Computational Linguistics II: Parsing

Formal Languages: Regular Languages II

Frank Richter & Jan-Philipp Söhn

`fr@sfs.uni-tuebingen.de, jp.soehn@uni-tuebingen.de`

Reminder: The Big Picture

hierarchy	grammar	machine	other
type 3	reg. grammar	DFA NFA	reg. expressions
det. cf.	LR(k) grammar	DPDA	
type 2	CFG	PDA	
type 1	CSG	LBA	
type 0	unrestricted grammar	Turing machine	

DFA: Deterministic finite state automaton

(D)PDA: (Deterministic) Pushdown automaton

CFG: Context-free grammar

CSG: Context-sensitive grammar

LBA: Linear bounded automaton

Form of Grammars of Type 0–3

For $i \in \{0, 1, 2, 3\}$, a grammar $\langle N, T, P, S \rangle$ of Type i, with N the set of non-terminal symbols, T the set of terminal symbols (N and T disjoint, $\Sigma = N \cup T$), P the set of productions, and S the start symbol ($S \in N$), obeys the following restrictions:

T3: Every production in P is of the form $A \rightarrow aB$ or $A \rightarrow \epsilon$, with $B, A \in N$, $a \in T$.

T2: Every production in P is of the form $A \rightarrow x$, with $A \in N$ and $x \in \Sigma^*$.

T1: Every production in P is of the form $x_1 A x_2 \rightarrow x_1 y x_2$, with $x_1, x_2 \in \Sigma^*$, $y \in \Sigma^+$, $A \in N$ and the possible exception of $C \rightarrow \epsilon$ in case C does not occur on the righthand side of a rule in P.

T0: No restrictions.

Regular Languages

- Regular grammars,

Regular Languages

- Regular grammars,
- deterministic finite state automata,

Regular Languages

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

Regular Languages

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

Regular Languages

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

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

Reminder: DFA

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.

Reminder: Acceptance

Definition 3 (Acceptance)

Given a DFA $M = (\Sigma, Q, i, F, \delta)$, the language $L(M)$ accepted by M is

$$L(M) = \{x \in \Sigma^* | \hat{\delta}(i, x) \in F\}.$$

Nondeterministic Finite-state Automata

Definition 4 (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

δ is the transition function from $Q \times \Sigma$ to $Pow(Q)$.

Theorem (Rabin/Scott)

For every language accepted by an NFA there is a DFA which accepts the same language.

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.

Theorem (Kleene)

The set of languages which can be described by regular expressions is the set of regular languages.

Pumping Lemma for Regular Languages

uvw theorem:

For each regular language L there is an integer n such that for each $x \in L$ with $|x| \geq n$ there are u, v, w with $x = uvw$ such that

1. $|v| \geq 1$,
2. $|uv| \leq n$,
3. for all $i \in \mathbb{N}_0$: $uv^iw \in L$.

A Non-regular Language

Corollary

Let Σ be {a,b}.
L = {a^{n}b^n | $n \in \mathbb{N}$} is not regular.

Proof

Assume $k \in \mathbb{N}$. For each a^{k}b^k = uvw with v$\neq \epsilon$

1. v = a^l, 0$<$ l $\leq$ k, or
2. v = a^{l_1}b^{l_2}, 0$<$ l$_1$, l$_2$ $\leq$ k, or
3. v = b^l, 0$<$ l $\leq$ k, or

In each case we have uv^{2}w $\notin$ L. The result follows with the Pumping Lemma.

Natural and Regular Languages

Corollary German is not a regular language.

Proof Consider

L$_1$={Ein Spion (der einen Spion)^k observiert^l wird meist selbst observiert}

L$_1$ is regular.

L$_1$ $\cap$ Deutsch =
{Ein Spion (der einen Spion)^k observiert^k wird meist selbst observiert}

is not regular.

Theorem (Myhill/Nerode)

The following three statements are equivalent:

1. The set $L \subseteq \Sigma^*$ is accepted by some DFA.
2. L is the union of some of the equivalence classes of a right invariant equivalence relation of finite index.
3. Let equivalence relation R_L be defined by: xR_Ly iff for all $z \in \Sigma^*$, $xz \in L$ iff $yz \in L$. Then R_L is of finite index.

Minimization

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

Closure Properties of Regular Languages

Regular languages are closed under

- union
- intersection
- complement
- product
- Kleene star

Closure Properties of Regular Languages

Regular languages are closed under

- union (regular expression)
- intersection
- complement
- product (regular expression)
- Kleene star (regular expression)

Closure Properties of Regular Languages

Regular languages are closed under

- union (regular expression)
- intersection (e.g. constructive)
- complement
- product (regular expression)
- Kleene star (regular expression)

Closure Properties of Regular Languages

Regular languages are closed under

- union (regular expression)
- intersection (e.g. constructive)
- complement (DFA)
- product (regular expression)
- Kleene star (regular expression)

Decidable Problems for Reg. Languages

1. Word problem

Decidable Problems for Reg. Languages

1. Word problem
2. Emptiness

Decidable Problems for Reg. Languages

1. Word problem
2. Emptiness
3. Finiteness

Decidable Problems for Reg. Languages

1. Word problem
2. Emptiness
3. Finiteness
4. Intersection

Decidable Problems for Reg. Languages

1. Word problem
2. Emptiness
3. Finiteness
4. Intersection
5. Equivalence