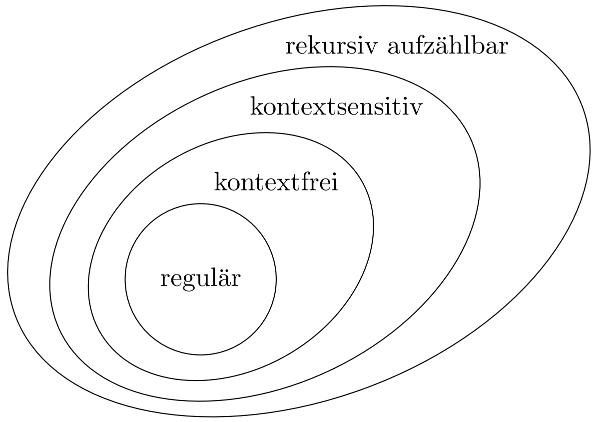
Unentscheidbare Probleme bei formalen Sprachen

Olaf Parczyk 17.01.2011

1 Einleitung

Was wissen wir?

Aus der Vorlesung „Grundlagen der Theoretischen Informatik“ und dem Vortrag über die „Chomsky Hierarchie“ haben wir ein Grundverständnis über den Zusammenhang von Sprachen und Grammatiken.



Was wollen wir?

Im folgenden wollen wir einige interessante Probleme im Zusammenhang mit den gerade vorgestellten Sprachen klären. Im Vordergrund stehen dabei das Wortproblem, das Leerheitsproblem und das Äquivalenzproblem. Dabei stellt sich uns immer die Frage, ob die Probleme entscheidbar sind, oder nicht. Wir erinnern uns an den Vortrag über „Berechenbarkeit“ und definieren:

Definition 1.1. *Eine Menge A aus dem Universum X heißt* ***entscheidbar****, genau dann, wenn die charakteristische Funktion $\chi_A : X \to \{0,1\}$ definiert durch*

$$\chi_A(x) = \begin{cases} 1, & \textit{falls } x \in A \\ 0, & \textit{sonst} \end{cases} \tag{1}$$

turing-berechenbar ist. Also dann, wenn es eine Turingmaschine S gibt, die für jede Eingabe $x \in X$ hält und deren akzeptierte Sprache $L(S) = A$ ist. Wir sagen dann S entscheidet A.

Des weiteren fassen wir eine Klasse von Sprache als entscheidbar auf, wenn ihr Wortproblem entscheidbar ist. Also wenn eine Turingmaschine entscheiden kann, ob ein bestimmtes Wort von einer bestimmten Grammatik oder einem bestimmten Automaten akzeptiert wird.

2 Entscheidungsprobleme formaler Sprachen

Reguläre Sprachen

Satz 2.1. *$A_{DFA} = \{\langle B, \omega\rangle | B$ ist ein DFA und akzeptiert $\omega\}$ ist entscheidbar.*

Beweisidee. Wir konstruieren eine Turingmaschine S, die die Eingabe liest, anschließend den DFA auf ω simuliert und damit entscheidet, ob $\langle B, \omega\rangle \in A_{\text{DFA}}$ □

Satz 2.2. *$E_{DFA} = \{\langle A\rangle | A$ ist ein DFA und $L(A) = \emptyset\}$ ist entscheidbar.*

Beweisidee. Diesmal entscheidet unsere Turingmaschine S die Sprache E_{DFA} indem sie vom Startzustand aus alle erreichbaren Zustände markiert und am Ende überprüft, ob ein akzeptierender markiert ist. □

Satz 2.3. *$EQ_{DFA} = \{\langle A, B\rangle | A, B$ sind DFA und $L(A) = L(B)\}$ ist entscheidbar.*

Beweisidee. Wir konstruieren den DFA C als symmetrische Differenz von L(A) und L(B), dann gilt

$$L(C) = \Big(L(A) \cap \overline{L(B)}\Big) \cup \Big(\overline{L(A)} \cap L(B)\Big) \tag{2}$$

und wir wenden die Turingmaschine S aus Satz 2.2 um $\langle C\rangle \in E_{\text{DFA}}$ zu entscheiden. □

Für die regulären Sprachen bzw. DFAs sind also alle Probleme entscheidbar.

Kontextfreie Sprachen

Wenden wir uns nun der nächstgrößeren Klasse, den kontextfreien Grammatiken zu.

Satz 2.4. *$A_{CFG} = \{\langle B, \omega\rangle | B$ ist ein CFG und akzeptiert $\omega\}$ ist entscheidbar.*

Beweisidee. Wir wanden die Grammatik in Chomsky-Normalform um und müssen nur noch $2|\omega| - 1$ viele Ableitungsschritte betrachten. □

Satz 2.5. *$E_{CFG} = \{\langle A\rangle | A$ ist ein CFG und $L(A) = \emptyset\}$ ist entscheidbar.*

Beweisidee. Wir gehen wieder mit einem Markierungsalgorithmus vor und beginnen bei den Terminalsymbolen. Wenn am Ende das Startsymbol markiert, dann akzeptieren wir. □

Satz 2.6. *$EQ_{CFG} = \{\langle A, B\rangle | A, B$ sind CFG und $L(A) = L(B)\}$ ist unentscheidbar.*

Hier stoßen wir zum ersten Mal an unsere Grenzen, denn im Gegensatz zu den regulären Sprachen sind die kontextfreien nicht bezüglich Durchschnitt und Komplement abgeschlossen. Um zu zeigen, dass eine Sprache unentscheidbar ist benötigen wir noch weitere Erkenntnisse aus der GTI und über Reduktion. Wir kommen später darauf zurück.

Rekursiv aufzählbare Sprachen

Um zu zeigen, dass ein bestimmtes Problem nicht entscheidbar ist, ist die Reduktion ein hilfreiches Mittel. Wenn wir wissen, dass ein Problem B unentscheidbar ist und zeigen können, dass sich B auf A reduzieren lässt, dann folgt daraus, dass auch A unentscheidbar ist. Um ein erstes unentscheidbares Problem zu haben erinnern wir uns an die Diagonalsprache.

Proposition 2.7. *$D = \{\langle\omega\rangle | M_\omega$ akzeptiert ω nicht$\}$, wobei M_ω die mit ω kodierte Turingmaschine bezeichnet, ist unentscheidbar.*

Beweisidee. Wenn es eine Turingmaschine S gäbe, die D entscheidet und ω ihre Kodierung ist, dann führt

$$\omega \in D \overset{L(S)=D}{\Leftrightarrow} S \text{ akzeptiert } \omega \overset{\text{Def. von } D}{\Leftrightarrow} \omega \notin D. \tag{3}$$

direkt zum Widerspruch. □

Nun haben wir eine Basis und können uns mit den rekursiv aufzählbaren Sprachen beschäftigen.

Satz 2.8. *$A_{TM} = \{\langle B, \omega\rangle | B$ ist eine TM und akzeptiert $\omega\}$ ist unentscheidbar.*

Beweisidee. Angenommen A_{TM} wäre entscheidbar. Dann gibt es eine TM S, die A_{TM} entscheidet. Daraus konstruieren wir eine TM T, die die Diagonalsprache entscheidet, indem wir die Turingmaschine M_ω hinzuziehen. Wegen $\langle M_\omega, \omega\rangle \in A_{\text{TM}} \Leftrightarrow \langle\omega\rangle \notin D$ entscheidet T D, ein Widerspruch. □

Satz 2.9. *$E_{TM} = \{\langle B\rangle | B$ ist eine TM und $L(B) = \emptyset\}$ ist unentscheidbar.*

Beweisidee. Wir nehmen an, dass E_{TM} von der TM S entschieden wird. Eine TM B', die nur auf ω B anwendet hilft uns dann $\langle B, \omega\rangle \in A_{\text{TM}}$ zu entscheiden. □

Satz 2.10. *$EQ_{TM} = \{\langle A, B\rangle | A, B$ sind TM und $L(A) = L(B)\}$ ist unentscheidbar.*

Beweisidee. Mit Hilfe eine zweiten TM, die jede Eingabe ablehnt können wir die Annahme, dass EQ_{TM} entscheidbar ist direkt zum Widerspruch führen, indem wir E_{TM} auf EQ_{TM} reduzieren. □

Kontextsensitive Sprachen

Um uns mit den kontextsensitiven Sprachen zu beschäftigen benötigen wir ein weiteres Hilfsmittel.

Definition 2.11. *Sei M eine TM und ω eine Eingabe. Dann ist die Folge der Konfigurationen $C_1, C_2, \cdots, C_k$ eine **akzeptierende Berechnungsvergangenheit**, wobei C_1 die Startkonfiguration, C_k eine akzeptierende Konfiguration ist und jedes C_i auf C_{i+1} entsprechend der Überführungsfunktion von M folgt. Es handelt sich um eine **ablehnende Berechnungsvergangenheit**, wenn C_k keine akzeptierende Konfiguration ist.*

Außerdem erinnern wir uns an linear beschränkte Turingmaschinen, die nur den Teil des Bandes benutzen darf, auf dem die Eingabe steht.

Satz 2.12. $A_{LBA} = \{\langle B, \omega\rangle | B$ *ist eine LBA und akzeptiert* $\omega\}$ *ist entscheidbar.*

Beweisidee. Wir müssen uns nur darum kümmern, was passiert, wenn B loopt. Dazu genügt es aber einzusehen, dass es beschränkt viele Konfigurationen gibt. □

Satz 2.13. $E_{LBA} = \{\langle B\rangle | B$ *ist eine LBA und* $L(B) = \emptyset\}$ *ist unentscheidbar.*

Beweisidee. Wir nehmen an, dass E_{LBA} entscheidbar wäre und konstruieren einen LBA, der nur eine akzeptierende Berechnungsvergangenheit für ω akzeptiert. So reduzieren wir A_{TM} auf E_{LBA}. □

Satz 2.14. $EQ_{LBA} = \{\langle A, B\rangle | A, B$ *sind LBA und* $L(A) = L(B)\}$ *ist unentscheidbar.*

Beweisidee. Analog zu EQ_{TM} nehmen wir einen zweiten LBA hinzu, der jede Eingabe ablehnt. □

Nun haben wir auch für die kontextsensitiven Sprachen unsere Probleme geklärt, erinnern uns aber noch an eine Sprache, deren Unentscheidbarkeit wir noch nicht beweisen konnten.

Proposition 2.15. $ALL_{CFG} = \{\langle A\rangle |, B$ *ist eine CFG und* $L(A) = \Sigma^*\}$ *ist unentscheidbar.*

Beweisidee. Wir reduzieren wieder A_{TM} auf ALL_{CFG}, welches angenommenerweise von der TM S entschieden wird. Diesmal konstruieren wir einen nichtdeterministischen Kellerautomaten, der alles akzeptiert bis auf eine akzeptierende Berechnungsvergangenheit für ω. Diesen wandeln wir in einen CFG D, es gilt

$$L(D) \subsetneqq \Sigma^* \Leftrightarrow \langle D\rangle \notin ALL_{\mathrm{CFG}} \Leftrightarrow \langle B, \omega\rangle \in A_{\mathrm{TM}} \tag{4}$$

und A_{TM} wäre entscheidbar. Widerspruch zu Satz 2.8. □

Beweisidee von Satz 2.6. Jetzt können wir auch diesen Satz beweisen, indem wir eine CFG B' hinzunehmen, die jedes Wort akzeptiert und aus $\langle B, B'\rangle \in EQ_{\mathrm{CFG}} \Leftrightarrow \langle B\rangle \in ALL_{\mathrm{CFG}}$ den Widerspruch zur Unentscheidbarkeit von ALL_{CFG} herleiten. □

Abschließend führt man sich am besten nochmal in einer Tabelle vor Augen, welche Zusammenhänge wir herausgefunden haben.

Entscheidbar?	DFA	CFG	LBA	TM
Wortproblem A	ja	ja	ja	nein
Leerheitsproblem E	ja	ja	nein	nein
Äquivalenzproblem EQ	ja	nein	nein	nein

Literatur

M. Sipser
Introduction to the Theory of Computation
PWS Publ.Comp. 1997