Vorlesungsmitschrift

Algorithmen und Berechenbarkeit

Vorlesung 11

Letztes Update: 2017/12/09 - 13:22 Uhr

Berechenbarkeit

Zentrale Fragen dieses zweiten großen Kapitels sind zum Beispiel: *Was kann man berechnen? Was sind geeignete Rechenmodelle?* oder *Was kann man effizient berechnen?*

Grundlagen: Notation

Für die Fragestellungen der Berechenbarkeit ist meist ein endliches Alphabet gegeben, dass man als Σ_i bezeichnet, also zum Beispiel $\Sigma = \{0, 1\}$. Die Menge aller Wörter der Länge k über Σ werden notiert als Σ^k. Für $\Sigma = \{0, 1\}, k = 3$ also

$$\{0, 1\}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$$

Grundlagen: Probleme

Probleme sind *Aufgaben/Fragestellungen*, die mit einem Rechner gelöst/berechnet werden sollen. Im Folgenden werden einige Probleme beispielhaft aufgezeigt.

Problem 1

→ Eingabe: eine binär kodierte Zahl $q \in \mathbb{N}$ für die außerdem gilt $q \geq 2$

→ Ausgabe: ein binär kodierter Primfaktor dieser Zahl

- Beispiel: Für die Eingabe der Binärzahl `110` soll die Ausgabe also `011` bzw. `010` lauten.

⇒ Dieses Problem entspricht der Relation

$$\begin{aligned} &R \subseteq \{0, 1\}^* \times \{0, 1\}^q \\ &R = \{(x, y) \in \{0, 1\}^* \times \{0, 1\}^* \mid \quad x = \text{bin}(q) \\ &\qquad\qquad\qquad\qquad\qquad\qquad\quad y = \text{bin}(p) \\ &\qquad\qquad\qquad\qquad\qquad\qquad q, p \in \mathbb{N} \\ &\qquad\qquad\qquad\qquad\qquad\qquad\quad q \geq 2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\quad p \text{ teilt } q \qquad \} \end{aligned}$$

Problem 2

→ Eingabe: zwei binär kodierte Zahlen $a, b \in \mathbb{N}$

→ Ausgabe: eine binär kodierte Zahl $c \in \mathbb{N}$ für die gilt: $c = a \cdot b$

Dieses Problem entspricht der Funktion

$$f : \Sigma^* \to \Sigma^*$$

Binärzahlkonkatenationen sind unschön, da sie je nach Leseart unterschiedliche Werte annehmen können. Die Zahl `1001011` kann zum Beispiel so verstanden werden: $\underbrace{\texttt{100}}_{4}\underbrace{\texttt{1011}}_{11}$ oder $\underbrace{\texttt{1001}}_{9}\underbrace{\texttt{011}}_{3}$ ...
Zur Vereinfachung wird deshalb das Alphabet um ein Trennsymbol # erweitert: $\Sigma = \{0, 1, \#\}$. Eine binär kodierte Eingabe ist dann: $\text{bin}(a)\#\text{bin}(b)$ und die ganze Aussage

$$f(\text{bin}(a)\#\text{bin}(b)) = \text{bin}(c) \text{ mit } c = a \cdot b$$

Problem 3

Graphkodierungen: Gegeben ist ein Graph $G(V, E)$ mit $|V| = n$. Dieser Graph könnte als $\text{bin}(n)\# < n^2$ kodiert und seine und Adjazenzmatrix folgendermaßen aussehen:

	0	**1**	**2**	**3**	**4**
0		1			
1	1				1
2					
3				1	
4		1			

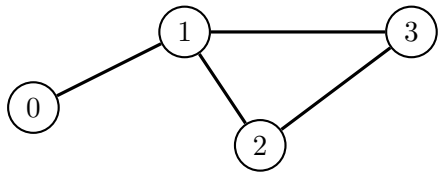
Jeder Graph hat eine solch eindeutige Kodierung aber nicht jeder String über $\{\texttt{0}, \texttt{1}, \#\}$ ist eine gültige Kodierung für einen Graphen. Sei $\mathcal{G}$ die Menge aller Graphen, $\mathcal{G}_Z \subseteq \mathcal{G}$ die Menge aller zusammenhängender Graphen und code(G) für $G \in \mathcal{G}$ die Kodierung eines Graphen G.

→ Eingabe: Kodierung eines ungerichteten Graphen $G(V, E)$

→ Ausgabe: `1` falls zusammenhängend, `0` sonst

Die diesem Problem entsprechende Sprache kann dargestellt werden als

$$L = \{w \in \{\texttt{0}, \texttt{1}, \#\}^* \mid \exists G(V, E) \in \mathcal{G}_Z \text{ mit } w = \text{code}(G)\}$$



	0	**1**	**2**	**3**
0	0	1	0	0
1	1	0	1	1
2	0	1	0	1
3	0	1	1	0

$$\text{code}(G) = \texttt{100\#0100101101010110}$$

Die Sprache L enthält also alle Eingaben, die einen zusammenhängenden Graphen kodieren.

Das Komplement von L ist definiert als $I := \Sigma^* \backslash L$ und enthält alle Eingaben, die der Kodierung eines nicht zusammenhängenden Graphen entsprechen aber auch alle Eingaben, die gar keiner Kodierung eines Graphen entsprechen.

Die Turingmaschine (TM)

Eine TM arbeitet auf einem beidseitig unendlichen Speicherband, das aus Zellen besteht:

···	B	B	B	0	1	1	1	1	0	1	B	B	···

Jede Zelle enthält ein Element des Bandalphabets Γ, zum Beispiel $\{\texttt{0}, \texttt{1}, \texttt{B}\}$. Des Weiteren hat die TM einen Schreib-Lese-Kopf (SLK), der über das Band wandern, und jeweils den Inhalt der Zelle, über der er steht, lesen und schreiben kann. Zusätzlich besitzt die TM eine Zustandsmenge Q und zu jedem Zeitpunkt befindet sich die TM in einem $q \in Q$.

Um mit der TM arbeiten zu können, müssen noch einige Regeln eingeführt werden:

- Am Anfang einer Berechnung befindet sich die TM im Anfangs-/Startzustand $q_0 \in Q$.
- Auf dem Band steht die Eingabe von links nach rechts, alle anderen Zellen des Bandes enthalten `B`'s.
- Der SLK steht auf dem ersten Zeichen der Eingabe
- Das Eingabealphabet ist definiert als $\Sigma \subseteq \Gamma \backslash \{\texttt{B}\}$
- Die Eingabeverarbeitung der TM ist durch eine Übergangsrelation definiert:

$$\delta : (Q\backslash\{\bar{q}\} \times \Gamma \times Q \times \Gamma \times \{L, R, N\})$$

$Q\backslash\{\bar{q}\}$ → aktueller Zustand

Γ → aktuelles Zeichen

Q → neuer Zustand

Γ → neues Zeichen (zu schreiben)

$\{L, R, N\}$ → Kopfbewegung

  Falls für den aktuellen Zustand und dem aktuellen Zeichen nur eine Aktion möglich ist, spricht man von einer *Übergangsfunktion*:

$$\delta : (Q\backslash\{\bar{q}\} \times \Gamma) \to Q \times \Gamma \times \{R, L, N\}$$

  Für $(q, a, q', a', d) \in \delta$ bzw. $\delta(q, a) = (q', a', d)$ mit $d \in \{R, L, N\}$ kann die Übergangsfunktion also gelesen werden als: Wenn sich die TM im Zustand q befindet und ein a unter dem SLK einließt, schreibt sie ein a' an diese Stelle, bewegt den SLK nach d und nimmt den Zustand q' ein.

Den Übergangsschritt nennt man auch Rechenschritt. Eine TM führt solange Rechenschritte aus, bis der Endzustand $\bar{q}$ erreicht wird. Ist δ eine Relation, so handelt es um eine Nichtdeterministische TM (NTM), ist δ eine Funktion, dann ist die TM deterministisch (DTM).

Zusammenfassung

Eine Turingmaschine ist ein 7-Tupel

$$(Q, \Sigma, \Gamma, B, q_o, \bar{q}, \delta)$$

Q	$\rightarrow$ endliche Zustandsmenge	
Σ	$\rightarrow$ endliches Eingabealphabet	
Γ	$\rightarrow$ endliches Bandalphabet	$\Gamma \supset \Sigma$
B	$\rightarrow$ Blankzeichen	$B \subset \Gamma \backslash \Sigma$
q_o	$\rightarrow$ Anfangszustand	$q_0 \subset Q$
$\bar{q}$	$\rightarrow$ Endzustand	$\bar{q} \subset Q$
δ	$\rightarrow$ Übergangsrelationen/-funktionen	

Konfiguration

Eine Konfiguration einer TM ist ein Schnappschuss der Berechnung einer TM zu einem bestimmten Zeitpunkt. Die Konfiguration beschreibt vollständig alle Details, die notwendig sind, um die Berechnung fortzusetzen (also Bandinhalt, Kopfposition und Zustand).

Formal: Eine Konfiguration ist ein String $\alpha q \beta$ mit $q \in Q$ und $\alpha, \beta \in \Gamma^*$. Das bedeutet, auf dem Band befindet sich wie gehabt α, eingerahmt von Blankzeichen. Der aktuelle Zustand ist q und der SLK befindet sich auf dem ersten Zeichen von β.

Man nennt $\alpha' q' \beta'$ *direkte Nachfolgekonfiguration* von $\alpha q \beta$, falls $\alpha' q' \beta'$ in einem Rechenschritt aus $\alpha q \beta$ entsteht. Man schreibt $\alpha q \beta \vdash \alpha' q' \beta'$.
Man nennt $\alpha'' q'' \beta''$ *Nachfolgekonfiguration* von $\alpha q \beta$, falls $\alpha'' q'' \beta''$ in endlich vielen Rechenschritten aus $\alpha q \beta$ entsteht. Man schreibt $\alpha q \beta \vdash \alpha'' q'' \beta''$.

Eine Rechnung der TM terminiert, sobald der Endzustand $\bar{q}$ erreicht wird. Die Laufzeit ist definiert als die Anzahl der Rechenschritte, die die TM bis zur Terminierung ausführt. Falls die Rechnung nicht terminiert, ist die Laufzeit unbeschränkt. Der Platzbedarf einer TM-Rechnung ist die Anzahl der besuchten Bandzellen (kann auch unbeschränkt sein).

TM-berechenbare Funktionen und Sprachen

Eine TM $\mathcal{M}$, die für jede Eingabe terminiert, berechnet eine totale Funktion $f_{\mathcal{M}} : \Sigma^* \rightarrow \Sigma^*$. Wenn der Endzustand $\bar{q}$ erreicht ist, kann das Ergebnis $f_{\mathcal{M}}(x)$ rechts vom SLK vom Band gelesen werden. Das Ergebnis kann auch ϵ sein, falls der SLK nach der Beendigung auf einem Zeichen $\notin \Sigma$ steht.

Einige TMs terminieren auf manchen Eingaben nicht, man erweitert deshalb $f_{\mathcal{M}}$ auf

$$f_{\mathcal{M}} : \Sigma^* \rightarrow \Sigma^* \cup \{\perp\}$$
$$\perp \rightarrow \textit{nicht definiert}$$

Definition: Eine Funktion $f_{\mathcal{M}} : \Sigma^* \rightarrow \Sigma^* \cup \{\perp\}$ heißt TM-berechenbar, wenn es eine TM $\mathcal{M}$ gibt, mit $f_{\mathcal{M}} = f$. Diese Definition schließt auch Funktionen ein, die für manche Eingaben keine wohldefinierte Ausgaben haben.

Im Spezialfall, dass $f : \Sigma^* \to \{0, 1\}$ (das bedeutet, man hat ein Entscheidungsproblem), soll eine TM eine Eingabe $w \in \Sigma^*$ akzeptieren, wenn sie terminiert und die Ausgabe mit einer `1` beginnt. Die TM verwirft $w \in \Sigma^*$, falls sie terminiert, aber die Ausgabe nicht mit einer `1` beginnt (Die TM wird nun nicht auf jeder Eingabe terminieren).

Definition: Eine Sprache $L \subseteq \Sigma^*$ heißt TM-entscheidbar, wenn es eine TM gibt, die auf allen Eingaben hält, die Eingabe w akzeptiert falls $w \in L$, und die Eingabe verwirft falls $w \notin L$.