sigma_n \setminus M}$ ein Wort, in dem genau die übrigen Zeichen vorkommen. Insbesondere ist w_Mv_M ein Wort, in dem jedes Zeichen in Σ_n genau einmal vorkommt, für beliebige $M \subseteq \Sigma_n$.
Angenommen, es gäbe einen NFA $N = (Q, \Sigma_n, \delta, q_0, F)$ für L_n mit echt weniger als 2^n Zuständen.
Da N das Wort w_Mv_M für jedes $M \subseteq \Sigma_n$ akzeptiert, gibt es einen Zustand q_M, der von N nach Einlesen von w_M erreicht werden kann, und durch Einlesen von v_M einen Finalzustand erreicht. Formal gilt also $q_M \in \hat{\delta}(q_0, w_M)$ und $F \cap \hat{\delta}(q_M, v_M) \neq \emptyset$.
Da N weniger als 2^n Zustände hat, gibt es $A, B \subseteq \Sigma$ mit $A \neq B$ und $q_A = q_B$. Sei $q := q_A = q_B$. Der NFA kann also q sowohl mit w_A als auch mit w_B erreichen, und von q aus kann N einen Finalzustand sowohl über v_A als auch v_B erreichen.
Somit gilt $w_Av_B, w_Bv_A \in L(N)$. Aber $A \neq B$ impliziert $w_A \neq w_B$; wir können also den Hinweis anwenden und es folgt $w_Av_B \notin L \vee w_Bv_A \notin L$, ein Widerspruch zu $L(N) = L$. Also kann so ein NFA nicht existieren, und jeder NFA für L_n hat mindestens 2^n Zustände.

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