Einführung in die Computerlinguistik

Reguläre Ausdrücke und endliche Automaten

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Wintersemester 2011/2012

Overview

1. Reguläre Ausdrücke
2. NFAs mit leeren Übergängen
3. Endliche Automaten und reguläre Ausdrücke
4. Abschlusseigenschaften regulärer Sprachen

Reguläre Ausdrücke (1)

Let Σ be an alphabet. The set of regular expressions over Σ is recursively defined:

- $\emptyset$ is a regular expression denoting the set $\emptyset$.
- ϵ is a regular expression denoting the set $\{\epsilon\}$.
- For each $a \in \Sigma$: a is a regular expression denoting $\{a\}$.
- If r and s are regular expressions denoting R and S, then $(r|s)$, (rs) and (r^*) are also regular expressions denoting $R \cup S$, $R \circ S$ and R^* respectively.

Assume that $*$ has a higher priority than concatenation which in turn has a higher priority than $|$.

a^+ is an abbreviation for aa^*.

Reguläre Ausdrücke (2)

Bsp.:

- $\varepsilon|a$ denotiert $\{\varepsilon\} \cup \{a\} = \{a, \varepsilon\}$.
- εa denotiert $\{\varepsilon\} \circ \{a\} = \{a\}$.
- $\emptyset|a$ denotiert $\emptyset \cup \{a\} = \{a\}$.
- $\emptyset a$ denotiert $\emptyset \circ \{a\} = \emptyset$.
- cc^* denotiert die Menge aller Wörter über $\{c\}$, deren Länge ≥ 1 ist.
- $(cc)^*$ denotiert die Menge aller Wörter über $\{c\}$, deren Länge eine gerade Zahl ist.

Reguläre Ausdrücke (3)

- $(a|\varepsilon)bcc^*$ denotiert $\{bc, abc, bcc, abcc, bccc, abccc, \ldots\} = \{a^n bc^m \mid 0 \leq n \leq 1, m \geq 1\}$
- $((a|b)^*|de)^*$ denotiert $\{w_1 \ldots w_n \mid n \geq \varepsilon$, jedes der w_i für $1 \leq i \leq n$ ist entweder aus $\{a, b\}^*$ oder hat die Form $(de)^k$ für irgendein $k \geq 0\}$

  $\Rightarrow$ die Sprache ist die Menge aller Wörter über den Alphabet $\{a, b, d, e\}$, in denen jedes d notwendig von einem e gefolgt ist.

  Äquivalent: $(a|b|de)^*$
- $a(bd^+b)^*a$ denotiert die Menge aller Wörter der Form awa, wobei $w = \varepsilon$ oder $w = bw'b$ und es gilt, dass $w' \in \{b, d\}^*$ mit d anfängt und endet und dass in w' jede Gruppe von bs genau die Länge 2 hat.

Reguläre Ausdrücke (4)

The following holds:

For each language L: there is a regular expression x with $L = L(x)$ iff there is a FSA M with $L = L(M)$.

In order to show this, we

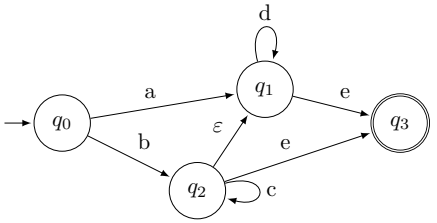
- define NFAs with empty transitions and show their equivalence to NFAs;
- show how to construct an equivalent NFA with empty transitions for a given regular expression;
- show how to construct an equivalent regular expression for a given DFA.

NFAs mit leeren Übergängen (1)

A NFA with empty transitions is defined like an NFA except that δ allows for ε as second argument:

A NFA with empty transitions M is a quintuple $\langle Q, \Sigma, \delta, q_0, F\rangle$ such that

- Q, Σ, q_0 and F are like in a NFA.
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$ is the transition function.



NFAs mit leeren Übergängen (2)

NFAs und NFAs mit leeren Übergängen sind äquivalent. D.h., es gilt folgendes:

Für jede Sprache L gilt: Es gibt einen deterministischen endlichen Automaten (DFA), der L erkennt gdw. es einen NFA gibt, der L erkennt gdw. es einen NFA mit leeren Übergängen gibt, der L erkennt.

FSAs und reguläre Ausdrücke (1)

To show: Endliche Automaten erkennen genau die von regulären Ausdrücken denotierten Sprachen.

1. For a given regular expression r there exists an NFA with empty transitions that accepts the language denoted by r.

   $\Rightarrow$ The set of languages denoted by regular expressions is a subset of the set of languages accepted by FSAs.

2. For a given DFA, there exists a regular expression r such that $L(r)$ is exactly the language acctepted by the DFA.

   $\Rightarrow$ The set of languages accepted by FSAs is a subset of the set of languages denoted by regular expressions.

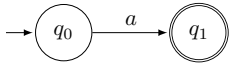
FSAs und reguläre Ausdrücke (2)

Construction of an equivalent NFA with empty transitions for a given regular expression r:

- $r = \epsilon$:

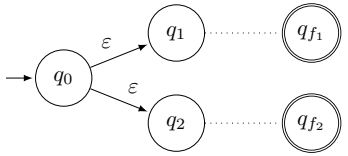
- $r = \emptyset$:

- $r = a$, $a \in \Sigma$:

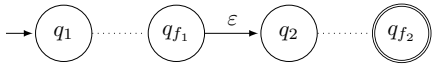


FSAs und reguläre Ausdrücke (3)

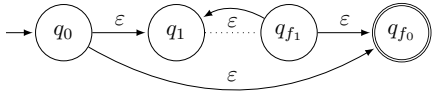
- $r = r_1|r_2$:



- $r = r_1 r_2$:



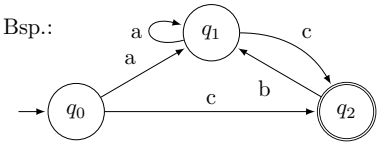
- $r = r_1^*$:



FSAs und reguläre Ausdrücke (4)

For each DFA, there is an equivalent regular expression.

Bsp.:



Um von q_0 zu q_2 zu gelangen, kann man entweder

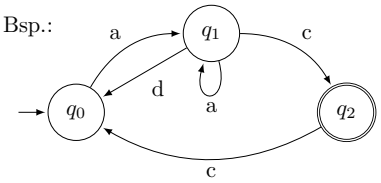
(1) einen Weg ohne Unterwegszustand q_2 wählen. Dies kann wiederum entweder ohne Durchlauf von q_1 sein (c) oder ein erstes Mal von a_0 zu q_1 (a), dann beliebig oft von q_1 zu q_1 ohne Durchlauf von q_1 (a^*), dann von q_1 nach q_2 (c) $\Rightarrow$ $(c|a^+c)$, oder

(2) man geht ein erstes Mal von q_0 zu q_2 $(c|a^+c)$ und dann beliebig oft von q_2 zu q_2 ohne Durchlauf von q_2 $((ba^*c)^*)$.

$\Rightarrow$ $(c|a^+c)(ba^*c)^*$

FSAs und reguläre Ausdrücke (5)

Bsp.:



von q_0 nach q_2 ohne q_1, q_2	$\emptyset$
von q_0 nach q_1 ohne q_1, q_2	a
von q_1 nach q_1 ohne q_1, q_2	$a\|da$
von q_1 nach q_2 ohne q_1, q_2	c
von q_0 nach q_2 ohne q_2	$\emptyset\|a(a\|da)^*c = a(a\|da)^*c$
von q_2 nach q_2 ohne q_2	$ca(a\|da)^*c$
von q_0 nach q_2	$a(a\|da)^*c(ca(a\|da)^*c)^*$

Abschlusseigenschaften regulärer Sprachen (1)

Die von regulären Ausdrücken denotierten Sprachen heißen reguläre Sprachen.

1. Wenn L_1 und L_2 reguläre Sprachen sind, dann
   - ist die Vereinigung von L_1 und L_2 ($L_1 \cup L_2$) ebenfalls eine reguläre Sprache.
   - ist die Schnittmenge von L_1 und L_2 ($L_1 \cap L_2$) ebenfalls eine reguläre Sprache.
   - ist die Konkatenation von L_1 und L_2 ($L_1 \circ L_2$) ebenfalls eine reguläre Sprache.
2. Das Komplement einer regulären Sprache ist eine reguläre Sprache.
3. Wenn L eine reguläre Sprache, dann ist L^* eine reguläre Sprache.

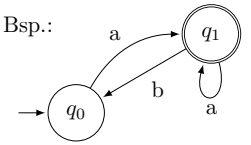
Abschlusseigenschaften regulärer Sprachen (2)

Für Vereinigung, Konkatenation und L^* kann man ganz einfach die entsprechenden regulären Ausdrücke angeben.

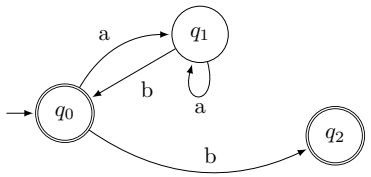
Für die Komplementbildung nimmt man den DFA, der die Sprache erkennt, macht die Übergangsfunktion vollständig, indem man einen weiteren Zustand (*trap state*) hinzufügt, und wählt als neue Endzustände alle Zustände, die im Ursprungsautomaten kein Endzustand sind.

Abschlusseigenschaften regulärer Sprachen (3)

Bsp.:



Komplementbildung:



Abschlusseigenschaften regulärer Sprachen (4)

Schnittbildung kann (de Morgan) durch Vereinigung und Komplementbildung ausgedrückt werden:

$$L_1 \cap L_2 = \overline{\overline{L_1} \cap \overline{L_2}} = \overline{\overline{L_1} \cup \overline{L_2}}$$

Damit ist $L_1 \cap L_2$ eine reguläre Sprache, falls L_1 und L_2 regulär sind.