Einführung in die Theoretische Informatik II

Abschnitt über die

Unentscheidbarkeit des Halteproblems

Rolf Wanka
Universität Erlangen-Nürnberg

rwanka@cs.fau.de

6. Mai 2005

1.4 Unentscheidbare Probleme

Gibt es Grenzen für das, was Turingmaschinen berechnen können? Wir werden sehen, daß es solche Grenzen gibt, die sehr wichtige Probleme betreffen. Genauer gesagt: Wir werden zeigen, daß Probleme, von denen wir uns wünschten, daß sie lösbar wären, algorithmisch nicht lösbar sind!

Die Gödelnummer einer Turingmaschine M, die wir in Definition 1.12 einführten, besteht nur aus Buchstaben aus $\{0,1\}$. Unter anderem ist in der Gödelisierung jedes Zeichen des endlichen Alphabets Σ von M durch eine feste 0-1-Folge kodiert. Deswegen gehen wir im folgenden davon aus, daß für alle Turingmaschinen, die uns begegnen, $\Sigma = \{0,1\}$ ist.

1.15 Definition:
Die Sprache

$$H := \{\langle M\rangle w \mid M \text{ ist eine 1-Band-Turingmaschine, die, gestartet mit } w, \text{ hält}\}$$

ist das (allgemeine) *Halteproblem*.

> Beachten Sie, daß wir hier eine Menge von Zeichenfolgen (ein anderes Wort für „Menge von Zeichenfolgen“ ist ja *Sprache*) als *Problem* bezeichnen. Das ist eine Konvention, hinter der steckt, daß wir letztlich das zugehörige *Wortproblem* oder auch, in äquivalenter Bedeutung, *Entscheidungsproblem* meinen. Wir haben also die Menge H definiert und meinen im Hinterkopf für alle $w \in \{0,1\}^*$ die (maschinelle) Beantwortung der Frage „$w \in H$?“. Trotzdem: Nur die Menge H ist das Halteproblem. Im übrigen gilt auch für H: $H \subseteq \{0,1\}^*$.

1.16 Satz: (Turing, 1936)
H ist unentscheidbar (eine andere, aber äquivalente Formulierung lautet: *H ist nicht rekursiv*).

Beweis:
Wir nehmen an, daß es eine Turingmaschine M_H gibt, die H entscheidet. D. h. M_H hält auf *jeder* Eingabe x, und man kann am erreichten Zustand erkennen, ob $x \in H$ ist oder nicht!

Diese Annahme werden wir nun auf einen Widerspruch führen.

Wenn es M_H gibt, dann gibt es auch die folgende Turingmaschine M_{schlau}:

Turingmaschine M_{schlau}

(1) die Eingabe sei y
(2) falls $y = \langle M\rangle$, dann
(3) entscheide mittels M_H, ob $\langle M\rangle\langle M\rangle \in H$
(4) falls ja: schreibe nach rechts endlos viele 1en auf das Band
(5) falls nein: bleibe stehen

Die Turingmaschine M_{schlau} können wir explizit hinschreiben („programmieren“), falls uns jemand die δ-Tabelle von M_H gibt (die als „Unterprogramm“ aufgerufen würde).

Was passiert, wenn wir M_{schlau} mit der Eingabe $y = \langle M_{\text{schlau}} \rangle$ starten? Nun, es gibt nur zwei Möglichkeiten, nämlich die, daß die Maschine hält („Fall (a)“), und die, daß sie nicht hält („Fall (b)“).

(a) Wenn M_{schlau}, gestartet mit $\langle M_{\text{schlau}} \rangle$, hält, heißt das, daß die Abfrage in (3) die Antwort „nein“ ergeben hatte, also $\langle M_{\text{schlau}} \rangle \langle M_{\text{schlau}} \rangle \notin H$. D. h. wiederum, daß M_{schlau}, gestartet mit $\langle M_{\text{schlau}} \rangle$, nicht hält, im Widerspruch zum Anfang der Argumentation!

Also kann Fall (a) nicht gelten. Nun betrachten wir Fall (b).

(b) Wenn M_{schlau}, gestartet mit $\langle M_{\text{schlau}} \rangle$, endlos läuft, muß sich die Maschine in Zeile (4) befinden. Dorthin kommt sie nur, wenn die Abfrage in (3) die Antwort „ja“ ergeben hatte, also $\langle M_{\text{schlau}} \rangle \langle M_{\text{schlau}} \rangle \in H$. D. h. wiederum, daß M_{schlau} gestartet mit $\langle M_{\text{schlau}} \rangle$ hält, im Widerspruch zum Anfang der Argumentation!

Also kann keiner der beiden Fälle gelten, wir müssen somit irgendwo von etwas ausgegangen sein, das falsch ist. Unsere gesamte Argumentation hat aber nur eine Schwachstelle, nämlich die Annahme, daß es M_H gibt. M_H gibt es also gar nicht, und als Konsequenz ist H nicht entscheidbar! □

1.17 Satz:

(a) H ist rekursiv aufzählbar.

(b) Das Komplement von H, d. h. $\overline{H} = \{0,1\}^* \setminus H$, ist nicht rekursiv aufzählbar.

Beweis:

(a) Die universelle Turingmaschine M_0 aus Satz 1.14 ist der rekursive Aufzähler von H, d. h.

$$\langle M \rangle w \in H \iff M_0, \text{ gestartet mit } \langle M \rangle w, \text{ hält}$$

(b) Wäre $\overline{H}$ rekursiv aufzählbar (mittels der Turingmaschine $\tilde{M}$), könnte man folgende Turingmaschine programmieren:

Entscheider für H

die Eingabe sei $\langle M \rangle w$
führe *gleichzeitig* aus und stoppe, wenn eine stoppt:
- starte mit $\langle M \rangle w$ auf Band 1 die Turingmaschine M_0 , die die Worte aus H akzeptiert (aus Teil (a))
- starte mit $\langle M \rangle w$ auf Band 2 die Turingmaschine $\tilde{M}$, die die Worte aus $\overline{H}$ akzeptiert

hält M_0, halte akzeptierend, hält $\tilde{M}$, halte verwerfend.

Diese Turingmaschine hält für jede Eingabe, denn eine der beiden Untermaschinen, M_0 oder $\tilde{M}$, muß ja halten! Also entscheidet diese Turingmaschine das Halteproblem. Folglich (wegen Satz 1.16) gibt es $\tilde{M}$ nicht. □

Mit der schlechten Nachricht der Unentscheidbarkeit des Halteproblems beginnt nun eine ganze Folge von derartigen schlechten Nachrichten.

1.18 Definition:
Die Sprache

$$H_\varepsilon := \{\langle M\rangle \mid M, \text{ gestartet mit leerem Band, hält}\}$$

heißt *spezielles Halteproblem*.

1.19 Satz:
H_ε ist nicht entscheidbar.

Beweis:
Wir zeigen, daß man einen Entscheidungsalgorithmus für H programmieren könnte, wenn H_ε entscheidbar wäre. Dazu programmieren wir um die Eingabe zum Halteproblem H so herum, daß wir eine Eingabe des speziellen Halteproblem H_ε bekommen.

Gegeben sei also die Eingabe $\langle M\rangle w$ und damit die Frage „$\langle M\rangle w \in H$?“. Wie kann man diese Frage beantworten, wenn man die Frage „$\langle M'\rangle \in H_\varepsilon$?“ für alle $\langle M'\rangle$ beantworten könnte?

Betrachten Sie zu $\langle M\rangle w$ folgende Turingmaschine:

feste_Maschine$_{\langle M\rangle w}$
(1) die Eingabe sei x
(2) starte M mit w $\quad(*$ das ist kein Tippfehler $*)$
(3) falls $x = \varepsilon$, dann halte.

Nun nehmen wir an, daß wir H_ε mittels der Turingmaschine $\tilde{M}$ entscheiden können. Dann haben wir:

(a) Wenn $\tilde{M}$ bei Eingabe $\langle$feste_Maschine$_{\langle M\rangle w}\rangle$ akzeptierend hält, muß feste_Maschine$_{\langle M\rangle w}$ bis Zeile (3) kommen, was nur möglich ist, wenn M, gestartet mit w, hält. Also ist $\langle M\rangle w \in H$.

(b) Wenn $\tilde{M}$ bei Eingabe $\langle$feste_Maschine$_{\langle M\rangle w}\rangle$ verwerfend hält, kann feste_Maschine$_{\langle M\rangle w}$ nicht bis Zeile (3) kommen, was nur möglich ist, wenn M, gestartet mit w, nicht hält. Also ist $\langle M\rangle w \notin H$.

D. h.: Die Antwort von $\tilde{M}$ auf die Eingabe $\langle$feste_Maschine$_{\langle M\rangle w}\rangle$ ist die Antwort auf die Frage „$\langle M\rangle w \in H$?“!
Und damit kann es $\tilde{M}$ nicht geben, mithin ist H_ε nicht entscheidbar. □

Den Trick, das Halteproblem (oder dann weiter andere unentscheidbare Probleme) durch Drumherumprogrammieren zu maskieren, kann man immer wieder anwenden. Er heißt *Reduktion* und wird im weiteren ausführlich behandelt.

> Eine Quelle permanenten Mißverständnisses für Newcomer auf diesem Gebiet ist, wer auf wen reduziert wird. Hier im Beweis wurde das Halteproblem H auf das spezielle Halteproblem H_ε reduziert. Im Fall der Unentscheidbarkeit wird also das Problem, von dem man bereits die Unentscheibarkeit weiß, auf das Problem, von dem man es noch nicht weiß, reduziert. Machen Sie sich bitte unbedingt mit dieser (in der Tat verstehbaren) Sprechweise vertraut. Und: Üben Sie Reduktionen!