Prof. T. Nipkow
K. Kappelmann, J. Rädle, L. Stevens **ABGABE 22.06.2020, 23:59 (Σ 5 P.)** Technische Universität München
Lehrstuhl für Logik und Verifikation

Einführung in die Theoretische Informatik

Sommersemester 2020 – Übungsblatt Lösungsskizze 8

AUFGABE 8.1.

1,5 Punkte

In dieser Aufgabe betrachten wir unerreichbare Zustände in Turingmaschinen. Ein Zustand q ist genau dann unerreichbar wenn es keine Eingabe w gibt, so dass die TM im Laufe der Auswertung mit Eingabe w eine Konfiguration (α, q, β) für beliebige α und β erreicht.
Wir kodieren hierzu Zustände in Analogie zu Turingmaschinen und definieren:

$$q_{w_1,w_2} := \begin{cases} q \text{ falls } w_2 \text{ Kodierung von } q \text{ in } M_{w_1} \text{ ist} \\ \hat{q} \text{ sonst} \end{cases}$$

wobei $\hat{q}$ ein beliebiger aber fester Zustand in M_{w_1} ist.
Zeigen Sie, dass die Menge

$$\mathsf{A}_{\mathrm{UR}} := \{w_1 \# w_2 \mid w_1, w_2 \in \{0,1\}^* \text{ und } q_{w_1,w_2} \text{ ist in } M_{w_1} \text{ unerreichbar}\}$$

unentscheidbar ist, indem Sie eine Reduktion durchführen.

Lösungsskizze

Wir reduzieren das Komplement des Halteproblems auf leerem Band $\overline{\mathsf{H}_0}$ auf A_{UR}.

Reduktion von $\overline{\mathsf{H}_0}$: Sei $w \in \{0,1\}^*$ beliebig. Wir berechnen zunächst die Kodierung w' einer Turingmaschine mit genau einem Endzustand q_{accept}, dessen Kodierung w_{accept} ist. $M_{w'}$ löscht bei jeder Eingabe das Band, führt dann $M_w[\varepsilon]$ aus und geht anschließend in den Zustand q_{accept} über. Dann wird $w' \# w_{\text{accept}}$ zurückgegeben.

Die Reduktion ist total: Für jede Eingabe w wird die Ausgabe $w' \# w_{\text{accept}}$ erzeugt.

Die Reduktion ist berechenbar: Die En- und Dekodierungsfunktionen für Turingmaschinen und Zustände sind berechenbar. Außerdem kann eine TM alle Zeichen auf dem Band, die nicht dem Leerzeichen entsprechen, durch geeignete Übergänge anfangs überschreiben.

Die Reduktion ist korrekt: Intuitiv ist der Zustand q_{accept} genau dann erreichbar wenn M_w auf leerer Eingabe hält.

$$\begin{aligned}
w \in \overline{\mathsf{H}_0} &\iff M_w[\varepsilon]\uparrow && \text{(Def. } \mathsf{H}_0\text{)} \\
&\iff \forall x \in \Sigma^*.\ M_{w'}[x]\uparrow && (M_{w'} \text{ führt stets } M_w \text{ auf leerem Band aus}) \\
&\iff \forall x \in \Sigma^*.\big(q_{\text{accept}} \text{ wird in } M_{w'} \text{ nicht erreicht}\big) && \text{(Def. } M_{w'}\text{)} \\
&\iff w' \# w_{\text{accept}} \in \mathsf{A}_{\mathrm{UR}} && \text{(Def. } \mathsf{A}_{\mathrm{UR}}\text{)}
\end{aligned}$$

□

AUFGABE 8.2. (*Wer wird Millionär?*)

0,5 + 0,5 + 0,5 + 0,5 P

Zeigen oder widerlegen Sie die folgenden Aussagen mit einer informellen Begründung.

(a) Wenn A und B unentscheidbar sind, dann ist auch $A \cup B$ unentscheidbar.

(b) Es ist entscheidbar, ob eine beliebige DTM M bei leerer Eingabe jemals ein anderes Symbol als das Leerzeichen schreibt.

Nun betrachten wir WHILE- und GOTO-Programme, wie sie in der Vorlesung auf Folie 247 bzw. 252 definiert sind. Sie dürfen annehmen, dass das Halteproblem für WHILE- und GOTO-Programme auf Eingabe 0 unentscheidbar ist. Wir nummerieren die Anweisungen eines Programms fortlaufend, wobei wir die Bedingung eines IFs bzw. WHILEs als eigene Anweisung betrachten.

(c) Ist es entscheidbar, ob bei der Ausführung eines WHILE-Programms P auf Eingabe 0 die zweite Anweisung mindestens einmal ausgeführt wird?

(d) Ist es entscheidbar, ob bei der Ausführung eines GOTO-Programms P auf Eingabe 0 die zweite Anweisung mindestens einmal ausgeführt wird?

Lösungsskizze

(a) Falsch. Sei $A = H_0$ und $B = \overline{H_0}$. Dann sind A und B unentscheidbar, aber

$$A \cup B = H_0 \cup \overline{H_0} = \{w \in \{0,1\}^* \mid M_w[\varepsilon]\downarrow\} \cup \{w \in \{0,1\}^* \mid \neg M_w[\varepsilon]\downarrow\} = \{0,1\}^*$$

ist entscheidbar.

(b) Korrekt. Wir führen M auf leere Eingabe so lange aus, bis M hält, das Band verändert oder wir stellen fest, dass wir einen bereits besuchten Zustand q erneut besuchen. Wir haben nur endlich viele Zustände, wodurch einer der beiden Fälle nach endlich vielen Schritten auftritt. Da die Konfiguration von M im letzteren Fall genau (q, ε) sein muss, haben wir eine Folge von Konfigurationen $(q, \varepsilon) \to^* (q, \varepsilon)$ und M terminiert nicht.

(c) Diese Eigenschaft ist entscheidbar. Wir machen eine Fallunterscheidung auf der ersten Anweisung des Programms. Wenn die erste Anweisung eine Zuweisung ist, dann wird die zweite Anweisung immer erreicht. Bei einem WHILE wird der Körper nie ausgeführt, da wir 0 als Eingabe annehmen. Die zweite Anweisung des Programms, welche sich im Körper des WHILEs befinden muss, wird dann nie ausgeführt. Bei einem IF ist die Bedingung immer war, und die erste Anweisung im THEN-Arm wird ausgeführt, welche die zweite Anweisung des gesamten Programms ist.

(d) Wir zeigen, dass diese Eigenschaft unentscheidbar ist. Für ein gegebenes Programm P konstruieren wir das Programm `GOTO 3; 2 : x := x + 1; 3: x := x; P; GOTO 2` wobei wir x_0 mit x abkürzen. Die zweite Anweisung `x := x + 1` wird genau dann mindestens einmal ausgeführt, wenn P mit Eingabe 0 terminiert. Wenn wir also die gegebene Eigenschaft entscheiden könnten, dann könnten wir auch das Halteproblem für Eingabe 0 für GOTO-Programme entscheiden, welches jedoch unentscheidbar ist.

AUFGABE 8.3. (*YOWO – You Only Write Once*) 1,5 Punkte

Eine You-Only-Write-Once (YOWO) Maschine ist eine Turingmaschine, die jede Zelle ihres Bandes höchstens einmal ändern darf. Dabei gelten Übergänge $\delta(q, x) = (p, y, X)$ mit $x \neq y$ als Änderung einer Zelle. Insbesondere darf eine YOWO-Maschine ihre Eingabe höchstens einmal überschreiben.
Zeigen Sie: YOWO-Maschinen sind Turing-vollständig[1]. Es reichte eine informelle aber umfassende Beschreibung ohne formalen Beweis.
Tipp: Überlegen Sie zunächst, wie Sie die Turing-vollständigkeit von you-only-write-twice (YOWT) Maschinen zeigen können.

Lösungsskizze
Idee: Zu jedem Zeitpunkt steht nur endlich viel Information auf dem Band einer Turingmaschine. Die Idee ist es nun, für jeden Schritt der originalen Turingmaschine den aktuellen Bandinhalt auf einen noch freien Platz (z.B. rechts) zu kopieren und dabei einen Schritt der TM auszuführen. Dabei wird außerdem die Kopfposition der originalen TM auf dem Band mit einem neuen Markierungszeichen vermerkt. Die Sequenz an Konfigurationen kann dabei durch Trennzeichen abgegrenzt werden.
Zunächst für YOWT-Maschinen: Initial fügt die Maschine das Kopfmarkierungszeichen ein. Die Maschine kopiert dann stets die aktuelle Konfiguration Zeichen für Zeichen nach rechts. Sobald ein Zeichen kopiert werden soll, wird es mit einer Markierung versehen. Falls das Zeichen nicht an der Kopfmarkierung anliegt, wird es einfach kopiert. Anderenfalls wird es entsprechend des Überganges der originalen Turingmaschine angepasst. Die Prozedur modifiziert jede Zelle somit höchstens zweimal: einmal, um ein Zeichen zum erstem mal hinzuschreiben und ein zweites mal, um das Zeichen im nächsten Schritt als kopiert zu markieren.
Nun für YOWO-Maschinen: Der Prozess ist analog zu dem vorherigen. Allerdings kann keine Markierung der Zeichen beim Kopieren in derselben Zelle passieren. Trick: Jede ursprüngliche Zelle wird durch zwei Zellen repräsentiert. Die erste Zelle enthält das ursprüngliche Zeichen und die zweite dient als Markierstelle während der Kopierprozedur. Der Input liegt nicht in diesem Format vor, allerdings darf dieser einmal überschrieben werden. Deshalb wird beim ersten Kopiervorgang einfach die Markierung, wie schon vorher, über das Eingabezeichen gelegt (oder alternativ erstmal in das neue Format kopiert).

> Can you "think of" an undefinable number?
>
> — Robert Israel

[1] D.h. jede Turingmaschine kann durch eine YOWO-Maschine simuliert werden