COM1026 EXAM NOTES, CONDENSED

01 IFH 02 - Ada
Lovelace Lab 129
Time: 15:30 - 17:30

Jim Lam

October 19, 2024

Contents

1 Set Theory

1.1 Introduction

Quick recap on Naive set theory, meaning:

1. Introducing the basic concept of sets;
2. Introduce notation;
3. Illustrate Union, Intersection, and Set Difference operations;
4. Venn diagrams as proof;
5. Power sets;

6. How to proof with more rigor.

1.2 Set Definition

Question: Why are sets relevant to computing?

We have to represent data to compute it. To group data, we put it into sets. Some sets that I have seen before:

$$\text{The set of natural numbers: } \mathbb{N} = \{1, 2, 3, ...\}$$
$$\text{The set of integers: } \mathbb{Z} = \{..., -3, -2, -1, 0, 1, 2, 3, ...\}$$

Question: What are sets?

A set is a collection of objects, called elements of the set. The elements of a set can be anything, but they must be distinct.

1.3 Notation

Sets are denoted with capital letters, e.g. A, B, C. The elements of a set are listed inside curly brackets:

$$A = \{1, 2, 3\}$$
$$B = \{a, b, c, d, e, f, g, h\}$$

1.3.1 Cardinality

The cardinality of a set is the number of elements in the set. The cardinality of a set is denoted with vertical bars, with the hash, or alternatively, the function *card*:

$$\text{Card}(A) = |A| = 3$$
$$\#\{1, 2, 3, 4, 5\} = 5$$
$$\#A = 3$$
$$\#\mathbb{N} = \infty$$

1.3.2 Abstraction axiom

GIven a property $P(x)$, we can define a set A as:

$$A = \{x | P(x)\}$$

In other words, whatever property P, there exists a set A containing the objects that satisfy P and only these objects.

1.3.3 Set builder notation

Other than enumerating the elements of a set, there are other ways to describe a set. Verbal descriptions and adding an inclusion rule to the set builder notation are two more examples:

$$\begin{aligned} C &= \{x | x \in \mathbb{N}, 0 \leq x \leq 5\} \\ &= \{x | x \text{ is in the appropriate set}, 0 \leq x \leq 5\} \end{aligned}$$

1.3.4 Empty set

The empty set is a set with no elements. It is denoted $\emptyset$ or $\{\}$.

$$\emptyset = \{\}$$
$$\emptyset \neq \{\emptyset\}$$

1.4 Basic set operations

1.4.1 Membership

The membership relation is denoted $\in$ and is used to indicate that an element is in a set. Conversely, $\notin$ is used to indicate that an element is not in a set.

$$\text{Let the set } A = \{1, 2, 3\}$$
$$1 \in A = \text{True}$$
$$3 \notin A = \text{False}$$

To denote a subset, we use $\subseteq$. Since a subset can include the set itself, we use $\subset$ to denote a proper subset.

$$\text{Let the set } A = \{1, 2, 3\}$$
$$\{1, 2\} \subseteq A = \text{True}$$
$$\{1, 2, 3\} \subset A = \text{False}$$

1.4.2 Union, intersection, and set difference

The union of two sets A and B is denoted $A \cup B$ and contains all elements of both sets.

The intersection $A \cap B$ contains elements common to both.

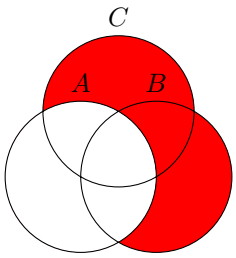
Set diffrence $A \setminus B$ contains elements in A but not in B. (Note: A must come first, otherwise the result is different.)

$$A \cup B = \{1, 2, 3, a, b, c\}$$
$$A \cap B = \{1, 2, 3\}$$
$$A \setminus B = \{1, 2, 3\}$$

1.5 Venn diagrams

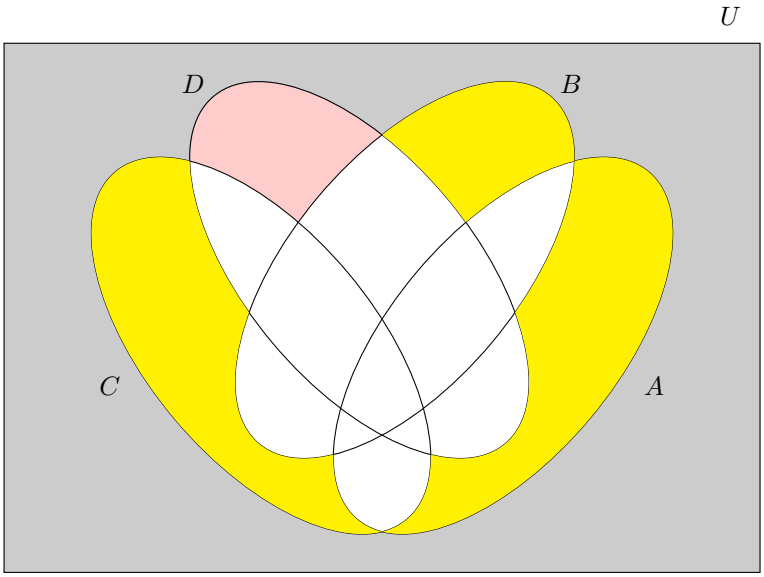
Venn diagrams are a way to visualise sets and their operations. Use circles to represent sets, and shading to distinguish areas of interest.

$$\begin{aligned}\text{Let the set } &= \{1, 2, 3\} \\ \text{Let the set } &= \{2, 3, 4\} \\ \text{Let the set } &= \{3, 4, 5\} \\ (C \cup B) \cap A &= \{4, 5\}\end{aligned}$$



Here is a more complicated example involving 4 sets:

Refrenced from `https://www.overleaf.com/latex/examples/example-venn-diagram-with-isolated-xjptmqsjfdlc`



This covers the basics of set notation and operations.

1.6 Power sets

The power set of a set is the set of all subsets of that set.

The power set of a set A is denoted in various ways, including 2^A, $\mathcal{P}(A)$, $\mathbb{P}(A)$, or $\wp(A)$.

$$\begin{aligned} \text{Let the set } A &= \{1, 2, 3\} \\ \wp(A) &= \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\} \\ \text{More generally: } \mathbb{P}(A) &= \{B | B \subseteq A\} \end{aligned}$$

1.7 Proofs

1.7.1 Proof by property

Proposition (example from lecutre notes): For any sets A, B, and C:

$$A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$$

Recapping some of the basic properties of sets, using sets S and T, and element x:

$$\begin{aligned} &\text{Property 1:} && S \subseteq T \text{ and } T \subseteq S &&\iff S = T \\ &\text{Property 2:} && (\text{For any x} \in S \implies x \in T) &&\iff S \subseteq T \\ &\text{Property 3:} && x \in S \text{ and } x \in T &&\iff x \in S \cap T \\ &\text{Property 4:} && x \in S \text{ or } x \in T &&\iff x \in S \cup T \\ &\text{Property 5:} && x \in S \text{ and } x \notin T &&\iff x \in S \setminus T \\ &\text{Property 6:} && x \notin S \text{ and } x \notin T &&\iff x \notin T \cup S \end{aligned}$$

Using property 1, we can prove that two sets are equal by proving that each is a subset of the other.

$$\begin{aligned} &\text{Let } x \in A \setminus (B \cup C) \\ &\iff x \in A \text{ and } x \notin B \cup C \\ &\iff x \in A \text{ and } x \notin B \text{ and } x \notin C \\ &\iff x \in A \setminus B \text{ and } x \in A \setminus C \\ &\iff x \in (A \setminus B) \cap (A \setminus C) \\ &\therefore A \setminus (B \cup C) \subseteq (A \setminus B) \cap (A \setminus C) \end{aligned}$$

Doing this for the other direction:

$$
\begin{aligned}
&A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C) \\
&\text{Let } x \in (A \setminus B) \cap (A \setminus C) \\
&\quad \iff x \in A \setminus B \text{ and } x \in A \setminus C \\
&\quad \iff x \in A \text{ and } x \notin B \text{ and } x \in A \text{ and } x \notin C \\
&\quad \iff x \in A \text{ and } x \notin B \cup C \\
&\quad \iff x \in A \setminus (B \cup C) \\
&\therefore (A \setminus B) \cap (A \setminus C) \subseteq A \setminus (B \cup C)
\end{aligned}
$$

1.8 Von Neumann Ordinals

The von Neumann ordinals are a way of representing the natural numbers using sets.

let $0 = \emptyset$, $n + 1 = n \cup \{n\}$:

$$
\begin{aligned}
0 &= \emptyset \\
1 &= \{0\} = \{\emptyset\} \\
2 &= \{0, 1\} = \{\emptyset, \{\emptyset\}\} \\
3 &= \{0, 1, 2\} = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\} \\
n &= \{0, 1, 2, ..., n - 1\} \text{ you get the idea}
\end{aligned}
$$

2 Relations

2.1 Cartesian product and relations

The cartesian product of two sets A and B is the set of all ordered pairs (x, y), where: $x \in A$ and $y \in B$. The cartesian product of A and B is denoted as $A \times B$.

Example:

$$
\begin{aligned}
\mathit{Books} &= \{1984, \text{Concrete}, \text{Incerto}\} \\
\mathit{Rating} &= \{\text{Good}, \text{Evil}, \text{Unimportant}\} \\
\mathit{Books} \times \mathit{Rating} &= \{(1984, \text{Good}), (1984, \text{Evil}), (1984, \text{Unimportant})\} \\
&\quad \cup \{(\text{Concrete}, \text{Good}), (\text{Concrete}, \text{Evil}), (\text{Concrete}, \text{Unimportant})\} \\
&\quad \cup \{(\text{Incerto}, \text{Good}), (\text{Incerto}, \text{Evil}), (\text{Incerto}, \text{Unimportant})\}
\end{aligned}
$$

A relation R from A to B is a subset of the cartesian product of A and B. i.e. $R \subseteq A \times B$. Taking the example above, we can define a relation R from Books to Rating as:

$$R = \{(1984, \text{Good}), (\text{Concrete}, \text{Evil}), (\text{Incerto}, \text{Unimportant})\}$$

Note that R is a subset of the cartesian product of Books and Rating.

2.2 Relation notation

There are three main ways to denote a relation R from A to B:

- Ordered pairs
- Table
- Mapping

Should you want to confuse yourself even further, the infix notation is a good option.

Let the relation "likes" be defined as: $\mathbf{L} = \{(x, y) | x \text{ likes } y\}$
You can denote (x, y) as a member of the likes by writing: x**L**y

2.3 Representing relations

Since winter is approaching, let's define a relation "likes" from people to clothing.

Let the set of people $P = \{\text{Jim}, \text{Bob}, \text{Alice}, \text{Eve}, \text{Mallory}\}$
Let the set of clothing $C = \{\text{Jacket}, \text{Scarf}, \text{Gloves}, \text{Hat}, \text{Socks}\}$
Let the relation $\mathbf{L} = \{(x, y) | x \text{ likes } y\}$
In table form:

P	L
Jim	Scarf
Bob	Jacket
Alice	Scarf
Eve	Gloves
Mallory	Hat
Mallory	Socks

Since a table popped into your view, it is as good a time as any to introduce it's application in databases.

Given this example relation of students and their various attributes, we can represent it in a table.

Let the set of students $S = \{\text{Jim}, \text{Bob}, \text{Alice}, \text{Eve}, \text{Mallory}\}$

Let the set of attributes $A = \{\text{name}, \text{Age}, \text{Height}, \text{Weight}, \textit{textHappiness}, \text{Net Worth}\}$

Let the relation $\mathbf{R} = \{(x, y) | x \text{ has attribute } y\}$

In table form:

Name	Age	Height	Weight	Happiness	Net Worth
Jim	20	180	80	0.5	0.1
Bob	21	170	70	0.6	0.2
Alice	19	160	60	0.7	0.3
Eve	18	150	50	0.8	0.4
Mallory	17	140	40	0.9	0.5

We can extract information by getting a subset of relations.
For example: $(\text{Jim}, 180) \in \text{Height}$

2.4 Domain and range

The domain of a relation is the set of all first elements of the ordered pairs in the relation.

$$\mathit{Dom}(\mathbf{L}) = \{x \in A \mid \exists x \in A : (x, y) \in \mathbf{L}\}$$

The range of a relation is the set of all second elements of the ordered pairs in the relation.

$$\mathit{Ran}(\mathbf{L}) = \{y \in B \mid \exists y \in B : (x, y) \in \mathbf{L}\}$$

2.5 Relational composition

The relational composition of two relations $\mathbf{R}$ and $\mathbf{S}$ is the relation $\mathbf{R} \circ \mathbf{S}$ (or $\mathbf{R}; \mathbf{S}$, used to avoid confusion with function composition) defined as:

$$\mathbf{R}; \mathbf{S} = \{(x, z) \mid \exists y \in B : (x, y) \in \mathbf{R} \text{ and } (y, z) \in \mathbf{S}\}$$

EXAMPLE:

$$\mathbf{R} = \{(1, 2), (2, 3), (3, 4)\}$$
$$\mathbf{S} = \{(2, 3), (3, 4), (4, 5)\}$$
$$\mathbf{R}; \mathbf{S} = \{(1, 3), (2, 4), (3, 5)\}$$

2.6 Closures and equivalence classes

2.6.1 Property of relations

A relation $\mathbf{R}$ is reflexive if $\forall x \in A : (x, x) \in \mathbf{R}$.

A relation $\mathbf{R}$ is symmetric if $\forall x, y \in A : (x, y) \in \mathbf{R} \implies (y, x) \in \mathbf{R}$.

A relation $\mathbf{R}$ is transitive if $\forall x, y, z \in A : (x, y) \in \mathbf{R}$ and $(y, z) \in \mathbf{R} \implies (x, z) \in \mathbf{R}$.

2.6.2 Closures

The closure of a relation $\mathbf{R}$ is the smallest relation containing $\mathbf{R}$ that is transitive.

Example of constructing a reflexive closure:

$$\text{Let the relation } \mathbf{R} = \{(1, 2), (2, 3), (3, 4)\}$$
$$\text{The reflexive closure of } \mathbf{R} = \{(1, 1), (1, 2), (2, 3), (3, 4), (2, 2), (3, 3), (4, 4)\}$$

Example of constructing a symmetric closure:

$$\text{Let the relation } \mathbf{R} = \{(1, 2), (2, 3), (3, 4)\}$$
$$\text{The symmetric closure of } \mathbf{R} = \{(1, 2), (2, 3), (3, 4), (2, 1), (3, 2), (4, 3)\}$$

Example of constructing a transitive closure:

$$\text{Let the relation } \mathbf{R} = \{(1, 2), (2, 3), (3, 4)\}$$
$$\text{The transitive closure of } \mathbf{R} = \{(1, 2), (2, 3), (3, 4), (1, 3), (2, 4), (1, 4)\}$$

2.6.3 Equivalence classes

let $\rho \subseteq A \times A$ be an equivalence relation on A, given $A \neq \emptyset$, $a \in A$ be an arbitrary element of A.

NOTE: MUST CONSTRUCT EQUIVALENCE RELATION BEFORE CONSTRUCTING EQUIVALENCE CLASSES.

$$[a]_\rho = \{x \in A \mid (a, x) \in \rho\}$$

2.6.4 More properties of relations

- A relation $\mathbf{R}$ is antisymmetric if $\forall x, y \in A : (x, y) \in \mathbf{R}$ and $(y, x) \in \mathbf{R} \implies x = y$.
- A relation $\mathbf{R}$ is a partial order if it is reflexive, antisymmetric, and transitive.
- A connex relation is a relation $\mathbf{R}$ such that $\forall x, y \in A : (x, y) \in \mathbf{R}$ or $(y, x) \in \mathbf{R}$.
- Total order means that a relation is a partial order and a connex relation.

3 Functions

3.1 Definition

What is a function in the context of discrete mathematics?

A function is a relation **f** from A to B such that every element in A is mapped to exactly one element in B, i.e.:

$\forall x \in A. \forall y, z \in B : ((x, y) \in f \wedge (x, z) \in f \implies y = z)$.

3.2 Notation

A function **f** from A to B is denoted as $\mathbf{f} : A \to B$.

3.3 Injective, surjective, bijective

- A function $\mathbf{f} : A \to B$ is injective if $\forall x, y \in A : \mathbf{f}(x) = \mathbf{f}(y) \implies x = y$.
- A function $\mathbf{f} : A \to B$ is surjective if $\forall y \in B : \exists x \in A : \mathbf{f}(x) = y$.
- A function $\mathbf{f} : A \to B$ is bijective if it is both injective and surjective.

3.4 Composition

The composition of two functions $\mathbf{f} : A \to B$ and $\mathbf{g} : B \to C$ is the function $\mathbf{g} \circ \mathbf{f} : A \to C$ defined as:

$$\mathbf{g} \circ \mathbf{f} = \{(x, z) \mid \exists y \in B : (x, y) \in \mathbf{f} \text{ and } (y, z) \in \mathbf{g}\}$$

3.5 Inverse

The inverse of a function $\mathbf{f} : A \to B$ is the function $\mathbf{f}^{-1} : B \to A$ defined as:

$$\mathbf{f}^{-1} = \{(y, x) \mid (x, y) \in \mathbf{f}\}$$

4 Language and regular expressions

4.1 Alphabet

We specify an Alphabet using the symbol Σ.

Examples:

$\Sigma_1 = \{a, b, c, d, e, f\}$

$\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$\Sigma_3 = \{S \mid S \subseteq \Sigma_1\}$ (The power set of $\Sigma_1, \#\Sigma_3 = 2^6 = 64$)

$\Sigma_4 = \{(x, y) \mid x \in \Sigma_1 \wedge y \in \Sigma_2\}$ Ordered pairs of characters work as well

An alphabet must be a set that contain finite elements, hence sets like $\mathbb{N} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, ...\}$ cannot be the alphabet of a language.

4.2 Strings

A string is a finite sequence of characters from an alphabet.

A string of length n is denoted as the n-tuple $w = a_1a_2a_3...a_n$, written without punctuation.

The set of all finite strings are denoted as Σ^*, and we can say that string s is in Σ^* if $s \in \Sigma^*$.

4.2.1 Empty string

Might be jarring to you, Jim, but you have the option to denote an empty string as ϵ. Will be useful later on.

4.2.2 String Operations

- Concatenation: w_1w_2 is the concatenation of strings w_1 and w_2.
- Length: $|w|$ is the length of string w. The length of concatenated strings are simply the sum of the lengths of the individual strings.

4.3 Language

A language is a set of strings.

4.4 Regular expressions

A regular expression is a string that denotes a language. Here are the rules:

- let Σ be an alphabet set.
- a denotes the language $\{a\}$, where $a \in \Sigma$, which is on its own a regular expression.
- ϵ and $\emptyset$ denote the languages $\{\epsilon\}$ and $\emptyset$, which are also regular expressions.
- Given r and s as regular expressions, the following are also regular expressions:
  - rs, $r|s$, r^*

4.4.1 Examples

The following are rules for matching strings to RegEx, let s be a string and r be a regular expression:

- s matches a when $s = a$
- ϵ matches ϵ when $s = \epsilon$
- $\emptyset$ matches nothing.
- $r|s$ matches r or s.
- rs matches r followed by s.
- r^* matches if $s = \epsilon$ or $s = s_1 s_2 ... s_n$, where s_i matches r for all i.

4.4.2 Regular languages

A language is regular if it is denoted by a regular expression.

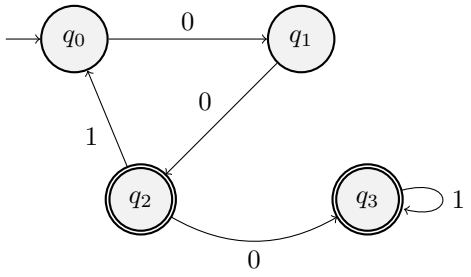
For alphabet Σ and regular expressions r:

$$L(r) = \{s \in \Sigma^* \mid s \textbf{ matches } r\}$$

5 Finite automata

5.1 Deterministic finite automata

$$\begin{aligned}
\text{States: } & Q = \{q_0, q_1, q_2, q_3\} \\
\text{Symbol: } & \Sigma = \{0, 1\} \\
\text{Transition Function} & \delta : Q \times \Sigma \to Q \\
\text{Start: } & = q_0 \in Q \\
\text{Accepting: } & = \{q_2, q_3\} \subseteq Q \\
\text{DFA}(^\epsilon)\text{: } & M = (Q, \Sigma, \delta, q_0, \{q_2, q_3\})
\end{aligned}$$



5.1.1 Criteria for a DFA

- DFAs have exactly one start state.
- May have one or more accepting states.
- For each state, there must be at most one outgoing transition **for each symbol in the alphabet.**

5.1.2 Language definition with DFAs

For automaton M, the language $L(M)$ consists of all strings s over its alphabet of input symbols statisfying:

$$q_0 \xrightarrow{s} *q \tag{1}$$

$$s = q_0, q_1, q_2, ..., q_n \text{ for the states: } q_0, q_1, q_2, ..., q_n \tag{2}$$

If (1) is the case, s is accepted by M. More formally:

$$L(M) = \{u \mid u \textbf{ is accepted by } M\}$$

5.2 Non-deterministic finite automata

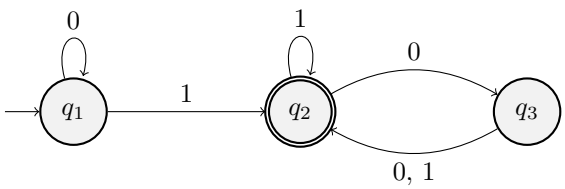
5.2.1 What is the difference?

- NDFAs can have multiple outgoing transitions for a given symbol.
- NDFAs can have ϵ-transitions, which are transitions that can be taken without consuming an input symbol.
- NDFAs can have multiple start states.
- NDFAs can have no accepting states.
- NDFAs can have multiple accepting states.

5.3 Examples

Referenced from `https://www3.nd.edu/~kogge/courses/cse30151-fa17/Public/other/tikz_tutorial.pdf`.

5.3.1 DFA 1



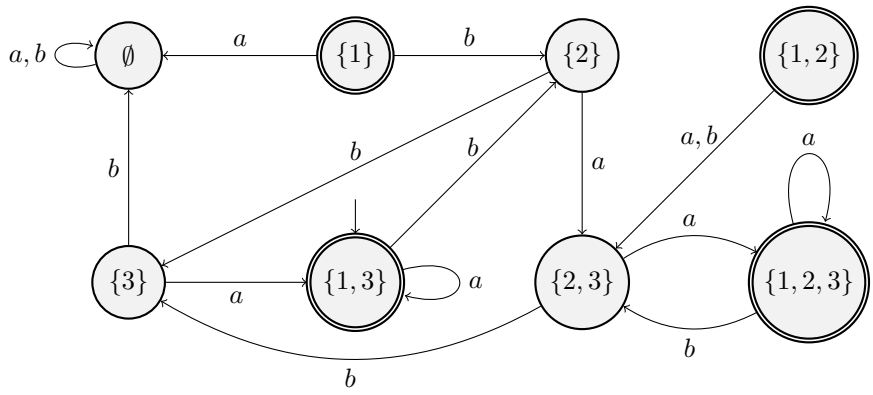
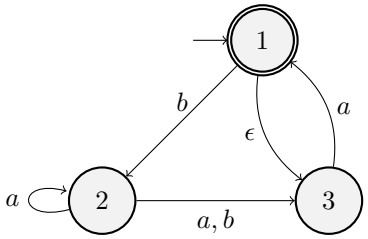


Figure 1: DFA

5.3.2 NFA 1



5.3.3 DFA 2

6 logic

What does logical thinking in practice look like?

In out daily life we use logic to make decisions, and to make sense of the world around us. As computers scientists, we have boolean logic machines at our disposal, and we can use them to solve problems. To take advantage of that, it helps to have a formal and systematic way of thinking about logic.

6.1 Propositional logic

The simplest form of formal reasoning is captured by propositional logic.

6.1.1 Propositional atoms

Propositional atoms are statements that can be directly evaluated to either true or false.

- I have a pen. $\rightarrow$ True

- The sky is blue. → True
- Love is a lie. → False
- Fire kills. → True
- The earth is flat. → False
- Peter is openly gay. → True
- The moon is made of cheese. → False
- My notes will not help my exam. → False

6.1.2 Propositional connective operator symbols

Here are the table of symbols:

- $\neg$ (negation: not)
- $\wedge$ (conjunction: and)
- $\vee$ (disjunction: or)
- $\implies$ (implication: implies)
- $\iff$ (double implication: iff or if and only if)

6.1.3 Truth tables

Truth tables are a tabular way of representing the truth values of propositional atoms for all possible combinations of truth values.

Here is an example compound proposition:

$$q \vee r \implies s$$

Here is the truth table for the above compound proposition:

q	r	s	q $\vee$ r $\implies$ s
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	T

Logic is cool.

All humans are mortal. Socrates is a human. Therefore, Socrates is mortal. (Example of a Aristotelian syllogism)

7 Predicate logic

A predicate is a function that returns a boolean value.

Let $P(x) =$ "x is a prime number"
Let $Q(x) =$ "x is a perfect square"

7.1 Quantifiers

Quantifiers are used to express the extent to which a predicate is true.

$\forall x \in \mathbb{N} : P(x) =$ "For all x in the set of natural numbers, P(x) is true"
$\exists x \in \mathbb{N} : P(x) =$ "There exists an x in the set of natural numbers such that P(x) is true"

7.2 Negation of quantifiers

$\neg\forall x \in \mathbb{N} : P(x) =$ "There exists an x in the set of natural numbers such that P(x) is false"
$\neg\exists x \in \mathbb{N} : P(x) =$ "For all x in the set of natural numbers, P(x) is false"

7.3 Quantifiers and truth tables

P(x)	Q(x)	$\forall x \in \mathbb{N} : P(x)$	$\forall x \in \mathbb{N} : Q(x)$
T	T	T	T
T	F	F	F
F	T	F	T
F	F	F	F

P(x)	Q(x)	$\exists x \in \mathbb{N} : P(x)$	$\exists x \in \mathbb{N} : Q(x)$
T	T	T	T
T	F	T	F
F	T	T	T
F	F	F	F

7.4 Quantifiers and negation

$$\neg\forall x \in \mathbb{N} : P(x) = \exists x \in \mathbb{N} : \neg P(x)$$
$$\neg\exists x \in \mathbb{N} : P(x) = \forall x \in \mathbb{N} : \neg P(x)$$

8 Graphs and trees

In this section we will be looking at graphs and trees.

8.1 Graphs

A graph is a set of vertices and edges (V, E), wjhere:

- V is a set of vertices, i.e.: $V = \{v_1, v_2, v_3, ..., v_n\}$
- E is a set of edges, i.e.: $E = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, ..., \{v_n, v_1\}\}$

NOTICE THAT THE SET OF edges E IS A SET OF SETS, meaning that an edge is an unordered pair of vertices.

Also, there are:

- Simple graphs: graphs with no loops or multiple edges
- Multigraphs: graphs with multiple edges between the same two vertices
- Pseudographs: graphs with loops

8.2 Directed graphs

A directed graph is a graph where the edges are directed, i.e.: $(v_1, v_2) \neq (v_2, v_1)$.

8.2.1 Degree of undirected graphs

THe degree of a vertex in an undirected graph is the number of edges incident to it, i.e.: $deg(v) = \#\{e \in E \mid v \in e\}$.

8.2.2 Degree of directed graphs

The degree of a vertex in a directed graph is the number of edges incident to it, i.e.:

- edges that start with v: $deg^+(v) = \#\{e \in E \mid v$ is the first item in the ordered pair $e\}$.
- edges that end with v: $deg^-(v) = \#\{e \in E \mid v$ is the second item in the ordered pair $e\}$.

note that the cardinality of the set of edges incident to a vertex is the sum of the in-degree and out-degree of the vertex, i.e.:

$\sum_{v \in V} deg^+(v) = \sum_{v \in V} deg^-(v) = \#\{e \in E\}$.

8.3 isomorphisms

Two graphs G and H are isomorphic if there exists a bijection $f : V(G) \to V(H)$ such that:

$$\forall u, v \in V(G) : \{u, v\} \in E(G) \iff \{f(u), f(v)\} \in E(H)$$

Use your fantastic intuition and you will be fine Jim.

8.4 Paths and Circuits

A path is a sequence of vertices $v_1, v_2, ..., v_n$ such that $\{v_i, v_{i+1}\} \in E$ for all $i = 1, 2, ..., n-1$.

A circuit is a path that starts and ends at the same vertex.

A path or circuit is simple if all vertices are distinct (does not use the same edge twice).

8.5 connectedness

An undirected graph is connected if there is a path between every pair of vertices.

A directed graph is strongly connected if there is a path between every pair of vertices.

A directed graph is weakly connected if the underlying undirected graph is connected.

There might be strongly connected components in a directed graph, where a strongly connected component is a maximal subset of vertices such that every pair of vertices is strongly connected.

8.6 Eulerian paths and circuits

Eulerians concern edges, not vertices.

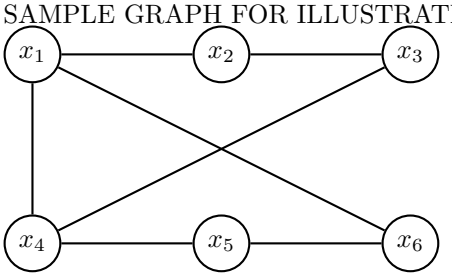
An Eulerian path is a simple path that uses every edge (exactly once, as per previsous definitions of "simple").

An Eulerian circuit is a simple circuit that uses every edge (exactly once, as per previsous definitions of "simple").

Theorem: A connected undirected graph has an Eulerian circuit if and only if every vertex has even degree.

Theorem: A connected undirected graph has an Eulerian path but not an Eulerian circuit if and only if exactly two vertices have odd degree.

SAMPLE GRAPH FOR ILLUSTRATION:



8.7 Hamiltonian paths and circuits

Hamiltonians concern vertices, not edges.

A Hamiltonian path is a simple path that uses every vertex (exactly once, as per previsous definitions of "simple").

A Hamiltonian circuit is a simple circuit that uses every vertex (exactly once, as per previsous definitions of "simple").

Unlike Eulerians, there is no easy way to determine if a graph has a Hamiltonian path or circuit.

Ayyy lmao.

8.8 Trees

A tree is a connected graph with no circuits.

A rooted tree is a tree with a designated root vertex.

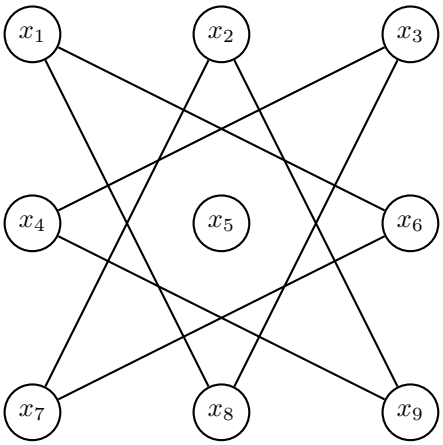
The following properties are equivalent for a graph G:

- G is a tree.
- There exists exactly one path between any pair of vertices in G.
- G is connected and has exactly n - 1 edges.
- G is connected and either n = 1 and G has no edges,or G ¡= 2 and removing any edge in G makes the graph unconnected.
- G has no cycles, and adding an edge between any pair of non-adjacent edges v, v creates a cycle containing v and v.

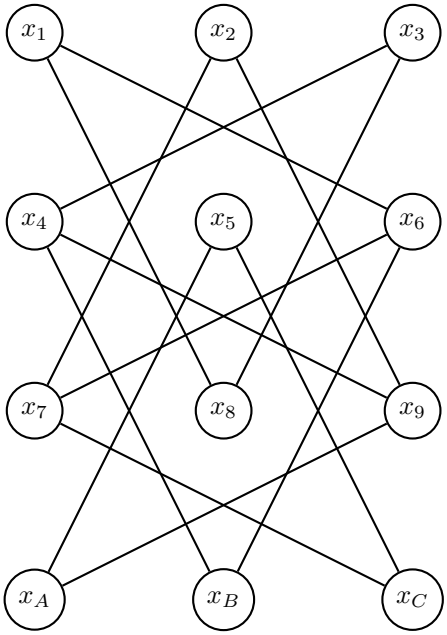
8.9 types of traversal

There are preorder, inorder, and postorder traversals.

- Preorder: visit the root, then recursively visit the left and right subtrees.
- Inorder: recursively visit the left subtree, then visit the root, then recursively visit the right subtree.
- Postorder: recursively visit the left and right subtrees, then visit the root.



Now onto the 3 × 4 example.


x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_A
x_B
x_C

9 Proofs

- Definitions are used to create new concepts in terms of existing ones
- Axioms are statements that are self evident or assumed to be true
- Theorems, Lemmas, Propositions are statements that are shown to be true (via a proof)
- Corollary is a property that follows from a Theorem, Lemma or proposition, e.g., as an instance
- A proof is a sequence of statements that form an argument using definitions axioms and proof rules
- Fanccy ending like "quod erat demonstrandum" is used to indicate the end of a proof.

9.1 Proof methods

There are four proof methods that we will be using:

- Direct proof: Assume the hypothesis and show that the conclusion follows
- Indirect proof (contrapositive, contradiction)
  - Contrapositive: Assume the negation of the conclusion and show that the negation of the hypothesis follows
  - Contradiction: Assume the hypothesis and the negation of the conclusion and show that a contradiction follows
- Existence proofs
  - Demonstrate an example that satisfies the hypothesis
- Proof by induction
  - Show that the hypothesis holds for the base case
  - Show that if the hypothesis holds for an arbitrary case assumed to be true, then it holds for the next case (k + 1)

9.2 outline

Proofs are hard to write, but easy to read. Proof that the sum of an even number and odd number is odd:

$$
\begin{aligned}
&\text{Let } x = 2k \text{ where } k \in \mathbb{Z} \\
&\text{Let } y = 2k + 1 \text{ where } k \in \mathbb{Z} \\
&x + y = 2k + 2k + 1 \\
&\qquad = 4k + 1 \\
&\qquad = 2(2k) + 1 \\
&\qquad = 2k' \text{ where } k' \in \mathbb{Z} \\
&\qquad \therefore x + y \text{ is odd}
\end{aligned}
$$

10 Combinatorics and probability

10.1 Combinatorics

10.1.1 Product rule

The product rule states that the number of ways to perform a sequence of k tasks, where the first task can be performed in n1 ways, the second task can be performed in n2 ways, and so on, is n1 × n2 × ... × nk.

EXTENDED PRODUCT RULE: The number of ways to perform a sequence of k tasks, where the first task can be performed in n1 ways, the second task can be performed in n2 ways, and so on, and the kth task can be performed in nk ways, is n1 × n2 × ... × nk.

$$
\begin{aligned}
&\text{Let } n_1, n_2, n_3, ..., n_k \in \mathbb{N} \\
&\text{The number of ways to perform a sequence of k tasks is:} \\
&n_1 \times n_2 \times n_3 \times ... \times n_k
\end{aligned}
$$

10.1.2 Sum rule

The sum rule states that the number of ways to perform a task is the sum of the number of ways to perform it in each of several different cases.

$$
\begin{aligned}
&\text{Let } n_1, n_2, n_3, ..., n_k \in \mathbb{N} \\
&\text{The number of ways to perform a task is:} \\
&n_1 + n_2 + n_3 + ... + n_k
\end{aligned}
$$

10.1.3 Permutations

A permutation is an ordered arrangement of objects.

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}$ where $k \leq n$

The number of permutations of n objects taken k at a time is:

$$P(n,k) = \frac{n!}{(n-k)!}$$

10.1.4 Combinations

A combination is an unordered selection of objects.

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}$ where $k \leq n$

The number of combinations of n objects taken k at a time is:

$$C(n,k) = \frac{n!}{k!(n-k)!}$$

The symbol $\binom{n}{k}$ is used to denote the number of combinations of n objects taken k at a time.

Sometimes r is used instead of k.

10.2 Probability

10.2.1 Sample space

The sample space is the set of all possible outcomes of an experiment.

10.2.2 Event

An event is a subset of the sample space.

10.2.3 Probability

The probability of an event is the number of outcomes in the event divided by the number of outcomes in the sample space.

Let S be the sample space and $E \subseteq S$ be an event

$$P(E) = \frac{\#E}{\#S}$$

10.2.4 Probability axioms

- $0 \leq P(E) \leq 1$
- $P(S) = 1$
- If $E_1, E_2, E_3, ...$ are disjoint events, then $P(E_1 \cup E_2 \cup E_3 \cup ...) = P(E_1) + P(E_2) + P(E_3) + ...$

10.2.5 Probability rules

- $P(E^c) = 1 - P(E)$
- $P(E \cup F) = P(E) + P(F) - P(E \cap F)$
- $P(E \cup F \cup G) = P(E) + P(F) + P(G) - P(E \cap F) - P(E \cap G) - P(F \cap G) + P(E \cap F \cap G)$

10.2.6 Conditional probability

The conditional probability of an event E given an event F is the probability of E given that F has occurred.

$$\text{Let } E, F \text{ be events such that } P(F) > 0$$
$$P(E \mid F) = \frac{P(E \cap F)}{P(F)}$$

10.2.7 Independence

Two events E and F are independent if $P(E \cap F) = P(E)P(F)$.

10.2.8 Bayes' theorem

$$\text{Let } E, F \text{ be events such that } P(E) > 0 \text{ and } P(F) > 0$$
$$P(E \mid F) = \frac{P(F \mid E)P(E)}{P(F)}$$

10.2.9 Random variables

A random variable is a function that assigns a real number to each outcome in the sample space.

10.2.10 Probability distribution

The probability distribution of a random variable X is the function that assigns a probability to each real number x.

$$\text{Let } X \text{ be a random variable}$$
$$p(x) = P(X = x)$$

10.2.11 Expected value

The expected value of a random variable X is the sum of the product of each possible value of X and its probability.

Let X be a random variable

$$E(X) = \sum_{x \in X(S)} x \cdot p(x)$$

10.2.12 Variance

The variance of a random variable X is the expected value of the squared difference between X and its expected value.

Let X be a random variable

$$\sigma_X =$$

10.2.13 Standard deviation

The standard deviation of a random variable X is the square root of its variance. here is the formula for standard deviation:

$$\text{Standard deviation} = \sqrt{\text{Variance}}$$

Or more formally:

$$\sigma = \sqrt{Var(X)}$$

10.2.14 Bernoulli trials

A Bernoulli trial is an experiment with two possible outcomes: success or failure, such that:

- The probability of success is p
- The probability of failure is 1 - p
- The trials are independent

10.2.15 Binomial distribution

The binomial distribution is the probability distribution of the number of successes in a sequence of n independent Bernoulli trials.

Let X be a random variable

$$p(x) = P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$$

11 Previous mistakes list

- A group of animals consists of m cats and m dogs. How many ways are there to arrange the animals in a line if cats must alternate with dogs?

  - $2(m!)^2$ because: The line can either start with a cat or with a dog, giving two possibilities.
    Once this is fixed, the cats can be permuted arbitrary, giving $m!$ possibilities, and the the dogs can be permuted arbitrary, giving $m!$ possibilities as well. Overall, there are $2(m!)^2$ possibilities.

- Below, you will see some specific types of logical statements.
  Sort them according to the following order:
  – rank top (number 1) those than are most often necessarily true
  – rank bottom (number 4) those than are most often necessarily false
  – in between cases a and b, rank bottom those that are necessarily false more often.

  - Contradiction: always false
    Tautology: always true
    Contingent: This is sometimes true, sometimes false. Since it is necessarily at least once false, then it ranks lower than satisfiable.
    Satisfiable: At least once trues. This can still mean always true and therefore never false.

- There can be a language corresponding to regular expression $\emptyset$.