

FGI-1 Repetitorium: Automatentheorie [Tag 3]

Jan Henrik Röwekamp
(8roeweka)

17.07., 18.07. & 19.07.2013

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- NP-Vollständigkeit

Abschlusseigenschaften

- Reguläre Mengen sind abgeschlossen gegenüber:
 - Vereinigung
 - Komplement
 - Produkt
 - Schnittbildung
 - Sternbildung
 - (Homomorphismen)

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- NP-Vollständigkeit

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

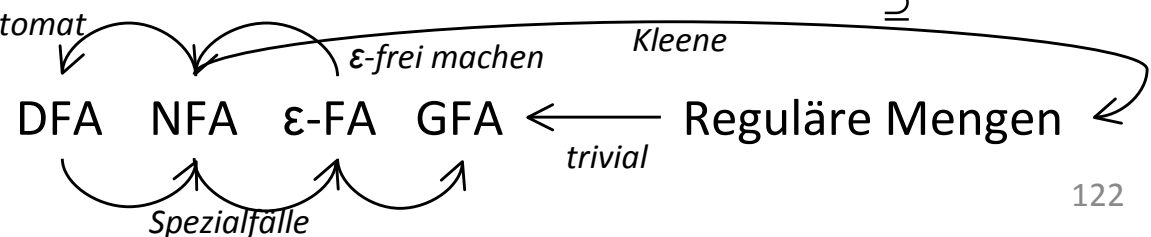
Deterministisch
Kontextfreie Mengen

uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik



Pumping Lemma

- Motivation: <Tafel>
- 2 Pumping Lemmas: uvw- und uvwxy-Theorem
- Beide sind sehr ähnlich.
- **Achtung: Obwohl das uvw-Theorem etwas über reguläre Mengen aussagt, kann damit nicht nachgewiesen werden, dass eine Menge regulär ist !!! Sondern nur, dass sie es nicht ist.**
- **Das Analoge gilt für das uvwxy-Theorem und kontextfreie Mengen!**

Pumping Lemma: uvw

- uvw-Theorem formal (Sei $L \subseteq \Sigma^*$) :
 $\forall L \in Reg \quad \exists n \in \mathbb{N} \quad \forall z \in L : \quad |z| \geq n \quad \Rightarrow$
 $\exists u, v, w \in \Sigma^*, z = uvw:$
 $|uv| \leq n$
 $|v| \geq 1$
 $\{u\}\{v\}^*\{w\} \subseteq L$
- Aufgabe: Verwenden Sie das uvw-Theorem um nachzuweisen, dass $a^n b^n$ nicht regulär ist.

Pumping Lemma: uvw: Lösung

- Sei $k \in \mathbb{N}$ die Zahl des Pumping Lemmas.
- Sei z in Abhängigkeit von k das Wort $a^k b^k \in L$
- Offensichtlich gilt $|z| \geq k$
- Alle möglichen Zerlegung des Wortes in Teile u , v und w (welche die Bedingungen nicht verletzen) bewirken, dass v aus a 's besteht.
- Da offensichtlich $uv^2w \notin L$, folgt $L \notin \text{Reg}$

Pumping Lemma: uvwxy

- uvwxy-Theorem formal (Sei $L \subseteq \Sigma^*$) :
 $\forall L \in Cf \quad \exists n \in \mathbb{N} \quad \forall z \in L : |z| \geq n \Rightarrow$
 $\exists u, v, w, x, y \in \Sigma^*, z = uvwxy:$
 $|vwx| \leq n$
 $|vx| \geq 1$
 $\{u\}\{v\}^*\{w\}\{x\}^*\{y\} \subseteq L$

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- **Grammatiken (CFG; Typ 2 und Typ 3)**
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

$\supseteq$

DPDA

Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinare
Grammatik

Potenzautomat

DFA

NFA

ϵ -FA

GFA

← trivial

Reguläre Mengen

Spezialfälle

ϵ -frei machen

Kleene

Kontextfreie Grammatiken

- Formal Quadrupel (4-Tupel): $G = (\Sigma, N, P, S)$
- Umgekehrter Ansatz zu Automaten:
Wort wird generiert, statt eingelesen.
- Ableitungen und Ableitungsbäume
- Praxisanwendung: Bspw. Compilerbau
- Typ 2 und Typ 3 Grammatiken
 - Rechts- und linkslineare Grammatik

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- **Grammatik-Normalformen**
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Normalformen

- 2 Normalformen betrachtet:
Chomsky und (Greibach)
- Jede Grammatik kann in beide Normalformen überführt werden.
- Chomsky-Normalform (CNF) wird in 6 Schritten erzeugt.

Chomsky-Normalform erzeugen

- 1. ε -frei machen
- 2. Reduzieren
- 3. Kettenregeln entfernen
- 4. Lange Terminalregeln ersetzen
- 5. Verkürzen zu langer Regeln
- 6. Wiederherstellen der Sprache (ε)

- Bringen Sie die folgende Grammatik in CNF:

$$\Sigma = \{a, b, d\}$$

$$S \rightarrow AB \mid aA$$

$$A \rightarrow B \mid A \mid bb$$

$$B \rightarrow DC \mid d \mid AB \mid \varepsilon$$

$$C \rightarrow AC \mid CB \mid C$$

$$D \rightarrow d \mid \varepsilon \mid S$$

ε -frei machen

- $M_0 := \{A' \in N \mid A' \rightarrow \varepsilon \in P\}$
 $M_{i+1} := M_i \cup \{A' \in N \mid \exists w \in M_i^* : A' \rightarrow w \in P\}$

ε -frei machen (Lösung)

- $M_0 = \{B, D\}$
- $M_1 = \{B, D\} \cup \{B, D, A\} = \{B, D, A\}$
- $M_2 = \{S, B, D, A\}$
- $M_3 = \{S, B, D, A\} = M_2 = N_\varepsilon$

$$S \rightarrow AB \mid aA$$

$$A \rightarrow B \mid A \mid bb$$

$$B \rightarrow DC \mid d \mid AB \mid \varepsilon$$

$$C \rightarrow AC \mid CB \mid C$$

$$D \rightarrow d \mid \varepsilon \mid S$$

$$S \rightarrow AB \mid aA \mid A \mid B \mid \varepsilon \mid a$$

$$A \rightarrow B \mid A \mid bb \mid \varepsilon$$

$$B \rightarrow DC \mid d \mid AB \mid \varepsilon \mid C \mid A \mid B$$

$$C \rightarrow AC \mid CB \mid C$$

$$D \rightarrow d \mid \varepsilon \mid S$$

$$S \in N_\varepsilon !$$

Das Startsymbol S lässt sich direkt oder indirekt auf ε ableiten, die Sprache beinhaltet also das leere Wort! Dieses muss in Schritt 6 wieder hinzugefügt werden!

Produktive Nonterminale

- $M_0 := \Sigma$
 $M_{i+1} := M_i \cup \{A' \in N \mid \exists w \in M_i^* : A' \rightarrow w \in P\}$

Produktive Nonterminale (Lösung)

- $M_0 = \{a, b, d\}$
- $M_1 = \{a, b, d, S, A, B, D\}$
- $M_2 = \{a, b, d, S, A, B, D\} = M_1$

$S \rightarrow AB \mid aA \mid A \mid B \mid a$

$A \rightarrow B \mid A \mid bb$

$B \rightarrow DC \mid d \mid AB \mid C \mid A \mid B$

$C \rightarrow AC \mid CB \mid C$

$D \rightarrow d \mid S$

$S \rightarrow AB \mid aA \mid A \mid B \mid a$

$A \rightarrow B \mid A \mid bb$

$B \rightarrow \cancel{DC} \mid d \mid AB \mid \cancel{C} \mid A \mid B$

~~$C \rightarrow AC \mid CB \mid C$~~

$D \rightarrow d \mid S$

Erreichbare Nonterminale

- $M_0 := \{S\}$
 $M_{i+1} := M_i \cup$
 $\{B' \in N \mid \exists A' \in M_i \ \exists u, v \in (N \cup \Sigma)^* :$
 $A' \rightarrow uBv \in P\}$

Erreichbare Nonterminale (Lösung)

- $M_0 = \{S\}$
- $M_1 = \{S, A, B\}$
- $M_2 = \{S, A, B\} = M_1$

$S \rightarrow AB \mid aA \mid A \mid B \mid a$

$A \rightarrow B \mid A \mid bb$

$B \rightarrow d \mid AB \mid A \mid B$

$D \rightarrow d \mid S$

$S \rightarrow AB \mid aA \mid A \mid B \mid a$

$A \rightarrow B \mid A \mid bb$

$B \rightarrow d \mid AB \mid A \mid B$

~~$D \rightarrow d \mid S$~~

Schritte 3-6 Lösung

$S \rightarrow AB \mid aA \mid A \mid B \mid a$

$A \rightarrow B \mid A \mid bb$

$B \rightarrow d \mid AB \mid A \mid B$

$S \rightarrow AB \mid aA \mid d \mid bb \mid a$

$A \rightarrow d \mid AB \mid bb$

$B \rightarrow d \mid AB \mid bb$

$S \rightarrow AB \mid \langle a \rangle A \mid d \mid \langle b \rangle \langle b \rangle \mid a$

$A \rightarrow d \mid AB \mid \langle b \rangle \langle b \rangle$

$B \rightarrow d \mid AB \mid \langle b \rangle \langle b \rangle$

$\langle a \rangle \rightarrow a$

$\langle b \rangle \rightarrow b$

$S \rightarrow AB \mid \langle a \rangle A \mid d \mid \langle b \rangle \langle b \rangle \mid a \mid \epsilon$

$A \rightarrow d \mid AB \mid \langle b \rangle \langle b \rangle$

$B \rightarrow d \mid AB \mid \langle b \rangle \langle b \rangle$

$\langle a \rangle \rightarrow a$

$\langle b \rangle \rightarrow b$

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- **Kellerautomaten**
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

Typ 2

Kontextfreie
Grammatik

PDA

Kontextfreie Mengen

$\supseteq$

DPDA

Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik

DFA NFA ϵ -FA GFA

Reguläre Mengen

Potenzautomat

ϵ -frei machen

Kleene

trivial

Spezialfälle

Kellerautomaten (PDAs)

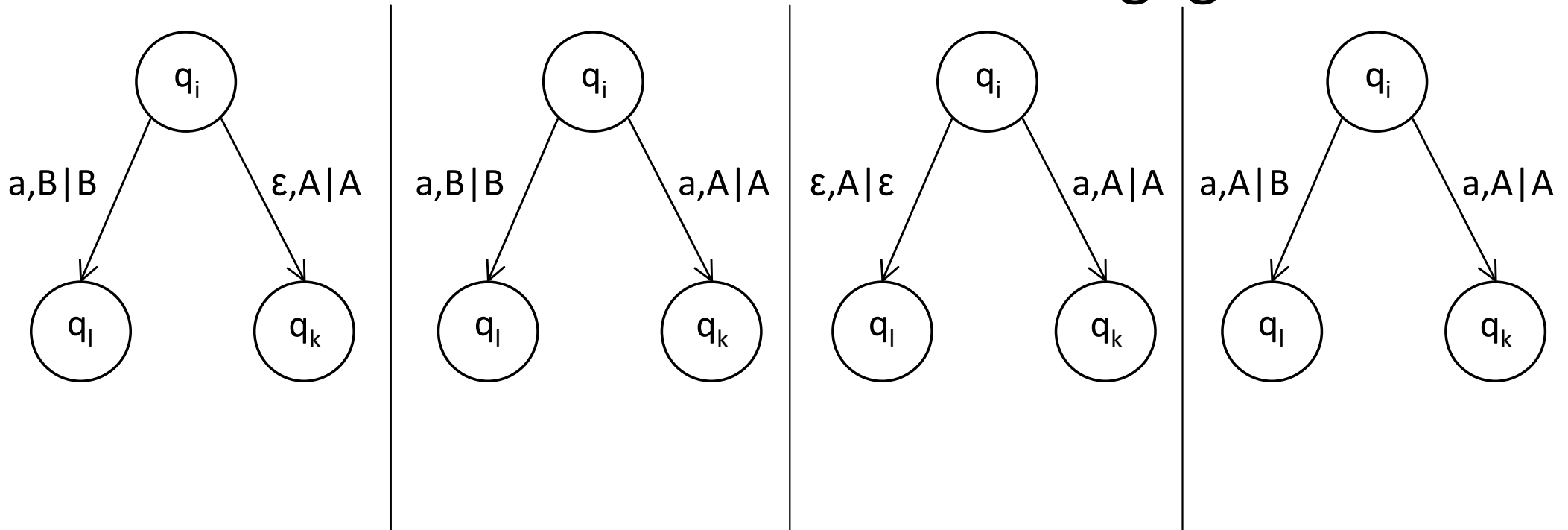
- Formalismus analog zu endlichen Automaten.
- Aber:
 - Zusätzlicher Stack-Speicher
 - Nichtdeterminismus nicht mehr so offensichtlich
 - DPDAs und NPDAs akzeptieren nicht die gleiche Sprachfamilie!
 - 2 Möglichkeiten der Akzeptanz:
Leerer Keller und Endzustand

PDA: Akzeptanz & Konfiguration

- Akzeptanz mit leerem Keller ($L_\varepsilon(A)$):
 - Wort wurde komplett gelesen, und der Keller ist komplett leer (auch kein Kellerbodensymbol!)
- Akzeptanz mit Endzustand ($L(A)$):
 - Wort wurde komplett gelesen, und befindet sich im Endzustand.
- Konfiguration eines PDA:
 - Element aus $Q \times \Sigma^* \times \Gamma^*$ also z.B.: $(q_0, abb, A\perp)$

PDA: Nichtdeterminismus

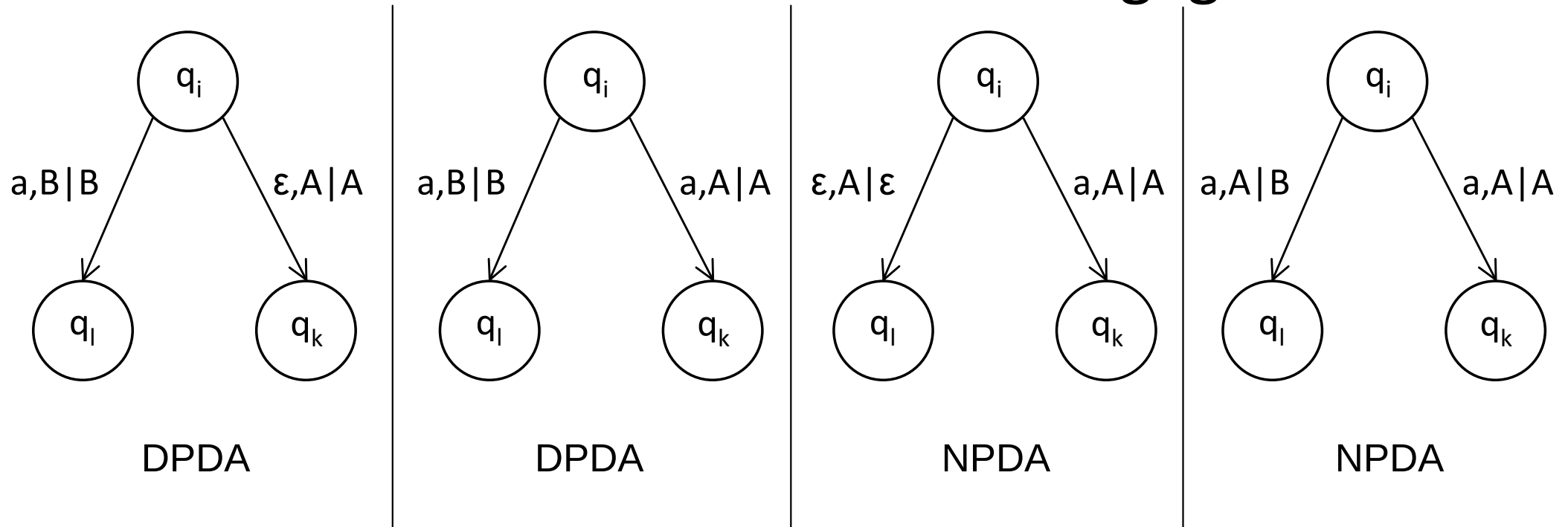
- Die folgenden Ausschnitte seien aus einem ansonsten deterministischen PDA gegeben:



- Welche Automaten sind DPDA's? Welche nicht?

PDA: Nichtdet.: Lösung

- Die folgenden Ausschnitte seien aus einem ansonsten deterministischen PDA gegeben:



- Welche Automaten sind DPDA's? Welche nicht?

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- **Andere Grammatiken (Typ 0 und Typ 1)**
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

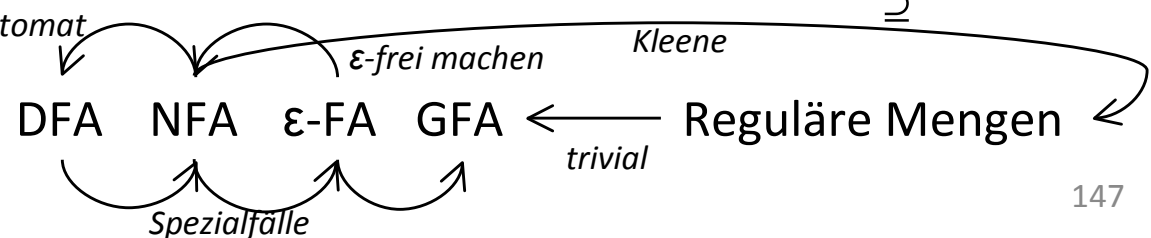
Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik



Typ 0 und Typ 1 Grammatiken

- Typ 1 Grammatiken werden auch „kontextsensitive Grammatiken“ oder „monotone Grammatiken“ genannt.
- Existenz von Grammatiken, die Typ 2 (CFG), aber nicht Typ 1 sind.
- Typ 0 Grammatiken unterliegen nicht mehr der Monotonie-Einschränkung.
- Typ 0 Grammatiken sind sehr mächtig.

PAUSE
(30 Minuten)

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- **Turing-Maschinen**
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Die Chomsky-Hierarchie

Überabzählbare Mengen

$\neq$

Abzählbare Mengen

$\supseteq$

Typ 0

Typ 0 -
Grammatik

DTM NTM

Aufzählbare Mengen

$\supseteq$

DTM NTM

Entscheidbare Mengen

$\supseteq$

Typ 1

Kontextsensitive
Grammatik

LBA

Kontextsensitive Mengen

– uvwxy-Theorem

$\supseteq$

PDA

Kontextfreie Mengen

Typ 2

Kontextfreie
Grammatik

DPDA

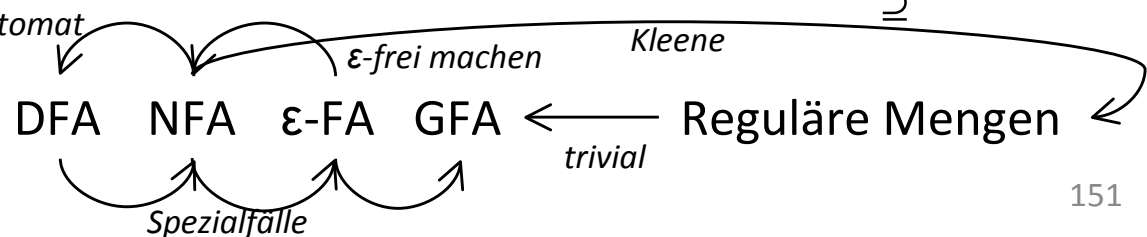
Deterministisch
Kontextfreie Mengen

– uvw-Theorem

$\supseteq$

Typ 3

Rechts- bzw.
linkslinere
Grammatik



Turing-Maschinen

- DTM formal 7-Tupel: $T = (Q, \Sigma, \Gamma, \delta, q_0, \#, F)$
 - Q ist eine Menge an Zuständen
 - Σ ist eine Eingabealphabet
 - Γ ein Bandalphabet
 - $\delta : (Q \times \Gamma) \rightarrow (Q \times \Gamma \times \{l, r, -\})$ eine Überföhrungsfkt.
 - $q_0 \in Q$ ist ein Startzustand
 - $\# \in \Gamma$ ist das Bandendsymbol
 - $F \subseteq Q$ ist eine Menge an Endzuständen

Turing-Maschinen

- Variabel beweglicher Lese-Schreibkopf
- NTM und DTM akzeptieren gleiche Sprachfamilie (Typ 0)
- Turing-Mächtigkeit & Satz von Turing
- Rechnungen und Erfolgsrechnungen
- Abbruch beim Erreichen des ersten Endzustands
- Nicht alles muss gelesen werden!
- Mehrere Bänder möglich (k-Band-TM)

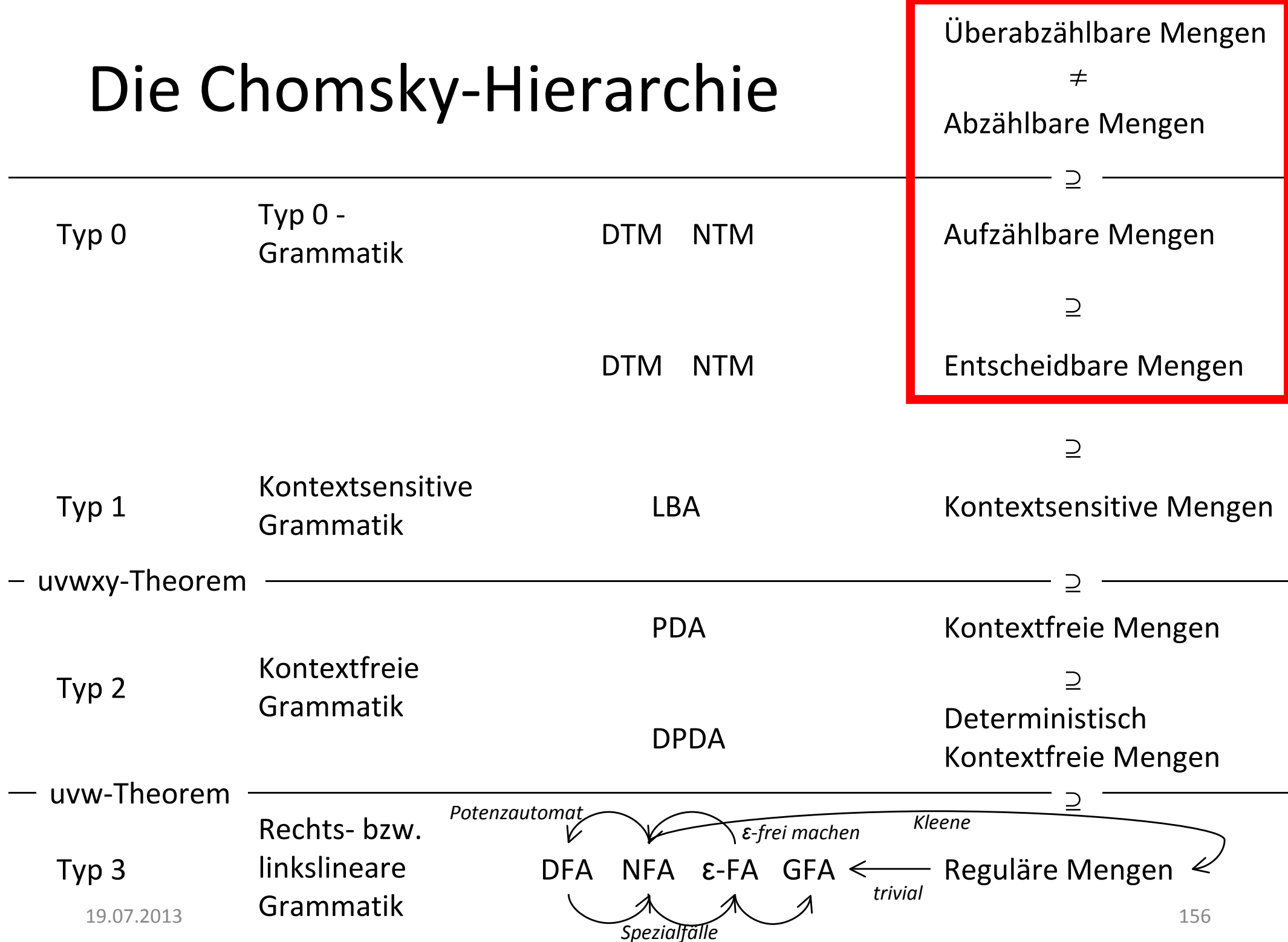
Konfiguration & Rechnung

- Eine Konfiguration einer TM ist ein Element aus:
 $\Gamma^* \cdot Q \cdot \Gamma^+$ also z.B. $\#q_0abb\#$
- Q gibt den Zustand an, in dem sich die TM befindet.
- Γ^* und Γ^+ den Inhalt des Arbeitsbandes links vom Schreibkopf (Γ^*) und auf und rechts vom SK (Γ^+).
- Eine **Rechnung** ist eine Folge von Konfigurationen
- Endet die Folge in einem Endzustand wird von einer **Erfolgsrechnung** gesprochen.

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- **Entscheidbarkeit & Berechenbarkeitstheorie**
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

Die Chomsky-Hierarchie



Berechenbarkeitstheorie: Begriffe

- **Abzählbare Menge:**

Es existiert eine Bijektion aus den natürlichen Zahlen auf die Menge.

- **Aufzählbare Menge:**

Die Menge ist abzählbar und Elemente der Menge sind in endlicher Zeit durch eine TM als solche identifizierbar.

- **Entscheidbare Menge:**

Die Menge und ihr Komplement sind aufzählbar

Berechenbarkeitstheorie: Begriffe

- **Überabzählbare Menge:**
 - Es existiert keine Bijektion aus den natürlichen Zahlen auf die Menge BZW.
 - Jede Liste bzw. Struktur einer Liste von Elementen der Menge ist unvollständig BZW.
 - Die Menge kann nicht geordnet werden (In dem Sinne, dass es ein benennbares „erstes“, „zweites“, „drittes“, usw. Element gäbe).
- Überabzählbare Mengen existieren: z.B. $\mathbb{R}$
- Beweis: Diagonalisierung

Das Halteproblem H

- Bezeichnet das Problem, ob ein Programm bei gegebener Eingabe nach endlich vielen Schritten hält, oder nicht.
- Das Halteproblem ist aufzählbar (Das gegebene Programm kann, wenn es hält, einfach ausgeführt werden bis es hält.
→ „Ja“-Instanzen können erkannt werden.)
- Das Halteproblem ist jedoch nicht entscheidbar!
- Sehr wichtig in der Informatik

Charakteristische Funktion

- Die charakteristische Funktion
 $\chi_L : \Sigma^* \rightarrow \{0, 1\}$
der Menge L gibt an, ob ein Element in der Menge liegt (1), oder aber nicht (0).
- Ist die charakteristische Funktion berechnbar, so ist die Menge entscheidbar. (und umgekehrt!)
- Bindeglied zwischen Berechenbarkeit (von Fkt.) und Entscheidbarkeit (von Mengen)

Beispiele zu den Mengen

- Überabzählbar: $\mathbb{R}, 2^{\mathbb{N}}$
- Abzählbar, aber nicht aufzählbar: L_d
- Aufzählbar (Typ 0), aber nicht entscheidbar: H
- Entscheidbar, aber nicht Typ 1: L_e
- Typ 1, aber nicht Typ 2: $a^n b^n c^n$
- N.det. Typ 2, aber nicht det. Typ 2: ww^{rev}
- Det. Typ 2, aber nicht Typ 3: $a^n b^n$
- Typ 3, aber nicht endlich: a^*

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- **Komplexitätstheorie & O-Notation**
- Komplexitätsklassen
- **NP-Vollständigkeit**

Komplexitätstheorie:

Jenseits der Chomsky-Hierarchie

- Chomsky-Hierarchie legt keinen Wert auf die **Geschwindigkeit** der Berechnung.
- In der Praxis: Geschwindigkeit hochgradig relevant
- → (strukturelle) Komplexitätstheorie
- Wichtiges Hilfsmittel der Komplexitätstheorie: O-Notation

O-Notation

- Etwaige asymptotische Abschätzung der Laufzeit in Abhängigkeit von der Eingabelänge
- Konstante Faktoren werden ignoriert
- Relevant ist nur der „schwerwiegendste“ Anteil der Laufzeitfunktion
- $O(f(n))$ bedeutet frei: Maximal so schlimm, wie $f(n)$ (ohne konstanten Faktor)
- Z.B.: $3n^3 + 5n^2 + 3n + 7$ liegt in $O(n^3)$. Aber auch in $O(n^5)$ und $O(2^n)$. Aber nicht in $O(n^2)$.

O-Notation

- Zusätzliche Mengen:

- $o(f(n))$ „ $<$ “

- $\Omega(f(n))$ „ $\geq$ “

- $\omega(f(n))$ „ $>$ “

- $\Theta(f(n))$ „ $=$ “

- Mathematischer Nachweis:

$g(n) \in O(f(n))$ gdw. $\lim_{n \rightarrow \infty} g(n)/f(n) < \infty$

$g(n) \in o(f(n))$ gdw. $\lim_{n \rightarrow \infty} g(n)/f(n) = 0$

$g(n) \in \Omega(f(n))$ gdw. $\lim_{n \rightarrow \infty} f(n)/g(n) < \infty$

Zeit- und Platzmaße

- Zeitmessung in Anzahl der Zustandsübergänge in Abhängigkeit von der Eingabelänge n :
 - Obergrenze $t(n)$
- Platzmessung in Anzahl der verwendeten Bandzellen in Abhängigkeit von der Eingabelänge n :
 - Obergrenze $s(n)$

Einschub: LBA

- LBA = „Linear beschränkter Automat“
- TM mit der Einschränkung:
 - $s(n) \in O(n)$
- Äquivalent zu Typ 1 Sprachen, bzw. kontextsensitiven Mengen C_s .

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- **Komplexitätsklassen**
- **NP-Vollständigkeit**

Komplexitätsklassen formal

- $DTime(t(n))$: Alle Sprachen, die eine DTM in maximal $t(n)$ Zeitschritten entscheidet
- $NTime(t(n))$: -- “ -- NTM
- $DSpace(s(n))$: Alle Sprachen, die eine DTM in maximal $s(n)$ Platzschritten entscheidet
- $NSpace(s(n))$: -- “ -- NTM

-
- $\mathbf{P} = \cup_{i \geq 1} DTime(n^i)$ Gilt
 - $\mathbf{NP} = \cup_{i \geq 1} NTime(n^i)$ $\cup_{i \geq 1} DTime(n^i)$
 - $\mathbf{PSpace} = \cup_{i \geq 1} DSpace(n^i)$ =
 - $\mathbf{NPSpace} = \cup_{i \geq 1} NSpace(n^i)$ $\cup_{i \geq 1} DTime((n^i)^2) ?$

Lösung zur Frage

- Ja, die Aussage ist korrekt.
- DTime gibt Auskunft über **obere Schranken** von Anzahlen von Berechnungsschritten.
- Quadriert man ein Polynom, ist es immernoch ein Polynom!
- Die Mengen sind exakt die gleichen.
- Bspw. Liegt n in $O(n)$,
aber auch in $O(n^2)$.
- Man verliert also keine Informationen durch die Quadratbildung.

$$\begin{array}{l} \text{Gilt} \\ \cup_{i \geq 1} \text{DTime}(n^i) \\ = \\ \cup_{i \geq 1} \text{DTime}((n^i)^2) \quad ? \end{array}$$

Komplexitätsklassen informell

- **P:** Menge aller Sprachen, die von einer deterministischen TM mit polynomiellem Zeitbedarf entschieden werden.
- **NP:** Menge aller Sprachen, die von einer nichtdeterministischen TM mit polynomiellem Zeitbedarf entschieden werden.
- **PSpace:** Menge aller Sprachen, die von einer deterministischen TM mit polynomiellem Platzbedarf entschieden werden.
- **NPSpace:** Menge aller Sprachen, die von einer nichtdeterministischen TM mit polynomiellem Platzbedarf entschieden werden.

Komplexitätsklassen

- $P \subseteq NP$ und $PSPACE \subseteq NPSPACE$:
 - Gilt, da jede DTM eine spezielle NTM ist.
- Satz von Savitch besagt: $NPSPACE \subseteq PSPACE$
 - Beweis zu umfangreich fürs Repetitorium.
- $NP \subseteq PSPACE$ gilt.
- Insgesamt: **$P \subseteq NP \subseteq PSPACE = NPSPACE$**
- $NP \subseteq PSPACE$ Begründung:
 - Sei $L \in NP$.
 - Folglich gibt es eine NTM, die L in poly. Zeit entscheidet.

$NP \subseteq PSpace$ (Fortsetzung)

- Damit die NTM in poly. Zeit laufen kann, muss die theoretisch längste (zeitintensivste) Rechnung auch noch polynomiell sein.
- In t Zeitschritten können **nur maximal t Bandzellen** verwendet werden (*Eine neue Zelle in jedem Schritt*).
- Wird die **NTM durch eine DTM simuliert**, so muss jede mögliche Rechnung durchlaufen werden, bis das Wort akzeptiert wird, oder aber alle möglichen Rechnungen betrachtet wurden.
- Das Ergebnis der letzten Rechnung spielt keine Rolle für die nächste Rechnung, der Platz kann also (gelöscht und) **wiederverwendet werden**.
- Im Worst Case braucht die DTM also t Bandzellen um L zu entscheiden. t ist polynomiell.
- Es folgt direkt $L \in PSpace \quad \rightarrow NP \subseteq PSpace$

Ablauf – Tag 3

- Pumping Lemma (uvw und uvwxy-Theorem)
- Grammatiken (CFG; Typ 2 und Typ 3)
- Grammatik-Normalformen
- Kellerautomaten
- Andere Grammatiken (Typ 0 und Typ 1)
- <Pause>
- Turing-Maschinen
- Entscheidbarkeit & Berechenbarkeitstheorie
- Komplexitätstheorie & O-Notation
- Komplexitätsklassen
- **NP-Vollständigkeit**

NP-Vollständigkeit

- Schwerwiegendste Probleme in NP.
- Mindestens so schwer, wie alle anderen Probleme in NP.
- Alle anderen Probleme in NP lassen sich als Spezialfälle eines NP-vollständigen Problem modellieren.
- Existenz lange Zeit ungeklärt.
- Erstes NP-vollständige Problem entdeckt durch Cook und Levin

Reduktionen

- Überführung eines Problems in einen Spezialfall eines schwerwiegenderen (oder mindestens so schweren) Problems.
- Reduktion von A (kleineres Problem) nach B (schwereres Problem) notiert als: $A \leq B$
 - Achtung: $\leq$ trifft keine Aussage über eine Richtung, sondern über die Schwere der Probleme
- Ist die Reduktion deterministisch in Polynomialzeit möglich, schreibt man: $A \leq_{\text{POL}} B$

NP-Vollständigkeit

- Ein Problem L ist NP-vollständig gdw.:
 - L in NP liegt. UND
 - Alle anderen Probleme A aus NP in deterministischer Polynomialzeit auf L reduzierbar sind. (L „NP-schwer“ ist)
- NP-Schwere ist transitiv:
 - NP-Schwere kann durch Reduktion eines NP-schweren Problem auf das neue Problem gezeigt werden

E N D E

Vielen Dank für eure Aufmerksamkeit!
Viel Erfolg bei der Klausur!

Wenn es aus irgendwelchen Gründen doch nicht
reichen sollte, und ihr zum 2. Termin müsst:
Ich gebe ich auch privat Nachhilfe! (25€ / 60 Min.).
Einfach an 8roeweka@inf... mailen.

Wenn ihr **Feedback** habt (negativ/positiv egal), könnt
ihr mir auch gern mailen, ich freue mich über jede
Rückmeldung! 😊