2. Klausur zur Vorlesung
Theoretische Grundlagen der Informatik
Wintersemester 2021/2022

Hier Aufkleber mit Name und Matrikelnummer anbringen

Vorname: _____

Nachname: _____

Matrikelnummer: _____

- Bringen Sie den Aufkleber mit Ihrem Namen und Ihrer Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes Aufgabenblatt mit Ihrem Namen und Matrikelnummer.
- Schreiben Sie die Lösungen auf die Aufgabenblätter und Rückseiten. Am Ende der Klausur sind zusätzliche Leerseiten. Fordern Sie zusätzliches Papier bitte nur an, falls Sie den gesamten Platz aufgebraucht haben.
- Die Tackernadel darf nicht gelöst werden.
- Als Hilfsmittel ist ein beschriebenes A4-Papier erlaubt.
- Einlesezeit: 15 min
  Bearbeitungszeit: 2 h

	Mögliche Punkte						Erreichte Punkte					
	a	b	c	d	e	Σ	a	b	c	d	e	Σ
Aufg. 1	1	1	3	3	1	9						
Aufg. 2	3	2	5	–	–	10				–	–	
Aufg. 3	1	1	1	6	–	9					–	
Aufg. 4	1	5	2	–	–	8				–	–	
Aufg. 5	1	2	6	–	–	9				–	–	
Aufg. 6	3	2	4	–	–	9				–	–	
Aufg. 7	1	2	3	–	–	6				–	–	
Σ						60						

Problem 1: Grammatiken/Pumping-Lemma $1 + 1 + 3 + 3 + 1 = 9$ Punkte

Eine *lineare Grammatik* ist eine kontextfreie Grammatik mit der zusätzlichen Eigenschaft, dass auf der rechten Seite jeder Regel höchstens eine Variable vorkommt. Wie immer bei kontextfreien Grammatiken besteht die linke Seite jeder Regel aus genau einer Variable. Eine *lineare Sprache* ist eine Sprache, die von einer linearen Grammatik erzeugt wird.

Wir wollen in dieser Aufgabe die Klasse der linearen Sprachen untersuchen. Dazu betrachten wir zwei Beispielsprachen:

- $L_1 = \{\mathsf{a}^j\mathsf{b}^j \mid j \in \mathbb{N}_0\}$. Aus der Vorlesung ist bekannt, dass L_1 kontextfrei, aber nicht regulär ist.
- $L_2 = \{\mathsf{a}^j\mathsf{b}^j\mathsf{c}^k\mathsf{d}^k \mid j, k \in \mathbb{N}_0\}$

(a) Geben Sie eine lineare Grammatik für L_1 an.

(b) Geben Sie eine kontextfreie Grammatik für L_2 an.

Für lineare Sprachen kann folgendes Pumping-Lemma gezeigt werden:

> Sei L eine lineare Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $z \in L$ mit $|z| > n$ eine Darstellung
>
> $$z = uvwxy \text{ mit } |uvxy| \leqslant n, vx \neq \varepsilon$$
>
> existiert, bei der auch $uv^iwx^iy \in L$ ist für alle $i \in \mathbb{N}_0$.

(c) Zeigen Sie (durch explizites Nachrechnen), dass L_1 die Aussage des Pumping-Lemmas für lineare Sprachen erfüllt.

Sei L eine lineare Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $z \in L$ mit $|z| > n$ eine Darstellung

$$z = uvwxy \text{ mit } |uvxy| \leqslant n, vx \neq \varepsilon$$

existiert, bei der auch $uv^i wx^i y \in L$ ist für alle $i \in \mathbb{N}_0$.

(d) Zeigen Sie, dass L_2 nicht linear ist.

(e) Wir bezeichnen mit $\mathcal{L}_i$ die Menge der Sprachen von Chomsky-Typ i und mit $\mathcal{L}_{\text{lin}}$ die Menge der linearen Sprachen. In welcher Beziehung stehen $\mathcal{L}_2$, $\mathcal{L}_3$ und $\mathcal{L}_{\text{lin}}$ zueinander? Kreuzen Sie die (einzige) richtige Wahl an. Begründen Sie für die richtige Wahl kurz beide Inklusionen bzw. (Un-)Gleichheiten.

- ☐ $\mathcal{L}_3 = \mathcal{L}_{\text{lin}} \subsetneq \mathcal{L}_2$
- ☐ $\mathcal{L}_3 \subsetneq \mathcal{L}_{\text{lin}} \subsetneq \mathcal{L}_2$
- ☐ $\mathcal{L}_{\text{lin}} \subseteq \mathcal{L}_3 \subseteq \mathcal{L}_2$
- ☐ $\mathcal{L}_3 \subsetneq \mathcal{L}_{\text{lin}} = \mathcal{L}_2$

Problem 2: Grammatiken/Automaten $3 + 2 + 5 = 10$ Punkte

Eine Grammatik $G = (\Sigma, V, S, R)$ heißt *fast-rechtslinear*, falls ihre Regeln folgende Form haben. Regeln ohne Startsymbol sind rechtslinear, haben also die Form

$$(i) \qquad Y \to wZ \mid \varepsilon \quad \text{mit } w \in \Sigma \text{ und } Y, Z \in V \setminus \{S\}.$$

Außerdem ist **genau** eine Regel erlaubt, die das Startsymbol auf eine oder mehrere Variablen abbildet. Diese Regel hat also die Form

$$(ii) \qquad S \to \alpha \qquad \text{mit } \alpha \in (V \setminus \{S\})^+.$$

Dabei darf niemals auf das Startsymbol abgebildet werden.

(a) Die Grammatik $G_{\text{Bsp}} = (\Sigma, V, S, R)$ mit $\Sigma = \{0, 1\}$, $V = \{S, A, B, C\}$ und folgender Regelmenge ist fast-rechtslinear.

$$\begin{aligned} R = \{ \;& S \to ACA \\ & A \to 0B \mid \varepsilon \\ & B \to 0A \\ & C \to 1C \mid \varepsilon \\ \} \end{aligned}$$

Geben Sie einen endlichen Automaten an, der die Sprache $L(G_{\text{Bsp}})$ erkennt.

Wir zeigen nun, dass fast-rechtslineare Grammatiken genau die regulären Sprachen erzeugen.

(b) Zeigen Sie, dass jede reguläre Sprache von einer fast-rechtslinearen Grammatik erzeugt werden kann.

(c) Sei G eine fast-rechtslineare Grammatik. Geben Sie einen endlichen Automaten an, der $L(G)$ erkennt.

Problem 3: NP-Vollständigkeit $1 + 1 + 1 + 6 = 9$ Punkte

Sei U eine Menge von Variablen und C eine Menge von Klauseln über U. In dieser Aufgabe darf jedes Literal höchstens einmal pro Klausel vorkommen. Aus der Vorlesung kennen Sie folgende Definition einer erfüllenden Wahrheitsbelegung.

Eine Wahrheitsbelegung ist *erfüllend* für (U, C), wenn jede Klausel ein wahres Literal enthält.

Wir definieren nun einen abgewandelten Erfüllbarkeitsbegriff.

Eine Wahrheitsbelegung ist NAE-*erfüllend* für (U, C), wenn jede Klausel ein wahres Literal und ein falsches Literal enthält.

Wir betrachten das folgende NP-schwere Entscheidungsproblem NOTALLEQUAL (NAE):

NOTALLEQUAL (NAE)	
Gegeben:	Menge U von Variablen Menge C von Klauseln über U
Frage:	Existiert NAE-erfüllende Wahrheitsbelegung für (U, C)?

Die folgende Variante von NAE verbietet negierte Variablen in den Klauseln, das heißt für jede Variable $u \in U$ darf das Literal u in den Klauseln vorkommen, aber das negierte Literal $\bar{u}$ darf in keiner Klausel vorkommen.

MONOTONENOTALLEQUAL (MNAE)	
Gegeben:	Menge U von Variablen Menge C von Klauseln über U, wobei kein Literal negiert ist
Frage:	Existiert NAE-erfüllende Wahrheitsbelegung für (U, C)?

(a) Geben Sie eine Lösung der folgenden MNAE-Instanz an, indem Sie die angegebene Tabelle ausfüllen.

$$U = \{a, b, c, d, e\}$$
$$C = \{(a, d), (b, e), (a, b, c), (b, c, d), (a, c, e)\}$$

a	b	c	d	e

(b) Sei $U = \{u, u'\}$ eine Menge von Variablen. Konstruieren Sie eine Menge C von Klauseln ohne negierte Literale über U, sodass für jede NAE-erfüllende Wahrheitsbelegung $t \colon U \to \{\texttt{wahr}, \texttt{falsch}\}$ gilt:

$$t(u) = \texttt{wahr} \iff t(u') = \texttt{falsch}.$$

(c) Zeigen Sie, dass MNAE in NP liegt. Geben Sie dazu an, woraus ein Lösungsvorschlag für eine MNAE-Instanz (U, C) besteht. Geben Sie außerdem die Laufzeit der Überprüfung in Abhängigkeit von $|U|$ und $|C|$ an, wobei $|M|$ die Anzahl der Elemente einer Menge M bezeichnet.

(d) Zeigen Sie, dass MONOTONENOTALLEQUAL NP-schwer ist.

Hinweis: Falls Sie Aufgabenteil (b) nicht gelöst haben, dürfen Sie die Notation $f(u, u')$ für eine Klauselmenge mit der angegebenen Eigenschaft verwenden.

Problem 4: Approximation $1 + 5 + 2 = 8$ Punkte

Wir betrachten in dieser Aufgabe ein beliebiges Minimierungsproblem Π.

(a) Sei $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ein PTAS für Π. Zeigen Sie: Für jede Konstante $\varepsilon > 0$ existiert ein polynomieller Approximationsalgorithmus mit relativer Gütegarantie $1 + \varepsilon$.

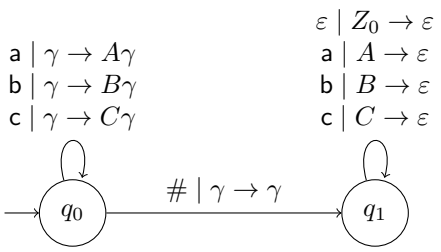
Sei nun $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$ ein FPTAS für Π. Sei außerdem p ein Polynom, sodass für alle Instanzen I von Π gilt: $\mathrm{OPT}(I) \leqslant p(|I|)$.

(b) Zeigen Sie: Für jede Konstante $\delta > 0$ existiert ein polynomieller Approximationsalgorithmus mit absoluter Gütegarantie δ.

(c) Für eine Instanz I von Π und eine (nicht notwendigerweise optimale) Lösung L von I sei $w(L)$ der Lösungswert von L. Wir nehmen nun zusätzlich an, dass es eine Konstante $c > 0$ gibt, sodass für jede Instanz I von Π und alle Lösungen L, L' von I mit $w(L) \neq w(L')$ gilt: $|w(L) - w(L')| \geqslant c$. Zeigen Sie, dass dann $\Pi \in \mathsf{P}$ gilt.

Problem 5: Kellerautomaten $1 + 2 + 6 = 9$ Punkte

Betrachten Sie den Kellerautomaten $\mathcal{M}_1 = (Q_1, \Sigma, \Gamma_1 = \{Z_0, A, B, C\}, q_0, Z_0, \delta_1)$ über dem Eingabealphabet $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \#\}$, der durch leeren Stack akzeptiert und folgenden Übergangsgraphen besitzt:



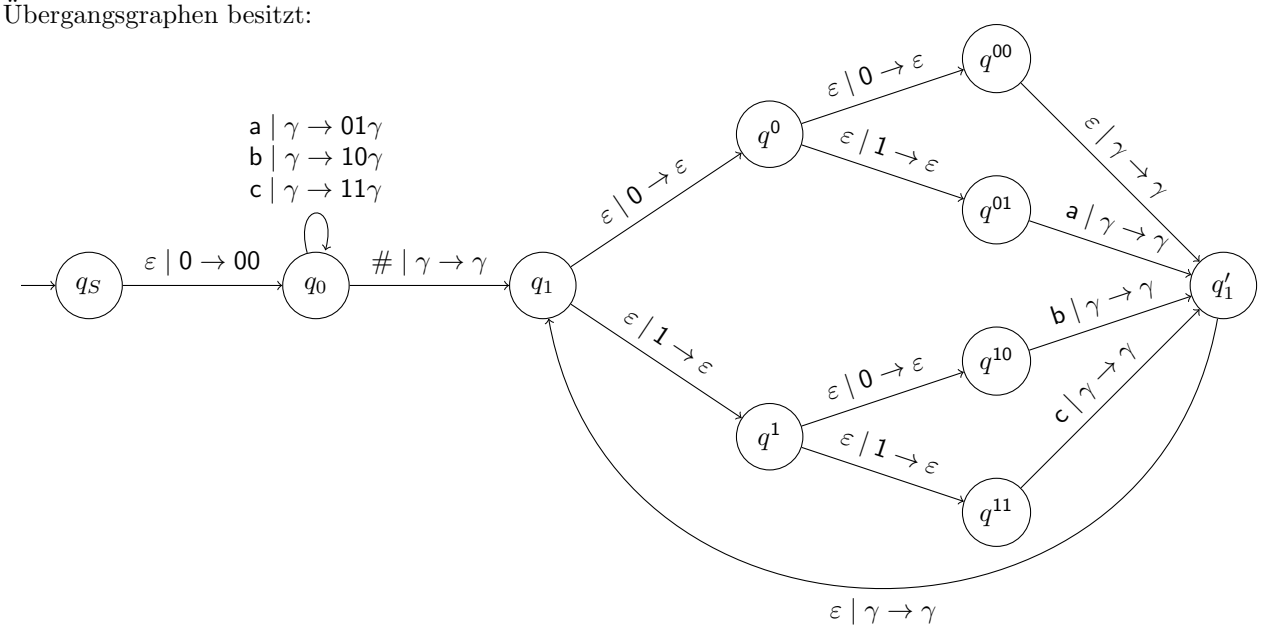
Das Symbol γ steht dabei für ein beliebiges Symbol $\gamma \in \Gamma_1$.

Wie aus der Übung bekannt, bedeutet die Notation $\mathsf{a} \mid B \to AB$: Beim zugehörigen Zustandsübergang wird das Eingabesymbol a gelesen, das oberste Stacksymbol B vom Stack gelöscht und AB auf den Stack gelegt. Dabei ist das linkeste Zeichen (hier also A) das oberste Zeichen auf dem Stack.

(a) Welche Sprache erkennt der Kellerautomat $\mathcal{M}_1$?

Im Folgenden betrachten wir *0-1-Kellerautomaten*. Das sind nichtdeterministische Kellerautomaten, deren Stackalphabet $\Gamma_2 = \{0, 1\}$ nur aus zwei Symbolen besteht. Das initiale Stacksymbol ist 0.

Wir definieren einen 0-1-Kellerautomaten, der die gleiche Sprache wie $\mathcal{M}_1$ erkennt. Dies ist der Automat $\mathcal{M}_2 = (Q_2, \Sigma, \Gamma_2 = \{0, 1\}, q_S, 0, \delta_2)$, der ebenfalls durch leeren Stack akzeptiert und folgenden Übergangsgraphen besitzt:



Auch hier steht das Symbol γ für ein beliebiges Symbol $\gamma \in \Gamma_2$.

(b) Geben Sie die Zustandsfolge an, die $\mathcal{M}_2$ bei der Abarbeitung von $w = \mathsf{ab\#ba}$ durchläuft. Geben Sie zusätzlich den Stackinhalt von $\mathcal{M}_2$ an, wenn er zum ersten Mal in den Zustand q_1 übergeht.

(c) Zeigen Sie, dass nichtdeterministische Kellerautomaten und 0-1-Kellerautomaten gleich mächtig sind.

Hinweis: Geben Sie dazu unter anderem eine geeignete Kodierung für die Stacksymbole in Γ *an und beschreiben Sie, wie ein 0-1-Kellerautomat ein kodiertes Zeichen auf den Stack schreibt bzw. vom Stack löscht.*

Problem 6: Maschinenmodelle $3 + 2 + 4 = 9$ Punkte

Die Komplexitätsklasse $\mathsf{DTAPE}(x)$ ist definiert als die Menge der Sprachen, die von einer deterministischen Turingmaschine mit Platzbedarf höchstens x entschieden werden können. Dabei wird angenommen, dass die Turingmaschine über ein separates Arbeitsband (mit eigenem Kopf) verfügt, auf dem der Platzbedarf gemessen wird. Auf dem Eingabeband darf nicht geschrieben werden.

Wir betrachten in dieser Aufgabe die Klassen $\mathsf{DTAPE}(x)$, bei denen x eine Konstante ist, also nicht von der Eingabelänge abhängt. Der Einfachheit halber dürfen Sie davon ausgehen, dass die x Felder des Arbeitsbandes, auf denen geschrieben werden darf, entsprechend markiert sind. Die Turingmaschine kann also erkennen, wenn sie aus dem Bereich hinausläuft, in dem geschrieben werden darf.

Sei Σ ein beliebiges, aber festes Alphabet. Zu einem Wort $w \in \Sigma^*$ bezeichne w^R das gespiegelte Wort. Wir betrachten folgende Sprache:

$$L = \{uvu^R \mid u, v \in \Sigma^*, |u| = 2022\}$$

Diese Sprache enthält also alle Wörter, bei denen die ersten 2022 Zeichen gespiegelt den letzten 2022 Zeichen entsprechen, aber nicht mit diesen überlappen.

(a) Geben Sie eine deterministische Turingmaschine mit Platzbedarf 2022 an, die L entscheidet und höchstens 1 Million Zustände benutzt. Sie müssen nicht begründen, dass Ihre Turingmaschine höchstens 1 Million Zustände benutzt.

(b) Zeigen Sie: $L \in \mathsf{DTAPE}(0)$.

(c) Zeigen Sie: $\mathsf{DTAPE}(2022) = \mathsf{DTAPE}(0)$.

Problem 7: Entscheidbarkeit $1 + 2 + 3 = 6$ Punkte

Sei $\Sigma = \{0, 1\}$ ein Alphabet. Aus der Vorlesung wissen Sie, dass das Halteproblem

$$\mathcal{H} = \{\langle \mathcal{M} \rangle \# w \mid \mathcal{M} \text{ hält bei Eingabe } w\}$$

unentscheidbar ist.

Betrachten Sie nun die folgende Sprache:

$$\mathcal{H}_{\text{eq}} = \{\langle \mathcal{M}_1 \rangle \# \langle \mathcal{M}_2 \rangle \mid \text{ für alle } v \in \Sigma^* \text{ gilt: } \mathcal{M}_1 \text{ hält auf } v \iff \mathcal{M}_2 \text{ hält auf } v\}$$

Im Folgenden sollen Sie zeigen, dass auch $\mathcal{H}_{\text{eq}}$ unentscheidbar ist.

(a) Konstruieren Sie eine Turingmaschine $\mathcal{N}$, sodass für jede Turingmaschine $\mathcal{M}$ gilt:

$$\langle \mathcal{N} \rangle \# \langle \mathcal{M} \rangle \in \mathcal{H}_{\text{eq}} \iff \mathcal{M} \text{ hält auf jeder Eingabe } v \in \Sigma^*$$

(b) Seien $\mathcal{M}$ eine Turingmaschine und $w \in \Sigma^*$ beliebig, aber fest. Konstruieren Sie eine Turingmaschine $\mathcal{N}_{w,\mathcal{M}}$ mit der folgenden Eigenschaft:

$$\mathcal{N}_{w,\mathcal{M}} \text{ hält auf jeder Eingabe } v \in \Sigma^* \iff \mathcal{M} \text{ hält auf } w$$

Ihre Konstruktion muss berechenbar sein.

(c) Zeigen Sie, dass $\mathcal{H}_{\text{eq}}$ nicht entscheidbar ist. Reduzieren Sie vom Halteproblem. Verwenden Sie nicht den Satz von Rice!