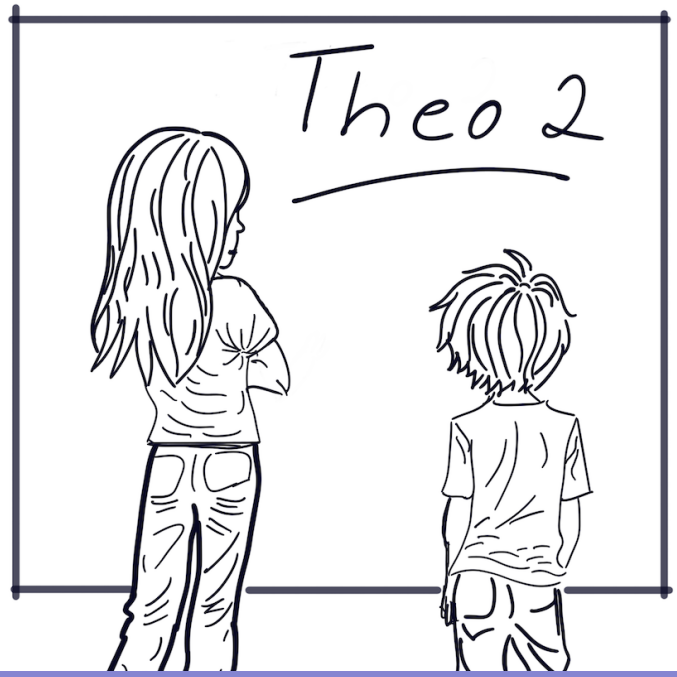
Theoretische Informatik 2

SoSe 2024

Berechenbarkeit und Komplexität

Prof. Dr. Sebastian Siebertz
AG Theoretische Informatik
MZH, Raum 3160
siebertz@uni-bremen.de

Universität Bremen



Platzbeschränkte Turingmaschinen

- Ressource Speicherplatz
- Unterscheiden zwischen Platz für die Eingabe und Arbeitsspeicher:
  - 2-Band TM
  - Eingabe (1. Band) darf nicht verändert werden
  - Es zählt nur der Speicherbedarf auf dem 2. Band

Definition (Platzkomplexitätsklassen)

Sei $S: \mathbb{N}_0 \to \mathbb{N}_0$ eine Funktion.

- $\mathrm{DSpace}(S(n)) :=$
  $\{L : \text{es ex. } S(n)\text{-platzbeschränkte DTM } M \text{ mit } L(M) = L\}$,
- $\mathrm{NSpace}(S(n)) :=$
  $\{L : \text{es ex. } S(n)\text{-platzbeschränkte NTM } M \text{ mit } L(M) = L\}$.

Platzkomplexitätsklassen

Satz

Sei $S: \mathbb{N}_0 \to \mathbb{N}_0$ eine Funktion. Es gilt

- $\mathrm{DTime}(S(n)) \subseteq \mathrm{DSpace}(S(n)) \subseteq \mathrm{NSpace}(S(n))$.
- $\mathrm{NSpace}(S(n)) \subseteq \mathrm{DTime}(2^{\mathcal{O}(S(n))})$, falls $S(n) \in \Omega(\log n)$.

Beweis (Skizze).

- $\mathrm{DTime}(S(n)) \subseteq \mathrm{DSpace}(S(n)) \subseteq \mathrm{NSpace}(S(n))$. ✓

Platzkomplexitätsklassen

- $\text{NSpace}(S(n)) \subseteq \text{DTime}(2^{\mathcal{O}(S(n))})$, falls $S(n) \in \Omega(\log n)$:
  - Sei $L \in \text{NSpace}(S(n))$ und sei M eine $S(n)$-platzbeschränkte NTM mit $L(M) = L$.
  - Berechnungsbaum von M auf Eingabe w:
    - kann unendlich tief sein (nicht jede Berechnung muss terminieren)
    - Konfigurationen können mehrfach auftauchen
  - Wenn Konfiguration α zwei mal auftaucht, so sind die Teilbäume unter diesen Vorkommen gleich.

  $\Rightarrow$ In einer Breitensuche nach akzeptierender Konfiguration muss der Teilbaum nicht mehrmals durchsucht werden.

Platzkomplexitätsklassen

- Deterministische Maschine M' führt modifizierte Breitensuche im Berechnungsbaum durch.
  - Speichere Liste aller Konfigurationen, die schon erreicht wurden.
  - Verfolge Berechnung nur, wenn es sich um neue Konfiguration handelt.
- Konfiguration: (q, k_1, k_2, v) mit $q \in Q$, $k_1 \leq n$, $k_2 \leq S(n)$, $v \in \Gamma^{S(n)}$ kann kodiert werden mit

$$\log |Q| \cdot \log n \cdot \log S(n) \cdot \log |\Gamma| \cdot S(n) \text{ Bits.}$$

- Anzahl der Konfigurationen mit beschränktem Speicher

$$|Q| \cdot n \cdot S(n) \cdot |\Gamma|^{S(n)} \in 2^{\mathcal{O}(S(n))},$$

da $|Q|, |\Gamma|$ feste Konstanten sind und $S(n) \in \Omega(\log n)$.

  - $x^y = 2^{\log x \cdot y}$.

Platzkomplexitätsklassen

- Für jeden Schritt der Breitensuche entsteht Overhead von $2^{\mathcal{O}(S(n)}$ zum Durchsuchen der Liste.
- Aber $2^{S(n)} \cdot 2^{S(n)} = (2^{S(n)})^2 = 2^{2 \cdot S(n)} \in 2^{\mathcal{O}(S(n))}$.
- Insgesamt kann DTM M' den modifizierten Berechnungsbaum in Zeit $2^{\mathcal{O}(S(n))}$ durchsuchen.

Platzkomplexitätsklassen

$$\text{LogSpace} := \bigcup_{S \in \mathcal{O}(\log n)} \text{DSpace}(S(n))$$

$$\text{NLogSpace} := \bigcup_{S \in \mathcal{O}(\log n)} \text{NSpace}(S(n))$$

$$\text{PSpace} := \bigcup_{p \text{ Polynom in } n} \text{DSpace}(p(n))$$

$$\text{NPSpace} := \bigcup_{p \text{ Polynom in } n} \text{NSpace}(p(n)).$$

Korollar

$\text{LogSpace} \subseteq \text{NLogSpace} \subseteq \text{P} \subseteq \text{NP} \subseteq \text{NPSpace} \subseteq \text{ExpTime}.$

Wo liegt PSpace?

Zeit- und platzkonstruierbare Funktionen

- Eine Funktion $T: \mathbb{N}_0 \to \mathbb{N}_0$ heißt zeitkonstruierbar, wenn eine $\mathcal{O}(T(n))$-zeitbeschränkte DTM ex., die bei jeder Eingabe der Länge n das Wort $0^{T(n)}$ auf das Arbeitsband schreibt und dann akzeptiert.
  - Genau $T(n)$ Positionen können auf dem Band von zeitbeschränkter DTM markiert werden.
- Eine Funktion $S: \mathbb{N}_0 \to \mathbb{N}_0$ heißt platzkonstruierbar, wenn eine $\mathcal{O}(S(n))$-platzbeschränkte DTM ex., die bei jeder Eingabe der Länge n das Wort $0^{S(n)}$ auf das Arbeitsband schreibt und dann akzeptiert.
  - Genau $S(n)$ Positionen können auf dem Band von platzbeschränkter DTM markiert werden.

Satz von Savitch

Satz von Savitch

Es sei $S(n) \in \Omega(\log n)$ eine platzkonstruierbare Funktion. Dann gilt

$$\text{NSpace}(S(n)) \subseteq \text{DSpace}(\mathcal{O}(S(n)^2)).$$

Korollar

Es gilt NPSpace = PSpace.

- Es folgt ***nicht*** LogSpace = NLogSpace.

Satz von Savitch

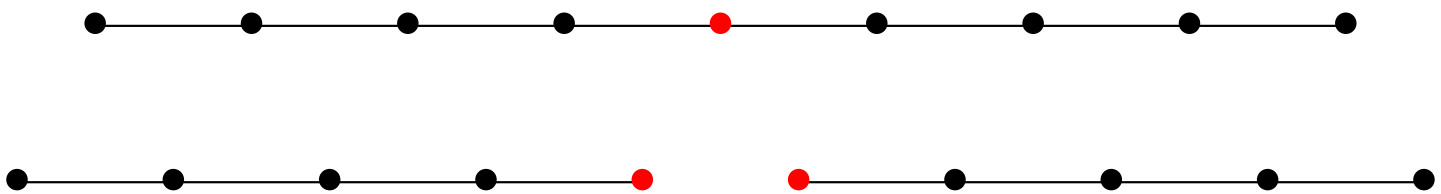
Beweis (Skizze).

- Akzeptierende Berechnung ohne Schleifen kann Länge höchstens

$$|Q| \cdot |w| \cdot S(n) \cdot |\Gamma|^{S(|w|)} \in 2^{\mathcal{O}(S(n))}$$

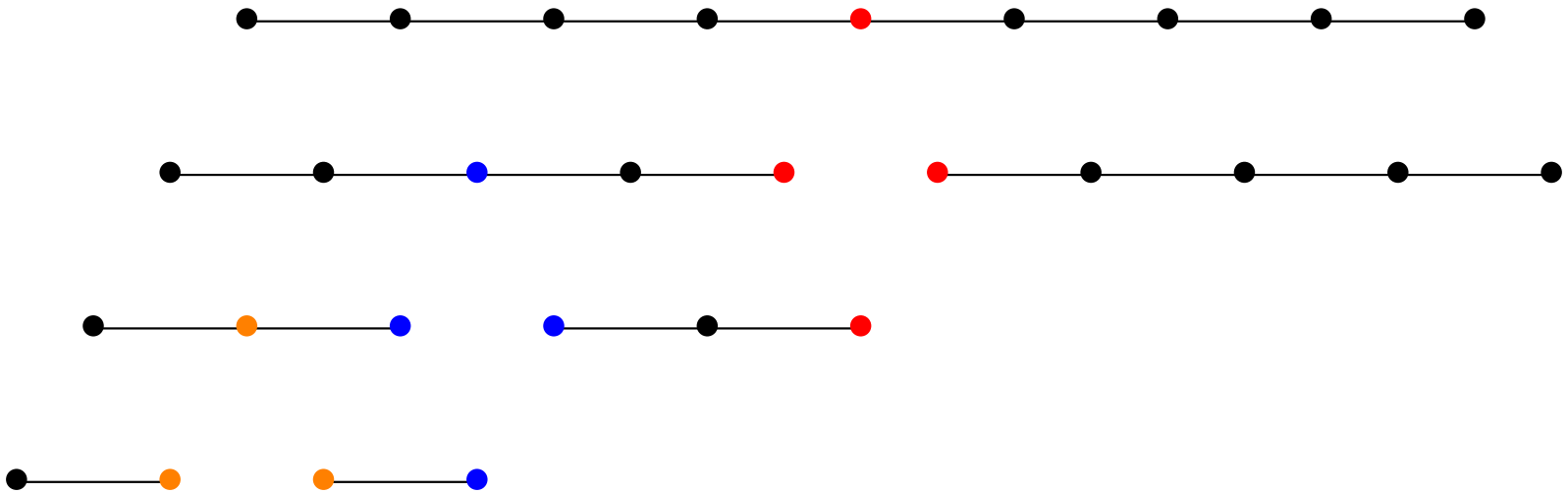
  haben.

- Zu entscheiden: Gibt es Konfigurationsfolge $\alpha_0 \vdash_M \ldots \vdash_M \alpha_\ell$ der Länge höchstens $\ell \in 2^{\mathcal{O}(S(n))}$?
  - α_0 Startkonfiguration, α_ℓ akzeptierende Konfiguration.
- Es existiert Folge $\alpha_0 \vdash_M \ldots \vdash_M \alpha_\ell$ genau dann, wenn Folgen $\alpha_0 \vdash_M \ldots \vdash_M \alpha_{\ell/2}$ und $\alpha_{\ell/2} \vdash_M \ldots \vdash_M \alpha_\ell$ existieren.



Satz von Savitch

- Teste rekursiv jede Möglichkeit für $\alpha_{\ell/2}$. (hier: Platzkonstruierbarkeit)



- Rekursionstiefe: $\log(2^{\mathcal{O}(S(n))}) = \mathcal{O}(S(n))$
- Auf jeder Stufe der Rekursion: speichere aktuell zu prüfende Konfigurationspaare: Speicher $\mathcal{O}(S(n))$
- Insgesamt: Speicher $\mathcal{O}(S(n)^2)$

Satz von Savitch

- Um zu zeigen, dass ein Problem in PSpace liegt:
  - zeige es liegt in NPSpace.

Satz

Das Wortproblem für monotone Grammatiken liegt in PSpace.

Beweis.

- Sprache L wird von monotoner Grammatik erzeugt $\Leftrightarrow L$ wird von nichtdeterministischen linear beschränkten Turingmaschine erkannt.

PSpace-Vollständigkeit

- L heißt PSpace-schwer, wenn für alle $L' \in \text{PSpace}$ gilt: $L' \leq_p L$.
- L heißt PSpace-vollständig, wenn $L \in \text{PSpace}$ und L PSpace-schwer ist.

Lemma

- *Ist* $L_2 \in \text{PSpace}$ *und gilt* $L_1 \leq_p L_2$, *so ist auch* L_1 *in* PSpace.
- *Ist* L_1 PSpace-*schwer und gilt* $L_1 \leq_p L_2$, *so ist auch* L_2 PSpace-*schwer.*

Satz

Das Wortproblem für monotone Grammatiken ist PSpace-vollständig.

Zeit- und Platzhierarchiesätze

- Es seien $T(n)$ und $S(n)$ zeit- bzw. platzkonstruierbare Funktionen. Dann gilt
  - $\mathrm{DTime}\big(T(n)\big) \subsetneq \mathrm{DTime}\big(T(n) \cdot \log^2(T(n))\big)$ und
  - $\mathrm{DSpace}\big(S(n)\big) \subsetneq \mathrm{DSpace}\big(S(n) \cdot \log(S(n))\big)$.

Korollar

Es gilt $\mathrm{P} \subsetneq \mathrm{ExpTime}$ und $\mathrm{LogSpace} \subsetneq \mathrm{PSpace}$.

Vermutung

Es gilt $\mathrm{LogSpace} \subsetneq \mathrm{NLogSpace} \subsetneq \mathrm{P} \subsetneq \mathrm{NP} \subsetneq \mathrm{PSpace} \subsetneq \mathrm{ExpTime}$.

Zusammenfassung Berechenbarkeit

- Berechenbarkeit: Was kann ein Computer berechnen?
- Seit den 1930er Jahren versuchen Mathematiker*innen und Informatiker*innen das Konzept der Berechenbarkeit zu formalisieren.
- Verschiedenste Berechnungsmodelle:
  - Turing Maschinen (Alan Turing),
  - μ-rekursive Funktionen (Kurt Gödel und Jaques Herbrand),
  - WHILE-Programme
  - Jede hinreichend starke Programmiersprache.
- Außerdem: der Zusammenhang zu Grammatiken

Church-Turing These

Die Klasse der Turing-berechenbaren Funktionen stimmt mit der Klasse der intuitiv berechenbaren Funktionen überein.

Zusammenfassung Berechenbarkeit

- Das spezielle Halteproblem ist das Problem:
  - **Gegeben:** Turingmaschine M.
  - **Frage:** Hält M auf ihrer eigenen Kodierung $\langle M \rangle$?
- Das spezielle Halteproblem ist unentscheidbar.
  - Beweis mit Diagonalisierung.
- Das Halteproblem ist das Problem:
  - **Gegeben:** Turingmaschine M und Wort w.
  - **Frage:** Hält M auf w?
- Das Halteproblem ist unentscheidbar.
  - Reduktion des speziellen Halteproblems auf das Halteproblem.

Zusammenfassung Berechenbarkeit

- Sei $L_1 \subseteq \Sigma^*$ und $L_2 \subseteq \Gamma^*$.
- Eine Reduktion von L_1 auf L_2 ist eine (totale) berechenbare Funktion

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

so dass für alle $w \in \Sigma^*$:

$$w \in L_1 \Leftrightarrow f(w) \in L_2.$$

- L_1 ist auf L_2 reduzierbar, $L_1 \leq L_2$, falls es eine Reduktion von L_1 nach L_2 gibt.

Lemma

Wenn $L_1 \leq L_2$, dann

- *L_2 (semi) entscheidbar $\Rightarrow$ L_1 (semi) entscheidbar.*
- *L_1 nicht (semi) entscheidbar $\Rightarrow$ L_2 nicht (semi) entscheidbar.*

Zusammenfassung Berechenbarkeit

Satz von Rice

Jede nichttriviale semantische Eigenschaft von Turingmaschinen ist unentscheidbar.

Satz von Rice, zweiter Teil

Jede nicht nach oben abgeschlossene Eigenschaft von Turingmaschinen ist nicht semi-entscheidbar.

Zusammenfassung Komplexität

- Komplexität: was ist effizient berechenbar?
  - Nicht beliebig viel Zeit und beliebig viel Speicher verfügbar!
- Probleme in P: theoretisch effizient lösbar.

$$\text{LogSpace} \subseteq \text{NLogSpace} \subseteq \text{P} \subseteq \text{NP} \subseteq \text{PSpace} = \text{NPSpace} \subseteq \text{ExpTime}.$$

- Die Frage, ob diese Teilmengenbeziehungen echt sind, insbesondere ob

$$\text{P} = \text{NP} \text{ oder } \text{P} \neq \text{NP}?$$

gilt als eine der größten offenen Fragen der Informatik.

Zusammenfassung Komplexität

- Eine Reduktion f von $L_1 \subseteq \Sigma^*$ auf $L_2 \subseteq \Gamma^*$ heißt Polynomialzeitreduktion, wenn es ein Polynom p und eine $p(n)$-zeitbeschränkte DTM gibt, die f berechnet.
- Wenn eine Polynomialzeitreduktion von L_1 auf L_2 existiert, dann schreiben wir $L_1 \leq_p L_2$.
- Eine Sprache L heißt NP-schwer, wenn für alle $L' \in \mathrm{NP}$ gilt: $L' \leq_p L$.
- L heißt NP-vollständig, wenn $L \in \mathrm{NP}$ und L NP-schwer ist.

Satz

Für jede NP-vollständige Sprache L gilt: wenn $L \in \mathrm{P}$, dann $\mathrm{P} = \mathrm{NP}$.

Zusammenfassung Komplexität

Satz von Cook und Levin

SAT ist NP-vollständig.

Lemma

Ist L_1 NP-schwer und gilt $L_1 \leq_p L_2$, so ist auch L_2 NP-schwer.

Zusammenfassung Komplexität

- Wenn $P \neq NP$, so kann SAT (und alle NP-schweren Probleme) nicht in Polynomialzeit gelöst werden.

Exponential Time Hypothesis (ETH)

3-SAT mit N Variablen und M Klauseln kann nicht in Zeit $2^{o(N+M)} \cdot (N+M)^{\mathcal{O}(1)}$ gelöst werden.

Strong Exponential Time Hypothesis (SETH), vereinfachte Form

KNF-SAT mit N Variablen und M Klauseln kann nicht in Zeit $(2-\varepsilon)^N \cdot (N+M)^{\mathcal{O}(1)}$ für beliebiges $\varepsilon > 0$ gelöst werden.

$$\text{SETH} \quad \Rightarrow \quad \text{ETH} \quad \Rightarrow \quad \text{P} \neq \text{NP}.$$

Berechenbarkeit und Komplexität – Motivation