Institut für Informatik der Ludwig-Maximilians-Universität München
Prof. Dr. David Sabel
Stephan Barth
Sebastian Sturm

SoSe 2019
24.09.2019

Klausur zur Vorlesung
Formale Sprachen und Komplexität

Die Bearbeitungszeit beträgt 120 Minuten. Hilfsmittel sind nicht erlaubt, auch der Besitz ausgeschalteter elektronischer Geräte wird als Betrug gewertet. Schreiben Sie Ihren vollständigen Namen und Ihre Matrikelnummer deutlich lesbar auf dieses Deckblatt, sowie Ihren Namen in die Kopfzeile auf **jedem Blatt** der Klausurangabe! Geben Sie alle Blätter ab, lassen Sie diese zusammengeheftet! Verwenden Sie nur dokumentenechte Stifte und weder die Farben rot noch grün.

Kontrollieren Sie, ob Sie alle Aufgabenblätter erhalten haben. Aufgabenstellungen befinden sich auf den Seiten 1–11. Sie dürfen die Rückseiten für Nebenrechnungen nutzen. Falls Sie die Rückseiten für Antworten nutzen, so markieren Sie klar, was zu welcher Aufgabe gehört und schreiben Sie in der entsprechenden Aufgabe, wo alle Teile Ihrer Antwort zu finden sind. Streichen Sie alles durch, was nicht korrigiert werden soll!

Lesen Sie die Aufgabenstellungen vollständig durch, bevor Sie mit der Bearbeitung der Aufgaben beginnen. Es gibt 8 unterschiedlich gewichtete Aufgaben zu insgesamt 120 Punkten. Mit 60 Punkten haben Sie sicherlich bestanden. Die Teilaufgaben können unabhängig voneinander bearbeitet werden.

Mit Ihrer Unterschrift bestätigen Sie, dass Sie zu Beginn der Klausur in ausreichend guter gesundheitlicher Verfassung sind, und diese Klausurprüfung verbindlich annehmen.

Nachname:

Vorname:

Matrikelnummer:

Studiengang:

Die Klausur soll gewertet werden: ☐ **JA** ☐ **NEIN**

Hinweis: Ist keines der beiden Felder angekreuzt, so wird **JA** angenommen.

Hiermit erkläre ich die Richtigkeit der obigen Angaben: ____________________
(Unterschrift)

Die folgende Tabelle **NICHT** ausfüllen:

Aufgabe	1	2	3	4	5	6	7	8	$\sum$	Bonus	Note
Punkte	18	12	12	18	15	15	18	12	**120**		
erreicht											

Aufgabe 1 (Allgemeines): (18 Punkte)

Pro Frage gibt es genau eine richtige Antwort.

Punkte in dieser Aufgabe werden nach folgender Tabelle vergeben. Eine Antwort, bei der 0 oder mindestens 2 Kästchen angekreuzt sind, zählt als falsch.

Richtige Antworten	0	1	2	3	4	5	6	7	8	9	10	11	12
Erzielte Punkte	0	0	0	0	2	4	6	8	10	12	14	16	18

Die primitiv rekursiven Funktionen umfassen alle

☐ WHILE-berechenbaren Funktionen. ☐ LOOP-berechenbaren Funktionen.
☐ μ-rekursiven Funktionen. ☐ Turingberechenbaren Funktionen.

Die Sprache $L = \{a^n b^n \mid n \in \mathbb{N}\}$

☐ ist nicht kontextfrei. ☐ ist endlich.
☐ hat einen endlichen Nerode-Index. ☐ ist nicht regulär.

Deterministische Kellerautomaten, die mit Endzuständen akzeptieren, erkennen

☐ genau die Typ 2-Sprachen.
☐ die gleichen Sprachen, wie deterministische Kellerautomaten, die bei leerem Keller akzeptieren.
☐ insbesondere alle regulären Sprachen.
☐ genau die Sprachen, die von NFAs erkannt werden.

Sei $a : (\mathbb{N} \times \mathbb{N}) \to \mathbb{N}$ die Ackermannfunktion, $bin : \mathbb{N} \to \{0,1\}^*$ die Binärkodierung einer Zahl und M die durch $w_M \in \{0,1\}^*$ kodierte Turingmaschine. Eine entscheidbare Sprache ist

☐ $\{w_M \in \{0,1\}^* \mid M \text{ hält bei leerer Eingabe}\}$
☐ $\{w_M \in \{0,1\}^* \mid M \text{ hält bei Eingabe } bin(a(|w_M|, |w_M|)) \}$
☐ $\{w_M \in \{0,1\}^* \mid M \text{ hält nicht bei leerer Eingabe}\}$
☐ $\{w_M \in \{0,1\}^* \mid M \text{ hält bei leerer Eingabe nach höchstens } a(|w_M|, |w_M|) \text{ Schritten}\}$

Welcher dieser Formalismen beschreibt eine andere Sprachklasse als die anderen drei Formalismen?

☐ Reguläre Ausdrücke ☐ Typ 1-Grammatiken
☐ Reguläre Grammatiken ☐ Nichtdeterministische endliche Automaten

---

Welche der folgenden Eigenschaften trifft für die Grammatik
$G = (\{S, A, B\}, \{a, b\}, \{S \to AB \mid BA, A \to a, B \to b, AA \to BB\}, S)$ zu?

☐ G ist in Kuroda-Normalform. ☐ G enthält Einheitsproduktionen.
☐ G ist kontextfrei. ☐ G enthält Epsilonproduktionen.

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 1:

Welche Aussage stimmt?

- ☐ Jede Typ 0-Sprache ist entscheidbar.
- ☐ Jede Typ 0-Sprache ist unentscheidbar.
- ☐ Jede entscheidbare Sprache ist vom Typ 0.
- ☐ Jede unentscheidbare Sprache ist vom Typ 0.

Aus welcher der folgenden Eigenschaften folgt, dass eine Sprache L über dem Alphabet Σ regulär ist?

- ☐ Es gibt eine Zahl $n > 0$, sodass jedes Wort $z \in L$ mit $|z| \geq n$, als $z = uvw$ geschrieben werden kann, so dass gilt: i) $|uv| \leq n$, ii) $|v| \geq 1$, iii) für alle $i \geq 0 : uv^i w \in L$
- ☐ Für jede Zahl $n > 0$, gibt es ein Wort $z \in L$ mit $|z| \geq n$ und für jede Zerlegung $z = uvw$ mit $|uv| \leq n$ und $|v| \geq 1$ gibt es ein $i \geq 0$, sodass $uv^i w \notin L$
- ☐ Es gibt eine Zahl $n > 0$, sodass jedes Wort $z \in L$ mit $|z| \geq n$, als $z = uvwxy$ geschrieben werden kann, so dass gilt: i) $|vwx| \leq n$, ii) $|vx| \geq 1$, iii) für alle $i \geq 0 : uv^i wx^i y \in L$
- ☐ Es gibt eine Zahl $n > 0$, sodass für jedes Wort $z \in \Sigma^*$ mit Mindestlänge n (d.h. $|z| \geq n$) gilt: für alle $i > 0 : z \notin L^i$.

Sei L eine kontextfreie Sprache.

- ☐ Dann ist der Nerode-Index von L unendlich.
- ☐ Dann gibt es eine Teilsprache $L' \subseteq L$, sodass L' regulär ist.
- ☐ Dann ist $\overline{L}$ keinesfalls eine reguläre Sprache.
- ☐ Dann verletzt L die Pumpingeigenschaft für reguläre Sprachen.

Seien L und M Sprachen und $M \in \mathcal{NP}$. Aus welcher der folgenden Eigenschaften folgt, dass M $\mathcal{NP}$-vollständig ist.

- ☐ $L \leq_p M$ und L ist $\mathcal{NP}$-hart.
- ☐ $L \leq_p M$ und $L \in \mathcal{NP}$.
- ☐ $M \leq_p L$ und L ist $\mathcal{NP}$-hart.
- ☐ $M \leq_p L$ und $L \in \mathcal{NP}$.

Welche der folgenden Sprachen ist von einem anderen Typ als die anderen drei Sprachen?

- ☐ $L(a^*b \mid ba^*)$
- ☐ $L(G)$ wobei $G = (\{S, B\}, \{a, b\}, \{S \to aS \mid B, B \to Bb \mid b\}, S)$
- ☐ $\{a^i b^i c^i \mid i \leq 100\}$
- ☐ $L(G')$ wobei $G' = (\{S, B\}, \{a, b\}, \{S \to aB, B \to Sb \mid b\}, S)$

Ein minimaler NFA für eine Sprache L ist ein NFA, der L erkennt und eine minimale Anzahl an Zuständen besitzt. Die Anzahl der Zustände eines minimalen NFA ist immer

- ☐ größer als der Nerode-Index.
- ☐ das Doppelte des Nerode-Index.
- ☐ kleiner oder gleich dem Nerode-Index.
- ☐ hat mit dem Nerode-Index nichts zu tun.

Aufgabe 2 (Formalismen für Reguläre Sprachen): **(12 Punkte)**

Sei F die formale Sprache über dem Alphabet $\{0, 1\}$, sodass F genau die Worte enthält, die mit einer 0 enden und nicht mit einer 1 beginnen, wobei das leere Wort nicht in F enthalten ist.

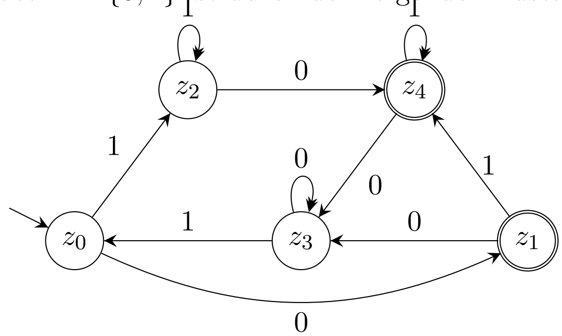
a) Geben Sie einen NFA A als Zustandsgraph an, sodass $L(A) = F$ gilt.

b) Geben Sie einen regulären Ausdruck α an, sodass $L(\alpha) = F$ gilt.

c) Geben Sie eine reguläre Grammatik G an, sodass $L(G) = F$ gilt.

Aufgabe 3 (DFA-Minimierung): **(12 Punkte)**

Der DFA A über dem Alphabet $\Sigma = \{0, 1\}$ ist durch den folgenden Zustandsgraphen gegeben:



a) Berechnen Sie einen Minimalautomaten B mit $L(A) = L(B)$.
Geben Sie den Rechenweg an (Minimierungstabelle).

b) Wie lautet der Nerode-Index von $L(A)$? Begründen Sie Ihre Antwort.

Aufgabe 4 (Akzeptanz von endlichen Automaten): (18 Punkte)

Spezielle deterministische endliche Automaten (SDFAs) sind DFAs, die das Alphabet $\{a\}$ verwenden und die genau einen Endzustand haben.

Definition: SDFA

Ein SDFA ist ein Tupel $A = (Z, \Sigma, \delta, z_0, z_E)$, wobei Z die Zustandsmenge, $\Sigma = \{a\}$ das Alphabet, $\delta : Z \times \Sigma \to Z$ die totale Übergangsfunktion, $z_0 \in Z$ der Startzustand und $z_E \in Z$ der einzige Endzustand ist.
Sei $\widehat{\delta} : (Z \times \Sigma^*) \to Z$ definiert als $\widehat{\delta}(z, aw) := \widehat{\delta}(\delta(z, a), w)$ und $\widehat{\delta}(z, \varepsilon) := z$.
Dann ist die vom SDFA A akzeptierte Sprache definiert als $L(A) := \{w \mid \widehat{\delta}(z_0, w) = z_E\}$.

Beweisen oder widerlegen Sie die folgenden Aussagen:

a) Es gibt einen SDFA A mit $|Z| = 2$, sodass $a \in L(A)$ und $aa \in L(A)$, aber $aaa \notin L(A)$.

b) Es gibt einen SDFA B mit $|Z| = 2$, sodass $a \notin L(B)$ und $aaa \notin L(B)$, aber $aaaa \in L(B)$.

c) Seien C und D beliebige SDFAs und $L(C)$ und $L(D)$ die akzeptierten Sprachen. Dann kann man einen SDFA konstruieren, der genau die Sprache $L(C) \cap L(D)$ akzeptiert.

Aufgabe 5 (CYK-Algorithmus): **(15 Punkte)**

Sei $G = (\{S, X, Y, A, B\}, \{a, b\}, P, S)$ mit

$$\begin{aligned} P = \{ \quad & S \to BX \mid YA \mid AA \mid BB \mid a \mid b, \\ & X \to SB, \\ & Y \to AS, \\ & A \to a, \\ & B \to b \quad \} \end{aligned}$$

a) Entscheiden Sie, ob

$$aababaa \in L(G)$$

gilt, indem Sie den Algorithmus von Cocke, Younger und Kasami (CYK-Algorithmus) ausführen. Geben Sie die dabei entstehende Tabelle und die Ausgabe des Algorithmus an und dokumentieren Sie Ihren Rechenweg.

b) Geben Sie an, welche Teilworte von *aababaa* ebenfalls in $L(G)$ liegen, indem Sie die Tabelle aus dem vorherigen Aufgabenteil verwenden (und keine neuen Tabellen berechnen). Erläutern Sie für jedes der Worte, warum sie schließen können, dass es in der Sprache $L(G)$ liegt.

Alternativ dürfen Sie diese Aufgabe bearbeiten (z.B. wenn Sie Aufgabenteil (a) nicht bearbeitet haben), indem Sie *allgemein* beschreiben, wie man den CYK-Algorithmus abändern kann, um herauszufinden, welche Teilworte für ein gegebenes Wort w in der Sprache $L(G)$ liegen.

Aufgabe 6 (Kellerautomat): (15 Punkte)

Bei verschachtelten arithmetischen Ausdrücken kann die Klammerung schnell unübersichtlich werden. Aus diesem Grund wollen wir auch eckige Klammern zulassen. Ausdrücke die eckige Klammern enthalten dürfen selbst nur noch mit eckigen Klammern geklammert werden. Der Einfachheit halber lassen wir nur den binären Operator + und nur die Zahl 2 zu.

Valide Ausdrücke sind also beispielsweise $[2 + ((2 + 2) + 2) + (2 + 2)]$ und $[[2 + (2 + (2 + 2))] + 2] + 2$, aber nicht $(2 + [(2 + (2 + 2))])$, da ein Ausdruck mit eckigen Klammern in runden geklammert wurde.

Geklammerte einzelne Zweien wie $((2))$ oder $[[((2))]]$ sind erlaubt.

Zeigen Sie, dass die beschriebene Sprache der geklammerten arithmetischen Ausdrücke kontextfrei ist, indem Sie einen Kellerautomaten angeben, der diese Sprache akzeptiert.

Erläutern Sie die Konstruktion Ihres Automaten in wenigen Sätzen: Welche Zustände dienen wozu? Welche Kellersymbole dienen wozu? Warum gibt es welche Zustandsübergänge?

Aufgabe 7 (Pumpinglemmata): **(18 Punkte)**

a) Zur Erinnerung:

Pumpinglemma für reguläre Sprachen.

Sei L eine reguläre Sprache. Dann gibt es eine Zahl $n \in \mathbb{N}_{>0}$, sodass jedes Wort $z \in L$, das Mindestlänge n hat (d.h. $|z| \geq n$), als $z = uvw$ geschrieben werden kann, so dass gilt:
i) $|uv| \leq n$, ii) $|v| \geq 1$ und iii) für alle $i \geq 0$: $uv^iw \in L$.

Zeigen Sie, dass die Sprache $L = \{w \in \{a,b\}^* \mid \#_a(w) < \#_b(w)\}$ nicht regulär ist.

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 7:

b) Zur Erinnerung:

Pumpinglemma für kontextfreie Sprachen.

Sei L eine kontextfreie Sprache. Dann gibt es eine Zahl $n \in \mathbb{N}_{>0}$, sodass jedes Wort $z \in L$, das Mindestlänge n hat (d.h. $|z| \geq n$), als $z = uvwxy$ geschrieben werden kann, so dass gilt:
i) $|vx| \geq 1$, ii) $|vwx| \leq n$ und iii) für alle $i \geq 0$: $uv^iwx^iy \in L$.

Zeigen Sie, dass die Sprache $L = \{a^i b^{2i} c^{3i} \mid i \in \mathbb{N}\}$ nicht kontextfrei ist.

Aufgabe 8 ($\mathcal{NP}$-Vollständigkeit): (12 Punkte)
Das PARTITION-Problem kann in der gegeben/gefragt-Notation definiert werden durch:

PARTITION	
gegeben:	Eine Folge von natürlichen Zahlen $S = (s_1, \ldots, s_n)$, wobei $\forall i : s_i \geq 1$.
gefragt:	Gibt es eine Menge I von Indizes $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$, sodass $\sum\limits_{i \in I} s_i = \sum\limits_{i \in \{1,\ldots,n\} \setminus I} s_i$ erfüllt ist (d. h. I partitioniert S in zwei Partitionen deren Summen identisch sind)?

Das PARTITION-Problem ist $\mathcal{NP}$-vollständig. Das Multimaschinen-Aufgaben-Scheduling-Problem ist:

MULTIMASCHINEN-AUFGABEN-SCHEDULING (MMAS)	
gegeben:	Ein Tupel $((M_1, \ldots, M_n), (B_1, \ldots, B_m), Z)$, wobei $M_1, \ldots, M_n$ eine Folge von parallel arbeitenden gleichartigen Maschinen ist, $A_1, \ldots, A_m$ eine Folge von Maschinenaufgaben ist, sodass B_i die benötigte Bearbeitungszeit für Aufgabe A_i (in Sekunden) ist, und Z eine Zeitschranke in Sekunden ist. Eine Maschine kann nur eine Aufgabe gleichzeitig bearbeiten und Aufgaben können während ihrer Bearbeitung die Maschine nicht wechseln.
gefragt:	Gibt es eine Zuordnung $\mathcal{M} : \{1, \ldots, m\} \to \{1, \ldots, n\}$, die jede Aufgabe A_k zu einer Maschine $M_{\mathcal{M}(k)}$ zuweist, welche die Aufgabe A_k in B_k Sekunden bearbeitet, sodass alle Aufgaben in nicht mehr als Z Sekunden ausgeführt werden können?

a) Zeigen Sie, dass die MMAS-Instanz

$$((M_1, M_2, M_3), (B_1, B_2, B_3, B_4, B_5), Z) = ((M_1, M_2, M_3), (15, 30, 25, 40, 50), 60)$$

lösbar ist, indem Sie die entsprechende Zuordnung $\mathcal{M} : \{1, 2, 3, 4, 5\} \to \{1, 2, 3\}$ angeben und prüfen, dass diese die Zeitschranke von 60 Sekunden einhält.

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 8:

b) Zeigen Sie, dass MMAS $\mathcal{NP}$-vollständig ist. Führen Sie dafür unter anderem eine Polynomialzeitreduktion von PARTITION auf MMAS durch.