Technische Universität München
Fakultät für Informatik
Prof. Tobias Nipkow, Ph.D.
Sascha Böhme, Lars Noschinski

Sommersemester 2011
Lösungsblatt Wiederholungsklausur
22. Oktober 2011

Einführung in die Theoretische Informatik

Name	Vorname	Studiengang	Matrikelnummer
....	....	☐ Diplom ☐ Inform. ☐ Bachelor ☐ BioInf. ☐ Lehramt ☐ Mathe.	....

Hörsaal	Reihe	Sitzplatz	Unterschrift
....	....	....	....

Allgemeine Hinweise

- Bitte füllen Sie obige Felder in Druckbuchstaben aus und unterschreiben Sie!
- Bitte schreiben Sie nicht mit Bleistift oder in roter/grüner Farbe!
- Die Arbeitszeit beträgt 180 Minuten.
- Alle Antworten sind in die geheftete Angabe auf den jeweiligen Seiten (bzw. Rückseiten) der betreffenden Aufgaben einzutragen. Auf dem Schmierblattbogen können Sie Nebenrechnungen machen. Der Schmierblattbogen muss ebenfalls abgegeben werden, wird aber in der Regel nicht bewertet.
- Es sind keine Hilfsmittel außer einem DIN-A4-Blatt zugelassen.

Hörsaal verlassen von ...... bis ...... / von ...... bis ......
Vorzeitig abgegeben um ......
Besondere Bemerkungen:

	A1	A2	A3	A4	A5	A6	A7	A8	Σ	Korrektor
Erstkorrektur										
Zweitkorrektur										

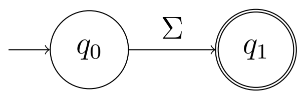
Aufgabe 1 (8 Punkte)

Wahr oder falsch? Begründen Sie im Folgenden Ihre Antworten möglichst knapp!

1. Seien L, K formale Sprachen. Aus $L^* = K^*$ folgt $L = K$.
2. Jede Sprache ist entscheidbar oder semientscheidbar.
3. Endliche Vereinigungen entscheidbarer Sprachen sind semientscheidbar.
4. Jeder NFA mit genau zwei Zuständen, der mehr als drei unterschiedliche Wörter akzeptiert, akzeptiert eine unendliche Sprache.
5. Für jede kontextfreie Sprache L gilt: L liegt in P.
6. Sei $D = \{L \mid \text{es gibt eine deterministische Turingmaschine } M \text{ mit } L(M) = L\}$. Dann gilt $NP \subseteq D$.
7. Sei $f(n) \in \{0, 1, 2, 3\}$ für alle natürlichen Zahlen n. Dann ist f berechenbar.
8. Eine Turingmaschine über Σ, die alle Wörter $w \in \Sigma^*$ mit $|w| > 13$ akzeptiert, beschreibt eine reguläre Sprache.

Lösungsvorschlag

1. Falsch. $L = \{a\}$ und $K = \{a, aa\}$ liefern ein Gegenbeispiel.
2. Falsch. $\overline{K}$ ist nicht semi-entscheidbar (also auch nicht entscheidbar).
3. Wahr. Entscheidbare Sprachen sind abgeschlossen unter Vereinigung und eine entscheidbare Sprache ist natürlich auch wieder semientscheidbar.
4. Falsch. Für $|\Sigma| > 3$ ist der folgende NFA ein Gegenbeispiel:

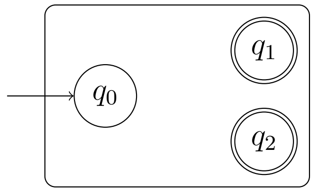


5. Wahr. Der CYK-Algorithmus entscheidet die Sprache in polynomieller Zeit.
6. Wahr. Alle Sprachen in NP sind entscheidbar, insbesondere können sie also von einer (deterministischen) TM erkannt werden.
7. Falsch. χ_K ist ein Gegenbeispiel.
8. Wahr. Das Komplement der erkannten Sprache ist endlich und damit regulär. Da reguläre Sprachen abgeschlossen sind unter Komplement, ist also auch die erkannte Sprache regulär.

Aufgabe 2 (8 Punkte)

Sei $\Sigma = \{a, b\}$ ein Alphabet.

1. Sei $M = (Q, \Sigma, \delta, q_0, \{q_1, q_2\})$ ein endlicher Automat mit zwei Endzuständen q_1 und q_2 aus Q, der wie folgt graphisch dargestellt werden kann:



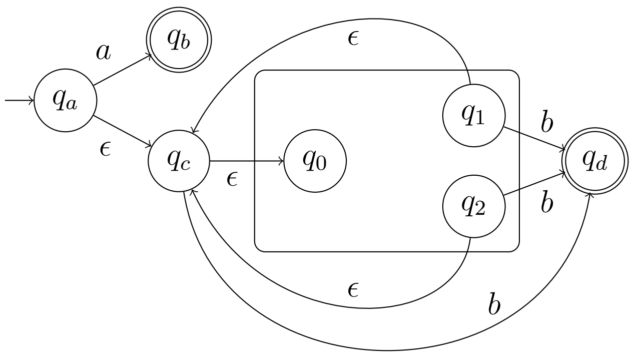
   Sei α ein regulärer Ausdruck mit $L(\alpha) = L(M)$. Konstruieren Sie einen endlichen Automaten M', gegebenenfalls mit $\boldsymbol{\epsilon}$-Übergängen, so dass $L(M') = L(a \mid \alpha^* b)$ gilt. Beschreiben Sie dazu die Zustandsübergänge graphisch.

2. Geben Sie eine rechts- oder linkslineare Grammatik G zu dem regulären Ausdruck $\alpha = a^*bba^*$ an, so dass $L(G) = L(\alpha)$ gilt. Geben Sie außerdem eine Ableitung des Wortes $aabba$ in G an.

3. Beweisen Sie, dass die Sprache $L = \{a^m b^n c^n \mid 1 \leq m \wedge 0 \leq n\}$ nicht regulär ist. Verwenden Sie dazu das Pumping-Lemma.

Lösungsvorschlag

1. $M' = (Q \cup \{q_a, q_b, q_c, q_d\}, \Sigma, \delta', q_a, \{q_b, q_d\})$ mit δ' wie folgt:



2. $G = (\{S, A, B\}, \Sigma, P, S)$ mit den folgenden Produktionen P:

$$
\begin{aligned}
S &\to Sa \mid Ab \\
A &\to Bb \\
B &\to \epsilon \mid Ba
\end{aligned}
$$

   Die einzig mögliche Ableitung von $aabba$ in G ist wie folgt:

$$S \to Sa \to Aba \to Bbba \to Babba \to Baabba \to aabba$$

3. Wir nehmen an, L wäre regulär. Dann gibt es eine Pumping-Lemma-Zahl $N > 0$, so dass gilt: Für alle $z \in L$ mit $N \leq |z|$ gibt es eine Zerlegung $z = uvw$ mit $|uv| \leq N$ und $v \neq \epsilon$, so dass $\forall i \in \mathbb{N}.\ uv^i w \in L$ gilt.

   Sei N eine solche Pumping-Lemma-Zahl. Wir betrachten jetzt das Wort $z = ab^N c^N$ und seine möglichen Zerlegungen in uvw. Wegen $|uv| \leq N$ gilt $\#_c(uv) = 0$ und damit reicht es aus, die folgenden zwei Fälle zu betrachten:

   - $\#_b(uv) = 0$. Dann ist $uv = a$ und wegen $v \neq \epsilon$ also $v = a$. Dann ist aber $uv^0w = b^N c^N \notin L$.
   - $\#_b(uv) > 0$. Dann ist der letzte Buchstabe von uv ein b und mit $v \neq \epsilon$ gilt auch $\#_b(v) > 0$. Also ist $\#_b(uv^0w) \neq \#_c(uv^0w)$ und damit ist $uv^0w \notin L$

   Entgegen der Annahme ist N also keine Pumping-Lemma-Zahl. Damit haben wir einen Widerspruch zu der Annahme, das L regulär wäre. L ist also nicht regulär.

Aufgabe 3 (8 Punkte)

Sei Σ ein Alphabet. Für eine Sprache $L \subseteq \Sigma^*$ sei

$$dehn(L) := \{a_1^{k_1} a_2^{k_2} \cdots a_n^{k_n} \mid a_1 a_2 \cdots a_n \in L \wedge a_1, a_2, \ldots, a_n \in \Sigma \wedge k_1, k_2, \ldots, k_n \geq 1\}$$

die *Dehnungssprache* von L.

Beispiel: Ist $aab \in L$, so sind unter anderem $aab, aabb, aaaaabb \in L$.

1. Betrachten Sie die Sprache L, die durch den regulären Ausdruck $(abb^*|a)^*(a|\boldsymbol{\epsilon})$ gegeben ist. Geben Sie einen regulären Ausdruck für $dehn(L)$ an.

2. Folgt aus $L \subseteq \Sigma^*$ regulär, dass $dehn(L)$ regulär ist? Konstruieren Sie dazu sowohl einen endlichen Automaten als auch einen regulären Ausdrück für die Dehnungssprache:

   (a) Geben Sie eine (rekursive) Funktion d an, so dass für jeden regulären Ausdruck α über Σ gilt: $L(d(\alpha)) = dehn(L(\alpha))$.

   (b) Gegeben ein endlicher Automat $M = (Q, \Sigma, \delta, q_0, F)$. Geben Sie einen endlichen ($\boldsymbol{\epsilon}$-)Automat M' an, so dass $L_{M'} = dehn(L_M)$. Beschreiben Sie ihre Idee.

   In beiden Fällen ist eine direkte Konstruktion (ohne Umweg über eine andere Darstellung der Sprache) gefordert.

3. Zeigen oder widerlegen Sie: Es gibt eine nicht-reguläre Sprache L, so dass $dehn(L)$ regulär ist.

Lösungsvorschlag

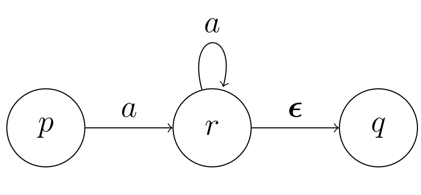
1. $dehn(L) = \{(a^+b^+|a^+)^*a^*\}$

2. (a) Für reguläre Ausdrücke definiere $d : RE \to RE$ wie folgt:

$$\begin{aligned} d(\boldsymbol{\epsilon}) &= \boldsymbol{\epsilon} \\ d(\boldsymbol{\emptyset}) &= \boldsymbol{\emptyset} \\ d(a) &= a^+ \quad \text{für alle } a \in \Sigma \\ d(\alpha|\beta) &= d(\alpha) \mid d(\beta) \\ d(\alpha\beta) &= d(\alpha)d(\beta) \\ d(\alpha^*) &= d(\alpha)^* \end{aligned}$$

   Dann ist $L(d(\alpha)) = dehn(L(\alpha))$.

   (b) Um aus einem DFA M einen $\boldsymbol{\epsilon}$-NFA für $dehn(M)$ zu konstruieren gehe wie folgt vor: Für jede Kante füge einen zusätzlichen Knoten ein und ersetze eine mit $a \in \Sigma$ beschrifteten Kante von p nach q durch folgendes Konstrukt:



Dabei ist r der zusätzliche Knoten für die ersetzte Kante; p und q können identisch sein. Formal sieht die Konstruktion wie folgt aus: Für den DFA $M = (Q, \Sigma, \delta, q_0, F)$ setze $M' = (Q', \Sigma, \delta', q_0, F)$. Da bei einem deterministischen Automaten eine Kante durch Startknoten und Beschriftung eindeutig identifiziert ist, definieren wir $Q' = Q \cup \{q_a \mid q \in Q \wedge a \in \Sigma\}$ und damit ergibt sich δ' für alle $q \in Q$, $a \in \Sigma$ wie folgt:

$$\begin{aligned} \delta'(q, a) &= \{q_a\} & \qquad \delta'(q_a, a) &= \{q_a\} \\ \delta'(q_a, \boldsymbol{\epsilon}) &= \{\delta(q, a)\} & \qquad \delta'(q_a, b) &= \varnothing \qquad \text{falls } a \neq b \end{aligned}$$

3. Die Aussage gilt. $L = \{a^n b^n | n \in \mathbb{N}\}$ über dem Alphabet $\Sigma = \{a, b\}$ ist bekanntermaßen nicht regulär, aber es gilt $dehn(L) = L(\boldsymbol{\epsilon}|a^+b^+)$ und damit ist $dehn(L)$ regulär.

Aufgabe 4 (10 Punkte)

Sei $\Sigma = \{a, b, c\}$.

1. Geben Sie für die Sprache $L = \{(aba)^n b^m (ba)^n b \mid n \geq 0, m \geq 2\}$ eine kontextfreie Grammatik G an, so dass $L(G) = L$ gilt.
2. Sei $G = (\{S\}, \Sigma, \{S \to \epsilon \mid aSc \mid aSbSc\}, S)$ eine kontextfreie Grammatik. Zeigen Sie, dass für alle in G ableitbaren Wörter w die Eigenschaft $\#_a(w) = \#_c(w)$ gilt.
3. Sei $N = \{S, T, U, A, B, C\}$ und $G = (N, \Sigma, P, S)$ die Grammatik mit den folgenden Produktionen P:

$$
\begin{array}{lll}
S \to TT \mid BU & T \to BT \mid TC \mid a & U \to AC \mid BA \\
A \to a & B \to b & C \to c
\end{array}
$$

   (a) Verwenden Sie den CYK-Algorithmus um zu beweisen, dass $bbac \notin L(G)$ gilt. Geben Sie die Berechnungstabelle für den CYK-Algorithmus an.

   (b) Sei G_X die Grammatik, bei der das Startsymbol S durch ein $X \in N$ ausgetauscht wurde. Geben Sie ein $X \in N$ an, so dass $bbac \in L(G_X)$ gilt.

Lösungsvorschlag

1. $G = (\{S, A, B\}, \Sigma, P, S)$ mit den folgenden Produktionen P:

$$
\begin{aligned}
S &\to Ab \\
A &\to abaAba \mid B \\
B &\to bb \mid Bb
\end{aligned}
$$

2. Induktion über die Erzeugung von Wörtern $w \in L(S)$:
   - $S \to \epsilon$: Dann ist $w = \epsilon$, und die Behauptung gilt offensichtlich.
   - $S \to aSc$: Sei $w = avc$ mit $v \in L(S)$. Nach Induktionshypothese gilt die Behauptung für v, und damit folgt:
   $$\#_a(w) = 1 + \#_a(v) = 1 + \#_c(v) = \#_c(w)$$
   - $S \to aSbSc$: Sei $w = aubvc$ mit $u, v \in L(S)$. Nach Induktionshypothese gilt die Behauptung für u und v, und damit folgt:
   $$\#_a(w) = 1 + \#_a(v) + \#_a(u) = 1 + \#_c(v) + \#_c(u) = \#_c(w)$$

3. Die Grammatik liegt bereits in Chomsky-Normalform vor, der CYK-Algorithmus kann also direkt angewendet werden.

   (a) Jetzt können wir den CYK-Algorithmus anwenden. Es ergibt sich die folgende Berechnungstabelle.

$_{14}$ T			
$_{13}$ S, T	$_{24}$ S, T		
$_{12}$ $\emptyset$	$_{23}$ T, U	$_{34}$ T, U	
$_{11}$ B	$_{22}$ B	$_{33}$ A, T	$_{44}$ C
b	b	a	c

Daraus können wir ablesen, dass das Wort $bbac$ nur von T, nicht aber von S produziert wird, also $bbac \notin L(G)$.

(b) Aus der Berechnungstabelle können wir ablesen, dass $bbac \in L(G_X)$.

Aufgabe 5 (4 Punkte)

1. Zeigen Sie, dass die Funktion $prim(n)$ mit folgender Definition primitiv rekursiv ist:

$$prim(n) = \begin{cases} 1 & \text{falls } n \text{ größer als 1 und eine Primzahl ist} \\ 0 & \text{sonst} \end{cases}$$

2. Sei $anzahl\_prim(m, n)$ die Anzahl aller Primzahlen, die echt größer als m und kleiner oder gleich n sind. Zeigen Sie, dass $anzahl\_prim(m, n)$ primitiv rekursiv ist. Sie dürfen dazu die Funktion aus der ersten Teilaufgabe verwenden.

Hinweis: Sie dürfen zusätzlich zu den Basisfunktionen der primitiven Rekursion die folgenden Funktionen als primitiv rekursiv annehmen: Addition $(m + n)$, Multiplikation $(m * n)$, $div(m, n)$, $mod(m, n)$, $pred(n)$, modifizierte Subtraktion $(m \dot{-} n)$, $ifthen(n, a, b)$, Gleichheit $(m = n)$, $c(m, n)$, $p_1(n)$, $p_2(n)$ sowie $q(m) = max\{x \leq m \mid P(x)\}$ für ein PR-Prädikat P. Sie dürfen die erweiterte Komposition und das erweiterte rekursive Definitionsschema verwenden. Alle weiteren Funktionen sind explizit zu definieren. LOOP- oder WHILE-Programme sind nicht erlaubt.

Lösungsvorschlag

1. Unter Verwendung des beschränkten Maximumsoperators:

$$prim(n) = ifthen(pred(n) = 0, 0, max\{x \leq pred(n) \mid n \ mod \ x = 0\} = 1))$$

Alternativlösung:

$$prim(n) = ifthen(pred(n) = 0, 0, prm(n, n)))$$

$$\begin{aligned} prm(0, n) &= 1 \\ prm(i + 1, n) &= ifthen(i = 0, 1, \\ &\quad ifthen(n \ mod \ i = 0, 0, \\ &\quad prm(i, n))) \end{aligned}$$

2.

$$anzahl\_prim(m, n) = anzahl(n) \dot{-} anzahl(m)$$

$$\begin{aligned} anzahl(0) &= 0 \\ anzahl(i + 1) &= prim(i + 1) + anzahl(i) \end{aligned}$$

Aufgabe 6 (9 Punkte)

Sei Σ ein Alphabet.

1. Geben Sie für jede der folgenden Eigenschaften ein Objekt an, dass diese Eigenschaften erfüllt und begründen Sie, dass die Eigenschaften erfüllt werden. Existiert kein solches Objekt, begründen Sie, warum ein solches Objekt nicht existiert.

   (a) Eine totale, nicht-berechenbare Funktion, die unbeschränkt ist (d.h., für alle $n \in \mathbb{N}$ existiert ein $m \in \mathbb{N}$, so dass $f(m) \geq n$).

   (b) Eine totale Funktion $f : \Sigma^* \to \Sigma^*$, für die gilt $|\{w \in \Sigma^* \mid \varphi_w = f\}| = 7$.

2. Zeigen oder widerlegen Sie:

   (a) Die Menge $A_1 = \{w \in \Sigma^* \mid \exists v \in \Sigma^*.\ \varphi_v = \varphi_w \land v \neq w\}$ ist entscheidbar.

   (b) Die Menge $A_2 = \{w \in \Sigma^* \mid \forall x.\ \varphi_w(x) = \varphi_w(0) + x\}$ ist entscheidbar.

   Bei Anwendung des Satzes von Rice muss die verwendete Menge von Funktionen F angegeben werden.

Lösungsvorschlag

1. (a) Sei M eine nicht entscheidbare Menge. Dann ist die Funktion f mit $f(2n) = \chi_M(n)$ und $f(2n+1) = n$ offensichtlich total (denn χ_M ist total) und unbeschränkt. Weiterhin ist f nicht entscheidbar, denn sonst wäre wegen $\chi_M(n) = f(2n)$ auch χ_M entscheidbar. Also erfüllt f die gesuchten Eigenschaften.

   (b) Eine solche Funktion existiert nicht, denn zu jeder berechenbaren Funktion gibt es unendlich viele Turingmaschinen, die diese Funktion berechnen.

2. (a) Wahr. Sei f die Funktion, die zu einer Turingmaschine M eine Turingmaschine $f(M)$ berechnet, die zuerst den Kopf einmal nach links und wieder nach rechts bewegt und danach M simuliert. Dann gilt $\varphi_M = \varphi_{f(M)}$ und $M \neq M'$.

   Insbesondere gilt für jedes $w \in \Sigma^*$: $\varphi_w = \varphi_{M_w} = \varphi_{f(M_w)} = \varphi_w$ und $w \neq w'$, wobei w' die Kodierung von $f(M_w)$ ist.

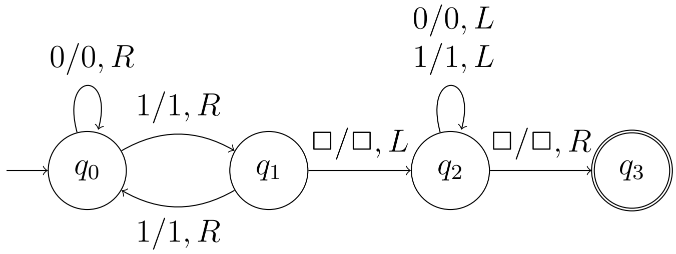
   Damit gilt $A_1 = \Sigma^*$ und damit ist A_1 entscheidbar.

   (b) Falsch. Sei $F = \{f : \mathbb{N} \to \mathbb{N} \mid f(x) = f(0) + x$ und f ist berechenbar$\}$. Dann ist offensichtlich nicht die Menge aller berechenbarer Funktionen und F enthält zumindest die Identitätsfunktion, ist also nicht trivial. Nach dem Satz von Rice ist $A_2 = \{w \in \Sigma^* \mid \varphi_w \in F\}$ daher unentscheidbar.

Aufgabe 7 (8 Punkte)

Wir nennen eine deterministische Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$ *motiviert*, wenn M in jedem Schritt das Band verändert. Formal soll also für alle $q \in Q$, $x \in \Gamma$ und $X \in \{L, N, R\}$ gelten: Wenn $\delta(q, x) = (q', y, X)$ gilt, dann ist $x \neq y$.

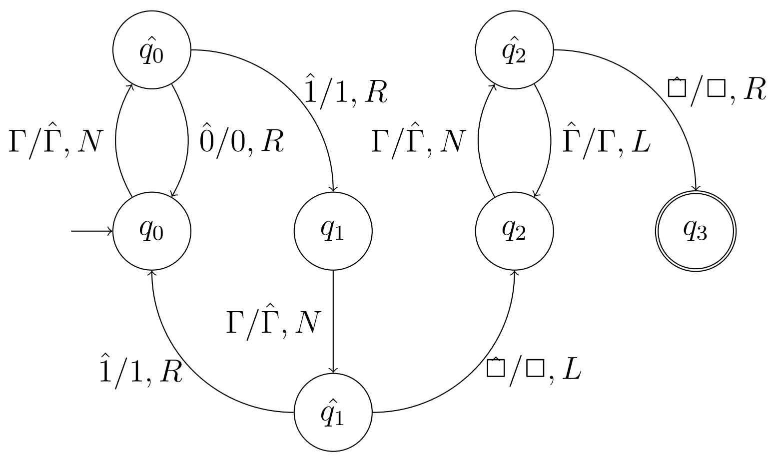
1. Betrachten Sie die folgende Turingmaschine M über $\Sigma = \{0, 1\}$ und $\Gamma = \Sigma \cup \{\Box\}$. Geben Sie eine motivierte TM M' an, so dass $\varphi_M = \varphi_{M'}$ gilt.



2. Zeigen Sie: Für jede deterministische Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$ existiert eine motivierte Turingmaschine M' mit $\varphi_M = \varphi_{M'}$. Begründen Sie die Korrektheit ihrer Konstruktion.

Lösungsvorschlag

1. Wir bezeichnen mit $M/\hat{M}, X$ die Transitionen $x/\hat{x}, X$ für alle $x \in M$. Die Bezeichnung $\hat{M}/M, X$ ist analog definiert.



2. Wir verwenden eine ähnliche Konstruktion wie in der ersten Teilaufgabe und setzen $M' = (Q', \Sigma, \Gamma', \delta', q_0, F)$. Dabei setzen wir $\Gamma' = \Gamma \cup \{\hat{x} \mid x \in \Gamma\}$ und $Q' = Q \cup \{\hat{q} \mid q \in Q\}$. Die Übergangsfunktion definieren wir als

$$\delta'(q, x) = (\hat{q}, \hat{x}, N) \qquad \qquad \delta'(\hat{q}, \hat{x}) = \delta(q, x)$$

für alle $q \in Q$ und $x \in \Gamma$.

Wir zeigen jetzt, dass gilt $\varphi_M = \varphi_{M'}$. Wie man leicht sieht, gilt

$$(\alpha, q, \beta) \to_M (\alpha', q', \beta') \Leftrightarrow (\alpha, q, \beta) \to_{M'}^2 (\alpha', q', \beta')$$

für alle $q \in Q$, $\alpha, \beta \in \Gamma^*$. Wie man per Induktion zeigen kann, gilt damit auch

$$(\Box, q_0, w) \to_M^n (\Box, q_f, w') \Leftrightarrow (\Box, q_0, w) \to_{M'}^{2n} (\Box, q_f, w')$$

für alle $w, w' \in \Sigma^*$ und $q_f \in F$. Da M' nach einer ungeraden Anzahl von Schritten immer in einem Zustand mit Hut ist, kann M' immer nur nach einer geraden Anzahl von Schritten in einem Endzustand landen. Da Eingabealphabet und Startzustand von M und M' gleich sind, gilt damit $\varphi_M = \varphi_{M'}$.

Aufgabe 8 (5 Punkte)

Sei ITE die Mege aller aussagenlogischen Formeln, die ausschließlich aus den Konstanten 0 und 1, logischen Variablen x_i mit $i \in \mathbb{N}$ und der trinären Operation $\mathsf{if}\ \_\ \mathsf{then}\ \_\ \mathsf{else}\ \_$ aufgebaut sind, wobei Klammern natürlich zugelassen sind. Dabei ist $\mathsf{if}\ x_i\ \mathsf{then}\ x_j\ \mathsf{else}\ x_l$ als x_j definiert, wenn $x_i = 1$ ist und als x_l andernfalls.
Wir betrachten das Problem $ISAT$.

> **Gegeben:** $F \in ITE$.
>
> **Problem:** Ist F erfüllbar, das heißt, gibt es eine Belegung der Variablen mit Konstanten 0 oder 1, so dass F den Wert 1 annimmt?

1. Zeigen Sie: $ISAT$ ist NP-hart.

   Sie dürfen dazu benutzen, dass das SAT-Problem NP-vollständig ist.

2. Indem man jedes Vorkommen von $\mathsf{if}\ a\ \mathsf{then}\ b\ \mathsf{else}\ c$ durch $(a \wedge b) \vee (\neg a \wedge c)$ ersetzt, kann man $ISAT$ auf SAT reduzieren. Warum beweist das nicht, dass $ISAT$ in NP liegt?

Lösungsvorschlag

1. Sei f die Abbildung, die in jeder Formel F aus SAT mit Teilformeln a und b jedes Vorkommen von

   - der Negation $\neg a$ durch $\mathsf{if}\ a\ \mathsf{then}\ 0\ \mathsf{else}\ 1$,
   - der Konjunktion $a \wedge b$ durch $\mathsf{if}\ a\ \mathsf{then}\ b\ \mathsf{else}\ 0$ und
   - der Disjunktion $a \vee b$ durch $\mathsf{if}\ a\ \mathsf{then}\ 1\ \mathsf{else}\ b$

   ersetzt. Dann ist f total und in polynomieller Zeit berechenbar. Da SAT bereits NP-hart ist, ist somit auch $ISAT$ NP-hart.

2. Wenn $\mathsf{if}\ a\ \mathsf{then}\ b\ \mathsf{else}\ c$ durch $(a \wedge b) \vee (\neg a \wedge b)$ ersetzt wird, verdoppelt sich das Vorkommen von a. Dadurch kann die Größe der Formel exponentiell ansteigen und damit ist die Reduktion nicht polynomiell.

   Beispiel (nicht gefordert): Wir definieren die Formel f_n für $n \in \mathbb{N}$ wie folgt:

   $$f_0 = \mathsf{if}\ a\ \mathsf{then}\ 0\ \mathsf{else}\ 0 \qquad \text{und} \qquad f_{n+1} = \mathsf{if}\ f_n\ \mathsf{then}\ 0\ \mathsf{else}\ 0$$

   Dann kommt a in der SAT-Formel zu f_{n+1} doppelt so häufig vor wie in der SAT-Formel zu f_n. Die Formel f_{n+1} ist aber nur um eine Konstante größer als f_n.