Universität Karlsruhe
Fakultät für Informatik
ILKD Prof. Dr. D. Wagner

Theoretische Informatik
WS 2003/04
20. Februar 2004

1. Klausur zur Vorlesung
Informatik III
Wintersemester 2003/2004

Mit Lösung!

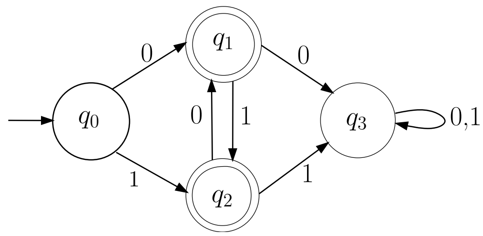
Beachten Sie:

- Bringen Sie den Aufkleber mit Ihrem Namen und Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes weitere Blatt mit Ihrem Namen und Ihrer Matrikelnummer.
- Schreiben Sie die Lösungen auf die Aufgabenblätter und Rückseiten. Zusätzliches Papier erhalten Sie bei Bedarf von der Aufsicht.
- Zum Bestehen der Klausur sind **20** der möglichen **60** Punkte hinreichend.
- Es sind keine Hilfsmittel zugelassen.

Aufgabe	Mögliche Punkte					Erreichte Punkte				
	a	b	c	d	Σ	a	b	c	d	Σ
1	2	2	4	4	12					
2	2	2	4	4	12					
3	4	4	4	-	12				-	
4	3	3	2	4	12					
5	12x1				12					
Σ					60					

Aufgabe 1: (2+2+4+4=12 Punkte)

(a) Gegeben sei folgender deterministischer endlicher Automat (DEA):



Werden die Wörter '01011' und '10101' von dem Automaten akzeptiert? Begründen Sie Ihre Antwort (durch Angabe der durchlaufenen Zustände).

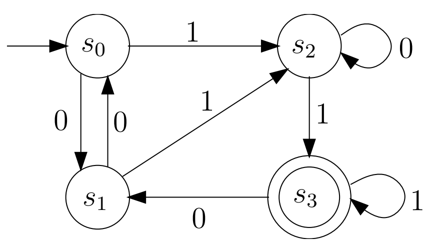
Geben Sie einen regulären Ausdruck für die von dem Automaten akzeptierte Sprache an.

Lösung: 10101 wird akzeptiert, denn $s \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{1} q_2$ und q_2 ist akzeptierender Endzustand.
01011 wird nicht akzeptiert, denn $s \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_3$ und q_3 ist kein akzeptierender Endzustand.
Regulärer Ausdruck für die Sprache: $0(10)^* \cup (01)^+ \cup 1(01)^* \cup (10)^+$.

(b) Seien $L_1 = \{ab, b, abc\}$ und $L_2 = \{b, bc\}$ zwei Sprachen über dem Alphabet $\Sigma = \{a, b, c\}$.
Bestimmen Sie $L_1 \cdot L_2$, L_1/L_2 und $(L_2)^2$.

Lösung: $L_1 \cdot L_2 = \{abb, abbc, bb, bbc, abcb, abcbc\}$
$L_1/L_2 = \{a, \varepsilon\}$
$(L_2)^2 = \{bb, bbc, bcb, bcbc\}$

(c) Minimieren Sie folgenden Automaten mittels Konstruktion des Äquivalenzklassenautomaten. Geben Sie die Übergangsfunktion des Minimalautomaten tabellarisch an.



Lösung: Schritte zur Berechnung eines äquivalenten Minimalautomaten:

- Trenne End- von Nichtendzuständen: $\{s_0, s_1, s_2\}, \{s_3\}$

- 0 trennt keine Zustände voneinander
- 1 trennt s_2 von s_0 und s_1: $\{s_0, s_1\}, \{s_2\}, \{s_3\}$
- 00 trennt keine Zustände voneinander
- 01 trennt keine Zustände voneinander
- 10 trennt keine Zustände voneinander
- 11 trennt keine Zustände voneinander

Neuer Startzustand: $\{s_0, s_1\}$, neuer Endzustand: $\{s_3\}$

$\tilde{q}$	a (gelesener Buchstabe)	$\tilde{\delta}(\tilde{q}, a)$
(s_0, s_1)	0	(s_0, s_1)
(s_0, s_1)	1	(s_2)
(s_2)	0	(s_2)
(s_2)	1	(s_3)
(s_3)	0	(s_0, s_1)
(s_3)	1	(s_3)

(d) Beweisen Sie, dass die Sprache $L = \{a^n b a^m b a^{n+m} \mid n, m \geq 1\}$ nicht regulär ist.

Lösung: Annahme: L regulär.
Mit dem Pumping Lemma folgt dann: $\exists n \in \mathbb{N}$, sodass $\forall w \in L$ mit $|w| > n$ gilt: $\exists$ Zerlegung $w = uvx$ mit $v \neq \varepsilon$, $|uv| \leq n$ und $uv^i x \in L$ für alle $i \in \mathbb{N}$.
Betrachte $w = a^n b a^n b a^{2n}$. Dann ist $|w| > n$. Sei $w = uvx$ eine Zerlegung gemäss dem Pumping-Lemma. Wegen $|uv| < n$ und $v \neq \varepsilon$ ist $v = a^k$ für ein k mit $1 \leq k \leq n$. Dann ist aber $uv^0 x = a^{n-k} b a^n b a^{2n} \notin L$, da $n - k + n \neq 2n$. Widerspruch!

Aufgabe 2: (2+2+4+4=12 Punkte)

Betrachten Sie eine Turingmaschine $\mathcal{M}$ mit $\Sigma = \{a\}$, $\Gamma = \Sigma \cup \{\sqcup\}$, $Q = \{s, q, f\}$, s Startzustand und f der einzige akzeptierende Endzustand. Die Übergangsfunktion ist gegeben durch

$$
\begin{aligned}
\delta(s, a) &= (q, a, R) \\
\delta(s, \sqcup) &= (f, \sqcup, N) \\
\delta(q, a) &= (s, a, R) \\
\delta(q, \sqcup) &= (s, a, N) \\
\delta(f, a) &= (f, a, N) \\
\delta(f, \sqcup) &= (f, \sqcup, N),
\end{aligned}
$$

und die von $\mathcal{M}$ akzeptierte Sprache sei $L_{\mathcal{M}}$.

(a) Kreuzen Sie für folgende Aussagen an, ob diese wahr oder falsch sind.

Hinweis: Für jede richtige Antwort gibt es einen Punkt, für jede falsche Antwort wird ein Punkt abgezogen. Es wird keine negative Gesamtpunktzahl für diese Teilaufgabe geben.

$aaaa \in L_{\mathcal{M}}$. *Lösung:* [X] Wahr [ ] Falsch

Für alle $n \in \mathbb{N}$ ist $a^n \in L_{\mathcal{M}}$. *Lösung:* [ ] Wahr [X] Falsch

(b) Geben Sie für obige Turingmaschine eine modifizierte Übergangsfunktion an, so dass die Funktion

$$f : \Sigma^* \to \Gamma^* : \ f(a^n) = a^{n+1} \quad (n \in \mathbb{N}_0)$$

realisiert wird und die neue Turingmaschine stets hält.

Lösung: Die veränderten Übergänge sind $\delta(s, a) = (s, a, R)$, $\delta(s, \sqcup) = (f, a, R)$. Alle anderen Übergänge bleiben gleich.

(c) Sei $L \subset \{\mathtt{0},\mathtt{1}\}^*$ eine Sprache, die nicht entscheidbar ist. Beweisen Sie, dass die Sprache

$$L' = \{\mathtt{1}w \mid w \in L\}$$

nicht entscheidbar ist.

Lösung: Angenommen L' ist entscheidbar. Dann gibt es eine Turingmaschine $\mathcal{M}'$, die L' akzeptiert und immer hält. Konstruiere nun eine Turingmaschine $\mathcal{M}$: Schreibe eine `1` vor die Eingabe und wende dann $\mathcal{M}'$ an. Betrachte eine Eingabe $w \in \{0, 1\}^*$: Falls $w \in L$, so wird w von M akzeptiert, denn $1w \in L'$. Falls $w \notin L$, so hält $\mathcal{M}$, weil $\mathcal{M}'$ immer hält, aber w wird von $\mathcal{M}$ nicht akzeptiert, da $1w \notin L'$. Die Maschine $\mathcal{M}$ akzeptiert also L und hält immer, d.h. L ist entscheidbar. Dies ist ein Widerspruch zur Voraussetzung, also ist die Annahme falsch und L' ist nicht entscheidbar.

(d) Zeigen Sie, dass das Post'sche Korrespondenzproblem semientscheidbar ist.

Lösung: Konstruiere eine Turingmaschine, die die Sprache aller lösbaren Instanzen des Post'schen Korrespondenzproblem akzeptiert (aber nicht unbedingt auf alle Eingaben hält): Betrachte nach und nach systematisch alle möglichen Indexfolgen. Zuerst alle der Länge 1, dann die der Länge 2 usw. Überprüfe jeweils, ob eine solche Indexfolge eine gültige Lösung ist, durch Konstruktion und Vergleich der beiden Wörter. Wenn eine gültige Lösung gefunden wird hält die Turingmaschine und akzeptiert die Instanz.

Falls eine Instanz lösbar ist, so gibt es eine Indexfolge, die zwei gleiche Wörter induziert. Diese Indexfolge wird dann auch gefunden, und die Turingmaschine hält und akzeptiert die Instanz. Falls eine Instanz nicht lösbar ist, hält die Maschine nie (und akzeptiert also auch nicht). Damit ist das Post'sche Korrespondenzproblem semientscheidbar.

Aufgabe 3: (4+4+4 = 12 Punkte)

[Hinweis: Für die in dieser Aufgabe vorkommenden Probleme sind jeweils am Ende des Aufgabentextes nochmals die Definitionen angegeben.]

(a) Zeigen Sie: Das Problem DOUBLE-SAT ist $\mathcal{NP}$-vollständig.

[Hinweis: Verwenden Sie SAT für die polynomiale Transformation.]

Problem DOUBLE-SAT

Gegeben: Eine Menge $U = \{u_1, \ldots, u_m\}$ von booleschen Variablen, eine Menge C von Klauseln über U.

Frage: Existieren *zwei verschiedene* Wahrheitsbelegungen von U, so dass C erfüllt wird, d.h., dass alle Klauseln aus C den Wahrheitswert `wahr` annehmen?

Lösung:

1. DOUBLE-SAT $\in \mathcal{NP}$:

   Es werden zwei Variablenbelegung geraten und für jede Klausel geprüft, ob bei beiden Belegungen mindestens ein Literal wahr wird, sowie, ob die Belegungen verschieden sind. Dies geht in polynomialer Zeit, da die Eingabe im Wesentlichen nur einmal durchlaufen zu werden braucht.

2. SAT $\propto$ DOUBLE-SAT:

   Aus einer Instanz von SAT wird eine äquivalente Instanz von DOUBLE-SAT, indem eine neue Variable v hinzugefügt wird; die Klauselmenge bleibt unverändert. Jede Belegung, die die SAT-Instanz erfüllt, induziert zwei Belegungen (mit v =wahr bzw. v =falsch) für die DOUBLE-SAT-Instanz. Umgekehrt induzieren zwei erfüllende Belegungen der DOUBLE-SAT-Instanz auch mindestens eine erfüllende Belegung der SAT-Instanz, indem die Belegung von v einfach vergessen wird.

   Die Größe der Instanz wächst nur um eine Konstante, also ist die Transformation in polynomialer Zeit berechenbar.

(b) Zeigen Sie, dass es keinen Approximationsalgorithmus mit absoluter Gütegarantie für die Minimierungsversion des $\mathcal{NP}$-vollständigen Problems VERTEX COVER gibt, falls $\mathcal{P} \neq \mathcal{NP}$.

Ein *Vertex-Cover* eines ungerichteten Graphen $G = (V, E)$ ist eine Teilmenge $V' \subseteq V$, so dass für jede Kante aus E mindestens einer ihrer Endknoten in V' ist.

Problem VERTEX COVER

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und eine Konstante K.

Frage: Hat G ein Vertex-Cover der Größe höchstens K?

Lösung:

Wir nehmen an, es gebe einen solchen Approximationsalgorithmus A mit Gütegarantie C und folgern daraus die Existenz eines exakten, polynomialen Algorithmus für VERTEX COVER. Dies ist ein Widerspruch zu $\mathcal{P} \neq \mathcal{NP}$, da VERTEX COVER $\mathcal{NP}$-vollständig ist.

Sei $I = (G, K)$ eine Instanz von VERTEX COVER. Wir konstruieren daraus eine Instanz $I' = (G')$ der Optimierungsversion. G' bestehe aus $C + 1$ Kopien von G, die untereinander nicht verbunden sind. Wir wenden A auf I' an. Es gilt $\mathrm{OPT}(I') = (C+1) \cdot \mathrm{OPT}(I)$ und $A(I')$ induziert eine Lösung der Größe $\lfloor A(I')/(C+1) \rfloor$. [Jede Knotenüberdeckung der Größe s' in G' induziert eine Knotenüberdeckung der Größe $s \leq s'/(C+1)$ in G, da die Kopien in G' unabhängig voneinander sind und somit in mindestens einer Komponente eine Überdeckung kleiner oder gleich $s'/(C+1)$ existiert. Umgekehrt liefert jede Überdeckung der Größe s in G eine Überdeckung der Größe $s(C+1)$ in G'.] A liefert eine Lösung mit $A(I') - OPT(I') \leq C$. Zusammen gilt also $\lfloor A(I')/(C+1) \rfloor \leq C/(C+1) + \mathrm{OPT}(I)$ und da alle Lösungen ganzzahlig sind und jede Lösung für I' eine Lösung für I induziert, gilt

$\lfloor A(I')/(C+1) \rfloor$ =OPT(I), was einen exakten Algorithmus impliziert. Die Transformation funktioniert in polynomialer Zeit, da nur konstant viele Kopien von G erzeugt werden müssen und A selber ist auch polynomial.

(c) Zeigen Sie: MISSING EDGE $\in$ co-$\mathcal{NP}$.

Problem MISSING EDGE

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und eine Konstante K

Frage: Existieren für jede Teilmenge $V' \subseteq V$ mit $|V'| = K$ zwei Knoten $s, t \in V'$ mit $\{s, t\} \notin E$.

Lösung: Bedingung für die Ja-Instanzen von MISSING EDGE:

$$\forall V' \subseteq V \text{ mit } |V'| = K : \exists (s,t) \in V' \times V' : s \neq t \text{ und } s,t \notin E$$

Verneinung:

$$\exists V' \subseteq V \text{ mit } |V'| = K : \forall (s,t) \in V' \times V' : s = t \text{ oder } s,t \in E$$

Ein Graph hat die K-MISSING-EDGE-Eigenschaft genau dann, wenn es in G keine Clique der Größe K gibt. Also ist MISSING EDGE = co-CLIQUE. Da CLIQUE in $\mathcal{NP}$ liegt, ist MISSING EDGE in co-$\mathcal{NP}$.

Aufgabe 4:

(3+3+2+4=12 Punkte)

Die Grammatik G sei gegeben durch das Alphabet {0,1,2}, die Variablen $\{S, A, B\}$, das Startsymbol S und die folgenden Regeln:

$$S \to AB,\ A \to 0A \mid 0,\ B \to 1B2 \mid 12$$

(a) Geben Sie einen Kellerautomaten an, der die Sprache $L(G)$ akzeptiert (ohne Begründung).

Lösung: $Q = \{q_0, q_1, q_2\}, \Gamma = \{1, Z_0\}$, Startzustand $q_0, F = \emptyset$,

$$\delta(q_0, 0, Z_0) = \{(q_0, Z_0), (q_1, Z_0)\}, \quad \delta(q_1, 1, Z_0) = \{(q_1, 1)\}, \quad \delta(q_1, 1, 1) = \{(q_1, 11)\},$$

$$\delta(q_1, 2, 1) = \{(q_2, \varepsilon)\}, \quad \delta(q_2, 2, 1) = \{(q_2, \varepsilon)\}, \quad \delta(p, a, Z) = \emptyset \text{ sonst.}$$

Der Kellerautomat akzeptiert durch leeren Stack.

(b) Bringen Sie G durch eine systematische Konstruktion in Chomsky-Normalform.

Lösung:

1. Schritt: rechts nur ein Terminal oder nur Variablen:

$$S \to AB, \quad A \to Y_0 A \mid 0, \quad B \to Y_1 B Y_2 \mid Y_1 Y_2, \quad Y_0 \to 0, \quad Y_1 \to 1, \quad Y_2 \to 2$$

2. Schritt: Umformung von Regeln der Länge größer 3:

$$S \to AB, \quad A \to Y_0 A \mid 0, \quad B \to Y_1 B', \quad B' \to B Y_2, \quad B \to Y_1 Y_2, \quad Y_0 \to 0, \quad Y_1 \to 1, \quad Y_2 \to 2$$

(c) Sei $G' = (\Sigma, V, S, R)$ die Grammatik mit $\Sigma = \{a, b, c\}$, $V = \{S, A, B, C, Q, X, Y\}$ und

$$R = \{S \to XY,\ X \to AQ \mid AB,\ Q \to XB,\ Y \to YC \mid c,\ A \to a,\ B \to b,\ C \to c\}.$$

Wenden Sie den CYK-Algorithmus auf das Wort *abcc* und die Grammatik G' an.

Lösung:

$$V_{11} = \{A\} \quad V_{22} = \{B\} \quad V_{33} = \{C, Y\} \quad V_{44} = \{C, Y\}$$
$$V_{12} = \{X\} \quad V_{23} = \{\} \quad V_{34} = \{Y\}$$
$$V_{13} = \{S\} \quad V_{24} = \{\}$$
$$V_{14} = \{S\}$$

(d) Zeigen Sie, dass die Sprache $\{0^i 1^j 2^i 3^j \mid i, j \geq 1\}$ nicht kontextfrei ist.

Lösung: Zu beliebigem n wähle $z = 0^n 1^n 2^n 3^n$. In jeder Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$ kann vx nicht gleichzeitig Nullen und Zweien bzw. gleichzeitig Einsen und Dreien enthalten. Wähle $i = 2$. Dann enthält $uv^i wx^i y$ eine unterschiedliche Anzahl von Nullen und Zweien bzw. von Einsen und Dreien, und gehört damit nicht zur Sprache. Also ist wegen des Pumping-Lemmas für kontextfreie Sprachen die Sprache nicht kontextfrei.

Aufgabe 5: (12x1=12 Punkte)

Kreuzen Sie für folgende Aussagen an, ob diese wahr oder falsch sind.
Hinweis: Für jede richtige Antwort gibt es einen Punkt, für jede falsche Antwort wird ein Punkt abgezogen. Es wird keine negative Gesamtpunktzahl für diese Aufgabe geben.

		Wahr	Falsch
Seien L_1, L_2 reguläre Sprachen. Dann ist auch $L_1 \cap L_2$ regulär.	*Lösung:*	X	
Die regulären Ausdrücke $(a^*b^*)^*$ und $(a^*b)^*$ sind äquivalent.	*Lösung:*		X
Jede Turingmaschine mit zwei Bändern kann durch eine Turingmaschine mit einem Band simuliert werden.	*Lösung:*	X	
Jeder Kellerautomat mit zwei Stacks kann durch einen Kellerautomat mit einem Stack simuliert werden.	*Lösung:*		X
Sei L_1 semientscheidbar und L_2 entscheidbar. Dann ist $L_1 \cap L_2$ immer entscheidbar.	*Lösung:*		X
Σ^* und $\emptyset$ sind $\mathcal{NP}$-vollständig.	*Lösung:*		X
Für jede Sprache L gilt: Aus $L \in \mathcal{P}$ folgt $L^C \in \mathcal{P}$.	*Lösung:*	X	
Für jedes Optimierungsproblem Π gilt: Es gibt für Π einen Approximationsalgorithmus mit relativer Gütegarantie 2 wenn es ein PAS für Π gibt.	*Lösung:*	X	
Das KNAPSACK-Problem, eingeschränkt auf Kosten aus $\{0,1\}$ und Gewichte aus $\{3,5,7\}$ ist in $\mathcal{P}$.	*Lösung:*	X	
Sei Σ ein endliches Alphabet. Dann ist Σ^* kontextfrei.	*Lösung:*	X	
Es existiert ein NEA, der die Sprache aller Java-Programme erkennt.	*Lösung:*		X
Wenn für eine Sprache L gilt: $\exists n \in I\!N : \ \forall w \in L, \lvert w \rvert > n : \ \exists$ Zerlegung $w = uvx, \lvert uv \rvert \leq n, v \neq \varepsilon : \forall i \in I\!N_0 : \ uv^ix \in L$, dann ist L regulär.	*Lösung:*		X