<u>QUICK RECAP</u>

AND

p	**q**	**p ^ q**
0	0	0
0	1	0
1	0	0
1	1	1

OR

p	**q**	**p V q**
0	0	0
0	1	1
1	0	1
1	1	1

NOT

p	**¬p**
0	1
1	0

IMPLICATION

p	**q**	**p → q**
0	0	1
0	1	1
1	0	0

1	1	1

XOR

p	q	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

EQUIVALENCE

p	q	$p \Leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

PIGEONHOLE PRINCIPLE

- If ‘n+1’ objects are to be partitioned into ‘n’ pieces at least one part must contain at least two objects.(Assuming objects cannot be divided into fractions)
- i.e., if n+1 objects are to be placed into n boxes, then at least one box contains two or more objects.
- Likewise, if there are (nk+1) objects and have to be distributed among k persons then one must have (n+1) items.

  Example:-
  n=10,
  k=5
  = (nk + 1) / n

= 51 / 10
In this case, each person will receive 5 objects but one person will receive the extra 1 object.

- There is no Limit on the number of formulas we construct using the rules of syntax which can be infinite.
- But, the evaluation using the truth table mechanism shows that the number of formulas can be 2^(2^k) which can be some number but not the same as infinite so in this case it will be finite.
- We can make infinite formulas using rules of syntax but there can be only 2^(2^k) distinct formulas.
- So, according to the Pigeonhole principle some two of them will be representing the same formula.

Example 1:-

1. $p \rightarrow q$
2. $\neg p \vee q$
3. $\neg(p \wedge (\neg q))$

- All 3 of these formulas are semantically same but syntactically different.
- Formulas are semantically same when they have the same truth table.
- With these 2 variable we will be able to make 16 formulas($2^{2^{\wedge}k} \rightarrow 2^{2^{\wedge}2}$)

Example 2:-

1. $\neg(p \oplus q)$
2. $p \Leftrightarrow q$

p	**q**	**p ⇔ q**
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**1**

p	**q**	**p ⊕ q**	**¬(p ⊕ q)**
0	0	0	**1**
0	1	1	**0**
1	0	1	**0**
1	1	0	**1**

- Both these formulas are also semantically same.

So by **pigeonhole principle**, it says that though we can make infinitely many formulas, they all fall under 16 equivalent classes.

There are two standard ways for writing propositional formula

1. CNF (Conjunctive normal form)
2. DNF (Disjunctive normal form)

Example:-

Converting (p ⟶ (q ⟶ r)) into CNF and DNF

DNF

(¬p) v (q ⟶ r) ¬p v (¬q v r) (¬p) v (¬q) v (r)	3 CLAUSE

CNF

(¬p v ¬q v r)	Only 1 CLAUSE

1) Conjunctive means **AND OF OR'S** in simpler terms –
   (v v) ^ (v) ^ (v)   there can be any number of OR, AND
   In CNF the outer operator (outside bracket) is AND, and inner operator is OR.
   Example: $A \rightarrow (B \land C) \equiv \neg A \lor (B \land C) \equiv (\neg A \lor B) \land (A \neg \lor C)$

2) Disjunctive means **OR OF AND'S**
   (^) v (^) v (^ ^)   there can be any number of OR, AND
   In DNF the outer operator (outside bracket) is OR and inner operator is AND.
   Example: P ^ (P->Q) <=> P ^ (~P V Q) <=> (P ^ ~P) V (P ^ Q

   - Whatever is inside ( ) are clauses.

- Variables → p1, p2, p3...pn
- Literals → Variables or their negation. Variable without a negation sign in positive literal otherwise it's called negative literal or negated literal.
  In proposition logic, variables are atomic prepositions/atomic variables/boolean variables
- Clauses are literals and in case of :-
  1. CNF - Conjunction of clauses where each clause is a disjunction/or of literals.
  2. DNF - Disjunction of clauses where each clause is a conjunction/AND of literals.

<u>**Evaluating Boolean formula for some assignment of truth values**</u>

<u>*$((p \rightarrow q) \rightarrow r)$*</u>

If,   p=T
      q=T
      r=F

- Value for this will be F.
- This evaluation of truth value of formula under specific assignment.
- There will be a total of 8 assignments under which it is checked.(2^3)

<u>Partial assignment/evaluation</u> <u>in ((p → q) → r)</u>

Step 1: If q = F then,

¬ p	**p**	**q**	**p → q**
F	T	F	**F**
T	F	F	**T**

p → q   can also be written as ¬p
Therefore,   ((¬p) → r)

Step 2: If q = T then,

p	**q**	**p → q**
T	T	**T**
F	T	**T**

As, p → q is true
Therefore T→ r

Step 3: Now If, r = T

p → q / T	**r**	**T→ r**
T	T	**T**

When r is T, T→ r is also T

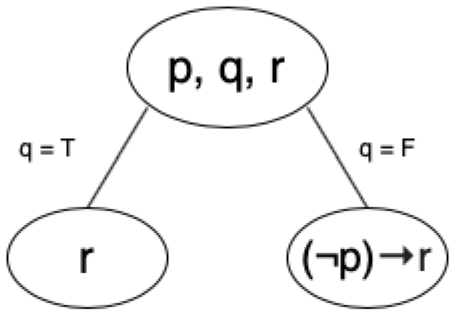
Step 4: Now If,   r = F

p → q /T	**r**	**T→ r**
T	**F**	**F**

When r is F, T→ r is also F

Step 5: This whole process is called Evaluation tree

Evaluation tree for the above example



- A formula is satisfiable if it has at least one satisfying assignment. A satisfying assignment is an assignment which evaluates to TRUE. A formula which has no such assignment is called unsatisfiable..

ABSTRACT TRUTH TABLE

	VAR 1	**...**	**VAR K**	
A1				
A2				**Combination of 0 and 1**
. **.**				
A 2^K				

- IN DNF, it’s AND of everything therefore only 1 will make it TRUE and rest all in make it FALSE.
- IN CNF, it’s OR of everything therefore only 1 will make it FALSE and rest all in make it TRUE.
- FALSE = 0, TRUE = 1
- For the case of FALSE we will use CNF.
  For the case of TRUE we will use DNF.

CNF construction

1. Take all rows where the formula evaluate to 0/F
2. Each row is a clause in the CNF.
3. The clause for a row is disjunction/OR of positive literals for variables which are false/0 and negate the literals for variables which are true/1.

<u>EXAMPLE:-</u>

	p	**q**	**r**	**Ψ**
A1	0	0	0	0
A2	0	0	1	1
A3	0	1	0	1
A4	0	1	1	0
A5	1	0	0	0
A6	1	0	1	1
A7	1	1	0	1
A8	1	1	1	0

CNF

Step 1: Take all rows where the formula evaluate to 0

A1

A4

A5

A8

Step 2: A1 $(p \vee q \vee r)$

A4 $(p \vee (\neg q \vee \neg r))$

A5 $(\neg p \vee q \vee r)$

A8 $(\neg p \vee \neg q \vee \neg r)$

$(p \vee q \vee r) \wedge (p \vee (\neg q \vee \neg r)) \wedge (\neg p \vee q \vee r) \wedge (\neg p \vee \neg q \vee \neg r)$

=Evaluates to 0

DNF

Step 1: Take all rows where the formula evaluate to 1

A2

A3

A6

A7

Step 2: A2 $(\neg p \wedge \neg q \wedge r)$

A3 $(\neg p \wedge q \wedge \neg r)$

A6 $(p \wedge \neg q \wedge r)$

A7 $(p \wedge q \wedge \neg r)$

$(\neg p \wedge \neg q \wedge r) \vee (\neg p \wedge q \wedge \neg r) \vee (p \wedge \neg q \wedge r) \vee (p \wedge q \wedge \neg r)$

=Evaluates to 1