Welcome to Intro to CS Theory

Introduction to CS Theory:
- formalization of computation
- various models of computation (increasing difficulty/power)
- what can / cannot be done ?

Why a theory course ?
- relevant to practice (grammars for programming languages, finite automata & regular expressions for pattern matching of strings, NP-completeness to determine required time complexity - e.g. for cryptography)
- problem solving skills independent of current technology (specific programming languages, etc.), ability to express ideas clearly, succinctly, and correctly
Introduction

Automata Theory
- mathematical models of computation

Computability Theory
- what can be computed?

Complexity Theory
- which problems are computationally hard / easy?

Need math background
- review Chapter 0
- discrete math quiz next class
**Alphabet** - non-empty finite set of *symbols*, typically denoted by $\Sigma$ or $\Gamma$, e.g.

$$\Sigma_1 = \{ 0, 1 \}, \quad \Sigma_2 = \{ a, b, c, d \}, \quad \Gamma = \{ \#, $, 0, 1, 2 \}$$

$$\Sigma_3 = \{ 0 \}$$

$\Sigma_4 = \mathbb{N}$ not OK (not finite)

$\Sigma_5 = \{ \} = \emptyset$ not OK (empty)
Strings and Languages

**Alphabet** - non-empty finite set of *symbols*, typically denoted by $\Sigma$ or $\Gamma$, e.g.

$$\Sigma_1 = \{ 0,1 \}, \quad \Sigma_2 = \{ a,b,c,d \}, \quad \Gamma = \{ #,\$,0,1,2 \}$$

**String** over an alphabet - a finite sequence of symbols from the alphabet, e.g.

$$w_1 = 00101 \text{ over } \Sigma_1, \quad w_2 = \text{badcab} \text{ over } \Sigma_2$$

- $\Sigma_3 = \{ \text{not a string bec. it is infinite} \}$
- $w_4 = \text{ not a string bec. it is infinite }$
- $w_5 = \epsilon$ special symbol for the empty (null) string
- $w_6 = 1$
Alphabet - non-empty finite set of symbols, typically denoted by $\Sigma$ or $\Gamma$, e.g.

$\Sigma_1 = \{0,1\}$, $\Sigma_2 = \{a,b,c,d\}$, $\Gamma = \{\#,\$,0,1,2\}$

String over an alphabet - a finite sequence of symbols from the alphabet, e.g.

$w_1 = 00101$ over $\Sigma_1$, $w_2 = \text{badcab}$ over $\Sigma_2$

The length of a string $w$ over $\Sigma$ (the number of symbols in $w$) is denoted $|w|$. 

$|w_1| = 5$  $|\varepsilon| = 0$  $|w_2| = 6$

The string with no symbols is called the empty string and denoted $\varepsilon$. 
Strings and Languages

Operations on strings (let \( w = w_1w_2...w_n \)): 

- **reverse**: \( w_R = w_nw_{n-1}...w_1 \) 
  \[ w^R = \underbrace{3b2q}_{\forall i, j : 1 \leq i < j \leq n} \]

- **substring**: \( w_iw_{i+1}...w_j \) 
  e.g. \( zb, b3, 3, qzb, qzb3, \varepsilon \) 

- **concatenation of \( w \) with a string \( z = z_1z_2...z_m \)**: 
  \[ wz = w_1w_2...w_nz_1z_2...z_m \] 
  \[ w_2 = qzb3qzb3qzb3 = (qzb3)^3 \]

- **\( w^k \)** means concatenation of \( k \) copies of \( w \) 

\[ k = 3 \quad w^3 = qzb3qzb3qzb3 = (qzb3)^3 \]

\[ w^2 = qzb3qzb3 \]

\[ w^1 = qzb3 = w \]

\[ w^0 = \varepsilon \]
Strings and Languages

Operations on strings (let $w = w_1w_2\ldots w_n$):

- **reverse**: $w^R = w_nw_{n-1}\ldots w_1$
- **substring**: $w_iw_{i+1}\ldots w_j$
- **concatenation of $w$ with a string $z = z_1z_2\ldots z_m$**:
  $$wz = w_1w_2\ldots w_nz_1z_2\ldots z_m$$
- **$w^k$** means concatenation of $k$ copies of $w$
- **lexicographic ordering** of strings: first by length, then “alphabetically,” e.g. for $\Sigma = \{0,1\}$:
  $\varepsilon,0,1,00,01,10,11,000,\ldots$

| strings of length | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |...
|-------------------|---|---|---|---|---|---|---|----|---
| count             | 8 | 16| 32| 64| 128|...|...

...
Language: a set of strings over an alphabet $\Sigma$, e.g.

$L_1 = \{ a, ab, bab \}$ \hspace{1cm} $|L_1| = 3$ \hspace{1cm} 3 strings over $\Sigma_1 = \{a, b\}$ or

$L_2 = \emptyset$

$L_3 = \{ \epsilon \}$

$L_4 = \{ w \text{ over } \{0,1\} \mid w \text{ contains more 1's than 0's} \}$

contains, e.g. $110, 1, 11$ \hspace{1cm} $|L_4| = \infty$

$\epsilon \in L_4$

is $L_2 \neq L_3$ \hspace{1cm} $|L_1| = 0$

$|L_3| = 1$
Operations on languages:
- typical set operations: $\cup$, $\cap$, etc.

\[
\{aa, bab, a\} \cup \{a, bb\} = \\
= \{a, aa, bb, bab\}
\]
Strings and Languages

Operations on languages:

- Typical set operations: $\cup$, $\cap$, etc.

- Concatenation: $L_1 \cdot L_2 = \{ w_1w_2 \mid w_1 \in L_1, w_2 \in L_2 \}$

\[
\begin{align*}
\{ab, aa\} \cdot \{b, bb\} &= \{ab, abb, aab, aabb\} \\
\{ab, aa\} \cdot \{\varepsilon\} &= \{ab, aa\} \quad \text{in general, for any } L: \\
\{ab, aa\} \cdot \emptyset &= \emptyset \\
\{ab, aa\} \cdot L &= L \quad \text{for any } L
\end{align*}
\]

Notice: $|L_1 \cdot L_2| \leq |L_1| \cdot |L_2|$

But it is not always equal
Strings and Languages

Operations on languages:

- Typical set operations: $\cup$, $\cap$, etc.

- Concatenation: $L_1 \cdot L_2 = \{ w_1 w_2 \mid w_1 \in L_1, w_2 \in L_2 \}$

- Kleene’s star: $L^* = \bigcup_{k=0}^{\infty} L^k$

\[
\bigcup_{k=0}^{\infty} L^k = L^1 \cup L^2 \cup L^3 \cup \ldots
\]

Consider

$A = \{0\}$

$A^1 = \{0\}$

$A^2 = \{00\}$

$A^3 = \{000\}$

$B = \{0,1\}$

$B^* = \text{the set of all strings over } \{0,1\}$

$\Sigma^* = \text{the set of all strings over } \Sigma$

$C = \{00\}$

$C^* = \text{the set of all even-length strings over } \{0,0\}$

Note: for any $L$: $L^0 = \{\varepsilon\}$

$A^\varepsilon = \text{all strings over } \{0\}$
Operations on languages:

- typical set operations: $\cup$, $\cap$, etc.
- concatenation: $L_1 \cdot L_2 = \{ w_1 w_2 \mid w_1 \in L_1, w_2 \in L_2 \}$
- Kleene’s star: $L^* = \bigcup_{k=0}^{\infty} L^k$

Note: a language: $L \subseteq \Sigma^*$
Operations on languages:

- typical set operations: \( \cup, \cap, \text{etc.} \)
- concatenation: \( L_1.L_2 = \{ w_1w_2 \mid w_1 \in L_1, w_2 \in L_2 \} \)
- Kleene’s star: \( L^* = \bigcup_{k=0}^{\infty} L^k \)
- reverse: \( L^R = \{ w^R \mid w \in L \} \)

\[\{a, bba, bab\}^R = \{a, abb, bab\}\]

Note: a language: \( L \subseteq \Sigma^* \)