A. Komplexitätstheorie

Definition A.1
Sei $f : \mathbb{N} \to \mathbb{N}$. DTIME($f(n)$), NTIME($f(n)$), DSPACE($f(n)$) und NSPACE($f(n)$) sind die Klassen aller Probleme, die von einer (nicht-)deterministischen Turingmaschine in Zeit/Platz $f(n)$ auf einer Eingabe der Länge n entschieden werden können. Dann definieren wir folgende Komplexitätsklassen.

$$\begin{aligned}
\text{P} &:= \bigcup_{p(n)\ \text{Polynom}} \text{DTIME}(p(n)) \\
\text{NP} &:= \bigcup_{p(n)\ \text{Polynom}} \text{NTIME}(p(n)) \\
\text{PSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{DSPACE}(p(n)) \\
\text{NPSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{NSPACE}(p(n)) \\
\text{EXPTIME} &:= \bigcup_{p(n)\ \text{Polynom}} \text{DTIME}(2^{p(n)}) \\
\text{NEXPTIME} &:= \bigcup_{p(n)\ \text{Polynom}} \text{NTIME}(2^{p(n)}) \\
\text{EXPSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{DSPACE}(2^{p(n)}) \\
\text{NEXPSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{NSPACE}(2^{p(n)}) \\
\text{2-EXPTIME} &:= \bigcup_{p(n)\ \text{Polynom}} \text{DTIME}(2^{2^{p(n)}})
\end{aligned}$$

Satz A.1 (ohne Beweis)
P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ NPSPACE $\subseteq$ EXPTIME $\subseteq$ NEXPTIME $\subseteq$ EXPSPACE $\subseteq$ NEXPSPACE $\subseteq$ 2-EXPTIME

Definition A.2
Sei $\mathcal{K}$ eine Komplexittsklasse. Dann ist co-$\mathcal{K} := \{L \mid L \notin \mathcal{K}\}$.

Lemma A.1
Es gelten: P = co-P, PSPACE = co-PSPACE, EXPTIME = co-EXPTIME, EXPSPACE = co-EXPSPACE, 2-EXPTIME = co-2-EXPTIME.

Beweis Bei deterministischen Turing Maschinen lassen sich leicht die Antworten "akzeptieren" und "verwerfen" vertauschen, ohne asymptotischen Mehraufwand. ∎

Eine Problem L ist in co-$\mathcal{K}$, wenn sich mit "Aufwand" $\mathcal{K}$ widerlegen statt beweisen lsst, dass eine vorgelegt Eingabe zu L gehrt.

Satz A.2 (Savitch [Sav69])
NPSPACE = PSPACE, NEXPSPACE = EXPSPACE.

Eine nicht-deterministische Turingmaschine kann man auch als Spiel zwischen einem aktiven und einem passiven Spieler auffassen. Der aktive Spieler wählt in jeder Konfiguration eine Nachfolgekonfiguration, der passive Spieler wartet nur ab. Gelingt es dem aktiven Spieler, eine akzeptierende Konfiguration zu erreichen, dann gewinnt er. Ansonsten gewinnt der passive Spieler.

Da der aktive Spieler im Gewinnfalle die Existenz einer akzeptierenden Berechnung belegt, nennen wir ihn auch den *existentiellen Spieler*. Der passive wird auch *universeller Spieler* genannt.

Dieses Modell lässt sich auf sogenannte *alternierende Turingmaschinen* erweitern. Dabei lassen wir auch den universellen Spieler aktiv werden.

Definition A.3
Eine alternierende Turingmaschine ist ein Tupel $\mathcal{A} = (Q, Q_\exists, Q_\forall, \Sigma, \Gamma, q_0, \delta, q_{acc}, q_{rej})$ wobei

- $Q = Q_\exists \uplus Q_\forall \uplus \{q_{acc}, q_{rej}\}$ die Zustandsmenge disjunkt in existentielle, universelle und je einen akzeptierenden und verwerfenden Zustand zerlegt ist,
- $\Sigma \subseteq \Gamma$ das Eingabe- und das Arbeitsalphabet sind,
- $q_0 \in Q$ der Anfangszustand ist,
- $\delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{-1, 0, 1\}$ die Transitionsrelation ist.

Die Berechnung einer alternierenden Turingmaschine $\mathcal{A}$ auf einem Wort w ist ein Baum mit folgenden Eigenschaften:

- die Wurzel ist beschriftet mit der Startkonfiguration $q_0 w$;
- genau die Blätter sind beschriftet mit Konfigurationen der Form vqv', wobei $q \in \{q_{acc}, q_{rej}\}$;
- ist ein Knoten beschriftet mit einer Konfiguration vqv', so dass $q \in Q_\exists$, dann hat dieser Knoten genau einen Nachfolger, welcher beschriftet ist mit einer Nachfolgekonfiguration von vqv';
- ist ein Knoten beschriftet mit einer Konfiguration vqv', so dass $q \in Q_\forall$, dann hat dieser Knoten einen Nachfolger für jede mögliche Nachfolgekonfiguration von vqv'.

$\mathcal{A}$ akzeptiert das Wort w, wenn es einen endlichen Berechnungsbaum gibt, dessen Blätter alle von der Form $vq_{acc}v'$ sind. Die von einer alternierenden Turingmaschine $\mathcal{A}$ akzeptierte Sprache ist $L(\mathcal{A}) = \{w \in \Sigma^* \mid \mathcal{A} \text{ akzeptiert } w\}$.

Eine alternierende Turingmaschine $\mathcal{A}$ heißt *platzbeschränkt* durch eine Funktion $f : \mathbb{N} \to \mathbb{N}$, wenn für jeden Berechnungsbaum auf einem Wort w gilt: ist ein Knoten beschriftet mit einer Konfiguration vqv', dann gilt $|vv'| \leq f(|w|)$.

$\mathcal{A}$ heisst *zeitbeschränkt* durch $f : \mathbb{N} \to \mathbb{N}$, wenn für jeden Berechnungsbaum auf einem Wort w gilt: die Tiefe des Baums ist höchstens $f(|w|)$.

Definition A.4
Sei $f : \mathbb{N} \to \mathbb{N}$. Dann sind ATIME($f(n)$) und ASPACE($f(n)$) die Klassen aller Probleme, die von einer durch $f(n)$ zeit-, bzw. platzbeschränkten alternierenden Turingmaschine akzeptiert werden. Wir definieren die folgenden Komplexitätsklassen.

$$\begin{aligned}
\text{ALOGSPACE} &:= \bigcup_{c \in \mathbb{N}} \text{ASPACE}(c \cdot \log n) \\
\text{APTIME} &:= \bigcup_{p(n)\ \text{Polynom}} \text{ATIME}(p(n)) \\
\text{APSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{ASPACE}(p(n)) \\
\text{AEXPTIME} &:= \bigcup_{p(n)\ \text{Polynom}} \text{ATIME}(2^{p(n)}) \\
\text{AEXPSPACE} &:= \bigcup_{p(n)\ \text{Polynom}} \text{ASPACE}(2^{p(n)})
\end{aligned}$$

Satz A.3 (Chandra/Kozen/Stockmeyer [CKS81])
Es gelten: ALOGSPACE = P, APTIME = PSPACE, APSPACE = EXPTIME, AEXPTIME = EXPSPACE, AEXPSPACE = 2-EXPTIME.

Damit ergibt sich die in Abb. A.1 gezeigt Hierarchie an Komplexitätsklassen.

Man kann den bekannten Begriff der NP-Härte auf alle diese Klassen erweitern. Da P-Härte unter Reduktionen, die in polynomieller Zeit berechenbar sind, jedoch ein triviales Konzept ist, definieren wir zuerst einen stärkeren Reduktionsbegriff.

Definition A.5
Seinen $L, L' \subseteq \Sigma^*$. $L \leq_p L'$ gdw. es eine Abbildung $f : \Sigma^* \to \Sigma^*$ gibt, so dass

- für alle $w \in \Sigma^*$: $w \in L$ gdw. $f(w) \in L'$,
- die Abbildung f ist in polynomieller Zeit berechenbar.

Die Definition von $L \leq_l L'$ unterscheidet sich von dieser nur dadurch, dass in der zweiten Klausel "polynomielle Zeit" durch "logarithmischen Platz" ersetzt wird.

Eine Sprache L heisst $\mathcal{K}$-hart unter P-Reduktionen, falls für alle $L' \in \mathcal{K}$ gilt: $L' \leq_p L$. L heisst $\mathcal{K}$-hart unter Logspace-Reduktionen, falls für alle $L' \in \mathcal{K}$ gilt: $L' \leq_l L$.

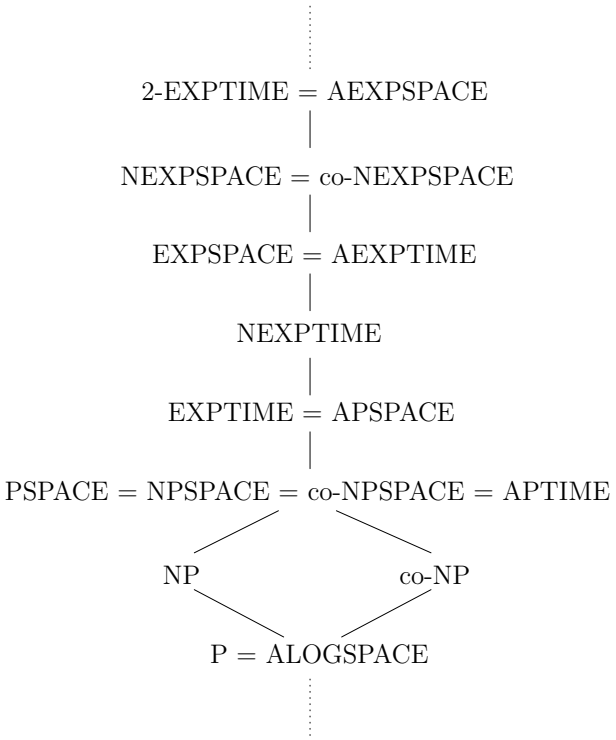

Abbildung A.1.: Einige Komplexitätsklassen und ihre Beziehungen zueinander.