



# **Technische Informatik**

**Vorlesungsskript**

Mitschrift von Falk-Jonatan Strube

Vorlesung von Dr. Boris Hollas

15. März 2016

## Inhaltsverzeichnis

|          |   |          |
|----------|---|----------|
| <b>1</b> | <b>Automaten und Formale Sprachen</b>               | <b>3</b> |
| 1.1      | Formale Sprachen . . . . .                          | 3        |
| 1.2      | Reguläre Sprachen . . . . .                         | 4        |
| 1.2.1    | Deterministische endliche Automaten (DFA) . . . . . | 4        |

## Inhalte

Grundlage: Grundkurs Theoretische Informatik [1]

- Formale Sprachen
  - Reguläre Sprachen
    - \* Endliche Automaten
    - \* Reguläre Ausdrücke
  - Nichtreguläre Sprachen
  - Kontextfreie Sprachen
    - \* Kellerautomaten
    - \* Grammatiken
- Berechenbarkeit
  - Halteproblem
- Komplexitätsklassen
  - $P$
  - $NP$
  - $NP$ -vollständige Probleme

## 1 Automaten und Formale Sprachen

### 1.1 Formale Sprachen

**Def.:** Ein Alphabet ist eine Menge  $\Sigma \neq \emptyset$  (Symbole in  $\Sigma$  – müssen nicht einzelne Buchstaben sein, auch Wörter usw. [bspw. „if“ oder „else“ im Alphabet der Programmiersprache C]).

**Def.:** Für  $w_1, \dots, w_n \in \Sigma$  ist  $w = w_1 \dots w_n$  ein Wort der Länge  $n$ .

Das Wort  $\varepsilon$  ist das *leere Wort*.

Die Menge aller Wörter bezeichnen wir mit  $\Sigma^*$  (einschließlich dem leeren Wort).

**Bsp.:**  $\Sigma = \{a, b, c\} \rightarrow \Sigma^* = \{\varepsilon, a, b, c, aa, ab, ac, aaa, \dots\}$

**Def.:** Für Wörter  $a, b \in \Sigma^*$  ist  $ab$  die Konkatenation dieser Wörter.

Für ein Wort  $w$  ist  $w^n$  die  $n$ -fache Konkatenation von  $w$ , wobei  $w^0 = \varepsilon$ .

**Bemerkung:** Für alle  $w \in \Sigma^*$  gilt  $\varepsilon w = w = w\varepsilon$ .  $\varepsilon$  ist also das neutrale Element der Konkatenation.

**Def.:** Eine *formale Sprache* ist eine Teilmenge von  $\Sigma^*$ .

**Def.:** Für Sprachen  $A, B$  ist  $AB = \{ab \mid a \in A, b \in B\}$  sowie  $A^n = \prod_{i=1}^n A$ , wobei  $A^0 = \{\varepsilon\}$ .

**Bemerkung:**  $\emptyset, \varepsilon, \{\varepsilon\}$  sind unterschiedliche Dinge (leere Menge, leeres Wort, Menge mit leerem Wort).

**Bemerkung:**  $\Sigma^*$  lässt sich ebenfalls definieren durch  $\Sigma^* = \bigcup_{n \geq 0} \Sigma^n$ .

Ferner ist  $\Sigma^+ = \Sigma^* - \{\varepsilon\}$ .

## 1.2 Reguläre Sprachen

### 1.2.1 Deterministische endliche Automaten (DFA)

**Bsp.:**

ABB1

(Pfeil zeigt auf Startzustand, Endzustand ist doppelt umrandet)

Dieser DFA akzeptiert alle Wörter über  $\Sigma = \{a, b, c\}$ , die  $abc$  enthalten.

Deterministisch: Es gibt genau ein Folgezustand. Von jedem Knoten aus gibt es genau eine Kante für jedes Zeichen, nicht mehrere und nicht keine.

**Bsp.:**

ABB2

Dieser DFA erkennt die Sprache  $\{a, b, c\}^*$ .

**Def.:** Ein DFA ist ein Tupel  $\mathcal{M} = (Z, \Sigma, \delta, z_0, E)$

- $Z$ : Menge der Zustände
- $\Sigma$ : Eingabealphabet
- $\delta$ : Überföhrungsfunktion  $Z \times \Sigma \rightarrow Z$ . Dabei bedeutet  $\delta(z, a) = z'$ , dass  $\mathcal{M}$  im Zustand  $z$  für das Zeichen  $a$  in den Zustand  $z'$  wechselt.
- $z_0 \in Z$ : Startzustand
- $E$ : Menge der Endzustände

$\delta$ : ABB3

**Def.:** Die erweiterte Überföhrungsfunktion  $\hat{\delta} : Z \times \Sigma^* \rightarrow Z$  ist definiert durch

$$\hat{\delta}(z, w) = \begin{cases} z & \text{für } w = \varepsilon \\ \hat{\delta}(\delta(z, a), x) & \text{für } w = ax \text{ mit } a \in \Sigma, x \in \Sigma^* \end{cases}$$

Dazu vergleichbarer C-Code:

```
1 int δ̂(int z, char* w){
2     if ( strlen(w) == 0 )
3         return z;
4     else
5         return δ̂(δ(z, w[0]), w[1]);
```

Veranschaulichung:

ABB4

Die erweiterte Überföhrungsfunktion bestimmt den Zustand nach dem vollständigen Lesen eines Wortes.

**Bsp.:**  
ABB5

$$\begin{aligned}\hat{\delta}(z_0, aaba) &= \hat{\delta}(\delta(z_0, a), aba) = \\ \hat{\delta}(z_0, aba) &= \hat{\delta}(\delta(z_0, a), ba) = \\ \hat{\delta}(z_0, ba) &= \hat{\delta}(\delta(z_0, b), a) = \\ \hat{\delta}(z_E, a) &= \hat{\delta}(\delta(z_E, a), \varepsilon) = \\ \hat{\delta}(z_E, \varepsilon) &= z_E\end{aligned}$$

Die von  $\mathcal{M}$  *akzeptierte Sprache* ist  $L(\mathcal{M}) = \{w \in \Sigma^* \mid \hat{\delta}(z_0, w) \in E\}$

## Literatur

- [1] Boris Hollas. *Grundkurs Theoretische Informatik: Mit Aufgaben und Anwendungen*. Springer Berlin, 2015.