

Theoretische Informatik I: Komplexität und formale Sprachen

Kurzschrift zur Vorlesung von Van Bang Le

17. November 2017

Inhaltsverzeichnis

0	Wichtige Begriffe aus der Berechenbarkeitstheorie	3
1	Komplexität	4
1.1	Graphen	4
1.2	Zeitaufwand und Komplexitätsklassen	4
1.3	Polynomielle Reduktion und NP-Vollständigkeit	6
2	Formale Sprachen	7

0 Wichtige Begriffe aus der Berechenbarkeitstheorie

Kernfrage: Ist ein Problem algorithmisch lösbar?

Definition 0.0.1 (Turingmaschine) Eine *deterministische Turingmaschine* (DTM) (oder ein det. Turingprogramm) M ist ein 5-Tupel $M = (Z, \Sigma, \delta, z_a, z_e)$ mit:

- Z ist eine endliche Menge von Zuständen
- z_a ist der ausgezeichnete Anfangszustand
- z_e ist der ausgezeichnete Endzustand
- Σ ist eine endliche Menge, das *Bandalphabet*
 \square ist ein ausgezeichnetes Symbol in Σ : es heißt *Leersymbol* (Blank) und zeigt an, dass die Bandzelle leer ist
- $Z \cap \Sigma = \emptyset$
- $\delta : Z \times \Sigma \rightarrow Z \times \Sigma \times \{+1, -1, 0\}$ ist eine nicht notwendig überall definierte Funktion, die *Übergangsfunktion*

Definition 0.0.2 (Turing-berechenbar) Eine (partielle) Funktion $f : \Sigma^* \rightarrow \Sigma^*$ heißt *Turingprogramm-berechenbar*, falls eine DTM M existiert mit $f = f_M$.

Definition 0.0.3 (Entscheidbarkeit) Eine Menge $A \subseteq \mathbb{N}$ (bzw. $A \subseteq \mathbb{N}^k$ oder $A \subseteq \Sigma^*$) heißt *entscheidbar*, wenn die charakteristische Funktion $\chi_A : \mathbb{N} \rightarrow \{0, 1\}$ (bzw. $\chi_A : \mathbb{N}^k \rightarrow \{0, 1\}$ oder $\chi_A : \Sigma^* \rightarrow \{0, 1\}$) von A ,

$$\chi_A(x) = \begin{cases} 1 & \text{falls } x \in A \\ 0 & \text{sonst, also falls } x \notin A \end{cases}$$

berechenbar ist.

1 Komplexität

Kernfrage: Wie schwierig ist ein lösbares Problem?

1.1 Graphen

Definition 1.1.1 Ein (ungerichteter, einfacher) Graph G ist ein Paar $G = (V, E)$ bestehend aus **Knotenmenge** V und **Kantenmenge** $E \subseteq \binom{V}{2}$.
 $\binom{V}{2}$ steht hierbei für die Menge aller 2-elementigen Teilmengen von V .

Definition 1.1.2 Sei $G = (V, E)$ ein Graph. Dann gilt:

1. Zwei Knoten $x, y \in V$ sind **verbunden**, wenn $\{x, y\} \in E$ ist.
2. Eine Menge $Q \subseteq V$ von Knoten ist eine **Clique**, wenn je zwei Knoten in Q verbunden sind.
3. Eine Menge $U \subseteq V$ von Knoten ist eine **unabhängige Menge** (independent set), wenn je zwei Knoten in U unverbunden sind.

1.2 Zeitaufwand und Komplexitätsklassen

Definition 1.2.1 (\mathcal{O} -Notation) Für Funktionen $f, g : \mathbb{N} \rightarrow \mathbb{R}$ schreiben wir $f \in \mathcal{O}(g)$ oder auch $f = \mathcal{O}(g)$, falls es eine Konstante $c > 0$ gibt mit: Es existiert ein n_0 , sodass $f(n) \leq c \cdot g(n)$ für alle $n \geq n_0$ gilt. Man sagt, dass f asymptotisch höchstens so stark wächst wie g .

Formal: $f \in \mathcal{O}(g) \Leftrightarrow \exists c > 0 \exists n_0 \in \mathbb{N} \forall n \geq n_0 : f(n) \leq c \cdot g(n)$

Definition 1.2.2 (Zeitaufwand von DTM-Programmen) Sei M eine DTM über dem Alphabet Σ . Der **Zeitaufwand** $t_M(w)$ von M bei Eingabe $w \in \Sigma^*$ ist

$$t_M(w) = \begin{cases} \text{Zahl der Konfigurationsübergänge der Be-} & \text{falls Berechnung abbricht} \\ \text{rechnung von } M \text{ bei Eingabe } w & \\ \infty & \text{sonst} \end{cases}$$

Der **Zeitaufwand** $t_M(n)$ von M bei Eingaben der Codierungslänge n ist

$$t_M(n) = \max\{t_M(w) \mid w \in \Sigma^n\}$$

Definition 1.2.3 Es sei $f : \mathbb{N} \rightarrow \mathbb{N}$ eine Funktion. Mit **DTIME(f)** bezeichnen wir die Menge aller Entscheidungsprobleme, die sich durch eine DTM M mit Zeitaufwand $t_m = \mathcal{O}(f)$ entscheiden lassen:

$$\text{DTIME}(f) = \{L \subseteq \Sigma^* \mid L \text{ ist entscheidbar durch eine DTM } M \text{ mit } t_m = \mathcal{O}(f)\}$$

Definition 1.2.4 (Komplexitätsklasse \mathbf{P}) $P = \bigcup_{k=0}^{\infty} DTIME(n^k)$ ist die Klasse aller in (deterministisch) polynomiell Zeit aufwand lösbaren (Entscheidungs-) Probleme. P ist also die Klasse von Problemen, die effizient gelöst werden können.

Definition 1.2.5 (Komplexitätsklasse $\mathbf{EXPTIME}$) $EXPTIME = \bigcup_{k=0}^{\infty} DTIME(2^{n^k})$ ist die Klasse aller in (deterministisch) exponentiellem Zeit aufwand lösbaren (Entscheidungs-) Probleme. (Es gilt: $P \subseteq EXPTIME$)

Definition 1.2.6 (Nichtdeterministische Turingmaschine) Eine *nichtdeterministische Turingmaschine* (NTM) (oder ein nichtdet. Turingprogramm) M ist ein 5-Tupel $M = (Z, \Sigma, \delta, z_a, z_e)$ mit:

- Z, z_a, z_e, Σ sind definiert wie bei einer deterministischen Turingmaschine
- $\delta \subseteq (Z \times \Sigma) \times (Z \times \Sigma \times \{+1, -1, 0\})$ ist eine *Relation*, die *Übergangsrelation*

In einem nichtdeterministischen Turingprogramm δ kann es zu einem Paar $(z, x) \in Z \times \Sigma$ mehr als einen Befehl mit der linken Seite z, x geben.

Definition 1.2.7 (Nichtdeterministische Berechnung) Jedem Eingabewort $w \in \Sigma^*$ kann man einen „Berechnungsbaum“ zuordnen, dessen maximale Pfade den möglichen Berechnungen entsprechen:

- Die Wurzel des Berechnungsbaums ist mit der Anfangskonfiguration $z_a w$ beschriftet
- Ist v ein Knoten des Baums, der mit der Konfiguration $K = \alpha z x \beta$ markiert ist und ist $\delta(z, x) = \{(z_1, x_1, \lambda_1), \dots, (z_r, x_r, \lambda_r)\}$, so hat v genau r Söhne, die jeweils mit den Nachfolgekonfigurationen K_i von K bezüglich (z_i, x_i, λ_i) beschriftet sind, $i = 1, \dots, r$.

Definition 1.2.8 (Nichtdeterministische Entscheidbarkeit) Eine NTM M entscheidet die Menge $L \subseteq \Sigma^*$, falls der Berechnungsbaum jedes Eingabewortes $w \in \Sigma^*$ endlich ist und für $w \in L$ mindestens einen erfolgreichen Berechnungspfad enthält.

Satz 1.2.1 Sei $L \subseteq \Sigma^*$. Dann ist L genau dann (deterministisch) entscheidbar, wenn L nichtdeterministisch entscheidbar ist.

Definition 1.2.9 (Zeitaufwand von NTM-Programmen) Sei M eine NTM über dem Alphabet Σ . Der *Zeitaufwand* $t_M(w)$ von M bei Eingabe $w \in \Sigma^*$ ist

$$t_M(w) = \begin{cases} \text{Tiefe des Berechnungsbaumes von } M \text{ bei } w & \text{falls Baum endlich} \\ \infty & \text{sonst} \end{cases}$$

Der *Zeitaufwand* $t_M(n)$ von M bei Eingaben der Codierungslänge n ist $t_M(n) = \max\{t_M(w) \mid w \in \Sigma^n\}$

Definition 1.2.10 Es sei $f : \mathbb{N} \rightarrow \mathbb{N}$ eine Funktion. Mit $\text{NTIME}(f)$ bezeichnen wir die Menge aller Entscheidungsprobleme, die sich durch eine NTM M mit Zeitaufwand $t_m = O(f)$ entscheiden lassen:

$$\text{NTIME}(f) = \{L \subseteq \Sigma^* \mid L \text{ ist entscheidbar durch eine NTM } M \text{ mit } t_m = \mathcal{O}(f)\}$$

Definition 1.2.11 (Komplexitätsklasse NP) $NP = \bigcup_{k=0}^{\infty} \text{NTIME}(n^k)$ ist die Klasse aller in nichtdeterministisch polynomiell Zeit aufwand lösbaren (Entscheidungs-) Probleme.

Satz 1.2.2 $P \subseteq NP \subseteq EXPTIME$

1.3 Polynomielle Reduktion und NP-Vollständigkeit

Definition 1.3.1 (Polynomielle Reduktion) Seien $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$ zwei Entscheidungsprobleme. L_1 ist auf L_2 *polynomiell reduzierbar*, wenn es eine überall definierte, polynomiell berechenbare Funktion $f : \Sigma_1^* \rightarrow \Sigma_2^*$ gibt, so dass für alle $x \in \Sigma_1^*$ gilt:

$$x \in L_1 \Leftrightarrow f(x) \in L_2$$

Ist L_1 polynomiell reduzierbar auf L_2 via f , so schreiben wir: $L_1 \leq_p L_2$

Definition 1.3.2 (NP-Vollständigkeit) Es sei $L \subseteq \Sigma^*$. L heißt *NP-vollständig*, falls gilt:

- (i) $L \in NP$
- (ii) $\forall M \in NP : M \leq_p L$

Lemma 1.3.1 Ist L NP-vollständig, so gilt: $L \in P \Leftrightarrow P = NP$

Satz 1.3.1 (Satz von Cook und Levin) SAT ist NP-vollständig.

Satz 1.3.2 Ist A NP-vollständig, $A \leq_p B$ und $B \in NP$, so ist B NP-vollständig.

Satz 1.3.3 3-SAT ist NP-vollständig.

Satz 1.3.4 CLIQUE ist NP-vollständig.

Satz 1.3.5 INDSET und VERTEX COVER sind NP-vollständig.

Satz 1.3.6 3-Färbbarkeit ist NP-vollständig.

Satz 1.3.7 SUBSET-SUM ist NP-vollständig.

2 Formale Sprachen