CENG 280 - Chapter 1: Sets, Relations, and Languages

Burak Metehan Tunçel - March 2022

1 Sets

- A **set** is a collection of objects. The objects comprising a set are called its **elements** or **members**.
- A set A is a **subset** of a set B -in symbols, $A \subseteq B$- if each element of A is also an element of B. If A is a subset of B, but A is not the same as B, we say that A is a **proper subset** of B and write $A \subset B$.
- **Union:** The union of two sets is that set having as elements the objects that are elements of at least one of the two given sets, and possibly of both. We use the symbol $\cup$ to denote union, so that

$$A \cup B = \{x : x \in A \text{ or } x \in B\}$$

- The **intersection** of two sets is the collection of all elements the two sets have in common; that is,

$$A \cap B = \{x : x \in A \text{ and } x \in B\}$$

- The **difference** of two sets A and B, denoted by $A - B$, is the set of all elements of A that are not elements of B.

$$A - B = \{x : x \in A \text{ and } x \notin B\}$$

- Certain properties of the set operations follow easily from their definition. For example, if A, **B**, and C are sets, the following laws hold.

Idempotency	$A \cup A = A$
	$A \cap A = A$
Commutativity	$A \cup B = B \cup A$
	$A \cap B = B \cap A$
Associativity	$(A \cup B) \cup C = A \cup (B \cup C)$
	$(A \cap B) \cap C = A \cap (B \cap C)$
Distributivity	$(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$
	$(A \cap B) \cup C = (A \cup C) \cap (B \cup C)$
Absorption	$(A \cup B) \cap A = A$
	$(A \cap B) \cup A = A$
DeMorgan's laws	$A - (B \cup C) = (A - B) \cap (A - C)$
	$A - (B \cap C) = (A - B) \cup (A - C)$

- Two sets are **disjoint** if they have no element in common, that is, if their intersection is empty.

2 Relations and Functions

Note: *This part can be studied from notes of previous semester.*

We write the ordered pair of two objects a and b as $(a,\ b)$; a and b are called the **components** of the ordered pair $(a,\ b)$. The ordered pair $(a,\ b)$ is not the same as the set $\{a,\ b\}$. First, the order matters: $(a,\ b)$ is different from $(b,\ a)$, whereas $\{a,\ b\} = \{b,\ a\}$.

The **Cartesian product** of two sets A and B, denoted by $A \times B$, is the set of all ordered pairs $(a,\ b)$ with $a \in A$ and $b \in B$.

A *binary relation* on two sets A and B is a *subset of* $A \times B$.

In general, we use letters such as f, g, and h for functions and we write $f : A \mapsto B$ to indicate that f is a function from A to B.

- A is called the **domain** of f.
- The object $f(a)$ is called the **image** of a under f.
- The **range** of a function is the complete set of all possible resulting values of the dependent variable.

If $f : A_l \times A_2 \times \cdots \times A_n \mapsto B$ is a function, and $f(a_1,\ ...,\ a_n) = b$, where $a_i \in A_i$ for $i = 1,\ ...,\ n$ and $b \in B$, then

- we sometimes call $a_1,\ ...,\ a_n$ the **arguments** of f
- b the corresponding **value** of f. Thus f may be specified by giving its value for each n-tuple of arguments.

Certain kinds of functions are of special interest.

- A function $f : A \mapsto B$ is **one-to-one** if for any two distinct elements $a,\ a' \in A$, $f(a) \neq f(a')$.
- A function $f : A \mapsto B$ is **onto** B if each element of B is the image under f of some element of A.
- A function $f : A \mapsto B$ is a **bijection** between A and B if it is both *one-to-one* and *onto* B.
- The **inverse** of a binary relation $R \subseteq A \times B$, denoted $R^{-1} \subseteq B \times A$, is simply the relation $\{(b,\ a) : (a,\ b) \in R\}$.

3 Special Types of Binary Relations

Note: *This part can be studied from notes of previous semester.*

- A relation $R \subseteq A \times A$ is **reflexive** if $(a,\ a) \in R$ for each $a \in A$.
- A relation $R \subseteq A \times A$ is **symmetric** if $(b,\ a) \in R$ whenever $(a,\ b) \in R$.
- A relation $R \subseteq A \times A$ is **antisymmetric** if whenever $(a,\ b) \in R$ and a and b are distinct $(a \neq b)$, then $(b,\ a) \notin R$.
- A relation $R \subseteq A \times A$ is **transitive** if whenever $(a,\ b) \in R$ and $(b,\ c) \in R$, then $(a,\ c) \in R$.
- A relation that is *reflexive*, *symmetric*, and *transitive* is called an **equivalence relation**. The "clusters" of an equivalence relation are called its **equivalence classes**.
- A relation that is *reflexive*, *antisymmetric*, and *transitive* is called a **partial order**.
- If $R \subseteq A \times A$ is a partial order, an element $a \in A$ is called **minimal** if the following is true: $(b,\ a) \in R$ only if $a = b$.
- A partial order $R \subseteq A \times A$ is a **total order** if, for all $a,\ b \in A$, either $(a,\ b) \in R$ or $(b,\ a) \in R$.

4 Finite and Infinite Sets

We call two sets A and B **equinumerous** if there is a bijection $f : A \mapsto B$. Recall that if there is a bijection $f : A \mapsto B$, then there is a bijection $f^{-1} : B \mapsto A$; hence equinumerosity is a symmetric relation.

In general, we call a set **finite** if, *intuitively*, it is equinumerous with $\{1,\ 2,\ ...,\ n\}$ for some natural number n. If A and $\{1,\ ...,\ n\}$ are equinumerous, then we say that the *cardinality* of A (in symbols, $|A|$) is n. The cardinality of a finite set is thus the number of elements in it.

A set is **infinite** if it is *not finite*. For example, the set $\mathbf{N}$ of natural numbers is *infinite*; so are sets such as the set of integers, the set of reals, and the set of perfect squares. However, *not all infinite sets are equinumerous.* A set is said to be **countably infinite** if it is equinumerous with $\mathbf{N}$, and **countable** if it is *finite* or *countably infinite.* A set that is *not countable* is **uncountable**.

Note: *The following pages are taken from Ebru Hoca's (Ebru Aydın Göl) notes.*

Alphabets and Languages

Lecture notes based on "Elements of the theory of computation" by H.R. Lewis and C. H. Papadimitriou.

Alphabets and Languages

"*In computational practice, data are encoded in the computer's memory as strings of bits or other symbols appropriate for manipulation by a computer. The mathematical study of the theory of computation must therefore begin by understanding the mathematics of strings of symbols.*".

Alphabet: a finite set of symbols, e.g., $\{a, b, \ldots, z\}$, $\{0, 1\}$.

String: A string over an alphabet is a finite sequence of symbols from the alphabet, e.g., "apple", "0011", "a", "e" (empty string). The set of all strings (including the empty string) over Σ is denoted by $\Sigma^\star$. The length of a string is the length of the sequence, e.g. $|abc| = 3$. A string $w \in \Sigma^\star$ can be considered as a function $w : \{1, \ldots, |w|\} \to \sigma$, where $w(j)$ is the symbol at the $j-$th position.

Language: Any subset L of $\Sigma^\star$ for an alphabet Σ is called a language over Σ.

String operations

Concatenation: Two strings x, y over the same alphabet, e.g. $x, y \in \Sigma^\star$, can be combined. $w = x \circ y$, or simply $w = xy$. $|w| = |x| + |y|$, $w(i) = x(i)$ for $i \leq |x|$, and $w(i) = y(i - |x|)$ for $i > |x|$.

$w \circ e = e \circ w = w$. Concatenation is associative, $x(yz) = (xy)z$

Substring: A string v is a substring of w if w can be written as $w = xvy$. If $w = vx$ then v is a prefix of w, and if $w = xv$, then v is a suffix of w.

w^i**:** For each string $w \in \Sigma^\star$ and natural number $i \in \mathbb{N}$, w^i is defined as (definition by induction):

$$w^0 = e$$
$$w^{i+1} = w^i \circ w \text{ for each } i \geq 0$$

Reversal The reverse of a string w, denoted by w^R, is the string spelled backwards, $w^R(i) = w(|w| - i + 1)$. Inductive definition:

- If $|w| = 0$, then $w^R = w = e$
- If $|w| = n + 1$ for some $n \in \mathbb{N}$, then $w = ua$ for some $a \in \Sigma$, and $w^R = au^R$.

Theorem 1 *For any two strings x, w, $(wx)^R = x^R w^R$.*

Languages

Given an alphabet Σ, any subset of $\Sigma^\star$ is called a language. E.g. $\Sigma = \{0, 1\}$, $L_1 = \{0, 1\}$, $L_2 = \{e, 0, 1, 00, 01, 10, 11\}$, $L_3 = \{w \in \Sigma^\star \mid |w| \leq 2\}$, $L_4 = \{w \in \Sigma^\star \mid w \text{ has an equal number of 0's and 1's}\}$. Representation:

$$L = \{w \in \Sigma^\star \mid w \text{ has the property } P\}$$

Language operations

Languages are sets, so set operations (union, intersection, difference) can be used on languages.

Complement: $\overline{L} = \Sigma^\star \setminus L$

Concatenation: L_1, L_2 are languages over Σ. $L = L_1 \circ L_2$ (or simply $L = L_1 L_2$) is defined as

$$L = \{w_1 w_2 \mid w_1 \in L_1 \text{ and } w_2 \in L_2\}$$

Kleene star: The Kleene star of a language L, denoted by $L^\star$, is the set of strings obtained by concatenating 0 or more strings from L.

Note that $\Sigma^\star$ and $L^\star$ definitions are consistent.

$L^+ = LL^\star$

See problems for section 1.7.

Finite Representations of Languages

Lecture notes based on " Elements of the theory of computation" by H.R. Lewis and C. H. Papadimitriou.

The goal is to represent the languages with finite specifications. Consider an alphabet Σ, the number of finite representations of languages (it should be a string), and the number of languages over Σ.
No matter how powerful our methods for representing the languages, only countably many languages can be represented.

Definition 1 (Regular expressions) *The regular expressions over an alphabet Σ are all strings over the alphabet $\Sigma \cup \{(,), \emptyset, U, \star\}$ that can be obtained as follows:*

1. *$\emptyset$ and each member of Σ is a regular expression*
2. *If α and β are regular expressions than so is $(\alpha\beta)$*
3. *If α and β are regular expressions than so is $(\alpha U \beta)$*
4. *If α is a regular expression than so is $\alpha^\star$*
5. *Nothing is a regular expression unless it follows from 1-4*

Every regular expression defines a language. Symbols U and $\star$ are interpreted as union and Kleene star, respectively, and juxtaposition is string concatenation.

Definition 2 (Languages defined by regular expressions) *For a regular expression α, $\mathcal{L}(\alpha)$ is the language represented by α and it is defined as*

1. *$\mathcal{L}(\emptyset) = \emptyset$ and $\mathcal{L}(a) = \{a\}$ for each $a \in \Sigma$*
2. *If α and β are regular expressions than $\mathcal{L}(\alpha\beta) = \mathcal{L}(\alpha)\mathcal{L}(\beta)$*
3. *If α and β are regular expressions than $\mathcal{L}(\alpha U \beta) = \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$*
4. *If α is a regular expression than so is $\mathcal{L}(\alpha^\star) = \mathcal{L}(\alpha)^\star$*

Note that, even though $\alpha U \beta U \gamma$ and $\alpha\beta\gamma$ are not officially a regular expression (why?), we use it due to the associativity of concatenation and union operations.
The class of **regular languages** consists of all languages L such that $L = \mathcal{L}(\alpha)$ for some regular expression α. The class of regular languages over Σ is precisely the closure of the set of languages $\{\{\sigma\} \mid \sigma \in \Sigma\} \cup \{\emptyset\}$ with respect to union, concatenation, and Kleene star.

For some language L, an algorithm that answers the question is $w \in L$ is called a **language recognition device.** On the other hand, descriptions of how a string from a language can be produced are called **language generators**.

How can you prove that L is regular? How can you prove that L is not regular (later in the course).

See questions 1.8.1-1.8.5 (and the rest).

Related concepts: Closure

Definition 3 *Let D be a set, let $n \geq 0$, and let $R \subseteq D^{n+1}$ be a $(n+1)$-ary relation on D. Then a subset B of D is said to be* ***closed under*** *R if $b_{n+1} \in B$ whenever*

- *$b_1, \ldots, b_n \in B$ and*
- *$(b_1, \ldots, b_n, b_{n+1}) \in R$*

Any property of the form "the set B is closed under relations $R_1, \ldots, R_m$" is called a closure property of B.

Definition 4 *The closure of a relation R with respect to property P is the relation obtained by adding the minimum number of ordered pairs to R to obtain property P.*