Formal Language

COMP3220 – Principle of Programming Languages

Zhitao Gong

2016 Spring

Outline

Why Study Formal Language

Connected to many other branches of languages

Useful in many application in computer system, particularly in programming languages and compilers

Stable the basics have not changed much in the last thirty years!

Term Overloading

Most specialized field add *crisp* and *rigorous* definitions for words whose common meaning is *fuzzy* and *intuitive*.

- Algebraists redefine the words *group*, *ring*, and *field*.
- Entomologists have precise meanings for *bug* and *fly*.
- We overload words like *alphabet*, *string* and *language*!!

Outline

Definition

Alphabet Any *finite* set of *symbols*.

- $\{0, 1\}$ – binary alphabet
- $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ – decimal alphabet
- ASCII, Unicode – machine text alphabet
- $\{a, b\}$ – arbitrary alphabet
- $\{\}$ – a *legal* but *trivial* and *uninteresting* alphabet.

We use Σ to denote the alphabet, e.g., $\Sigma = \{a, b\}$.

Symbols Are Meaningless

E.g., $\Sigma = \{0, 10, 110, 100, \dots\}$.

- Usually we describe *interpretively*, e.g., "the set of even binary numbers".
- However, our goal is to describe them *rigorously*, i.e., avoiding any interpretations, e.g., "the set of strings of 0's and 1's that end in 0".

The basic principle is

- We don't define what a symbol is, and
- we don't ascribe meaning to symbols.

Outline

Definition

String A finite sequence of zeros or more symbols.

- Length of a string: $|aabb| = 4$.
- A *string over the alphabet* Σ means the symbols of the string are in Σ. E.g., the set of all strings of length 2 over $\Sigma = \{a, b\}$ is $\{aa, ab, ba, bb\}$.

Variables We use a *variable* to stand for a string, e.g., $x = abc$.

In formal language, we rely on context and naming conventions to tell them apart.

Empty String

We use ϵ to denote *empty string*. E.g., ="" in C++.

- Length of ϵ is zero, i.e., $|\epsilon| = 0$.
- Empty string and empty alphabet
  - $\{\} \neq \epsilon$
  - $\{\} \neq \{\epsilon\}$

Concatenation

Concatenation The concatenation of two strings x and y is a *string* containing all the symbols of x in order, followed by all the symbols of y in order. Usually denoted by writing two strings (or string variables) together.

- If $x = ab$ and $y = def$, then the concatenation of x and y is written as $xy = abdef$.
- For any x, $\epsilon x = x\epsilon = x$.

Exponent

Exponent n means concatenating a string to itself n times.

E.g., if $x = ab$, then

- $x^0 = \epsilon$
- $x^1 = ab$
- $x^2 = abab$, etc.

We may use parentheses (or some other form in case Σ contains ()) to for *group exponentials*. E.g., $(ab)^4 = abababab$.

Outline

Definition

Language A set of ~~finite~~ strings over some fixed alphabet.

- Language is not restricted to *finite* set which is usually uninteresting.
- All our alphabets and strings are *finite*, however most of the interesting languages are *infinite*.

Kleene Star

Kleene Closure of an alphabet Σ, denoted as Σ^*, is the *language* of all strings over Σ.

- $\{a\}^* = \{\epsilon, a, aa, aaa, \dots\}$, the set of all strings of zero or more a's, i.e., .
- $\{a, b\}^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, \dots\}$, the set of all strings of zero or more symbols, each of which is either a or b.
- Unless $\Sigma = \{\}$, Σ^* is infinite.

$x \in \Sigma^*$ means x is a string over Σ.

Define A Language

Set-builder Notation Describe a set by stating the properties that its members must satisfy. It is a simple way to define a language.

- $\{x \in \{a, b\}^* \mid |x| \leq 2\} = \{\epsilon, a, b, aa, bb, ab, ba\}$
- $\{xy \mid x \in \{a, aa\} \text{ and } y \in \{b, bb\}\} = \{a, abb, aab, aabb\}$
- $\{x \in \{a, b\}^* \mid x \text{ contains one } a\text{'s and two } b\text{'s}\} = \{abb, bab, bba\}$
- $\{a^n b^n \mid n = 1, 2, 3, \dots\} = \{ab, aabb, aaabbb, \dots\}$

Formal Language Classes

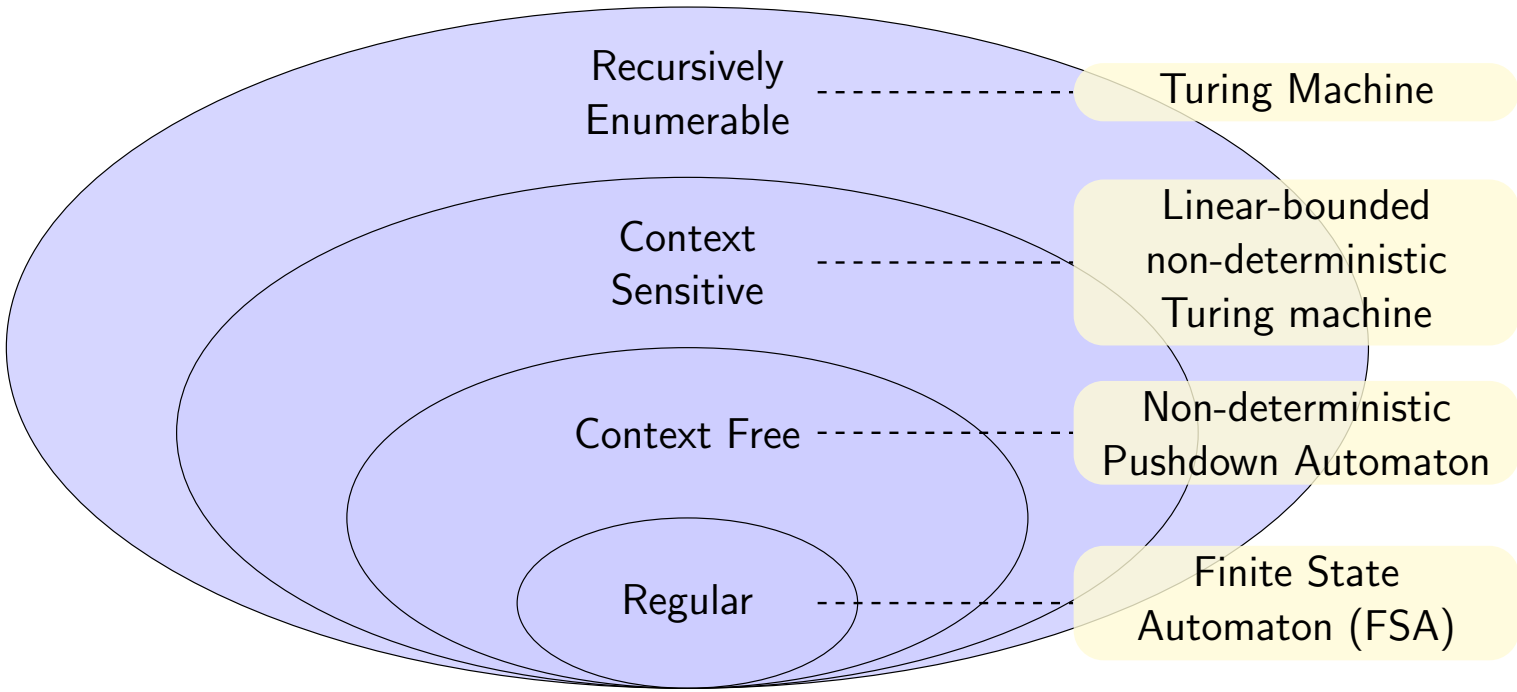

Figure: Chomsky Hierarchy