MATH 2301 — 19 Sep 2022

* Assignment 6 due Friday

* Before break: Möbius inversion

* Today: "Machines" ← regular expressions

- An alphabet is a finite set of "letters" or "symbols", typically denoted Σ (sigma).

E.g. ① $\Sigma = \{a, b, c, \dots, z\}$

② Default example is $\Sigma = \{0, 1\}$

- A word or string on Σ is a finite ordered list of elements of Σ, not necessarily distinct.
~~Eg.~~ Typically denoted $w = a_1 a_2 a_3 \cdots a_n$, where each $a_i \in \Sigma$.

E.g. $w = 000$, $w = 1101$, $w = 011$, etc.

Note: The empty word is denoted ε.
(It is a valid word.)
[We always assume that $\varepsilon \notin \Sigma$.]

- Def: A language on Σ is a set of words on Σ; not necessarily finite.

Def: Let Σ be an alphabet.
Σ^* is the set of all possible words on Σ.
Then, a language L is simply a subset $L \subseteq \Sigma^*$.

* Examples

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

$L = \{10, 01\}$

$L = \{\varepsilon, 00\}$

$L = \{\varepsilon\}$ ← has one word, namely ε

$L = \phi$ ← has no words.

(finite languages)

$L =$ the set of all strings that begin with a 0
$= \{0, 00, 01, 000, 001, 010, \dots\}$ ← infinite

$L =$ the set of all strings that don't contain a "1".
$= \{\varepsilon, 0, 00, 000, \dots\}$

* Rmk: There need not always be a pattern/rule to what elements do/don't belong to a language.
[We'll use regular expressions to codify some languages that follow certain rules.]

* Basic operations on strings & languages

Fix some alphabet Σ.

- Concatenation (strings): If v, w are strings

$$\left.\begin{array}{l} v = a_1 \dots a_k \\ w = b_1 \dots b_\ell \end{array}\right\} \text{ then } vw = \text{concatenation} = a_1 \dots a_k b_1 \dots b_\ell .$$

- Concatenation (languages): If L_1 & L_2 are languages, then

$L_1 \circ L_2$ = concatenated language

$$= \{ vw \in \Sigma^* \mid v \in L_1 \text{ and } w \in L_2 \}$$

E.g.: $L_1 = \{10, 01\}$ & $L_2 = \{\varepsilon, 00\}$

$$L_1 \circ L_2 = \{ \underset{10}{\underset{\parallel}{10\varepsilon}}, \underset{01}{\underset{\parallel}{01\varepsilon}}, 1000, 0100 \}$$

- Union & intersection (languages)

If L_1, L_2 are languages, we can write $L_1 \cup L_2$, $L_1 \cap L_2$ for the union & intersection respectively.

E.g. (L_1 & L_2 as before)

$L_1 \cup L_2 = \{10, 01, \varepsilon, 00\}$

$L_1 \cap L_2 = \phi$

- Star (of a language)

If L is a language, then L^* is the set of all possible successive concatenations (0 or more times) of possibly different words in L.

$$L^* = \{ w_1 w_2 \cdots w_k \mid w_i \in L \} \cup \{\varepsilon\}$$

E.g.: $L = \{10, 01\}$

$L^* = \{ \varepsilon, 10, 01, 1010, 1001, 0101, 0110, 101010, 101001, \dots \}$

$$\left.\begin{array}{l} L = \{\varepsilon\} \\ L^* = \{\varepsilon\} \\ \\ L = \phi \\ L^* = \{\varepsilon\} \end{array}\right\}$$

Other than these examples, L^* is always infinite [check!]

Note:

$$L^* = \{\varepsilon\} \cup (L) \cup (L \circ L) \cup (L \circ L \circ L) \cup \dots$$

* <u>Lexicographic / dictionary order on Σ^*</u>

Choose & fix a total order on Σ.

Now we can order Σ^* (total order) as follows:

Let $v, w \in \Sigma^*$

(1) If length(v) < length(w) then $v < w$
(and vice-versa)

(2) If length(v) = length(w), then follow dictionary order.

i.e, if $v = a_1 \cdots a_n$
$w = b_1 \cdots b_n$, then let i be the first index such that $a_i \neq b_i$.

If $a_i < b_i$, we say $v < w$
if $a_i > b_i$, we say $v > w$.

If there is no such i, then $v = w$.

The same order is inherited by any $L \subseteq \Sigma^*$.