Formale Grundlagen der Informatik 1
Kapitel 12
Aufzählbarkeit und (Un-)Entscheidbarkeit (Teil 2)

Frank Heitmann
heitmann@informatik.uni-hamburg.de

10. Mai 2016

Entscheidbarkeit vs. Aufzählbarkeit

Entscheidbarkeit von M:

- Es gibt eine TM A mit $L(A) = M$.
- **und** A hält auf jeder Eingabe (in einem Endzustand, wenn das vorgelegte Wort in M ist, sonst in einem Nicht-Endzustand).

Aufzählbarkeit von M:

- Es gibt eine TM A mit $L(A) = M$.

Definition

Die von Turing-Maschinen akzeptierten Sprachen bilden die Sprachfamilie RE der **aufzählbaren Sprachen**. Die Sprachen, die von TMs akzeptiert werden, so dass die TM zusätzlich auf jeder Eingabe anhält, bilden die Sprachfamilie REC der **entscheidbaren Sprachen**.

Entscheidbare Sprachen

Satz

Folgende Sprachen sind entscheidbar:

$$
\begin{aligned}
DFA_{acc} &:= \{\langle A, w\rangle \mid A \text{ ist ein DFA und akzeptiert } w\} \\
NFA_{acc} &:= \{\langle A, w\rangle \mid A \text{ ist ein NFA und akzeptiert } w\} \\
DFA_{\emptyset} &:= \{\langle A\rangle \mid A \text{ ist ein DFA und } L(A) = \emptyset\} \\
DFA_{=} &:= \{\langle A, B\rangle \mid A \text{ und } B \text{ sind DFAs mit } L(A) = L(B)\} \\
CFG_{acc} &:= \{\langle G, w\rangle \mid G \text{ ist eine CFG und generiert } w\} \\
L_{+1} &:= \{\langle x, y\rangle \mid y = x + 1\} \\
L_{+} &:= \{\langle x, y, z\rangle \mid x + y = z\} \\
L &:= \{\langle G, v\rangle \mid G \text{ ist ein Graph und } v \text{ ein erreichbarer Knoten}\}
\end{aligned}
$$

Ferner ist jede reguläre und jede kontextfreie Sprache entscheidbar.
(Und daneben noch viele, viele mehr ...)

Abschlusseigenschaften

Zum warmwerden:

Satz

1. *Die entscheidbaren Sprachen sind gegenüber $\cap$, $\cup$ und Komplementbildung abgeschlossen.*
2. *Die aufzählbaren Sprachen sind gegenüber $\cap$ und $\cup$ abgeschlossen. Sie sind nicht gegenüber Komplementbildung abgeschlossen.*

Beweis.

Mündlich ... □

Abschlusseigenschaften

Zum warmwerden:

Satz

1. *Die entscheidbaren Sprachen sind gegenüber $\cap$, $\cup$ und Komplementbildung abgeschlossen.*
2. *Die aufzählbaren Sprachen sind gegenüber $\cap$ und $\cup$ abgeschlossen. Sie sind nicht gegenüber Komplementbildung abgeschlossen.*

Beweis.

Mündlich ... $\square$

Das Akzeptanz-/Halteproblem

Satz

Die Sprache

$$TM_{acc} := \{\langle M, w\rangle \mid M \text{ ist eine TM und akzeptiert } w\}$$

ist aufzählbar.

Satz

Die Sprache TM_{acc} ist nicht entscheidbar.

Das Akzeptanz-/Halteproblem

$$TM_{acc} := \{\langle M, w\rangle \mid M \text{ ist eine TM und akzeptiert } w\}$$

Beweis

Wir nehmen an, wir hätten eine TM H, die TM_{acc} entscheidet und führen dies zu einem Widerspruch. H ist also eine TM mit

- H akzeptiert $\langle M, w\rangle$, wenn M das Wort w akzeptiert
- H lehnt $\langle M, w\rangle$ ab, wenn M das Wort w nicht akzeptiert

oder kürzer:

- $H(\langle M, w\rangle) = 1$, wenn $M(w) = 1$
- $H(\langle M, w\rangle) = 0$, wenn $M(w) \neq 1$

Das Akzeptanz-/Halteproblem

$$TM_{acc} := \{\langle M, w\rangle \mid M \text{ ist eine TM und akzeptiert } w\}$$

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

Beweis (Fortsetzung)

Wir konstruieren nun eine TM D, die H als Subroutine benutzt. D nimmt als Eingabe eine TM M (die Eingabe ist also $\langle M\rangle$) und arbeitet wie folgt:

1. Starte H mit Eingabe $\langle M, \langle M\rangle\rangle$
2. Drehe das Ergebnis von H um und gebe es aus

D.h. es ist

- $D(\langle M\rangle) = 1$, wenn M die Eingabe $\langle M\rangle$ nicht akzeptiert
- $D(\langle M\rangle) = 0$, wenn M die Eingabe $\langle M\rangle$ akzeptiert

Das Akzeptanz-/Halteproblem

$$TM_{acc} := \{\langle M, w\rangle \mid M \text{ ist eine TM und akzeptiert } w\}$$

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

Beweis (Fortsetzung)

Wir konstruieren nun eine TM D, die H als Subroutine benutzt. D nimmt als Eingabe eine TM M (die Eingabe ist also $\langle M\rangle$) und arbeitet wie folgt:

1. Starte H mit Eingabe $\langle M, \langle M\rangle\rangle$
2. Drehe das Ergebnis von H um und gebe es aus

D.h. es ist

- $D(\langle M\rangle) = 1$, wenn M die Eingabe $\langle M\rangle$ nicht akzeptiert
- $D(\langle M\rangle) = 0$, wenn M die Eingabe $\langle M\rangle$ akzeptiert

Das Akzeptanz-/Halteproblem

$$TM_{acc} := \{\langle M, w\rangle \mid M \text{ ist eine TM und akzeptiert } w\}$$

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

Beweis (Fortsetzung)

Wir konstruieren nun eine TM D, die H als Subroutine benutzt. D nimmt als Eingabe eine TM M (die Eingabe ist also $\langle M\rangle$) und arbeitet wie folgt:

1. Starte H mit Eingabe $\langle M, \langle M\rangle\rangle$
2. Drehe das Ergebnis von H um und gebe es aus

D.h. es ist

- $D(\langle M\rangle) = 1$, wenn M die Eingabe $\langle M\rangle$ nicht akzeptiert
- $D(\langle M\rangle) = 0$, wenn M die Eingabe $\langle M\rangle$ akzeptiert

Das Akzeptanz-/Halteproblem

Beweis (Fortsetzung)

Insgesamt haben wir jetzt:

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

$$D(\langle M\rangle) = \begin{cases} 1, & \text{wenn } M(\langle M\rangle) \neq 1 \\ 0, & \text{wenn } M(\langle M\rangle) = 1 \end{cases}$$

Was passiert nun, wenn man D mit $\langle D\rangle$ ausführt?

$$D(\langle D\rangle) = \begin{cases} 1, & \text{wenn } D(\langle D\rangle) \neq 1 \\ 0, & \text{wenn } D(\langle D\rangle) = 1 \end{cases}$$

Widerspruch! *Daher kann H nicht existieren!* □

Das Akzeptanz-/Halteproblem

Beweis (Fortsetzung)

Insgesamt haben wir jetzt:

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

$$D(\langle M\rangle) = \begin{cases} 1, & \text{wenn } M(\langle M\rangle) \neq 1 \\ 0, & \text{wenn } M(\langle M\rangle) = 1 \end{cases}$$

Was passiert nun, wenn man D mit $\langle D\rangle$ ausführt?

$$D(\langle D\rangle) = \begin{cases} 1, & \text{wenn } D(\langle D\rangle) \neq 1 \\ 0, & \text{wenn } D(\langle D\rangle) = 1 \end{cases}$$

Widerspruch! *Daher kann H nicht existieren!* □

Das Akzeptanz-/Halteproblem

Beweis (Fortsetzung)

Insgesamt haben wir jetzt:

$$H(\langle M, w\rangle) = \begin{cases} 1, & \text{wenn } M(w) = 1 \\ 0, & \text{wenn } M(w) \neq 1 \end{cases}$$

$$D(\langle M\rangle) = \begin{cases} 1, & \text{wenn } M(\langle M\rangle) \neq 1 \\ 0, & \text{wenn } M(\langle M\rangle) = 1 \end{cases}$$

Was passiert nun, wenn man D mit $\langle D\rangle$ ausführt?

$$D(\langle D\rangle) = \begin{cases} 1, & \text{wenn } D(\langle D\rangle) \neq 1 \\ 0, & \text{wenn } D(\langle D\rangle) = 1 \end{cases}$$

Widerspruch! *Daher kann H nicht existieren!* □

Der Stand bisher

Wichtige Anmerkung

Bezüglich Sprachfamilien wissen wir nun bisher

$$\text{REG} \subsetneq \text{CF} \subsetneq \text{REC} \subsetneq \text{RE}$$

Beispiele für Sprachen, die in den einzelnen Mengen liegen und nicht darunter?

Wichtige Anmerkung

Zur Betonung: Wir wissen, dass es nicht entscheidbare Probleme gibt.

Der Stand bisher

Wichtige Anmerkung

Bezüglich Sprachfamilien wissen wir nun bisher

$$\text{REG} \subsetneq \text{CF} \subsetneq \text{REC} \subsetneq \text{RE}$$

Beispiele für Sprachen, die in den einzelnen Mengen liegen und nicht darunter?

Wichtige Anmerkung

Zur Betonung: Wir wissen, dass es nicht entscheidbare Probleme gibt.

Der weitere Ablauf ...

Was heute noch kommt:

- Eine Sprache, die nicht mal mehr in RE ist
- Technik zum Nachweis der Unentscheidbarkeit

Die Sprache L_d

Man kann die Wörter eines Alphabets aufzählen, z.B. indem man sie erst nach Länge und dann lexikalisch sortiert:

$$a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \ldots$$

So kann man sowohl die Kodierungen aller Turing-Maschinen aufzählen als auch die Wörter, die sie lesen. Wir können dann von den Turing-Maschinen $M_1, M_2, M_3, \ldots$ und den Wörtern $w_1, w_2, w_3, \ldots$ sprechen.

Anmerkung

Genauer kann man hier die lexikalische Ordnung und Gödelisierungen einführen.

Die Sprache L_d

Man kann die Wörter eines Alphabets aufzählen, z.B. indem man sie erst nach Länge und dann lexikalisch sortiert:

$$a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \ldots$$

So kann man sowohl die Kodierungen aller Turing-Maschinen aufzählen als auch die Wörter, die sie lesen. Wir können dann von den Turing-Maschinen $M_1, M_2, M_3, \ldots$ und den Wörtern $w_1, w_2, w_3, \ldots$ sprechen.

Anmerkung

Genauer kann man hier die lexikalische Ordnung und Gödelisierungen einführen.

Die Sprache L_d

Man kann die Wörter eines Alphabets aufzählen, z.B. indem man sie erst nach Länge und dann lexikalisch sortiert:

$$a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \ldots$$

So kann man sowohl die Kodierungen aller Turing-Maschinen aufzählen als auch die Wörter, die sie lesen. Wir können dann von den Turing-Maschinen $M_1, M_2, M_3, \ldots$ und den Wörtern $w_1, w_2, w_3, \ldots$ sprechen.

Anmerkung

Genauer kann man hier die lexikalische Ordnung und Gödelisierungen einführen.

Die Sprache L_d

Satz

$L_d := \{w_i \mid w_i \notin L(M_i)\}$ ist nicht aufzählbar.

Die Sprache L_d

Satz

$L_d := \{w_i \mid w_i \notin L(M_i)\}$ ist nicht aufzählbar.

Beweis

Man kann die Wörter und die TMs in einer Matrix anordnen:

	M_1	M_2	M_3	M_4	$\ldots$	L_d
w_1	1	1	0	0		$Nein(w_1 \notin L_d)$
w_2	1	0	0	0	$\ldots$	$Ja(w_2 \in L_d)$
w_3	0	1	1	1	$\ldots$	$Nein(w_3 \notin L_d)$
w_4	0	0	1	0		$Ja(w_4 \in L_d)$
.	.	.	.	.	.	.

L_d entspricht gerade der Diagonalen, von der nur die Einträge mit 0 in L_d aufgenommen werden.

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d

$$L_d := \{w_i \mid w_i \notin L(M_i)\} \notin RE$$

Das Problem ist, dass eine TM, die L_d akzeptiert, ein M_j in der Matrix sein müsste und dann wird es einen Widerspruch mit w_j geben (akzeptieren/nicht akzeptieren).
Genauer: Angenommen L_d wäre aufzählbar. Dann gibt es eine TM M_j aus der Aufzählung mit $L(M_j) = L_d$. Wir betrachten w_j. Es muss entweder $w_j \in L_d$ oder $w_j \notin L_d$ sein. Beide Fälle führen zu einem Widerspruch:

- $w_j \in L_d$, dann $w_j \notin L(M_j)$ (nach Def. von L_d), dann $w_j \notin L_d$ (nach Annahme $L(M_j) = L_d$).
- $w_j \notin L_d$, dann $w_j \notin L(M_j)$ (nach Annahme $L(M_j) = L_d$), dann $w_j \in L_d$ (Def. von L_d).

Also ist L_d nicht aufzählbar! □

Die Sprache L_d und darüber hinaus...

- L_d ist also nicht einmal aufzählbar!
- Sie ist aber *ab*zählbar!
- Eine Menge, die dann nicht einmal mehr abzählbar ist, ist z.B. die Menge der reellen Zahlen.

Bemerkung

Eine Menge ist abzählbar, wenn sie endlich ist oder eine Bijektion von den natürlichen Zahlen angegeben werden kann.

Die Sprache L_d und darüber hinaus...

- L_d ist also nicht einmal aufzählbar!
- Sie ist aber *ab*zählbar!
- Eine Menge, die dann nicht einmal mehr abzählbar ist, ist z.B. die Menge der reellen Zahlen.

Bemerkung

Eine Menge ist abzählbar, wenn sie endlich ist oder eine Bijektion von den natürlichen Zahlen angegeben werden kann.

Zusammenfassung

Zusammenfassung

$$\text{REG} \subsetneq \text{CF} \subsetneq \text{REC} \subsetneq \text{RE}$$

- In REG: a^*
- In CF, nicht in REG: $a^n b^n$
- In REC, nicht in CF: $a^n b^n c^n$
- In RE, nicht in REC: TM_{acc}
- Nicht in RE: L_d

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Reduktion

Das Ziel

Wir wollen nun weitere Probleme als unentscheidbar nachweisen

Will man eine Sprache L als unentscheidbar nachweisen, so ist das übliche **Vorgehen**:

1. Wir nehmen an L wäre entscheidbar. Sei M eine TM, die L entscheidet.
2. Unter Nutzung von M nun eine TM konstruieren, die ein schon als unentscheidbar nachgewiesenes Problem entscheidet. (M also quasi als Unterroutine benutzen.)
3. Das ist dann ein Widerspruch, also kann M nicht existieren.

Dies nennt man eine *Reduktion*, und sagt, dass das unentscheidbare Problem auf das neue Problem reduziert wurde.

Weitere unentscheidbare Probleme

Satz

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

ist nicht entscheidbar.

Anmerkung

TM_{halt} ist das **Halteproblem**. Dies wird oft in Unentscheidbarkeitsbeweisen benutzt.

Weitere unentscheidbare Probleme

Satz

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

ist nicht entscheidbar.

Beweis

Angenommen TM_{halt} wäre entscheidbar. Sei H eine TM, die TM_{halt} entscheidet. Wir konstruieren nun eine TM S, die TM_{acc} (unter Nutzung von H) entscheidet. Da aber TM_{acc} nicht entscheidbar ist, muss die Annahme falsch sein und daher ist auch TM_{halt} nicht entscheidbar.

Anmerkung

TM_{halt} ist das **Halteproblem**. Dies wird oft in Unentscheidbarkeitsbeweisen benutzt.

Weitere unentscheidbare Probleme

Satz

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

ist nicht entscheidbar.

Beweis

Angenommen TM_{halt} wäre entscheidbar. Sei H eine TM, die TM_{halt} entscheidet. Wir konstruieren nun eine TM S, die TM_{acc} (unter Nutzung von H) entscheidet. Da aber TM_{acc} nicht entscheidbar ist, muss die Annahme falsch sein und daher ist auch TM_{halt} nicht entscheidbar.

Anmerkung

TM_{halt} ist das **Halteproblem**. Dies wird oft in Unentscheidbarkeitsbeweisen benutzt.

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Weitere unentscheidbare Probleme

$$TM_{halt} := \{\langle M, w\rangle \mid M \text{ ist eine TM und hält auf } w\}$$

Beweis (Fortsetzung)

H entscheidet nach Annahme TM_{halt}. S arbeitet nun bei Eingabe $\langle M, w\rangle$ wie folgt:

1. Starte H mit Eingabe $\langle M, w\rangle$
2. Lehnt H ab, lehne ab.
3. Akzeptiert H, simuliere M auf w (M terminiert, da H akzeptiert hat!)
4. Wenn M akzeptiert, akzeptiere; sonst lehne ab.

Wie eben beschrieben, kann TM_{halt} nicht entscheidbar sein. □

Zum Aufschreiben

Zur Nachbereitung

Schreibt man einen solchen Beweis auf (z.B. in den Übungsaufgaben), dann muss das, was eben mündlich einfloss, auch noch dazu. So muss z.B. noch kurz begründet werden, dass die neue TM tatsächlich immer anhält, warum es richtig ist beim zweiten Schritt abzulehnen usw. Also, dass die neue TM tatsächlich TM_{acc} entscheidet. Dann kann, weil wir wissen, dass TM_{acc} nicht entscheidbar ist, geschlussfolgert werden, dass TM_{halt} dies auch nicht ist.

Weitere unentscheidbare Probleme

Satz

$$TM_{print0} := \{\langle M \rangle \mid \textit{gestartet auf } \lambda \textit{ gibt } M \textit{ irgendwann } 0 \textit{ aus}\}$$

ist nicht entscheidbar.

Weitere unentscheidbare Probleme

Satz

$$TM_{print0} := \{\langle M\rangle \mid \textit{gestartet auf } \lambda \textit{ gibt } M \textit{ irgendwann } 0 \textit{ aus}\}$$

ist nicht entscheidbar.

Beweis.

In den Präsenzaufgaben! □

Weitere unentscheidbare Probleme

Definition

Ein *nutzloser Zustand* in einer TM ist ein Zustand, der bei keinem Eingabewort jemals betreten wird. Sei

$$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ ist eine TM und } q \text{ ein nutzloser Zustand von } A\}$$

Bemerkung

Dieses Problem ist eng verwandt mit der Frage, ob eine bestimmte Stelle bspw. in einem Java-Programm jemals erreicht wird.

Weitere unentscheidbare Probleme

Satz

$$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM, } q \text{ nutzloser Zustand von } A\}$$

ist unentscheidbar.

Beweis

Angenommen `UselessState` wäre entscheidbar und A_{US} eine TM, die das Problem entscheidet. Wir wollen damit nun das Halteproblem TM_{halt} entscheiden.

Die Idee

- Zu einer gegebenen TM M mit Zustandsmenge Z und Bandalphabet Γ konstruieren wir eine TM M' mit:
  - einem neuen Zustand z_{neu}
  - für jeden Zustand $z \in Z$ von M und jedes Symbol $x \in \Gamma$, für das es keinen Übergang aus z in M gab (d.h. es gab keine Kante (z, x, X, Y, z') in M mit X, Y, z' beliebig) wird eine neue Kante (z, x, x, R, z_{neu}) hinzugefügt

  Hält M, so kann M' noch einen Übergang nach z_{neu} machen. Dort hält M'. *M hält also genau dann in irgendeinem Zustand, wenn M' in z_{neu} hält.*
- Zu $\langle M, w \rangle$ kann außerdem eine TM M'' konstruiert werden, die bei jeder Eingabe das Band löscht und dann w auf M' startet, wobei M' wie eben beschreiben aus M hervorgeht.

Die Idee

- Zu einer gegebenen TM M mit Zustandsmenge Z und Bandalphabet Γ konstruieren wir eine TM M' mit:
  - einem neuen Zustand z_{neu}
  - für jeden Zustand $z \in Z$ von M und jedes Symbol $x \in \Gamma$, für das es keinen Übergang aus z in M gab (d.h. es gab keine Kante (z, x, X, Y, z') in M mit X, Y, z' beliebig) wird eine neue Kante (z, x, x, R, z_{neu}) hinzugefügt

  Hält M, so kann M' noch einen Übergang nach z_{neu} machen. Dort hält M'. *M hält also genau dann in irgendeinem Zustand, wenn M' in z_{neu} hält.*
- Zu $\langle M, w \rangle$ kann außerdem eine TM M'' konstruiert werden, die bei jeder Eingabe das Band löscht und dann w auf M' startet, wobei M' wie eben beschreiben aus M hervorgeht.

Die Idee

- Zu einer gegebenen TM M mit Zustandsmenge Z und Bandalphabet Γ konstruieren wir eine TM M' mit:
  - einem neuen Zustand z_{neu}
  - für jeden Zustand $z \in Z$ von M und jedes Symbol $x \in \Gamma$, für das es keinen Übergang aus z in M gab (d.h. es gab keine Kante (z, x, X, Y, z') in M mit X, Y, z' beliebig) wird eine neue Kante (z, x, x, R, z_{neu}) hinzugefügt

  Hält M, so kann M' noch einen Übergang nach z_{neu} machen. Dort hält M'. *M hält also genau dann in irgendeinem Zustand, wenn M' in z_{neu} hält.*
- Zu $\langle M, w \rangle$ kann außerdem eine TM M'' konstruiert werden, die bei jeder Eingabe das Band löscht und dann w auf M' startet, wobei M' wie eben beschreiben aus M hervorgeht.

Die Idee

- Zu einer gegebenen TM M mit Zustandsmenge Z und Bandalphabet Γ konstruieren wir eine TM M' mit:
  - einem neuen Zustand z_{neu}
  - für jeden Zustand $z \in Z$ von M und jedes Symbol $x \in \Gamma$, für das es keinen Übergang aus z in M gab (d.h. es gab keine Kante (z, x, X, Y, z') in M mit X, Y, z' beliebig) wird eine neue Kante (z, x, x, R, z_{neu}) hinzugefügt

  Hält M, so kann M' noch einen Übergang nach z_{neu} machen. Dort hält M'. *M hält also genau dann in irgendeinem Zustand, wenn M' in z_{neu} hält.*
- Zu $\langle M, w \rangle$ kann außerdem eine TM M'' konstruiert werden, die bei jeder Eingabe das Band löscht und dann w auf M' startet, wobei M' wie eben beschreiben aus M hervorgeht.

Die Idee

- Zu einer gegebenen TM M mit Zustandsmenge Z und Bandalphabet Γ konstruieren wir eine TM M' mit:
  - einem neuen Zustand z_{neu}
  - für jeden Zustand $z \in Z$ von M und jedes Symbol $x \in \Gamma$, für das es keinen Übergang aus z in M gab (d.h. es gab keine Kante (z, x, X, Y, z') in M mit X, Y, z' beliebig) wird eine neue Kante (z, x, x, R, z_{neu}) hinzugefügt

  Hält M, so kann M' noch einen Übergang nach z_{neu} machen. Dort hält M'. *M hält also genau dann in irgendeinem Zustand, wenn M' in z_{neu} hält.*
- Zu $\langle M, w \rangle$ kann außerdem eine TM M'' konstruiert werden, die bei jeder Eingabe das Band löscht und dann w auf M' startet, wobei M' wie eben beschreiben aus M hervorgeht.

Die Idee

Zwischenstand

- Wenn M auf w hält, dann hält M'' bei jeder Eingabe in z_{neu}.
- Wenn M nicht auf w hält, dann besucht M'' z_{neu} niemals.

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Abschluss

$\texttt{UselessState} = \{\langle A, q\rangle \mid A \text{ TM}, q \text{ nutzloser Zustand von } A\}$

Bei Eingabe $\langle M, w\rangle$

1. Konstruiere aus $\langle M, w\rangle$ die TM M''.
2. Starte A_{US} auf $\langle M'', z_{neu}\rangle$.
3. Akzeptiert A_{US}, dann ist z_{neu} ein nutzloser Zustand und nach obigem hält dann M nicht auf w und wir lehnen ab.
4. Lehnt A_{US} ab, so ist z_{neu} nicht nutzlos, wird also besucht. Dann aber hält M auf w an und wir akzeptieren.

Dies entscheidet das Halteproblem, was nicht sein kann! A_{US} existiert also nicht! □

Zum Aufschreiben

Zur Nachbereitung

Oft ist man nicht ganz so konkret wie in dem vorherigen Beispiel, wo explizit Kanten angegeben wurden. Es reicht z.B. in leichter Variation zu sagen, dass eine TM M'' konstruiert wird, die ihre Eingabe ignoriert und die stets M auf w simuliert. M'' wechselt, sollte M anhalten, dann noch in einen neuen Zustand z_{neu}, der sonst nicht besucht wird. (Wichtig hierfür ist sich klar zu machen, dass M'' tatsächlich M simulieren kann und prüfen kann, ob M anhält.) Der Rest des Beweises geht ganz analog. Wieder ist allerdings zu begründen, was hier mündlich einfloss, dass die neue TM stets anhält und tatsächlich das Halteproblem entscheidet.

Zusammenfassung

Wir haben gestern und heute

- entscheidbare und
- aufzählbare

Sprachen kennengelernt und gesehen, dass

- $REG \subsetneq CF \subsetneq REC \subsetneq RE$

gilt!

Ferner haben wir das Halteproblem und weitere **nicht entscheidbare** Probleme kennengelernt und eine Technik (Reduktion), wie man neue Probleme als unentscheidbar nachweisen kann.

Ausblick

Wir wissen nun, dass es viele entscheidbare Probleme und leider auch wichtige unentscheidbare Probleme gibt. Für die Praxis genügt es aber nicht, dass ein Problem entscheidbar ist! Die "**Kosten**" sind auch wichtig!

- Wie lange brauchen wir um das Problem zu lösen?
- Wie viel Platz/Speicher benötigen wir?

Damit beschäftigen wir uns zum Ende der Veranstaltung noch... (und dann vor allem in AuD).

Und auch die unentscheidbaren kann man angehen, in dem man versucht die Problemstellung zu variieren und ähnliches. Dies ist aufgrund der Bedeutung der Probleme (z.B. das Halteproblem für die Verifikation) nötig.

Ausblick

Wir wissen nun, dass es viele entscheidbare Probleme und leider auch wichtige unentscheidbare Probleme gibt. Für die Praxis genügt es aber nicht, dass ein Problem entscheidbar ist! Die "**Kosten**" sind auch wichtig!

- Wie lange brauchen wir um das Problem zu lösen?
- Wie viel Platz/Speicher benötigen wir?

Damit beschäftigen wir uns zum Ende der Veranstaltung noch... (und dann vor allem in AuD).

Und auch die unentscheidbaren kann man angehen, in dem man versucht die Problemstellung zu variieren und ähnliches. Dies ist aufgrund der Bedeutung der Probleme (z.B. das Halteproblem für die Verifikation) nötig.

Ausblick

Wir wissen nun, dass es viele entscheidbare Probleme und leider auch wichtige unentscheidbare Probleme gibt. Für die Praxis genügt es aber nicht, dass ein Problem entscheidbar ist! Die "**Kosten**" sind auch wichtig!

- Wie lange brauchen wir um das Problem zu lösen?
- Wie viel Platz/Speicher benötigen wir?

Damit beschäftigen wir uns zum Ende der Veranstaltung noch... (und dann vor allem in AuD).

Und auch die unentscheidbaren kann man angehen, in dem man versucht die Problemstellung zu variieren und ähnliches. Dies ist aufgrund der Bedeutung der Probleme (z.B. das Halteproblem für die Verifikation) nötig.

Ausblick

Nach den Pfingstferien:

- Aussagenlogik
  - Syntax & Semantik
  - Folgerbarkeit, Äquivalenz
  - Normalformen
  - Resolution
- Prädikatenlogik
- Einführung in die Komplexitätstheorie
  - Zeit- und Platzkomplexität
  - Die Komplexitätsklassen P und NP
  - NP-Vollständigkeit

Literaturhinweis

Der Logikteil wird auf dem Buch "Logik für Informatiker" von Uwe Schöning basieren (siehe Literaturliste).

Wiederholung

Wiederholung
(der Vorlesungen 1-12)

Überblick

Wir hatten die Sprachfamilien

- der regulären Sprachen (REG),
- der kontextfreien Sprachen (CF),
- der entscheidbaren Sprachen (REC) und
- der aufzählbaren Sprachen (RE).

Dabei gilt

- $\mathrm{REG} \subsetneq \mathrm{CF} \subsetneq \mathrm{REC} \subsetneq \mathrm{RE}$

Reguläre Sprachen - Modelle

Die Sprachfamilie der regulären Sprachen wird erfasst von:

- deterministischen, endlichen Automaten (DFAs)
- nichtdeterministischen, endlichen Automaten (NFAs)
  - mit/ohne λ-Kanten
- reguläre/rationale Ausdrücke
- (rechts-)lineare Grammatiken

Reguläre Sprachen - Begriffe

Begriffe:

- Zustände, Eingabesymbole, Alphabet, Überführungsfunktion, Übergangsrelation, Startzustände, Endzustände,
- vollständig, initial zusammenhängend,
- Konfiguration, Rechnung, akzeptierte Sprache,
- deterministisch, nichtdeterministisch,
- Abschlusseigenschaften

Reguläre Sprachen - Verfahren

Techniken/Verfahren:

- Beweis, dass ein Automat eine Sprache akzeptiert (zwei Richtungen zu zeigen!)
- Techniken zum Konstruieren eines Automaten
- Potenzautomatenkonstruktion
- Verschiedene Konstruktionen, um Abschlusseigenschaften zu zeigen
- Pumping Lemma der regulären Sprachen

Kontextfreie Sprachen - Modelle

Die Sprachfamilie der kontextfreien Sprachen wird erfasst von:

- kontextfreien Grammatiken
- (nichtdeterministische Kellerautomaten)

Kontextfreie Sprachen - Begriffe

Begriffe bei Grammatiken:

- Nonterminale, Terminale, Produktionen, Startsymbol,
- Ableitung, erzeugte/generierte Sprache
- λ-Produktion

(Begriffe zum PDA)

- wie beim DFA/NFA

Kontextfreie Sprachen - Verfahren

Techniken/Verfahren:

- Beweis, dass eine Grammatik eine Sprache generiert (zwei Richtungen zu zeigen!)
- Techniken zum Konstruieren einer Grammatik
- Herstellung der Chomsky-Normalform
  - Technik zum Reduzieren (produktiv, erreichbar)
- Verschiedene Konstruktionen, um Abschlusseigenschaften zu zeigen
- Pumping Lemma der kontextfreien Sprachen

Entscheidbare und aufzählbare Sprachen - Modelle

Die Sprachfamilie der aufzählbaren Sprachen wird erfasst von:

- Turing-Maschinen
  - deterministische (DTMs)
  - nichtdeterministische (NTMs)
  - einseitig/beidseitig unendliches Band
  - mehrere Bänder
  - ...
- jedes Modell, dass die intuitiv berechenbaren Funktionen erfasst ($\Rightarrow$ Church-Turing-These)

Entscheidbare und aufzählbare Sprachen - Begriffe

Begriffe:

- Zustände, Eingabesymbole, Bandsymbole, Überführugnsfunktion, Überführungsrelation, Startzustand, Endzustände, Symbol für das leere Feld,
- Konfiguration
- Schrittrelation, Rechnung, Erfolgsrechnung, akzeptierte Sprache
- (Turing-)berechenbar
- aufzählbar, entscheidbar

Entscheidbare und aufzählbare Sprachen - Verfahren

Techniken/Verfahren:

- TM konstruieren, die eine Sprache akzeptiert
- TM konstruieren, die eine Funktion berechnet
- TM konstruieren, die eine Sprache entscheidet
- Verschiedene Konstruktionen, um Abschlusseigenschaften zu zeigen
- Beweisen, dass ein Problem unentscheidbar ist

Was wir ausgelassen haben...

Ausgelassen haben wir...

- Kellerautomaten (PDAs) (nur skizziert)
  - Akzeptanz mit leerem Keller
  - Akzeptanz mit Endzustand
- deterministische Kellerautomaten (DPDAs)
- linear beschränkte Automaten (LBAs)
- kontextsensitive Grammatiken (Typ-1)

Anmerkung

Mit den letzteren beiden hätte man die Sprachfamilie der kontextsensitiven Sprachen (CS). Dies führt zu

- $\text{REG} \subsetneq \text{CF} \subsetneq \text{CS} \subsetneq \text{REC} \subsetneq \text{RE}$

Eine Sprache, die in CS ist, nicht aber in CF ist $a^n b^n c^n$. Eine Sprache, die dann in REC ist, aber nicht in CS, ist der Äquivalenztest für reguläre Ausdrücke mit Exponenten.

Zusammenfassung

Zusammenfassung

Hier solltet ihr euch auskennen:

$$\text{REG} \subsetneq \text{CF} \subsetneq \text{REC} \subsetneq \text{RE}$$

Für jede Sprachfamilie: Automatenmodell? Grammatik? Typische Sprachen? Sprachen, die in der einen sind, in der anderen nicht mehr? Nachweis davon (Konstruktionsmethoden und Techniken zum Grenzen zeigen wie Pumping Lemma, Reduktion usw.)? Abschlusseigenschaften? Weitere Verfahren wie z.B. Potenzautomatenkonstruktion, Produktautomatenkonstruktion, Erstellung der Chomsky-Normalform? ...