

# 1 Begriffe

**Alphabet**  $\Sigma = \{\dots\}$  ist die Menge von Zeichen, aus welchen die Wörter der definierten Sprache aufgebaut werden.

**Menge aller Wörter eines Alphabets**  $\Sigma^*$  ist die Menge aller aus dem Alphabet  $\Sigma$  gebauten Wörter. Beispiel:  $\Sigma = \{a, b\}$

$$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb, \dots\}$$

**Wort**  $w \in \Sigma^*$  ist eine Kombination aus Zeichen aus dem Alphabet.

**Leeres Wort**  $\varepsilon \in \Sigma^*$ ,  $|\varepsilon| = 0$  ist das Wort mit der Länge 0.

**Sprache**  $L(A)$  ist die Sprache, welche von einem Automaten akzeptiert wird. Man sagt  $A$  *akzeptiert*  $L$ .

**regulär** Eine Sprache ist dann regulär, wenn es einen DEA gibt, der sie akzeptiert.

**DEA** Deterministischer Endlicher Automat

**deterministisch** Der Determinismus ist die Auffassung, dass zukünftige Ereignisse durch Vorbedingungen eindeutig festgelegt sind (für jedes Zeichen ist in jedem Zustand ein Übergang vorhanden).

**nicht deterministisch** Gegenteil von zuvor (es kann für ein Zeichen mehrere Übergänge geben und es werden nur Übergänge eingezeichnet, welche für das Erreichen des Akzeptierzustandes nötig sind).

**NEA** Nichtdeterministischer Endlicher Automat. Kann auch  $\varepsilon$ -Übergänge enthalten, dabei wird von einem  $NEA_\varepsilon$  gesprochen.

**akzeptieren** Ein Wort wird von einem Automaten akzeptiert, wenn sich der Automat nach dem einlesen des gesamten Wortes in einem Akzeptierzustand befindet.

**akzeptierte Sprache eines DEA**  $L(A) = [w \in \Sigma^* | w \text{ wird von } A \text{ akzeptiert}]$  beinhaltet alle Wörter, welche vom Automaten  $A$  akzeptiert werden.

**regulärer Ausdruck** Notation nach Vorgaben, welche einen Automaten mit \*-Operationen, Alternativen und Verkettungen beschreibt.

**Grammatik** Beschreibung einer kontextfreien Sprache in der Form  $S \rightarrow SS$  oder  $S \leftarrow s$ , welche beschreibt, wie sich Nichtterminalsymbole (in diesem Fall  $S$  und  $SS$ ) "entwickeln" können (i entwickelt sich  $S$  zu  $SS$  oder dem Terminalsymbol  $s$ ).

**kontextfrei** Unabhängig von der Umgebung anwendbar (z.B. Kommentare in der Programmiersprache C können in jedem Programm erkannt werden).

**Variable** Nichtterminalsymbol

**Regel**  $\rightarrow$

**Ableitung** Erstellen einer Kombination von Terminalsymbolen (a.k.a. Wort) aus einer Variable (Frage ist jeweils: Kann  $w$  aus  $S$  abgeleitet werden?)

**Parse Tree** "Baum", welcher beim ableiten eines Wortes aus einer Variable entsteht. Falls ein Wort mehrere Parse Trees erlaubt, ist die Grammatik nicht in Chomsky-Normalform.

**erzeugte Sprache einer Grammatik** Alle Wörter, die aus einer Grammatik abgeleitet werden können.

**Stackautomat** NEA, welcher über einen Speicher (Stack) verfügt, bei welchem er jeweils auf das Oberste Element zugreifen kann.

**Turing-Maschine** NEA, der über einen unendlichen Speicher (Band) verfügt und einen Lesekopf beliebig auf diesem Band bewegen kann.

**erkannte Sprache einer TM** ???

**Entscheider** Ein Entscheider ist eine TM, die auf jeden beliebigen Input anhält.

**entscheidbare Sprache** Eine Sprache  $L$  ist entscheidbar, wenn es einen Entscheider  $M$  gibt mit  $L = L(M)$ . Man sagt  $M$  *entscheidet*  $L$ .

**Akzeptanzproblem** Es kann nicht entschieden werden, ob:

$$A_{DEA} = \{\langle A, w \rangle \mid A \text{ ein DEA, der } w \text{ akzeptiert.}\}$$

**abzählbar unendlich** Die Elemente einer abzählbar unendlichen Menge lassen sich mit  $n \in \mathbb{N}$  nummerieren.

**überabzählbar unendlich** Die Elemente einer überabzählbar unendlichen Menge werden mit  $r \in \mathbb{R}$  nummeriert. Es lässt sich also noch immer eine Zahl finden, die in mindestens einer Stelle zu allen anderen unterschiedlich ist.

**Reduktion** ???

**P und NP**

**NP-vollständig**

**polynomielle Reduktion**

**SAT**

**3SAT**

**Clique**

**Turing-vollständig**

**LOOP**

## 2 Fragen

### 1. Was ist $\Sigma^*$ ?

Die Menge aller Wörter aus dem Alphabet  $\Sigma$ ?

### 2. Was ist der Unterschied zwischen $\varepsilon$ , $\emptyset$ und $\{\varepsilon\}$

$\varepsilon$  ist das leere Wort mit der Länge  $|w| = 0$ ,  $\emptyset$  ist die leere Sprache, welche gar keine Wörter enthält und  $\{\varepsilon\}$  ist Menge/Sprache, die nur das leere Wort enthält.

### 3. Wie unterscheidet sich ein *DEA* von einem *NEA*?

Ein *DEA* hat zu jedem Zustand für jedes Zeichen genau einen Übergang (Voraussetzung für den Determinismus). Ein *NEA* wiederum darf für jeden Zustand beliebige Übergänge haben, also für jedes Zeichen mehrere, einen oder gar keinen Übergang. Ein *NEA* <sub>$\varepsilon$</sub>  kann auch sogenannte  $\varepsilon$ -Übergänge haben, welche ohne Input funktionieren.

### 4. Wie kann man zwei endliche Automaten vergleichen?

Man bildet von beiden Automaten den minimalen Automaten z.B. mittels "Kreuzchenalgorithmus". Diese Minimalautomaten sind bei gleichen Automaten ebenfalls gleich.

### 5. Nennen Sie drei Methoden, mit denen sie zeigen können, dass eine Sprache regulär ist.

Eine Sprache ist regulär, wenn sie einen DEA besitzt, durch einen regulären Ausdruck beschrieben werden kann oder nur die Operationen Alternative, Verkettung oder \*-Operation verwendet wird.

### 6. Nennen Sie zwei Methoden, mit denen sie zeigen können, dass eine Sprache nicht regulär ist.

Pumping Lemma oder wenn sie keinen DEA besitzt.

### 7. Beschreiben Sie den Zusammenhang zwischen DEAs, NEAs und regulären Ausdrücken.

Ein NEA kann zu einem DEA gemacht werden (Zustandsmengenalgorithmus), ein DEA kann dann zu einem regulären Ausdruck gemacht werden.

### 8. Beschreiben Sie die fünf typischen Schritte, die notwendig sind, um mit dem Pumping Lemma zu beweisen, dass eine Sprache nicht regulär ist.

- (a) Annehmen, dass die Sprache  $L$  kontextfrei ist.
- (b) Es gibt eine Pumping Length  $N$ .
- (c) Ein Wort  $w \in L$  mit  $|w| \geq N$  Festlegen.
- (d) Wort aufteilen  $w = xyz$ ,  $|xy| \leq N$ ,  $|y| > 0$ .
- (e) Was passiert, wenn mit  $y$  gepumpt wird? Ist das neue Wort immer noch in der Sprache?
- (f) Widerspruch und Schlussfolgerung.

9. Beschreiben Sie DEAs für die Sprachen  $\emptyset$ ,  $\{\varepsilon\}$  und  $\Sigma^*$ .
10. Warum sind endliche Sprachen regulär?
11. Geben Sie ein typisches Beispiel für eine nicht reguläre Sprache.  

$$L = \{0^n 1^n \mid n \geq 0\}$$
12. Was ist eine kontextfreie Grammatik?
13. Was bedeutet  $w \in L(G)$ ?  
 $w$  ist ein Wort aus der Sprache  $L$  welche durch die Grammatik  $G$  definiert ist.
14. Welche Eigenschaften hat eine kontextfreie Grammatik in Chomsky-Normalform?  
Jedes Wort aus einer CFG in CNF hat einen eindeutigen Parse Tree.
15. Wie kann man eine Grammatik in Chomsky-Normalform bringen?  
Nach den folgenden Regeln:  
(a)
16. Ist die Chomsky-Normalform eindeutig?
17. Was müssen Sie tun um nachzuweisen, dass eine Sprache kontextfrei ist?
18. Wie funktioniert ein Stack-Automat?
19. Welche Eigenschaften muss eine Sprache haben, damit es einen Stack-Automaten gibt, der sie akzeptieren kann?
20. Beschreiben Sie eine Technik, mit der Sie zeigen können, dass eine Sprache nicht kontextfrei ist.
21. Geben Sie Grammatiken an für die Sprachen  $\emptyset$ ,  $\{\varepsilon\}$  und  $\Sigma^*$ .
22. Beschreiben Sie die fünf typischen Schritte, die notwendig sind, um mit dem Pumping Lemma zu beweisen, dass eine Sprache nicht regulär ist.
23. Geben Sie ein typisches Beispiel für eine nicht kontextfreie Sprache.
24. Gibt es eine Turing-Maschine mit nur einem Zustand?
25. Wieviele verschiedene Sprachen können von Turing-Maschinen mit zwei Zuständen erkannt werden. Warum?
26. Zählen Sie drei Varianten von Turing-Maschinen auf.
27. Sei  $M$  eine nicht deterministische Turing-Maschine und  $w \in \Sigma^*$ .  
Was heisst  $w \in L(M)$ ?
28. Warum gibt es Sprachen, die nicht Turing-erkennbar sind?

29. Was ist der Unterschied zwischen einer Turing erkennbaren Sprache und einer Turing entscheidbaren Sprache?
30. Was müssen Sie tun um nachzuweisen, dass eine turing-erkennbare Sprache entscheidbar ist?
31. Beschreiben Sie das prototypische nicht entscheidbare Problem für Turing-Maschinen.
32. Was bedeutet Reduktion eines Problems auf ein anderes?
33. Erklären Sie eine Standardtechnik, mit der man nachweisen kann, dass ein Problem nicht entscheidbar ist.
34. Geben Sie ein Beispiel eines nicht entscheidbaren Problems.
35. Was besagt das Halte-Theorem?
36. Wie ändert sich die Laufzeit eines Algorithmus, wenn man von einer Variante einer TM zu einer Standard TM übergeht?
37. Was bedeutet polynomielle Reduktion?
38. Wie ändert sich die Laufzeit, wenn man eine nichtdeterministische Maschine auf einer deterministischen Maschine simuliert?
39. Was bedeutet das Problem SAT?
40. Was unterscheidet ein Problem wie SAT von einem Problem wie die ganzzahlige Division von Zahlen?
41. Wie können Sie herausfinden, ob eine Programmiersprache Turing-vollständig ist?
42. Warum ist die Sprache LOOP nicht Turing-vollständig?
43. Gibt es Probleme, die man WHILE lösen kann, nicht aber GOTO?
44. Ist es möglich, einen Compiler zu schreiben, der C-Code in Brainfuck übersetzt?