Introduction to Logic and Computer Science
Foundation of Computer Science
Logic in Computer Science

Lecture #10: regular expressions

Tamás Mihálydeák, László Aszalós

This work was supported by the construction EFOP-3.4.3-16-2016-00021.
The project was supported by the European Union, co-financed by the European Social Fund.

1. Problems
2. Formal languages
3. Regular expression
4. Regular language

Problems

- Analysis and generation are typical tasks in informatics:
  - In communication different standards are used
    - HTML, XML, json, YAML, etc
  - We have to generate on sending side, and analyse on receiving side.
- In the case of forms:
  - Is the input an e-mail address?
  - Does the password contain numbers, capitals, etc.
- Which source file contains TODO?
- If we use standard tools we can be more efficient.

Formal languages in computer graphics

example - Lindenmayer system

Formal language

Definition

Let Σ be a finite non empty set! The members of Σ are characters, the set Σ is an alphabet.

- The members of the set Σ^+ are finite non empty sequences from the characters of Σ ($w = a_1 \ldots, a_n$, where $a_i \in \Sigma$)
  - the set Σ^+ is the set of finite non empty words over Σ.
- If $w \in \Sigma^+$, then the length of the word w is the number of its characters ($|a_1 \ldots, a_n| = n$).
- ε is the empty word
  - It does not contain any character, and its length is 0.
- The set $\Sigma^\star = \Sigma^+ \cup \{\varepsilon\}$ is the set of words over Σ.
- If $L \subseteq \Sigma^\star$ then the set L is a (formal) language over the alphabet Σ.
- $L_\emptyset = \emptyset$ is the empty language (it has no words).

Operations of formal languages

Definition

Let L, L_1 and L_2 be formal languages over Σ. Then

- $L_1 \cup L_2 = \{x | x \in L_1 \text{ or } x \in L_2\}$ (union);
- $L_1 \cap L_2 = \{x | x \in L_1 \text{ and } x \in L_2\}$ (intersection);
- $L_1 \cdot L_2 = \{xy | x \in L_1 \text{ and } y \in L_2\}$ (concatenation);
- Power of a formal language is concatenation with itself:
  - $L^0 = \{\varepsilon\}$
  - $L^{n+1} = L^n \cdot L$
- Transitive closure (Kleene-star operation):
  - $L^\star = \bigcup_{i=0}^{\infty} L_i$, i.e. $L^\star = \{x_1 x_2 ... x_k | k \geq 0 \text{ and } x_i \in L\}$

Operations of formal languages

Example

Let $\Sigma = \{a, b, c\}$, $A = \{a, bb, ccc\}$ and $B = \{ac, bc\}$. Then

- $A \cup B = \{a, ac, bb, bc, ccc\}$,
- $A \cdot B = \{aac, alphabet, bbac, bbbc, cccac, cccbc\}$,
- $A^\star = \{\varepsilon, a, bb, ccc, aa, abb, accc, bba, bbbb, bbccc, \dots\}$.

Remark

- $L_\varepsilon^\star = \{\varepsilon\}$, where $L_\varepsilon = \{\varepsilon\}$
- $L_\emptyset^\star = \{\varepsilon\}$, because by definition $L_\emptyset^0 = \{\varepsilon\}$

Defining formal languages

- By specification: see in the previous examples:
  - these cases formal languages may have only finite many words.
- By generalisation (by inductive definition):
  - generative grammars (Chomsky type classes of formal languages).
- By an automaton:
  - an automata decides whether a given word is a member of our language.
- By expressions which characterise the words of a language:
  - for example the system of regular expressions.

Regular expression

Problem

Let $\Sigma = \{0, 1\}$ be an alphabet. How can we define the language L (that is, how can we express the language L) with the help of other (possibly simpler) languages where L contains the words with subwords 010?
Informal solution: if $w_1, w_2 \in \Sigma^\star$, then $w_1 010 w_2 \in L$.

- How to define the words w_1 and w_2?
- We know that w_1 and w_2 contain characters 0 and 1 in non-empty cases.
- Let $L_0 = \{0\}$ and $L_1 = \{1\}$. Then $L_0 \cup L_1 = \{0, 1\}$, and $(L_0 \cup L_1)^\star$ contains the possible words w_1 and w_2 (including the empty word).
- If $L_{010} = \{010\}$, then $L = (L_0 \cup L_1)^\star \cdot L_{010} \cdot (L_0 \cup L_1)^\star$.
- L_0, L_1, L_{010} are languages with only one word!
- With other notation: $(0 + 1)^\star \cdot 010 \cdot (0 + 1)^\star$

Examples for regular expressions

- Define the language over the alphabet $\{0, 1\}$ containing words with subwords 000 or 111!
  - $(0 + 1)^\star \cdot (000 + 111) \cdot (0 + 1)^\star$
- Define the language over the alphabet $\{0, 1\}$ containing words in which the number of the character 1 can be divided by 5!
  - $0^\star + (0^\star \cdot 1 \cdot 0^\star \cdot 1 \cdot 0^\star \cdot 1 \cdot 0^\star \cdot 1 \cdot 0^\star \cdot 1 \cdot 0^\star)^\star$

History

- Stephen C. Klenne, *regular sets* (1956)
- Ken Thompson, QED text editor, Just-In-Time compiler (1968)
- Unix: vi, lex, sed, awk, emacs (197?)
- Perl (1986)

Definition

Let Σ be an alphabet such that $\emptyset, \varepsilon, +, \cdot, \star, (\ ,) \notin \Sigma$. The set $\mathcal{R}$ of regular expressions over the alphabet Σ can be defined by the following inductive expressions:

1. $\emptyset \in \mathcal{R}$
2. $\varepsilon \in \mathcal{R}$
3. if $a \in \Sigma$, then $a \in \mathcal{R}$.
4. if $R_1 \in \mathcal{R}$ and $R_2 \in \mathcal{R}$ then $(R_1 + R_2) \in \mathcal{R}$
5. if $R_1 \in \mathcal{R}$ and $R_2 \in \mathcal{R}$ then $(R_1 \cdot R_2) \in \mathcal{R}$
6. if $R \in \mathcal{R}$ then $R^\star \in \mathcal{R}$

Formal languages described by regular expressions

Definition

Let $\mathcal{R}$ be the set of regular expressions over the alphabet Σ and $R \in \mathcal{R}$. The language described by the regular expression R is given by the following inductive definition:

1. if $R = \emptyset$, then $L_R = \emptyset$.
2. if $R = \varepsilon$, then $L_R = \{\varepsilon\}$.
3. if $R = a$, then $L_R = \{a\}$ (where $a \in \Sigma$).
4. if $R = (R_1 + R_2)$, then $L_R = L_{R_1} \cup L_{R_2}$.
5. if $R = (R_1 \cdot R_2)$, then $L_R = L_{R_1} \cdot L_{R_2}$.
6. if $R = R_1^\star$, then $L_R = L_{R_1}^\star$.

Regular language

Definition

A language L over alphabet Σ is regular, if there is a regular expression R over alphabet Σ such that L_R=L\$

Theorem

If languages L_1 and L_2 over alphabet Σ are regular, then $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 \backslash L_2$, $\overline{L_1} = \Sigma^\star \backslash L_1$, $L_1 \cdot L_2$ and $L_1^\star$ are regular.

Regular expressions in the practice

There are many semi-standards containing the following constructions:

- $R? = R + \varepsilon$ (0 or 1 occurrence)
- $R^+ = RR^\star$ (1 or more occurrences)
- $R\{n\}$ (exactly n occurrences)
- $R\{n, m\}$ (at least n, at most m occurrences)
- $[a_1 \ldots a_n] = a_1 + \cdots + a_n$ (set of letters)
- $[a_1 - a_n]$ (interval of letters, eg. $a - z$)

Unix uses the character $|$ instead of $+$ in regular expressions: $foo|bar$.