Introduction to Computational Linguistics

Frank Richter

fr@sfs.uni-tuebingen.de.

Seminar für Sprachwissenschaft

Eberhard Karls Universität Tübingen

Germany

Incremental Linguistic Analysis

- tokenization
- morphological analysis (lemmatization)
- part-of-speech tagging
- named-entity recognition
- partial chunk parsing
- full syntactic parsing
- semantic and discourse processing

Finite State Technology

Regular languages and finite state automata

- deterministic finite state automata,

Finite State Technology

Regular languages and finite state automata

- deterministic finite state automata,
- nondeterministic finite state automata,

Finite State Technology

Regular languages and finite state automata

- deterministic finite state automata,
- nondeterministic finite state automata,
- finite state automata, and

Finite State Technology

Regular languages and finite state automata

- deterministic finite state automata,
- nondeterministic finite state automata,
- finite state automata, and
- regular expressions

Finite State Technology

Regular languages and finite state automata

- deterministic finite state automata,
- nondeterministic finite state automata,
- finite state automata, and
- regular expressions

characterize the same class of languages, *viz.* Type 3 languages

Regular Expressions

Given an alphabet Σ of symbols the following are all and only the regular expressions over the alphabet $\Sigma \cup \{$Ø$, 0, |, *, [,]\}$:

Ø	empty set	
0	the empty string	(ϵ, [])
σ	for all $\sigma \in \Sigma$	
[α | β]	union (for α, β reg.ex.)	($\alpha \cup \beta$, $\alpha + \beta$)
[α β]	concatenation (for α, β reg.ex.)	
[α*]	Kleene star (for α reg.ex.)	

Meaning of Regular Expressions

L(Ø) = $\emptyset$	the empty language
L(0) = $\{0\}$	the empty-string language
L(σ) = $\{\sigma\}$	
L([α | β]) = L(α) $\cup$ L(β)	
L([α β]) = L(α) $\circ$ L(β)	
L([α^*]) = (L(α))*	

Σ^* is called the universal language. Note that the universal language is given relative to a particular alphabet.

Remarks on Regular Expressions

- $\emptyset^* =_{def} \{0\}$
- The empty string, i.e., the string containing no character, is denoted by 0. The empty string is the neutral element for the concatenation operation. That is:

  $$\texttt{for any string } w \in \Sigma^* : \quad w0 = 0w = w$$

- Square brackets, [], are used for grouping expressions. Thus [A] is equivalent to A while (A) is not.
  We leave out brackets for readability if no confusion can arise.

Regular Expressions: Syntax

- ( ) is (sometimes) used for optionality; e.g. (A) ; definable in terms of union with the empty string.
- ? denotes any symbol; L(?) = Σ
- A^+ denotes iteration; one or more concatenations of A. Equivalent to A (A*).
- Note the following simple expressions:
  - [ ] denotes the empty-string language
  - ?* denotes the universal language

Deterministic Finite-State Automata

Definition 1 (DFA) A deterministic FSA (DFA) is a quintuple $(\Sigma, Q, i, F, \delta)$ where

- Σ is a finite set called *the alphabet*,
- Q is a finite set of *states*,
- $i \in Q$ is the *initial state*,
- $F \subseteq Q$ the set of *final states*, and
- δ is the transition function from $Q \times \Sigma$ to Q.

Generalizing Finite-State Automata

Definition 2 (rNFA) A restricted nondeterministic finite-state automaton is a quintuple $(\Sigma, Q, i, F, \Delta)$ where

Σ is a finite set called *the alphabet*,

Q is a finite set of *states*,

$i \in Q$ is the *initial state*,

$F \subseteq Q$ the set of *final states*, and

$\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$ is the set of edges (the transition *relation*).

Nondeterministic Finite-State Automata

Definition 3 (NFA) A nondeterministic finite-state automaton is a quintuple $(\Sigma, Q, S, F, \Delta)$ where

Σ is a finite set called *the alphabet*,

Q is a finite set of *states*,

$S \subseteq Q$ is the set of *initial states*,

$F \subseteq Q$ the set of *final states*, and

$\Delta \subseteq Q \times \Sigma^* \times Q$ is the set of edges (the transition *relation*).

Some Important Properties of FSAs (1)

- Determinization: For every nondeterministic finite-state automaton there exists an equivalent deterministic automaton.
- Minimization: For every nondeterministic finite-state automaton there exists an equivalent deterministic automaton with a minimal number of states.