Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Probeklausur zu „Berechenbarkeit und Komplexität“ WS 00/01

Klausurtermine:
Übungsscheinklausur: Mo, 12.2.01, 17-19 Uhr
Prüfungsklausur für technische Redakteure: voraussichtlich Mo, 12.3.01, 14-17 Uhr
Prüfungsklausur für Vordiplom Informatik: Do, 29.3.01, 9-12 Uhr

Aufgabe 1
Geben Sie detailliert und mit Erläuterung eine Turingmaschine an, welche die Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ mit $f(w) = 0w1$ berechnet.

Aufgabe 2

(a) Sei $f : \Sigma^*_{\text{bool}} \longrightarrow \Sigma^*_{\text{bool}}$ total und berechenbar. Zeigen Sie, dass der Bildbereich $f(\Sigma^*_{\text{bool}})$ von f semi-entscheidbar ist.

(b) Zeigen Sie die gleiche Behauptung für den Fall einer partiellen berechenbaren Funktion $f : \Sigma^*_{\text{bool}} - \to \Sigma^*_{\text{bool}}$.

Aufgabe 3
Gegeben seien folgende LOOP-Programme:

P_1:
```
X3 := X1 + X2;
X3 := X3 - 1;
X2 := X2 - 1;
X1 := X1 - 1
```

P_2:
```
loop X1
    X4 := X4 + X2
end;
```

P:
```
if X1 > 0 then
   if X2 > 0 then
      P1;
      P2;
      X1 := X3 + X4
   else X1 := 0 end
else X1 := 0 end
```

Seien $k_1, k_2, k_3, k_4 \in \mathbb{N}$. Bestimmen Sie $[\![P_1]\!](k_1, k_2, k_3, k_4, 0, \ldots)$ und $[\![P_2]\!](k_1, k_2, k_3, k_4, 0, \ldots)$, und geben Sie $f_P^{(2)}$ an.

Aufgabe 4
Geben Sie eine geeignete Übersetzung an, um REPEAT-Programme in WHILE-Programme zu überführen: Definieren Sie induktiv über den Aufbau der REPEAT-Programme P das jeweilige WHILE-Programm P'.
Erinnerung: Das repeat-Statement lautet `repeat P until Xi = 0`.

Aufgabe 5
Es sei $\underline{TOTAL}$ das folgende Entscheidungsproblem:
Gegeben: Turingmaschine M über Σ_{bool}.
Frage: Stoppt M für alle Eingaben?
Zeigen Sie $\underline{H} \leq \underline{TOTAL}$.

Aufgabe 6
Begründen Sie anhand der Arbeitsweise einer geeigneten Turingmaschine, dass die Spra
$\{0^k 1^k 2^k \mid k \in \mathbb{N}\}$ über dem Alphabet $\Sigma = \{0, 1, 2\}$ in $\texttt{TIME}(n^2)$ ist.

Aufgabe 7
In der Variante PCP′ vom PCP gibt es zusätzlich eine Zahl $l \in \mathbb{N}$ als Eingabe. Die Frage dann, ob es eine Lösung der Länge $\leq l$ gibt.

(a) Geben Sie eine genaue Formulierung des PCP und der Variante PCP′.

(b) Zeigen Sie, dass das PCP′ entscheidbar ist.

Abgabe: **freigestellt (zählt nicht für den Übungsschein)**
ggf. am Dienstag, 9.1.2001, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Lösungen zur Probeklausur zu „Berechenbarkeit und Komplexität“ WS 00/01

Aufgabe 1
Geben Sie detailliert und mit Erläuterung eine Turingmaschine an, welche die Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ mit $f(w) = 0w1$ berechnet.

Lösung zu 1: Drei verschiedene Zustände sind für die Turingmaschine nötig:

- Der Anfangszustand q_0 beschreibt zugleich den Zustand “gehe weiter nach rechts bis zum ersten Blank (hinter dem Eingabewort)”;
- Der Zustand q_1 wird angenommen, wenn das erste Blank direkt hinter dem Eingabewort erreicht wurde und dort die 1 geschrieben wird, und bedeutet “gehe weiter nach links bis zum ersten Blank (vor dem Eingabewort)”;
- Der Endzustand q_s wird angenommen, wenn das erste Blank direkt vor dem Eingabewort erreicht wurde und dort die 0 geschrieben wird, da dann die Turingmaschine M stoppen soll.

Wir erhalten die Turingmaschine $M = (\{q_0, q_1, q_s\}, \Sigma_{\text{bool}}, \Sigma_{\text{bool}} \cup \{\sqcup\}, q_0, q_s, \delta)$, wobei wir die Transitionsfunktion δ durch die folgenden Turingzeilen angeben:

q_0	0	0	R	q_0	($\sqcup$ hinter der Eingabe noch nicht
q_0	1	1	R	q_0	erreicht, also weiter nach rechts laufen.)
q_0	$\sqcup$	1	N	q_1	($\sqcup$ durch 1 überschreiben)
q_1	0	0	L	q_1	($\sqcup$ vor der Eingabe noch nicht
q_1	1	1	L	q_1	erreicht, also weiter nach links laufen.)
q_1	$\sqcup$	0	N	q_s	($\sqcup$ durch 0 ersetzen, stoppen)

Beispiel-Konfigurationsfolge (gehört *nicht* zur Lösung der Aufgabe!):
$q_0 1100 \vdash_M 1q_0100 \vdash_M 11q_000 \vdash_M 110q_00$
$\vdash_M 1100q_0\sqcup \vdash_M 1100q_11 \vdash_M 110q_101 \vdash_M 11q_1001$
$\vdash_M 1q_11001 \vdash_M q_111001 \vdash_M q_1\sqcup 11001 \vdash_M q_s011001$

Aufgabe 2

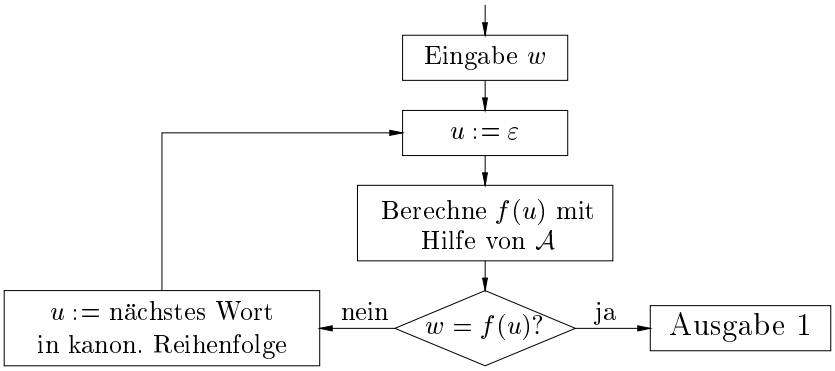
(a) Sei $f : \Sigma^*_{\text{bool}} \longrightarrow \Sigma^*_{\text{bool}}$ total und berechenbar. Zeigen Sie, dass der Bildbereich $f(\Sigma^*_{\text{bool}})$ von f semi-entscheidbar ist.

(b) Zeigen Sie die gleiche Behauptung für den Fall einer partiellen berechenbaren Funktion $f : \Sigma^*_{\text{bool}} - \to \Sigma^*_{\text{bool}}$.

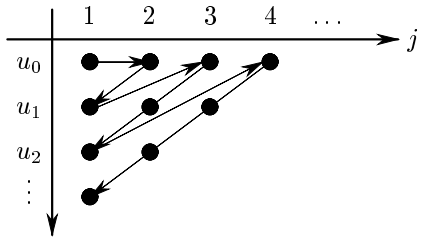
Lösung zu 2: (a): $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ ist nach Voraussetzung total und berechenbar. Es existiert also ein Algorithmus $\mathcal{A}$, der für alle Eingabewörter $u \in \Sigma^*_{\text{bool}}$ mit der Ausgabe $f(u)$ terminiert.
Zu zeigen ist, dass der Bildbereich $f(\Sigma^*_{\text{bool}})$ von f, also die Menge $\{v \in \Sigma^*_{\text{bool}} \mid \exists u \in \Sigma^*_{\text{bool}} : v = f(u)\}$, semi-entscheidbar ist. Dies wiederum bedeutet, dass es einen Algorithmus $\mathcal{A}'$ gibt, der $f(\Sigma^*_{\text{bool}})$ semi-entscheidet.
$\mathcal{A}'$ arbeitet auf der Eingabe w wie folgt: Wir prüfen mit Hilfe von $\mathcal{A}$ alle Eingabewörter $u \in \Sigma^*_{\text{bool}}$ in kanonischer Reihenfolge durch, bis wir ein Wort u mit $f(u) = w$ gefunden haben. Existiert ein solches Wort u, so werden wir es auf diese Art finden und lassen dann $\mathcal{A}'$ mit 1 terminieren. Existiert kein solches Wort u, so werden wir unendlich lange weitersuchen, d. h. $\mathcal{A}'$ terminiert nicht.

Damit ist gezeigt, dass der Bildbereich von f semi-entscheidbar ist.

Anmerkung zur Lösung:
$\mathcal{A}'$ kann alternativ auch graphisch dargestellt werden:



(b): Mit den gleichen Überlegungen wie im Teil (a) müssen wir auch in diesem Fall zu einer Eingabe w alle möglichen Wörter $u \in \Sigma^*_{\text{bool}}$ unter Verwendung von $\mathcal{A}$ daraufhin überprüfen, ob $f(u) = w$ gilt. Da es aber Wörter gibt, für die $\mathcal{A}$ nicht stoppt, müssen wir ein etwas anderes Verfahren anwenden, um alle Wörter testen zu können.
Für jedes zu prüfende Eingabewort u_i wird $\mathcal{A}$ j Schritte lang laufen gelassen für alle Kombinationen aus u_i und j, wie die folgende Abbildung veranschaulicht.



Man geht dabei wieder in kanonischer Reihenfolge vor. Terminiert die Berechnung von $\mathcal{A}$ für eine Eingabe u_i, so vergleicht man den Ausgabewert mit w.
Gilt $w = f(u)$ für ein $u \in \Sigma^*_{\text{bool}}$, so gibt es eine Schrittzahl j, so dass $\mathcal{A}$ bei Eingabe von u nach j Schritten mit der Ausgabe w terminiert. Mit oben genanntem Verfahren wird diese Kombination aus Eingabewort und Schrittzahl in endlicher Zeit erreicht. In diesem Fall gibt der neue Algorithmus dann also 1 aus.
Gibt es kein $u \in \Sigma^*_{\text{bool}}$ mit $w = f(u)$, so kann auch keine solche Kombination aus u und Schrittzahl j gefunden werden. Der neue Algorithmus terminiert dann also nicht.
Damit ist gezeigt, dass der Bildbereich von f semi-entscheidbar ist.

Aufgabe 3
Gegeben seien folgende LOOP-Programme:

P_1:
```
X3 := X1 + X2;
X3 := X3 - 1;
X2 := X2 - 1;
X1 := X1 - 1
```

P_2:
```
loop X1
    X4 := X4 + X2
end;
```

P:
```
if X1 > 0 then
    if X2 > 0 then
      P1;
      P2;
      X1 := X3 + X4
    else X1 := 0 end
else X1 := 0 end
```

Seien $k_1, k_2, k_3, k_4 \in \mathbb{N}$. Bestimmen Sie $[\![P_1]\!](k_1, k_2, k_3, k_4, 0, \ldots)$ und $[\![P_2]\!](k_1, k_2, k_3, k_4, 0, \ldots)$, und geben Sie $f_P^{(2)}$ an.

Lösung zu 3: Es ist

$$[\![P_1]\!](k_1, k_2, k_3, k_4, 0, \ldots) = (k_1 - 1, k_2 - 1, k_1 + k_2 - 1, k_4, 0, \ldots),$$
$$[\![P_2]\!](k_1, k_2, k_3, k_4, 0, \ldots) = (k_1, k_2, k_3, k_4 + k_1 \cdot k_2, \ldots).$$

Durch Ausrechnen bekommt man heraus

$$f_P^{(2)}(k_1, k_2) = \begin{cases} 0 \text{ falls } k_1 = 0 \text{ oder } k_2 = 0 \\ k_1 + k_2 - 1 + (k_1 - 1)(k_2 - 1) \text{ sonst.} \end{cases}$$

Etwas einfacher also $f_P^{(2)}(k_1, k_2) = k_1 \cdot k_2$.

Aufgabe 4
Geben Sie eine geeignete Übersetzung an, um REPEAT-Programme in WHILE-Programme zu überführen: Definieren Sie induktiv über den Aufbau der REPEAT-Programme P das jeweilige WHILE-Programm P'.
Erinnerung: Das repeat-Statement lautet `repeat P until Xi = 0`.

Lösung zu 4:

Induktionsanfang: Handelt es sich bei dem REPEAT-Programm P um eine Wertzuweisung, so ist $P' = P$ ein zu P äquivalentes WHILE-Programm.

Induktionsschritt: Ist P von der Form

(a) $P_1;P_2$,

(b) `if Xi > 0 then` P_1 `else` P_2 `end`,

(c) `loop Xi begin` P_1 `end` oder

(d) `repeat` P_1 `until Xi = 0`,

so existieren nach Induktionsvoraussetzung zu P_1, P_2 äquivalente WHILE-Programme P_1', P_2'. Dann ist

(a) $P_1';P_2'$,

(b) `if Xi > 0 then` P_1' `else` P_2' `end` oder

(c) `loop Xi begin` P_1' `end`

(d) P_1' ; `while Xi>0 do` P_1' `end`

jeweils ein zu P äquivalentes WHILE-Programm.

Aufgabe 5
Es sei $\underline{TOTAL}$ das folgende Entscheidungsproblem:
Gegeben: Turingmaschine M über Σ_{bool}.
Frage: Stoppt M für alle Eingaben?
Zeigen Sie $\underline{H} \leq \underline{TOTAL}$.

Lösung zu 5: Es ist eine berechenbar Funktion f anzugeben, die Turingmaschinen auf Turingmaschinen abbildet, so dass für alle Turingmaschinen M über Σ_{bool} gilt

$$M : \varepsilon \longrightarrow \text{ stop } \Leftrightarrow \forall w \in \Sigma_{\text{bool}}^* : f(M) : w \longrightarrow \text{ stop.}$$

Sei M eine Turingmaschine über Σ_{bool}. Die Maschine $f(M)$ arbeitet wie folgt: $f(M)$ löscht die Eingabe und arbeitet dann (auf dem jetzt leeren Band) so wie M.
Die Funktion f ist berechenbar. Es bleibt zu zeigen:

$$M : \varepsilon \longrightarrow \text{ stop } \Leftrightarrow \forall w \in \Sigma_{\text{bool}}^* : f(M) : w \longrightarrow \text{ stop.}$$

Wenn M auf dem leeren Band stoppt, dann stoppt $f(M)$ für jede Eingabe, da $f(M)$ den Input zunächst löscht und dann wie M arbeitet.
Wenn M auf dem leeren Band nicht stoppt, dann stoppt $f(M)$ auch nicht auf dem leeren Band, da $f(M)$ sich auf dem leeren Band so verhält wie M. Also stoppt $f(M)$ nicht für jede Eingabe.

Aufgabe 6
Begründen Sie anhand der Arbeitsweise einer geeigneten Turingmaschine, dass die Sprache $\{0^k1^k2^k \mid k \in \mathbb{N}\}$ über dem Alphabet $\Sigma = \{0, 1, 2\}$ in $\texttt{TIME}(n^2)$ ist.

Lösung zu 6: Die Arbeitsweise einer Turingmaschine, die die Sprache $\{0^k1^k2^k \mid k \in \mathbb{N}\}$ entscheidet, lässt sich in folgende Phasen aufteilen. Die Eingabe sei $w \in \{0,1,2\}^*$.

1. Ist $w = \varepsilon$, dann Ausgabe 1.
2. Wenn in w eines der Segmente 02, 10, 20, 21 vorkommt, dann Ausgabe 0.
3. Wiederhole die folgenden Schritte.

   (a) Gehe an den Anfang von w.
   (b) Durchlaufe w und ersetze jeweils eine 0, eine 1 und eine 2 durch \$.

   Bis keine 0 oder keine 1 oder keine 2 mehr auf dem Band steht.
4. Wenn noch eine 0, eine 1 oder eine 2 auf dem Band steht, dann Ausgabe 0, sonst Ausgabe 1.

Korrektheit: Nach der Phase 2 ist klar, dass w die Form $0^k1^l2^m$ hat. Nach der Phase 3 sind in der Eingabe jeweils gleich viele Nullen, Einsen und Zweien durch \$ ersetzt worden. Stehen danach nur noch \$ auf dem Band, dann war die Eingabe korrekt, ansonsten waren die Anzahlen der Buchstaben nicht gleich.

Hat die Eingabe w die Länge n, dann haben die verschiedenen Phasen die folgenden Laufzeiten.

1. Test, ob der Lesekopf zu Beginn auf einem Blank steht: $\mathcal{O}(1)$.
2. Einmaliges Durchlaufen der Eingabe: $\mathcal{O}(n)$.
3. Maximal n Durchläufe der Schleife. Bei jedem Schleifendurchlauf einmaliges Durchlaufen der Eingabe: $\mathcal{O}(n^2)$.
4. Einmaliges Durchlaufen der Eingabe: $\mathcal{O}(n)$.

Also ergibt sich insgesamt eine Laufzeit von $\mathcal{O}(n^2)$.

Aufgabe 7

In der Variante PCP$'$ vom PCP gibt es zusätzlich eine Zahl $l \in \mathbb{N}$ als Eingabe. Die Frage ist dann, ob es eine Lösung der Länge $\leq l$ gibt.

(a) Geben Sie eine genaue Formulierung des PCP und der Variante PCP$'$.

(b) Zeigen Sie, dass das PCP$'$ entscheidbar ist.

Lösung zu 7:

(a): Formulierung des PCP:

Gegeben: Zwei Wortlisten $(u_1, \ldots, u_n)$, $(v_1, \ldots, v_n)$.

Frage: Existiert eine Indexfolge $(i_1, \ldots, i_k)$ über $\{1, \ldots, n\}$ mit $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$?

Formulierung des PCP$'$:

Gegeben: Zwei Wortlisten $(u_1, \ldots, u_n)$, $(v_1, \ldots, v_n)$ und ein $l \in \mathbb{N}$.

Frage: Existiert eine Indexfolge $(i_1, \ldots, i_k)$ über $\{1, \ldots, n\}$ mit $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$ und $k \leq l$?

(b): Es gibt nur endlich viele Indexfolgen der Länge $\leq l$ über $\{1, \ldots, n\}$. Für alle dieser möglichen Indexfolgen $(i_1, \ldots, i_k)$ mit $k \leq l$ kann man testen, ob Sie die Bedingung $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$ erfüllen.

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Scheinklausur zu „Berechenbarkeit und Komplexität“ WS 00/01

Beachten Sie die Hinweise auf dem Deckblatt.

Aufgabe 1 *6 Punkte*

Geben Sie detailliert und mit Erläuterung eine Turingmaschine an, welche die Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ mit

$$f(w) = \begin{cases} w & \text{falls } w \text{ eine 1 enthält,} \\ \varepsilon & \text{sonst} \end{cases}$$

berechnet.

Aufgabe 2 *6 Punkte*

Zeigen Sie: Wenn eine Wortmenge $W \subseteq \Sigma^*_{\text{bool}}$ aufzählbar ist, so ist sie der Definitionsbereich einer berechenbaren Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$. (Für den Algorithmus $\mathcal{A}_f$, der f berechnet, muss also gelten: $w \in W$ gdw. $\mathcal{A}_f$ terminiert bei Eingabe w.)

Aufgabe 3 *6 Punkte*

Seien $f_1, f_2 : \mathbb{N} \to \mathbb{N}$ LOOP-berechenbar. Zeigen Sie, dass auch $f : \mathbb{N} \to \mathbb{N}$ mit $f(n) = f_2(f_1(n))$ LOOP-berechenbar ist.

Aufgabe 4 *6 Punkte*

Geben Sie eine geeignete Übersetzung an, um WHILE-Programme in REPEAT-Programme zu überführen: Definieren Sie induktiv über den Aufbau der WHILE-Programme P das jeweilige REPEAT-Programm P'. Führen Sie den Induktionsschritt nur für die Verkettung von Programmen und für das while-Statement aus.

Erinnerung: Das repeat-Statement lautet `repeat P until Xi = 0` und das while-Statement lautet `while Xi > 0 do P end`.

Aufgabe 5 *6 Punkte*

Wir betrachten das „Wertproblem“ $\underline{WeP}$ für Turingmaschinen:

Gegeben: Turingmaschine M über Σ_{bool} und $u, v \in \Sigma^*_{\text{bool}}$.
Frage: Liefert M für die Eingabe u die Ausgabe v?

Zeigen Sie $\underline{WP} \leq \underline{WeP}$.

Erinnerung: Beim „Wortproblem“ $\underline{WP}$ ist eine Turingmaschine M über Σ_{bool} und $w \in \Sigma^*_{\text{bool}}$ gegeben. Die Frage ist, ob M auf der Eingabe w stoppt.

Aufgabe 6 *6 Pun*

Zeigen Sie, dass das folgende Problem entscheidbar ist.

Gegeben: Turingmaschine M über Σ_{bool}, die auf einer Eingabe der Länge n höchstens viele Felder des Bandes besucht, und ein $w \in \Sigma^*_{\text{bool}}$.
Frage: Stoppt M auf w?

Hinweis: Es sei $M = (Q, \Sigma, \Gamma, q_0, q_s, \delta)$. Bestimmen Sie zunächst die Höchstanzahl der mö chen Konfigurationen in Abhängigkeit von $|Q|$, $|\Gamma|$ und $n = |w|$.

Aufgabe 7 *7 Pun*

Skizzieren Sie in ca. 10 Zeilen die Arbeitsweise einer Turingmaschine M, die die Funkt $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ mit $f(w) = w^{\text{R}}$ berechnet (dabei ist w^{R} das Wort w umgedreht). Begründen Sie anhand dieser Turingmaschine M, dass f in $\texttt{TIME}(n^2)$ ist.

Aufgabe 8 *7 Pun*

Das Problem Vertex Cover (VC) ist wie folgt definiert:

Gegeben: Ungerichteter Graph $G = (V, E)$, $k \geq 1$ (in Binärdarstellung).
Frage: Existiert $U \subseteq V$ mit k Knoten, so dass jede Kante von G wenigstens einen E punkt in U hat?

(a) Skizzieren Sie in 5 - 10 Zeilen die Arbeitsweise einer nichtdeterministischen polynomz beschränkten Turingmaschine, die VC entscheidet. (Auf dem Band sei eine Beschreib von G in Form der Liste der Knotennamen und der Liste der Kanten vorgegeben.)

(b) Man kann die NP-Vollständigkeit von VC zeigen, indem man 3SAT $\leq_p$ VC nachwe Formulieren Sie genau, was 3SAT $\leq_p$ VC bedeutet (nur die Definition ist gefragt).

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Anwesenheitsübung

A 1
Seien $l, m \in \mathbb{N}$ mit $m \geq 2$. Zeigen Sie, dass es

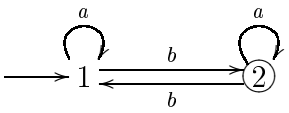
(a) m^l Wörter der Länge $= l$ und

(b) $\frac{m^{l+1}-1}{m-1}$ Wörter der Länge $\leq l$

über dem Alphabet Σ_m gibt.

A 2
Sei $m \geq 1$. Eine Kodierung der Wörter über Σ_m durch Wörter über Σ_{bool} ist eine Funktion $f : \Sigma_m \to \Sigma_{\text{bool}}^*$ derart, dass für alle $a_1, \ldots, a_k, b_1, \ldots, b_n \in \Sigma_m$ $(k, n \geq 0)$ gilt: Wenn $f(a_1) \cdots f(a_k) = f(b_1) \cdots f(b_n)$, so $k = n$ und $a_i = b_i$ für alle $i \leq n$.
Geben Sie verschiedene Kodierungen f_m von Σ_m durch Wörter über Σ_{bool} an und zeigen Sie, dass es sich bei f_m um eine Kodierung handelt.

A 3
Es sollen „Transitionsgraphen“ wie in folgendem Beispiel kodiert werden.



Beschreiben Sie über einem geeigneten Alphabet ein Kodierungssschema durch Wörter und geben Sie für das Beispiel die Kodierung an.

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 1

Aufgabe 1 *4 Punkte*

Wir betrachten das Alphabet $\Sigma_3 = \{\underline{1}, \underline{2}, \underline{3}\}$.

(a) Bestimmen Sie die Werte $\gamma_3(\underline{1}\,\underline{2})$ und $\gamma_3(\underline{3}\,\underline{1}\,\underline{3}\,\underline{2})$.

(b) Bestimmen Sie die Wörter $\delta_3(12)$ und $\delta_3(79)$.

Aufgabe 2 *4 Punkte*

Die lexikographische Ordnung $<_{\text{lex}}$ auf den nichtleeren Wörtern über einem geordnenten Alphabet geht wie in einem Lexikon vor.

(a) Geben Sie eine präzise Definition für den Fall des Alphabets Σ_m ($m \geq 1$). Es ist festzulegen, wann für zwei Wörter $a_1 \dots a_l$, $b_1 \dots b_k$ (mit $k, l > 0$) über Σ_m

$$a_1 \dots a_l <_{\text{lex}} b_1 \dots b_k$$

gilt.

(b) Geben Sie über Σ_2 eine unendliche absteigende Kette

$$w_0 >_{\text{lex}} w_1 >_{\text{lex}} w_2 >_{\text{lex}} \cdots$$

an.

Abgabe: Dienstag, 24.10.2000

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 2

Aufgabe 3 *4 Punkte*

Eine Wortmenge $W \subseteq \Sigma^*$ heißt wiederholungsfrei aufzählbar, wenn es einen Aufzählungsalgorithmus gibt, der die Elemente von W jeweils genau einmal ausgibt. Zeigen Sie:

$$W \text{ aufzählbar} \Rightarrow W \text{ wiederholungsfrei aufzählbar.}$$

Aufgabe 4 *4 Punkte*

Eine Wortmenge $W \subseteq \Sigma^*$ heißt monoton aufzählbar, wenn es einen Aufzählungsalgorithmus gibt, der die Elemente von W gemäß kanonischer Reihenfolge (insbesondere wiederholungsfrei) ausgibt. Zeigen Sie (für den Fall, dass W unendlich ist):

$$W \text{ entscheidbar} \Leftrightarrow W \text{ monoton aufzählbar.}$$

Aufgabe 5 *4 Punkte*

Wir betrachten Modula3-Programme mit Integervariablen unter der Vorgabe, dass deren Werte durch eine Zahl k beschränkt sind (etwa gegeben durch die Längenbeschränkung der Bitwörter für die Zahlendarstellung). Alle arithmetischen Funktionen seien dennoch immer definiert (bei Wertebereichsüberschreitung sei etwa der Wert 0 vereinbart).
Zeigen Sie, dass folgendes Problem entscheidbar ist

Gegeben: Modula3-Programm P nur mit Integervariablen für Werte vom Betrag $\leq k$.

Frage: Terminiert P bei Initialisierung der Variablen mit 0?

Abgabe: Dienstag, 31.10.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 3

Aufgabe 6 *4 Punkte*

Für $a \in \Sigma_{\text{bool}}$ sei $\overline{a} = 0$ falls $a = 1$ und $\overline{a} = 1$ falls $a = 0$. Geben Sie (mit Erklärung) eine Turingmaschine an, die die Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ mit $f(a_1 \cdots a_l) = \overline{a_1} \cdots \overline{a_l}$ berechnet. Geben Sie die Konfigurationsfolge Ihrer Turingmaschine auf der Eingabe 001 an.

Aufgabe 7 *4 Punkte*

In dieser Aufgabe soll gezeigt werden, dass man auf die Möglichkeit des „N“ (der Kopf wird nicht bewegt) in der Transitionsfunktion einer Turingmaschine verzichten kann. Geben Sie also an, wie man zu einer gegebenen Turingmaschine $M = (Q, \Sigma, \Gamma, q_0, q_s, \delta)$ eine äquivalente Turingmaschine $M' = (Q', \Sigma, \Gamma', q'_0, q'_s, \delta')$ erhält (d.h. M' soll die gleiche Funktion wie M berechnen), deren Transitionsfunktion die Form $\delta' : Q' \times \Gamma' \to \Gamma' \times \{L, R\} \times Q'$ hat.

Aufgabe 8 *4 Punkte*

Zeigen Sie (auf intuitiver Ebene):
Eine Wortmenge $W \subseteq \Sigma^*$ ist aufzählbar genau dann, wenn sie semi-entscheidbar ist.

Abgabe: Dienstag, 7.11.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 4

Aufgabe 9 *4 Punkte*

Wir betrachten die Funktion $ggT : \mathbb{N}^2 - \to \mathbb{N}$, mit $ggT(x, y) =$ größter gemeinsamer Teiler von x und y (wobei $ggT(0, y) = ggT(x, 0) = \perp$ vereinbart sei). Geben Sie als Formulierung des „Euklidischen Algorithmus“ ein WHILE-Programm an, das ggT berechnet.

Aufgabe 10 *4 Punkte*

Zeigen Sie, dass folgende Funktionen LOOP-berechenbar sind:

(a) $f_1 : \mathbb{N} \to \mathbb{N},\ f_1(n) = \sum_{i=1}^{n} i,$

(b) $f_2 : \mathbb{N} \to \mathbb{N},\ f_2(n) = \lfloor \sqrt{n} \rfloor.$

Aufgabe 11 *4 Punkte*

Wir betrachten „LOOP-freie additive Programme“. Diese sind WHILE-Programme, welche kein `loop`, kein `while`, und als zweistellige Operationen in Wertzuweisungen nur $+$ und $-$ enthalten. Zeigen Sie: Es gibt eine totale LOOP-berechenbare Funktion, die durch kein LOOP-freies additives Programm berechenbar ist.
Hinweis: Ihr Beweis sollte folgendes enthalten:

1. eine Funktion f mit zugehörigem LOOP-Programm P,
2. eine Eigenschaft E, die jede Funktion g besitzt, welche durch ein LOOP-freies additives Programm berechnet wird, die aber auf f nicht zutrifft.

Abgabe: Dienstag, 14.11.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 5

Aufgabe 12 *4 Punkte*

Geben Sie WHILE- oder LOOP-Programme zur Berechnung der folgenden Funktionen an:

(a) $fib : \mathbb{N} \to \mathbb{N}$ mit $fib(0) = fib(1) = 1$ und $fib(n+2) = fib(n) + fib(n+1)$ für alle $n \in \mathbb{N}$,

(b) $f : \mathbb{N} \to \mathbb{N}$ mit $f(n) = 2^{(2^n)}$ für alle $n \in \mathbb{N}$.

Aufgabe 13 *4 Punkte*

Zeigen Sie, dass man bei WHILE-Programmen auch ohne das `if .. then .. else`-Konstrukt auskommt. Geben Sie dazu eine geeignete Ersetzung an, mit der man `if .. then .. else`-Konstrukte aus WHILE-Programmen eliminieren kann.

Aufgabe 14 *4 Punkte*

Sei $f : \mathbb{N} \to \mathbb{N}$ LOOP-berechenbar, und sei $F : \mathbb{N}^2 \to \mathbb{N}$, $F(x, y) = f^y(x)$ (d.h. die y-fache Iteration von f, angewandt auf x).
Zeigen Sie, dass F LOOP-berechenbar ist.

Abgabe: Dienstag, 21.11.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 6

Aufgabe 15 *4 Punkte*

Zeigen Sie, dass $f : \mathbb{N} \to \mathbb{N}$,

$$f(n) = \begin{cases} 1 \text{ falls } n \text{ Primzahl ist,} \\ \perp \text{ sonst} \end{cases}$$

WHILE-berechenbar ist. Kommentieren Sie Ihr Programm, nennen Sie insbesondere die Zwecke der benutzten Variablen.

Aufgabe 16 *4 Punkte*

(a) Sei A eine Menge und $R \subseteq A \times A$. Für alle $a \in A$ definieren wir $R_a := \{b \in A \mid (a, b) \in R\}$.

Zeigen Sie, dass $D := \{b \in A \mid (b, b) \notin R\}$ verschieden ist von allen R_a mit $a \in A$.

(b) Sei $R = \{(u, v) \in \Sigma_{\text{bool}} \times \Sigma_{\text{bool}} \mid u \text{ ist nicht Kodewort einer TM, die angesetzt auf } v \text{ stoppt}\}$.

Zeigen Sie: Für jede entscheidbare Wortmenge L über Σ_{bool} gibt es ein $u \in \Sigma_{\text{bool}}$ mit $L = R_u$. (Hinweis: Ist L entscheidbar, so ist auch das Komplement von L entscheidbar und somit insbesondere semi-entscheidbar.)

Sei D definiert wie in (a). Beschreiben Sie D möglichst einfach. Folgern Sie aus (a), dass D nicht Turing-entscheidbar ist.

Aufgabe 17 *4 Punkte*

REPEAT-Programme seien induktiv analog den WHILE-Programmen aufgebaut, jedoch mit Verwendung folgender Klausel an Stelle der Einführung der while-Schleife: Ist P_1 REPEAT-Programm, so auch `repeat` P_1 `until Xi = 0 end`.
Das Programm P_1 wird also so oft durchgeführt (mindestens einmal), bis `Xi` den Wert 0 hat.
Zeigen Sie folgende Behauptung: Für jedes REPEAT-Programm P gilt: Ist $m \geq maxind(P)$, so existiert ein WHILE-Programm P' mit

$$[\![P]\!](k_1, \ldots, k_m, 0, \ldots) = [\![P']\!](k_1, \ldots, k_m, 0, \ldots).$$

Führen Sie den Beweis durch Induktion über den Aufbau der REPEAT-Programme. Den Induktionsschritt der Konstruktion von P' sollten Sie für das repeat-Konstrukt und ein weiteres (Verkettung oder if-then-else oder loop) genau formulieren.

Abgabe: Dienstag, 28.11.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 7

Aufgabe 18 *4 Punkte*

Betrachten Sie folgende zwei Entscheidungsprobleme:

(P1) Gegeben: TM M über $\{0,1\}$, Wort $w \in \{0,1\}^*$, Zahl $n \geq 0$.
Frage: Stoppt M, angesetzt auf w, nach $\leq n$ Schritten?

(P2) Gegeben: TM M über $\{0,1\}$, Wort $w \in \{0,1\}^*$, Zahl $n \geq 0$.
Frage: Stoppt M, angesetzt auf w, nach $> n$ Schritten?

Klären Sie (mit Beweis), ob (P1) bzw. (P2) entscheidbar ist.

Aufgabe 19 *4 Punkte*

Sei $\underline{H}_{111}$ das folgende Entscheidungsproblem:
Gegeben: TM M über $\{0,1\}$.
Frage: Akzeptiert M das Wort 111?
Zeigen Sie: $\underline{H}_{111} \leq \underline{H}$ und $\underline{H} \leq \underline{H}_{111}$. (Hierbei ist $\underline{H}$ das Halteproblem im Sinne der Vorlesung.)

Aufgabe 20 *4 Punkte*

Eine Turingmaschine M über $\{0,1\}$ heißt eingabebeschränkt, wenn sie den Bandbereich des Eingabewortes (einschließlich der zwei Begrenzungsfelder) nicht verlässt. Man arbeitet hierzu mit den Sonderzeichen \$ und ¢ im Arbeitsalphabet, lässt die Maschine jeweils mit einer Anfangskonfiguration $¢q_0w\$$ beginnen und vereinbart, dass für die Endmarker ¢, \$ nur Turingzeilen der Form $q¢¢Rq'$ und $q\$\$Lq'$ zur Verfügung stehen.
Zeigen Sie, dass folgendes Problem entscheidbar ist:
Gegeben: Eingabebeschränkte Turingmaschine M über $\{0,1\}$ und Wort $w \in \{0,1\}^*$.
Frage: Stoppt M von Anfangskonfiguration $¢q_0w\$$ aus?
Hinweis: Wieviele verschiedene Konfigurationen kann es geben?

Abgabe: Dienstag, 5.12.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“
WS 00/01

Serie 8

Aufgabe 21 *4 Punkte*

Wir betrachten Entscheidungsprobleme der Form $\underline{P} = (\mathbb{N}, P)$ mit Eingabemenge $\mathbb{N}$. Für $\underline{P_1} = (\mathbb{N}, P_1)$ und $\underline{P_2} = (\mathbb{N}, P_2)$ sei $\underline{P_1} \oplus \underline{P_2} = (\mathbb{N}, P_1 \oplus P_2)$ mit

$$P_1 \oplus P_2 := \{2n \mid n \in P_1\} \cup \{2n+1 \mid n \in P_2\}.$$

Zeigen Sie, dass $\underline{P_1} \oplus \underline{P_2}$ ein „kleinstes Problem $\geq \underline{P_1}, \underline{P_2}$“ ist:

(a) $\underline{P_1} \leq \underline{P_1} \oplus \underline{P_2}$ und $\underline{P_2} \leq \underline{P_1} \oplus \underline{P_2}$.

(b) Für alle Entscheidungsprobleme $\underline{Q}$ mit Eingabemenge $\mathbb{N}$ gilt: Wenn $\underline{P_1} \leq \underline{Q}$ und $\underline{P_2} \leq \underline{Q}$, so $\underline{P_1} \oplus \underline{P_2} \leq \underline{Q}$.

Aufgabe 22 *4 Punkte*

Entscheiden Sie für die folgenden Beispiele (mit Begründung), ob das Postsche Korrespondenzproblem jeweils eine Lösung hat.

(a) $x = (bbb, abb)$ und $y = (bb, babbb)$.

(b) $x = (a, bba, aab)$ und $y = (ba, aaa, ba)$.

(c) $x = (a, aab, baaa)$ und $y = (aa, bb, a)$.

(d) $x = (bb, baa, ab, aa)$ und $y = (ab, ba, aa, aba)$.

Lösen Sie alternativ eine der Aufgaben 23 bzw. 23′.

Aufgabe 23 *4 Punkte*

Das eindimensionale Dominospiel bezieht sich auf Dominos, bei denen die zwei gegenüberliegenden Seiten jeweils mit einer „Farbe“ (hier einer natürlichen Zahl) gekennzeichnet sind. Ein Dominotyp d ist damit gegeben durch seine zwei Farben; man schreibt $d = (n_1, n_2)$. (Ein Domino ist gerichtet und darf nicht gedreht werden.) Ein Dominospiel ist eine endliche Menge $D = \{d_1, \ldots, d_k\}$ von Dominotypen; von jedem Typ stehen beliebig viele Dominos zur Verfügung. Eine Zeilenparkettierung mit D ist eine unendliche Folge $(m_0, m_1)(m_1, m_2)(m_2, m_3)(m_3, m_4)\cdots$ wobei jeder Dominotyp (m_i, m_{i+1}) zu D gehört.
Zeigen Sie durch Angabe eines Algorithmus, dass folgendes Problem entscheidbar ist.

Gegeben: Dominospiel D
Frage: Gibt es eine Zeilenparkettierung mit D?

Aufgabe 23′ *4 Punkte*

Sei PCP(1) ds Postsche Korrespondenzproblem für Wörter über dem einelementigen Alphabet $\{a\}$. Zeigen Sie, dass PCP(1) entscheidbar ist.

Abgabe: Dienstag, 12.12.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 9

Aufgabe 24 *5 Punkte*

Zeigen Sie durch Rückgriff auf die Definition:

(a) $6n^2 + 12n + 10 \in \mathcal{O}(n^2)$,

(b) $2^{n+10} \in \mathcal{O}(2^n)$.

Überprüfen Sie mit kurzer Begründung Ihrer Antwort die folgenden Aussagen:

(c) $n^2 \in \mathcal{O}(n \cdot \log^2 n)$,

(d) $n \cdot \log n \in \mathcal{O}(n^2)$,

(d) $3^n \in 2^{\mathcal{O}(n)}$.

Aufgabe 25 *4 Punkte*

Beschreiben Sie informell die Arbeitsweise einer Turingmaschine M (mit Angabe typischer Konfigurationen), die die Eingabe „verdoppelt“: $q_0 w \vdash_M^* q_s w \sqcup w$.
Geben Sie anhand dieser informellen Darstellung eine Laufzeitschranke (in $\mathcal{O}$-Notation) für M an.

Aufgabe 26 *3 Punkte*

Die Funktionen $f_1 : \{0,1\}^* \to \{0,1\}^*$ und $f_2 : \{0,1\}^* \to \{0,1\}^*$ seien durch polynomzeitbeschränkte Turingmaschinen berechenbar (etwa mit den Polynomzeitschranken $p_1(n), p_2(n)$). Zeigen Sie, dass die Verkettungsfunktion $f : \{0,1\}^* \to \{0,1\}^*$ mit $f(w) = f_2(f_1(w))$ polynomzeitberechenbar ist, und geben Sie ein geeignetes Polynom an.
Lösen Sie die Aufgabe unter der Annahme, dass Turingmaschinen bei Terminierung nur das Ausgabewort auf dem Band stehen haben.

Abgabe: Dienstag, 19.12.2000, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 10

Aufgabe 27 *4 Punkte*

Wir betrachten die Menge der Dezimaldarstellungen $d(n)$ natürlicher Zahlen n (ohne führende Nullen). Eine Zahl ≥ 2 heißt zusammengesetzt, wenn sie nicht prim ist. Sei Z die Wortmenge

$$Z = \{d(n) \mid n \text{ zusammengesetzt}\}.$$

Zeigen Sie, dass Z zu NP gehört

(a) durch grobe Skizzierung der Arbeitsweise einer geeigneten polynomzeitbeschränkten NTM,

(b) durch Formulierung eines polynomial größenbeschränkten Suchproblems, das die Frage „Gehört $d(n)$ zu Z“ kodiert.

Zusatzfrage (ohne Wertung): Können Sie zeigen, dass $Z \in$ P?

Aufgabe 28 *4 Punkte*

Wir betrachten zwei Wortmengen $K, L \subseteq \{0,1\}^*$. Die Wortmenge $K \cdot L$ enthält alle Wörter w, die sich zerlegen lassen in der Form $w = uv$ mit $u \in K$, $v \in L$.
K, L seien in P, d.h. in Polynomzeit entscheidbar. Zeigen Sie:

(a) $K \cdot L \in$ NP (durch Anwendung einer der Definitionen von NP).

(b) $K \cdot L \in$ P (durch grobe Skizzierung der Arbeitsweise einer geeigneten polynomzeitbeschränkten DTM).

Aufgabe 29 *4 Punkte*

Beim Quine-McCluskey-Verfahren zur Schaltkreisminimierung ergibt sich in der zweiten Phase ein „Überdeckungsproblem“: Aus einer 0-1 Matrix (mit wenigstens einer 1 in jeder Spalte) sind möglichst wenige Zeilen auszuwählen, so dass bei Restriktion auf diese Zeilen in jeder Spalte eine 1 steht (vgl. Oberschelp, Vossen, Rechneraufbau und Rechnerstrukturen, Abschnitt 2.4.2).
Wir betrachten folgendes Problem **(Ü)**:

Gegeben:	0-1 Matrix M mit Spalten $\neq$ Nullvektor, Zahl $k \geq 1$.
Frage:	Genügen k Zeilen zur Überdeckung von M?

Zeigen Sie mit Hilfe einer geeigneten Kodierung von M und k durch Wörter, dass **(Ü)** in NP liegt.

Abgabe: Dienstag, 16.1.2001, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 11

Aufgabe 30 *4 Punkte*

Die Sprache L_0 über dem Alphabet Σ sei weder $= \Sigma^*$ noch $= \emptyset$. Es gelte $L_0 \in \mathrm{P}$. Zeigen Sie

$$\mathrm{P} = \mathrm{NP} \Leftrightarrow L_0 \text{ ist NP-vollständig.}$$

Aufgabe 31 *4 Punkte*

Zeigen Sie, dass das Erfüllbarkeitsproblem für aussagenlogische Ausdrücke in DNF (disjunktiver Normalform) in Polynomzeit lösbar ist. (Ein Algorithmus in Pseudocode mit Begründung seiner Korrektheit genügt.)

Aufgabe 32 *4 Punkte*

Betrachten Sie das Problem PCP′ (Probeklausur, Aufgabe 7):

Gegeben: Zwei Wortlisten $(u_1, \ldots, u_n)$, $(v_1, \ldots, v_n)$ und eine Zahl $l \in \mathbb{N}$.
Frage: Existiert eine Indexfolge $i_1, \ldots, i_m$ mit $m \leq l$ und $u_{i_1} \cdots u_{i_m} = v_{i_1} \cdots v_{i_m}$?

(a) Geben Sie eine formale Sprache L' an, die PCP′ kodiert.

(b) Zeigen Sie, dass L' zu NP gehört (durch Anwendung einer Definition von NP Ihrer Wahl).

Alternative: Unterstellen Sie, dass (a) und (b) gezeigt sind. Zeigen Sie, dass PCP′ NP-vollständig ist.
Hinweis: Gehen Sie vom Unentscheidbarkeitsbeweis für PCP aus.

Abgabe: Dienstag, 23.1.2001, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 12

Aufgabe 33 *4 Punkte*

Wir betrachten das Problem 3SAT (für KNF-Formeln mit genau 3 Literalen pro Klausel). Geben Sie in Pseudocode einen deterministischen Algorithmus zur Lösung von 3SAT an, der in einem Backtracking-Verfahren aus jeder Klausel ein Literal sucht, das wahr gesetzt sein soll; hierbei werden die Literale einer Klausel $l_1 \vee l_2 \vee l_3$ in der Reihenfolge l_1, l_2, l_3 betrachtet.

Aufgabe 34 *4 Punkte*

Zeigen Sie, dass die Sprache $\{wcw \mid w \in \{0,1\}^*\}$ über dem Alphabet $\{0,1,c\}$ in DLOG ist.

Aufgabe 35 *4 Punkte*

Wir betrachten das Problem INDEPENDENT SET:

Gegeben: Ein ungerichteter Graph $G = (V, E)$ und ein $k \in \mathbb{N}$.

Frage: Gibt es eine Teilmenge U von V mit $|U| \geq k$ so, dass keine zwei Knoten aus U durch eine Kante verbunden sind?

Zeigen Sie: CLIQUE $\leq_p$ INDEPENDENT SET.

Abgabe: Dienstag, 30.1.2001, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Serie 13

Aufgabe 36 *4 Punkte*

Stellen Sie die Arbeitsweise des QBF-Auswertungsalgorithmus anhand der beiden Formeln

$$\phi_1 = \exists x_1 \forall x_2 \exists x_3 ((x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3})),$$

$$\phi_2 = \exists x_1 \forall x_2 \exists x_3 ((x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_2 \vee \overline{x_3}))$$

dar. Notieren Sie hierzu, für welche Formeln der Test-Algorithmus nacheinander aufgerufen wird und geben Sie eventuelle Auswertungsergebnisse an.

Aufgabe 37 *4 Punkte*

Wir betrachten das Problem aus Aufgabe 27, Serie 10 (Zusammengesetztheit von natürlichen Zahlen aus Faktoren $\neq 1$), nun jedoch für Zahlen in unärer Darstellung durch Strichfolgen. Sei also

$$Z_0 = \{|^k \mid k \geq 2 \text{ und } k \text{ zusammengesetzt}\}.$$

Zeigen Sie (durch grobe Beschreibung der Arbeitsweise einer Turingmaschine), dass $Z_0 \in$P.

Hinweis: 1. Teilbarkeit durch i lässt sich durch Verschiebung eines „Zeigers“ (zunächst auf Position i, dann $2i$ etc.) testen.

2. Alternativ: Sie dürfen auch eine polynomzeitbeschränkte Mehrband-Turingmaschine benutzen (die, wie in der Vorlesung erwähnt, durch eine polynomzeitbeschränkte Einband-Turingmaschine simulierbar ist).

Aufgabe 38 *4 Punkte*

Geben Sie in der Umgangssprache (und mit Rechtfertigung der Korrektheit) ein effizientes (d.h. polynomial zeitbeschränktes) Verfahren an, das zu jeder KNF-Formel mit höchstens 2 Literalen pro Klausel entscheidet, ob sie erfüllbar ist. (Eine formale Ausgestaltung durch eine Turingmaschine würde zeigen, dass 2SAT $\in$P ist.)

Abgabe: Dienstag, 6.2.2001, in der Vorlesung

Lehrstuhl für Informatik VII
RWTH Aachen
Prof. Dr. W. Thomas
Dipl. Inf. C. Löding

Übungen zu „Berechenbarkeit und Komplexität“ WS 00/01

Aufgaben für die „Technischen Redakteure“

Aufgabe 1 Die Funktion $f : \Sigma^*_{\text{bool}} \to \Sigma^*_{\text{bool}}$ sei wie folgt definiert: Für $w \in \Sigma^*_{\text{bool}}$ entstehe $f(w)$ durch Streichen der ersten beiden Buchstaben; hat w weniger als 2 Buchstaben, so sei $f(w)$ das leere Wort.
Geben Sie detailliert eine Turingmaschine an, die f berechnet.

Aufgabe 2 Die Turingmaschine M habe die Eigenschaft, dass sie für ein Eingabewort der Länge n jeweils höchstens $7n$ Felder des Rechenbandes besucht. Das Arbeitsalphabet Γ habe m Buchstaben.

(a) Wie viele mögliche Inschriften mit Γ gibt es auf einem Bandabschnitt der Länge $7n$? Wieviele verschiedene Konfigurationen kann es höchstens geben, die M für ein Eingabewort der Länge n erreicht?

(b) Beschreiben Sie, wie man für ein Eingabewort w entscheidet, ob M angesetzt auf w stoppt.

Aufgabe 3 Wir betrachten die Menge L der Wörter der Form wcw^{R} mit $w \in \{0,1\}^*$.

(a) Skizzieren Sie anhand des Beispielwortes $1101c1011$, wie eine Turingmaschine M arbeitet, die L entscheidet. (Es genügen ca. 10 Zeilen Erläuterung.)

(b) Begründen Sie, möglichst mit Hilfe der Antwort zu (a), dass die Sprache L durch eine n^2-zeitbeschränkte Turingmaschine entschieden wird.

Aufgabe 4 (a) Geben Sie die Definition von „NP-vollständig“ und von $L \leq_p L_0$ an (bezogen auf Sprachen über einem festen Alphabet Σ).

(b) Begründen Sie die folgende Behauptung: Wenn $L \leq_p L_0$ und $L_0 \in \text{P}$, dann gilt $L \in \text{P}$.

Literatur

Einführende Bücher

- W.S. Brainerd, L.H. Landweber: *Theory of Computation.* Wiley, 1974
  (Berechenbarkeitstheorie über Wörtern, ausführlich und umfassend)
- D. Harel: *Algorithmics - The Spirit of Computing.* Addison-Wesley, 1987
  (sehr lesenswerte, informelle Einführung!)
- H.R. Lewis, C. Padadimitriou: *Elements of the Theory of Computation.* Prentice-Hall, 1981
  (gut lesbares Lehrbuch zur Theoretischen Informatik)
- Z. Manna: *Mathematical Theory of Computation.* McGraw-Hill, New York, 1974
  (enthält auch Ergebnisse zur Theorie der Programmierung)
- R. McNaughton: *Elementary Computability, Formal Languages and Automata.* Prentice-Hall, 1982
  (gut motivierende, ausführliche Einführung)
- A. Salomaa: *Computation and Automata.* Cambridge Univrsity Press, 1985
  (kompakte Darstellung)
- U. Schöning: *Theoretische Informatik – kurzgefasst.* Spektrum Akademischer Verlag, 1997
  (knapp gehaltenes Skript)
- M. Sipser: *Introduction to the Theory of Computation.* PWS Publ. Comp. 1997
  (sehr gut verständlich geschrieben, auch mit Einführungen in viele aktuelle Forschungsgebiete)
- K.W. Wagner: *Theoretische Informatik.* Springer-Verlag, 1994
  (sehr sorgfältige Darstellung der elementaren Berechenbarkeitstheorie)
- D. Wood: *Theory of Computation.* Harper & Row, 1987
  (ein „Course Book“ mit vielen Beispielen)

Weiterführende Bücher

- E. Börger: *Berechenbarkeit, Komplexität, Logik.* Vieweg, 1985
  (ein reichhaltiges Werk mit vielen Literaturhinweisen)
- M.R. Garey, D.S. Johnson: *Computers and Intractability.* Freeman, 1979
  (die „Bibel der NP-Vollständigkeit“)
- P. Odifreddi: *Classical Recursion Theory.* North-Holland, 1989
  (übersichtliches, umfassendes Werk mit vielen Literaturhinweisen)
- C. Papadimitriou: *Computational Complexity.* Addison-Wesley, 1994
  (das derzeitige Standardwerk zur Komplexitätstheorie)