Theoretische Informatik 2

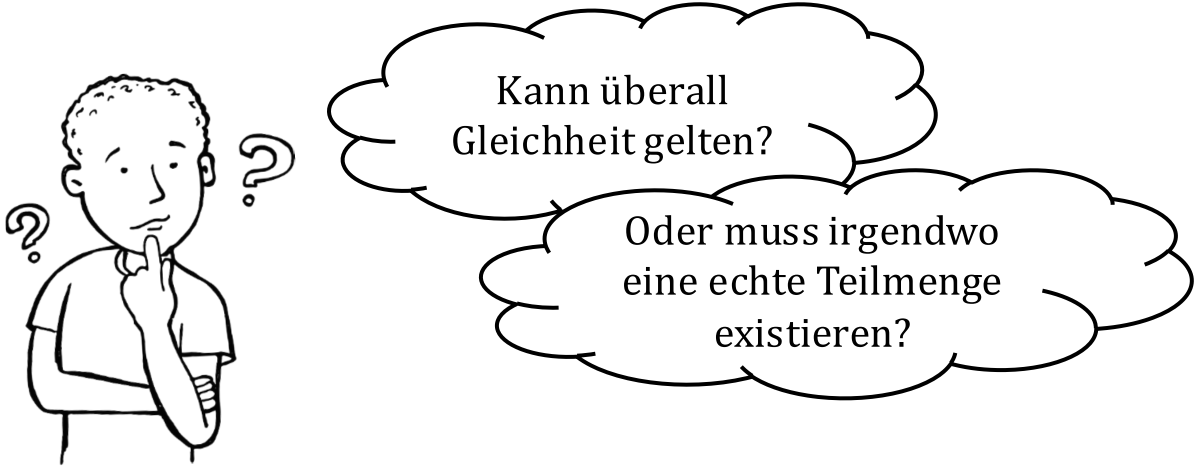
Arne Schmidt

Inklusionen

Zur Erinnerung

Theorem 7.32
Die robusten Komplexitätsklassen bilden eine aufsteigende Kette,

$$\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{NPSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{NEXP} \subseteq \mathrm{EXPSPACE} \subseteq \cdots$$

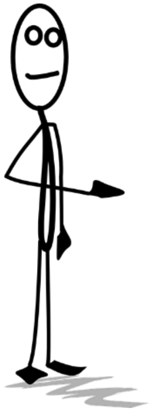


Heute

Zur Erinnerung

Theorem 7.32
Die robusten Komplexitätsklassen bilden eine aufsteigende Kette,

$$\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{NPSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{NEXP} \subseteq \mathrm{EXPSPACE} \subseteq \cdots$$



Wir zeigen heute: Es kann nicht überall Gleichheit gelten.

Wir wissen nur nicht, wo in dieser Kette echte Teilmengen existieren.

Genauer zeigen wir:
$\mathrm{NL} \subsetneq \mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE}$,
$\mathrm{P} \subsetneq \mathrm{EXP}$, $\mathrm{NP} \subsetneq \mathrm{NEXP}$

Kapitel 13 – Hierarchiesätze

Klein-o-Notation

Definition 13.1
Zu einer Funktion $f: \mathbb{N} \to \mathbb{N}$ ist
$$o(f) = \{\, g: \mathbb{N} \to \mathbb{N} \mid \forall m \in \mathbb{N}, \exists n_0 \in \mathbb{N}: \forall n \geq n_0: m \cdot g(n) < f(n) \,\}$$
die Klasse der Funktionen, die asymptotisch **echt kleiner als** f sind.

Beispiel 13.2:
- $n^2 \in O(n^3)$ und $n^2 \in o(n^3)$, aber $n^2 \notin o(n^2)$
- $2^n \in o(3^n)$
- $\log n \in o(n)$, und sogar $\log^a(n) \in o\left(n^b\right)$ für bel. $a, b \in \mathbb{R}^+$

Platzkonstruierbarkeit

Definition 13.3
Eine Funktion $f: \mathbb{N} \to \mathbb{N}$ heißt **platzkonstruierbar**, wenn
- $\forall n \in \mathbb{N}\colon\ f(n) \geq \log n$
- Es eine TM M mit read-only Eingabe und write-only Ausgabe gibt, die auf Eingabe y der Länge $|y|$ eine Ausgabe der Länge $f(|y|)$ zurückgibt und Platzverbrauch (auf den Arbeitsbändern) $O(f)$ hat.

Beispiel 13.4:
Die folgenden Funktionen sind platzkonstruierbar

$$\log n\,, n, n^2, n^3, \dots, 2^n, 2^{n^2}, 2^{n^3}, \dots$$

Platzhierarchiesatz

Satz 13.5
Es sei $f: \mathbb{N} \to \mathbb{N}$ platzkonstruierbar. Dann gilt
$$DSPACE\big(o(f)\big) \subsetneq DSPACE\big(O(f)\big)$$

Beweisskizze:

- Gib eine Sprache $\mathcal{L}(M) \in DSPACE\big(O(f)\big) \setminus DSPACE\big(o(f)\big)$ an.
- Konstruiere dazu Maschine M mit Platzverbrauch $O(f)$.
- Zeige dann, $\mathcal{L}(M) \notin DSPACE\big(o(f)\big)$ mittels Diagonalisierung

Folgerungen

Korollar 13.6
Es gilt
a) $\mathrm{L} \subsetneq \mathrm{PSPACE}$
b) $\mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE}$
c) Für jedes $k \in \mathbb{N}$: $\mathrm{DSPACE}\left(O(n^k)\right) \subsetneq \mathrm{DSPACE}\left(O(n^{k+1})\right)$

Mit dem Satz von Savitch und dem det. Hierarchiesatz ergibt sich:

Korollar 13.7
Es sei $f: \mathbb{N} \to \mathbb{N}$ platzkonstruierbar. Dann gilt

$$NSPACE\big(o(f)\big) \subsetneq NSPACE\big(O(f \cdot f)\big)$$

Korollar 13.8
Es gilt $\mathrm{NL} \subsetneq \mathrm{NPSPACE} = \mathrm{PSPACE}$

Zeithierarchiesatz

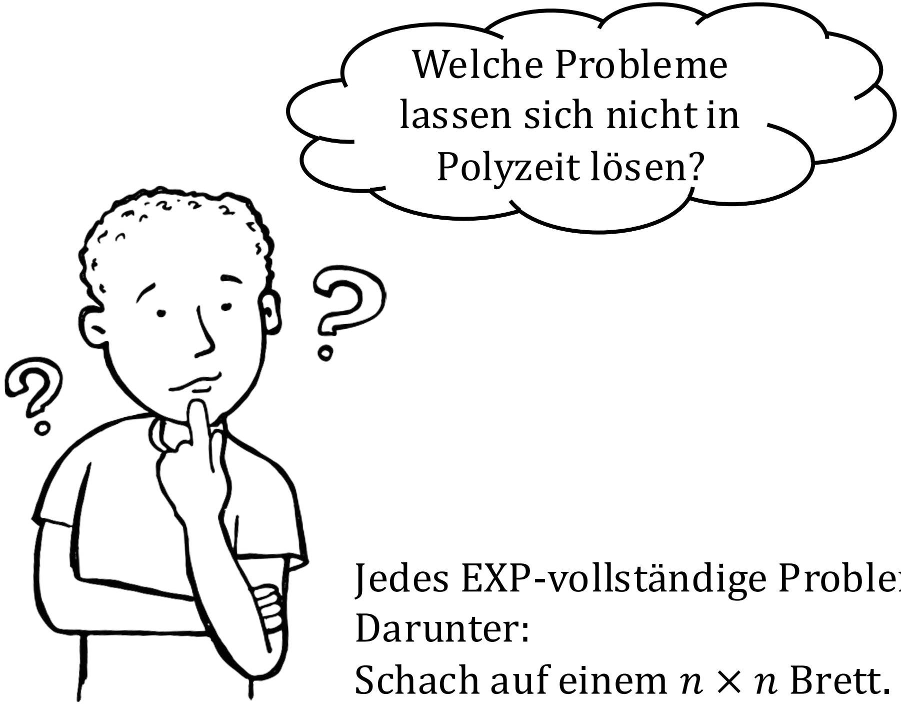
Satz 13.9
Es sei $f : \mathbb{N} \to \mathbb{N}$ zeitkonstruierbar und g eine Funktion mit $g(n) \log g(n) \in o(f)$. Dann gilt
$$DTIME(g) \subsetneq DTIME(f)$$

Beweis ähnlich wie für den Platzhierarchiesatz.

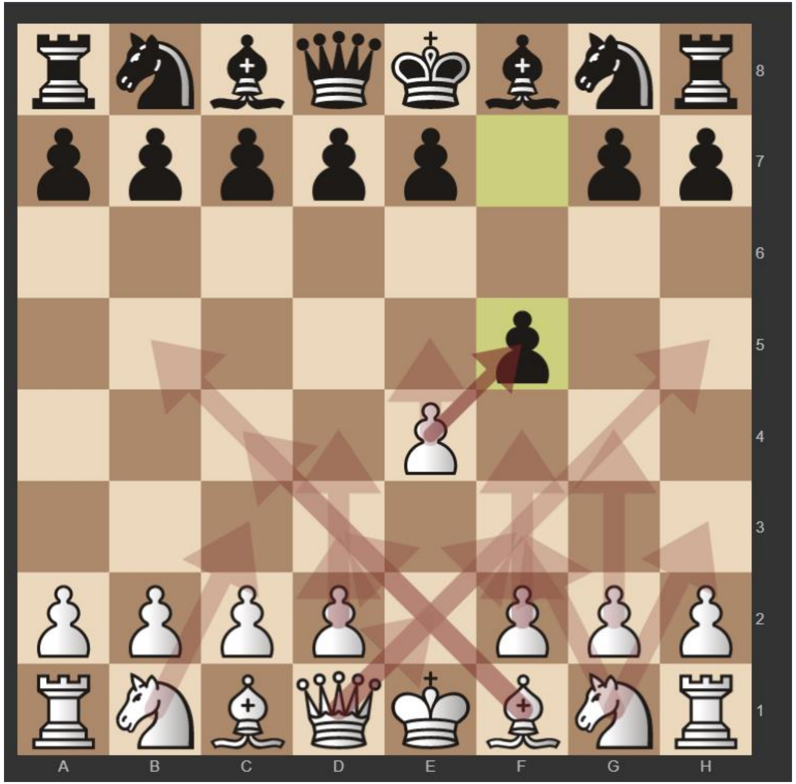
Korollar 13.10
Es gilt
a) $\mathrm{P} \subsetneq \mathrm{EXP}$
b) $\mathrm{NP} \subsetneq \mathrm{NEXP}$
c) Für jedes $k \in \mathbb{N}$: $\mathrm{DTIME}\left(O(n^k)\right) \subsetneq \mathrm{DTIME}\left(O(n^{k+1})\right)$

Damit gibt es bspw. keine Konstante $k \in \mathbb{N}$, sodass $\mathrm{P} = \mathrm{DTIME}\left(O(n^k)\right)$

Probleme in EXP

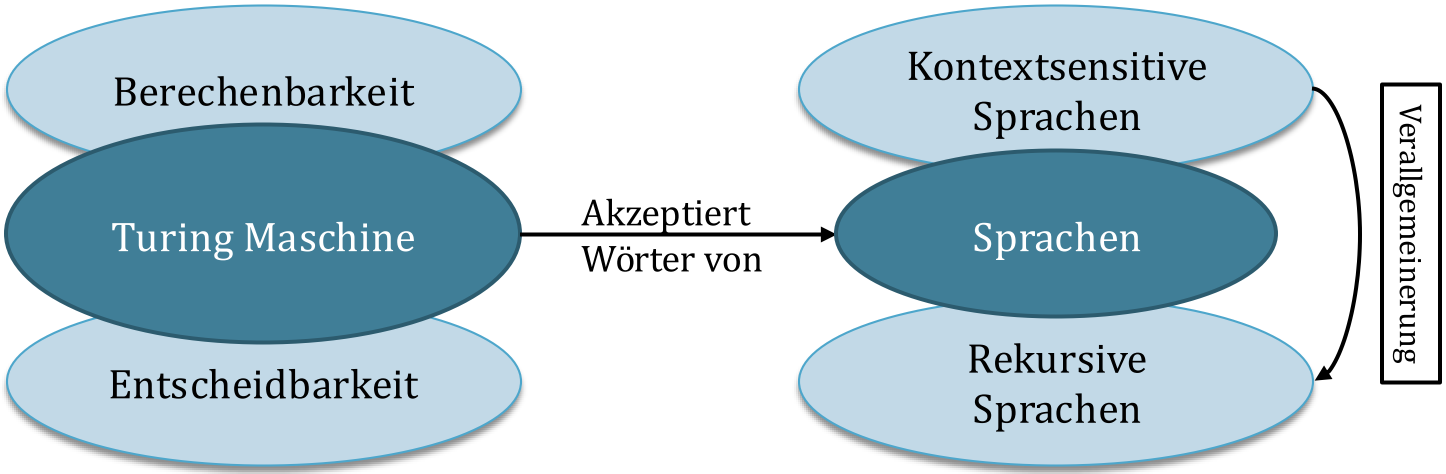


Jedes EXP-vollständige Problem!
Darunter:
Schach auf einem $n \times n$ Brett.



Das Ende!

Theoretische Informatik 2 – Part 1



TMs, Sprachen und Entscheidbarkeit

Typische Aufgaben:

- Konstruiere eine TM, die eine Sprache (semi-)entscheided
- Konstruiere eine TM, die eine (partielle) Funktion berechnet
- Analysiere eine gegebene TM
- Analysiere eine Grammatik
- Zeige (Über-)Abzählbarkeit
- Zeige (Un-)Berechenbarkeit / (Un-)Entscheidbarkeit
- Wende Satz von Rice an, betrachte nicht-monotone Eigenschaften

Kenne:

- Nicht-semi-entscheidbare Probleme
- Nicht-co-semi-entscheidare Probleme
- Nicht-co- und nicht-semi-entscheidbare Probleme

(Nicht-)Entscheidbare Probleme

Semi-entscheidbar, aber nicht co-semi-entscheidbar:

- **ACCEPT** (Akzeptiert eine gegebene TM M das Wort x?)
- **HP** (Hält eine gegebene TM M auf dem Wort x?)

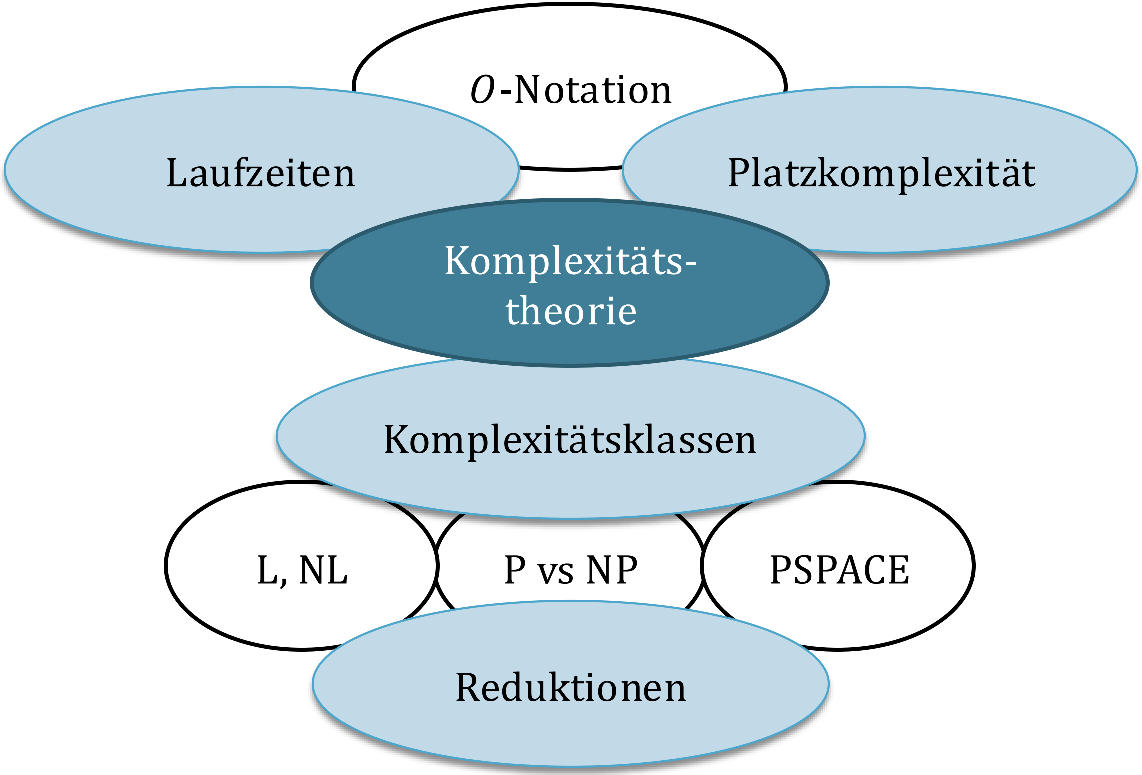
Co-Semi-Entscheibar, aber nicht semi-entscheidbar

- **Emptiness** (Hält eine gegebene TM M nie?)

Weder co-semi-entscheidbar, noch semi-entscheidbar:

- **Totality** (Hält eine gegebene TM M immer?)

Theoretische Informatik 2 – Part 2



Algorithmen und Reduktionen

Typische Aufgaben:
- Zeige Problem ist in der Klasse K
- Zeige Problem ist K-schwer
- Zeige Problem ist K-vollständig

Unterscheide:
- Logspace-Reduktion
- Polyzeit-Reduktion

Besitze:
- Für jede Klasse ein paar Probleme (vor allem für NL bis PSPACE)

Masterprobleme

Klasse	Masterproblem	Notiz
NL	Wegsuche in Graphen	Existiert in einem gegebenen Graphen ein Weg zwischen zwei Knoten?
P	Circuit Value Problem	Eingänge sind bekannt und müssen nur ausgewertet werden.
NP	Circuit-Sat / 3SAT	Eingänge sind variabel. Existiert eine erfüllende Belegung?
PSPACE	Quantified Sat	Variablen sind mit Existenz- und Allquantoren versehen.

Das Ende?

Was macht Probleme wirklich schwer?

Was ist der **Kern** einer schweren Instanz?

Unter welchen Betrachtungen lassen sich Probleme effizient lösen?

Neue Vorlesung im Winter:
Parametrisierte Algorithmen