... bis zur Terminierung
für Eingabekonfiguration dargestellt durch n
(falls TM terminiert)

2.3.2 Äquivalenz RM/TM-Berechenbarkeit

Jede RM läßt sich in eine TM übersetzen. Dazu brauchen wir eine Darstellung der Konfig. der RM durch eine TM:

(1) Die Registerinhalte werden auf dem TM-Band dargestellt (z.B. Strichzahlen).

(2) Aus RM-Programm konstruieren wir den endlichen Automaten, der die TM steuert. succ, pred, ;, while ... jeweils mit dem Kopf zur Stelle i auf dem Band fahren, ...

Zu einer geg. TM kann ich eine RM konstruieren, die TM simuliert. Idee: Konfig. der TM werden gödelisiert und in den Registern dargestellt. Die Fahrbewegungen und Zustandsübergänge der TM werden durch Programmschritte der RM simuliert.

2.3.3 Church's These

Alonzo Church (1936):

Jede intuitiv berechenbare Funktion ist μ-rekursiv. nicht beweisbar, da „intuitiv berechenbar" genau genug def.

wir haben eine Reihe von Berechenbarkeitsbegriffen kennengelernt, die jeweils auf das gleiche Konzept von „berechenbarer Funktion" führen (Churchsche These).

Wir wenden uns nun der Frage zu, welche Funktionen berechenbar / nicht berechenbar und welche Prädikate entscheidbar (d.h. durch Algorithmen eindeutig mit ja / nein beantwortbar) sind.

2.4.1 Nichtberechenbare Funktionen

Jeder Formalismus zur Berechenbarkeit (TM, RM, MR) definiert Programme / Algorithmen durch Wörter über einem Zeichensatz. D.h. in jedem Fall existiert eine formale Sprache $PRG \subseteq A^*$ mit geeignetem Zeichensatz A, so dass gilt: jeder Algorithmus wird dargestellt durch ein Wort $p \in PRG$. Für jede berechenbare Funktion f existiert ein $p \in PRG$, das f berechnet.

Die Menge der Funktionen

$$\mathbb{N}^m \to \mathbb{N}$$

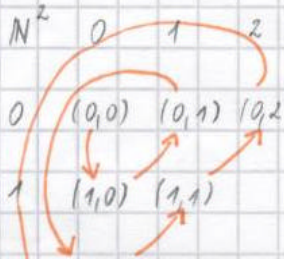
ist überabzählbar! (Ergebnis der Mengenlehre)

$\mathbb{N}$ ist abzählbar.

A^* ist abzählbar (falls A endlich ist).

PRG ist abzählbar.

Dies ergibt sofort: Es gibt viel mehr Funktionen $\mathbb{N}^m \to \mathbb{N}$ als Programme in PRG, d.h. fast jede Funktion ist nicht berechenbar.



$\mathbb{N}^2$ ist abzählbar

$\mathbb{N}^k$ ist abzählbar

wir können Programme durch Zahlen codieren, d. h. es existiert eine Gödelisierung

$$code : PRG \to \mathbb{N}$$

Satz: Es existiert eine totale Funktion $g: \mathbb{N} \to \mathbb{N}$, die nicht berechenbar ist.

Beweis: Wir spezifizieren g durch (für alle $n \in \mathbb{N}$):

$$g(n) = \begin{cases} f_p(n) + 1 & \text{falls } \exists p \in PRG : code(p) = n \text{ und } f_p \text{ definiert für Argument } n \\ 0 & \text{sonst} \end{cases}$$

Dabei sei für $p \in PRG$ die Funktion $f_p : \mathbb{N} \to \mathbb{N}$ die durch p berechnete partielle Funktion.

Achtung: Die Spezifikation von g ist konsistent und eindeutig, da code eine injektive Funktion ist.

Annahme: g ist berechenbar! Dann existiert ein Programm $q \in PRG$ mit $f_q = g$. Mit $code(q) = m$ gilt:

$$g(m) = f_q(m) + 1 = g(m) + 1 \quad \lightning \quad \square$$

Bemerkung: g ist zunächst nur von theoretischem Interesse.

2. 4. 2 (Nicht-) Entscheidbare Prädikate

Eine ja/nein-Frage entspricht logisch einem Prädikat.
Ein Prädikat

$$p : M \to \mathbb{B}$$

kann auch als Funktion

$$p' : M \to \{0, 1\} \subseteq \mathbb{N}$$

aufgefasst werden. Damit können wir das Konzept der Berechenbarkeit auf Prädikate übertragen. Ein „berechenbares"

heißt

(1) entscheidbar, wenn es einen Algorithmus gibt, der für jedes Argument $m \in M$ terminiert und korrekt 1 für $p(m) = true$ und 0 für $p(m) = false$ ausgibt.

(2) positiv semientscheidbar, wenn es einen Algorithmus gibt, der für jedes Argument $m \in M$ terminiert und korrekt 1 ausgibt, falls $p(m) = true$ und falls $p(m) = false$ gilt, entweder 0 ausgibt oder nicht terminiert.

(3) negativ semientscheidbar, falls $\neg p$ positiv semientscheidbar ist.

Beispiele:

(1) Entscheidbare Probleme:

- Gleichheit zweier Zahlen in Binärdarstellung
- Primzahleigenschaft
- Zugehörigkeit eines Wortes zum Sprachschatz einer Ch.-3-Spr.
- Existenz der Nullstelle eines Polynoms über den ganzen Zahlen.

(2) Positiv semientscheidbare (i.A. aber nicht allgemein entscheidbare) Prädikate:

- Terminierung einer gegebenen TM für bestimmte Eingaben.
- Zugehörigkeit eines Wortes zum Sprachschatz einer gegebenen Ch.-0-Sprache.

(3) Negativ semientscheidbar:

- Gleichheit zweier durch primitive Rekursion gegebener Funktionen

(4) Unentscheidbare Probleme (weder positiv noch negativ semientscheidbar):

- Terminierung einer der μ-Rekursion beschriebenen Funktion

Satz: Das Terminierungsproblem für μ-rek. Funktionen ist nicht entscheidbar.

Beweis: Sei μ-PRG die Menge der Wörter die μ-rekursive Programme darstellen.

Widerspruchsannahme: Terminierungsproblem ist entscheidbar!

Dann ex. μ-rek. Programm $p \in \mu\text{-PRG}$, das eine Funktion f_p berechnet so dass gilt:

$$f_p(x) = \begin{cases} 0 & \text{falls } \exists q \in \mu\text{-PRG} : code(q) = x \wedge f_q \text{ für } x \text{ nicht def.} \\ \text{undef} & \text{sonst} \end{cases}$$

Wir erhalten für $m = code(p)$:

$$f_p(m) = \begin{cases} 0 & \text{falls } f_p(m) \text{ undef.} \\ \text{undefiniert} & \text{sonst} \end{cases} \quad \lightning$$

Widerspruch, da $f_p(m)$ entweder undef. ist (dann wäre $f_p(m) = 0$ $\lightning$) oder $f_p(m) = 0$ (dann wäre $f_p(m)$ undef. $\lightning$). □

2.4.3 Rekursive und rekursiv aufzählbare Mengen

Wir betrachten jetzt eine Obermenge U sowie Teilmenge $M \subseteq U$.

Ein Prädikat auf U

$p : U \to \mathbb{B}$ — <u>Charakteristische Prädikat</u> zu $M \subseteq U$.

ist dann gegeben durch

$$p(x) = \begin{cases} true & \text{falls } x \in M \\ false & \text{falls } x \notin M \end{cases}.$$

Eine Menge heißt <u>rekursiv</u>, wenn das charakteristische Prädikat entscheidbar ist.

Satz: Jede Ch.-1-Sprache ist rekursiv. □

heißt Aufzählung der Menge M, falls

$$M = \{e(n) : n \in \mathbb{N}\} = \bigcup_{n \in \mathbb{N}} \{e(n)\}.$$

M heißt rekursiv aufzählbar, falls e berechenbar ist.

Jede rekursiv aufzählbare Menge hat ein positiv semientscheidbares charakteristisches Prädikat. $e(0) \stackrel{?}{=} x$, $e(1) \stackrel{?}{=} x$, $e(2) \stackrel{?}{=} x$, ...
solange bis x gefunden