

1 Mathematical Notions and Terminology

Here, we will review several topics from discrete mathematics.

1.1 Sets and Subsets

Throughout this class, we will be using the concept of sets, tuples, and more. We begin with sets.

1.1.1 Sets

Definition 1.1: Set

A **set** is a collection of objects represented as a unit.

Sets may contain any type of objects, including numbers, symbols, and even other sets. The objects in a set are called its **elements** or **members**. For example, consider the set:

$$S = \{7, 21, 57\}$$

The set S contains 3 elements. To indicate set membership or nonmembership, we use \in and \notin , respectively. So, $7 \in S$ means that 7 *is in* S while $8 \notin S$ means that 8 *is not in* S .

Additionally, we note that the order of describing a set doesn't matter, nor does repetition of its members. So, the set $A = \{1, 2, 3\}$ and $Z = \{2, 1, 3\}$ are exactly the same; that is $A = Z$. A generalization of a set that does allow for duplicates does exist, though (see multisets).

Several other common examples of sets are:

- **Singleton sets**, or sets that contain one element. For example, $\{5\}$ is a singleton set.
- **Empty set**, or sets that contain no elements. These are written \emptyset . It should be noted that $\{\emptyset\} \neq \emptyset$!
- **Unordered pair**, or sets that contain two elements. For example, $\{5, 2\}$ is an unordered pair because $\{5, 2\} = \{2, 5\}$.

1.1.2 Defining Sets

There are two main ways to define a set.

- Explicitly writing out the elements. For example, the set S of all positive integers between 1 and 5 can be written like so:

$$S = \{2, 4\}$$

- Through a rule. Generally, to describe a set containing elements that adhere to a rule, we use:

$$\{n \mid \text{Rule about } n\}$$

So, for example, the example used in the previous bullet point could have been written like so:

$$S = \{n \mid 1 < n < 5 \text{ and } n = 2m \text{ for some } m \in \mathbb{Z}\}$$

1.1.3 Subsets

Definition 1.2: Subset

A set A is called a **subset** of a set B , written $A \subseteq B$, if every member of A is also a member of B .

For example, the set $A = \{7\}$ is a *subset* of S since $7 \in A$ and $7 \in S$. However, the set $B = \{7, 8\}$ is *not a subset* of S since $7 \in B$ and $7 \in S$ but $8 \in B$ while $8 \notin S$.

Definition 1.3: Proper Subset

A set A is called a **proper subset** of a set B , written $A \subset B$ or $A \subsetneq B$, if A is a subset of B and is not equal to B .

So, for example, the set $A = \{7, 21\}$ is a proper subset of S , written $A \subsetneq S$, since every element in A is in S but both aren't equal. However, the set $A = \{7, 21, 57\}$ is not a proper subset of S .

1.1.4 Multisets

Definition 1.4: Multiset

A **multiset** is a generalization of a set that allows for multiple instances for each of its elements.

For example, define $X = \{1, 1, 2, 3\}$ and $Z = \{1, 2, 3\}$. Then, X and Z are precisely the same as *sets* but are different as *multisets*. However, note that order still doesn't matter; for example, $\{1, 1, 2, 3\}$ and $\{1, 2, 1, 3\}$ are still the same *multisets*.

1.1.5 Infinite Sets

Definition 1.5: Infinite Set

An **infinite set** is a set that contains infinitely many elements.

Several common examples of infinite sets are \mathbb{Z} (the set of all integers), \mathbb{N} (the set of all positive integers), \mathbb{R} (the set of all real numbers).

1.1.6 Union, Intersections, and Complements

Definition 1.6: Union

The **union** of two sets A and B , written $A \cup B$, is the set we get by combining all of the elements in A and B into a single set. Mathematically, we write this as:

$$A \cup B = \{n \mid n \in A \text{ or } n \in B\}$$

For example, given $A = \{1, 2\}$ and $B = \{2, 3\}$, $A \cup B = \{1, 2, 3\}$.

Definition 1.7: Intersection

The **intersection** of two sets A and B , written $A \cap B$, is the set we get by combining all *common* elements of A and B into a single set. Mathematically, we write this as:

$$A \cap B = \{n \mid n \in A \text{ and } n \in B\}$$

For example, given the same two sets A and B used in the union example, $A \cap B = \{2\}$.

Definition 1.8: Complement

The **complement** of a set A is the set of all elements that are not in A .

For example, consider the set E of all even integers. Then, the complement of E is the set of all odd integers.

1.1.7 Power Set**Definition 1.9: Power Set**

The **power set** of A , denoted $\mathcal{P}(A)$ is the set of all subsets of A .

For example, if A is the set $\{0, 1\}$, then $\mathcal{P}(A)$ would be:

$$\{\emptyset, \{0\}, \{1\}, \{0, 1\}\}$$

1.2 Sequences and Tuples

Here, we will review sequences and tuples (kind of like ordered sets).

1.2.1 Sequences**Definition 1.10: Sequence**

A **sequence** of objects is a list of these objects in some order.

We usually write a sequence by writing the objects within parentheses. For example, the sequence 7, 21, 57 would be written:

$$(7, 21, 57)$$

As implied, the order matters in a sequence. In other words, the sequence $(7, 21, 57)$ is not the same as $(57, 7, 21)$. Additionally, repetition matters in a sequence. So, $(7, 21, 57)$ is not the same sequence as $(7, 21, 21, 57)$.

1.2.2 Tuples**Definition 1.11: Tuple**

A finite sequence is often called a **tuple**. A sequence with k elements is a **k -tuple**.

So, $(7, 21, 57)$ is a 3-tuple. As a side note, a 2-tuple is also called an **ordered pair**.

1.2.3 Cartesian Product**Definition 1.12: Cartesian Product**

The **Cartesian product** of two sets A and B , written $A \times B$, is the set of all ordered pairs wherein the first element is a member of A and the second element is a member of B .

If $A = \{1, 2\}$ and $B = \{x, y\}$, then:

$$A \times B = \{(1, x), (1, y), (2, x), (2, y)\}$$

Definition 1.13: Extended Cartesian Product

The **Cartesian product** of k sets, A_1, A_2, \dots, A_k , written $A_1 \times A_2 \times \dots \times A_k$, is the set of all k -tuples (a_1, a_2, \dots, a_k) where $a_i \in A_i$ for $1 \leq i \leq k$.

Using the definition of A and B from the previous definition, it follows that:

$$A \times B \times A = \{(1, x, 1), (1, x, 2), (1, y, 1), (1, y, 2), (2, x, 1), \dots\}$$

If we have the Cartesian product of a set A with itself, we can use the shorthand:

$$\underbrace{A \times \dots \times A}_{k \text{ times}} = A^k$$

Another (familiar) example is $\mathbb{R} \times \mathbb{R}$, or the set of all real points. We can define this set like so:

$$\{(i, j) \mid i, j \in \mathbb{R}\}$$

Some example and non-example of elements are:

$$(1, 5) \in \mathbb{R} \times \mathbb{R}$$

$$\left(\frac{1}{2}, \sqrt{2}\right) \in \mathbb{R} \times \mathbb{R}$$

$$(-0.8213, -92.3) \in \mathbb{R} \times \mathbb{R}$$

$$(i, 3 + 2i) \notin \mathbb{R} \times \mathbb{R}$$

1.3 Functions

A function is an object that sets up an input-output relationship. In other words, a function takes an input and produces an output. In every function, the same input always produces the same output. So, if f is a function whose output value is b when the input value is a , we write:

$$f(a) = b$$

Alternatively, a function is called a **mapping**. In other words, for the above function definition, we say that f maps a to b .

Definition 1.14: Function Definitions

For a function $f : A \mapsto B$:

- The **domain** A is the set of possible inputs to the function.
- The **codomain** B is the set of possible output values^a.
- The **range** is the set of actual output values. This is a subset of B .

^aThe book states that the range is the set of output values, but later states that the function may not use all the elements in the range.

Consider the function $\text{abs} : \mathbb{R} \mapsto \mathbb{R}$ defined by:

$$\text{abs}(x) = |x|$$

Here, the domain is the set of all real numbers. The codomain is the set of all real numbers as well. However, the range is the set of all *non-negative* real numbers, that is, $\mathbb{R}_{\geq 0}$.

1.3.1 Cartesian Product Input

The domain of a function f can be the Cartesian product of multiple sets $A_1 \times \cdots \times A_k$ for some sets A_1, \dots, A_k . In this case, f 's input is a k -tuple (a_1, \dots, a_k) where a_i are the **arguments** to f . In this case, the function with k -arguments is called a **k -ary function** and k is called the **arity** of the function. Note that:

- If $k = 1$, then f has a single argument and f is called a **unary function**.
- If $k = 2$, f is a **binary function**.

Certain binary functions can be written in a special **infix notation**; for example, the function:

$$\text{add} : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$$

Is often written as $a + b$ (with the $+$ symbol between the two arguments) instead of the **prefix notation** (with symbol preceding) $\text{add}(a, b)$.

1.3.2 Predicates

Definition 1.15: Predicate

A **predicate** is a function whose range is $\{\text{true}, \text{false}\}$.

For example, the function:

$$\text{even} : \mathbb{Z} \mapsto \{\text{true}, \text{false}\}$$

Takes in an integer and outputs **true** if the integer is even and **false** otherwise. So:

$$\text{even}(10) = \text{true}$$

$$\text{even}(11) = \text{false}$$

A predicate whose domain is a set of k -tuples $A \times \cdots \times A$ is called a **relation**, a **k -ary relation**, or a **k -ary relation on A** . A common case is a 2-ary relation, called a **binary relation**, for which we often use infix notation. For example, several very common binary relations are the “less than” and “equals” relation.

If R is a binary relation, then the statement aRb means that aRb is **true**. For example, if R is the binary relation $<$, then $3 < 5$ is **true**. Similarly, if R is a K -ary relation, the statement $R(a_1, \dots, a_k)$ means that $R(a_1, \dots, a_k)$ is **true**.

1.3.3 Equivalence Relations

Suppose R is a relation over a set S . Then:

- R is called **reflexive** if $\forall x \in S, xRx$. That is, every $x \in S$ is related to itself.
- R is called **symmetric** if $\forall x, y \in S, xRy \implies yRx$. In other words, if x is related to y , is y related to x ?
- R is called **transitive** if $\forall x, y, z \in S, xRy$ and yRz implies that xRz .

For example, define the equivalence relation on \mathbb{N} , written \equiv_7 . For any $i, j, k \in \mathbb{N}$, say $i \equiv_7 j$ if $i - j$ is a multiple of 7. This is an equivalence relation because:

- It is reflexive as $i - i = 0$ and 0 is a multiple of 7.
- It is symmetric as $i - j$ is a multiple of 7 if $j - i$ is a multiple of 7.
- It is transitive as whenever $i - j$ and $j - k$ are both multiples of 7, then $i - k = (i - j) + (j - k)$ is the sum of two multiples of 7, and hence a multiple of 7.

1.4 Undirected Graphs

Definition 1.16: Undirected Graph

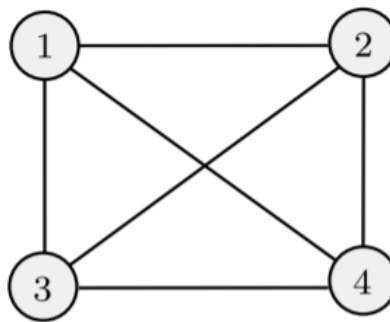
An **undirected graph**, or simply a **graph** is a pair $G = (V, E)$, where:

- V is a set whose elements are called *nodes* or *vertices*.
- E is a set of paired nodes, whose elements are called *edges*.

Informally, an undirected graph is a set of points with lines connecting some of these points. The points are called nodes or vertices and the lines are called edges.

In an undirected graph G that contains nodes i and j , the pair (i, j) or (j, i) represents the edge that connects i and j ; in other words, (i, j) and (j, i) are the same. So, it follows that the order in which we put i and j in the tuple doesn't matter. Because order doesn't matter, we may also represent (i, j) as $\{i, j\}$.

For example, the following graph is an undirected graph:



Definition 1.17: Degree of a Node

The number of edges at a particular node is the **degree** of that node.

We note that no more than one edge is allowed between any two nodes; however, we may allow an edge from a node to itself (called a *self-loop*).

Looking at the graph above, we can define $G = (V, E)$ where¹:

- $V = \{1, 2, 3, 4\}$
- $E = \{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\}$

Additionally, each node has degree 3 because each node has 3 edges.

1.4.1 Labeled Graph

A graph is a **labeled graph** if the nodes and/or edges of a graph are labeled. For example, we might represent each node as a city and each edge as the number of miles from node A to node B.

¹Remember, because we're working with an undirected graph, an edge like $(1, 2)$ and $(2, 1)$ refer to the same edge!

1.4.2 Subgraph

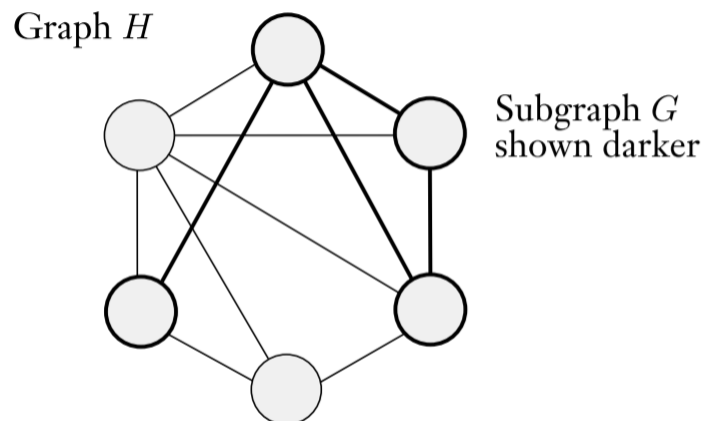
Definition 1.18: Subgraph

A graph G is a **subgraph** of graph H if the nodes of G are a subset of the nodes of H *and* the edges of G are the edges of H on the corresponding nodes.

In other words, if we had $G = (V', E')$ and $H = (V, E)$, we must have:

- $V' \subseteq V$
- $E' \subseteq E$

The following figure shows a graph H and a subgraph G :

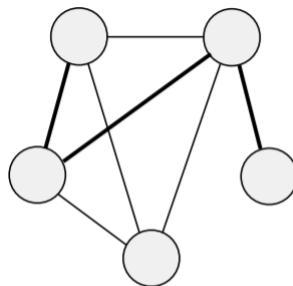


1.4.3 Path

Definition 1.19: Path

A **path** in a graph is a sequence of nodes connected by edges.

For example, a path might look like:

**Definition 1.20: Simple Path**

A **simple path** is a path that doesn't repeat any nodes.

1.4.4 Connections Between Nodes

Definition 1.21: Connected Graph

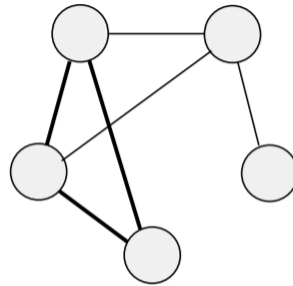
A graph is **connected** if every two nodes have a path between them.

So, the graph in the above example is a connected graph.

Definition 1.22: Cycles

A path is a **cycle** if it starts and ends in the same node.

For example, a cycle might look like:



Definition 1.23: Simple Cycle

A **simple cycle** is a cycle that contains at least three nodes and repeats only the first and last node.

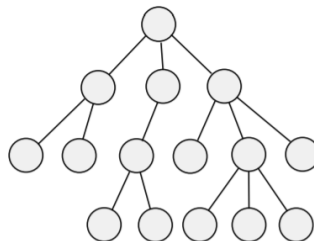
1.4.5 Trees

Definition 1.24: Tree

A graph is a **tree** if it is connected and has no simple cycles.

A tree can contain a specially designated node called the *root*, which is usually a singular node at the “top” that branches out to every other node. In this case, the nodes of degree 1 in a tree, other than the root, are called the *leaves* of the tree.

A tree might look like:



1.5 Directed Graphs

Definition 1.25: Directed Graph

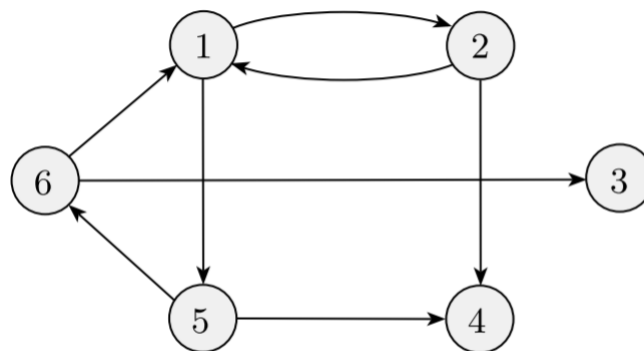
A **directed graph** is a pair $G = (V, E)$ where:

- V is a set whose elements are called *nodes* or *vertices*.
- E is a set of paired nodes, whose elements are called *directed edges*.

Informally, a directed graph is a set of points with arrows instead of lines connecting some of these points.

In a directed graph that contains nodes i and j , the pair (i, j) and (j, i) **do not represent** the same edges. In particular, (i, j) means that there is an arrow pointing *from* i to j ; then, (j, i) means that there is an arrow pointing from j to i . In other words, the order **does** matter.

For example, the following graph is a directed graph:



We can define $G = (V, E)$ as:

- $V = \{1, 2, 3, 4, 5, 6\}$
- $E = \{(1, 2), (2, 1), (6, 1), (6, 3), (1, 5), (2, 4), (5, 4), (5, 6)\}$

1.5.1 Directed Path

Definition 1.26: Directed Path

A path in which all the arrows point in the same direction as its steps is called a **directed path**.

1.5.2 Connections

Definition 1.27

A directed graph is **strongly connected** if a directed path connects every two nodes.

1.5.3 Relationship to Binary Relations

We can use directed graphs to depict binary relations. In particular, if R is a binary relation whose domain is $D \times D$, a labeled graph $G = (D, E)$ represents R , where $E = \{(x, y) \mid xRy\}$. For example, for $D = \{1, 2, 3\}$ and binary relation $<$, the edges would be:

$$E = \{(1, 2), (2, 3), (1, 3)\}$$

2 Boolean Logic

Boolean logic is a mathematical system built around **true** and **false**, which are often represented as 1 and 0, respectively. We can manipulate boolean values with boolean operations. The operations are as follows:

Negation: The negation, or **not**, operation is designated with the symbol \neg . The negation of a Boolean value is simply the opposite value. Therefore:

| a | $\neg a$ |
|-----|----------|
| 0 | 1 |
| 1 | 0 |

Conjunction: The conjunction, or **and**, operation is designated with the symbol \wedge . The conjunction of two Boolean values is 1 if both values are 1 and 0 otherwise.

| a | b | $a \wedge b$ |
|-----|-----|--------------|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Disjunction: The disjunction, or **or**, operation is designated with the symbol \vee . The disjunction of two Boolean values is 1 if either of those values are 1 and 0 otherwise.

| a | b | $a \vee b$ |
|-----|-----|------------|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

There are several other operators that may occasionally appear; they are:

Exclusive OR: The *exclusive or*, or **XOR**, is designated by \oplus ; it is 1 if either, but not both, of the operands is 1.

| a | b | $a \oplus b$ |
|-----|-----|--------------|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Equality: The *equality* operator is designated by \longleftrightarrow ; it is 1 if both operands have the same value.

| a | b | $a \longleftrightarrow b$ |
|-----|-----|---------------------------|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Implication: The *Implication* operator is designated by \longrightarrow ; it is 0 if its first operand is 1 and its second operand is 0 and 1 otherwise.

| a | b | $a \longrightarrow b$ |
|-----|-----|-----------------------|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Finally, the **distributive law** for AND and OR is useful when manipulating Boolean expressions. That is:

$$P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$$

$$P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$$

3 Finite Automata

TODO

4 Strings and Languages

Definition 4.1: Alphabet

An **alphabet** is any nonempty finite set. Generally, we use Σ and Γ to designate alphabets.

Definition 4.2: Symbols

The members of the alphabet are the **symbols** of the alphabet.

Some example of alphabets are:

$$\Sigma_1 = \{0, 1\}$$

$$\Sigma_2 = \{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$$

$$\Gamma = \{0, 1, x, y, z\}$$

Definition 4.3: String

A **string** over an alphabet is a finite sequence of symbols from that alphabet, usually written next to one another and not separated by commas.

For example, if we use Σ_1 as our alphabet, then 01001 is a **string** over Σ_1 . Likewise, if we use Σ_2 as our alphabet, then **something** is a string over Σ_2 .

Definition 4.4: Length

If w is a string over Σ , then the **length** of w , written $|w|$, is the number of symbols that it contains.

Remarks:

- The string of length zero is called the **empty string** and is written ϵ . The empty string plays the role of 0 (like an identity) in a number system.
- If w has length n , then we can write $w = w_1w_2 \dots w_n$ where each $w_i \in \Sigma$.

Definition 4.5: Reverse

The **reverse** of a string w , written $w^{\mathcal{R}}$, is the string obtained by writing w in the opposite order.

Remark: In other words, for a string w , we can write the reverse of w as $w^{\mathcal{R}} = w_nw_{n-1} \dots w_2w_1$.

Definition 4.6: Substring

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

For example, if we look at the string $w = \text{something}$, then $z_1 = \text{some}$ and $z_2 = \text{thing}$ are both substrings of w .

Definition 4.7: Concatenation

For a string x of length m and a string y of length n , the **concatenation** of x and y , written xy , is the string obtained by appending y to the end of x , as in:

$$x_1 \dots x_m y_1 \dots y_n$$

If we want to concatenate a string x with itself many times, we use the superscript notation x^k to mean:

$$\underbrace{xx \dots x}_k$$

If we consider the two substrings z_1 and z_2 in the previous example, then $z_1 z_2 = \text{something}$.

Definition 4.8: Prefix

A string x is a **prefix** of a string y if a string z exists where $xz = y$.

For example, if we look at the two substrings z_1 and z_2 in the previous example yet again, we say that z_1 is a prefix of w since $z_1 z_2 = w$.

Definition 4.9: Proper Prefix

A string x is a proper prefix of a string y if, in addition to x being a prefix of y , $x \neq y$.

So, **some** is a proper prefix of **something** while **something** is not a proper prefix of **something**.

Definition 4.10: Language

A **language** is a set of strings.

Definition 4.11: Prefix-Free

A language is **prefix-free** if no member is a proper prefix of another member.