

Automaten, Formale Sprachen und Berechenbarkeit I

Wichtige Begriffe

- Eine partielle Funktion ist eine Relation $f \subseteq A \times B$; für jedes $x \in \text{dom}(f)$ gibt es ein $y \in \text{range}(f)$ mit $x f y$; wir schreiben statt $f \subseteq A \times B$ und $x f y$ zukünftig $f : A \rightarrow B$ und $y = f(x)$.
- Eine Funktion f ist total, wenn $\text{dom}(f) = A$ gilt.
- Eine Funktion $f : A \rightarrow B$ ist
 - injektiv, falls $x \neq y$ impliziert $f(x) \neq f(y)$;
 - surjektiv, falls $B = \text{range}(f)$ und
 - bijektiv, falls f sowohl injektiv als auch surjektiv ist.
- Eine Menge M ist abzählbar, wenn es eine Injektion von M nach \mathbb{N} gibt.
- Zwei Mengen A und B haben die selbe Kardinalität, wenn es eine bijektive Funktion $f : A \rightarrow B$ gibt.

- Ein WHILE-Programm besteht aus den Instruktionen $X := 0$, $X := succ(Y)$, $X := pred(Y)$ und der Kontrollflußanweisung

while $X \neq Y$ do ... od

Hier bezeichnen X, Y stets Variablen und keine Konstanten.

- Eine (partielle) Funktion $\psi : \mathbb{N} \rightarrow \mathbb{N}$ heißt (effektiv) berechenbar, falls es ein WHILE-Programm P gibt mit $\psi = \varphi_P$, wobei φ_P jene Funktion ist, die P berechnet.
- Man kann alle WHILE-Programme aufzählen: $\varphi_0, \varphi_1, \varphi_2, \dots$
(Gödelisierung, Arithmetisierung der Syntax)
- Ein Ja-Nein Problem ist entscheidbar, falls es eine totale berechenbare 0-1 Funktion (charakteristische Funktion) gibt, die auf den Ja-Instanzen 1 und auf den Nein-Instanzen 0 ausgibt.

- Verschiedene Techniken:

- Church-Turing These
- Diagonalisierung
- Enumerationstheorem: es gibt eine Funktion $\Phi(e, x) = \varphi_e(x)$; Φ heißt universelle Funktion.
- Reduktion: löse Problem A mit Hilfe von Problem B .
- Parametrisierung (s-m-n Theorem):

$$\varphi_{s_n^m(i, y_1, \dots, y_m)}(z_1, \dots, z_n) = \varphi_i(y_1, \dots, y_m, z_1, \dots, z_n),$$

wobei s_n^m total und berechenbar ist.

- Rekursionstheorem: für jede totale berechenbare Funktion f gibt es ein n mit

$$\varphi_n(x) = \varphi_{f(n)}(x);$$

φ_n ist Fixpunkt.

- Beispiele für nicht berechenbare Funktionen: Halteproblem, allgemeines Halteproblem, Totalitätsproblem, Äquivalenzproblem, ...

- Ein akzeptables Programmiersystem ist eine Aufzählung A_0, A_1, \dots von Algorithmen bezüglich einer algorithmischen Spezifikation, mit der man die j -stelligen Funktionen $\alpha_0^{(j)}, \alpha_1^{(j)}, \dots$ assoziieren kann und die 3 Eigenschaften erfüllt:
 - Jede berechenbare Funktion hat einen Index in den $\alpha^{(j)}$'s.
 - Es gibt eine universelle Funktion für die $\alpha^{(j)}$'s.
 - Es gilt das s-m-n Theorem.
- Rogers Isomorphie Theorem: Zwischen je zwei akzeptablen Programmiersystemen gibt es eine bijektive, totale und berechenbare Funktion f mit $\alpha_i^{(j)} = \beta_{f(i)}^{(j)}$.
- Konsequenz: Berechenbarkeitstheorie ist *unabhängig* von dem gewählten Formalismus.

- Ein LOOP-Programm besteht aus den Instruktionen $X := 0$, $X := succ(Y)$, $X := pred(Y)$ und der Kontrollflußanweisung **loop X do ... od**. Hier bezeichnen X, Y stets Variablen und keine Konstanten.
- Eine Funktion $\psi : \mathbb{N} \rightarrow \mathbb{N}$ heißt LOOP-berechenbar, falls es ein LOOP-Programm P gibt mit $\psi = \varphi_P$, wobei φ_P jene Funktion ist, die P berechnet.
- Man kann alle LOOP-Programme aufzählen: $\psi_0, \psi_1, \psi_2, \dots$ (Gödelisierung, Arithmetisierung der Syntax)
- Eigenschaften:
 - LOOP-berechenbare Funktionen sind immer total.
 - Die “busy-beaver” Funktion für LOOP-Programme ist nicht LOOP-berechenbar (analog für WHILE-Programme und Berechenbarkeit).
 - Die Ackermann-Funktion ist nicht LOOP-berechenbar.
 - Es gibt kein LOOP-universelles LOOP-Programm.

- Eine Menge A ist entscheidbar (rekursiv, lösbar) falls ihre charakteristische Funktion eine totale und berechenbare Funktion ist.
- Eine Menge A ist rekursiv aufzählbar (semi-entscheidbar, r.e.), falls entweder $A = \emptyset$ oder $A = \text{range}(f)$ für eine totale und berechenbare Funktion f (Enumerationsfunktion) ist.
- Eigenschaften:
 - Jede endliche Menge ist rekursiv.
 - Das Komplement einer rekursiven Menge ist rekursiv.
 - Die Vereinigung und der Schnitt rekursiver Mengen ist rekursiv (analog für r.e.).
 - Jede rekursive Menge ist rekursiv aufzählbar.

- Charakterisierung von rekursiven Mengen:
 - A ist rekursiv genau dann wenn A und \overline{A} rekursiv aufzählbar sind.
 - Unendliches A ist rekursiv, genau dann wenn A rekursiv aufzählbar in aufsteigender Reihenfolge ist.
- Charakterisierung von rekursiv aufzählbaren Mengen:
 - A ist r.e. genau dann wenn $A = \text{dom}(f)$ für eine berechenbare Funktion f .
 - A ist r.e. genau dann wenn $A = \text{range}(f)$ für eine berechenbare Funktion f .
- Die r.e. Mengen können aufgezählt werden: W_0, W_1, \dots mit $W_i = \text{dom}(\varphi_i)$.

- Eine Menge $I \subseteq \mathbb{N}$ respektiert Funktionen, wenn aus $i \in I$ und $\varphi_i = \varphi_j$ folgt: $j \in I$.
- Für zwei Funktionen f und g schreiben wir $f \leq g$, wenn aus $f(x)$ definiert folgt $g(x)$ definiert und $f(x) = g(x)$.
- Satz von Rice: Sei I eine Menge von Indizes, die Funktionen respektiert. Dann ist I rekursiv genau dann wenn $I = \emptyset$ oder $I = \mathbb{N}$. (“Jede nichttriviale Eigenschaft über den Funktionen ist unentscheidbar”).
- 2. Satz von Rice: Sei I eine Menge von Indizes, die Funktionen respektiert. Wenn eine berechenbare Funktion Θ existiert mit
 - $\{i \mid \varphi_i = \Theta\} \subseteq I$ und
 - es gibt eine berechenbare Funktion $\hat{\Theta}$ mit $\Theta \leq \hat{\Theta}$ und die Menge $\{i \mid \varphi_i = \hat{\Theta}\} \subseteq \bar{I}$,

dann ist die Menge I nicht r.e.

- 3. Satz von Rice: Sei I eine Menge von Indizes, die Funktionen respektiert. Wenn eine berechenbare Funktion Θ existiert mit
 - $\{i \mid \varphi_i = \Theta\} \subseteq I$ und
 - $\{i \mid \varphi_i \leq \Theta \text{ und } \text{dom}(\varphi_i) \text{ endlich}\} \subseteq \bar{I}$,

dann ist die Menge I nicht r.e.