Lecture 2 — Languages and Grammars

CSE205: Theory of Computation

Minseok Jeon

2026 Fall

Alphabet

Definition

An *alphabet* is a finite, non-empty set of symbols, denoted Σ.

Examples:

1. $\Sigma = \{0, 1\}$: the binary alphabet.
2. $\Sigma = \{\boldsymbol{a}, \boldsymbol{b}, \ldots, \boldsymbol{z}\}$: the set of all lowercase letters.
3. The set of all ASCII characters.

String

Definition

A *string* is a finite sequence of symbols chosen from an alphabet.

Examples:

1. $\Sigma = \{0, 1\}$: $\boldsymbol{\varepsilon}$, 0, 1, 00, 01, ...
2. $\Sigma = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$: $\boldsymbol{\varepsilon}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, $\boldsymbol{ab}$, $\boldsymbol{bc}$, ...

Notations:

- $\boldsymbol{\varepsilon}$: the empty string
- $\boldsymbol{wv}$: the concatenation of $\boldsymbol{w}$ and $\boldsymbol{v}$
- $\boldsymbol{w}^R$: the reverse of $\boldsymbol{w}$
- $|\boldsymbol{w}|$: the length of string $\boldsymbol{w}$
- $\boldsymbol{w} = \boldsymbol{vu}$: $\boldsymbol{v}$ is a *prefix* and $\boldsymbol{u}$ a *suffix* of $\boldsymbol{w}$

Powers of an Alphabet

- Σ^k: the set of strings (over Σ) of length k
- The set of all strings over Σ:

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots = \bigcup_{k \geq 0} \Sigma^k$$

- The set of all non-empty strings over Σ:

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \cdots = \bigcup_{k \geq 1} \Sigma^k$$

Note that $\Sigma^0 = \{\varepsilon\}$, so $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$.

Language

Definition

A *language* L is a set of strings, i.e., $L \subseteq \Sigma^*$ (equivalently, $L \in 2^{\Sigma^*}$).

When $\Sigma = \{0, 1\}$:

$L_1 = \{0,\ 00,\ 001\}$

$L_2 = \{0^n 1^n \mid n \geq 0\}$

$L_3 = \{\varepsilon,\ 01,\ 10,\ 0011,\ 0101,\ 0110,\ 1001,\ \ldots\}$ (equally many 0's and 1's)

$L_4 = \{10,\ 11,\ 101,\ 111,\ 1011,\ \ldots\}$ (the primes in binary)

Language Operations

- union, intersection, difference: $L_1 \cup L_2, \quad L_1 \cap L_2, \quad L_1 - L_2$
- reverse: $L^R = \{w^R \mid w \in L\}$
- complement: $\overline{L} = \Sigma^* - L$
- concatenation of L_1 and L_2:

$$L_1 L_2 = \{xy \mid x \in L_1 \wedge y \in L_2\}$$

Language Operations: Power and Closure

Power:

$$L^0 = \{\varepsilon\}, \qquad L^n = L^{n-1}L \quad (n \geq 1)$$

Closures:

$$L^* = L^0 \cup L^1 \cup L^2 \cup \cdots = \bigcup_{i \geq 0} L^i$$

$$L^+ = L^1 \cup L^2 \cup L^3 \cup \cdots = \bigcup_{i \geq 1} L^i$$

L^* is the *Kleene star* (or star closure) of L, and L^+ is the *positive closure*.

Exercises

1. Consider $L = \{a^n b^n \mid n \geq 0\}$. Show that
   1.1 $L^2 = \{a^n b^n a^m b^m \mid n, m \geq 0\}$
   1.2 $L^R = \{b^n a^n \mid n \geq 0\}$
2. Prove that $(uv)^R = v^R u^R$ for all $u, v \in \Sigma^*$.

Exercise 2 (Solution)

Reverse is defined inductively on strings:

$$\varepsilon^R = \varepsilon, \qquad (aw)^R = w^R a \quad (a \in \Sigma,\ w \in \Sigma^*).$$

Proof by induction on $|u|$.

- Base case ($u = \varepsilon$): for every $v \in \Sigma^*$, $(\varepsilon v)^R = v^R = v^R \varepsilon = v^R \varepsilon^R$.
- Inductive case ($u = au'$ with $a \in \Sigma$): assume $(u'v)^R = v^R u'^R$ for all v (I.H.). Then

$$\begin{aligned}
(uv)^R = (au'v)^R &= (u'v)^R a && \text{(def. of reverse)} \\
&= (v^R u'^R) a && \text{(I.H.)} \\
&= v^R (u'^R a) && \text{(concatenation is associative)} \\
&= v^R (au')^R = v^R u^R. && \text{(def. of reverse)}
\end{aligned}$$

□

Exercise 1 (Solution)

(1.1) By definition of power and concatenation, $L^2 = LL = \{xy \mid x \in L \wedge y \in L\}$.

- ($\subseteq$) Let $w \in L^2$, so $w = xy$ with $x, y \in L$. Then $x = a^n b^n$ and $y = a^m b^m$ for some $n, m \geq 0$, hence $w = a^n b^n a^m b^m$.
- ($\supseteq$) For any $n, m \geq 0$ we have $a^n b^n \in L$ and $a^m b^m \in L$, so $a^n b^n a^m b^m \in LL = L^2$.

Note that n and m are chosen *independently*: e.g., $ab\,aabb \in L^2$, so $L^2 \neq \{a^n b^n a^n b^n \mid n \geq 0\}$.

(1.2) By definition of reverse, $L^R = \{w^R \mid w \in L\} = \{(a^n b^n)^R \mid n \geq 0\}$. By Exercise 2 with $u = a^n$ and $v = b^n$,

$$(a^n b^n)^R = (b^n)^R (a^n)^R = b^n a^n,$$

since reversing a string of identical symbols gives the same string. Hence $L^R = \{b^n a^n \mid n \geq 0\}$.

Grammar

Definition

A *grammar* G is a quadruple $G = (V, T, S, P)$:

- V: a finite set of *variables* (non-terminal symbols)
- T: a finite set of *terminal* symbols
- $S \in V$: the *start variable*
- P: a finite set of *productions*. A production has the form

$$x \to y, \qquad x \in (V \cup T)^+, \quad y \in (V \cup T)^*$$

Example:

$$G = (\{S\},\ \{a, b\},\ S,\ \{S \to aSb,\ \ S \to \varepsilon\})$$

Applying Productions to Strings

- A production $x \to y$ means: replace x by y. Applying it to the string

$$w = uxv, \qquad u, v \in (V \cup T)^*$$

  gives

$$z = uyv.$$

  In this case, we write $w \Rightarrow z$.

- $w \Rightarrow^* z$ iff $w = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n = z$ for some $n \geq 1$ (i.e., $\Rightarrow^*$ is the reflexive-transitive closure of $\Rightarrow$; in particular, $w \Rightarrow^* w$).

Example: Derivations

$$G = (\{S\},\ \{a, b\},\ S,\ \{S \to aSb,\quad S \to \epsilon\})$$

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb,$ so $S \Rightarrow^* aabb$

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb,$ so $S \Rightarrow^* aaabbb$

A Grammar Specifies a Language

Definition

Let $G = (V, T, S, P)$ be a grammar. Then the set

$$L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$$

is the *language generated by* G.

- If $w \in L(G)$, the sequence

  $$S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n \Rightarrow w \qquad (n \geq 0)$$

  is a *derivation* of the sentence w.
- $S, w_1, w_2, \ldots, w_n$ are called *sentential forms*.

Example: $L(G) = \{a^n b^n \mid n \geq 0\}$

$$G = (\{S\},\ \{a, b\},\ S,\ \{S \to aSb,\quad S \to \epsilon\})$$

The language of G is

$$L(G) = \{\epsilon,\ ab,\ aabb,\ aaabbb,\ aaaabbbb,\ \ldots\} = \{a^n b^n \mid n \geq 0\}.$$

Summary

- Alphabets, strings, and string operations.
- Languages and their operations: union, concatenation, power, Kleene star and positive closure.
- Grammars $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{T}, \boldsymbol{S}, \boldsymbol{P})$, derivations, and the language $\boldsymbol{L}(\boldsymbol{G})$ generated by a grammar.