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

Informatik D: Einführung in die Theoretische Informatik

Klausur — SoSe 2019 — 10. Juli 2019

Haupttermin, Prüfungsnr. 1007049 — Gruppe: Mallorca / Cabrera

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	Σ
Punkte (max)	**0**
Punkte (erreicht)	

Punkte	0..-1	0..-2	-1..-2	-1..-3	-2..-4	-3..-5	-4..-5	-4..-6	-5..-7	-6..-7	-6..0
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)**

(a) Faktenwissen (4 Punkte)

Von welchem Maschinenmodell werden genau die kontextsensitiven Sprachen beschrieben?

Welche Einschränkungen haben die Regeln einer Grammatik, die eine kontextfreie Sprache beschreibt?

(b) Zuordnung von Sprachen (8 Punkte)

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

$\{a^n b^{n+1} \mid n \geq 2\} \cup \{a^{n+1} b^n \mid n \geq 2\}$

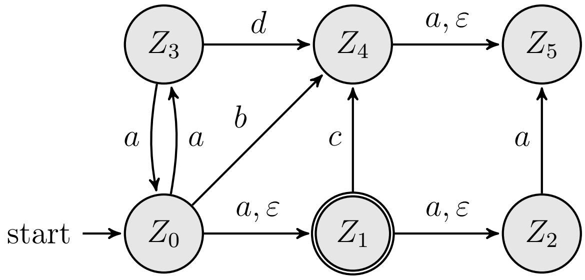
$\left\{ z \;\middle|\; \begin{array}{r} z \text{ ist Dezimaldarstellung einer} \\ \text{durch 3 teilbaren Zahl} \end{array} \right\} \varnothing^* \{ndka\}$

$\mathcal{L}(M)$, wobei M eine TM mit nur 3 Zuständen ist

$\left\{\alpha 0 \alpha^R 0 \alpha \mid \alpha \in \{1\}^*\right\} \cap \left\{1\beta 1\beta 1 \mid \beta \in \{0,1\}^*\right\}$

Aufgabe 2: Umwandlung NDEA → DEA (8 Punkte)

Wandeln Sie den folgenden endlichen Automaten – gemäß dem Vorgehen aus der Vorlesung – in einen deterministischen endlichen Automaten um.



Aufgabe 3: Pumping Lemma **(12 Punkte)**

(a) Definition (4 Punkte)

Wie lautet das Pumping Lemma für kontextfreie Sprachen?

Sei L eine kontextfreie Sprache. Dann...

... $z = uvwxy$ mit den Eigenschaften

(1) , (2) und (3) .

(b) Anwendung (8 Punkte)

Beweisen Sie, dass $L := \{a^i b^j c^k d^j \mid i, j \geq 0, k = i \cdot j\}$ nicht kontextfrei ist.

Aufgabe 4: Rechnende Turingmaschine **(10 Punkte)**

Für eine Zahl $n \in \mathbb{N}$ sei $\mathbb{U}_a(n)$ ihre Unärkodierung mittels Symbol a und $\mathbb{B}(n)$ ihre Binärkodierung. Seien $n, k \in \mathbb{N}_+$ gegeben durch den Eingabestring $\mathbb{U}_a(k)\square\mathbb{B}(n)$. Konstruieren Sie eine Turingmaschine, die $\mathbb{B}(f(n,k))$ für die folgende Funktion berechnet:

$$f(n,k) := \begin{cases} n \cdot (k \bmod 3) & \text{wenn } k \text{ ungerade ist,} \\ \mathsf{undef} & \text{sonst.} \end{cases}$$

Aufgabe 5: Berechenbarkeit **(12 Punkte)**

(a) Definition (4 Punkte)

Vervollständigen Sie die folgende Definition ohne den Begriff *berechnen* zu verwenden.

Eine Funktion f mit Wertebereich $\mathbb{N} \cup \{\mathsf{undef}\}$ ist berechenbar, wenn ...

(b) Beweis (8 Punkte)

Sei $k \geq 10$ konstant. Sei f_k die Funktion, die bei Eingabe einer Turingmaschine M mit Eingabealphabet Σ die Anzahl der Eingabewörter $w \in \Sigma^k$ ausgibt, auf denen M anhält.

Ist f_k berechenbar? Begründen Sie.

Aufgabe 6: ***NP***-Vollständigkeit (20 Punkte)

(a) Definition (8 Punkte)

Definieren Sie ***NP***-Vollständigkeit, ohne den Begriff „***NP***-schwer“ zu benutzen. Definieren Sie auch den Begriff der Reduktion. Sie können davon ausgehen, dass ***NP*** definiert ist.

(b) Reduktion (12 Punkte)

Wir sagen, dass eine Zweierpotenz 2^i in einer Zahl $z \in \mathbb{N}$ *vorkommt*, wenn die i-te Stelle der Binärdarstellung von z eine 1 ist, z.B. kommt 4 in 5 vor, aber nicht in 9.

Eine gute Hexe möchte eine Zahl $z \in \mathbb{N}$ herbeizaubern, die allen Leuten des Dorfes gefällt. Vorher hat jede Person beliebig viele Wünsche geäußert: Ein Wunsch ist dabei ein 3-Tupel, bestehend aus der wünschenden Person p, einem Zeichen aus $\{\oplus, \ominus\}$ und einer Zweierpotenz x. Ein Wunsch $(p, \oplus, x)$ ist genau dann erfüllt, wenn x in z vorkommt; ein Wunsch $(p, \ominus, x)$ ist genau dann erfüllt, wenn x nicht in z vorkommt.

ZAUBERZAHL

Gegeben: Eine Menge von Personen P, eine Menge von Wünschen W.

Gefragt: Kann die Hexe eine Zahl $z \in \mathbb{N}$ herbeizaubern, sodass für jede Person in P mindestens ein Wunsch erfüllt wird?

Zeigen Sie, dass ZAUBERZAHL ***NP***-schwer ist.

Definieren Sie das bei Ihrer Reduktion verwendete Problem:

Name:

Gegeben:

Gefragt:

Aufgabe 7: Fixed Parameter Tractability **(14 Punkte)**

(a) Definition (4 Punkte)

Wie lautet die Definition von ***FPT***?

Ein Entscheidungsproblem liegt in ***FPT*** in Bezug auf Parameter k, wenn es sich in Zeit $\left\{ \square\ \mathcal{O}\big(n^c \cdot h(k)\big) \quad \square\ \mathcal{O}\big(n^k \cdot h(c)\big) \quad \square\ \mathcal{O}\big(2^c \cdot h(n)\big) \quad \square\ \mathcal{O}\big(2^k \cdot h(n)\big) \right\}$ lösen lässt.

Dabei ist $c \in \mathbb{N}$ und h $\left\{ \square \text{ beliebig}, \square \text{ exponentiell}, \square \text{ polynomiell}, \square \text{ berechenbar} \right\}$.

(b) Kernelization (10 Punkte)

Benjamin Blümchen feiert seinen n-ten Geburtstag und natürlich haben seine Freunde ihm eine große Geburtstagstorte mit n Kerzen mitgebracht. Mit einem Luftstoß kann Benjamin beliebig viele Kerzen auspusten, vorausgesetzt sie stehen exakt hintereinander auf einer Geraden.

KerzenAusblasen

Gegeben: $n \in \mathbb{N}$ Kerzen mit Koordinaten in $\mathbb{N}^2$, $k \in \mathbb{N}$.

Gefragt: Gibt es eine Menge von $\leq k$ Geraden, sodass jede der n Kerzen auf mindestens einer der Geraden liegt?

Geben Sie einen Algorithmus an, der in Polynomialzeit entweder das Problem entscheidet oder eine äquivalente Instanz erzeugt, deren Größe ausschließlich von k abhängt.

Hinweis: Stellen Sie sich eine Gerade g durch k Punkte vor. Was muss für eine Lösung gelten, falls g nicht ausgewählt wird?

Fortsetzung Aufgabe 7:

Aufgabe 8: Randomisierte Algorithmen (16 Punkte)

(a) Definition (4 Punkte)

Wie lautet die Definition von ***PP*** ?

(b) Algorithmus (12 Punkte)

Gegeben eine Zahl $n \in \mathbb{N}_+$ und eine Blackbox B, die bei Eingabe eines Bitstrings aus $\{0,1\}^n$ in konstanter Zeit ein einzelnes Bit ausgibt. Es gibt zwei Arten von Blackboxen: Eine *konstante Box* gibt immer dasselbe Bit aus, wohingegen eine *balancierte Box* für genau die Hälfte der Eingaben 0 und für die andere Hälfte 1 ausgibt.

Geben Sie einen randomisierten Algorithmus an, der bei Eingabe n mit konstant vielen Blackboxanfragen entscheidet, ob B eine konstante Box ist. Auf *JA*-Instanzen soll Ihr Algorithmus immer korrekt antworten und auf *NEIN*-Instanzen eine Fehlerwahrscheinlichkeit $< 1/7$ aufweisen. Beweisen Sie dies!

Geben Sie zuerst die kleinstmögliche randomisierte Komplexitätsklasse an, der dieser Algorithmus zugeordnet werden kann:

Algorithmus ***und*** *Analyse:*

Fortsetzung Aufgabe 8:

Notizen: