Theory of Computation

Languages and Operations

Lecture 2a - Manuel Mazzara

Languages

- A <u>language</u> is a set of strings over an <u>alphabet</u>
- Languages:
  - Russian, Italian, English, French
  - C, Java, Pascal, Eiffel

  but also
  - Graphical languages
  - Music
  - Multimedia

Formally

- A language **L** over an alphabet **A** is a <u>subset</u> of **A***
- Examples
  - $A=\{a, b, c\}$

    $A^*=\{\varepsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, \ldots\}$

    $L_1= \{\varepsilon, a, b, c, bc, ca\}$

    $L_2= \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$

Operations

- Operations on <u>sets</u> apply also to <u>languages</u>
  - A language is a <u>set of strings</u>
- Operations on languages are
  - Union
  - Intersection
  - Difference
  - Complement
  - Concatenation
  - Power of n
  - Kleene closure

Set operations (1)

- $L_1 \cup L_2$
  - Example:

    $L_1 = \{\varepsilon, a, b, c, bc, ca\}$

    $L_2 = \{ba, bb, bc, ca, cb, cc\}$

    $L_1 \cup L_2 = \{\varepsilon, a, b, c, ba, bb, bc, ca, cb, cc\}$
- $L_1 \cap L_2$
  - Example: $L_1 \cap L_2 = \{bc, ca\}$

Set operations (2)

- $L_1 \setminus L_2$ (or $L_1 - L_2$)
  - Generally used when $L_2 \subseteq L_1$
  - Example:

    $L_1 = \{$ ba, bb, bc, ca, cb, cc $\}$

    $L_2 = \{$ bc, ca $\}$

    $L_1 \setminus L_2 = \{$ba, bb, cb, cc$\}$
- $L^c = A^* \setminus L$
  - A is the alphabet over which L is defined
  - Example: L_1^c = set of all strings on $\{a,b,c\}^*$ except the strings of length 2 that start with a 'b' or a 'c'

Concatenation

- $L_1 \cdot L_2$ (or L_1L_2)$=\{x\cdot y \mid x\in L_1,\ y\in L_2\}$
  - Remark: '$\cdot$' is <u>not commutative</u>
  - $L_1 \cdot L_2 \neq L_2 \cdot L_1$
- Example

  $L_1= \{\varepsilon$, a, b, c, bc, ca$\}$

  $L_2= \{$ba, bb, bc, ca, cb, cc$\}$

  $L_1L_2 = \{$ba, bb, bc, ca, cb, cc, aba, abb, abc, aca, acb, acc, bba, bbb, bbc, bca, bcb, bcc, cba, cbb, cbc, cca, ccb, ccc, bcba, bcbb, bcbc, bcca, bccb, bccc , caba, cabb, cabc, caca, cacb, cacc$\}$

Power

- L^n is obtained by concatenating L with itself n times
  - $L^0 = \{\varepsilon\}$
  - $L^i = L^{i-1} \cdot L$
- Examples:
  - $L^2 = L \cdot L$
  - $L^3 = L \cdot L \cdot L$
  - $L^4 = L \cdot L \cdot L \cdot L$
  - …
- Remark: '$\cdot$' is associative

Kleene closure

- $L^* = \bigcup_{n=0}^{\infty} L^n$

- $L^+ = \bigcup_{n=1}^{\infty} L^n$

  hence $\quad L^* = L^+ \cup L^0 = L^+ \cup \{\varepsilon\}$

  and $\quad L^+ = L \cdot L^*$

  L^* and L^+ coincide iff $\varepsilon \in L$

- Remark: $\{\varepsilon\} \neq \varnothing$

  $\{\varepsilon\} \cdot L = L$

  $\varnothing \cdot L = \varnothing$

What do formal languages represent?

- A language is a <u>set of strings</u>
  - $L_1 = \{bc, ca\}$
  - $L_2 = \{ba, bb, bc, ca, cb, cc\}$
  - $L_3 = \{x \in \{a,b\}^* \mid x = ay \wedge y \in \{a,b\}^*\}$
- How can sets of strings be applied in computer science?
  - Formal languages are not only mere mathematical representations

Languages in CS

- A language is a way of <u>representing</u> or <u>communicating</u> information
  - Not just meaningless strings
- There are many kinds of languages
  - Natural languages
  - Programming languages
  - Logic languages
  - …

Example (1)

- Consider the following languages:
  - L_1: set of "Word@Mac" documents
  - L_2: set of "Word@PC" documents
- Operations:
  - L_1^c is set of documents that are not compatible with "Word@Mac"
  - $L_1 \cup L_2$ is the set of documents that are compatible with either Mac or PC
  - $L_1 \cap L_2$ is the set of documents that are compatible with both Mac and PC

Example (2)

- Consider the following languages:
  - L_1: set of e-mail messages
  - L_2: set of spam messages
- Operations:
  - L_1-L_2 implements a filter

Languages in practice

- A language can represent
  - Computations
  - Documents
  - Part of documents
  - Programs
  - Multimedia
- Operations on languages create new classes of languages