Übung zur Vorlesung

Berechenbarkeit und Komplexität

Antwortenkatalog zum Thema Berechenbarkeit WS 07/08

von Marco Vreydal und Marius Kischel.

Keine Gewährleistung für Korrektheit der Antworten.

1.
$\Sigma^k =$ Alle Wörter der Länge k
$\Sigma^0 =$ Alle Wörter der Länge 0, also ϵ
$\Sigma^* =$ Der Kleenesche Abschluss, also alle möglichen Kombinationen aus dem Eingabealphabet

Eine Sprache ist eine Menge von Wörtern aus dem Alphabet Σ

2.
Eine TM ist ein 7-Tupel ($Q, \Sigma, \Gamma, B, q_{0,} q_f, \delta$), wobei Σ das Eingabealphabet ist, Γ das Bandalphabet, B das Blank, also das Zeichen, das anzeigt, dass an der entsprechenden Stelle auf dem Band nichts steht, Q die Zustandsmenge, q_0 der Startzustand, q_f der Endzustand, und δ die Übergangsfunktion ist.

3. Die Konfiguration ist eine Art „Snapshot" einer bestimmten TM zu einem bestimmten Moment. Sie ist eine Zeichenkette $\alpha q \beta$, das bedeutet auf dem Band steht $\alpha \beta$ eingerahmt von Blanks, und der aktulle Zustand ist q. Die TM steht auf dem ersten Zeichen von β. Eine Nachfolgekonfiguration ist eine Kombination $\alpha' q' \beta'$, die in einem Berechnungsschritt aus $\alpha q \beta$ folgt.

4.

$(Q, \Sigma, \Gamma, \mathrm{B}, q_{0,} q_f, \delta)$
mit
$Q = (q_{0,} q_{1,} q_f)$
$\Sigma = \{0,1\}$
$\Gamma = \Sigma \cup \{\mathrm{B}\}$
δ

	0	1	B
q_0	$(q_{0,} 0, R)$	$(q_{1,} 1, R)$	accept
q_1	reject	$(q_{1,} 1, R)$	accept

5. Eine Mehrband-TM hat mehrere Leseköpfe, die unabhängig voneinander benutzt werden können. Eine Mehrspur-TM hat nur einen Lesekopf, jedoch hält jede Speicherstelle bei einer k-Spur TM einen k-Vektor.

6. Eine k-Band TM kann durch eine 2k-Spur TM simuliert werden. Jede zweite Spur dient

dazu, die aktuelle Kopfposition des jeweiligen Bandes mithilfe einer # zwischenzuspeichern. Auf der ersten und dann jeweils jeder zweiten Spur wird der Inhalt der einzelnen Bänder kopiert.
Die TM läuft zum am weitestens links stehenden Kreuz und beginnt dann, solange nach rechts zu gehen, bis sie k Kreuze gefunden hat. Dies kann sie im Zustandsraum speichern, da die Anzahl an Spuren endlich ist. Die Zeichen an den durch Kreuzen markierten Stellen merkt sie sich ebenfalls im Zustandsraum. Sobald sie am letzten Kreuz angekommen ist, wird die Übergangsfunktion der Mehrband-TM mithilfe der Zwischengespeicherten Werte ausgewertet und die Kreuze werden entsprechend verschoben, sowie die Inhalte an den gegeben Stellen falls nötig überschrieben.
Die Laufzeit ist $O(n^2)$

7. Die RAM wird durch eine 2-Band-TM simuliert werden. Dabei hält ein Band den Speicher der RAM, das andere Band ist ein Arbeitsband für die TM in den Unterprogrammen. Die TM beinhaltet Unterprogramme, und zwar für jede Zeile des RAM-Programms (p-Stück) jeweils eines. Weiterhin gibt es ein Unterprogramm M_0 zur Initialisierung der TM und M_{p+1} zur Ergebnisaufbereitung. Der Programmzähler der RAM kann im Zustandsraum gespeichert werden, da die Zeilenanzahl konstant ist. Mit Speicher, Unterprogrammen und dem Programmcounter ist die Konfiguration der RAM voll erfasst.
   Die TM wird erst initialisiert, liest dann den Programmcounter aus und kopiert die angesprochenen Register auf Band 1. Dann führt sie das zum Programmcounter gehörige Unterprogramm auf und schreibt das Ergebnis von Arbeitsband zurück in das im Speicher angesprochene Register. Zuletzt wird der Programmcounter aktualisiert (b = b+1)
   Die Laufzeit beträgt $O(n+t(n))$

8. Zunächst gehen wir davon aus, dass die TM mit einem einseitig beschränkten Band arbeitet. Das geht, wie wir in der Übung gezeigt haben.
   Register 1 enthält die aktuelle Bandposition, Register 2 enthält den aktuellen Zustand der TM. Die Register 3,4,5,... enthalten den Inhalt der Speicherstellen 0,1,2,3.... .
   Zuerst überprüft die RAM die Übergangsfunktion derart, dass die TM nicht in dem aktuellen Schritt terminiert. Falls doch, so terminiert auch die RAM.
   Ansonsten selektiert die RAM mit $|Q|$ vielen if-Abfragen. Dann wird das aktuelle Zeichen (zu lesendes Register kommt aus Register 1) mit $|(\Gamma)|$ - vielen if-Abfragen selektiert und die entsprechenden Operationen ausgeführt, also das entsprechende Register verändert, die Bandposition in Register 1 umgesetzt (+1 / -1 oder gleichbleibend) und der neue Zustand in Register 2 geschrieben.
   Laufzeit bei uniformen-Kostenmaß: $O(n+t(n))$
   Laufzeit bei logarithmischen-Kostenmaß: $O(n+t(n)*\log(n+t(n)))$

9. Die Gödelnummerierung ist eine injektive Abbildung aus der Menge der Turingmaschinen nach $\{0,1\}^*$. Das bedeutet, jeder möglichen TM wird eine eindeutige Nummer zugewiesen, die auch wieder zur Turingmaschine zurückgeführt werden kann. Prinzipiell gibt es viele verschiedene denkbare Nummerierungen.

10. Die universelle Turingmaschine simuliert eine Eingabe auf einer gegebenen Turingmaschine. Dies geschieht in der Form, dass die universelle Turingmaschine eine Eingabe der Form <M>w erhält, und anschließend M auf w simuliert und das Akzeptanzverhalten von M übernimmt. Falls die Eingabe nicht der Form <M>w entspricht, gibt die universelle Turingmaschine eine Fehlermeldung aus.
    Die universelle Turingmaschine entspricht im Gegensatz zu einer normalen TM, die ein äquivalent zum special purpose Rechner ist, den general purpose Rechner.

11. Akumulator, Programmcounter, unendlich viele Register. Der Akumulator ist implizites Argument der Rechenbefehle. Der Programmcounter steht initial auf 1. Er kann mit Goto x auf x gesetzt werden. END beendet das Programm. Wenn keiner der beiden Befehle

aufgerufen wird, wird der Programmcounter nach jedem Schritt um eins inkrementiert. Eine RAM arbeitet ein endliches Programm mit endlichen vielen Zeilen ab. Der Programmcounter entspricht der jeweils zu bearbeitenden Zeile. Die Eingabe steht zu Anfang in den Registern $x_{1,} x_{2,} \ldots$ Der Speicher ist unbegrenzt.

12. Das uniforme Kostenmaß veranschlagt eine Zeiteinheit pro Operation. Das logarithmische Kostenmaß bedeutet, dass die Laufzeit proportional zur Länge der binären Zahlen in den angesprochenen Registern einer Operation ist.

13. $x_i = x_j + c$, mit $c \in \{-1, 0, 1\}$ – Weist x_i den Wert x_{i+1} zu.
    Wenn P_1 und P_2 WHILE-Programme sind, dann ist auch $P_1 ; P_2$ ein WHILE-Programm
    - führt zuerst P_1 aus und anschließend P_2.
    $WHILE\ x_i \neq 0\ DO\ P_1\ END \mid P_1$ ein gültiges WHILE-Programm
    - Führe so lange P_1 aus, bis x_i Null ist.

    LOOP ist dasselbe, nur die Schleife ist
    LOOP x_i DO P_1 END, wobei in x_i der Wert steht, wie oft die Schleife ausgeführt wird.
    Dabei darf x_i nicht in P_i vorkommen.

    LOOP ist nicht Turing-mächtig, WHILE ist es.

14. $D = \{w \mid w = w_i, M_i \text{ akzeptiert } w_i \text{ nicht}\}$
    $H = \{<M> w \mid M \text{ hält auf } w\}$
    $H_\epsilon = \{<M> \mid M \text{ hält auf } \epsilon \text{ (leere Eingabe)}\}$

15. Eine Sprache L ist rekursiv genau dann wenn es eine TM M gibt, die auf allen Eingaben w stoppt und akzeptiert, falls $w \in L$ und verwirft falls $w \notin L$.

16. Eine Sprache L heißt semi-entscheidbar/aufzählbar, wenn es eine TM M gibt, die L erkennt. (siehe Aufgabe 17). Das bedeutet sie akzeptiert falls $w \in L$, hält jedoch nicht zwingend falls $w \notin L$ (d.h. sie hält nicht oder verwift, darf aber auf keinen Fall akzeptieren).

17. Eine Sprache L wird von einer TM M erkannt, falls M akzeptiert wenn $w \in L$, jedoch nicht akzeptiert, wenn $w \notin L$.
    Eine Sprache L wird von einer TM M entschieden, falls M akzeptiert wenn $w \in L$ und verwirft, falls $w \notin L$.

18. $\textit{Seien } L_1 \textit{ und } L_2 \textit{ Sprachen. Es gilt } L_1 \leq L_2 \Leftrightarrow \exists \textit{ eine berechenbare Funktion } f \textit{ mit}$
    $x \in L_1 \Leftrightarrow f(x) \in L_2$

19. Prinzipiell ist die Reduktion einfach nur eine spezielle Form der Unterprogrammtechnik mit eingeschränkten Möglichkeiten. Um auch Aussagen über rekursiv aufzählbare Probleme zu ermöglichen, ist bei der Reduktion im Gegensatz zur Unterprogrammtechnik untersagt, die Ausgabe nach Aufruf des Unterprogrammes noch zu verändern.

20. siehe Übungen

21. Nicht-triviale Eigenschaften der von einer TM berechneten Funktion sind nicht rekusiv / entscheidbar.

22. Ja, L ist entscheidbar.

    Seien $M \textit{ und } \overline{M}$ TM die jeweils L und $\overline{L}$ erkennen.
    Sei $x \in L \Rightarrow M$ akzeptiert.
    Sei $x \notin L \Leftrightarrow x \in \overline{L} \Rightarrow \overline{M}$ akzeptiert
    $\Rightarrow$ Sowohl L als auch $\overline{L}$ sind rekursiv aufzählbar
    $\Rightarrow L$ ist rekursiv

23. (a) Ja!

Seien $M_1, \ldots, M_n$ *TM die* $L_1, \ldots, L_n$ *erkennen.*
Sei M die TM die $M_1, \ldots, M_n$ *simuliert und genau dann akzeptiert,*
wenn M_1 *und* M_2 *und ... und* M_n *akzeptiert haben.* Dann erkennt M
$M_1 \cap \ldots \cap M_n$

(b) Ja! Man muss die Turingmaschinen „parallel" zueinander simulieren. Dabei wird bei jede TM c Zeitschritte ausgeführt. Würde man zuerst eine TM komplett aufzählen wollen, kämen die anderen ggf. nicht zum Zuge.

(c) Nein!
Angenommen Aussage (c) gelte. Dann gilt L rek. aufzählbar $\Rightarrow \overline{L}$ rek. aufzählbar. $\Rightarrow L$ ist rekursiv. $\Rightarrow$ Alle aufzählbaren Sprachen sind rekursiv: Widerspruch!

24. (a) Halteproblem H .
(b) Komplement vom Halteproblem $\overline{H}$
(c) H_{all} Hält auf allen Eingaben / Hält auf allen Eingaben nicht

25. Das zehnte Hilbertsche Problem bezeichnet das Finden der ganzzahligen Nullstellen eines Polynoms im $\mathbb{R}^n$ mit ganzzahligen Koeffizienten.
$N = \{p \mid p \textit{ ist ein Polynom mit einer ganzzahligen Nullstelle}\}$

$K = \{\frac{x_1}{y_1}, \ldots, \frac{x_n}{y_n}\}$ mit Wörtern x_i, y_i *aus* Σ^* so dass gilt
$\exists$ korrespondierende Folge von Indizes so dass $x_{i1} \ldots x_{in} = y_{i1} \ldots y_{in}$