Universität Osnabrück / FB6 / Theoretische Informatik — Prof. Dr. M. Chimani

Informatik D: Einführung in die Theoretische Informatik

Klausur — SoSe 2017 — 25. September 2017

Nebentermin, Prüfungsnr. 1007049 — Gruppe: Drogon und Rhaegal

Unbedingt ausfüllen

Matrikelnummer	Studiengang/Abschluss	Fachsemester

Nachname	Vorname

Unterschrift	Identifikator (Beliebiges Wort zur Identifikation im anonymen Notenaushang)

Grundregeln

- Die Bearbeitungszeit der Klausur beträgt **120 Minuten**.
- Sie schreiben diese Klausur **vorbehaltlich** der Erfüllung der **Zulassungsvoraussetzung**. Das heißt: Wir werden Ihre Zulassung vor Korrektur prüfen; die Tatsache, dass Sie die Klausur mitschreiben, bedeutet keine implizite Zulassung.
- Es sind **keine Unterlagen** und auch **keine** anderen **Hilfsmittel** erlaubt.
- Benutzen Sie nur dokumentenechten (blauen oder zur Not schwarzen) **Kugelschreiber**! Bleistiftlösungen werden nicht gewertet!
- Es zählt die Antwort, die sich im dafür vorgesehenen Kästchen befindet! Soll eine andere Antwort gewertet werden, so ist diese **eindeutig** zu kennzeichnen!
- Jegliches Schummeln, und auch der Versuch desselben, führt zum Ausschluss von der Klausur und einer Bewertung mit **5,0**.

Wird vom Korrektor/Prüfer ausgefüllt

Aufgabe	1	2	3	4	5	6	7	8	9	10	Σ
Punkte (max)	12	10	12	12	16	10	8	16	10	28	**134**
Punkte (erreicht)											

Punkte	0..66	67..74	75..82	83..87	88..92	93..98	99..103	104..108	109..113	114..121	122..134
Note	**5,0**	**4,0**	**3,7**	**3,3**	**3,0**	**2,7**	**2,3**	**2,0**	**1,7**	**1,3**	**1,0**

Note:

Aufgabe 1: Sprachgrundlagen **(12 Punkte)**

Welche Form haben die Regeln einer (allgemeinen) Grammatik?

Sei Σ die Menge der Symbole und V die Menge der Variablen.

Welche Einschränkungen haben die Regeln kontextsensitiver Grammatiken?

Durch welches Maschinenmodell wird bzw. durch welche Maschinenmodelle werden kontextfreie Sprachen beschrieben?

Sei L eine reguläre Sprache. Beschreiben Sie, wie man das Leerheitsproblem für L entscheiden kann. Welche Laufzeit hat dieser Algorithmus?

Aufgabe 2: Sprachen klassifizieren **(10 Punkte)**

Zu welcher der *fünf* in der Vorlesung besprochenen Sprachfamilien innerhalb der Chomsky-Hierarchie gehören die folgenden Sprachen? Geben Sie dabei die *kleinste* korrekte Antwort an, also die Sprachfamilie, die gerade mächtig genug ist, die entsprechende Sprache zu erkennen.

$\{a^i a^j a^k b^i b^j c^k d^\ell \mid i, j, k, \ell \in \mathbb{N}\}$

$S \to abcS \mid abcA \qquad A \to \varepsilon \mid a \mid b \mid c \mid AA$

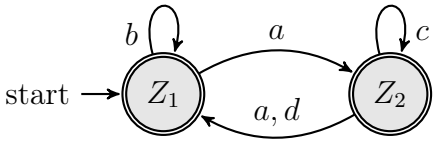
$\{w \mid w$ ist ein GOTO-Programm, nach dessen Ausführung die Variable x_1 den Wert 0 hat$\}$

L^+, wobei L deterministisch kontextfrei ist.

$\mathcal{L}(M)$, wobei M eine Turingmaschine ist, die den Schreib-Lese-Kopf nie nach links bewegt.

Aufgabe 3: Endlicher Automat **(12 Punkte)**

Gegeben sei der rechts abgebildete endliche Automat $\mathcal{A}$ über dem Alphabet $\Sigma = \{a, b, c, d\}$.
Sei $L := \mathcal{L}(\mathcal{A})$.



(a) Umwandlung in RegEx (8 Punkte)

Geben Sie einen regulären Ausdruck E mit $\mathcal{L}(E) = L$ an. Nutzen Sie dazu das Vorgehen aus der Vorlesung.

(b) Komplementbildung (4 Punkte)

Geben Sie einen endlichen Automaten für die Sprache $\overline{L}$ an.

Aufgabe 4: Kontextfreie Grammatik **(12 Punkte)**

Gegeben sei die folgende Grammatik G:

$$S \to aSc \mid cSa \mid aba \mid cbc$$

(a) Deterministisch kontextfrei (8 Punkte)

Zeigen Sie, dass die Sprache $\mathcal{L}(G)$ deterministisch kontextfrei ist.

(b) Chomsky-Normalform (4 Punkte)

Geben Sie eine Chomsky-Normalform für die Sprache $\mathcal{L}(G)$ an. (Sie dürfen, müssen aber nicht das Verfahren aus der Vorlesung anwenden. Das Endergebnis genügt.)

Aufgabe 5: Pumping Lemma (16 Punkte)

(a) Definition (4 Punkte)

Wie lautet das Pumping Lemma für kontextfreie Sprachen?

(b) Anwendung (12 Punkte)

Beweisen Sie mittels Pumping Lemma, dass die Sprache $L := \{(a^k b)^i \mid i, k \in \mathbb{N}_+\}$ nicht kontextfrei ist.

Angenommen ...

Aufgabe 6: Kontextsensitive Sprache **(10 Punkte)**

Zeigen Sie, dass die Sprache $L := \{(a^k b)^i \mid i, k \in \mathbb{N}_+\}$ kontextsensitiv ist.

Aufgabe 7: Palindrom-PCP **(8 Punkte)**

Sie kennen das Post'sche Korrespondenzproblem für eine gegebene Instanz $\{(x_i, y_i)\}_{i=1,\ldots,k}$. Zeigen Sie, dass es co-semi-entscheidbar ist, ob jedes Lösungswort des PCP ein Palindrom ist.

Aufgabe 8: Zusammenhänge in Berechenbarkeit und Komplexität (16 Punkte)

In jeder Teilaufgabe führt genau ein Kreuz zu einer wahrheitsgemäßen Aussage. Kreuzen Sie die korrekten Aussagen an.
Achtung: Pro Teilaufgabe gibt es $2/0/-1$ *Punkte bei einer richtigen/keinen/falschen Antwort! Es gibt jedoch keine negativen Punkte für die gesamte Aufgabe.*

Sei $\mathcal{F}$ die Menge aller berechenbaren Funktionen. Es ist unentscheidbar, ob ein gegebenes Programm eine Funktion aus $\mathcal{T} \subsetneq \mathcal{F}$ mit $\mathcal{T} \neq \varnothing$ berechnet. Dies besagt

- ☐ ...der Satz von Kleene.
- ☐ ...der Satz von Rice.
- ☐ ...das Cook-Levin-Theorem.

Sei $f\colon \mathbb{N} \to \mathbb{N}$ eine beliebige Funktion, die bijektiv (d. h. zu jedem $y \in \mathbb{N}$ gibt es genau ein $x \in \mathbb{N}$ mit $f(x) = y$) und berechenbar ist.

- ☐ Die Umkehrfunktion f^{-1} ist berechenbar.
- ☐ Die Umkehrfunktion f^{-1} ist nicht berechenbar.
- ☐ Man kann über die Berechenbarkeit von f^{-1} keine generelle Aussage treffen.

Sei $S(n)$ die Busy-Beaver-Funktion und $A(n)$ die Ackermannfunktion. Bekannt sind unter anderem die Werte $S(1) = 1, S(2) = 6, S(3) = 21$ und $A(1) = 2, A(2) = 4, A(3) = 27$. Es gilt

- ☐ ...$A(n) \in \mathcal{O}(S(n))$.
- ☐ ...$S(n) \in \mathcal{O}(A(n))$.
- ☐ ...$A(n) \in \Theta(S(n))$.

Sei L eine Sprache bei der die negative Hälfte ihrer charakteristischen Funktion nicht berechenbar ist. Dann kann L trotzdem

- ☐ ...entscheidbar sein.
- ☐ ...semi-entscheidbar sein.
- ☐ ...co-semi-entscheidbar sein.

Man kann zeigen, dass ein NP-vollständiges Problem schwach NP-vollständig ist, indem man zeigt, dass es

- ☐ ...einen Spezialfall des Problems gibt, der in polynomieller Zeit lösbar ist.
- ☐ ...eine dynamische Programmierung zur Lösung des Problems gibt.
- ☐ ...einen pseudopolynomiellen Algorithmus zur Lösung des Problems gibt.

Wenn FACTORIZATION $\in \mathsf{P}$, dann

- ☐ ...folgt $\mathsf{P} = \mathsf{NP}$.
- ☐ ...folgt $\mathsf{P} \neq \mathsf{NP}$.
- ☐ ...ist die Frage zum Verhältnis von P und NP weiterhin offen.

Aus $\mathsf{P} = \mathsf{Co\text{-}NP}$

- ☐ ...folgt $\mathsf{NPI} = \varnothing$.
- ☐ ...folgt $\mathsf{PSPACE} = \mathsf{NL}$.
- ☐ ...folgt $\mathsf{NP} \neq \mathsf{Co\text{-}NP}$.

Für Probleme aus PSPACE gilt nach aktuellem Wissensstand, dass

- ☐ ...alle in Polynomialzeit gelöst werden können.
- ☐ ...keines in Polynomialzeit gelöst werden kann.
- ☐ ...einige, aber nicht alle, in Polynomialzeit gelöst werden können.

Aufgabe 9: Turing-Vollständigkeit **(10 Punkte)**

Wir definieren die Programmiersprache TapeMachine. Sie benutzt ein einseitig abgeschlossenes, unendlich langes Band B, das in Zellen $B_0, B_1, \ldots$ eingeteilt ist. Für alle Indizes $i \in \mathbb{N}$ hat die Zelle B_i einen Wert $b_i \in \mathbb{N}$. TapeMachine benutzt einen (Schreib-Lese-)Kopf, der auf dem Band entlangfahren kann und den Inhalt einer Zelle lesen und schreiben kann. Zu Beginn steht der Kopf auf Zelle B_0.

Ein TapeMachine-Programm ist eine Befehlsfolge aus den folgenden Befehlen:

Befehl	Aktion
`>`^{k}	bewegt den Kopf von der aktuellen Zelle B_i zur Zelle B_{i+k}.
`<`^{k}	bewegt den Kopf von der aktuellen Zelle B_i zur Zelle B_{i-k}, falls $i \geq k$. (Geht in eine Endlosschleife, falls $i < k$.)
`+`^{k}	setzt den Wert b_i der aktuellen Zelle auf $b_i + k$.
`-`^{k}	setzt den Wert b_i der aktuellen Zelle auf $b_i - k$, falls $b_i \geq k$. (Geht in eine Endlosschleife, falls $b_i < k$.)
`[`F`]`	führt die Befehlsfolge F aus, falls die aktuelle Zelle B_i einen Wert $b_i > 0$ hat. Solange die nach der Ausführung von F aktuelle Zelle einen Wert > 0 hat, wird F abermals ausgeführt.

Für $k = 1$ werden die Exponenten einfach weggelassen, d. h. `<>+-` entspricht `>`^{1}`<`^{1}`+`^{1}`-`^{1}.

Beispiel „$b_1 := 5 \cdot b_0$“: Sei $\alpha \in \mathbb{N}$. Zu Beginn ist $b_0 = \alpha$ und $b_i = 0$ für $i > 0$.

`[->+>+<`^{2}`]>`	setzt $b_0 = 0$ und $b_1 = b_2 = \alpha$, der Kopf steht auf B_1,
`[-<+>]`	setzt daraufhin $b_0 = \alpha$ und $b_1 = 0$, der Kopf steht auf B_1,
`>[-<+`^{5}`>]<`^{2}	setzt daraufhin $b_1 = 5 \cdot \alpha$ und $b_2 = 0$, der Kopf steht auf B_0.

Zeigen Sie, dass TapeMachine Turing-vollständig ist.

Aufgabe 10: Ein Rechenrätsel **(28 Punkte)**

Betrachten Sie das folgende Problem:

Problem: RechenRätsel

Gegeben: eine Sequenz von Paaren $(s_1, z_1), \ldots, (s_n, z_n)$ mit $s_i \in \{+1, -1, \square\}$ und $z_i \in \mathbb{N}$, und eine Zahl $r \in \mathbb{Z}$.

Gefragt: Ist es möglich, die einzelnen s_i mit $s_i = \square$ durch $+1$ oder -1 ersetzen, sodass $\sum_{i=1}^{n} s_i z_i = r$?

Sie sollen im Folgenden die ***NP***-Vollständigkeit von RechenRätsel zeigen.
Welches Problem nutzen Sie dafür? ☐ SubsetSum ☐ BinPacking ☐ Partition

Definieren Sie dieses Problem. (4 Punkte)

Zeigen Sie nun die ***NP***-Vollständigkeit von RechenRätsel. Beschreiben Sie, was Sie genau tun und zeigen. Begründen Sie alle notwendigen Eigenschaften. (14 Punkte)

Fortsetzung des Beweises zur NP-Vollständigkeit

Ist RechenRätsel unter der Einschränkung, dass (6 Punkte)

(1) keines der Paare ein □ enthält,
(2) jedes der Paare ein □ enthält,
(3) genau die Hälfte der Paare ein □ enthält,

jeweils auch ***NP***-vollständig? Warum oder warum nicht?

Sei k die Anzahl der □ in einer RechenRätsel-Instanz. Das Problem liegt in ***FPT*** in Bezug auf Parameter k. Warum? (4 Punkte)