Discrete Math

Notes

May 4, 2025

Contents

Chapter 1

Propositional Logic

1.1 Proposition

A statement that is either true or false but not both.

These statements are denoted by small case variable names

Eg: p: roses are red

A proposition that cannot be expressed in terms of simpler propositions are called atomic propositions.

The area of logic that deals with propositions is called the propositional calculus or propositional logic

1.1.1 Compound Propositions

compound propositions, are formed from existing propositions using logical operators

Some operators combining proposition

Operator	Name	Logical Function	Translation	Remarks
~ ¬ – ′ N !	tilde logical not	negation	not, it is not the case that	
& ^ .	ampersand logical and dot	conjunction	and, also, moreover, but	
∨	wedge logical or	disjunction	or, unless, nor	
⊃ →	horseshoe	material conditional	if . . . then . . ., only if	
≡ ↔	triple bar double arrow	material bi-conditional (equivalence)	if and only if, just in case	
p⊕ *q*		exclusive or XOR		

Truth tables

Basic Setup	Negation	Conjunction	Disjunction	Material Conditional	Material Bi-conditional	Exclusive or
P Q	¬P ¬Q	P & Q	P ∨ Q	P ⊃ Q	P ≡ Q	p ⊕ q
T T	F F	T	T	T	T	F
T F	F T	F	T	F	F	T
F T	T F	F	T	T	F	T
F F	T T	F	F	T	T	F

1.1.2 Conditionals

These are other important ways of combining proposition to make new propositions using conditionals (if this then that)

Implication (p $\rightarrow$ q)

If p then q

p is called the hypothesis (or antecedent or premise) and q is called the conclusion (or consequence).

TABLE 5 The Truth Table for the Conditional Statement $p \to q$.

p	q	$p \to q$
T	T	T
T	F	F
F	T	T
F	F	T

Converse contrapositive and Inverse

The proposition $q \to p$ is called the converse of $p \to q$.

The proposition $\not q \to \not p$ is called the contrapositive of $p \to q$.

The proposition $\not p \to \not q$ is called the inverse of $p \to q$.

When two compound propositions always have the same truth values, regardless of the truth

values of its propositional variables, we call them equivalent.

The contrapositive, but neither the converse or inverse, of a conditional statement is equivalent to it.

The converse and the inverse of a conditional statement are also equivalent to each other.

Biconditionals ($p \leftrightarrow q$)

P iff (if and only if) q

the statement $p \leftrightarrow q$ is true when both the conditional statements $p \to q$ and $q \to p$ are true and is false otherwise.

Truth table

TABLE 6 The Truth Table for the Biconditional $p \leftrightarrow q$.

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

1.2 Precedence of operators

TABLE 8 Precedence of Logical Operators.

Operator	*Precedence*
$\neg$	1
$\wedge$	2
$\vee$	3
$\rightarrow$	4
$\leftrightarrow$	5

1.3 Propositional equivalences

Tautology – a compound proposition that is always true

Contradiction – a compound proposition that is always false

Contingency – a compound proposition that is neither tautology nor contradiction

Logical equivalence – compound proposition that have the same truth values in all possible cases

Denoted by p ≡ q

Some examples of logical equivalences

TABLE 6 Logical Equivalences.

Equivalence	***Name***
$p \wedge \mathbf{T} \equiv p$ $p \vee \mathbf{F} \equiv p$	Identity laws
$p \vee \mathbf{T} \equiv \mathbf{T}$ $p \wedge \mathbf{F} \equiv \mathbf{F}$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws
$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's laws
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv \mathbf{T}$ $p \wedge \neg p \equiv \mathbf{F}$	Negation laws

TABLE 7 **Logical Equivalences Involving Conditional Statements.**
$p \to q \equiv \neg p \vee q$
$p \to q \equiv \neg q \to \neg p$
$p \vee q \equiv \neg p \to q$
$p \wedge q \equiv \neg(p \to \neg q)$
$\neg(p \to q) \equiv p \wedge \neg q$
$(p \to q) \wedge (p \to r) \equiv p \to (q \wedge r)$
$(p \to r) \wedge (q \to r) \equiv (p \vee q) \to r$
$(p \to q) \vee (p \to r) \equiv p \to (q \vee r)$
$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$

TABLE 8 **Logical Equivalences Involving Biconditional Statements.**
$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$
$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$
$p \leftrightarrow q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$
$\neg(p \leftrightarrow q) \equiv p \leftrightarrow \neg q$

Staisfiabilty - A compound statement is called satisfiable if there exist an assignment of truth values if that assignment makes the statement true.

1.4 Predicates and quantifiers

Predicate - predicate is functional proposition that takes a subject apples logic to the subject and gives a truth values.

P(x) is computer x is under attack.

When checking the correctness of a computer program The statements that describe valid input are known as preconditions and the conditions that the output should satisfy when the program has run are known as postconditions.

Quantifiers - to quantify a predicate (to tell whether it is T or F) we need to assign a value to subject however there are other methods of quantification,

universal quantification, which tells us that a predicate is true for every element under consideration.

The universal quantification of P(x) is the statement "P(x) for all values of x in the domain."

The notation ∀xP(x) denotes the universal quantification of P(x). Here ∀ is called the universal quantifier.

We read ∀xP(x) as "for all xP(x)" or "for every xP(x)." An element for which P(x) is false is called a counterexample to ∀xP(x).

existential quantification, which tells us that there is one or more element under consideration for which the predicate is true.

The existential quantification of P(x) is the proposition "There exists an element x in the domain such that P(x)." We use the notation ∃xP(x) for the existential quantification of P(x). Here ∃ is called the existential quantifier.

quantification ∃xP(x) is read as "There is an x such that P(x)," "There is at least one x such that P(x),"

uniqueness quantifier, denoted by ∃! or ∃1. The notation ∃!xP(x) [or ∃1xP(x)] states "There exists a unique x such that P(x) is true."

Precedence of quantifiers

The quantifiers ∀ and ∃ have higher precedence than all logical operators from propositional calculus.

1.4.1 Equivalences

TABLE 2 De Morgan's Laws for Quantifiers.

Negation	*Equivalent Statement*	*When Is Negation True?*	*When False?*
$\neg\exists x P(x)$	$\forall x \neg P(x)$	For every x, $P(x)$ is false.	There is an x for which $P(x)$ is true.
$\neg\forall x P(x)$	$\exists x \neg P(x)$	There is an x for which $P(x)$ is false.	$P(x)$ is true for every x.

TABLE 1 Quantifications of Two Variables.

Statement	*When True?*	*When False?*
$\forall x \forall y P(x, y)$ $\forall y \forall x P(x, y)$	$P(x, y)$ is true for every pair x, y.	There is a pair x, y for which $P(x, y)$ is false.
$\forall x \exists y P(x, y)$	For every x there is a y for which $P(x, y)$ is true.	There is an x such that $P(x, y)$ is false for every y.
$\exists x \forall y P(x, y)$	There is an x for which $P(x, y)$ is true for every y.	For every x there is a y for which $P(x, y)$ is false.
$\exists x \exists y P(x, y)$ $\exists y \exists x P(x, y)$	There is a pair x, y for which $P(x, y)$ is true.	$P(x, y)$ is false for every pair x, y.

1.5 Rules OF Inference

An argument in propositional logic is a sequence of propositions. All but the final proposition in the argument are called premises and the final proposition is called the conclusion. An argument is valid if the truth of all its premises implies that the conclusion is true.

An argument form in propositional logic is a sequence of compound propositions involving propositional variables. An argument form is valid if no matter which particular propositions are substituted for the propositional variables in its premises, the conclusion is true if the premises are all True

TABLE 1 Rules of Inference.

Rule of Inference	*Tautology*	*Name*
p $p \to q$ $\therefore q$	$(p \wedge (p \to q)) \to q$	Modus ponens
$\neg q$ $p \to q$ $\therefore \neg p$	$(\neg q \wedge (p \to q)) \to \neg p$	Modus tollens
$p \to q$ $q \to r$ $\therefore p \to r$	$((p \to q) \wedge (q \to r)) \to (p \to r)$	Hypothetical syllogism
$p \vee q$ $\neg p$ $\therefore q$	$((p \vee q) \wedge \neg p) \to q$	Disjunctive syllogism
p $\therefore p \vee q$	$p \to (p \vee q)$	Addition
$p \wedge q$ $\therefore p$	$(p \wedge q) \to p$	Simplification
p q $\therefore p \wedge q$	$((p) \wedge (q)) \to (p \wedge q)$	Conjunction
$p \vee q$ $\neg p \vee r$ $\therefore q \vee r$	$((p \vee q) \wedge (\neg p \vee r)) \to (q \vee r)$	Resolution

TABLE 2 Rules of Inference for Quantified Statements.

Rule of Inference	*Name*
$\dfrac{\forall x P(x)}{\therefore\ P(c)}$	Universal instantiation
$\dfrac{P(c) \text{ for an arbitrary } c}{\therefore\ \forall x P(x)}$	Universal generalization
$\dfrac{\exists x P(x)}{\therefore\ P(c) \text{ for some element } c}$	Existential instantiation
$\dfrac{P(c) \text{ for some element } c}{\therefore\ \exists x P(x)}$	Existential generalization

1.6 Wel-Formed Formulas

- if P is a propositional variable it is a wff
- if a is a wff $\neg a$ is a wff
- if α and β are wff then -
  - $\alpha \vee \beta$
  - $\alpha \wedge \beta$
  - $\alpha \implies \beta$
  - $\alpha \iff \beta$

  all are wff.

1.6.1 Normal Form

A formula is in *disjunctive normal form* DNF if it is sum of elementry products
eg : $P \vee (Q \wedge R) \vee (\neg Q \wedge R)$ etc.

A formula is in *principal disjunctive normal form* PDNF if it is sum of products or min terms only difference from DNF is that while in DNF the length of products need not be same but in PDNF all the products should have same length.
eg :$(Q \wedge R) \vee (\neg Q \wedge R)$ etc.
A formula is in *conjunctive normal form* CNF if it is products of elememtry sum
eg : $P \wedge (Q \vee R) \wedge (\neg Q \vee R)$ etc.
A formula is in *principal conjunctive normal form* PCNF if it is products of sum or max terms only difference from CNF is that while in CNF the length of sum need not be same but in PCNF all the sum should have same length.
eg : $(Q \vee R) \wedge (\neg Q \vee R)$ etc.

Chapter 2

Permutations and combinations

2.1 Known Formula

- number of Permutations of n things taken all together, when p things are identical of one type q of another type and r of yet another type is $\frac{n!}{p!q!r!}$
- number of circular arrangements if clockwise and anticlockwise arrangements differ (persons siting on a round table) is $(n-1)!$ if clockwise and anticlockwise does not differ (beads in garland) than it is $\frac{(n-1)!}{2}$
- number of combinations of n different things taking one or more at a time is $2^n - 1$
- Total number of selection of one or more things from p identical things of one type, q identical things of another type, r identical things of third type, and n different things is $(p+1)(q+1)(r+1)2^n - 1$
- $(x+y)^n =^n c_0 x^0 y^n +^n c_1 x^1 y^{n-1} +^n c_2 x^2 y^{n-2} + ... +^n c_n x^n y^0$

Chapter 3

Sets

3.1 Relation

Relation of two sets A and B is the subset of the cross product A x B let $R \subset AxB$ if $(x, y) \in R$ then we say x is related to y by the relation R.

we call Domain(R) = x:(x,y) ∈ R and Range(R) = y:(x,y) ∈ R

3.1.1 R-relative sets

for any element x such that (x,y) ∈ R, R-relative set $R(x) = \{y|xRy\}$

3.1.2 Representation of Relations

Set Builder : $R = \{(x, y) : x < y\}$
Listing : $\{(1, 2), (2, 3)\}$
Matrix : $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
here if (x,y) ∈ R then m[row representing x,column representing y]=1 other wise 0
arrow diagram : arrow between two sets representing this related to that
Diagraph : A directed graph representing the relation

3.1.3 Composition of relation

RoS = {(x,y):(x,z) ∈ S & (z,y) ∈ R}

3.1.4 Types of relation

Reflexive relation : A relation R on A is called reflexive, if

$$\boxed{\forall x \in A(x,x) \in R}$$

Symmetric relation : A relation R on A is called Symmetric, iff

$$\boxed{(x,y) \in R \implies (y,x) \in R}$$

Anti-symmetric relation : A relation R on A is called Anti-symmetric, if

$$\boxed{(x,y) \in R \implies (y,x) \notin R \text{ unless x=y}}$$

Transitive relation : A relation R on A is called transitive, iff

$$\boxed{(x,y),(y,z) \in R \implies (x,z) \in R}$$

irreflexive relation : A relation R on A is called irreflexive, if

$$\boxed{\forall x \in A(x,x) \notin R}$$

Asymmetric relation : A relation R on A is called asymmetric, iff

$$\boxed{(x,y) \in R \implies (y,x) \notin R}$$

Equivalence relation : A relation R on A is called equivalence, if R is reflexive, symmetric and transitive simultaneously.

Partial Order relation : A relation R on A is called Partial Order, if R is reflexive, antisymmetric and transitive simultaneously.

Total Order relation : A relation R on A is called total Order, if R is reflexive, antisymmetric,transitive and the elements are comparable simultaneously.

3.1.5 Equivalence Classes & Quotient Set

Let R be an equivalence relation on A x A

Equivalence class of x ∈ A = [x] = {y:xRy}

The set of all equivalence Classes is called quotient set

3.2 Partitioning a Set

Partition of a set is a set of non empty subsets $(A_1, A_2, A_3...A_n)$ such that $A_1 \bigcup A_2 \bigcup A_3 \bigcup .....A_n = A$ and

$A_i \cap A_j = \phi$ for every combinations of i and j

3.3 Groups

Algebraic Structure : A non empty set S along with one or more binary operations. eg: (S,*),(R,+,x) etc.
semi Group : An Algebraic Structure (G,*) is called semi Group if the binary operation * is closed on G and is associative on G.
Monoid : An Algebraic Structure (M,*) is called Monoid if the binary operation * is closed on M,

- is associative on M
- their exists an identity element in M

Group : An Algebraic Structure (G,*) is called group if the binary operation * is closed on G,

- is associative on G
- their exists an identity element in G
- their exists an inverse in G

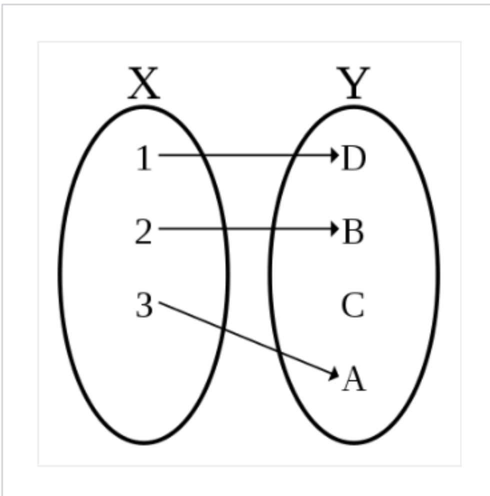
Abelian Group or Commutative group A group G is said to be Abelian or commutative. if the binary relation on G

- is associative on G
- their exists an identity element in G
- their exists an inverse in G
- is commutative on G

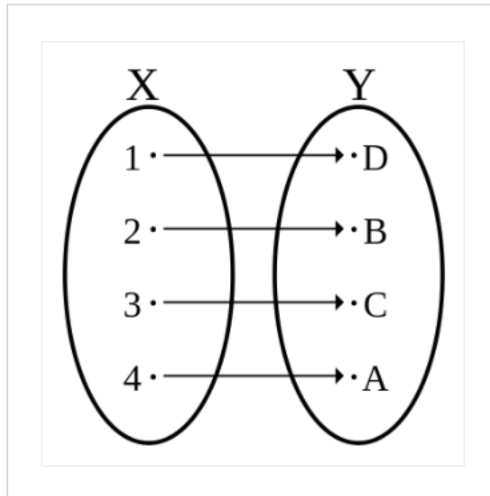
Cayley Table : The binary operation table of a finite group are called cayley tables.
Cyclic Group : A group (G,*) is called cyclic if every element of G can be written as $a^n n \in Integer$, a is called the generator of G.
sub Group : if a group is a subset of another group then the subset is called sub group.
Coset : let (G,*) be a group and (H,*) be a sub groub then for any $a \in G$
$aH = \{a * h | h \in H\}$ is left coset and $Ha = \{h * a | h \in H\}$ is right coset
if right and left coset are equal then H is said to be **Normal subgroup**.

3.3.1 Lattice

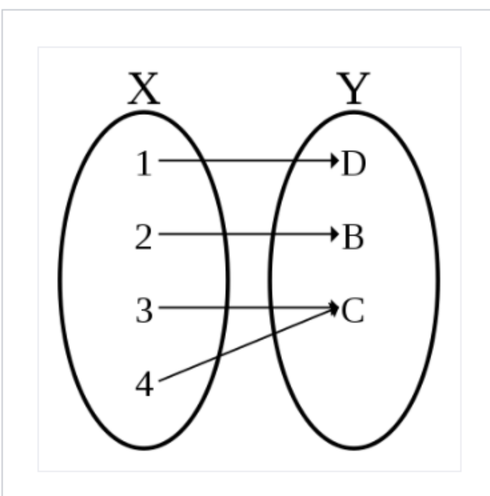
Poset : A non-empty set P, together with a partial order binary relation R is called a partial order set or poset.
hasse diagrma and poset diagram are same
Toset : a poset in which every pair of element is compareable.
hasse Diagram : a diagram to represent poset see book for further detail.
maximal and minimal element : the elements that are highest on a hasse diagram are maximal element while the element which are lowest are called minimal element.
Upper bounds and lower bounds : starting from two elements on travarsing the hasse diagram upwards all the points where the two travarsal meets are upper bounds and the first place where they meet is lowest upper bound (suprimum) and vice versa.
Lattice : A poset is said to be a lattice if for every pair of elements in lattice there exists suprimum and infimum.



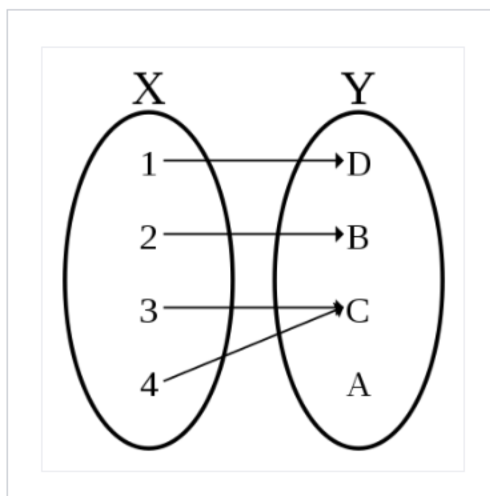
An injective non-surjective function (injection, not a bijection)



An injective **surjective** function (bijection)



A non-injective **surjective** function (surjection, not a bijection)



A non-injective non-surjective function (also not a bijection)

Chapter 4

Graph Theory

4.1 Intro

Graph : A graph G is a triple consisting of vertex set, edges set and the relation that associates each edges to its end points.