Automaten und Formale Sprachen Tutorium

Teil I.

Christopher Blöcker (inf8871)

FH Wedel, SS 2010

JFLAP

Programm zur Simulation von Automaten etc., u.a.

- Endliche Automaten (+ Reguläre Ausdrücke)
- Kellerautomaten
- Turingmaschinen
- Grammatiken
- ...

Download

Google $\rightarrow$ "JFLAP download".

Start

Linux : java -jar /pfad/JFLAP.jar &

Windows : durch Anklicken (?)

Sprachen

Frage

Warum interessieren wir uns für Formale Sprachen?

Antwort

Die Theorie der Formalen Sprachen bildet die Grundlage für Programmiersprachen.
Um zu entscheiden, ob ein gegebenes Programm syntaktisch korrekt ist, benötigen wir die Beschreibung einer (Programmier-)Sprache.
Mit Hilfe von Automaten kann dann entschieden werden, ob ein Programm der jeweiligen Sprachdefinition genügt.

Sprachen

Definitionen

Frage

Was ist eine Formale Sprache?

Definition

Eine Formale Sprache ist eine Menge von Wörtern über einem Alphabet Σ.

Anmerkung

Sprachen werden allgemein mit $\mathcal{L}$ bezeichnet.
Für jede Sprache gilt $\mathcal{L} \subseteq \Sigma^*$,
wobei Σ^* die Menge aller Wörter bezeichnet, die aus den gegebenen Buchstaben gebildet werden können.

Sprachen

Definitionen

Definition

Ein Alphabet Σ ist eine Menge von Zeichen, die Zeichen des Alphabets werden auch als Buchstaben bezeichnet.

Definition

Ein Wort über einem Alphabet Σ ist eine endliche Folge von Zeichen aus Σ.

Anmerkung

Das Wort, das aus 0 Zeichen besteht, wird als das Leere Wort ϵ bezeichnet.

Sprachen

Beispiele

Beispiele

Sei $\Sigma = \{a, b, c\}$.

- Σ^1 bezeichnet die Menge aller Wörter der Länge 1, also $\Sigma^1 \leftarrow \{a, b, c\}$
- Σ^2 bezeichnet die Menge aller Wörter der Länge 2, also $\Sigma^2 \leftarrow \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$
- Σ^k bezeichnet die Menge aller Wörter der Länge k, $k \in \mathbb{N}$
- Σ^0 bezeichnet die Menge aller Wörter der Länge 0, also $\Sigma^0 \leftarrow \{\epsilon\}$
- Σ^+ bezeichnet die Menge aller nicht-leeren Wörter mit $\Sigma^+ \leftarrow \bigcup_{k \in \mathbb{N}^+} \Sigma^k$
- Σ^* bezeichnet die Menge aller Wörter mit $\Sigma^* \leftarrow \Sigma^0 \cup \bigcup_{k \in \mathbb{N}^+} \Sigma^k$

Sprachen

Konkatenation

Konkatenation

Zweistellige Operation in Σ^*.

$$\circ : \Sigma^* \times \Sigma^* \to \Sigma^*$$

$(\Sigma^*, \circ, \epsilon)$ ist ein Monoid.

Beispiel

Sei

$$x = \text{abc}, y = \text{def}.$$

Dann ist

$$x \circ y = \text{abcdef}.$$

Exkurs

Gruppentheorie

Definition

Ein Monoid ist ein Tripel $(\mathcal{M}, *, e)$ mit

1. Die Verknüpfung $*$ ist assoziativ

   $$a * (b * c) = (a * b) * c = a * b * c$$

   mit $a, b, c \in \mathcal{M}$
2. e ist neutrales Element bezüglich $*$

   $$a * e = e * a = a$$

   mit $a \in \mathcal{M}$

Ein Monoid ist eine Halbgruppe mit ausgezeichnetem neutralen Element.

Beschreibung von Sprachen

Frage

Wie lassen sich Sprachen beschreiben?

Antwort

- Explizite Aufzählung

$$\mathcal{L} = \{a, ab, abc\}$$

- Mengendefinition

$$\mathcal{L} = \{w \in \Sigma^* \mid w \text{ endet mit a}\}$$

- Angabe eines akzeptierenden Automaten
- Angabe eines Regulären Ausdrucks

Deterministischer Endlicher Automat

Definition

Definition

Ein deterministischer Endlicher Automat ist ein 5-Tupel.

$$\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$$

mit

$\mathcal{Q}$	endliche, nichtleere Menge von Zuständen
Σ	endliche, nichtleere Menge, das Eingabealphabet mit $\Sigma \cap \mathcal{Q} = \emptyset$
δ	Überführungsfunktion mit $\delta : \mathcal{Q} \times \Sigma \to \mathcal{Q}$
q_0	Startzustand
$\mathcal{F}$	Menge von Endzuständen mit $\mathcal{F} \subseteq \mathcal{Q}$

Deterministischer Endlicher Automat

Zustandmenge $\mathcal{Q}$

$\mathcal{Q}$

- endlich
  Ein Endlicher Automat darf nur aus endlich vielen Zuständen bestehen. Er hat keinen Speicher, lediglich über den aktuellen Zustand kann er sich merken, was zuletzt (bzw. bisher) geschehen ist.
  Wäre die Menge der Zustände nicht endlich, so könnte kein Computer den Automaten simulieren.
- nicht leer
  Sonst kann er nichts machen.

Deterministischer Endlicher Automat

Eingabealphabet Σ

Σ

- endlich
  Wäre das Eingabealphabet nicht endlich, so wäre auch die Überführungsrelation (bezogen auf den zu ihrer Darstellung benötigten Speicherplatz) unendlich groß und kein Computer könnte den Automaten simulieren.
- nicht leer
  Sonst könnte die Sprache nur ϵ enthalten und wäre uninteressant.
- $\Sigma \cap \mathcal{Q} = \emptyset$
  Ein Zustand kann nicht gleichzeitig Eingabezeichen sein.

Deterministischer Endlicher Automat

Überführungsfunktion δ, Startzustand q_0

δ

$$\delta : \mathcal{Q} \times \Sigma \rightarrow \mathcal{Q}$$

1. Der Automat befindet sich stets in einem Zustand
2. Er liest ein Zeichen der Eingabe
3. Die Überführungsfunktion δ berechnet den daraus resultierenden Nachfolgezustand

q_o

Der Startzustand des Automaten wird mit q_0 bezeichnet.
Er kann auch einen beliebigen anderen Namen erhalten und muss gekennzeichnet werden.

Deterministischer Endlicher Automat

Endzustandsmenge $\mathcal{F}$, Akzeptierte Sprache

$\mathcal{F}$

$$\mathcal{F} \subseteq \mathcal{Q}$$

Die Endzustände haben folgende Bedeutung:
Wenn sich der Automat nach Verarbeiten der gesamten Eingabe (also des gesamten Eingabewortes w) in einem Zustand aus $\mathcal{F}$ befindet, dann gehört w zu der vom Automaten akzeptierten Sprache.

Akzeptierte Sprache

Die von $\mathcal{A}$ akzeptierte Sprache $\mathcal{L}_{\mathcal{A}}$ ist

$$\mathcal{L}_{\mathcal{A}} = \{w \in \Sigma^* \mid \begin{array}{l}\mathcal{A} \text{ befindet sich nach der Berechnung} \\ \text{in einem Endzustand.}\}\end{array}$$

Exkurs

Mengenlehre

Frage

Wann ist eine Menge endlich?

$$|\mathcal{M}| \overset{?}{<} \infty$$

Antwort

Eine Menge ist genau dann endlich, wenn sie keine zu sich selbst gleichmächtige, echte Teilmenge enthält.

$$|\mathcal{M}| < \infty \Leftrightarrow \neg \bigvee_{\mathcal{N} \subsetneq \mathcal{M}} |\mathcal{N}| = |\mathcal{M}|$$

Deterministischer Endlicher Automat

Erweiterte Überführungsfunktion, Akzeptierte Sprache

Erweiterte Überführungsfunktion

$$\delta^* : \mathcal{Q} \times \Sigma^* \to \mathcal{Q}$$

mit

$$\delta^*(q_i, w) \leftarrow \begin{cases} q_i & \text{für } w = \epsilon \\ \delta(\delta^*(q_i, u), a) & \text{für } w = u \cdot a \end{cases}$$

Akzeptierte Sprache

Sei $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$ ein Endlicher Automat und δ^* die zu $\mathcal{A}$ gehörende erweiterte Überführungsfunktion.
Die von $\mathcal{A}$ akzeptierte Sprache $\mathcal{L}_\mathcal{A}$ ist

$$\mathcal{L}_\mathcal{A} = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in \mathcal{F}\}.$$

Nichtdeterministischer Endlicher Automat

Definition

Definition

Ein nichtdeterministischer Endlicher Automat ist ein 5-Tupel.

$$\mathcal{A} = (\mathcal{Q}, \Sigma_\epsilon, \delta, q_0, \mathcal{F})$$

mit

$\mathcal{Q}$	endliche, nichtleere Menge von Zuständen
Σ_ϵ	endliche, nichtleere Menge, das Eingabealphabet mit $\Sigma_\epsilon \cap \mathcal{Q} = \emptyset$
δ	Überführungsrelation mit $\delta \subseteq \mathcal{Q} \times \Sigma_\epsilon \times \mathcal{Q}$
q_0	Startzustand
$\mathcal{F}$	Menge von Endzuständen mit $\mathcal{F} \subseteq \mathcal{Q}$

Nichtdeterministischer Endlicher Automat

Unterschied zum deterministischen Endlichen Automaten

Σ_ϵ

Das Eingabealphabet enthält nun auch das leere Wort

$$\Sigma_\epsilon \leftarrow \Sigma \cup \{\epsilon\}.$$

Der Automat kann einen Zustandsübergang durchführen, ohne ein Eingabezeichen zu lesen (bzw. indem er ϵ liest).

δ

Ein Zustand kann nun mehrere Folgezustände haben

$$\delta \subseteq \mathcal{Q} \times \Sigma_\epsilon \times \mathcal{Q}$$

bzw.

$$\delta : \mathcal{Q} \times \Sigma_\epsilon \rightarrow \mathcal{P}(\mathcal{Q}).$$

Reguläre Ausdrücke

Regulärer Ausdruck

Ein Regulärer Ausdruck beschreibt eine Reguläre Menge. Jede Reguläre Menge kann als Reguläre Sprache aufgefasst werden.

Reguläre Ausdrücke über einem Alphabet Σ sind

- ϵ
- $\emptyset$
- $\mathbf{a} \Leftrightarrow a \in \Sigma$
- $\mathcal{R} \leftarrow \mathcal{R}_1 \circ \mathcal{R}_2$
- $\mathcal{R} \leftarrow \mathcal{R}_1 + \mathcal{R}_2$
- $\mathcal{R} \leftarrow \mathcal{R}^+$
- $\mathcal{R} \leftarrow \mathcal{R}^* = \mathcal{R}^+ + \epsilon$

Beispiele für Reguläre Ausdrücke mit $\Sigma = \{0, 1\}$

Regulärer Ausdruck	Erzeugte Sprache
$\mathcal{R}_1 = \mathbf{1}$	$\mathcal{L}(\mathcal{R}_1) = \{1\}$
$\mathcal{R}_2 = \mathbf{0} + \mathbf{1}$	$\mathcal{L}(\mathcal{R}_2) = \{0, 1\}$
$\mathcal{R}_3 = \mathbf{1} \cdot \mathbf{0}\ (= \mathbf{10})$	$\mathcal{L}(\mathcal{R}_3) = \{10\}$
$\mathcal{R}_4 = \mathbf{1} \cdot (\mathbf{0} + \mathbf{1})$	$\mathcal{L}(\mathcal{R}_4) = \{10, 11\}$
$\mathcal{R}_5 = \mathbf{0}^+ \cdot \mathbf{1}$	$\mathcal{L}(\mathcal{R}_5) = \{01, 001, 0001, 00001, ...\}$
$\mathcal{R}_6 = \mathbf{1}^*$	$\mathcal{L}(\mathcal{R}_6) = \{\epsilon, 1, 11, 111, 1111, ...\}$
$\mathcal{R}_7 = \mathcal{R}_2 \cdot \mathcal{R}_4$	$\mathcal{L}(\mathcal{R}_7) = \{010, 011, 110, 111\}$
$\mathcal{R}_8 = \mathcal{R}_2 + \epsilon$	$\mathcal{L}(\mathcal{R}_8) = \{\epsilon, 0, 1\}$
$\mathcal{R}_9 = \mathcal{R}_7 \cdot \emptyset$	$\mathcal{L}(\mathcal{R}_9) = \emptyset$
$\mathcal{R}_{10} = \mathcal{R}_2^+ + \mathcal{R}_8$	$\mathcal{L}(\mathcal{R}_{10}) = \{\epsilon, 0, 1, 00, 01, 10, 11, ...\}$