Formal languages, grammars, and automata

Herman Geuvers

(All info on

`http://www.cs.ru.nl/~herman/onderwijs/FLGA/`

or google me)

Overview

Topics

Languages:	regular	context-free	[natural languages]	[enumerable]
Automata:	finite	push-down	[bounded Turing machine]	[Turing machin
Grammars:	regular	context-free	[context-sensitive]	[NL grammars?

Automata: accept words of a language

given a word, compute if it is in the language

Grammars: generate words of a language

produce all correct words in the language

Languages

Alphabet

An *alphabet* Σ is a (finite) set of symbols

Examples

$$\begin{aligned}
\Sigma_1 &= \{a\} \\
\Sigma_2 &= \{0,1\} \\
\Sigma_3 &= \{A,C,G,T\} \\
\Sigma_4 &= \{a,b,c,d,\ldots,x,y,z\} \\
\Sigma_5 &= \{s \mid s \text{ is an ascii symbol}\} \\
\Sigma_6 &= \{\} \\
& \quad \text{Japanese alphabet: } 2 \times 52 \text{ signs} \\
\Sigma_7 &= \{,\ldots\} \\
& \quad \text{Chinese alphabet: } \pm 40.000 \text{ signs} \\
\Sigma_8 &= \{0,1,+,\times,x_0,x_1,x_2,\ldots\} \\
& \quad \text{mathematical alphabet, number of signs: countably infinite} \\
\Sigma_9 &= \{0,1,+,\times,x_0,x_1,x_2,\ldots\} \cup \{c_r \mid r \in \mathbb{R}\} \\
& \quad \text{mathematical alphabet, number of signs uncountably infinite}
\end{aligned}$$

Words

A *word* (string) over Σ is a finite sequence of elements from Σ

The set Σ^* consists of all *words over* Σ

Inductive generation of words

$$\begin{array}{rcll} & & \lambda \in \Sigma^*, & \lambda \text{ denotes the empty word} \\ w \in \Sigma^* & \Rightarrow & ws \in \Sigma^*, & \text{for all } s \in \Sigma \end{array}$$

Note that λw is just w

Note the difference between $a \in \Sigma$ and $a \in \Sigma^*$

Think of a word as a chain of letters on a necklace:

$$\begin{array}{rcl} \lambda & = & \text{——} \\ Eva & = & \text{–}E\text{-}v\text{-}a\text{–} \end{array}$$

The difference between a and $-a-$ is clear

Operation on words; Language

Operations on words

$$u \in \Sigma^*,\ v \in \Sigma^* \ \Rightarrow\ u \cdot v \in \Sigma^* \qquad \textit{concatenation}$$

$$u \in \Sigma^*,\ n \in \mathbb{N} \ \Rightarrow\ u^n \in \Sigma^* \qquad \text{repeating words}$$

Inductive definitions

$$\begin{array}{rcl} u \cdot \lambda & = & u \\ u \cdot (vs) & = & (u \cdot v)s \end{array} \qquad \begin{array}{rcl} u^0 & = & \lambda \\ u^{k+1} & = & u^k \cdot u \end{array}$$

usually we write concatenation $u \cdot v$ as uv

A *language over* Σ is a subset of Σ^*, notation $L \subseteq \Sigma^*$

Examples

$$L_1 = \{w \in \{a,b\}^* \mid abba \text{ is a substring of } w\}$$
$$L_2 = \{w \in \{a,b\}^* \mid abba \text{ is not a substring of } w\}$$

Examples of languages

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

1. $L_1 = \{a^n \mid n \in \mathbb{N} \text{ is even}\}$
2. $L_2 = \{a^n b^n \mid n \in \mathbb{N}\}$
3. $L_3 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$
4. $L_4 = \{a^n \mid n \in \mathbb{N} \text{ is prime}\}$
5. $L_5 = \{n \mid n \text{ denotes an integer number}\}$
6. $L_6 = \{e \mid e \text{ is a well-formed arithmetical expression}\}$
7. $L_7 = \{P \mid P \text{ is a syntactically correct Java program}\}$
8. $L_8 = \{S \mid S \text{ is a grammatically correct English sentence}\}$

Operations on languages

Given languages $L_1, L_2, L \subseteq \Sigma^*$ we can define

$$L_1 \cup L_2$$

$$L_1 L_2$$

$$L^*$$

again languages over Σ

$$\begin{aligned}
L_1 \cup L_2 &= \{w \mid w \in L_1 \text{ or } w \in L_2\} \\
L_1 L_2 &= \{w_1 w_2 \mid w_1 \in L_1 \ \& \ w_2 \in L_2\} \\
L^0 &= \{\lambda\} \\
L^{n+1} &= L^n L \\
L^* &= \bigcup_{n \in \mathbb{N}} L^n = L^0 \cup L^1 \cup L^2 \cup \ldots \\
&\neq \{w^n \mid w \in L,\ n \in \mathbb{N}\}
\end{aligned}$$

Regular expressions & languages over Σ

Let $\Sigma = \{a, b\}$. Then $a(ba)^*bb$ is a *regular expression* denoting

$$
\begin{aligned}
L &= \{a(ba)^n bb \mid n \in \mathbb{N}\} \\
&= \{abb, ababb, abababb, ababababb, \ldots, a(ba)^n bb, \ldots\}
\end{aligned}
$$

For general Σ the regular expressions over Σ are generated by

$$\boxed{\mathtt{rexp}_\Sigma ::= \emptyset \mid \lambda \mid \mathtt{s} \mid \mathtt{rexp}_\Sigma\, \mathtt{rexp}_\Sigma \mid \mathtt{rexp}_\Sigma \cup \mathtt{rexp}_\Sigma \mid \mathtt{rexp}_\Sigma^*}$$

with $\mathtt{s} \in \Sigma$

This means $\emptyset \in \mathtt{rexp}_\Sigma$, $\lambda \in \mathtt{rexp}_\Sigma$, and $s \in \mathtt{rexp}_\Sigma$ for $s \in \Sigma$ and

$$
\begin{aligned}
e_1, e_2 \in \mathtt{rexp}_\Sigma &\Rightarrow (e_1 \cup e_2) \in \mathtt{rexp}_\Sigma \\
e_1, e_2 \in \mathtt{rexp}_\Sigma &\Rightarrow (e_1 e_2) \in \mathtt{rexp}_\Sigma \\
e \in \mathtt{rexp}_\Sigma &\Rightarrow (e^*) \in \mathtt{rexp}_\Sigma
\end{aligned}
$$

For example $(abb)^*(a \cup \lambda)$ is a regular expression

Regular languages

For a regular expression e over Σ we define the language $L(e)$:

$$\begin{aligned} L(\emptyset) &= \emptyset \\ L(\lambda) &= \{\lambda\} \\ L(s) &= \{s\} \\ L(e_1 e_2) &= L(e_1)L(e_2) \\ L(e_1 \cup e_2) &= L(e_1) \cup L(e_2) \\ L(e^*) &= L(e)^* \end{aligned}$$

A language L is called *regular* if $L = L(e)$ for some $e \in \texttt{rexp}$

Examples

$L = \{w \mid bb \text{ occurs in } w\}$ over $\Sigma = \{a, b\}$ is regular:

$$L = L((a \cup b)^* bb(a \cup b)^*)$$

Also $L' = \Sigma^* - L = \{w \mid bb \text{ does not occur in } w\}$:

$$L' = L(a^*(baa^*)^* \cup a^*(baa^*)^* b)$$

To come

Regular Languages	Context-free Languages	Context-sensitive Languages	Enumerable Languages
Finite Automata	Push-down Automata	Bounded Turing Machines	Turing Machines
Regular Grammars	Context-free Grammars	Context-sensitive Grammars	General Grammars

Topics described in the last two columns are treated in other course