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

SoSe 2019
30.07.2019

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	8	24	11	20	14	10	15	**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

Typ 0-Sprachen

- ☐ sind unter Komplement abgeschlossen
- ☐ sind unter Schnitt abgeschlossen
- ☐ haben ein entscheidbares Wortproblem
- ☐ haben ein entscheidbares Leerheitsproblem

Die Ackermannfunktion ist

- ☐ nicht LOOP-berechenbar, da sie so schnell wächst
- ☐ nicht LOOP-berechenbar, da sie nicht μ-rekursiv ist
- ☐ nicht WHILE-berechenbar, da sie so schnell wächst
- ☐ nicht WHILE-berechenbar, da sie nicht μ-rekursiv ist

LOOP-Programme können alle Programme berechnen, die von welchem der folgenden Formalismen berechnet werden können

- ☐ WHILE
- ☐ GOTO
- ☐ μ-Rekursion
- ☐ primitive Rekursion

Das Problem, ob eine gegebene Turingmaschine T auf der Eingabe w in höchstens k Schritten anhält, ist

- ☐ entscheidbar
- ☐ semi-entscheidbar, aber nicht entscheidbar
- ☐ nicht LOOP-berechenbar
- ☐ in $\mathcal{P}$

Sei f eine totale, WHILE-berechenbare Funktion; welche Aussage trifft dann für
$f^{-1}(x) := \{y \mid f(y) = x \wedge \forall z < y . f(z) \neq x\}$ sicher zu?

- ☐ f^{-1} ist total
- ☐ f^{-1} ist μ-rekursiv
- ☐ f^{-1} ist LOOP-berechenbar
- ☐ f^{-1} ist nicht LOOP-berechenbar

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 1:

Das Halteproblem von LOOP-Sprachen

- [ ] ist $\mathcal{NP}$-vollständig
- [ ] ist entscheidbar
- [ ] ist semi-entscheidbar, aber nicht entscheidbar
- [ ] ist unentscheidbar

Für welche dieser Sprachklassen wurde in der Vorlesung ein Pumpinglemma vorgestellt?

- [ ] endliche Sprachen
- [ ] Typ 0-Sprachen
- [ ] Typ 1-Sprachen
- [ ] Typ 2-Sprachen

Der CYK-Algorithmus löst das Wortproblem für

- [ ] Typ 0-Sprachen
- [ ] Typ 1-Sprachen, aber nicht Typ 0-Sprachen
- [ ] Typ 2-Sprachen, aber nicht Typ 1-Sprachen
- [ ] Typ 3-Sprachen, aber nicht Typ 2-Sprachen

Welcher dieser Sprachen hat den niedrigsten Nerode-Index?

- [ ] $\{a^i b^j \mid i, j \in \mathbb{N}_0, i + j < 10\}$
- [ ] $\{a^i b^j \mid i, j \in \mathbb{N}_0, i > j\}$
- [ ] $\{a^i b^i c^i \mid i \in \mathbb{N}_0\}$
- [ ] $\{a^i \mid i \in \mathbb{N}_0, i \text{ ist Primzahl}\}$

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

- [ ] deterministische endliche Automaten
- [ ] nichtdeterministische endliche Automaten
- [ ] deterministische Kellerautomaten
- [ ] reguläre Ausdrücke

Seien L, M Sprachen, so dass sich L in Polynomialzeit auf M reduzieren lässt. Was folgt daraus für L und M?

- [ ] L und M sind $\mathcal{NP}$-vollständig
- [ ] Wenn M $\mathcal{NP}$-vollständig ist, dann ist auch L $\mathcal{NP}$-vollständig
- [ ] Wenn L $\mathcal{NP}$-vollständig ist, dann ist M $\mathcal{NP}$-hart
- [ ] Wenn L in $\mathcal{NP}$ liegt, dann liegt auch M in $\mathcal{NP}$

Kontextfreie Sprachen

- [ ] sind unter Schnitt abgeschlossen
- [ ] sind unter Komplement abgeschlossen
- [ ] sind unter Vereinigung abgeschlossen
- [ ] sind unter keiner dieser Operationen abgeschlossen

Aufgabe 2 (Sprachtypen der Chomsky-Hierarchie): (8 Punkte)

Geben Sie für die folgenden Sprachen an, welche Typen der Chomsky-Hierarchie sie haben.

Dabei reicht es aus, jeweils diejenigen Typen anzugeben, mit denen bereits alle Typen eindeutig bestimmt sind, also beispielsweise: Typ 2 aber nicht Typ 3.

Alle Sprachen sind über dem Alphabet $\Sigma = \{a, b\}$ definiert.

a) $\{a, b, ab\}$

b) $L(((aa^*bb^*)^*b^* \mid abababbb^* \mid bba)^* \mid bba^*bba^*bb)$

c) $\{a^nb^na^n \mid n \in \mathbb{N}_0, n > 100\}$

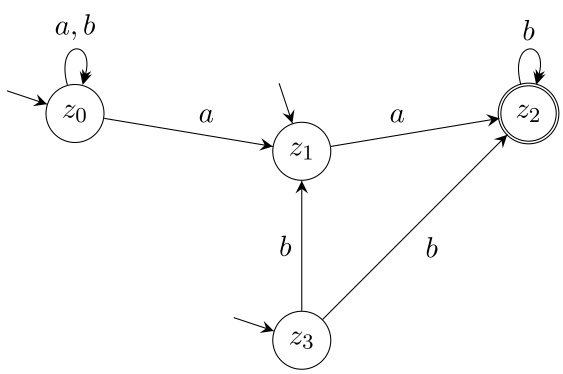
d) $L(G)$, wobei $G = (V, \Sigma, P, A)$, wobei

- $V = \{A, B\}$
- $\Sigma = \{a, b\}$
- $P = \{A \to ab \mid aBb, B \to ba \mid BB \mid bAa\}$

Aufgabe 3 (Endliche Automaten und Akzeptanzstärken): **(24 Punkte)**

a) Zeigen Sie, dass jeder NFA, der genau einen Zustand hat, und der die Wörter aaa und bbb akzeptiert, auch das Wort $abab$ akzeptiert.

b) Der NFA A über dem Alphabet $\Sigma = \{a, b\}$ ist durch den folgenden Zustandsgraph gegeben:

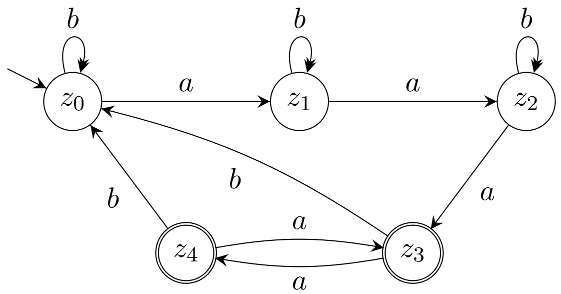


Berechnen Sie einen DFA B aus dem NFA A mit der Potenzmengenkonstruktion. Es ist ausreichend die erreichbaren Zustände von B anzugeben.

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 3:

c) Der DFA C über dem Alphabet $\Sigma = \{a, b\}$ ist durch den folgenden Zustandsgraph gegeben:



Berechnen Sie einen Minimalautomaten D mit $L(C) = L(D)$.
Geben Sie den Rechenweg an (Minimierungstabelle).

d) Zeigen Sie, dass die Sprachen, die von NFAs $(Z, \Sigma, \delta, S, E)$ mit $S = \varnothing$ erkannt werden können, genau diejenigen Sprachen sind, die von einem NFA mit $E = \varnothing$ erkannt werden können.

Aufgabe 4 (Der CYK-Algorithmus): (11 Punkte)

a) Bekanntlich benötigt der CYK-Algorithmus eine Grammatik in Chomsky-Normalform. Definieren Sie die Chomsky-Normalform in eigenen Worten.

b) Sei $G = (V, \Sigma, P, S)$ mit

- $V = \{S, A, B, U, T\}$
- $\Sigma = \{a, b\}$
- $P = \{S \to SU \mid AT, \quad U \to AT, \quad T \to UB \mid b, \quad A \to a, \quad B \to b\}$

Führen Sie den CYK-Algorithmus für die Eingabe

$$aabbab$$

vollständig aus und geben Sie die dabei entstehende Tabelle an.

c) Beschreiben Sie die Sprache $L(G)$ (mit G aus Teilaufgabe b)) möglichst kompakt.

Aufgabe 5 (Pumpinglemma): (20 Punkte)

Zur Erinnerung geben wir die beiden Pumpinglemmas an:

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 $i \geq 0$: $uv^iw \in L$.

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 $i \geq 0$: $uv^iwx^iy \in L$.

a) Zeigen Sie, dass die Sprache $M = \{a^ib^{i+j}a^j \mid i, j \in \mathbb{N}_0\}$ nicht regulär ist.

Die Aufgabe wird auf dem nächsten Blatt fortgesetzt.

Fortsetzung von Aufgabe 5:

b) Zeigen Sie, dass die Sprache $N = \{(a^i b)^j \mid i, j \in \mathbb{N}_0\}$ nicht kontextfrei ist.

c) Für einen regulären Ausdruck α ist $s(\alpha)$ der Shuffle-Operator, der formal durch

$$s(\alpha) := \{w \mid \exists v \in L(\alpha) \forall a \in \Sigma. \#_a(v) = \#_a(w)\}.$$

definiert ist. Informell erlaubt der Shuffle-Operator die Buchstabenreihenfolge der in $L(\alpha)$ liegenden Worte beliebig zu ändern.

Sei $S = L(s(a^*b))$.

- Falls S eine Typ-3-Sprache ist, dann geben Sie einen DFA A an mit $L(A) = S$.
- Falls S eine Typ-2-Sprache, aber keine Typ-3-Sprache ist, dann geben Sie einen PDA B mit $L(B) = S$ an und beweisen Sie, dass S keine Typ-3-Sprache ist.
- Ansonsten beweisen Sie, dass S weder eine Typ-3-Sprache, noch eine Typ-2-Sprache ist.

Aufgabe 6 (Reguläre Ausdrücke und LOOP-Programme): (14 Punkte)

a) Für einige Kodierungen ist es wichtig, dass gleiche Zeichen nicht zu oft hintereinander auftauchen. Es sei darum

$$L = \{w \mid 000 \text{ ist kein Teilwort von } w \wedge 111 \text{ ist kein Teilwort von } w\}$$

über dem Alphabet $\Sigma = \{0, 1\}$.

- Geben Sie einen regulären Ausdruck α an, mit $L(\alpha) = L$.

- Geben Sie einen regulären Ausdruck β an, mit $L(\beta) = \overline{L}$ (das Komplement von L).

b) Verwenden Sie in dieser Aufgabe ausschließlich die folgenden Konstrukte für LOOP-Programme:

$$\begin{array}{l} x_i := x_j + c \\ x_i := x_j - c \\ x_i := c \\ P_1;\, P_2 \\ \text{LOOP } x_i \text{ DO } P \text{ END} \end{array}$$

Weitere abkürzende Konstrukte aus Vorlesung, Übung oder sonstigen Quellen sind nicht erlaubt!

Schreiben Sie ein LOOP-Programm, welches h berechnet, mit

$$h(x, y) = \begin{cases} 0 & \text{wenn} \quad x < y \\ 1 & \text{wenn} \quad x = y \\ x + y & \text{wenn} \quad x > y \end{cases}$$

Aufgabe 7 (Reduktion): (10 Punkte)

Wir erinnern zunächst an die Definition des Postschen Korrespondenzproblems:

> Gegeben sei ein Alphabet Σ mit $|\Sigma| > 1$ und eine Folge von Wortpaaren $K = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ mit $x_i, y_i \in \Sigma^+$. Das Postsche Korrespondenzproblem (PCP) ist die Frage, ob es für die gegebene Folge K eine Folge von Indizes $i_1, \ldots, i_m$ mit $i_j \in \{1, \ldots, k\}$ gibt, sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$ gilt.

In der Vorlesung wurde die Unentscheidbarkeit des PCP gezeigt.

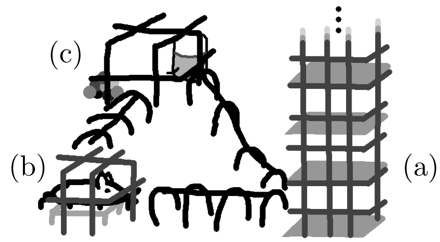
Das *123-PCP* ist das PCP-Problem über dem Alphabet $\Sigma = \{1, 2, 3\}$.

Zeigen Sie die Unentscheidbarkeit von 123-PCP, indem Sie PCP auf 123-PCP reduzieren.

Aufgabe 8 (Kellerautomat mit der Maus): (15 Punkte)

Eine Maus wohnt in einem Netzwerk aus verbundenen Käfigen.

Diese sind:
Turm-Käfig (a), unendlich hoch;
Schlaf-Käfig (b);
Essens-Käfig (c), mit Loch.



Schematische Darstellung der Mausbehausung, Maus im Bett

Die Maus kann folgende Aktionen durchführen:

- Nach **o**ben gehen, aber nur, wenn sie im Turm-Käfig ist.
- Nach **u**nten gehen, aber nur, wenn sie im Turm-Käfig ist.
- Auf den Boden (0-ter Stock) des Turm-Käfigs **s**pringen, aber nur, wenn sie im Turm-Käfig ist.
- **I**m Uhrzeigersinn den Käfig wechseln (Schlaf-Käfig $\to$ Essens-Käfig $\to$ 0-ter Stock im Turm-Käfig $\to$ Schlaf-Käfig)
- **G**egen den Uhrzeigersinn den Käfig wechseln.
- **E**ssen und Trinken, aber nur, wenn sie im Essens-Käfig ist.
- **F**lüchten, aber nur, wenn sie sich im Essens-Käfig befindet (da hinter dem Essen ein Loch im Käfig ist).

Geben Sie einen DPDA über dem Alphabet $\{o, u, s, i, g, e, f\}$ an, der alle möglichen Abfolgen von beobachtbaren Verhaltensweisen der Maus an einem Tag akzeptiert. Dabei bedeutet beobachtbar:

Sie wacht im Schlaf-Käfig auf und schläft dort wieder ein, wenn sie nicht geflüchtet ist. Wenn sie flüchtet, ist danach kein Verhalten mehr beobachtbar, die Flucht ist das letzte beobachtete Verhalten.

Beispiele:

Damit ist beispielsweise folgendes beobachtbar: iii (nach 3 mal im Uhrzeigersinn gehen, ist sie wieder im Schlafkäfig), $ieeef$ (in den Essenskäfig, dort ein paar mal gegessen und geflüchtet), $gooouooosgf$ (in den Turmkäfig gewechselt, dort ein paar Stockwerke nach oben, eines nach unten, wieder einige nach oben, dann nach unten gesprungen, dann in den Essenskäfig gewechselt und geflüchtet).

Nicht beobachtbar sind hingegen z.B.: g (die Maus ist am Ende des Tages damit weder in Freiheit, noch im Schlafkäfig), $gooi$ (die Maus kann nicht von einem oberen Stockwerk aus den Käfig wechseln).