**2. Klausur zur Vorlesung**
**Theoretische Grundlagen der Informatik**
**Wintersemester 2017/2018**

# Lösung!

**Beachten Sie:**

- Bringen Sie den Aufkleber mit Ihrem Namen und Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes Aufgabenblatt mit Ihrem Namen und Matrikelnummer.
- Schreiben Sie die Lösungen auf die Aufgabenblätter und Rückseiten. Zusätzliches Papier erhalten Sie bei Bedarf von der Aufsicht.
- Es sind keine Hilfsmittel zugelassen.

| | Mögliche Punkte | | | | | Erreichte Punkte | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | a | b | c | d | $\Sigma$ | a | b | c | d | $\Sigma$ |
| Aufg. 1 | 3 | 5 | – | – | 8 | | | – | – | |
| Aufg. 2 | 4 | 2 | 4 | – | 10 | | | | – | |
| Aufg. 3 | 2 | 2 | 3 | – | 7 | | | | – | |
| Aufg. 4 | 2 | 3 | 3 | 2 | 10 | | | | | |
| Aufg. 5 | 6 | 3 | – | – | 9 | | | – | – | |
| Aufg. 6 | 2 | 6 | 2 | – | 10 | | | | – | |
| Aufg. 7 | $6 \times 1$ | | | | 6 | | | | | |
| $\Sigma$ | | | | | 60 | | | | | |

**Problem 1:** Reguläre Sprachen $3 + 5 = 8$ Punkte

(a) Geben Sie einen vollständigen deterministischen endlichen Automaten an, der die Sprache

$$L = \{w \in \{0,1\}^* \mid w \text{ endet mit } 101\}$$

erkennt. Vervollständigen Sie dazu folgende Skizze ohne weitere Zustände zu verwenden.

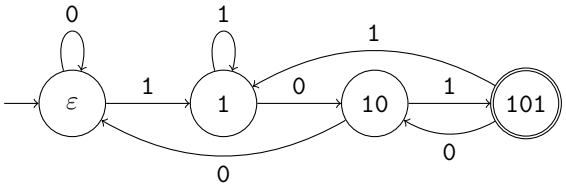

(b) Zeigen Sie, dass die Sprache

$$L = \{w \in \{0,1\}^* \mid w \text{ enthält in der ersten Hälfte } 101\}$$

nicht regulär ist. Die erste Hälfte eines Wortes $w$ ist das Teilwort mit den ersten $\lceil |w|/2 \rceil$ Zeichen.

*Lösung:*

(a) Betrachte z.B. folgenden Automaten:




(b) Wäre $L$ regulär, dann gäbe es nach dem Pumping-Lemma ein $n \in \mathbb{N}$, sodass für jedes $w \in L$ mit $|w| > n$ gilt: Es existiert eine Zerlegung $w = uvx$ mit $|uv| \leq n$ und $v \neq \epsilon$, sodass für alle $i \in \mathbb{N}_0$ auch $uv^i x \in L$ gilt.

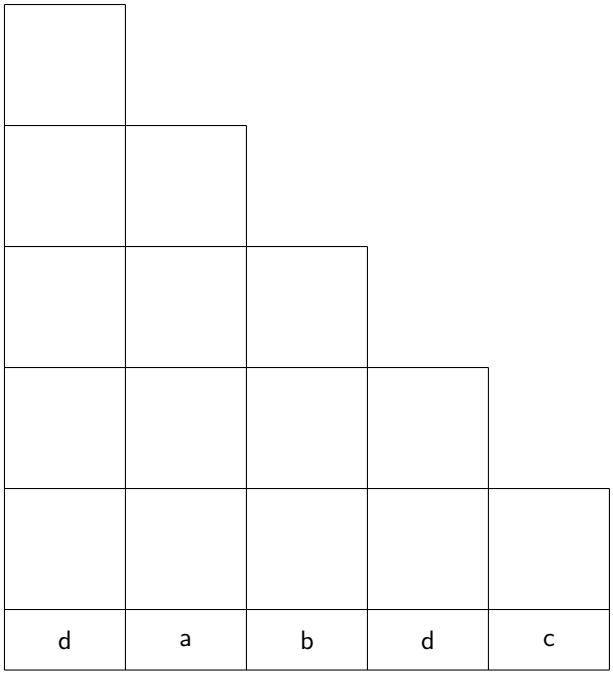
Gegeben dieses $n$, betrachte das Wort $w = 0^n 101\, 0^{n+3}$ der Länge $2n+6$. Offensichtlich gilt $w \in L$. Für jede Zerlegung $w = uvx$ mit den obigen Bedingungen gilt $v = 0^j$ für ein $1 \leq j \leq n$. Dann gilt $uv^i x = 0^{n+(i-1)j} 101\, 0^{n+3}$. Für $i \geq 3$ ist $uv^i x \notin L$. Widerspruch zum Pumping-Lemma, also ist $L$ nicht regulär.

**Problem 2:** Kontextfreie Grammatiken $4 + 2 + 4 = 10$ Punkte

Gegeben sei die Grammatik $G = (\Sigma, V, S, R)$ mit Terminalen $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$, Nichtterminalen $V = \{S, A, B, C, D\}$, Startsymbol $S$ und Produktionen

$$\begin{aligned} R = \{ & S \to CB \\ & A \to CD \mid \mathsf{a} \mid \mathsf{d} \\ & B \to AC \mid \mathsf{b} \\ & C \to DA \mid \mathsf{b} \mid \mathsf{c} \\ & D \to AB \mid \mathsf{d} \end{aligned}$$

(a) Überprüfen Sie mit dem CYK-Algorithmus, ob das Wort dabdc in der Sprache $L(G)$ enthalten ist.

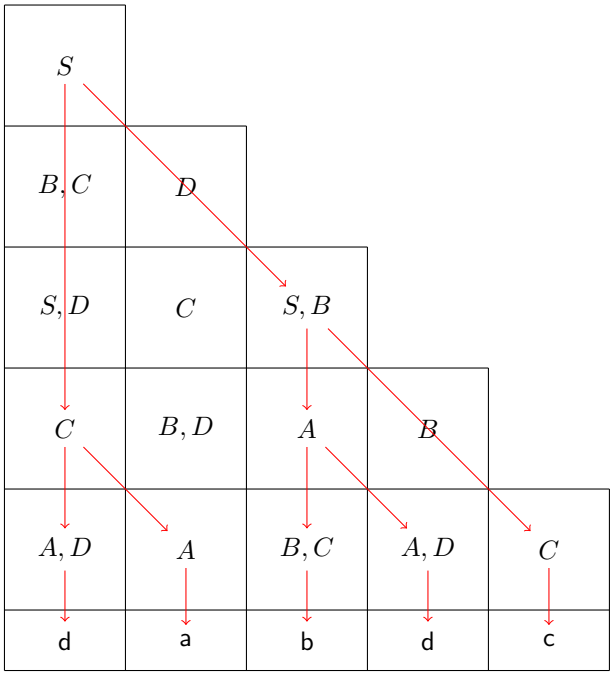



(b) Geben Sie einen Ableitungsbaum für das Wort dabdc an.

(c) Erweitern Sie den CYK-Algorithmus so, dass falls das überprüfte Wort tatsächlich von der Grammatik erzeugt wird, ein Ableitungsbaum ausgegeben wird.

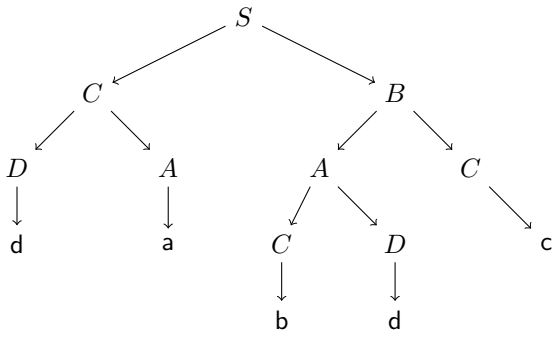
*Lösung:*

(a) Vergleiche folgende Tabelle:




Das Wort ist also in der Sprache enthalten.

(b) Vergleiche folgenden Ableitungsbaum:




(c) Zu jedem Symbol werden außerdem die ein oder zwei Vorgängersymbole gespeichert. In der obigen Tabelle sind diese Vorgänger durch rote Pfeile markiert. Wird das Wort von der Grammatik erzeugt, kann der Ableitungsbaum über die Vorgängersymbole rekursiv ausgegeben werden.

**Problem 3:** Entscheidbarkeit $2 + 2 + 3 = 7$ Punkte

(a) Erläutern Sie in eigenen Worten die Begriffe *entscheidbar* und *rekursiv aufzählbar*.

Sei nun $L$ eine nicht-endliche Sprache über einem endlichen Alphabet $\Sigma$.

(b) Zeigen Sie: Wenn $L$ entscheidbar ist, existiert eine berechenbare Bijektion zwischen $\Sigma^*$ und $L$.

(c) Zeigen Sie, dass es eine nicht-entscheidbare Sprache $L' \subseteq L$ gibt. Verwenden Sie nicht den Satz von Rice.

*Hinweis: Unterscheiden Sie zunächst die zwei Fälle, ob $L$ entscheidbar ist oder nicht.*

*Lösung:*

(a) Eine Sprache $L$ ist entscheidbar, wenn es eine Turing-Maschine gibt, die für jede Eingabe hält und eine Eingabe genau dann akzeptiert, wenn sie in $L$ ist.

Eine Sprache $L$ ist rekursiv aufzählbar, wenn es eine Turing-Maschine gibt, die genau die Wörter in $L$ (in beliebiger Reihenfolge) nacheinander auf das Band schreibt.

(b) Da $L$ entscheidbar ist, ist es auch rekursiv aufzählbar. Dann existiert eine berechenbare Bijektion $f : L \to \mathbb{N}$, die jedem Wort aus $L$ seinen Rang zuordnet und dadurch eine totale Ordnung von $L$ beschreibt. Auch für $\Sigma^*$ existiert eine berechenbare Bijektion $g : \Sigma^* \to \mathbb{N}$, die jedem Wort aus $\Sigma^*$ seinen Rang zuordnet. Mit diesen beiden Bijektionen können wir eine dritte berechenbare Bijektion $h : \Sigma^* \to L$ eindeutig dadurch definieren, dass $g(w) = f(h(w))$ gilt.

(c) Falls $L$ nicht entscheidbar ist, gilt die Behauptung trivialerweise.

Nehme also an, dass $L$ entscheidbar ist. Nach (b) existiert eine berechenbare Bijektion $h : \Sigma^* \to L$. Für jede beliebige nicht-entscheidbare Sprache $L''$ ist dann auch die Sprache $L' = \{h(w) \mid w \in L''\} \subseteq L$ nicht-entscheidbar. Wäre nämlich $L'$ entscheidbar, so wäre auch $L''$ entscheidbar, weil $h^{-1}$ berechenbar ist.

## **Problem 4:** Kellerautomaten $2 + 3 + 3 + 2 = 10$ Punkte

In Vorlesung und Übung wurden typischerweise *nichtdeterministische* Kellerautomaten betrachtet. Sei $\mathcal{A}$ nun ein *deterministischer* Kellerautomat, d.h. es gibt keine $\varepsilon$-Übergänge und für jedes Eingabesymbol ist der Übergang von $\mathcal{A}$ eindeutig definiert. Gehen Sie davon aus, dass das Eingabealphabet $\Sigma$ mindestens zwei Symbole enthält, dass $\mathcal{A}$ durch akzeptierende Zustände akzeptiert und dass der Stack von $\mathcal{A}$ niemals leer ist.

Definiere für solche deterministischen Kellerautomaten das Konzept der *Essenz* folgendermaßen. Für jedes Wort $w$ existiert ein[1] Wort $w'$, sodass der Stack nach Verarbeitung von $ww'$ für alle möglichen $w'$ die minimale Höhe hat. Ist nach Abarbeitung von $ww'$ die Höhe des Stacks $h$, so ist also für jedes Wort $v$ die Höhe des Stacks nach Abarbeitung von $ww'v$ mindestens $h$. Sei $\alpha_1\alpha_2\ldots\alpha_h$ der Inhalt des Stacks nach Abarbeitung von $ww'$. Das oberste Symbol auf dem Stack ist $\hat{\alpha} = \alpha_h$. Der aktuelle Zustand sei $q$. Bezeichne $(q, \hat{\alpha})$ als die *Essenz* von $w$.

(a) Zeigen Sie, dass es mindestens zwei unterschiedliche Wörter $w_1 \neq w_2$ mit der gleichen Länge und derselben Essenz gibt.

*Hinweis: Wie viele unterschiedliche Essenzen kann es höchstens geben?*

Erinnern Sie sich daran, dass die Konfiguration $(q, v, \alpha)$ eines Kellerautomaten aus den drei Teilen Zustand $q$, Teil der Eingabe $v \in \Sigma^*$, der noch nicht gelesen wurde, und dem Stackinhalt $\alpha \in \Gamma^*$ besteht.

(b) Seien $w_1, w_2$ Wörter mit derselben Essenz $(q, \hat{\alpha})$. Zeigen Sie, dass dann für jedes Wort $v \in \Sigma^*$ gilt

$$w_1 w_1' v \in L(\mathcal{A}) \iff w_2 w_2' v \in L(\mathcal{A}).$$

[1] Das Wort $w'$ ist nicht notwendigerweise eindeutig.

(c) Definieren Sie die Sprache, die genau aus allen Palindromen besteht. Nutzen Sie das Ergebnis der letzten Teilaufgabe, um zu zeigen, dass $\mathcal{A}$ die Palindromsprache nicht erkennen kann.

(d) Bestimmen Sie das minimale $k$, sodass deterministische Kellerautomaten alle Sprachen von Typ-$k$ erkennen können. Begründen Sie kurz.

*Lösung:*

(a) Es gilt $\hat{\alpha} \in \Gamma$ und $q \in Q$. Damit kann es höchstens $|\Gamma \times Q|$, also endlich viele Essenzen geben. Da $|\Sigma^n| \geq 2^n$ ist, gibt es ein $n_0 > \lceil \log(|\Gamma \times Q|) \rceil$, sodass mehr Wörter der Länge $n_0$ als Essenzen existieren. Also muss es mindestens zwei unterschiedliche Wörter mit gleicher Länge und derselben Essenz geben.

(b) Betrachte für $i \in \{1, 2\}$ die Konfigurationen $(q_i, w_i, \alpha_i)$ von $\mathcal{A}$ nach der Abarbeitung von $w_i w_i'$. Da $w_1$ und $w_2$ dieselbe Essenz haben, gilt $q_1 = q_2$. Außerdem ist das oberste Symbol auf dem Stack $\hat{\alpha}$. Der Teil der Eingabe, der noch nicht gelesen wurde, ist in beiden Fällen $v$. Der einzige Unterschied zwischen den Konfigurationen kann also im Stackinhalt außer dem obersten Stacksymbol $\hat{\alpha}$ liegen. Da der Stack aber nach Definition der Essenz bei der weiteren Abarbeitung nicht schrumpfen kann, kann der Inhalt des Stacks bis auf das oberste Symbol nicht gelesen, geschrieben oder geändert werden.

Damit ist das Verhalten (insbesondere das Akzeptanzverhalten) von $\mathcal{A}$ nach dem Lesen von $w_i w_i'$ identisch für jedes $v \in \Sigma^*$.

(c) Die Palindromsprache ist $L_P = \{w \in \Sigma^\star \mid w = w^R\}$, wobei $w^R$ die Umkehrung von $w$ sei. Nach Teil (a) gibt es Wörter $w_1 \neq w_2$ mit gleicher Länge und derselben Essenz. Nach Teil (b) gilt dann

$$w_1 w_1' (w_1 w_1')^{\mathsf{R}} \in L(\mathcal{A}) \iff w_2 w_2' (w_1 w_1')^{\mathsf{R}} \in L(\mathcal{A}).$$

Da $w_1 w_1' (w_1 w_1')^{\mathsf{R}}$ ein Palindrom ist, wegen $w_1 \neq w_2$ das Wort $w_2 w_2' (w_1 w_1')^{\mathsf{R}}$ jedoch nicht, kann $\mathcal{A}$ nicht die Palindromsprache erkennen.

(d) Die Palindromsprache ist kontextfrei. Deterministische Kellerautomaten können also nicht die Typ-2-Sprachen erkennen. Deterministische Kellerautomaten können aber trivialerweise deterministische endliche Automaten simulieren. Deterministische Kellerautomaten können also die Typ-3-Sprachen erkennen.

## **Problem 5:** NP-Vollständigkeit 6 + 3 = 9 Punkte

Das Entscheidungsproblem FREQUENCY ALLOCATION ist wie folgt definiert.

| | |
|---|---|
| **Input:** | endliche Menge $T$ von Transmittern<br>für jeden Transmitter $t \in T$ endliche Menge $F_t \subseteq \mathbb{N}$ von möglichen Frequenzen<br>eine Menge von Paaren von Transmittern, die interferieren |
| **Lösung:** | für jeden Transmitter $t \in T$ eine Frequenz $f_t \in F_t$,<br>sodass je zwei interferierende Transmitter unterschiedliche Frenquenzen erhalten,<br>d.h. $f_{t_1} \neq f_{t_2}$ wenn $t_1$ und $t_2$ interferieren |

Das Optimierungsproblem MAXIMUM FREQUENCY ALLOCATION ist wie folgt definiert.

| | |
|---|---|
| **Input:** | endliche Menge $T$ von Transmittern<br>für jeden Transmitter $t \in T$ endliche Menge $F_t \subseteq \mathbb{N}$ von möglichen Frequenzen<br>eine Menge von Paaren von Transmittern, die interferieren |
| **Lösung:** | eine Teilmenge $S \subseteq T$ und für jeden Transmitter $t \in S$ eine Frequenz $f_t \in F_t$,<br>sodass je zwei interferierende Transmitter unterschiedliche Frenquenzen erhalten,<br>d.h. $f_{t_1} \neq f_{t_2}$ wenn $t_1$ und $t_2$ interferieren |
| **Ziel:** | maximiere $\lvert S \rvert$ über alle Lösungen $S$ |

(a) Ist das Problem FREQUENCY ALLOCATION NP-vollständig? Beweisen Sie!

*Lösung:* Wir zeigen, dass FREQUENCY ALLOCATION NP-vollständig ist. Zunächst ist FREQUENCY ALLOCATION in NP, da eine Lösung in polynomieller Zeit geraten und verifiziert werden kann. Um zu zeigen, dass FREQUENCY ALLOCATION auch NP-schwer ist, reduzieren wir das NP-schwere Problem COLOR auf FREQUENCY ALLOCATION.

Sei $G, k$ eine Instanz von COLOR, d.h. $G = (V, E)$ ist ein Graph und $k \in \mathbb{N}$ ist eine natürliche Zahl. Wir konstruieren in polynomieller Zeit eine Instanz von FREQUENCY ALLOCATION wie folgt:

- $T = V$, d.h. die Transmitter entsprechen den Knoten von $G$
- $F_t = \{1, \dots, k\}$, d.h. jeder Transmitter hat die gleichen $k$ möglichen Frequenzen
- zwei Transmitter $t_1, t_2 \in T$ interferieren genau dann, wenn $\{t_1, t_2\}$ eine Kante in $G$ bilden

Wir behaupten nun, dass die Instanz $G, k$ von COLOR genau dann lösbar ist, wenn die zugehörige Instanz von FREQUENCY ALLOCATION lösbar ist.

$\Rightarrow$: Sei $V_1, \dots, V_k$ eine Lösung von COLOR, d.h. $V_1 \cup \dots \cup V_k = V$ und je zwei Knoten im gleichen $V_i$ sind nicht benachbart in $G$. Für $i = 1, \dots, k$ weisen wir einem Transmitter $t \in T$ die Frequenz $i$ zu, wenn $t \in V_i$. Dann gilt $f_t = i \in \{1, \dots, k\} = F_t$. Außerdem gilt $f_{t_1} = f_{t_2} = i$ nur, wenn $t_1, t_2 \in V_i$. Dann ist aber $\{t_1, t_2\}$ keine Kante von $G$ und daher interferieren $t_1$ und $t_2$ nicht. Also ist diese Frequenzzuweisung eine Lösung von FREQUENCY ALLOCATION.

$\Leftarrow$: Sei gegeben eine Frequenzzuweisung $f_t \in F_t$ für jeden Transmitter $t \in T$, sodass keinen zwei interferierenden Transmittern die gleiche Frequenz zugewiesen wird. Für $i = 1, \dots, k$ definieren wir $V_i = \{t \in V : f_t = i\}$ als die Menge der Knoten in $V$, denen als Transmitter die Frequenz $i$ zugewiesen ist. Dann gilt $V_1 \cup \dots \cup V_k = V$, da $F_t = \{1, \dots, k\}$ für jeden Transmitter $t$. Außerdem sind zwei Knoten im gleichen $V_i$ nicht benachbart, da die zugehörigen Transmitter die gleiche Frequenz erhalten und daher nicht interferieren. Also ist diese Aufteilung der Knotenmenge eine Lösung von COLOR.

Da FREQUENCY ALLOCATION in NP liegt und COLOR NP-schwer ist, folgt, dass FREQUENCY ALLOCATION NP-vollständig ist.

(b) Ist bekannt, ob das Problem MAXIMUM FREQUENCY ALLOCATION in polynomieller Laufzeit gelöst werden kann? Argumentieren Sie!

*Lösung:* Wenn MAXIMUM FREQUENCY ALLOCATION in polynomieller Laufzeit gelöst werden könnte, so könnte auch FREQUENCY ALLOCATION in polynomieller Laufzeit gelöst werden. Dazu genügt für eine optimale Lösung von MAXIMUM FREQUENCY ALLOCATION mit Wert $k$ ein Test, ob $k = |T|$, also ob allen Transmittern eine Frequenz zugeordnet werden konnte.

Da FREQUENCY ALLOCATION nach Aufgabenteil (a) NP-vollständig ist, würde dann folgen, dass jedes Problem in NP in polynomieller Zeit gelöst werden könnte. Es würde dann also P = NP folgen. Da nicht bekannt ist, ob P = NP, kann auch nicht bekannt sein, ob MAXIMUM FREQUENCY ALLOCATION in polynomieller Laufzeit gelöst werden kann.

Jedes der folgenden Entscheidungsprobleme ist NP-vollständig.

TRAVELING SALESMAN

**Input:** Graph $G = (V, E)$
Längenfunktion $\ell : E \to \mathbb{R}$
Zahl $k \in \mathbb{N}$
**Lösung:** geschlossene Tour $T$ durch alle Knoten von $G$ mit Gesamtlänge höchstens $k$,
d.h. $\sum_{e \in T} \ell(e) \leq k$.

HAMILTONPFAD

**Input:** Graph $G = (V, E)$
**Lösung:** Pfad $P$ durch alle Knoten von $G$

CLIQUE

**Input:** Graph $G = (V, E)$
Zahl $k \in \mathbb{N}$
**Lösung:** Menge $U \subseteq V$ von mindestens $k$ Knoten, in der je zwei Knoten benachbart sind,
d.h. $|U| \geq k$ und für alle $u, v \in U$ mit $u \neq v$ gilt $\{u, v\} \in E$.

COLOR

**Input:** Graph $G = (V, E)$
Zahl $k \in \mathbb{N}$
**Lösung:** Knotenfärbung mit höchstens $k$ Farben, in der keine gleichfarbigen Knoten benachbart sind,
d.h. $V = V_1 \cup \cdots \cup V_k$ und für $i = 1, \ldots, k$ und alle $u, v \in V_i$ gilt $\{u, v\} \notin E$.

## **Problem 6:** Approximationsalgorithmen $2 + 6 + 2 = 10$ Punkte

Das Optimierungsproblem MAXIMUM CUT ist wie folgt definiert.

| | |
|---|---|
| **Input:** | Graph $G = (V, E)$ |
| **Lösung:** | Zwei-Färbung der Knoten in rot und blau, d.h.<br>jedes $v \in V$ bekommt genau eine Farbe in $\{\text{rot}, \text{blau}\}$ |
| **Ziel:** | maximiere die Anzahl der Kanten zwischen roten und blauen Knoten, d.h.<br>maximiere $\#\{uv \in E\colon (u \text{ rot und } v \text{ blau}) \vee (u \text{ blau und } v \text{ rot})\}$ |

(a) Zeigen Sie, dass in jeder optimalen Lösung von MAXIMUM CUT für jeden roten Knoten $v$ mindestens die Hälfte aller Nachbarn von $v$ blau sind.

*Lösung:* Angenommen der rote Knoten $v$ hat $k$ rote Nachbarn und $\ell$ blaue Nachbarn. Dann sind genau $\ell$ von den zu $v$ inzidenten Kanten im Schnitt. Nach Umfärben von $v$ zu blau sind genau $k$ von den zu $v$ inzidenten Kanten im Schnitt. Da jede Kante, die nicht zu $v$ inzident ist, von dem Umfärben von $v$ nicht betroffen ist (solch eine Kante ist auf dem Schnitt vor dem Umfärben genau dann, wenn sie nach dem Umfärben auf dem Schnitt ist), enthält der Schnitt nach dem Umfärben von $v$ genau $k - \ell$ Kanten mehr. Da der ursprüngliche Schnitt maximal ist, gilt $k - \ell \leq 0$ und daher $k \leq \ell$. Also hat $v$ mindestens so viele blaue Nachbarn wie rote Nachbarn (vor dem Umfärben).

(b) Zeigen Sie, dass der folgende Algorithmus $\mathcal{A}$ ein Approximationsalgorithmus für MAXIMUM CUT mit einer relativen Gütegarantie von 2 ist:

- Färbe jeden Knoten beliebig; rot oder blau (zum Beispiel alle Knoten rot).
- Solange es einen Knoten $v \in V$ gibt, bei dem ein Umfärben von $v$ die Anzahl der Kanten zwischen roten und blauen Knoten erhöht, färbe solch einen Knoten $v$ um.

*Hinweis: Benutzen Sie eine triviale obere Schranke an $OPT(I)$.*

*Lösung:* Offensichtlich gilt $\text{OPT}(I) \leq |E|$, da höchstens alle Kanten auf dem Schnitt liegen können. Betrachte eine Lösung, die von $\mathcal{A}$ berechnet wurde. Sei für einen Knoten $v$ die Anzahl der Nachbarn von $v$ mit $\deg(v)$ bezeichnet. Analog zur Argumentation in Aufgabenteil (a) sehen wir, dass jeder rote Knoten $v$ mindestens $\deg(v)/2$ blaue Nachbarn hat. Denn hätte $v$ nur $k < \deg(v)/2$ blaue Nachbarn, dann würde das Umfärben von $v$ den Wert der Lösung um $(\deg(v) - k) - k > 0$ erhöhen, und $\mathcal{A}$ wäre noch nicht fertig. Entsprechend hat jeder blaue Knoten $v$ mindestens $\deg(v)/2$ rote Nachbarn.

Nun gilt für den Wert $\mathcal{A}(I)$ der von $\mathcal{A}$ gefundenen Lösung:

$$\begin{aligned} 2 \cdot \mathcal{A}(I) &= \sum_{v\text{rot}} \#\text{blaue Nachbarn von } v + \sum_{v\text{blau}} \#\text{rote Nachbarn von } v \\ &\geq \sum_{v\text{rot}} \deg(v)/2 + \sum_{v\text{blau}} \deg(v)/2 \\ &= \sum_{v \in V} \deg(v)/2 = |E| \geq \text{OPT}(I). \end{aligned}$$

Also gilt $\mathcal{A}(I) \geq \frac{1}{2}\text{OPT}(I)$ für jede Instanz $I$ und damit hat $\mathcal{A}$ eine relative Gütegarantie von 2.

(c) Zeigen Sie, dass Algorithmus $\mathcal{A}$ polynomielle Laufzeit hat.

*Hinweis: Betrachten Sie nach jedem Schleifendurchlauf den Wert der aktuellen Lösung.*

*Lösung:* Die initiale Färbung und jeder Schleifendurchlauf in $\mathcal{A}$ (inklusive des Testens des Bedingung) kann offensichtlich in polynomialer Zeit durchgeführt werden. Um die Anzahl der Schleifendurchläufe zu beschränken, beobachten wir den Wert $x$ der aktuellen Lösung nach jedem Durchlauf.

Anfangs gilt $x \geq 0$. In jeder Iteration erhöht sich $x$ um mindestens 1. Außerdem gilt zu jeder Zeit $x \leq \mathrm{OPT}(I) \leq |E|$. Demnach gibt es höchstens $|E|$ Schleifendurchläufe, was polynomial in der Eingabe ist.

## **Problem 7:** Gemischtes 6 Punkte

Sind die folgenden Aussagen korrekt? Begründen Sie jeweils kurz. Ohne Begründung gibt es keine Punkte.

(a) Es ist nicht entscheidbar, ob eine gegebene Grammatik kontextsensitiv ist.

*Lösung:* **Falsch.**

Man kann überprüfen, ob jede Produktion entweder die Form $S \to \varepsilon$ oder $u \to v$ mit $u \in V^+, v \in ((V \cup \Sigma) \setminus \{S\})^+$ und $|u| \leq |v|$ hat.

(b) Wenn eine reguläre Sprache $L$ ein Wort der Länge 12 und ein Wort der Länge 14 enthält, dann enthält $L$ auch ein Wort der Länge 13.

*Lösung:* **Falsch.**

Ein Gegenbeispiel ist die Sprache zu dem regulären Ausdruck $(aa)^*$.

(c) Wenn $L_1$ und $L_2$ zwei $NP$-vollständige Sprachen sind, dann kann $L_1$ auf $L_2$ polynomiell reduziert werden.

*Lösung:* **Richtig.**

Da $L_1$ und $L_2$ $NP$-vollständig sind, ist insbesondere $L_1$ in $NP$ und $L_2$ ist $NP$-schwer. Nach Definition von $NP$-schwer kann jede Sprache in $NP$ (also auch $L_1$) auf $L_2$ polynomiell reduziert werden.

(d) Ein polynomieller Approximationsalgorithmus mit absoluter Gütegarantie 1 liefert stets optimale Lösungen.

*Lösung:* **Falsch.**

Der Wert der Lösung weicht um höchstens 1 vom Optimum ab.

(e) Die Laufzeit von deterministischen endlichen Automaten ist polynomiell in der Größe der Eingabe.

*Lösung:* **Richtig.**

Jedes Symbol der Eingabe erzeugt genau einen Übergang. Damit ist die Laufzeit linear in der Länge des Eingabewortes.

(f) Der Huffman-Code erkennt Einzelfehler.

*Lösung:* **Falsch.**

Der Huffman-Code erkennt keine Fehler.