Unentscheidbarkeit

Steffen Puhlmann

25.01.2011

Der Entscheidbarkeitsbegriff

Eine Sprache L heißt entscheidbar, falls es eine Turingmaschine M gibt, die auf eine Eingabe ω akzeptierend hält, wenn ω in L liegt und für ω ablehnend hält, wenn ω nicht in L liegt.
Der Entscheidbarkeitsbegriff lässt sich auf den Berechenbarkeitsbegriff zurückführen. Wir betrachten dafür folgende Funktion:

$$f_L(\omega) = \begin{cases} 1 & \omega \in L \\ 0 & sonst \end{cases}$$

f ist berechenbar, falls ein Algorithmus existiert (hier: ein Turingprogram), der diese Funktion berechnet.

Semi-Entscheidbarkeit

Eine Sprache L heißt semi-entscheidbar, falls es eine Turingmaschine M gibt, die auf eine Eingabe ω akzeptierend hält, wenn ω in L enthalten ist. Sollte ω nicht in L enthalten sein, so muss M nicht unbedingt halten. Ist eine Sprache L semi-entscheidbar, so gilt sie bereits als unentscheidbar.

Beispiele für entscheidbare Sprachen

Die Sprache $L_{DEA} = \{< B, \omega > | \omega \in L(B)\}$ ist entscheidbar.

Beweis-Skizze: Wir simulieren B mit Hilfe einer Turingmaschine M folgendermaßen:

1. M überprüft, ob $< B, \omega >$ eine korrekte Eingabe ist. B muss aus folgenden Bestandteilen bestehen: $B = (Q, \Sigma, \delta, q_0, F)$, $\omega \in \Sigma^*$
2. M simuliert B direkt: Die Position und die Zustände von B werden auf dem Band von M gespeichert. Zu Beginn ist B im Zustand q_0 und liest das erste Zeichen von ω.
3. Die Zustände und Positionen werden entsprechend den Regeln von δ aktualisiert.
4. Wenn M das letzte Symbol von w gelesen hat, so hält M akzeptierend, wenn B in einem akzeptierenden Zustand ist. Ansonsten hält M ablehnend.

Die Sprache $L_{KfG} = \{< G, \omega > | \omega \in L(G)\}$ ist entscheidbar.

Beweis-Skizze: Wir können nicht alle möglichen Ableitungen der Grammatik mit einer TM M durchgehen und überprüfen, ob eine davon das Wort ω erzeugt, da es unendliche viele Ableitungen geben kann und M demnach nie halten würde. Ebensowenig können wir den zu L_{KfG} äquivalenten Kellerautomaten P durch M simulieren, da P in eine Endlosschleife geraten kann, in welcher unendliche lange auf dem Stack gelesen und geschrieben wird, während P mittels ϵ-Übergängen die Zustände wechselt. Daher überführen wir G in die Chomsky-Normalform.
Satz: Ein Wort ω, mit $|\omega| = n > 0$, welches sich durch eine kontextfreie Grammatik in CNF erzeugen lässt, braucht für die Ableitung genau $2n - 1$ Produktionsschritte.

Beweis: Vollständige Induktion über n
Daher gehen wir mit M nur alle Ableitungen durch, die maximal $2n-1$ Produktionsschritte benötigen. Davon gibt es endlich viele. Findet M eine Ableitung, die das Wort ω erzeugt, so hält sie akzeptierend. Erzeugt keine dieser Ableitungen das Wort ω, so hält M ablehnend.

Unentscheidbarkeit

Das Prinzip der Reduktion

Eine Reduktion bedeutet, ein Problem A auf ein zweites Problem B zu übertragen, indem die Lösung von Problem B auch das Problem A löst. Die Reduzierbarkeit sagt nichts über die Lösung von A oder B aus, lediglich über die Schwierigkeit der Lösung von Problem A in Bezug auf die Lösung von Problem B. Ist Problem A auf Problem B reduzierbar, so kann A nicht schwerer zu lösen sein als B. Daraus folgt: Ist A reduzierbar auf B und B ist lösbar, so ist auch A lösbar. Äquivalent gilt: Ist A nicht lösbar und reduzierbar auf B, so ist auch B nicht lösbar. Reduktion ist ein häufig gebrauchtes Beweisverfahren um die Nichtlösbarkeit von Problemen zu zeigen, indem Probleme, von denen bereits bekannt ist, dass sie nicht lösbar sind, auf diese reduziert werden.

Die Diagonalsprache

Definition: $D = \{\omega \,|\, \omega \notin L(M_\omega)\}$ ist die Menge aller Codierungen von Turingmaschinen, die für ihre eigene Codierung als Eingabe nicht akzeptierend halten.
Soll M_ω die Eingabe ω entscheiden, so gibt es folgende Möglichkeiten:

- Angenommen $\omega \in D$:
  - M_ω müsste akzeptierend halten
  - laut Definition von D gilt aber: $\omega \notin D$
  - Widerspruch!
- Angenommen $\omega \notin D$:
  - M_ω dürfte ω nicht akzeptieren
  - laut Definition von D gilt aber: $\omega \in D$
  - Widerspruch!

Demnach ist die Sprache D nicht semi-entscheidbar.

Das spezielle Halteproblem: $H_{spez} = \{\omega \,|\, M_\omega \, hält \, auf \, \omega\}$

- Das spezielle Halteproblem ist das Komplement zur Diagonalsprache.
- H_{spez} ist unentscheidbar. Wäre H_{spez} entscheidbar, so wäre auch die komplementäre Sprache entscheidbar, weil Enscheidbarkeit unter Komplementbildung abgeschlossen ist.
- H_{spez} ist semientscheidbar.

Korollar: Semi-Entscheidbarkeit und Nicht-Semi-Entscheidbarkeit sind unter Komplementbildung *nicht* abgeschlossen.

Das allgemeine Halteproblem: $H = \{< M, \omega > \,|\, M \, hält \, auf \, \omega\}$

Wir reduzieren das spezielle Halteproblem auf das allgemeine Halteproblem: Sei ω die Eingabe für das spezielle Halteproblem und M_ω die zu ω gehörige Turingmaschine. Dann wird die Eingabe ω in die Eingabe $< M_\omega, \omega >$ für das allgemeine Halteproblem überführt. Das Ergebnis für das allgemeine Halteproblem ist das gleiche wie für das spezielle. Also gilt: Da das spezielle Halteproblem unentscheidbar ist, ist auch das allgemeine Halteproblem unentscheidbar.

Die universelle Sprache: $L_u = \{< M, \omega > \mid \omega \in L(M)\}$

Sei U die universelle Turingmaschine, die L_u entscheidet. Dann operiert U auf die Eingabe $< M, \omega >$ folgendermaßen:

- U simuliert M auf die Eingabe ω
- hält M akzeptierend, so hält auch U akzeptierend
- hält M ablehnend, so hält auch U ablehnend
- gerät aber M in eine Endlosschleife, so wird auch U niemals halten.

Die universelle Sprache ist also semi-entscheidbar.

Beispiel für Unentscheidbarkeit $L_{reg} = \{M \mid L(M)\, ist\, regulär\}$

Wir wollen die universelle Sprache L_u auf die Sprache L_{reg} reduzieren. Dazu nehmen wir an, dass die Turingmaschine R die Sprache L_{reg} entscheidet. Mit Hilfe von R wollen wir eine Turingmaschine S konstruieren, die L_u entscheidet. S operiert dabei auf die Eingabe $< M, \omega >$ folgendermaßen:

- S modifiziert die TM M zur TM M_2
- akzeptiert M die Eingabe ω, so akzeptiert M_2 die Sprache $L_r = \Sigma^*$
- wenn M die Eingabe ω nicht akzeptiert, so akzeptiert M_2 die Sprache $L_{kfr} = \{a^n b^n \mid n \geq 0\}$
- wir lassen R auf M_2 als Eingabe laufen
  - akzeptiert R die Eingabe M_2, so akzeptiert S die Eingabe $< M, \omega >$
  - wenn nicht, so lehnt S die Eingabe $< M, \omega >$ ab

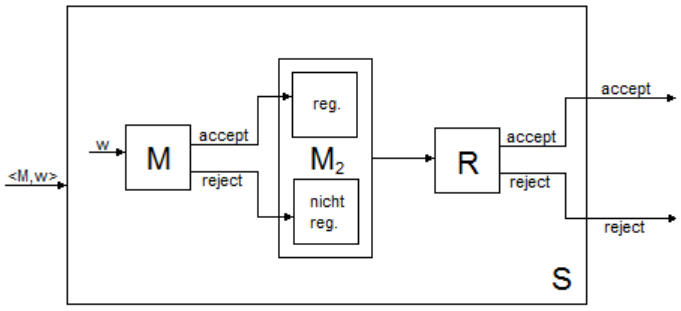

Figure 1: Darstellung von S

Wenn R die Eingabe M_2 entscheidet, so entscheidet auch S die Eingabe $< M, \omega >$. Weil die Sprache L_u unentscheidbar ist, ist auch die Sprache L_{reg} unentscheidbar.

Der Satz von Rice

Alle nichttrivialen Eigenschaften von rekursiv aufzählbaren Sprachen sind unentscheidbar.

Eigenschaft

Eine Eigenschaft P lässt sich als Menge von Sprachen definieren. Eine Sprache besitzt diese Eigenschaft, wenn sie Element dieser Menge ist. Beispielsweise ist die Eigenschaft *kontextfrei* durch die Menge $P_{kfr} = \{L \mid L\, ist\, kontextfrei\}$ definiert. Eine eigenschaft P heißt *trivial*, wenn gilt:

- $P = \emptyset$, keine rekursiv aufzählbare Sprache hat diese Eigenschaft
- $P = \{L \mid L\, ist\, r.a.\}$, alle r.a. Sprachen haben diese Eigenschaft

Wir betrachten nun die Sprache $L_P = \{M \mid L(M)\, hat\, die\, Eigenschaft\, P\}$
Das Problem, ob die durch M erzeugte Sprache die Eigenschaft P besitzt ist das gleiche Problem, wie jenes, ob M in L_P enthalten ist.

Beweis

Sei P eine nichttriviale Eigenschaft, mit $\emptyset \notin P$. Da P nichttrivial ist, gibt es eine Sprache L, welche diese Eigenschaft hat. Sei M_L eine Turingmaschine, mit $L(M_L)$= L. Also gilt: $M_L \in L_P$. Wir reduzieren nun die universelle Sprache L_u auf die Sprache L_P um zu zeigen, dass L_P unentscheidbar ist. Der Algorithmus dafür bekommt als Eingabe das Paar $< M, \omega >$ und erzeugt eine neue Turingmaschine M', für die folgendes gilt:

$$L(M') = \begin{cases} \emptyset & \omega \notin L(M) \\ L & \omega \in L(M) \end{cases}$$

Beschreibung von M'

M' ist eine 2-Band-Turingmaschine. Auf dem ersten Band wird die Eingabe $< M, \omega >$ codiert gespeichert. Auf dem zweiten Band wird die Turingmaschine M_L und die Eingabe x gespeichert. Hierfür ist M_L dem Algorithmus bekannt und x dient als Eingabe für M'. Die Turingmaschine M' operiert folgendermaßen:

1. Auf Band 1 wird M auf ω simuliert.
2. Wird ω von M nicht akzeptiert, so macht auch M' nichts weiter und wird niemals die eigene Eingabe x akzeptieren. Daraus folgt: $L(M) = \emptyset$
3. Wird ω von M akzeptiert, so beginnt M' mit der Simulation von M_L auf x. M_L akzeptiert nur Wörter die in L liegen. Daher akzeptiert auch M' nur Wörter aus L. Weil die Sprache L die Eigenschaft P hat, hat auch $L(M')$ die Eigenschaft P. Daher ist M' Element von L_P.

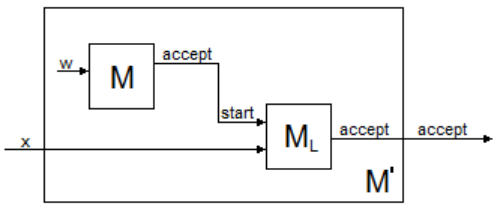

Figure 2: Darstellung von M'

Der Algorithmus wandelt die Eingabe $< M, \omega >$ in eine Turingmaschine M' um, die in L_P liegt, wenn $< M, \omega >$ auch in L_u enthalten ist. Demnach ist der Algorithmus eine Reduktion von L_u auf L_P und beweist, dass die Eigenschaft P unentscheidbar ist.
Wir betrachten nun den Fall, dass $\emptyset$ in P enthalten ist. Sei $\overline{P}$ die komplementäre Eigenschaft zu P. Dann sei $\overline{L_P}$ die Menge aller Turingmaschinen, die jene Sprachen erkennen, die nicht in P liegen. Damit ist $\overline{L_P}$ äquivalent zu der Sprache $L_{\overline{P}}$. Angenommen L_P ist entscheidbar, dann wäre auch auch $L_{\overline{P}}$ entscheidbar, da Entscheidbarkeit gegenüber Komplementbildung abgeschlossen ist.