

**1. Klausur zur Vorlesung
Theoretische Grundlagen der Informatik
Wintersemester 2020/2021**

Hier Aufkleber mit Name und Matrikelnummer anbringen	
Vorname:	_____
Nachname:	_____
Matrikelnummer:	_____

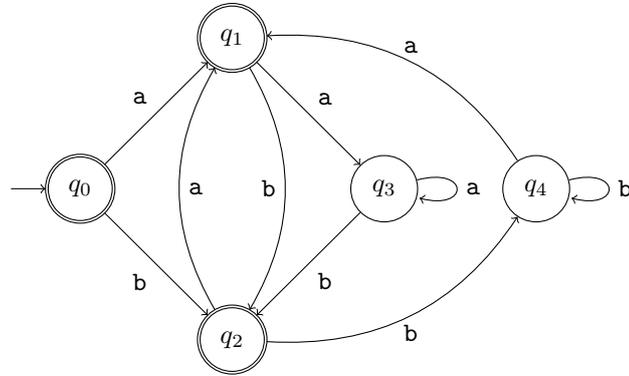
Beachten Sie:

- 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.
- Es sind keine Hilfsmittel zugelassen.

	Mögliche Punkte					Erreichte Punkte				
	a	b	c	d	Σ	a	b	c	d	Σ
Aufg. 1	2	3	4	–	9				–	
Aufg. 2	4	4	–	–	8			–	–	
Aufg. 3	4	2	1	3	10					
Aufg. 4	2	3	3	–	8				–	
Aufg. 5	1	4	4	2	11					
Aufg. 6	4	4	–	–	8			–	–	
Aufg. 7	3	3	–	–	6			–	–	
Σ					60					

Problem 1: Reguläre Sprachen

2 + 3 + 4 = 9 Punkte

Gegeben ist folgender endlicher Automat \mathcal{A} :

Dieser Automat erkennt die Sprache

$$L(\mathcal{A}) = \{w \in \{a, b\}^* \mid w \text{ endet nicht auf } \mathbf{aa} \text{ oder } \mathbf{bb}\}.$$

Für ein Alphabet Σ und ein Wort $w \in \Sigma^*$ bezeichnen wir mit $\text{pre}(w)$ die Menge aller Präfixe von w , außer w selbst.

Beispiel: $\text{pre}(\text{POMMES}) = \{\varepsilon, \text{P}, \text{PO}, \text{POM}, \text{POMM}, \text{POMME}\}$

Für eine Sprache $L \subseteq \Sigma^*$ sei $\text{pre}_{\text{all}}(L)$ die Menge aller Wörter aus L , für die alle Präfixe in L liegen. Formal ist

$$\text{pre}_{\text{all}}(L) = \{w \in L \mid \text{pre}(w) \subseteq L\}.$$

(a) Welche Sprache ergibt sich für $\text{pre}_{\text{all}}(L(\mathcal{A}))$?

(b) Geben Sie einen endlichen Automaten an, der $\text{pre}_{\text{all}}(L(\mathcal{A}))$ erkennt.

(c) Zeigen Sie, dass die regulären Sprachen unter der Operation pre_{all} abgeschlossen sind.

Problem 2: Endliche Klassifikatoren

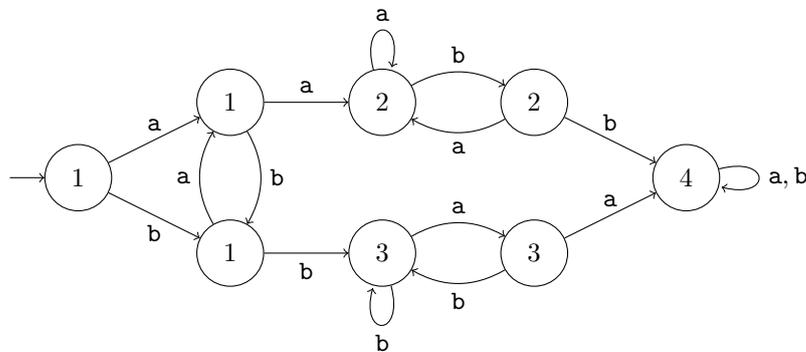
4 + 4 = 8 Punkte

Endliche Automaten, wie wir sie aus der Vorlesung kennen, unterteilen Wörter in zwei Klassen: „akzeptiert“ und „nicht akzeptiert“. Deshalb werden sie auch *endliche Akzeptoren* genannt. In dieser Aufgabe wollen wir das Konzept zu *endlichen Klassifikatoren* erweitern, die Wörter in mehr als 2 Klassen unterteilen.

Gegeben seien ein Alphabet Σ und ein festes $k > 2$. Analog zum Konzept der formalen Sprache definieren wir eine *formale Klassifikation* als eine Abbildung $K : \Sigma^* \rightarrow \{1, \dots, k\}$, die jedem Wort $w \in \Sigma^*$ eine Klasse $K(w)$ zuweist.

Ein *deterministischer endlicher Klassifikator* (DEK) ist ein 5-Tupel $\mathcal{A} = (Q, \Sigma, \delta, s, P)$ mit Zustandsmenge Q , Alphabet Σ , Überföhrungsfunktion δ und Startzustand s genau wie bei einem DEA. Statt einer Menge F von Endzuständen gibt es eine Zustandsfunktion $P : Q \rightarrow \{1, \dots, k\}$, die jedem Zustand $q \in Q$ eine Klasse $P(q)$ zuweist. Ein endlicher Klassifikator \mathcal{A} *erkennt* eine formale Klassifikation K , wenn er jedes Wort $w \in \Sigma^*$ in einen Zustand q mit $P(q) = K(w)$ überföhrt.

- (a) Betrachten Sie folgenden DEK für $k = 4$. Die Zustände sind jeweils mit ihrer Klasse beschriftet.



Vervollständigen Sie die Definition der formalen Klassifikation K , die von diesem DEK erkannt wird.

$$K(w) = \begin{cases} 1, & \text{falls} \\ 2, & \text{falls} \\ 3, & \text{falls} \\ 4, & \text{falls} \end{cases}$$

Hinweis: Betrachten Sie die Teilwörter aa und bb . Welche Zustände werden beim Lesen dieser Teilwörter erreicht?

Zu einer formalen Klassifikation K definieren wir die Sprachen L_1, \dots, L_k :

$$L_i = \{w \in \Sigma^* \mid K(w) = i\}$$

Also ist L_i die Sprache aller Wörter, die zur Klasse i gehören. Die L_i sind paarweise disjunkt und ihre Vereinigung entspricht Σ^* , d.h. jedes Wort ist in genau einer Sprache L_i enthalten.

(b) Zeigen Sie: Wenn es einen DEK gibt, der K erkennt, dann sind L_1, \dots, L_k alle regulär.

Problem 3: CYK-Algorithmus

4 + 2 + 1 + 3 = 10 Punkte

Gegeben sei die Grammatik $G = (\Sigma, V, S, R)$ mit Terminalen $\Sigma = \{a, b, c, d\}$, Nichtterminalen $V = \{S, A, B, C, D\}$, Startsymbol S und Produktionen

$$\begin{aligned}
 R = \{ & S \rightarrow CB \\
 & A \rightarrow BA \mid CD \mid a \mid d \\
 & B \rightarrow AC \mid b \\
 & C \rightarrow DA \mid b \mid c \\
 & D \rightarrow AB \mid d\}.
 \end{aligned}$$

- (a) Überprüfen Sie mit dem CYK-Algorithmus, ob das Wort $dabdc$ in der Sprache $L(G)$ enthalten ist.

d	a	b	d	c

- (b) Es gibt zwei Ableitungsbäume für $dabdc$ in G . Geben Sie beide an.

Wir führen nun Kosten für die Produktionen aus R ein.

- Produktionen mit Kosten 1: $A \rightarrow a \mid d, B \rightarrow b, C \rightarrow b \mid c, D \rightarrow d$
- Produktionen mit Kosten 2: $S \rightarrow CB, A \rightarrow CD, B \rightarrow AC, C \rightarrow DA$
- Produktionen mit Kosten 3: $A \rightarrow BA, D \rightarrow AB$

Wir definieren die Kosten eines Ableitungsbaums als die Summe der Kosten aller im Ableitungsbaum verwendeten Produktionen. Wenn eine Produktion mehrfach im Ableitungsbaum verwendet wird, zählt sie auch mehrfach. Die Ableitungskosten für ein Wort w definieren wir als das Minimum über die Kosten aller Ableitungsbäume, die w erzeugen.

(c) Berechnen Sie die Ableitungskosten von **dabdc**.

(d) Wie kann der CYK-Algorithmus so angepasst werden, dass er für das geprüfte Wort zusätzlich die Ableitungskosten berechnet?

Problem 4: Approximationsalgorithmen

2 + 3 + 3 = 8 Punkte

Gegeben sei ein Minimierungsproblem mit ganzzahligen, positiven Lösungswerten (z.B. COLOR). Welche der folgende Aussagen sind richtig und welche falsch? Zeigen bzw. widerlegen Sie jeweils.

(a) Jeder Approximationsalgorithmus mit absoluter Gütegarantie 1 hat auch eine relative Gütegarantie von 2.

(b) Für jedes $c \geq 1$ gilt: Jeder Approximationsalgorithmus mit absoluter Gütegarantie c hat auch eine relative Gütegarantie von 2.

(c) Für jedes $c \geq 1$ gilt: Jeder Approximationsalgorithmus mit absoluter Gütegarantie c hat auch eine relative Gütegarantie von $c + 1$.

Problem 5: NP-Vollständigkeit

1 + 4 + 4 + 2 = 11 Punkte

Wir betrachten Kreise in der Ebene. Zu einem Kreis gehört sowohl der Rand als auch das Innere. Wir sagen, dass sich zwei Kreise *schneiden*, wenn sie mindestens einen gemeinsamen Punkt haben. Insbesondere schneiden sich Kreise, wenn sie sich berühren, teilweise überlappen oder vollständig enthalten. Ein Kreis wird durch den Mittelpunkt und den Radius definiert.

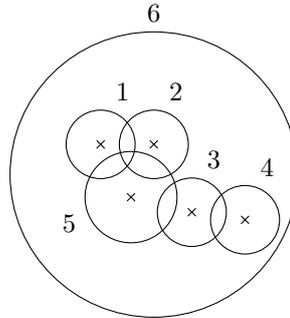


Abbildung 1: Kreise in der Ebene, die mit vier Farben gefärbt werden können, aber nicht mit drei Farben

Eine *Kreisfärbung* ordnet jedem Kreis eine Farbe zu, sodass je zwei Kreise, die sich schneiden, verschiedene Farben haben. Für eine natürliche Zahl k ist das k -FARBEN-KREISPROBLEM (k -FKP) wie folgt definiert:

Gegeben: Menge von Kreisen in der Ebene

Frage: Gibt es eine Kreisfärbung mit k Farben?

Sie dürfen voraussetzen, dass das 3-FARBEN-KREISPROBLEM NP-vollständig ist.

- (a) Färben Sie die sechs Kreise in Abbildung 1 mit den Farben rot, blau, grün und gelb. Tragen Sie hierzu die verwendeten Farben in die Tabelle ein.

1	2	3	4	5	6

- (b) Gegeben sei eine Menge M von Kreisen in der Ebene. Konstruieren Sie eine Menge $M' \supseteq M$ von Kreisen, die genau dann eine Kreisfärbung mit $k + 1$ Farben hat, wenn M eine Kreisfärbung mit k Farben hat. Begründen Sie, warum Sie genau eine Farbe mehr benötigen.

(c) Zeigen Sie, dass das 4-FARBEN-KREISPROBLEM NP-vollständig ist.

Hinweis: Sie dürfen die Aussage von Aufgabenteil (b) verwenden, auch wenn Sie (b) nicht gelöst haben.

- (d) Beschreiben Sie, wie der Beweis so erweitert werden kann, dass er die NP-Vollständigkeit des k -FARBEN-KREISPROBLEMS für jedes $k \geq 3$ zeigt.

Problem 6: In-Place-Turingmaschine

4 + 4 = 8 Punkte

Wir betrachten Turingmaschinen, deren Band auf die Felder beschränkt ist, auf denen die Eingabe steht. Zusätzlich gibt es vor dem ersten und nach dem letzten Symbol der Eingabe ein Blanksymbol, das aber nicht überschrieben werden darf. Die Turingmaschine darf auf den Feldern der Eingabe lesen und schreiben, aber keinen weiteren Platz verwenden. Der Platzbedarf einer Turingmaschine bezeichnet die Anzahl der Felder, auf denen im Laufe der Abarbeitung (mindestens einmal) geschrieben wird. Eine solche Turingmaschine nennen wir eine *In-Place-Turingmaschine*.

Sie dürfen in dieser Aufgabe verwenden, dass

- die universelle Sprache semi-entscheidbar ist, aber nicht kontextsensitiv,
 - die Sprache $\{w\#w \mid w \in \{0, 1\}^*\}$ kontextsensitiv ist, aber nicht kontextfrei und
 - die Sprache $\{w\#w^R \mid w \in \{0, 1\}^*\}$ kontextfrei ist, aber nicht regulär.
- (a) Zeigen Sie, dass es eine Sprache gibt, die von einer In-Place-Turingmaschine erkannt wird, aber von keinem Kellerautomaten.

Hinweis 1: Wählen Sie eine der angegebenen Sprachen und beschreiben Sie, wie eine In-Place-Turingmaschine diese Sprache akzeptiert. Begründen Sie, warum Ihre Wahl die Aufgabe löst.

Hinweis 2 (falls Sie nicht wissen, welche Sprache Sie wählen müssen): Sie erhalten Teilpunkte, wenn Sie für eine beliebige der angegebenen Sprachen zeigen, wie sie von einer In-Place-Turingmaschine akzeptiert wird.

(b) Zeigen Sie, dass In-Place-Turing-Maschinen weniger mächtig sind als klassische Turingmaschinen (d.h. 1 Kopf, 1 Band, wie in der Vorlesung definiert). Zeigen Sie hierfür, dass

- jede Sprache, die von einer In-Place-Turingmaschine akzeptiert wird, auch von einer klassischen Turingmaschine akzeptiert wird und
- es eine Sprache gibt, die von einer klassischen Turingmaschine akzeptiert wird, aber von keiner In-Place-Turingmaschine.

Hinweis: Kennen Sie ein Maschinenmodell, das zwischen In-Place-Turingmaschinen und klassischen Turingmaschinen liegt?

Problem 7: Entscheidbarkeit

3 + 3 = 6 Punkte

Wir betrachten die folgende Sprache

$$L = \{(\langle \mathcal{M} \rangle, q, w) \mid \langle \mathcal{M} \rangle \text{ ist die Gödelnummer einer Turingmaschine } \mathcal{M} \\ \text{ mit Zustandsmenge } Q \text{ und Eingabealphabet } \Sigma, \\ q \in Q, w \in \Sigma^*, \\ \mathcal{M} \text{ geht bei Eingabe } w \text{ mindestens einmal in den Zustand } q\}.$$

- (a) Gegeben sei eine Turingmaschine \mathcal{M} und ein Wort w . Konstruieren Sie eine Turingmaschine \mathcal{M}' mit einem Zustand q' und ein Wort w' , sodass

$$\mathcal{M}' \text{ bei Eingabe } w' \text{ in Zustand } q' \text{ geht} \iff \mathcal{M} \text{ auf der Eingabe } w \text{ hält.}$$

(b) Zeigen Sie, dass L nicht entscheidbar ist. Verwenden Sie nicht den Satz von Rice!

Hinweis 1: Nehmen Sie an, dass es einen Entscheider für L gibt und bauen Sie daraus einen Entscheider für das Halteproblem.

Hinweis 2: Sie dürfen die Aussage aus (a) verwenden, auch wenn Sie (a) nicht gelöst haben.

