Grundlagen der theoretischen Informatik

Kurt Sieber

Fakultät IV, Department ETI
Universität Siegen

SS 2013

Vorlesung vom 09.04.2013

Inhalt der Vorlesung

Teil I: Automaten und formale Sprachen (Kurt Sieber)

Zentrale Fragestellungen

- Wie definiert man (formal) die Syntax einer Sprache (z.B. einer Programmiersprache)?
- Wie überprüft man, ob ein Wort (d.h. eine Zeichenreihe) zur Sprache gehört?

Teil II: Berechenbarkeit und Komplexität (Hannes Diener)

Zentrale Fragestellungen

- Welche Probleme (Aufgabenstellungen) sind prinzipiell mit einem Computer lösbar bzw. nicht lösbar?
- Welche Probleme sind mit vernünftigem Aufwand (an Speicherplatz und Laufzeit) lösbar?

Zeichen, Wörter, Sprachen

Grundlagen für beide Teile: Zeichen, Wörter und Sprachen

Definition 1.1 *Ein* Alphabet *oder* Zeichenvorrat *ist eine nichtleere endliche Menge. Die Elemente eines Alphabets nennen wir* Zeichen *oder* Symbole*.*

Konvention: Alphabete werden mit $\Sigma, \Sigma_1, \Sigma', \ldots$ bezeichnet.

Beispiele

- $\Sigma_1 = \{1\}$
- $\Sigma_2 = \{0, 1\}$
- $\Sigma_3 = \{0, \ldots, 9\}$
- $\Sigma_4 = \{a, \ldots, z, A, \ldots, Z\}$
- $\Sigma_5 =$ Menge aller ASCII-Zeichen
- $\Sigma_6 = \{\mathit{begin}, \mathit{end}, \mathit{if}, \mathit{then}, \mathit{else}, ;, :=, +, *, \ldots\}$

Zeichen, Wörter, Sprachen

Definition 1.2 *Ein Wort über dem Alphabet Σ ist eine endliche Folge $w = (a_1, \ldots, a_n)$, wobei $n \geq 0$ und $a_i \in \Sigma$ für $i = 1, \ldots, n$. n heißt Länge des Wortes w, a_i heißt i-tes Zeichen von w. Das Wort $()$ heißt leeres Wort.*

Kurzschreibweise

- $a_1 \ldots a_n$ statt $(a_1, \ldots, a_n)$
  Vorsicht: Zeichen a und Wort a nicht mehr unterscheidbar!
- ε statt $()$

Weitere Schreibweisen

- $|w|$ = Länge von w
- Σ^* = Menge aller Wörter über Σ
- $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$ = Menge aller nichtleeren Wörter über Σ

Beispiele

- $\{1\}^* = \{\varepsilon, 1, 11, 111, \ldots\}$
- $\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \ldots\}$
- $\varepsilon, 0, 123, 0815 \in \{0, \ldots, 9\}^*$
- Der Inhalt einer Datei ist *ein* Wort über dem ASCII-Alphabet (denn Leerzeichen, Zeilenwechsel etc. sind Zeichen dieses Alphabets).
- Ein Programm (in irgendeiner Programmiersprache) ist ebenfalls ein Wort über dem ASCII-Alphabet.
- Der Inhalt eines Buches ist ein Wort (über dem Alphabet der Sprache, in der es geschrieben ist).

Alternative Definition für Σ^* und Σ^+

- $\Sigma^n = \{(a_1, \ldots, a_n) \mid a_i \in \Sigma \text{ für } i = 1, \ldots, n\}$
  $= \{a_1 \ldots a_n \mid a_i \in \Sigma \text{ für } i = 1, \ldots, n\}$
  $=$ Menge aller *Wörter der Länge n* über Σ

- Speziell: $\Sigma^0 = \{()\}$
  $= \{\varepsilon\}$

- $\Sigma^* = \bigcup_{n \geq 0} \Sigma^n$
  $=$ Menge *aller Wörter* über Σ

  $\Sigma^+ = \bigcup_{n \geq 1} \Sigma^n$
  $= \Sigma^* \setminus \{\varepsilon\}$

Definition 1.3 *Seien* $v = a_1 \dots a_m$ *und* $w = b_1 \dots b_n$ *Wörter über* Σ. *Dann ist die* Konkatenation $v \circ w$ *von* v *und* w *definiert durch*

$$v \circ w \;=\; a_1 \dots a_m b_1 \dots b_n$$

Kurzschreibweise: vw statt $v \circ w$

Eigenschaften der Konkatenation

- *Assoziativität*: $(uv)w = u(vw)$ für alle $u, v, w \in \Sigma^*$
- *Neutrales Element*: $w\varepsilon = \varepsilon w = w$ für alle $w \in \Sigma^*$
- Also: $(\Sigma^*, \circ)$ ist ein *Monoid* mit neutralem Element ε (vgl. natürliche Zahlen mit Multiplikation)

Definition 1.4 *Sei $w \in \Sigma^*$ und $n \in \mathbb{N}$. Dann ist die n-te Potenz w^n von w definiert durch $w^n = \underbrace{w \ldots w}_{n}$ (speziell: $a^n = \underbrace{a \ldots a}_{n}$ falls $a \in \Sigma$).*

Besser: Definition von w^n durch Induktion über n

- $w^0 = \varepsilon$
- $w^n = ww^{n-1}$ falls $n > 0$

Oder:

- $w^0 = \varepsilon$
- $w^n = w^{n-1}w$ falls $n > 0$

Beispiele

- $(bla)^2 = blabla$
- $(bla\ )^3 = bla\ bla\ bla$

Definition 1.5 *Für* $w = a_1 \dots a_n \in \Sigma^*$ *sei* $w^R \in \Sigma^*$ *das Wort, das durch "Spiegelung" aus* w *entsteht , d.h.*

$$w^R = a_n \dots a_1$$

(R *steht für engl. reverse)*

Beispiel

$(01001)^R = 10010$

Alternative

Definition von w^R durch Induktion über $|w|$
(s. Übung!)

Definition 1.6 *Sei $x \in \Sigma^*$.*

- *u heißt Anfangswort oder Präfix von x, falls ein $v \in \Sigma^*$ existiert mit $x = uv$.*
- *v heißt Endwort oder Suffix von x, falls ein $u \in \Sigma^*$ existiert mit $x = uv$.*
- *v heißt Teilwort von x, falls $u, w \in \Sigma^*$ existieren mit $x = uvw$.*

Beispiel

- Präfixe von 010: $\varepsilon, 0, 01, 010$
- Suffixe von 010: $\varepsilon, 0, 10, 010$
- Teilwörter von 010: $\varepsilon, 0, 1, 01, 10, 010$

Zeichen, Wörter, Sprachen

Die Beziehungen ‘Anfangswort’, ‘Endwort’ und ‘Teilwort’ sind zweistellige Relationen auf Σ^*. Jede dieser Relationen ist

- *reflexiv*: Jedes $w \in \Sigma^*$ steht in Relation zu sich selbst.
- *transitiv*: Wenn u in Relation zu v und v in Relation zu w steht, dann steht auch u in Relation zu w.
- *antisymmetrisch*: Wenn u in Relation zu v steht und v in Relation zu u, dann ist $v = u$.

Also: Jede der Relationen definiert eine *partielle Ordnung* auf Σ^*.

Definition 1.7 *Eine Sprache (oder formale Sprache) über dem Alphabet Σ ist eine (beliebige) Teilmenge von Σ^*.*

Konvention:

Sprachen bezeichnen wir meist mit $L, L_1, L', \ldots$ (engl. *language*).

Alternative Formulierung

Eine Sprache über Σ ist ein Element der *Potenzmenge* $\wp(\Sigma^*)$.

Beispiele

- $L_1 = \emptyset$
- $L_2 = \Sigma^*$
- $L_3 = \{\varepsilon\}$
- $L_4 = \{1^n \mid n \text{ ist eine Primzahl}\}$
- $L_5 =$ Menge aller Binärdarstellungen natürlicher Zahlen
- $L_6 =$ Menge aller Dezimaldarstellungen natürlicher Zahlen
- $L_7 =$ Menge aller Java-Programme
- $L_8 =$ Menge aller Sätze der deutschen Sprache
- $L_9 = \{0^n 1^n \mid n \geq 0\}$
- $L_{10} = \{vw \mid v \in \{0\}^*, w \in \{1\}^*\}$

Definition 1.8 *Seien* $L, L_1, L_2 \subseteq \Sigma^*$. *Dann sei*

- $L_1 \cup L_2$ *die* Vereinigung *von* L_1 *und* L_2.
- $L_1 \cap L_2$ *der* Durchschnitt *von* L_1 *und* L_2.
- $L_1 \setminus L_2 = \{w \in L_1 \mid w \notin L_2\}$ *die* Mengendifferenz *von* L_1 *und* L_2.
- $\bar{L} = \Sigma^* \setminus L$ *das* Komplement *von* L.
- $L_1 \circ L_2 = \{vw \mid v \in L_1, w \in L_2\}$ *die* Konkatenation *von* L_1 *und* L_2.

Kurzschreibweise: $L_1 L_2$ statt $L_1 \circ L_2$

Eigenschaften der Konkatenation

- *Assoziativität*: $(L_1 L_2)L_3 = L_1(L_2 L_3)$ für alle $L_1, L_2, L_3 \subseteq \Sigma^*$
- *Neutrales Element*: $L\{\varepsilon\} = \{\varepsilon\}L = L$ für alle $L \subseteq \Sigma^*$
- Also: $(\wp(\Sigma^*), \circ)$ ist ein *Monoid* mit neutralem Element $\{\varepsilon\}$.

Definition 1.9 *Seien* $L \subseteq \Sigma^*$ *und* $n \in \mathbb{N}$. *Dann ist die* n-te Potenz L^n *von* L *definiert durch*

$$\begin{aligned} L^n &= \underbrace{L \circ \ldots \circ L}_{n} \\ &= \underbrace{L \ldots L}_{n} \\ &= \{w_1 \ldots w_n \mid w_i \in L \text{ für } i = 1, \ldots, n\} \end{aligned}$$

Besser: Definition von L^n durch Induktion über n

- $L^0 = \{\varepsilon\}$
- $L^n = LL^{n-1}$ falls $n > 0$

Oder:

- $L^0 = \{\varepsilon\}$
- $L^n = L^{n-1}L$ falls $n > 0$

Definition 1.10 *Für $L \subseteq \Sigma^*$ sei*

- $L^R = \{w^R \mid w \in L\}$ *die "Spiegelung" von* L
- $L^* = \bigcup_{n \geq 0} L^n$

  $= \{w_1 \dots w_n \mid n \geq 0, w_i \in L \text{ für } i = 1, \dots, n\}$

  der Kleene-Abschluss von L.
- *Speziell:*

  $\{w\}^* = \{w^n \mid n \geq 0\}$ *falls* $w \in \Sigma^*$

  $\{a\}^* = \{a^n \mid n \geq 0\}$ *falls* $a \in \Sigma$
- $L^+ = \bigcup_{n \geq 1} L^n$

  $= \{w_1 \dots w_n \mid n \geq 1, w_i \in L \text{ für } i = 1, \dots, n\}$

  $= LL^*$

Folgerung: $L^* = L^+ \cup L^0 = L^+ \cup \{\varepsilon\}$

Rechenregeln für Potenzen

. . . von Wörtern

- $w^0 = \varepsilon$ (neutrales Element)
- $w^1 = w$
- $w^m w^n = w^{m+n}$
- $(w^m)^n = w^{mn}$

. . . von Sprachen

- $L^0 = \{\varepsilon\}$ (neutrales Element)
- $L^1 = L$
- $L^m L^n = L^{m+n}$
- $(L^m)^n = L^{mn}$

Diese Regeln gelten in *jedem* Monoid (wenn man das Potenzieren nach dem üblichen Schema definiert). Denn sie folgen allein aus der Assoziativität und der Eigenschaft des neutralen Elements.

Warnung: Es gilt *nicht* $(w_1 w_2)^n = w_1^n w_2^n$ oder $(L_1 L_2)^n = L_1^n L_2^n$. Denn dazu bräuchte man die Kommutativität.

Rechenregeln für * und $^+$

- $L \subseteq L^*$
- $(L^*)^* = L^*$
- $L \subseteq L^+$
- $(L^+)^+ = L^+$

Mengenoperatoren mit diesen beiden Eigenschaften nennt man *Abschluss-* oder *Hüllenoperatoren*.

Beweis für *

1. $L \subseteq L^*$ ist klar wegen $L = L^1$ und $L^* = \bigcup_{n \geq 0} L^n$.
2. $L^* = (L^*)^*$:
   '$\subseteq$' ist klar wegen 1.
   '$\supseteq$': Sei $w \in (L^*)^*$, d.h. $w = w_1 \dots w_n$ mit $n \geq 0$ und $w_i \in L^*$ für $i = 1, \dots, n$. Dann existiert für jedes i ein $m_i \geq 0$ mit $w_i \in L^{m_i}$, also folgt $w = w_1 \dots w_n \in L^{m_1} \dots L^{m_n} = L^{m_1 + \dots + m_n} \subseteq L^*$. $\square$

Zeichen, Wörter, Sprachen

Die Operatoren $\cup, \circ, ^*, \ldots$ lassen sich auch zur *Beschreibung* von Sprachen verwenden. Konvention: Die Postfixoperatoren ^{n}, $^+$ und * binden am stärksten, $\circ$ bindet stärker als $\cup$.

Beispiele

- Die Menge aller Dezimaldarstellungen natürlicher Zahlen:
  $L_{nat} = \{0, \ldots, 9\}^+ = \{0, \ldots, 9\}\{0, \ldots, 9\}^*$
- Die Menge aller Dezimaldarstellungen ganzer Zahlen:
  $L_{int} = \{-, \varepsilon\} L_{nat}$
- Die Menge aller dezimalen Festpunktzahlen:
  $L_{fix} = L_{int}\{.\} \cup \{.\}L_{nat} \cup L_{int}\{.\}L_{nat}$
- Die Menge aller dezimalen Fließpunktzahlen:
  $L_{float} = L_{int}\{e, E\}L_{int} \cup L_{fix}(\{\varepsilon\} \cup \{e, E\}L_{int})$

Zeichen, Wörter, Sprachen

Die Menge aller Wörter über dem Alphabet $\{0,1\}$, ...

- ... die mit 00 beginnen:
  $L_1 = \{00\}\{0,1\}^*$
- ... die 00 als Teilwort enthalten:
  $L_2 = \{0,1\}^*\{00\}\{0,1\}^*$
- ... die mindestens zwei Nullen enthalten:
  $L_3 = \{0,1\}^*\{0\}\{0,1\}^*\{0\}\{0,1\}^* = \{1\}^*\{0\}\{1\}^*\{0\}\{0,1\}^*$
- ... die genau zwei Nullen enthalten:
  $L_4 = \{1\}^*\{0\}\{1\}^*\{0\}\{1\}^*$
- ... die höchstens zwei Nullen enthalten:
  $L_5 = \{1\}^*\{0,\varepsilon\}\{1\}^*\{0,\varepsilon\}\{1\}^*$
- ... deren erstes und letztes Zeichen übereinstimmen:
  $L_6 = \{0,1\} \cup \{0\}\{0,1\}^*\{0\} \cup \{1\}\{0,1\}^*\{1\}$

Reguläre Sprachen

Definition 2.1 *Die Menge aller regulären Sprachen über Σ ist induktiv definiert durch:*

- *Jede endliche Menge $L \subseteq \Sigma^*$ ist regulär.*
- *Wenn $L_1, L_2 \subseteq \Sigma^*$ regulär sind, dann sind auch $L_1 \cup L_2$ und $L_1 \circ L_2$ regulär.*
- *Wenn L regulär ist, dann ist auch L^* regulär.*

Mit anderen Worten:

Eine Sprache ist genau dann regulär, wenn sie sich durch wiederholte Anwendung der Operatoren $\cup, \circ$ und * aus endlichen Sprachen aufbauen lässt.

Damit haben wir eine *endliche Beschreibung* für jede reguläre Sprache.

Beispiele: $L_{nat}, L_{int}, L_{fix}, L_{float}, L_1, \ldots, L_6$ sind reguläre Sprachen.

Reguläre Sprachen

Die 'Mengenausdrücke' aus Definition 2.1 sind gut lesbare Beschreibungen für reguläre Sprachen (zumindest für die Beispiele aus der Praxis wie Zahldarstellungen, Variablennamen, ...)

Aber:

- Nachteil in der Praxis:
  Ein Mengenausdruck für eine Sprache liefert noch keinen Algorithmus, um die Zugehörigkeit zur Sprache zu testen.
- Nachteil in der Theorie:
  Mit Definition 2.1 kann man nur schwer zeigen, dass eine Sprache *nicht* regulär ist.

Deshalb:

- Alternative Charakterisierungen der regulären Sprachen
- Insbesondere *algorithmische* Charakterisierungen

Endliche Automaten

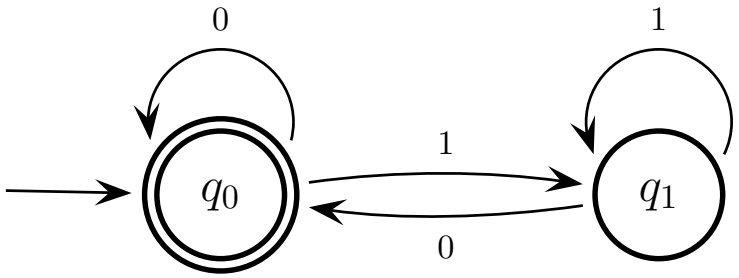
Ein *endlicher Automat* ist eine 'Maschine', die (nur) *endlich viele Zustände* annehmen kann. Einer der Zustände heißt *Startzustand*, einige Zustände heißen *Endzustände* (besser: *akzeptierende Zustände*).

Arbeitsweise

- Der Automat erhält ein Wort w als Eingabe.
- Zu Beginn befindet er sich im Startzustand.
- Er liest w zeichenweise von links nach rechts, wobei jedes Zeichen einen Zustandsübergang bewirkt.
- Ist er nach Abarbeitung von w in einem Endzustand, so wird w akzeptiert, andernfalls wird w abgelehnt.

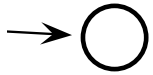
Ein endlicher Automat kann also dazu benutzt werden, die Zugehörigkeit zu einer Sprache zu testen.

Graphische Darstellung eines endlichen Automaten



Konvention:


sind die Zustände


ist der Startzustand


sind die Endzustände

- **Akzeptierte Wörter:** 0, 010, 01100
- **Abgelehnte Wörter:** 1, 101, 11011