Formale Methoden 1

Gerhard Jäger
Gerhard.Jaeger@uni-bielefeld.de

Uni Bielefeld, WS 2007/2008

16. Januar 2008

Reguläre Ausdrücke

- vierte Art (neben Typ-3-Grammatiken, deterministischen und nicht-deterministischen endlichen Automaten), reguläre Sprachen zu beschreiben
- sehr nützlich für Suche in Texten (siehe Computer-Propädeutikum)
- daher wichtige Technik bei Korpus-Untersuchungen
- viele Software-Pakete enthalten Implementierungen von regulären Ausdrücken, z.B.
  - Emacs
  - Word
  - OpenOffice
  - Perl
  - Python
  - Unix-Tools wie grep/egrep oder sed
- Syntax kann sich im Detail unterscheiden

Reguläre Audrücke

Definition (Syntax von regulären Ausdrücken)

Sein Σ ein endliches Alphabet.

- $\emptyset$ ist ein regulärer Ausdruck.
- ϵ ist ein regulärer Ausdruck.
- Für jedes $a \in \Sigma$: a ist ein regulärer Ausdruck.
- Wenn α und β reguläre Ausdrücke sind, dann sind auch
  - $\alpha\beta$,
  - $(\alpha|\beta)$, und
  - α^*

  reguläre Ausdrücke.

In praktischen Implementierungen wird die Syntax meist stark erweitert mit Ausdrücken für Wort- und Zeilenanfang/-ende, Klassen von Einzelzeichen, endlich viele Iterationen usw.

Reguläre Ausdrücke

Regulären Ausdrücken werden rekursiv formale Sprachen über Σ^* zugewiesen. Dabei müssen zwei Operationen über formale Sprachen definiert werden, die **Verkettung** und die **Iteration**.

Die Verkettung zweier formaler Sprachen

Definition

Seien L_1 und L_2 zwei formale Sprachen. Die *Verkettung* $L_1 \frown L_2$ von L_1 und L_2 ist definiert als

$$L_1 \frown L_2 = \{x \frown y | x \in L_1, y \in L_2\}$$

Die Verkettung zweier formaler Sprachen

Beispiel:

- $L_1 = \{a^n b^n | n > 1\}$
- $L_2 = \{c^m | m \geq 0\}$
- $L_1 \frown L_2 =$
  $\{aabb, aabbc, aabbcc, aabbccc, aabbcccc, aaabbbc, ...\} =$
  $\{a^n b^n c^m | n > 1, m \geq 0\}$

- Schreibkonvention:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^1 &= 1 \\ L^2 &= L \frown L \\ L^{n+1} &= L^n \frown L \end{aligned}$$

Iteration

Definition

Sei L eine formale Sprache. Die Iteration von L ist definiert als

$$L^* = \{x | \text{es gibt ein } n \in \mathbb{N}, \text{ so dass } x = y_1 \frown y_2 \frown \cdots \frown y_n \text{ und } y_i \in L \text{ für alle } i \leq n\}$$

- Beachte, dass auch die leere Kette ϵ ein Element von L^*, für beliebige L. (n ist in dem Fall gleich 0.)
- Man kann L^* auch definieren als

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

Reguläre Ausdrücke

Die Funktion $L(\cdot)$ ordnet jedem regulären Ausdruck eine formale Sprache zu.

Definition

$$
\begin{aligned}
L(\emptyset) &= \emptyset \\
L(\epsilon) &= \{\epsilon\} \\
L(a) &= \{a\} \text{ (wenn } a \in \Sigma) \\
L(\alpha\beta) &= L(\alpha) \frown L(\beta) \\
L((\alpha|\beta)) &= L(\alpha) \cup L(\beta) \\
L(\alpha^*) &= L(\alpha)^*
\end{aligned}
$$

Reguläre Ausdrücke, Typ-3-Grammatiken und endliche Automaten

Mit regulären Ausdrücken kann man drei Arten von Operationen über formale Sprachen ausdrücken, Vereinigung, Verkettung und Operation. Die Klasse der regulären Sprachen ist unter diesen Operationen abgeschlossen.

Vereinigung von regulären Sprachen

Theorem

Wenn L_1 und L_2 reguläre Sprachen sind, dann ist $L_1 \cup L_2$ auch eine reguläre Sprache.

Vereinigung von regulären Sprachen

Beweisidee:

Wenn L_1 eine reguläre Sprache ist, gibt es eine Typ-3-Grammatik $G_1 = \langle V_{T,1}, V_{N,1}, S_1, R_1$, die L_1 generiert. (Wir nehmen ohne Einschränkung der Allgemeinheit an, dass $V_{N,1} \cap V_{N,2} = \emptyset$.)
Genauso gibt es eine Typ-3-Grammatik $G_2 = \langle V_{T,2}, V_{N,2}, S_2, R_2$, die L_2 generiert. Wir konstruieren eine neue Grammatik $G = \langle V_N, V_T, S, R\rangle$ (mit $S \notin V_{N,1} \cup V_{N,2}$), die $L_1 \cup L_2$ generiert:

- $V_T = V_{T,1} \cup V_{T,2}$
- $V_N = V_{N,1} \cup V_{N,2} \cup \{S\}$
-

$$\begin{aligned} R &= R_1 \cup R_2 \cup \\ &\quad \{S \to \alpha | S_1 \to \alpha \in R_1\} \cup \\ &\quad \{S \to \alpha | S_2 \to \alpha \in R_2\} \end{aligned}$$

Verkettung von regulären Sprachen

Theorem

Wenn L_1 und L_2 reguläre Sprachen sind, dann ist auch $L_1 \frown L_2$ eine reguläre Sprache.

Verkettung von regulären Sprachen

Beweisidee:
Wenn L_1 eine reguläre Sprache ist, gibt es eine Typ-3-Grammatik $G_1 = \langle V_{T,1}, V_{N,1}, S_1, R_1$, die L_1 generiert. (Wir nehmen ohne Einschränkung der Allgemeinheit an, dass $V_{N,1} \cap V_{N,2} = \emptyset$.)
Genauso gibt es eine Typ-3-Grammatik $G_2 = \langle V_{T,2}, V_{N,2}, S_2, R_2$, die L_2 generiert. Wir konstruieren eine neue Grammatik $G = \langle V_N, V_T, S_1, R \rangle$, die $L_1 \frown L_2$ generiert:

- $V_T = V_{T,1} \cup V_{T,2}$
- $V_N = V_{N,1} \cup V_{N,2} \cup \{S\}$
-

$$\begin{aligned} R &= R_2 \cup \\ &\quad \{A \to xS_2 | A \to x \in R_1\} \end{aligned}$$

Iteration von regulären Sprachen

Theorem

Wenn L eine reguläre Sprachen ist, dann ist auch L^* eine reguläre Sprache.

Iteration von regulären Sprachen

Beweisidee:
Wenn L eine reguläre Sprache ist, gibt es eine Typ-3-Grammatik $G = \langle V_T, V_N, S, R$, die L generiert.
Wir konstruieren eine neue Grammatik $G' = \langle V_N, V_T, S, R' \rangle$, die L^* generiert:

$$\begin{aligned} R' &= R \cup \\ &\quad \{A \to xS | A \to x \in R\} \end{aligned}$$

Endliche Sprachen sind regulär

Theorem

Jede endliche Sprache ist eine reguläre Sprache.

Beweisidee:
Wir konstruieren eine Typ-3-Grammatik, die L generiert, wie folgt:

$$R = \{S \to x | x \in L\}$$

Reguläre Sprachen und reguläre Ausdrücke

Theorem

Wenn α ein regulärer Ausdruck ist, dann ist $L(\alpha)$ eine reguläre Sprache.

Beweisidee:
Wenn $\alpha = \epsilon$, $\alpha = \{\epsilon\}$ oder $\alpha = \{a\}$ für ein $a \in \Sigma$, dann ist $L(\alpha)$ endlich — und daher auch regulär. Außerdem folgt aus den vorangehenden Theoremen:

- Wenn $L(\alpha)$ und $L(\beta)$ regulär sind, dann sind auch $L((\alpha|\beta)) = L(\alpha)(\beta)$ und $L(\alpha\beta) = L(\alpha) \frown L(\beta)$ regulär.
- Wenn $L(\alpha)$ regulär ist, dann ist auch $L(\alpha^*) = L(\alpha)^*$ regulär.

Allgemein gilt daher: Wenn α keine Vorkommen von Verkettung, Vereinigung oder Iteration enthält, ist $L(\alpha)$ regulär. Außerdem gilt: wenn für alle regulären Ausdrücke α, die maximal n Vorkommen von Verkettung, Vereinigung oder Iteration enthalten, $L(\alpha)$ regulär ist, dann ist $L(\alpha)$ auch dann regulär, wenn α $n+1$ Vorkommen dieser Operationen enthält. Das Theorem folgt somit durch vollständige Induktion.

Reguläre Sprachen und reguläre Ausdrücke

Theorem

Wenn L eine reguläre Sprache ist, dann gibt es eine regulären Ausdruck α, so dass $L(\alpha) = L$.

Der Beweis für dieses Theorem ist im Rahmen dieser Vorlesung zu aufwändig. Er basiert auf einer Konstruktion, die aus einem DFA einen äquivalenten regulären Ausdruck gewinnt. (siehe Schöning, S. 37)

Reguläre Ausdrücke, Grammatiken und Automaten

Theorem

Reguläre Ausdrücke, Typ-3-Grammatiken, deterministische endliche Automaten und nicht-deterministische Automaten beschreiben alle die selbe Klasse von Sprachen.