Technische Universität Berlin, Berlin, 13.10.2016

Name: ..... **Matr.-Nr.:** .....

Wiederholung Schriftliche Prüfung: Berechenbarkeit und Komplexität

(Niedermeier/Chen/Froese/Sorge, Sommersemester 2016)

Einlesezeit:	15 Minuten
Bearbeitungszeit:	60 Minuten
Max. Punktezahl:	50 Punkte

1	**2**	**3**	**4**	**5**	**Σ**
(12)	(8)	(8)	(12)	(10)	(50)

Allgemeine Hinweise:

- Es sind keinerlei Hilfsmittel erlaubt.
- Benutzen Sie einen dokumentenechten Stift in der Farbe schwarz oder blau. Insbesondere also keinen Bleistift, sondern einen Kugelschreiber oder einen nicht löschbaren Füller.
- Beschriften Sie jedes Blatt mit Vor- und Nachnamen und Ihrer Matrikelnummer.
- **Falls es in der Aufgabenstellung nicht explizit ausgeschlossen wird, so sind alle Antworten zu begründen! Antworten ohne Begründung erhalten 0 Punkte.**

Viel Erfolg!

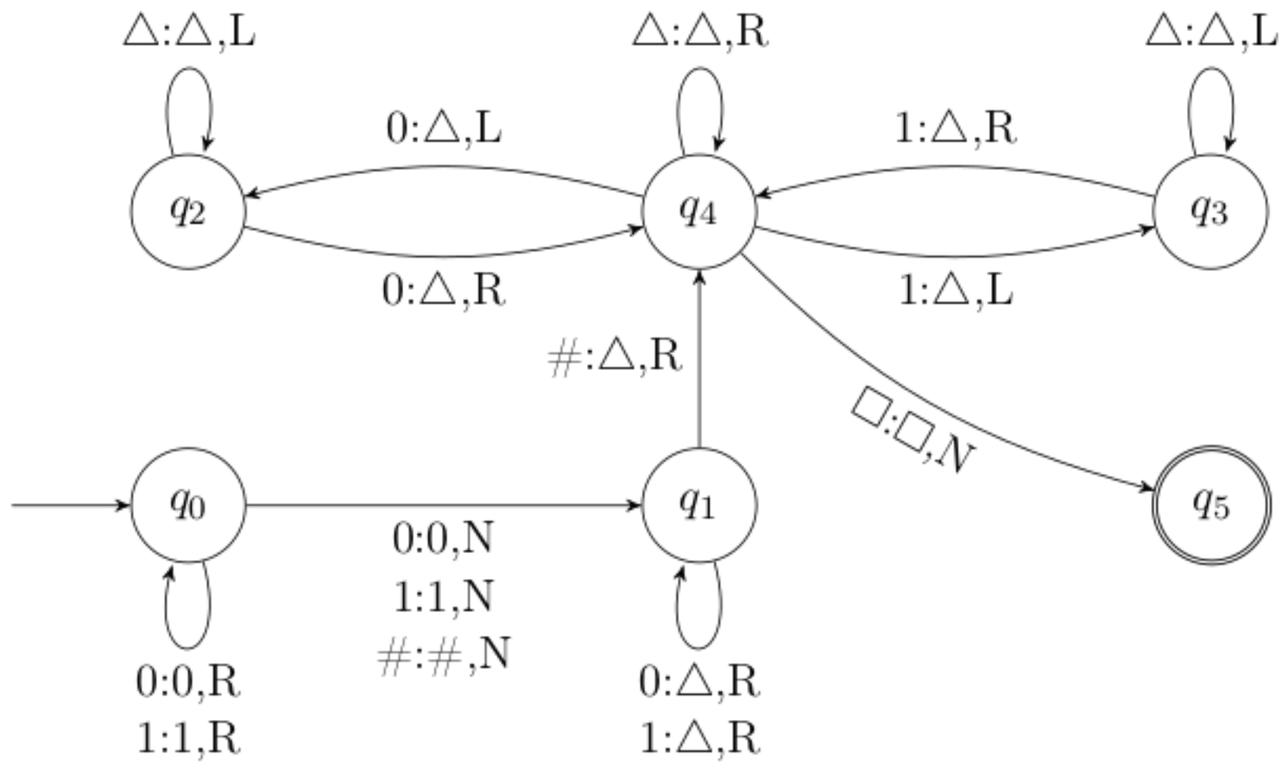
Name: ..... **Matr.-Nr.: .....**

Aufgabe 1: **Eine Turing-Maschine** (7 + 5 Punkte)

Betrachten Sie die Turing-Maschine

$$M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{0, 1, \#\}, \{0, 1, \#, \triangle, \square\}, \delta, q_0, \square, \{q_5\}),$$

wobei δ wie folgt definiert ist:



Hinweis: Beispielsweise bedeutet der Pfeil mit Beschriftung „#:△,R“ von q_1 zu q_4, dass M im Zustand q_1 beim Lesen von # in den Zustand q_4 übergeht, die # durch △ ersetzt, und ihren Leseschreibkopf nach rechts bewegt.

a) Die von der Turing-Maschine M akzeptierte Sprache $T(M)$ entspricht genau einer der folgenden Sprachen A, B, C. Welcher? Dabei genügt es, für jede nicht erkannte Sprache ein Gegenbeispiel und eine dazugehörige Begründung anzugeben. Hierbei ist $\mathrm{rev}\colon \{0,1\}^* \to \{0,1\}^*$ die Funktion, die jedem Wort $v \in \{0,1\}^*$ seine Umkehrung zuordnet. Beispielsweise ist $\mathrm{rev}(011) = 110$.

- $A = \{w\#v \mid (w, v \in \{0,1\}^*) \wedge \big(w = \mathrm{rev}(v)\big)\}$
- $B = \{w\#v \mid (w, v \in \{0,1\}^*) \wedge \big(\exists u \in \{0,1\}^* \colon w = u\,\mathrm{rev}(v)\big)\}$
- $C = \{w\#v \mid (w, v \in \{0,1\}^*) \wedge \big(\exists u_1, u_2 \in \{0,1\}^* \colon w = u_1\,\mathrm{rev}(v) u_2\big)\}$

b) Finden Sie ein Wort $w\#v$ mit $w \in \{0,1\}^2$ und $v \in \{0,1\}$ das von der Turing-Maschine M *nicht* akzeptiert wird und beweisen Sie, warum das so ist.

————————Lösungsskizze————————

a) Die akzeptierte Sprache ist C, denn das Wort 010#1 ist weder in A (in dem Teilwort vor # darf vor und nach der 1 kein anderes Symbol stehen) noch in B (in dem Teilwort vor # darf nach der 1 kein Symbol stehen) enthalten, wird aber von M akzeptiert:
$q_0010\#1 \vdash 0q_010\#1 \vdash 01q_00\#1 \vdash 01q_10\#1 \vdash 01\triangle q_1\#1 \vdash 01\triangle\triangle q_41 \vdash 01\triangle q_3\triangle\triangle \vdash 01q_3\triangle\triangle\triangle \vdash 0q_31\triangle\triangle\triangle \vdash 0\triangle q_4\triangle\triangle\triangle \vdash 0\triangle\triangle q_4\triangle\triangle \vdash 0\triangle\triangle\triangle q_4\triangle \vdash 0\triangle\triangle\triangle\triangle q_4\square \vdash 0\triangle\triangle\triangle\triangle q_5\square$

b) $11\#0$ wird von M nicht akzeptiert. Wir zeigen das indem wir alle möglichen Konfigurationsfolgen von M auf der Eingabe 101 untersuchen:

- $q_011\#0 \vdash 1q_01\#0 \vdash 11q_0\#0 \vdash 11q_1\#0 \vdash 11\triangle q_40 \vdash 11q_2\triangle\triangle \vdash 1q_21\triangle\triangle$
- $q_011\#0 \vdash 1q_01\#0 \vdash 1q_11\#0 \vdash 1\triangle q_1\#0 \vdash 1\triangle\triangle q_40 \vdash 1\triangle q_2\triangle\triangle \vdash 1q_2\triangle\triangle\triangle \vdash q_21\triangle\triangle\triangle$
- $q_011\#0 \vdash q_111\#0 \vdash \triangle q_11\#0 \vdash \triangle\triangle q_1\#0 \vdash \triangle\triangle\triangle q_40 \vdash \triangle\triangle q_2\triangle\triangle \vdash \triangle q_2\triangle\triangle\triangle \vdash q_2\triangle\triangle\triangle\triangle \vdash q_2\square\triangle\triangle\triangle\triangle$

Alternativ können wir $T(M) = C$ ausnutzen und per Widerspruchzeigen, dass $11\#0 \notin C$. Um in C zu sein setzen wir $v = 0$ so dass $w\#v = 11\#0$, dann muss laut C in w mindestens eine 0 vorkommen. Dies ist aber nicht der Fall. Also gilt $11\#0 \notin C$.

Name: ..... **Matr.-Nr.: .....**

Aufgabe 2: **Rekursive Aufzählbarkeit** (4+4 Punkte)

Zur Erinnerung: Eine Sprache $L \subseteq \Sigma^*$ heißt rekursiv aufzählbar, falls $L = \emptyset$ oder falls es eine totale, berechenbare Funktion $f\colon \mathbb{N} \to \Sigma^*$ derart gibt, dass $L = \{f(0), f(1), f(2), \ldots\} = f(\mathbb{N})$. Wir sagen auch: „$f$ zählt L auf."

Geben Sie für die folgenden Sprachen jeweils eine Funktion f an, die die entsprechende Sprache aufzählt, und begründen Sie die Korrektheit (inklusive Berechenbarkeit) von f.

a) $L_1 = \{a^p b^p \mid p \in \mathbb{N} \wedge (p \text{ ist eine Primzahl})\}$.
Hinweis: Sie können davon ausgehen, dass die $i$te Primzahl berechenbar ist.

b) $L_2 = A \cup B$, wobei $A, B \subseteq \Sigma^*$ zwei Sprachen sind, sodass $A \setminus B$, $B \setminus A$ und $A \cap B$ rekursiv aufzählbar sind.

——————Lösungsskizze——————

a) Folgende Funktion $f\colon \mathbb{N} \to \Sigma^*$ zählt L_1 auf: $f(i) = a^{p_i} b^{p_i}$, wobei p_i die $i+1$'te Primzahl ist, geordnet nach Größe. Jede Primzahl kommt in dieser Ordnung vor und damit ist $f(\mathbb{N}) = L_1$. Funktion f is berechenbar, da man nur die i+1'te Primzahl berechnen muss, und entsprechend viele a's und b's schreiben muss.

b) Da $A \setminus B, B \setminus A, A \cap B$ rekursiv aufzählbar sind, existieren entsprechende aufzählende Funktionen f_1, f_2, f_3. Folgende Funktion $f : \mathbb{N} \to \Sigma^*$ zählt $A \cup B$ auf:

$$f(i) = \begin{cases} f_1(\lfloor i/3 \rfloor), & i \equiv 0 \pmod 3 \\ f_2(\lfloor i/3 \rfloor), & i \equiv 1 \pmod 3 \\ f_3(\lfloor i/3 \rfloor), & i \equiv 2 \pmod 3 \end{cases}$$

Da f nur aus arithmetischen Operationen angewendet auf berechenbare Funktionen besteht, ist f berechenbar. Offensichtlich is $f(\mathbb{N}) \subseteq A \cup B$. Außerdem gilt $\forall i \in \{1,2,3\}\colon f_i(\mathbb{N}) \subseteq f(\{3j + (i-1) \mid j \in \mathbb{N}\})$ und damit $A \setminus B, B \setminus A, A \cap B \subseteq f(\mathbb{N})$, also $f(\mathbb{N}) = A \cup B$, da $A \cup B = (A \setminus B) \uplus (B \setminus A) \uplus (A \cap B)$.

Name: ..... **Matr.-Nr.: .....**

Aufgabe 3: **Reduzierbarkeit und Halteprobleme** (4+4 Punkte)

Gegeben seien zwei Sprachen $A \subseteq \Sigma^*$ und $B \subseteq \Gamma^*$. Begründen oder widerlegen Sie die folgenden Aussagen.

a) Wenn $A \leq B$, dann $(\Sigma^* \setminus A) \leq (\Gamma^* \setminus B)$.

b) Wenn $A \leq B$, dann $(\Gamma^* \setminus B) \leq (\Sigma^* \setminus A)$.

Hinweis: $A \leq B$ bedeutet, dass A reduzierbar auf B ist. Nutzen Sie die Eigenschaften von Reduktionen und den Fakt, dass zum Beispiel das spezielle Halteproblem K,

$$K = \{w \mid \text{die durch } w \text{ kodierte Turing-Maschine } M_w \text{ hält auf Eingabe } w\},$$

unentscheidbar ist.

————————Lösungsskizze————————

a) Die Aussage ist wahr, denn

$$\begin{aligned} A \leq B &\Leftrightarrow (\exists \text{ eine totale, berechenbare Funktion } f : \Sigma^* \to \Gamma^* \text{ so dass } \forall x \in \Sigma^* : x \in A \Leftrightarrow f(x) \in B) \\ &\Leftrightarrow (\exists \text{ eine totale, berechenbare Funktion } f : \Sigma^* \to \Gamma^* \forall x \in \Sigma^* : (x \in \Sigma^* \setminus A) \Leftrightarrow (f(x) \in \Gamma^* \setminus B)) \\ &\Leftrightarrow (\Sigma^* \setminus A) \leq (\Gamma^* \setminus B). \end{aligned}$$

b) Die Aussage ist falsch. Ein Gegenbeispiel ergibt sich mit $B = K$ und A einer beliebigen Sprache, die nur ein Wort $v \in \Sigma^*$ enthält: Es gilt $A \leq B$. Eine entsprechende Reduktionsfunktion prüft, ob die Eingabe v ist und gibt in dem Fall ein beliebiges, fest kodiertes $w \in K$ aus und ansonsten eine beliebiges, fest kodiertes $w' \notin K$. Allerdings gilt nicht $(\Gamma^* \setminus B) \leq (\Sigma^* \setminus A)$: Ansonsten wäre $\Gamma^* \setminus B$ entscheidbar, und damit auch K entscheidbar, weil offenbar $\Sigma^* \setminus A$ entscheidbar ist.

Name: ..... **Matr.-Nr.:** .....

Aufgabe 4: **NP-Vollständigkeit und Polynomzeitreduktion** (3 + 6 + 3 Punkte)

Eine *Bipartition* einer Menge V ist ein Tupel (A, B) mit $A, B \subseteq V$, $A \cap B = \emptyset$ und $A \cup B = V$. Sei $G = (V, E)$ ein Graph und (A, B) eine Bipartition seiner Knotenmenge V. Wir sagen, dass eine Kante $\{u, v\} \in E$ die Bipartition (A, B) *kreuzt*, wenn $u \in A$ und $v \in B$, oder $u \in B$ und $v \in A$. Beispielsweise wird die Bipartition $(\{b\}, \{a, c, d\})$ der Knotenmenge des in Teilaufgabe a) dargestellten Graphen von drei Kanten gekreuzt.

Betrachten Sie folgende Probleme:

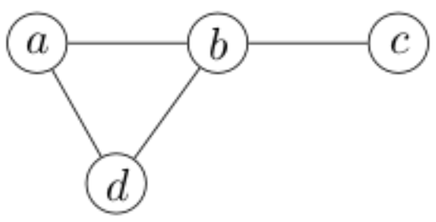
Maximum Cut
Eingabe: Ein ungerichteter Graph $G = (V, E)$ und eine natürliche Zahl $k \in \mathbb{N}$.
Frage: Existiert eine Bipartition der Knotenmenge V, die von *mindestens* k Kanten gekreuzt wird?

Minimum Bisection
Eingabe: Ein ungerichteter Graph $G = (V, E)$ und eine natürliche Zahl $\ell \in \mathbb{N}$.
Frage: Existiert eine Bipartition (A, B) der Knotenmenge V mit $|A| = |B|$, die von *höchstens* ℓ Kanten gekreuzt wird?

Betrachten Sie folgende Transformation f einer beliebigen Instanz $\langle G = (V, E), k\rangle$ von Maximum Cut in eine Instanz $f(\langle G, k\rangle) = \langle H, \ell\rangle$ von Minimum Bisection:

- Die Knotenmenge von H setzt aus der Knotenmenge V von G und einer neuen Knotenmenge V' zusammen, wobei $|V'| = |V|$.
- Die Kantenmenge von H enthält die Menge $E' = \{\{u, v\} \mid (u, v \in V) \wedge (\{u, v\} \notin E)\}$ aller Kanten, die *nicht* in G enthalten sind, sowie die Kantenmenge $E'' = \{\{u, v\} \mid (u \in V \cup V') \wedge (v \in V')\}$, die jeden neuen Knoten in V' mit allen anderen Knoten verbindet.
- Zusammengefasst ist $H \coloneqq (V \cup V', E' \cup E'')$.
- Setze $\ell \coloneqq |V|^2 - k$.

a) Geben Sie $f(\langle G, k\rangle) = \langle H, \ell\rangle$ an, wobei $k = 3$ und G wie folgt definiert ist:

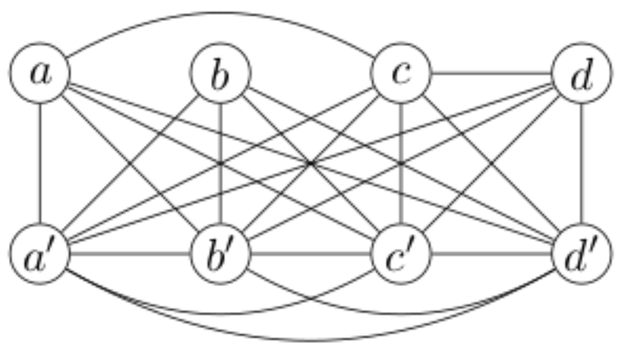


Hinweis: Für den Graphen H in $f(\langle G, k\rangle)$ genügt eine bildliche Darstellung.

b) Beweisen Sie, dass obige Transformation f eine Polynomzeitreduktion von Maximum Cut auf Minimum Bisection ist.

c) Maximum Cut ist NP-schwer. Beweisen Sie, dass Minimum Bisection NP-vollständig ist.

——————Lösungsskizze——————



a) $\langle H, 13\rangle$, wobei H wie folgt definiert ist:

b) Die Transformation ist in Polynomialzeit berechenbar, da sich die Knotenmenge offensichtlich in $O(|V|)$ Schritten berechnen lässt, und die Kantenmenge in $O(|V|^2)$ Schritten indem man über alle Knotenpaare geht. Die Korrektheit sieht man wie folgt:

Sei (A, B) eine Lösung von Maximum Cut, d.h. (A, B) wird in G von mindestens k Kanten gekreuzt. Definiere eine Bipartition $(A \cup A', B \cup B')$ von $V \cup V'$, wobei (A', B') eine beliebige Bipartition von V' mit $|A'| = |B|$. Damit ist $|A \cup A'| = |B \cup B'|$, und außerdem wird $(A \cup A', B \cup B')$ (in H) genau von denjenigen Kanten nicht gekreuzt, die (A, B) in G kreuzen. Also kreuzen $(A \cup A', B \cup B')$ höchstens $|V|^2 - k$ viele Kanten.

Sei nun (A, B) Lösung von Minimum Bisection, d.h. (A, B) wird in H von höchstens $|V|^2 - k$ Kanten gekreuzt. Dann verläuft jede der mindestens k „Kanten“ (Knotenpaare), die (A, B) nicht kreuzen, zwischen $A \cap V$ und $B \cap V$. Also wird $(A \cap V, B \cap V)$ in G von mindestens k Kanten gekreuzt.

c) Wegen der Polynomialzeitreduktion aus Teilaufgabe b) ist Minimum Bisection NP-schwer. Um zu sehen, dass Minimum Bisection in NP liegt, genügt es, ein Zertifikat anzugeben, sodass jede Ja-Instanz in Polynomialzeit als solche verifiziert werden kann. Ein solches Zertifikat ist die Bipartition (A, B) von V: Zur Verifikation genügt es, für jede Kante zu überprüfen, ob sie (A, B) kreuzt. Dies gescheht in Polynomzeit.

Name: ..... **Matr.-Nr.: .....**

Aufgabe 5: **Vermischtes zu P und NP** (4 + 3 + 3 Punkte)

Beweisen oder widerlegen Sie folgende Aussagen.

a) Eine Sprache $L \subseteq \Sigma^*$ ist in NP, wenn sie die folgende Eigenschaft erfüllt: Es existieren ein Polynom $p\colon \mathbb{N} \to \mathbb{N}$ und eine *deterministische* Turing-Maschine M mit $\mathsf{time}_M(n) \in n^{O(1)}$, sodass für jedes Wort $x \in \Sigma^*$ gilt

$$(x \in L) \Leftrightarrow \big(\exists u \in \Sigma^{p(|x|)} \colon \langle x, u\rangle \in T(M)\big).$$

Hierbei ist $T(M)$ die von der Turing-Maschine M akzeptierte Sprache.

——————Lösungsskizze——————

Wahr. Wir konstruieren eine NTM M' aus M, die in Polynomialzeit L entscheidet. Beim Eingabewort $x \in \Sigma^*$ erzeugt M' zuerst nichtdeterministisch ein Wort $u \in \Sigma^{p(|x|)}$ und lässt M bei der Eingabe $\langle x, u\rangle$ laufen. M' gibt die Antwort von M zurück. Offensichtlich entscheidet M' die Sprache L. Sie arbeitet insgesamt auch in Polynomialzeit. Also gilt $L \in$ NP.

---

b) Gegeben seien zwei Sprachen $A, B \subseteq \Sigma^*$ mit $A, B \notin \{\emptyset, \Sigma^*\}$. Falls $A, B \in$ NP, dann ist A reduzierbar auf B.

——————Lösungsskizze——————

Ja. Da $A \in$ NP, existiert eine TM M, die A entscheidet. Folgender Algorithmus reduziert von A auf B:

- Lass M auf w laufen.
- Falls M das Wort w akzeptiert, dann gebe ein Wort $w' \in B$ zurück.
- Falls M das Wort w nicht akzeptiert, dann gebe ein Wort $w' \in \Sigma^* \setminus B$ zurück.

---

c) Falls eine Sprache $L \subseteq \Sigma^*$ in P ist, dann ist auch $(\Sigma^* \setminus L)$ in P.

——————Lösungsskizze——————

Ja. Wenn $L \in$ P, dann existiert eine Turing-Maschine M, die L in Polynomialzeit entscheidet. Die Turing-Maschine M', die M simuliert und akzeptiert, genau dann wenn M ablehnt, entscheidet $\Sigma^* \setminus L$. Offensichtlich gilt $\mathrm{time}_{M'}(n) \in n^{O(1)}$. Da M deterministisch ist, ist M' auch deterministisch.
