Automaten und Sprachen

§ 12: Die Struktur kontextfreier Sprachen

Überblick und Ausblick

Überblick

Teil I: Endliche Automaten und reguläre Sprachen

0. Grundbegriffe
1. Endliche Automaten
2. Nachweis der Nichterkennbarkeit
3. Abschlusseigenschaften
4. Entscheidungsprobleme
5. Reguläre Ausdrücke und Sprachen
6. Minimale DEAs und die Nerode-Rechtskongruenz

Teil II: Grammatiken, kontextfreie Sprachen und Kellerautomaten

7. Die Chomsky-Hierarchie
8. Rechtslineare Grammatiken und reguläre Sprachen
9. Normalformen und Entscheidungsprobleme
10. Abschlusseigenschaften und Pumping-Lemma
11. Kellerautomaten
12. Die Struktur kontextfreier Sprachen

§ 12: Die Struktur kontextfreier Sprachen

Dyck-Sprachen

Was unterscheidet kontextfreie von regulären Sprachen?

Eine bereits erwähnte Intuition ist:
kontextfreie Sprachen können unbeschränkt zählen, reguläre nicht.

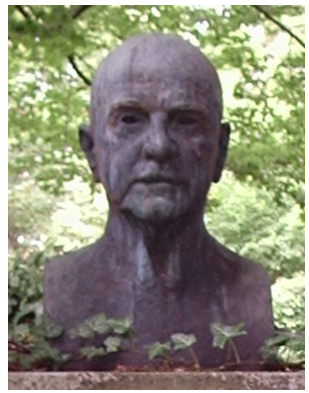
„Typische“ kontextfreie Sprachen, die nicht regulär sind:

- $\{a^n b^n \mid n \geq 0\}$
- Klammersprachen, auch genannt Dyck-Sprachen

  Dyck-Sprache D_n, $n \geq 1$, wird erzeugt durch Grammatik

$$S \longrightarrow \underset{1}{(}\, S \,\underset{1}{)} \qquad \ldots \qquad S \longrightarrow \underset{n}{(}\, S \,\underset{n}{)}$$

$$S \longrightarrow SS \qquad S \longrightarrow \varepsilon$$



Walther von Dyck
Foto: Zeitlupe CC BY-SA 3.0

Gibt es noch „andere Arten“ echt kontextfreier Sprachen?

Dyck-Sprachen

Wir wollen zeigen: in gewisser Weise ist das nicht der Fall

Jede kontextfreie Sprache kann dargestellt werden

- als Schnitt einer Dyck-Sprache und einer regulären Sprache,
- plus „ein wenig Umbenennung“

Die „Umbenennung“ erledigen wir mit Homomorphismen.

Beispiel 12.1

$$\{a^n b^n \mid n \geq 0\} \quad = \quad h(D_1 \cap R), \qquad \text{wobei}$$

- D_1 ist Dyck-Sprache, definiert durch $S \longrightarrow SS,\ S \longrightarrow (S),\ S \longrightarrow \varepsilon$
- R ist die reguläre Sprache $(^*\,)^*$
- h benennt „(“ in a um und „)“ in b

Homomorphismen

Zur Erinnerung:

Definition 3.3 (Homomorphismus)

Seien Σ und Γ Alphabete. Ein Homomorphismus von Σ^* nach Γ^* ist eine Abbildung $h : \Sigma^* \to \Gamma^*$, so dass $h(wv) = h(w)h(v)$ für alle $w, v \in \Sigma^*$.

Aus dieser Definition folgt unmittelbar:

1. $h(\varepsilon) = \varepsilon$
2. $h(a_1 \cdots a_n) = h(a_1) \cdots h(a_n)$,

Also kann man h durch Angabe von $h(a) \in \Gamma^*$ für alle $a \in \Sigma$ definieren

Beispiel: $h(a) = ccc$ $h(b) = \varepsilon$ $h(c) = ab$

Dann gilt $h(abcab) = cccabccc$

Homomorphismen

Wir haben bereits gesehen: die regulären Sprachen sind unter Homomorphismen abgeschlossen.

Dasselbe gilt für die kontextfreien Sprachen:

Satz 12.2

Sei $L \subseteq \Sigma^*$ eine kontextfreie Sprache und $h : \Sigma^* \to \Gamma^*$ ein Homomorphismus.

Dann ist $h(L)$ ebenfalls eine kontextfreie Sprache.

Beweis.

Sei $G = (N, \Sigma, P, S)$ eine kontextfreie Grammatik für L.

Konstruiere Grammatik $G' = (N, \Sigma, P', S)$ mit

$$P' = \{A \longrightarrow \widehat{h}(w) \mid A \longrightarrow w \in P\}$$

h + Identität af N

Man prüft leicht, dass $L(G') = h(L)$. ❑

Dyck-Sprachen

Beispiel 12.3

$$\{ww^R \mid w \in \{a,b\}^*\} \quad = \quad h(D \cap R), \qquad \text{wobei}$$

- $D = D_2$ Dyck-Sprache, definiert durch

$$S \longrightarrow \underset{1}{(}S\underset{1}{)} \qquad S \longrightarrow \underset{2}{(}S\underset{2}{)}$$

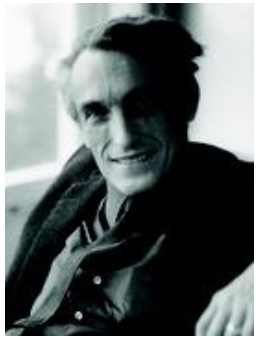
$$S \longrightarrow SS \qquad S \longrightarrow \varepsilon$$

- R ist die reguläre Sprache $(\underset{1}{(} + \underset{2}{()}^*\underset{1}{()} + \underset{2}{)})^*$

- h benennt „$\underset{1}{(}$“ und „$\underset{1}{)}$“ in a um sowie „$\underset{2}{(}$“ und „$\underset{2}{)}$“ in b

Satz von Chomsky und Schützenberger

Noam Chomsky
Foto: Duncan Rawlinson
CC BY 2.0



Marcel Schützenberger
Foto: Konrad Jacobs
CC BY-SA 2.0 DE, MFO

Satz 12.4 (Chomsky-Schützenberger)

Jede kontextfreie Sprache L ist das homomorphe Bild des Schnittes einer Dyck-Sprache D und einer regulären Sprache R.

Es gibt also einen Homomorphismus h, so dass:

$$L = h(D \cap R)$$

D. h.: jede kontextfreie Sprache ist quasi (modulo Umbenennung durch Homomorphismen) eine reguläre Teilmenge einer Dyck-Sprache.

Satz von Chomsky und Schützenberger

Beweis. Sei L kontextfrei.

Dann gibt es kfG $G = (N, \Sigma, P, S)$ für L in Chomsky-Normalform.

Für jede Produktion $\pi \in P$ definiere

$$\pi' = \begin{cases} A \longrightarrow \overset{1}{\underset{\pi}{(}} B \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} C \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow BC \\ A \longrightarrow \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow a \end{cases}$$

Dabei sind die $\overset{i}{\underset{\pi}{(}}$ und $\overset{i}{\underset{\pi}{)}}$ Terminalsymbole

Setze G' := (N, Γ, P', S) mit

$$P' = \{\pi' \mid \pi \in P\} \qquad \Gamma = \left\{ \overset{1}{\underset{\pi}{(}}, \overset{1}{\underset{\pi}{)}}, \overset{2}{\underset{\pi}{(}}, \overset{2}{\underset{\pi}{)}} \;\middle|\; \pi \in P \right\}$$

Wir betrachten als Beispiel eine Grammatik für $\{a^n b^n \mid n \geq 1\}$

Satz von Chomsky und Schützenberger

Beweis. Sei L kontextfrei.

Dann gibt es kfG $G = (N, \Sigma, P, S)$ für L in Chomsky-Normalform.

Für jede Produktion $\pi \in P$ definiere

$$\pi' = \begin{cases} A \longrightarrow \overset{1}{\underset{\pi}{(}} B \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} C \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow BC \\ A \longrightarrow \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow a \end{cases}$$

Folgender Homomorphismus h stellt die alten Terminalsymbole wieder her:

- $h\left(\overset{1}{\underset{\pi}{(}}\right) = h\left(\overset{1}{\underset{\pi}{)}}\right) = h\left(\overset{2}{\underset{\pi}{(}}\right) = h\left(\overset{2}{\underset{\pi}{)}}\right) = \varepsilon$ wenn $\pi = A \longrightarrow BC$
- $h\left(\overset{1}{\underset{\pi}{(}}\right) = a$ und $h\left(\overset{1}{\underset{\pi}{)}}\right) = h\left(\overset{2}{\underset{\pi}{(}}\right) = h\left(\overset{2}{\underset{\pi}{)}}\right) = \varepsilon$ wenn $\pi = A \longrightarrow a$

Man zeigt leicht: $L(G) = h(L(G'))$ Aber ist $L(G')$ eine Dyck-Sprache?

Braucht man also gar keinen Schnitt mit einer regulären Sprache?

Satz von Chomsky und Schützenberger

$$\pi' = \begin{cases} A \longrightarrow \overset{1}{\underset{\pi}{(}} B \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} C \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow BC \\ A \longrightarrow \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow a \end{cases}$$

Sei D_Γ die Dyck-Sprache mit den Klammern $\Gamma = \left\{ \overset{1}{\underset{\pi}{(}}, \overset{1}{\underset{\pi}{)}}, \overset{2}{\underset{\pi}{(}}, \overset{2}{\underset{\pi}{)}} \;\middle|\; \pi \in P \right\}$

Man sieht leicht, dass alle Wörter in $L(G')$ wohlgeklammert sind, also

$$L(G') \subseteq D_\Gamma$$

Die Umkehrung gilt allerdings nicht, z. B.: $\overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \notin L(G')$

$L(G')$ erfüllt also zusätzliche Eigenschaften!

Wenn wir diese als reguläre Sprache R beschreiben können, gilt:

$$L(G) = h(L(G')) = h(D_\Gamma \cap R)$$

Satz von Chomsky und Schützenberger

$$\pi' = \begin{cases} A \longrightarrow \overset{1}{\underset{\pi}{(}} B \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} C \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow BC \\ A \longrightarrow \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow a \end{cases}$$

Zusätzliche Eigenschaften, die alle Wörter in $L(G')$ erfüllen:

1. Auf jedes $\overset{1}{\underset{\pi}{)}}$ folgt $\overset{2}{\underset{\pi}{(}}$
2. Auf $\overset{2}{\underset{\pi}{)}}$ folgt nie eine öffnende Klammer (sondern schließende Klammer oder Wortende)
3. Wenn $\pi = A \longrightarrow BC$, dann
   - folgt auf $\overset{1}{\underset{\pi}{(}}$ immer $\overset{1}{\underset{\rho}{(}}$ mit $\rho = B \longrightarrow \cdots$
   - folgt auf $\overset{2}{\underset{\pi}{(}}$ immer $\overset{1}{\underset{\sigma}{(}}$ mit $\sigma = C \longrightarrow \cdots$
4. Wenn $\pi = A \longrightarrow a$, dann folgt auf $\overset{1}{\underset{\pi}{(}}$ immer $\overset{1}{\underset{\pi}{)}}$ und auf $\overset{2}{\underset{\pi}{(}}$ immer $\overset{2}{\underset{\pi}{)}}$

Satz von Chomsky und Schützenberger

$$\pi' = \begin{cases} A \longrightarrow \overset{1}{\underset{\pi}{(}} B \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} C \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow BC \\ A \longrightarrow \overset{1}{\underset{\pi}{(}} \overset{1}{\underset{\pi}{)}} \overset{2}{\underset{\pi}{(}} \overset{2}{\underset{\pi}{)}} & \text{wenn } \pi = A \longrightarrow a \end{cases}$$

Für alle Wörter w mit $A \vdash_{G'}^{*} w$ gilt:

5_A. w beginnt mit $\overset{1}{\underset{\pi}{(}}$ wobei $\pi = A \longrightarrow \cdots$

Jede dieser Eigenschaften ist als reguläre Sprache beschreibbar

Also ist für jedes $A \in N$ die folgende Sprache regulär:

$$R_A := \{ w \in \Gamma^* \mid w \text{ erfüllt Eigenschaften 1–4 sowie } 5_A \}$$

Ist diese Liste vollständig, erfasst alle Unterschiede zwischen $L(G')$ und D_Γ?

Es stellt sich heraus, dass das der Fall ist

Satz von Chomsky und Schützenberger

Behauptung

Für alle $A \in N$ und $w \in \Gamma^*$ gilt: $A \vdash_{G'}^* w$ gdw. $w \in D_\Gamma \cap R_A$

„⇒“ Wir hatten uns bereits überzeugt: wenn $A \vdash_{G'}^* w$, dann

1. ist w wohlgeklammert, also $w \in D_\Gamma$

   (formaler Beweis per Induktion über die Länge von w)

2. erfüllt w Eigenschaften 1-5$_A$, also $w \in R_A$

„⇐“ per Induktion über die Länge von w

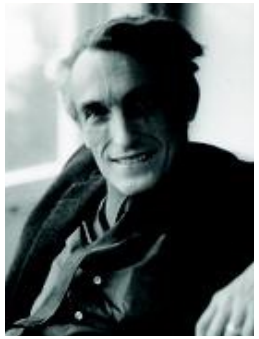
Details im Skript

Wir wählen nun R_S als reguläre Sprache R, erhalten

$$L(G) = h(L(G')) = h(D_\Gamma \cap R)$$

Satz von Chomsky und Schützenberger

Noam Chomsky
Foto: Duncan Rawlinson
CC BY 2.0



Marcel Schützenberger
Foto: Konrad Jacobs
CC BY-SA 2.0 DE, MFO

Satz 12.4 (Chomsky-Schützenberger)

Jede kontextfreie Sprache L ist das homomorphe Bild des Schnittes einer Dyck-Sprache D und einer regulären Sprache R.

Es gibt also einen Homomorphismus h, so dass:

$$L = h(D \cap R)$$

D. h.: jede kontextfreie Sprache ist quasi (modulo Umbenennung durch Homomorphismen) eine reguläre Teilmenge einer Dyck-Sprache.

Zusammenfassung von Automaten und Sprachen

Behandelte Themen

- **Sprachklassen (Chomsky-Hierarchie):**
  regulär = erkennbar = rechtslinear, deterministisch kontextfrei, kontextfrei, kontextsensitiv, Typ 0

- **Automatenmodelle zur Beschreibung von Sprachen:**
  NEAs, DEAs, ε-NEAs, Wort-NEAs; PDAs, dPDAs

- **Andere Mechanismen, um Sprachen endlich zu beschreiben:**
  reguläre Ausdrücke, verschiedene Arten von Grammatiken

- **Eigenschaften von Sprachklassen:**
  Abschlusseigenschaften, Entscheidbarkeit und Komplexität von Problemen

- **Konstruktionen und Beweistechniken:**
  Potenzmengenkonstruktion, Produktautomat, Quotientenautomat, Nerode-Rechtskongruenz, zwei Pumping-Lemmas, Normalformen von Grammatiken etc.

Überblick Abschlusseigenschaften

	$\cap$	$\cup$	$^-$	$\bullet$	*
Typ 0	✓	✓	✗	✓	✓
kontext-sensitiv	✓	✓	✓	✓	✓
kontextfrei	✗	✓	✗	✓	✓
determ. kontextfrei	✗	✗	✓	✗	✗
regulär	✓	✓	✓	✓	✓

nächstes Semester

Überblick Entscheidungsprobleme

	Wortproblem	Leerheitsprob.	Äquivalenzprob.
Typ-0-Gramm.	*unentscheidbar*	*unentscheidbar*	*unentscheidbar*
Typ-1-Gramm.	entscheidbar, „nicht Polyzeit“	*unentscheidbar*	*unentscheidbar*
Typ-2-Gramm./ PDA	Polyzeit	Polyzeit	*unentscheidbar*
dPDA	Linearzeit	Polyzeit	entscheidbar [2001]
Typ-3-Gramm./ NEA/reg. Ausdr.	Linearzeit	Linearzeit	entscheidbar, „nicht Polyzeit“
DEA	Linearzeit	Linearzeit	Polyzeit

nächstes Semester

Kurzer Ausblick auf Berechenbarkeit

Hauptthemen

In Berechenbarkeit betrachten wir zwei fundamentale Themen der Informatik:

- **Entscheidbarkeit / Berechenbarkeit**

  Welche Probleme sind algorithmisch entscheidbar und welche nicht?

  Beispiele für unentscheidbare Probleme:

  Äquivalenzproblem für PDAs, Wortproblem für Typ-0-Grammatiken

- **Komplexität**

  Wenn ein Problem entscheidbar ist,
  wie viel Zeit und Speicherplatz benötigt man mindestens/höchstens?

  Beispiel: Das Äquivalenzproblem für NEAs
  kann man (wahrscheinlich) nicht in polynomieller Zeit entscheiden.

Entscheidbarkeit / Berechenbarkeit

Wir haben **Entscheidbarkeit** zahlreicher Probleme nachgewiesen, z. B.:

- Wortproblem für kontextfreie Grammatiken
- Äquivalenzproblem für DEAs

Verwendete Methoden:

- Angabe eines Algorithmus in Pseudocode (z. B. CYK-Algorithmus)
- Beschreibung des Verfahrens, so dass Implementierung möglich ist, z. B.:

  Konstruktion des Quotientenautomaten

  Test auf Isomorphie

☞ genug Information für Implementierung in konkreter Programmiersprache!

Entscheidbarkeit / Berechenbarkeit

Wie beweist man Unentscheidbarkeit, also dass für ein Problem kein Algorithmus existiert?

Dazu muss man zunächst die Frage beantworten:

Was ist ein Algorithmus?

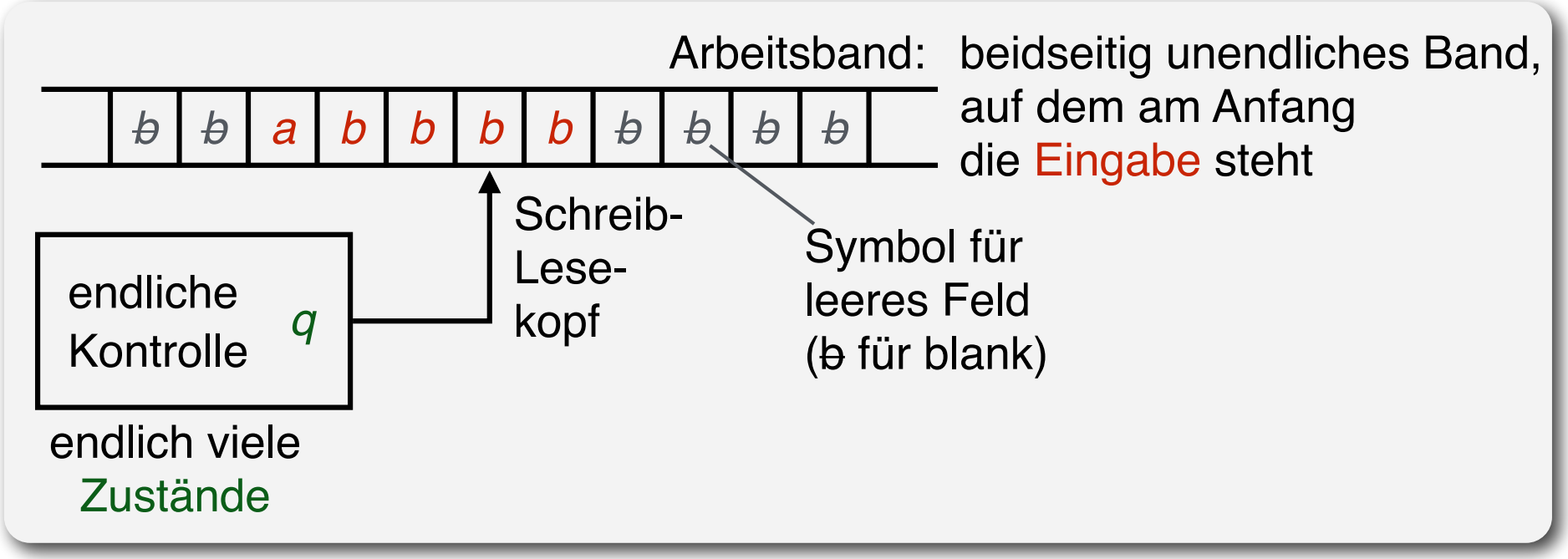
Mögliche Antworten:

- **Programmiersprachen**
  C, Pascal, Java, Lisp, Prolog, Assembler, ...
- **Mathematische Formalismen**
  Turingmaschine, Registermaschine, *while*-Programme,
  μ-berechenbare Funktionen, λ-Kalkül, Abstract State Machines, ...

Interessante Beobachtung: alle diese Modelle sind gleichmächtig!

Turingmaschinen

Wir wählen ein möglichst einfaches Modell, die **Turingmaschine:**



- Kopf bewegt sich in jedem Schritt um max. ein Feld nach links oder rechts
- Akzeptieren/Verwerfen über akzeptierende Zustände

Die mit TM entscheidbaren Probleme sind genau die
mit Java-Programmen, Lisp-Programmen usw. entscheidbaren

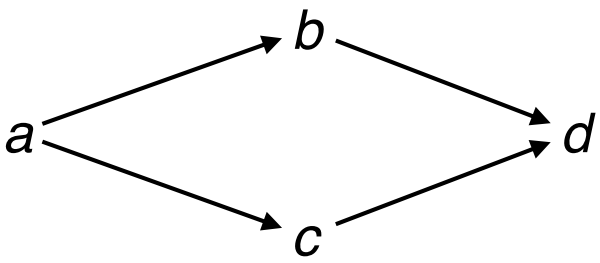
Christos Papadimitriou: „It's amazing how little we need to have everything."

Entscheidungsprobleme

Um (Un)entscheidbarkeit zu definieren, betrachtet man Entscheidungsprobleme als formale Sprachen.

Beispiel: Erreichbarkeit in gerichteten Graphen

Eingabe



Frage

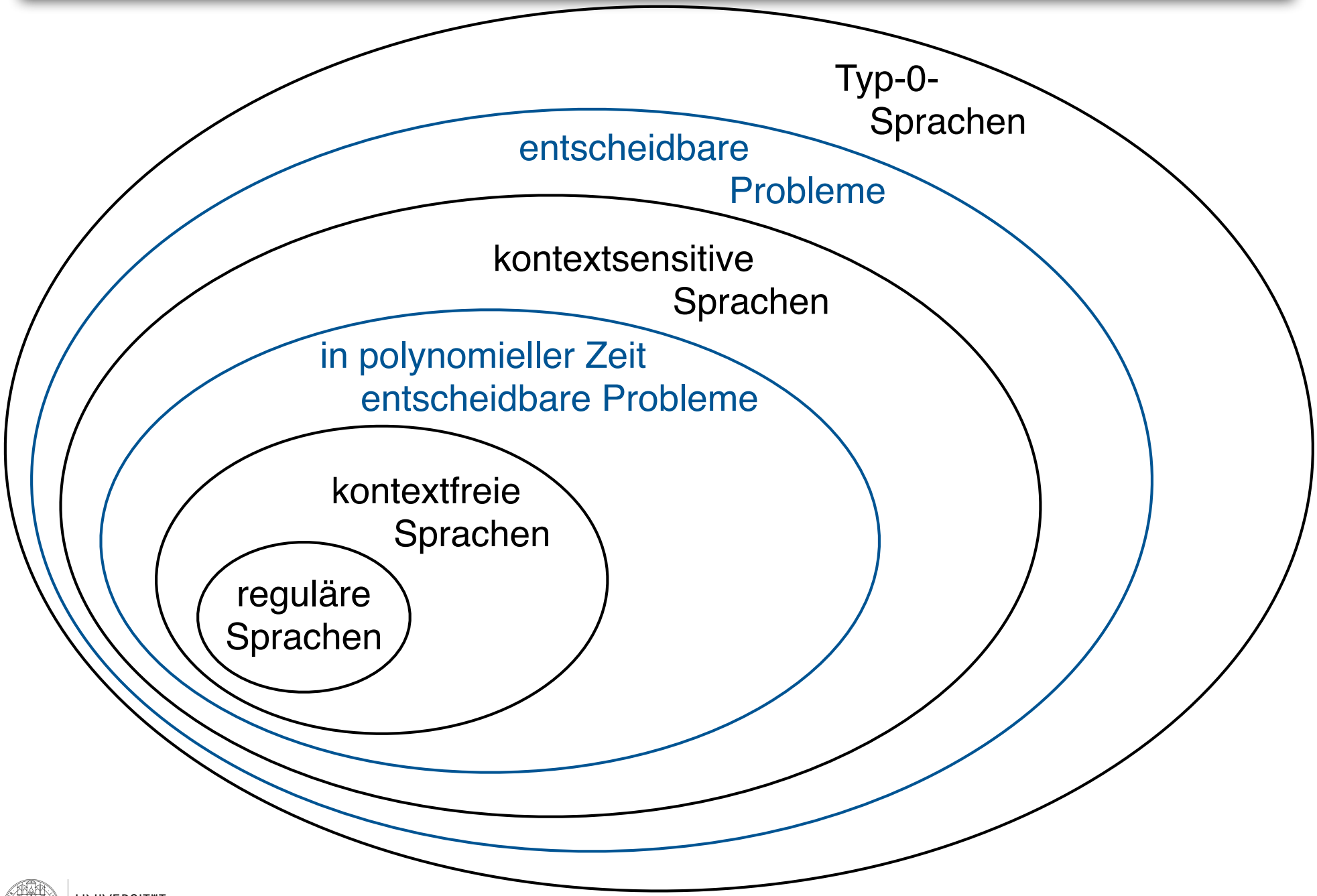
Ist d erreichbar von a?

dargestellt als Wort: 00/01#00/10#01/11#10/11##00/11

Eingabe | Frage

Auch die Betrachtung von Entscheidbarkeit und Komplexität ist also in gewisser Weise nichts weiter als das Studium formaler Sprachen.

Entscheidbare Probleme im Kontext



Turingmaschinen als Automaten

Turingmaschinen liefern zudem Automatenmodelle für Typ 0 und Typ 1:

- **Typ 0**

  Eine Sprache ist von Typ 0 (mit Grammatik erzeugbar)
  gdw. sie von einer Turingmaschine erkannt wird.

- **Typ 1**

  Eine Sprache ist von Typ 1 (mit monotoner Grammatik erzeugbar)
  gdw. sie von einem linear beschränkten Automaten (LBA) erkannt wird.

  LBA:

  TM, die nur den von der Eingabe belegten Teil des Bandes nutzen darf

Komplexitätstheorie

… klassifiziert Entscheidungsprobleme in Komplexitätsklassen gemäß der Ressourcen, die zum Entscheiden benötigt werden

Wichtige Komplexitätsklassen z. B.:

- **P**

  Menge der Probleme, die mit einer polynomiell zeitbeschränkten deterministischen TM entschieden werden können

- **NP**

  Menge der Probleme, die mit einer polynomiell zeitbeschränkten nichtdeterministischen TM entschieden werden können

- **PSpace**

  Menge der Probleme, die mit einer polynomiell platzbeschränkten TM entschieden werden können

Komplexitätstheorie

Damit kann man dann auch präzisere Aussagen über Probleme treffen,

die in dieser VL als “vermutlich nicht in Polyzeit lösbar” bezeichnet wurden:

- das Äquivalenzproblem für NEAs / reguläre Ausdrücke /

  Typ 3 Grammatiken ist PSpace-vollständig

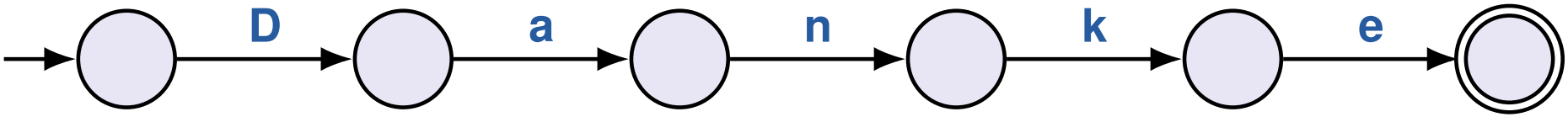
- die Komplexität des Äquivalenzproblems für dPDAs ist

  ungeklärt:

  das Problem ist NP-schwer und (primitiv rekursiv) entscheidbar,

  die genaue Komplexität ist offen.

Das war's!



für eure Aufmerksamkeit!