Alphabets and Languages: the mathematics of strings

Strings and symbols

- An *alphabet* is a finite set of *symbols*, e.g., the binary or Roman alphabet. We denote an arbitrary alphabet by Σ
- A string over an alphabet is a finite sequence of symbols from the alphabet.
- The *empty string* is the string with no symbols and is denoted ϵ.
- The set of all strings, including the empty string, over an alphabet is denoted Σ^*.
    - What is the cardinality of Σ^*?
- The *length* of a string is its length as a sequence.
    - There is only one string of length 0. What is it?
- The length of a string w is denoted $|w|$. The symbol in the ith position is denoted w_i. We say that symbol w_i occurs in position i. A symbol may have more than one occurrence in a string.

Operations and relations on strings

- The operation of *concatenation* takes two string x and y and produces a new string xy by putting them together end to end. The string xy is called the *concatenation* of x and y.
    - Concatenation is an associative operation. So we will write, e.g., xyz for $(xy)z$ or $x(yz)$
- A string v is a *substring* of a string w iff there are strings x and y such that $w = xvy$. If $y = \epsilon$ then v is a *suffix* of w. If $x = \epsilon$ then v is a *prefix* of w.
- We write x^n for the string obtained by concatenating n copies of x.
- The *reversal* of a string w, denoted w^R is the string w "written backwards".

Languages: Sets of strings

- A *language* is set of strings over an alphabet.
- We may apply set operations like union, intersection, and set difference to languages.
- The *complement* of a language A is $\Sigma - A$, and is denoted $\bar{A}$ if Σ is understood.
- If L_1 and L_2 are languages over Σ their *concatenation* is $L = L_1 \cdot L_2$ or L_1L_2 where
$$L = \{w \in \Sigma^* : w = xy \text{ for some } x \in L\_1 \text{ and } y \in L\_2 \}$$
- The *Kleene star* of a language L, denotedL^* is the set of all strings obtained by concatenating **zero** or more strings from L. Thus,
$$L^* = \{w \in \Sigma^* : w = w_1 w_2 \dots w_k \text{ for some } k \geq 0\}$$
- Examples: The star of Σ is Σ^*; The star of $\emptyset$ is $\{\epsilon\}$
- L^+ denotes LL^* and is the *closure* of L under concatenation. That is, it is the smallest language that includes L and all strings that are concatenations of strings in L.

Representing a language with a finite specification

- The vast majority of languages over a finite alphabet cannot be represented by a finite specification.

- Why not?
    - The set Σ^* of strings over Σ is *countably infinite*, (i.e., we can construct a bijection $f: \mathbb{N} \to \Sigma^*$ (Exercise: Show that this remains true even if Σ is not finite, but countably infinite.)
    - A specification for a language is given by a string over a finite alphabet. Therefore, the set of specifications is a subset of Γ^* for some finite Γ and is countably infinite, or even finite.
    - But the set of possible languages is the set of subsets of Σ^*, i.e., it is the power set of a countably infinite set. It has size 2^{Σ^*} and is therefore uncountably infinite (Cantor's argument.)
- What languages can we specify? This is the primary question we will address in this course

Languages and Problems

Recall from the first lecture that we said we will be concerned with *computational solutions* to *problems.* A problem is a mapping from *problem instances* to YES, NO. Languages may be viewed as an abstract representation of problems. For a problem P, the associated language is

$$L_P = \{x \in \Sigma^*: x \text{ is a YES instance of } P\}$$

So studying "specifiable" languages is analogous to studying "solvable" problems.