Zwei Berechnungsmodelle: Turingmaschinen, Register-Programme

Allgemeines Format: Zeichenmanipulation über fixiertem endlichem Alphabet; endliche Zeichenreihen (Wörter) über diesem Alphabet bilden das Datenmaterial (input/output und auch Arbeitsdaten).

$\mathbb{A}$ endliches, nicht-leeres Alphabet

$\mathbb{A}^*$ die (abzählbar unendliche Menge aller Wörter über $\mathbb{A}$

Zu berechnen sind primär (partielle) Funktionen $f\colon \mathbb{A}^* \xrightarrow{\text{part}} \mathbb{A}^*$.

Konstitutiv für Algorithmen/Berechnungverfahren sind einfache und lokale Elementaroperationen sowie die regelhafte und uniforme Steuerung der Abfolge dieser Operationen. Es gibt viele unterschiedliche Modellierungen, die sich aber als äquivalent erweisen.

Turingmaschinen (nach Alan Turing, 1912-54)

Speichermedium: beidseitig unendliches Band von Bandzellen, die leer sein können ($\Box$) oder Zeichen $a \in \mathbb{A}$ aufnehmen können. Kompletter Bandinhalt lässt sich als durch $i \in \mathbb{Z}$ indizierte Folge von Zeichen in $\mathbb{A} \cup \{\Box\}$ beschreiben, mit $\Box$ an fast allen Stellen.

Zugriff: Lese-/Schreibkopf in aktueller Kopfposition am Band. Liest das aktuelle Zeichen in der Bandzelle und kann es überschreiben.

Kontrolle/Steuerung: endliche Zustandsmenge Q (Kontrollzustände); aktueller Zustand $q \in Q$ und aktuell gelesenes Zeichen bestimmen (anhand einer Übergangsfunktion) die Schreibaktion, eine Kopfbewegung um max. einen Schritt nach links oder rechts und den Nachfolgezustand.

$$\begin{aligned}
\mathcal{M} = \;& (\mathbb{A}, Q, q_0, \delta, F) \\
Q \;& \textit{Zustandsmenge (endlich),} \\
q_0 \in Q \;& \textit{Startzustand,} \\
F \subseteq Q \;& \textit{Haltezustände,} \\
\delta \;& \textit{Übergangsfunktion} \\
& \delta\colon Q \times (\Sigma \cup \{\Box\}) \to (\Sigma \cup \{\Box\}) \times \{<, \circ, >\} \times Q.
\end{aligned}$$

Berechnung/Lauf auf Eingabewort $w \in \mathbb{A}^*$: Initialisierung mit Kopf in Position 0; Eingabewort in Bandzellen $1, \ldots, |w|$, Band ansonsten leer, Startzustand q_0. Abfolge von Konfigurations-Übergängen anhand der Übergangsfunktion durch lokale Modifikation, bis ggf. ein als Haltezustand gekennzeichneter Kontrollzustand erreicht wird. Für so terminierende Läufe kann man z.B. die Ausgabekonvention vereinbaren, dass das Wort rechts vom Kopf (in der Stopkonfiguration) die Ausgabe ist. Will man z.B. nur Boolesche Werte 0 oder 1 als Ausgabe, kann man auch zwei Haltezustände q^- und q^+ auszeichnen und den erreichten Haltezustand als Ausgabe interpretieren. Ein nicht terminierender Lauf wird als divergent bezeichnet und liefert kein Ergebnis. Daher wird i.d.R. eine partielle Funktion berechnet. Symbolische Kurznotation:

$w \xrightarrow{\mathcal{M}} \infty$	die Berechnung von $\mathcal{M}$ auf w divergiert.
$w \xrightarrow{\mathcal{M}} \text{STOP}$	die Berechnung von $\mathcal{M}$ auf w terminiert.
$w \xrightarrow{\mathcal{M}} w'$	$\mathcal{M}$ berechnet w' als Ausgabe auf Eingabe w.

Registermaschinen und -Programme

Speichermedium: endliche Folge von Registern R_i, die Wörter $w_i \in \mathbb{A}^*$ aufnehmen können.

Zugriff/Elementaroperationen: Letztes/oberstes Symbol eines Registers kann entfernt werden ($R_i := R_i - a$ streicht ggf. ein endständiges a in w_i; lässt w_i unverändert sonst); ein Symbol kann angefügt werden ($R_i := R_i + a$ macht aus w_i den Inhlat $w_i a$); das letzte Symbol von w_i wird auch für bedingte Verzweigungen im Programmlauf benuzt (s.u.).

Kontrolle/Steuerung: das Register-Programm. Ein Registerprogramm ist eine endliche Folge von durchlaufend nummerierten Programmzeilen

$$\begin{array}{lll} P\colon & 0 & \langle \text{Anweisung } 0 \rangle \\ & \vdots & \\ & \ell & \langle \text{Anweisung } \ell \rangle \\ & \vdots & \\ & k & \text{STOP} \end{array}$$

deren letzte die Stopanweisung ist. Als sonstige Anweisungen stehen zur Verfügung:

- $R_i := R_i - a$ bzw. $R_i := R_i + a$ für jedes $a \in \mathbb{A}$
- bedingte Verzweigung: IF $R_i = \Box$ THEN ℓ_0 ELSE ℓ_1 OR $\cdots$ OR ℓ_r

Die bedingte Verzweigung nimmt Bezug auf eine feste Numerierung der Zeichen in $\mathbb{A}$ als $\mathbb{A} = \{a_1, \ldots, a_r\}$; im Programmlauf verzweigt die Berechnung in Zeile ℓ_i je nachdem ob R_i leer ist oder der aktuelle Inhalt von R_i in a_i endet.

Berechnung/Lauf auf Eingabewort $w \in \mathbb{A}^*$ (oder Tupel von Wörtern): Initialisierung in Zeile 0, Eingabewort(tupel) in Registern. Abfolge von Konfigurations-Übergängen durch Ausführung der Anweisung in der jeweils aktuellen Programmzeile mit Übergang in die nächste Zeile bzw. gemäß Verzweigungsbefehl.

Wie bei Turingmaschinen können Berechnungen terminieren oder divergieren; also werden wieder i.a. partielle Funktionen berechnet. Symbolische Notation wie oben.

Grundlegende allgemeine Definitionen

Berechenbarkeit einer partiellen Funktion $f\colon \mathbb{A}^* \xrightarrow{\text{part}} \mathbb{A}^*$:

Existenz einer Turingmaschine $\mathcal{M}$, die f berechnet, d.h.:
$\mathrm{Def}(f) = \{w \in \mathbb{A}^* : w \xrightarrow{\mathcal{M}} \text{STOP}\}$ und $w \xrightarrow{\mathcal{M}} f(w)$ für alle $w \in \mathrm{def}(f)$.

Entscheidbarkeit einer Menge $R \subseteq \mathbb{A}^*$:

Existenz einer Turingmaschine $\mathcal{M}$, die die charakteristische Funktion von $R \subset \mathbb{A}^*$, $\chi_R\colon \mathbb{A}^* \to \{0,1\}$, berechnet, d.h.:

$$w \xrightarrow{\mathcal{M}} \begin{cases} 1 & \text{für } w \in R, \\ 0 & \text{für } w \notin R. \end{cases}$$

Rekursive Aufzählbarkeit einer Menge $R \subseteq \mathbb{A}^*$:

Existenz einer Turingmaschine $\mathcal{M}$, die genau auf den Eingaben aus R terminiert:

$$w \xrightarrow{\mathcal{M}} \begin{cases} \text{STOP} & \text{für } w \in R, \\ \infty & \text{für } w \notin R, \end{cases}$$

Äquivalent: Berechenbarkeit einer totalen Funktion mit $\mathrm{Bild}(f) = R$.

Bemerkung: Fast alle Teimengen $R \subseteq \mathbb{N}$ sind unentscheidbar und nicht einmal rekursiv aufzählbar (warum?). Ein konkretes Beispiel für eine rekursiv aufzählbare, aber nicht entscheidbare Menge liefert das *Halteproblem*.