Languages

U.D. Computación

DSIC - UPV

September 5, 2018

Definitions: Alphabet

- *Alphabet*: Finite set of symbols
  - $\Sigma = \{a, b, c\}$
  - $\Gamma = \{0, 1\}$
  - $\Delta_1 = \{\triangle, \square, \bigcirc\}$
  - $\Delta_2 = \{N, S, E, W\}$
- Example of sets that are not alphabets:
  - $\emptyset$
  - $\mathbb{N}$

Definitions: Word

- (Also known as *string* o *phrase*) finite and ordered sequence of symbols from a given alphabet
  - words over $\{a, b\}$: $x = aaba$, $y = aa$
  - words over $\{0, 1, 2\}$: $x = 2110$, $y = 0101$
- *empty word*: λ.

Definitions: Length

- *Length* of a word: number of symbols of the word
  Let x and y be words over Σ, and let a be a symbol in Σ:

$$|x| = \begin{cases} 0 & \text{if } x = \lambda \\ 1 + |y| & \text{if } x = ay \end{cases}$$

- $|x|_a$ number of symbols a in the word x

Definitions: Words over Σ

- Σ^n set of words of length n over the alphabet
- $\Sigma^* = \bigcup_{i \geq 0} \Sigma^i$
- $\Sigma^+ = \bigcup_{i > 0} \Sigma^i$

Definitions: Canonic order

- The alphabetic order ($<_\Sigma$) does not allow an effective enumeration of the words over Σ
- Given two words x and y over Σ, the *canonic order* is defined as follows:

$$x < y \text{ if } \begin{cases} |x| < |y| \\ (|x| = |y|) \ \wedge \ (x = uav,\ y = ubw,\ a <_\Sigma b) \end{cases}$$

Operations on words: Concatenation

Given $x = a_1 a_2 \cdots a_m$ and $y = b_1 b_2 \cdots b_n$, $a_i, b_j \in \Sigma$, the *concatenación* of x and y is defined as:

$$x \cdot y = xy = a_1 a_2 \cdots a_m b_1 b_2 \cdots b_n$$

The *power of a word* is defined taking into account the concatenation:

$$x^n = \begin{cases} \lambda & \text{if } n = 0 \\ x \cdot x^{n-1} = x^{n-1} \cdot x & \text{if} n > 0 \end{cases}$$

Operations on words: Concatenation

Properties of the concatenation

Let x, y, z be words in Σ^* and $a \in \Sigma$

1. Asociative: $(x \cdot y) \cdot z = x \cdot (y \cdot z)$.
2. *Neutral element* (λ): $x\lambda = \lambda x = x$.
3. $|xy| = |x| + |y|$

Operations on words: Segment, prefix, sufix

Given x and t, words over Σ^*

- t is a *segment* of x if there exist u and v such that $x = u \cdot t \cdot v$.
- If $u = \lambda$, then t is a *prefix* of x.
- If $v = \lambda$, then t is a *sufix* of x.

Operations on words: Reverse

Given $x, y \in \Sigma^*$ and a symbol a of the alphabet, the *reverse* of a word is defined as:

$$\begin{cases} \lambda^r = \lambda \\ a^r = a \\ (ax)^r = x^r a \\ (xa)^r = a x^r \end{cases}$$

Operations on words: Reverse

Properties of the reverse

Let x and y be two words in Σ^*

1. $(x^r)^r = x$
2. $(xy)^r = y^r x^r$
3. $(x^n)^r = (x^r)^n$ for any integer $n \geq 0$

Languages: Definitions

A *language* L is a subset of Σ^*

and therefore, these sets are also languages:

- $\emptyset$ (empty language, it does not contain any word)
- Σ^* (all the possible words over Σ)

- A language is *finite* if it contains a finite set of words
- Otherwise, the language is infinite enumerable

Languages: Boolean operations

- Union: $L_1 \cup L_2 = \{x \in \Sigma^* \ : \ x \in L_1 \vee x \in L_2\}$
- Intersection: $L_1 \cap L_2 = \{x \in \Sigma^* \ : \ x \in L_1 \wedge x \in L_2\}$
- Complementation: $\overline{L} = \{x \in \Sigma^* \ : \ x \notin L\}$

Languages: Boolean operations

Properties of the union and intersection

- Associative
- Commutative
- Neutral element $(\emptyset, \Sigma^*)$
  - Union: $\emptyset$
  - Intersection: Σ^*
- Distributive:
  - $L_1 \cup (L_2 \cap L_3) = (L_1 \cup L_2) \cap (L_1 \cup L_3)$
  - $L_1 \cap (L_2 \cup L_3) = (L_1 \cap L_2) \cup (L_1 \cap L_3)$

Languages: Boolean operations

Properties of the complementation

- $\overline{\Sigma^*} = \emptyset$
- $\overline{\emptyset} = \Sigma^*$
- $\overline{\overline{L}} = L$

Languages: Boolean operations

Other operations

- Difference: $L_1 - L_2 = L_1 \cap \overline{L}_2$.
- Symmetric difference: $L_1 \ominus L_2 = \left(L_1 \cap \overline{L}_2\right) \cup \left(\overline{L}_1 \cap L_2\right)$.

Languages: Rational operations. Product

$$L_1 \cdot L_2 = \{xy \in \Sigma^* \;:\; x \in L_1 \wedge y \in L_2\}$$

Properties

- (Non-commutative). $L_1 \cdot L_2$ is not, necessarily, equal to $L_2 \cdot L_1$
- (Associative) $(L_1 \cdot L_2) \cdot L_3 = L_1 \cdot (L_2 \cdot L_3)$
- (Neutral element) $L \cdot \{\lambda\} = \{\lambda\} \cdot L = L$
- (Zero) $L \cdot \emptyset = \emptyset \cdot L = \emptyset$
- $\lambda \in L_1 \cdot L_2 \Leftrightarrow \lambda \in L_1 \wedge \lambda \in L_2$
- $L_1 \cdot (L_2 \cup L_3) = L_1 \cdot L_2 \cup L_1 \cdot L_3$
- $L_1 \cdot (L_2 \cap L_3) \subseteq L_1 \cdot L_2 \cap L_1 \cdot L_3$
  - Example: $L_1 = \{a, ab\}$, $L_2 = \{a\}$, $L_3 = \{ba\}$.

Languages: Rational operations. Power

$$L^n = \begin{cases} \{\lambda\} & \text{si } n = 0 \\ LL^{n-1} = L^{n-1}L & \text{si } n > 0 \end{cases}$$

Examples

- $\{aa, b\}^2 = \{bb, aab, baa, aaaa\}$
- $\emptyset^0 = (\Sigma^*)^0 = \{\lambda\}^0 = \{\lambda\}$

Languages: Rational operations. Closure

Star closure

$$L^* = \bigcup_{i \geq 0} L^i$$

Positive closure

$$L^+ = \bigcup_{i > 0} L^i$$

Languages: Rational operations. Closure

Relationship between star and positive closures

$$L^{+} = \begin{cases} L^{*} & \text{si } \lambda \in L \\ L^{*} - \{\lambda\} & \text{si } \lambda \notin L \end{cases}$$

Languages: Rational operations. Closure

Properties:

1. $L \subseteq L^+ \subseteq L^*$ (because $L = L^1$).
2. $L_1 \subseteq L_2 \Rightarrow L_1^n \subseteq L_2^n$ ($\forall n \in \mathbb{N}$).
3. $L_1 \subseteq L_2 \Rightarrow L_1^* \subseteq L_2^* \ \left(L_1^+ \subseteq L_2^+\right)$
4. $(L^*)^* = L^*$
5. $(L^+)^+ = L^+$
6. $L^+ = L^*L = LL^*$
7. $(L^+)^* = L^*$
8. $(L^*)^+ = L^*$

Languages: Quotient

Right quotient

$$u^{-1}L = \{v \in \Sigma^* \ : \ uv \in L\}$$

Left quotient

$$Lu^{-1} = \{v \in \Sigma^* \ : \ vu \in L\}$$

Languages: Quotient

The quotient with respect to a word is usually referred to as *derivative*

Properties ($u, v \in \Sigma^*$, $a \in \Sigma$)

- $L_1 \subseteq L_2 \Rightarrow u^{-1}L_1 \subseteq u^{-1}L_2$
- $u^{-1}(L_1 \cup L_2) = u^{-1}L_1 \cup u^{-1}L_2$
- $u^{-1}(L_1 \cap L_2) = u^{-1}L_1 \cap u^{-1}L_2$
- $$a^{-1}(L_1L_2) = \begin{cases} \left(a^{-1}L_1\right)L_2 & \text{si } \lambda \notin L_1 \\ \left(a^{-1}L_1\right)L_2 \cup a^{-1}L_2 & \text{si } \lambda \in L_1 \end{cases}$$
- $a^{-1}L^* = \left(a^{-1}L\right)L^*$
- $(uv)^{-1}L = v^{-1}\left(u^{-1}L\right)$

Languages: Homomorphisms

Given two alphabets Σ and Γ, an *homomorphism* is a mapping:

$$h : \Sigma \to \Gamma^*$$

This definition can be extended to words:

$$h : \Sigma^* \to \Gamma^*$$

$$\begin{cases} h(\lambda) = \lambda \\ h(xa) = h(x)h(a) \end{cases}$$

as well as to languages:

$$h(L) = \{h(x) : x \in L\}$$

Languages: Homomorphism

Examples:

$$L_1 = \{\lambda, aa, bab, bbba\}$$

$$L_2 = \{x \in \{a,b\}^* \ : \ aa \notin Seg(x)\}$$

$$\begin{cases} h(a) = \lambda \\ h(b) = 1 \end{cases} \qquad \begin{cases} g(a) = 01 \\ g(b) = 1 \end{cases}$$

1. $h(L_1) = \{\lambda, 11, 111\}$
2. $h(L_2) = \{1\}^*$
3. $g(\{a,b\}^*) = \{x \in \{0,1\}^* \ : \ 00 \notin Seg(x) \wedge 0 \notin Suf(x)\}$

Languages: Inverse homomorphism

Given an homomorphism $h : \Sigma^* \rightarrow \Gamma^*$, the *inverse homomorphism* is defined as:

$$h^{-1}(y) = \{x \in \Sigma^* \ : \ h(x) = y\}$$

This operation can be extended to languages:

$$h^{-1}(L) = \{x \in \Sigma^* : h(x) \in L\}$$

Languages: Inverse homomorphism

Examples:

$$L_1 = \{\lambda, aa, abab, bbba\}$$

$$L_2 = \{x \in \{a, b\}^* \;:\; aa \notin Seg(x)\}$$

$$\begin{cases} h(0) = ab \\ h(1) = ba \end{cases} \qquad \begin{cases} g(0) = aa \\ g(1) = bab \end{cases}$$

1. $h^{-1}(L_1) = \{\lambda, 00\}$
2. $g^{-1}(L_2) = \{x \in \{0, 1\}^* \;:\; |x|_0 = 0\} = \{1\}^*$
3. $h^{-1}(L_2) = \{x \in \{0, 1\}^* \;:\; 10 \notin Seg(x)\}^*$

Languages: Other operations. Reverse

It is possible to extend an operation defined on words to operate on languages. Reverse is an example of the first approach:

$$L^r = \{x^r : x \in L\}$$

Languages: Other operations. Reverse

Properties

1. Si $\Sigma = \{a\}$, $L^r = L$.
2. $(L_1 L_2)^r = L_2^r L_1^r$
3. $(L^n)^r = (L^r)^n$
4. $(L^*)^r = (L^r)^*$
5. $(L^r)^r = L$

Languages: Other operations. Segment, prefix, sufix

First approach is not valid if the operation on words return a set (language). A second approach is needed:

$$Seg(L) = \bigcup_{x \in L} Seg(x)$$

$$Pref(L) = \bigcup_{x \in L} Pref(x)$$

$$Suf(L) = \bigcup_{x \in L} Suf(x)$$

Classes of languages

A *class of languages* is a collection or non-empty set of languages.

Examples

1. $\mathcal{L}_{FIN}$ Class of finite languages
2. $\mathcal{L}_{PAL} = \{L \subseteq \Sigma^* \ : \ x \in L \rightarrow x = x^r\}$
   (class of palindromic languages)
3. $\mathcal{L}_{EVE} = \{L \subseteq \Sigma^* \ : \ x \in L \rightarrow |x| \mod 2 = 0\}$
   (class of even languages)
4. $\mathcal{L}_{no\lambda} = \{L \subseteq \Sigma^* \ : \ \lambda \notin L\}$
   (class of languages that do not contain λ)