Please check your attendance using Blackboard!

Lecture 1
Mathematical Preliminaries and Notations

COSE215: Theory of Computation

Seunghoon Woo

Fall 2023

Contents

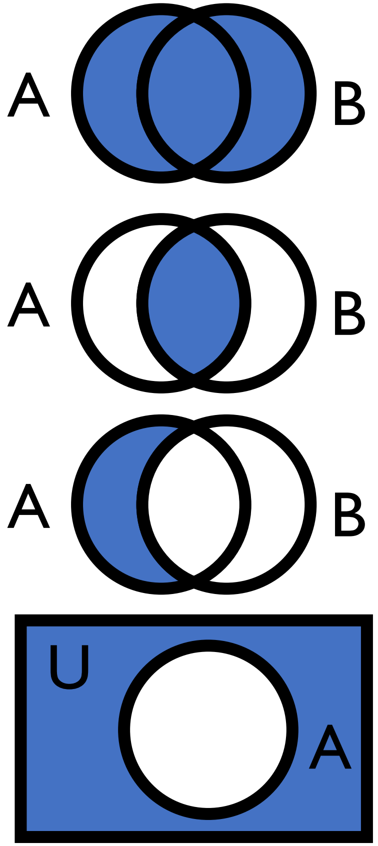
- **Basic concepts of**
  - Sets
  - Functions
  - Graphs & Trees
  - Proof techniques
  - Alphabets & Strings
  - Languages & Grammars

Sets

- **A set is a collection of elements**
  - If *x* is an element of set *S*, we can write this as follow
    - $x \in S$
  - A set can be represented by naming all its elements
    - $S = \{x, y, z\}$
  - If the rules of the elements in the set are clear, we can use explicit notation
    - $S = \{k : k > 0, k \text{ is even}\}$
  - A set with no elements is called the empty set (or null set)
    - $\emptyset = \{\}$
  - The size of a finite set is the number of elements in it
    - If $S = \{x, y, z\}$, then $|S| = 3$

Sets

- **Set operations**
  - Union
    - $A \cup B = \{x: x \in A \text{ } or \text{ } x \in B\}$
  - Intersection
    - $A \cap B = \{x: x \in A \text{ } and \text{ } x \in B\}$
  - Difference
    - $A - B = \{x: x \in A \text{ } and \text{ } x \notin B\}$
  - Complementation
    - $\bar{A} = \{x: x \in U, x \notin A\}$



Sets

- **Subset**
  - If every element of A is also an element of B, we write this as
    - $A \subseteq B$
  - If $A \subseteq B$, but B contains an element not in A
    - We say that A is a **proper** subset of B: $A \subset B$
- **Disjoint**
  - If A and B have no common element
    - Then the sets are said to be **disjoint**: $A \cap B = \emptyset$

Sets

- **Powerset**
  - The set of all subsets of a set S is called the powerset of S
    - ❖ Denoted by 2^s
    - ❖ For example, if S is the set {a, b, c}, then its powerset is
      - $2^s = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$
    - ❖ $|2^s| = 2^{|s|}$

Sets

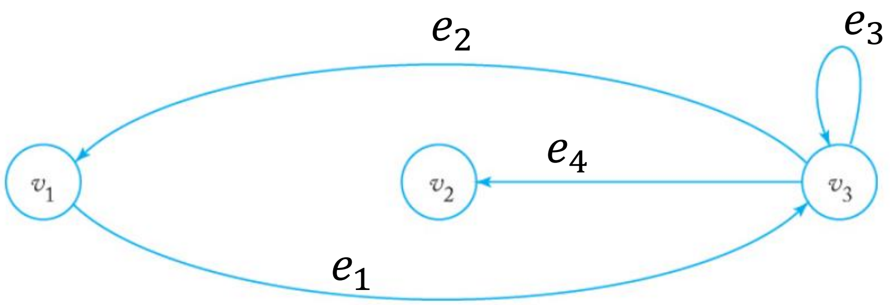
- **Cartesian product**
  - Cartesian product of two sets
    - $A \times B = \{(x, y): x \in A \text{ and } y \in B\}$
    - **Ordered** pairs

Functions

- **Function**
  - Rules for assigning elements in one set to a unique element in another set
    - $f : A \rightarrow B$
    - A = **Domain**
    - B = **Range**
  - If the domain of f is all of A, we say that f is a **total function**
    - Otherwise, f is said to be a **partial function**

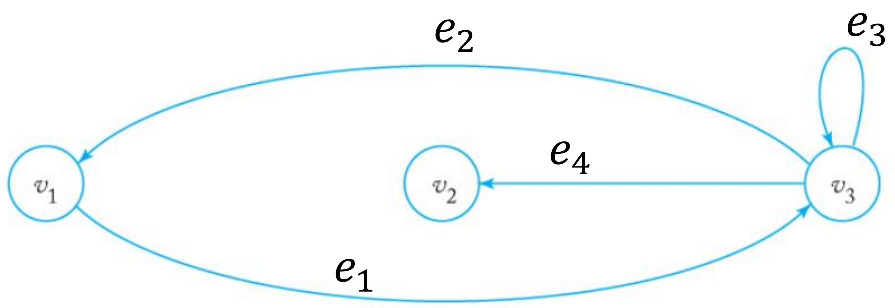
Graphs & Trees

- **Graph**
  - A graph consists of two finite sets: **vertices** and **edges**
    - $G = (V, E)$, where $V = \{v_1, v_2, \dots, v_n\}$ and $E = \{e_1, e_2, \dots, e_m\}$
    - Each edge is a pair of vertices from V
      - $e_i = (v_j, v_k)$
- **Directed graph (digraph)**
  - Associate a direction with each edge



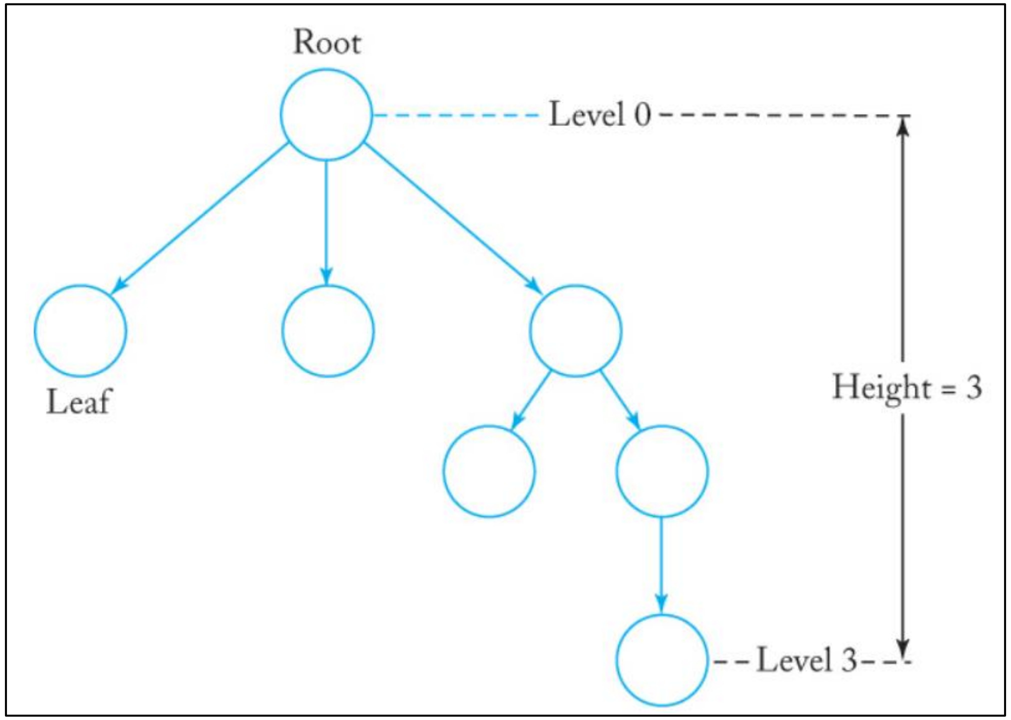
Graphs & Trees

- **Walk**
  - Sequence of edges
- **Path**
  - Walk with no repeated edges
- **Simple path**
  - Path with no vertices repeated
- **Cycle**
  - A walk from v_i to itself with no repeated edges



Graphs & Trees

- **Tree**
  - Directed graph with no cycles
  - One vertex designated as "**root**"
    - ❖ Exactly one path from root to every other vertex
  - **Leaves**
    - ❖ Vertices without outgoing edges
  - **Level**
    - ❖ The number of edges in the path from the root to a vertex
  - **Height**
    - ❖ The largest level number of any vertex



Proof techniques

- **How can we prove the truth of a claim?**
  - Proof by induction
  - Proof by contradiction

Proof techniques

- **Proof by induction**
  - Truth of a few instances => Truth of a number of statements
  - Suppose we want to prove P_1, P_2, … to be true
    - ❖ We first prove that it is true when n = 1 (P_1)
    - ❖ Assuming it is true for n = k (P_k) and showing it is true for n = k+1 (P_{k+1})

=> Then, every Pi is true

Proof techniques

- **Proof by induction: example**
  - A binary tree is a tree in which no parent can have more than two children. Prove that a binary tree of height n has at most 2^n leaves.
    - ❖ $l(n)$: Maximum number of leaves
    - ❖ We want to show that $l(n) \leq 2^n$

Proof techniques

- **Proof by induction: example**
  - A binary tree is a tree in which no parent can have more than two children. Prove that a binary tree of height n has at most 2^n leaves.
    1. When $n = 0,\ l(0) = 1 = 2^0$

Proof techniques

- **Proof by induction: example**
  - A binary tree is a tree in which no parent can have more than two children. Prove that a binary tree of height n has at most 2^n leaves.
    1. When $n = 0,\ l(0) = 1 = 2^0$
    2. Assumption: $l(i) \leq 2^i,\ \text{for } i = 0, 1, \ldots, n$

Proof techniques

- **Proof by induction: example**
  - A binary tree is a tree in which no parent can have more than two children. Prove that a binary tree of height n has at most 2^n leaves.
    1. When $n = 0,\ l(0) = 1 = 2^0$
    2. Assumption: $l(i) \leq 2^i,\ \text{for } i = 0, 1, \dots, n$
    3. To get a binary tree of height n+1 from one of height n, at most, two leaves in place of each previous one
       - $l(n+1) \leq 2l(n)$

Proof techniques

- **Proof by induction: example**
  - A binary tree is a tree in which no parent can have more than two children. Prove that a binary tree of height n has at most 2^n leaves.
    1. When $n = 0,\ l(0) = 1 = 2^0$
    2. Assumption: $l(i) \leq 2^i,\ \text{for } i = 0, 1, \ldots, n$
    3. To get a binary tree of height n+1 from one of height n, at most, two leaves in place of each previous one
       - $l(n+1) \leq 2l(n)$
    4. Therefore, $l(n+1) \leq 2l(n) \leq 2 \times 2^n = 2^{n+1}$

Proof techniques

- **Proof by contradiction**
  - To prove P is true, assume P is false
  - If we arrive at a conclusion that we know is incorrect => P is true
  - E.g., prove that $\sqrt{2}$ is an irrational number
    1. Assume that $\sqrt{2}$ is a rational number: $\sqrt{2} = \frac{n}{m}$ (n, m are integers without a common factor)

Proof techniques

- **Proof by contradiction**
  - To prove P is true, assume P is false
  - If we arrive at a conclusion that we know is incorrect => P is true
  - E.g., prove that $\sqrt{2}$ is an irrational number
    1. Assume that $\sqrt{2}$ is a rational number: $\sqrt{2} = \frac{n}{m}$ (n, m are integers without a common factor)
    2. Then $2m^2 = n^2$, which implies that n is even (let n = 2k)

Proof techniques

- **Proof by contradiction**
  - To prove P is true, assume P is false
  - If we arrive at a conclusion that we know is incorrect => P is true
  - E.g., prove that $\sqrt{2}$ is an irrational number
    1. Assume that $\sqrt{2}$ is a rational number: $\sqrt{2} = \frac{n}{m}$ (n, m are integers without a common factor)
    2. Then $2m^2 = n^2$, which implies that n is even (let n = 2k)
    3. Then $2m^2 = 4k^2$, which implies that m is even => contradict

Proof techniques

- **Proof by contradiction**
  - To prove P is true, assume P is false
  - If we arrive at a conclusion that we know is incorrect => P is true
  - E.g., prove that $\sqrt{2}$ is an irrational number
    1. Assume that $\sqrt{2}$ is a rational number: $\sqrt{2} = \frac{n}{m}$ (n, m are integers without a common factor)
    2. Then $2m^2 = n^2$, which implies that n is even (let n = 2k)
    3. Then $2m^2 = 4k^2$, which implies that m is even => contradict
    4. Hence, $\sqrt{2}$ is an irrational number

Alphabets & Strings

- **Alphabets (Σ)**
  - Finite, non-empty set of symbols
  - E.g., $\Sigma = \{a, b, c\}$
- **Strings**
  - Sequence of symbols
  - E.g., "aaabbb", "abcbca"
  - Empty string λ: $|\lambda| = 0$

Alphabets & Strings

- $\mathbf{\Sigma^*}$
  - A set of strings obtained by concatenating **zero** or more symbols from Σ
  - E.g., if $\Sigma = \{a\}$, then $\Sigma^* = \{\lambda, a, aa, aaa, \dots\}$
- $\mathbf{\Sigma^+}$
  - A set of strings obtained by concatenating **one** or more symbols from Σ
  - E.g., if $\Sigma = \{a\}$, then $\Sigma^+ = \{a, aa, aaa, \dots\}$
  - $\mathbf{\Sigma^+} = \mathbf{\Sigma^*} - \{\lambda\}$

Languages & grammars

- **Language**
  - A set of character strings
  - A subset of Σ^*
  - A string in a language L is called a **sentence** of L
  - E.g., $\Sigma = \{a, b\}$
    - Then $\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \dots\}$
    - $\{a, aa, aaa\}$ is a language for Σ
    - $L = \{a^n b^n : n \geq 0\}$ is also a language for Σ

Languages & grammars

- **Language operations**
  - Union, intersection, and difference of two languages
  - Complementation
    - $\bar{L} = \Sigma^* - L$
  - Reverse
    - $L^R = \{w^R : w \in L\}$
  - Concatenation
    - $L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$
  - Star-closure
    - $L^* = L^0 \cup L^1 \cup L^2 \ldots$ ($L^0 = \{\lambda\}$ and L^i as L concatenated with itself i times)
  - Positive-closure
    - $L^+ = L^1 \cup L^2 \ldots$

Languages & grammars

- **Grammar (G)**
  - A set of rules used to define the structure of the strings in a language
  - G = (V, T, S, P)
    - V: Set of variables (non-empty)
    - T: Set of terminal symbols (non-empty; V and T are disjoint)
    - S: Start variable ($S \in V$)
    - P: Set of productions

Languages & grammars

- **Production rules**
  - Specify how the grammar transforms one string into another
    - ❖ $x \to y$, where $x \in (V \cup T)^+$ and $y \in (V \cup T)^*$
  - Given a string $w = uxv$
    - ❖ If we apply $x \to y$ then a new string z is obtained: $z = uyv$
    - ❖ This is written as $w \Rightarrow z$ (w **derives** z)
  - Shorthand representation
    - ❖ $w \Rightarrow z$ (derives in one step)
    - ❖ $w \overset{+}{\Rightarrow} z$ (derives in one or more steps)
    - ❖ $w \overset{*}{\Rightarrow} z$ (derives in zero or more steps)

Languages & grammars

- **Example grammar**
  - G = ({S}, {a, b}, S, P) with P given by
    - ❖ $S \rightarrow aSb$ and S → λ ($S \rightarrow aSb \mid \lambda$)
  - We can derive the string "aabb"
    - ❖ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$
    - ❖ Therefore, $S \overset{*}{\Rightarrow} aabb$

Languages & grammars

- **Grammar specifies a language**
  - The language of G
    - Set of strings derived from the start symbol of G
    - Denoted by L(G)
    - For the previous example, L(G) can be defined as follows
      - G = ({S}, {a, b}, S, P) with P given by
        - $S \rightarrow aSb$ and $S \rightarrow \lambda$ ($S \rightarrow aSb \mid \lambda$)
      - $L(G) = \{a^n b^n : n \geq 0\}$

Next Lecture

- **Finite automata**
  - Deterministic finite automata (DFA)
  - Nondeterministic finite automata (NFA)

Appendix

- **Equivalence relation**
  - To indicate that a pair (x, y) is in an equivalence relation
    - $x \equiv y$
  - Satisfy three rules
    - Reflexivity rule $x \equiv x$ for all x
    - Symmetry rule if $x \equiv y$, then $y \equiv x$
    - Transitivity rule if $x \equiv y$ and $y \equiv z$, then $x \equiv z$
  - E.g., $x \equiv y$ if and only if $x \bmod 3 = y \bmod 3$
    - $x \bmod 3 = x \bmod 3$
    - $x \bmod 3 = y \bmod 3$ then $y \bmod 3 = x \bmod 3$
    - $x \bmod 3 = y \bmod 3$, and $y \bmod 3 = z \bmod 3$, then $x \bmod 3 = z \bmod 3$