Grundlagen der Theoretischen Informatik

Sommersemester 2021

15.04.2021

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Bis jetzt

- Organisatorisches
- Literatur
- Motivation und Inhalt
- Kurzer Überblick: Logik
- Kurzer Überblick: Beweismethoden und mathematische Konzepte

Inhalt der Vorlesung

1. Terminologie
2. Endliche Automaten und reguläre Sprachen
3. Kellerautomaten und kontextfreie Sprachen
4. Turingmaschinen und rekursiv aufzählbare Sprachen
5. Berechenbarkeit, (Un-)Entscheidbarkeit
6. Komplexitätsklassen P und NP

Inhalt der Vorlesung

1. Terminologie
2. Endliche Automaten und reguläre Sprachen
3. Kellerautomaten und kontextfreie Sprachen
4. Turingmaschinen und rekursiv aufzählbare Sprachen
5. Berechenbarkeit, (Un-)Entscheidbarkeit
6. Komplexitätsklassen P und NP

Mathematische Konzepte

Grundkonzepte (z.B. aus DAS)

- Elementare Mengentheorie
  - $\{x|P(x)\}$
  - $\cup$, $\cap$, $\setminus$
- Bezug zwischen Mengen, Relationen und Funktionen **(wichtig!)**
- Elementare Gesetze der Algebra
- Strukturen wie Listen, (endliche) Folgen, Graphen, Bäume
- Wörter (Strings)

Mathematische Konzepte

Funktionen

Funktion $f : S \to S'$: Abbildung zwischen den Grundmengen S und S', nicht unbedingt auf allen Elementen von S definiert

Definitionsbereich D $\quad D = \{x \in S \mid \exists y \in S' \text{ mit } (x, y) \in f\}$

Wertebereich W $\quad W = \{y \in S' \mid \exists x \in S \text{ mit } (x, y) \in f\}$

f eine totale Funktion: f für alle Elemente in S definiert

$$\forall x{\in}S\ \exists y{\in}W\ (x, y) \in f$$

$$(x, y) \in f \qquad \mapsto \qquad f(x) = y$$

Injektiv, surjektiv, bijektiv

Injektiv: $\forall x, y(f(x) = f(y) \to x = y)$

Surjektiv: $\forall y \in S'\ \ \exists x \in S : \quad f(x) = y$

Bijektiv: injektiv + surjektiv

Terminologie

In den folgenden Abschnitten führen wir die Begriffe

- **Sprache**
- **Grammatik**

ein.

Wir untersuchen insbesondere

1. wie man Probleme aus der Mathematik, Graphentheorie und Logik als **Probleme über Sprachen** formulieren kann.
2. wie man Klassen von Grammatiken von steigendem Schwierigkeitsgrad definiert: **Chomsky-Hierarchie**.
3. **wie viele** Grammatiken und Sprachen **es überhaupt gibt** (so viele wie natürliche Zahlen, reelle Zahlen oder komplexe Zahlen?)

Sprache, Grammatik

Alphabete, Wörter

Definition (Alphabet)

Ein **Alphabet** ist eine Menge von Zeichen/Buchstaben

- Grundlage einer Sprache (die zur Verfügung stehenden Zeichen)
- Meist endlich

Alphabete, Wörter

Definition (Alphabet)

Ein **Alphabet** ist eine Menge von Zeichen/Buchstaben

- Grundlage einer Sprache (die zur Verfügung stehenden Zeichen)
- Meist endlich

Definition (Wort)

Ein **Wort** (über einem Alphabet Σ)
ist eine endliche Folge von Zeichen aus Σ

- $|w|$ bezeichnet Länge eines Wortes w
- ε bezeichnet das leere Wort
- $\#_a(w)$ ist die Anzahl der Vorkommen des Buchstabens a im Wort w.

Alphabete, Wörter

Beispiele

- $\Sigma_1 = \{0, 1\}$.
  Wörter über Σ_1: ε, 0, 1, 01, 1001, 100101

- $\Sigma_2 = \{a, b, c, d, e, \ldots, x, y, z\}$.
  Wörter über Σ_2: ε, a, b, aba, $baab$, $informatik$

- $\Sigma_3 = \{(,), +, -, *, a\}$.
  Wörter über Σ_3: ε, a, $(a + a)$, $(a + a) * (a - a)$, $+ +a)$

Alphabete, Wörter

Beispiele

- $\Sigma_1 = \{0, 1\}$.
  Wörter über Σ_1: ε, 0, 1, 01, 1001, 100101
  $|\varepsilon| = 0$, $|0| = 1$, $|01| = 2$, $|100101| = 6$.

- $\Sigma_2 = \{a, b, c, d, e, \ldots, x, y, z\}$.
  Wörter über Σ_2: ε, *a*, *b*, *aba*, *baab*, *informatik*
  $|\varepsilon| = 0$, $|baab| = 4$, $|informatik| = 10$

- $\Sigma_3 = \{(,), +, -, *, a\}$.
  Wörter über Σ_3: ε, a, $(a + a)$, $(a + a) * (a - a)$, $+ + a)$
  $|\varepsilon| = 0$, $|(a + a) * (a - a)| = 11$, $| + + a| = 3$

Alphabete, Wörter

Beispiele

- $\Sigma_1 = \{0, 1\}$.
  Wörter über Σ_1: ε, 0, 1, 01, 1001, 100101
  $|\varepsilon| = 0$, $|0| = 1$, $|01| = 2$, $|100101| = 6$.
  $\#_0(\varepsilon) = 0, \#_1(\varepsilon) = 0$, $\#_0(100101) = \#_1(100101) = 3$.
- $\Sigma_2 = \{a, b, c, d, e, \dots, x, y, z\}$.
  Wörter über Σ_2: ε, a, b, aba, $baab$, $informatik$
  $|\varepsilon| = 0$, $|baab| = 4$, $|informatik| = 10$
  $\#_a(\varepsilon) = \dots \#_z(\varepsilon) = 0$, $\#_a(aba) = 2, \#_b(aba) = 1, \#_c(aba) = 0$.
- $\Sigma_3 = \{(,), +, -, *, a\}$.
  Wörter über Σ_3: ε, a, $(a + a)$, $(a + a) * (a - a)$, $++a)$
  $|\varepsilon| = 0$, $|(a + a) * (a - a)| = 11$, $|++a| = 3$
  $\#_a((a + a) * (a - a)) = 4$, $\#_+((a + a) * (a - a)) = 1$

Alphabete, Wörter

Operationen auf Wörtern

Verknüpfung (Konkatenation):

$w \circ w'$

assoziativ, oft geschrieben als ww'

i-te Potenz:

$w^0 = \varepsilon, \qquad w^{i+1} = ww^i$

Reverse:

$w^R =$ das Wort w rückwärts

Alphabete, Wörter

Beispiele

- $\Sigma_1 = \{0, 1\}$.
  Wörter über Σ_1: ε, 0, 1, 01, 1001, 100101

  $$\varepsilon \circ 01001 = 01001, \quad 01 \circ 1001 = 011001$$
  $$1001^0 = (1001)^0 = \varepsilon, \quad 1001^3 = (1001)^3 = 100110011001$$
  $$\varepsilon^R = \varepsilon, \quad 01101^R = (01101)^R = 10110$$

- $\Sigma_2 = \{a, b, c, d, e, \ldots, x, y, z\}$.
  Wörter über Σ_2: ε, a, b, aba, $baab$, $informatik$

  $$\varepsilon \circ \varepsilon = \varepsilon, \quad ab \circ \varepsilon = ab, \quad aba \circ baab = ababaab$$
  $$aba^2 = (aba)^2 = abaaba$$
  $$informatik^R = kitamrofni.$$

- $\Sigma_3 = \{(,), +, -, *, a\}$.
  Wörter über Σ_2: ε, a, $(a + a)$, $(a + a) * (a - a)$, $+ + a)$

Sprache

Definition (Sprache)

Eine Sprache L (über einem Alphabet Σ) ist eine Menge von Wörtern über Σ.

Alphabete, Wörter

Beispiele

- $\Sigma_1 = \{0, 1\}$.

  Gerade $=$ Menge aller Wörtern über Σ_1, die die binäre Representation einer geraden Zahl sind.

- $\Sigma_2 = \{a, b, c, d, e, \dots, x, y, z\}$.

  ENGLISH $= \{w \mid w$ wort über Σ_2, das auf English eine Bedeutung hat$\}$

- $\Sigma_3 = \{(,), +, -, *, a\}$.

  EXPR $=$ Menge der Wörter über Σ_2, die korrekt geklammerten Ausdrücke sind.

Operationen auf Sprachen

Konkatenation:

$$L \circ M = \{w \circ w' \mid w \in L,\ w' \in M\}$$

oft geschrieben als LM

***i*-te Potenz:**

$$L^0 = \{\varepsilon\}, \qquad L^{i+1} := LL^i$$

Reverse:

$$L^R = \{w^R \mid w \in L\}$$

Operationen auf Sprachen

Kleene-Hülle

$$L^* = L^0 \cup L^1 \cup L^2 \cup \dots$$

Variante:

$$L^+ = LL^* = L^1 \cup L^2 \cup \dots$$

Beispiele

Konkatenation:

$$L \circ M = \{w \circ w' \mid w \in L,\ w' \in M\}$$

oft geschrieben als LM

Beispiele:

1. $L = \{a, ab, bab\}, \quad M = \{\varepsilon, bc, bcc\}$

$$L \circ M = LM = \{a, ab, bab, abc, abbc, babbc, abcc, abbcc, babbcc\}$$

2. $L = \{\varepsilon\}, M = \{\varepsilon, bc, bcc\}$

$$L \circ M = LM = \{\varepsilon, bc, bcc\} = M$$

$$M \circ L = ML = \{\varepsilon, bc, bcc\} = M$$

3. $L = \emptyset$, M beliebig: $\quad L \circ M = M \circ L = \emptyset$

Beispiele

***i*-te Potenz:**

$$L^0 = \{\varepsilon\}, \qquad L^{i+1} := LL^i$$

Reverse:

$$L^R = \{w^R \mid w \in L\}$$

Beispiele:

1. $L = \{a, ab, bab\}$: $\qquad L^R = \{a, ba, bab\}$

   $L^0 = \{\varepsilon\}$
   $L^1 = L = \{a, ab, bab\}$
   $L^2 = LL = \{aa, aba, baba, aab, abab, babab, abab, abbab, babbab\}$

2. $L = \{\varepsilon\}$

   $L^0 = \{\varepsilon\}, L^1 = \{\varepsilon\}; \qquad L^n = \{\varepsilon\}$ für alle $n \in \mathbb{N}$.

3. $L = \emptyset$

   $L^0 = \{\varepsilon\}, \qquad L^n = \emptyset$ for all $n \in \mathbb{N}$ with $n \geq 1$.

Beispiele

Kleene-Hülle $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots = \bigcup_{n \geq 0} L^n$

$L^+ = LL^* = L^1 \cup L^2 \cup \ldots = \bigcup_{n \geq 1} L^n$

Beispiele:

1. $L = \{a\}$:

   $L^0 = \{\varepsilon\}, \quad L^1 = L = \{a\}, \quad L^2 = LL = \{aa\}, \quad L^n = \{a^n\}$ für $n \in \mathbb{N}$.

   $L^* = \{a^n \mid n \in \mathbb{N}\} = \{\varepsilon, a, a^2, a^3, \ldots\}$

   $L^+ = \{a^n \mid n \in \mathbb{N}, n \geq 1\} = \{a, a^2, a^3, \ldots\}$

2. $L = \{\varepsilon\}$

   $L^n = \{\varepsilon\}$ für alle $n \in \mathbb{N}$.

   $L^* = L^+ = \{\varepsilon\}$

3. $L = \emptyset$

   $L^0 = \{\varepsilon\}, \quad L^n = \emptyset$ for all $n \in \mathbb{N}$ with $n \geq 1$.

   $L^* = \{\varepsilon\}, \ L^+ = \emptyset$.

Beispiele

Kleene-Hülle	$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots = \bigcup_{n\geq 0} L^n$
	$L^+ = LL^* = L^1 \cup L^2 \cup \ldots = \bigcup_{n\geq 1} L^n$

Beispiele:

4. $L = \{a, b, c\}$:

$$L^0 = \{\varepsilon\}, \quad L^1 = L = \{a, b, c\},$$

$$L^2 = LL = \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$$

für $n \in \mathbb{N}$:

$$L^n = \{a_1 \ldots a_n \mid a_1, \ldots, a_n \in \{a, b, c\}\}$$

$$L^* = \{a_1 \ldots a_n \mid n \in \mathbb{N}\}$$

$$L^+ = \{a_1 \ldots a_n \mid n \in \mathbb{N}, n \geq 1\}$$

Operationen auf Sprachen

Kleene-Hülle

$$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$$

Variante:

$$L^+ = LL^* = L^1 \cup L^2 \cup \ldots$$

Σ^* bezeichnet die Menge aller Wörter über Σ

Operationen auf Sprachen

Kleene-Hülle

$$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$$

Variante:

$$L^+ = LL^* = L^1 \cup L^2 \cup \ldots$$

Σ^* bezeichnet die Menge aller Wörter über Σ

Genau genommen besteht ein Unterschied:
ein Buchstabe $\neq$ Wort, das nur aus dem einen Buchstaben besteht

Darum ist Σ selbst keine Sprache über Σ

(Oft wird über diesen Unterschied hinweggesehen)

Reguläre Ausdrücke

Definition (Reguläre Ausdrücke)

Menge $\mathfrak{Reg}_\Sigma$ der **regulären Ausdrücke** (über Σ) ist definiert durch:

1. 0 ist ein regulärer Ausdruck
2. Für jedes $a \in \Sigma$ ist a ein regulärer Ausdruck
3. Sind r und s reguläre Ausdrücke, so auch
   - $(r + s)$ (Vereinigung),
   - (rs) (Konkatenation),
   - (r^*) (Kleene Stern)

Klammern können weggelassen werden, dann

- * hat Vorrang vor Konkatenation
- Konkatenation hat Vorrang vor $+$

Reguläre Ausdrücke

Definition (Semantik regulärer Ausdrücke)

Ein regulärer Ausdruck r stellt eine Sprache $\mathfrak{I}(r)$ über Σ wie folgt dar:

$$
\begin{array}{rcll}
\mathfrak{I}(0) & := & \emptyset & \\
\mathfrak{I}(a) & := & \{a\} & \text{für } a \in \Sigma \\
\mathfrak{I}(r+s) & := & \mathfrak{I}(r) \cup \mathfrak{I}(s) & \\
\mathfrak{I}(rs) & := & \mathfrak{I}(r)\mathfrak{I}(s) & \\
\mathfrak{I}(r^*) & := & \mathfrak{I}(r)^* &
\end{array}
$$

Wir benutzen auch das Makro . . .

$$1 := 0^*$$

Es gilt: $\mathfrak{I}(1) = \{\varepsilon\}$

Reguläre Ausdrücke

Übung

Welche Sprachen werden durch die folgenden regulären Ausdrücke dargestellt?

- aa
- $(a + b)^*$
- $aa^* + bb^*$

Reguläre Ausdrücke

Übung

Welche Sprachen werden durch die folgenden regulären Ausdrücke dargestellt?

- aa
- $(a+b)^*$
- $aa^* + bb^*$

$\mathfrak{I}(aa) = \mathfrak{I}(a)\mathfrak{I}(a) = \{a\} \circ \{a\} = \{aa\}$

$\mathfrak{I}((a+b)^*) = \mathfrak{I}((a+b))^* = (\mathfrak{I}(a) \cup \mathfrak{I}(b))^* = (\{a\} \cup \{b\})^* = \{a, b\}^*$

$\mathfrak{I}((aa^* + bb^*)) = \mathfrak{I}(aa^*) \cup \mathfrak{I}(bb^*)) = \{a\}\{a\}^* \cup \{b\}\{b\}^* = \{a\}^+ \cup \{b\}^+$

Reguläre Ausdrücke

Übung

Geben Sie einen regulären Ausdruck über $\Sigma = \{a, b, c\}$ an, der die Sprache darstellt, die genau die Wörter enthält, die mit b beginnen.

Reguläre Ausdrücke

Übung

Geben Sie einen regulären Ausdruck über $\Sigma = \{a, b, c\}$ an, der die Sprache darstellt, die genau die Wörter enthält, die mit b beginnen.

Antwort: $b(a + b + c)^*$

Reguläre Ausdrücke als Suchmuster für `grep`

Das Kommando `grep` (bzw. `egrep`)

- Sucht Wörter (Strings) in Dateien
- Benutzt reguläre Ausdrücke als Suchmuster
- Sehr schnell
- Volle Funktionalität mit `egrep` (Unix/Linux)

Reguläre Ausdrücke als Suchmuster für `grep`

Das Kommando `grep` (bzw. `egrep`)

- Sucht Wörter (Strings) in Dateien
- Benutzt reguläre Ausdrücke als Suchmuster
- Sehr schnell
- Volle Funktionalität mit egrep (Unix/Linux)

grep: global/regular expression/print

global search for a regular expression and print out matched lines

Reguläre Ausdrücke als Suchmuster für `grep`

Syntax bei `grep`

`grep`	Regulärer Ausdruck
ww'	ww'
$w \,\vert\, w'$	$w + w'$
w`*`	w^*
w`+`	w^+

Reguläre Ausdrücke als Suchmuster für `grep`

Syntax bei `grep`

`grep`	Regulärer Ausdruck
ww'	ww'
$w \mid w'$	$w + w'$
w`*`	w^*
w`+`	w^+

Syntactic Sugar

`grep`	Regulärer Ausdruck
`[abc]`	$a + b + c$
`[a-d]`	$a + b + c + d$
`.`	beliebiges Zeichen aus Σ