# Fundamental Concepts

# Sets

- $a \in S$: $a$ is an **element** of $S$; $a$ is a **member** of $S$.

  **Example** $1 \in \{1, 2, 3\}$.

- $S \subseteq T$: $S$ is a **subset** of $T$; $S$ is **contained** in $T$. This means that every member of $S$ is a member of $T$.

  **Example** $\{1, 2\} \subseteq \{1, 2, 3\}$, $\{1, 2\} \subseteq \{1, 2\}$, and $\{1, 4\} \not\subseteq \{1, 2, 3\}$.

- $S \subset T$: $S$ is a **proper subset** of $T$; $S$ is **properly contained** in $T$. This means that $S \neq T$ and $S \subseteq T$.

  **Example** $\{1, 2\} \subset \{1, 2, 3\}$ and $\{1, 2\} \not\subset \{1, 2\}$.

- $\emptyset$ is the **empty set**, the set without elements.

- $2^S$ is a **power set** of $S$; i.e., the set of all subsets of $S$.

# Set Operations

- $S \cap T$: the **intersection** (**meet**) of $S$ and $T$; the set of all common members between $S$ and $T$.

  **Example** $\{1,2,3\} \cap \{1,2,4\} = \{1,2\}$.

- $S \cup T$: the **union** (**join**) of $S$ and $T$; the set of all members of $S$ or $T$.

  **Example** $\{1,2,3\} \cup \{1,2,4\} = \{1,2,3,4\}$.

- $S \setminus T$: the **set difference** of $S$ and $T$; i.e., the set consisting of all members of $S$ that are nonmembers of $T$.

  **Example** $\{1,2,3\} \setminus \{1,2,4\} = \{3\}$.

- If $T \subseteq S$, we write $S - T$ to mean $S \setminus T$.

  **Example** $\{1,2,3\} - \{1,2\} = \{3\}$.

- $S \triangle T$: the **disjoint union** of $S$ and $T$. $S \triangle T = (S \setminus T) \cup (T \setminus S)$.

  **Example** $\{1,2,3\} \triangle \{1,2,4\} = \{3,4\}$.

# Set Operations (cont'd)

- $S \times T$: the **Cartesian product** of $S$ and $T$; i.e., $\{(a, b) \mid a \in S$ and $b \in T\}$.

  **Example** $\{a, b, c\} \times \{1, 2\} = \{(a, 1), (a, 2), (b, 1), (b, 2), (c, 1), (c, 2)\}$.

- $\|S\|$: the **cardinality** of the set $S$; i.e., the number of elements in $S$.

  **Example** $\|\{a, b, c\}\| = 3$.

- Quite often $|\cdot|$ is used for the cardinality.

# Alphabet, Strings, Languages, etc.

- An **alphabet** is any finite set, whose members are called **symbols**.
- A **string (or word) over an alphabet** is a sequence of symbols from the alphabet written one after another.

  **Example** $aba$ is a word over an alphabet $\{a, b\}$
- The **length** of a word $w$, denoted by $|w|$, is the number of symbols in it.

  **Example** If $w = aba$, then $|w| = 3$.
- The **empty string** or **null string**, denoted by $\epsilon$, is the string with no symbols in it.

# Alphabet, Strings, Languages, etc. (cont'd)

- A string $z$ is a **substring** of $w$ if $z$ appears consecutively within $w$.

  **Example** Let $z = 001111010$. Then $1111$ is a substring of $z$ while $11111$ is not.

- The **concatenation** of strings $x$ and $y$ is the string constructed by appending $y$ after $x$.

  **Example** The concatenation of $a = 000$ and $b = 111$ is $000111$.

- A **language** is a collection of strings.
- A **class** is a collection of languages.

# Alphabet, Strings, Languages, etc. (cont'd)

- For an alphabet $\Sigma$, $\Sigma^*$ is the set of all strings over $\Sigma$.
- The **complement** of a language is the collection of all non-members; for a language $L$ over an alphabet $\Sigma$, its complement is $\Sigma^* - L$ and is denoted by $L^c$ or $\overline{L}$.

  **Example** If $\Sigma = \{a, b\}$ and $L$ is the set of all strings over $\Sigma$ having an even number of $a$'s, then $\overline{L}$ is the set of all strings over $\Sigma$ having an odd number of $a$'s.

# Alphabet, Strings, Languages, etc. (cont'd)

- If $\Sigma$ is a single-letter alphabet with $a$ as its unique symbol, we often write $a^*$ for $\Sigma^*$.
- For a language $L$, $L^*$ is the set of all strings constructed by concatenating any strings from $L$ in any order. That is, $L^* = \{\epsilon\} \cup \{x_1 \cdots x_m \mid m \geq 1, x_1, \ldots, x_m \in L\}$.

  **Example** $\{a, ab\}^*$ is the set of all strings $w$ over $a$ and $b$ such that either $w$ is empty or ($w$ begins with an $a$ and has no $bb$ as a substring).

# Boolean Logic

A **Boolean variable** takes on one of $0$ (FALSE) and $1$ (TRUE). The **negation** of $x$, denoted by $\overline{x}$ or $\neg x$, is $1 - x$.

We will be using six binary Boolean operators:

| $(x, y)$ | $(0, 0)$ | $(0, 1)$ | $(1, 0)$ | $(1, 1)$ |
|---|---|---|---|---|
| $\wedge$ | $0$ | $0$ | $0$ | $1$ |
| $\vee$ | $0$ | $1$ | $1$ | $1$ |
| $\rightarrow$ | $1$ | $1$ | $0$ | $1$ |
| $\leftarrow$ | $1$ | $0$ | $1$ | $1$ |
| $\leftrightarrow$ | $1$ | $0$ | $0$ | $1$ |
| $\oplus$ | $0$ | $1$ | $1$ | $0$ |

# Boolean Logic (cont'd)

A **predicate** is a **function** whose **range** is { TRUE, FALSE }. A **relation** is a predicate whose number of arguments is fixed to a constant.

Properties of binary relation $R$ over domain $D$.

- **Reflexive**: For all $x \in D$, $xRx$.
- **Symmetric**: For all $x, y \in D$, $xRy \leftrightarrow yRx$.
- **Transitive** For all $x, y, z \in D$, $xRy \wedge yRz \rightarrow xRz$.

An **equivalence relation** is a binary relation that is reflexive, symmetric, and transitive

# Proof by Induction

A method for proving a statement $P$. Divide the statement $P$ into cases $P(n), n = a, a+1, a+2, \ldots$ For the base case, prove $P(a)$. For the induction step, assume that $P(n)$ is true for all values of $n \leq k$ and show that $P(k+1)$ holds.

# Graphs

A **graph** consists of **nodes (vertices)** and **edges**. A **path** is a sequence of edges (or a sequence of nodes) that connects from a node to another. A **tree** is a **connected, undirected graph** without **cycles**.