Formal Languages and Compilers

Matteo Secco

March 3, 2021

Contents

1 Formal Language Theory **3**
1.1 Operations on strings . . . 3
1.2 Operations on Languages . . . 4

2 Regular Expressions and Languages **6**
2.1 Algebraic definition . . . 6
2.2 Language Families . . . 7

1 Formal Language Theory

Alphabet Σ: any <u>finite</u> set of symbols $\Sigma = \{a_1, a_2, ..., a_k\}$

String: a sequence of alphabeth elements

Language: a set (possibly infinite) of strings

$$\Sigma = \{a, b, c\} \quad L_1 = \{ab, ac\} \quad L_2 = \{ab, aab, aaab, aaaab, ...\}$$

Sentences/Phrases: strings belonging to a language

Language cardinality: number of sentences of the language

$$|L_1| = |\{ab, ab\}| = 2 \quad |L_2| = |\{ab, aab, aaab, aaaab, ...\}| = \infty$$

Number of occurrences of a symbol in a string: $|bbc|_b = 2$, $|bbc|_a = 0$

Length of a string: number of its elements

$$|bbc| = 3 \quad |abbc| = 4$$

String equality: two strings $x = a_1a_2...a_h$ and $y = a_1a_2...a_k$ are equal $\iff$

- have same length: $|x| = |y| \iff h = k$
- elements from left to right coincide: $a_i = b_i \quad \forall i \in \{1..h\}$

1.1 Operations on strings

Concatenation $x = a_1a_2...a_h \wedge y = b_1b_2...b_k \implies x \cdot y = a_1a_2...a_hb_1b_2...b_k$

- associative: $(xy)z = x(yz)$
- length: $|xy| = |x| + |y|$

Empty string ϵ is the neutral element for concatenation: $x\epsilon = \epsilon x = x \, \forall x$.

- length: $|\epsilon| = 0$
- **NB:** $\epsilon \neq \emptyset$

Substrings: if $x = uyv$ then

- y is a substring of x
- y is a <u>proper substring</u> of $x \quad \iff u \neq \epsilon \vee v \neq \epsilon$
- u is a <u>prefix</u> of x
- v is a <u>suffix</u> of y

Reflection: if $x = a_1a_2...a_h$ then $x^R = a_ha_{h-1}...a_1$

- $(x^R)^R = x$
- $(xy)^R = y^R x^R$
- $\epsilon^R = \epsilon$

Repetition: $x^m = \underbrace{xxx...x}_{m \text{ times}}$. Inductive definition:

- $x^0 = \epsilon$
- $x^m = x^{m-1}x \quad \text{if } m > 0$

1.2 Operations on Languages

Reflection: $L^R = \{x | \exists y \left(y \in L \wedge x = y^R\right)\}$

Prefixes(L): $\{y | y \neq \epsilon \wedge \exists x \exists z \left(x \in L \wedge z \neq \epsilon \wedge x = yz\right)\}$

- **Prefix-free language:** $L \cap Prefixes(L) = \emptyset$

Concatenation: $L'L'' = \{xy | x \in L' \wedge y \in L''\}$

Power: inductive definition:

- $L^0 = \{\epsilon\}$
- $L^m = L^{m-1}L \quad \text{for } m > 0$
- Consequences:
  - $\emptyset^0 = \{\epsilon\}$
  - $L \cdot \emptyset = \emptyset \cdot L = \emptyset$
  - $L \cdot \{\epsilon\} = \{\epsilon\} \cdot L = L$

Universal language: over alphabet Σ: $L_{\text{universal}} = \Sigma^0 \cup \Sigma^1 \cup ...$

Complement: of L over Σ: $\neg L = L_{\text{universal}} \backslash L$

Star: formally called **reflexive and transitive closure** or **Klenee star**

$$L^* = \bigcup_{h=0}^{\infty} L^h = L^0 \cup L^1 \cup ... = \epsilon \cup L^1 \cup L^2$$

$$\Sigma^* = L_{\text{universal}}$$

Monotonic: $L \subseteq L^*$

Close under concatenation: $x \in L^* \wedge y \in L^* \implies xy \in L^*$

Idempotent: $(L^*)^* = L^*$

Commutative with reflection: $(L^*)^R = (L^R)^*$

$\emptyset^* = \{\epsilon\}$

$\{\epsilon\}^* = \{\epsilon\}$

Cross: $L^+ = L \cdot L^*$

Quotient: $L_1/L_2 = \{y | \exists x \in L_1 \exists z \in L_2 (x = yz)\}$

- **Not set quotient!**
- Removes from L_1 suffixes contained in L_2

2 Regular Expressions and Languages

Regular languages are the simplest family of laguages.
They can be defined in three ways:

- Algebraically
- Using generative grammars
- Using recognizer automata

2.1 Algebraic definition

Regular expressions are expression on languages that composes languages operations.
Formally

- Is a string r
- Over the alphabet $\Sigma = \{a_1, a_2, ..., a_n\} \cup \{\emptyset, \cup, \cdot, *\}$

Moreover, assuming s and t are regular expressions, then r is a regular expression if any of the following rules applies:

- $r = \emptyset$
- $r = a, \quad a \in \Sigma$
- $r = s \cup t$ (alternative notation is $s|t$)
- $r = s \cdot t$ (the $\cdot$ can be omitted)
- $r = s^*$

The meaning of a r.e. is a **language** L_r of alphabet Σ according to the table:

Expression	**Language**
$\emptyset$	$\emptyset$
ϵ	$\{\epsilon\}$
$a \in \Sigma$	$\{a\}$
$s \cup t$	$L_s \cup L_t$
$s \cdot t$	$L_s \cdot L_t$
s^*	L_s^*

Regular Languages are languages denoted by a regular expression

2.2 Language Families

REG is the collection of all regular languages

FIN is the collection of all languages with finite cardinality

Every finite language is regular $FIN \subset REG$:

- $L \in FIN \implies L = \bigcup_{i=1}^{k \in \mathbb{N}} x_i \implies L \in FIN$
- $L = a^* \implies L \in REG \land L \notin FIN$