

UNIT-I

Statements and notations :-

A proposition or statement is a declarative sentence that is either true or false but not both.

For instance, the following are propositions:

"Paris is in France" (true),

"London is in Denmark" (false),

" $2 < 4$ " (true), " $4 = 7$ (false)".

However the following are not propositions:

"What is your name?" (This is a question),

"Do your homework" (this is a command),

"This sentence is false" (neither true nor false),

" x is an even number" (it depends on what x represents),

"Socrates" (it is not even a sentence).

The truth or falsehood of a proposition is called its truth value,

Connectives:

Connectives are used for making compound propositions. The main ones are the following (P and Q represent given propositions).

Name	Represented	Meaning
Negation	$\neg P$	"not P "
Conjunction	$P \wedge q$	" P and q "
Disjunction	$P \vee q$	" P or q (or both)"
Exclusive OR	$P \oplus q$	"either P or q , but not both"
Implication	$P \rightarrow q$	"if P then q "
Biconditional	$P \leftrightarrow q$	" P if and only if q "

Truth Tables :-

Logical identity :-

logical identity is an operation on one logical value, typically the value of a proposition that produces a value of true if its operand is true if its operand is true and a value of false if its false

The true table for the logical identity operator is as follows:

Logical identity	
P	P
T	T
F	F

logical negation

logical negation is an operation on the logical value, typically the value of a proposition that produces a value of true if its operand is false and a value of false if its operand is true.

The truth table for $\text{NOT } p$ (also written as $\neg p$ or $\sim p$) is as follows:

logical Negation

p	$\neg p$
-----	----------

T	F
-----	-----

F	T
-----	-----

Binary operations

Truth table for all binary logical operations

There is a truth table giving definitions of all of the possible truth functions of 2 binary variables (P, Q are thus boolean variables).

$P \oplus Q$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
T	T	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T
F	F	F	F	F	T	T	F	F	F	F	F	T	T	T	T	T
F	T	F	F	T	T	F	F	T	T	F	F	T	F	T	T	T
F	F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	T

key: 0, false, contradiction

1. NOR, Logical NOR

2. converse nonimplication

3. $\neg p$, Negation

4. Material nonimplication

5. $\neg q$, Negation

6. XOR, Exclusive disjunction

7. NAND, Logical NAND

8. AND, Logical conjunction

9. xNOR, if and only if, logical biconditional

10. q. projection function

11. if / then, logical implication

12. P. projection function

B. then/if, converse implication

14. OR, Logical disjunction

15. true, Tautology.

Logical conjunction

logical conjunction is an operation on two logical value typically the values of two propositions, that produces a value of true if both of its operands are true

The truth table for $p \text{ AND } q$ (also written as $p \wedge q$, $p \& q$, or $p \cdot q$) is as follows:

logical conjunction

P	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

In ordinary language terms. if both p and q are true, then the conjunction $p \wedge q$ is true. for all other assignments of logical value to p and to q the conjunction $p \wedge q$ is false.

It can also be said that if p, the $p \wedge q$ is q. Otherwise $p \wedge q$ is p.

logical disjunction:-

logical disjunction is an operation on two logical values typically the value of two propositions, that produces a value of true if at least one of its operands is true.

The truth table for $p \text{ or } q$ (also written as $p \vee q$, $p \parallel q$, $\text{or} p, q$) is as follows:

logical Disjunction

P	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

logical implication:-

logical implication and the material conditional are both associated with an operation on two logical values, typically the values of two propositions, that produces a value of false just in the singular case the first operand is true and the second operand is false. the truth table associated with the material conditional if p then q (symbolized as $p \rightarrow q$) and the logical implication p implies q (symbolized as $p \Rightarrow q$) as follows

logical implication

P	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

logical equality

Logical equality is an operation on two logical values, typically the values of two propositions, that produces a value of true if both operands are false or both operands are true. The truth table for $P \text{ XNOR } q$ (also written as $P \leftarrow q$, $P = q$, or $P \equiv q$) is as follows,

logical equality

P	q	$P \equiv q$
T	T	T
T	F	F
F	T	F
F	F	T

exclusive disjunction :-

Exclusive disjunction is an operation on two logical values, typically the values of two propositions, that produces a value of true if one but not both of its operands is true. The truth table for $P \text{ XOR } q$ (also written as $P \oplus q$, or $P \neq q$) is as follows;

Exclusive Disjunction

P	q	$P \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

The logical NAND is a operation on two logical values, typically the values of two propositions, that produces a value of false if both of its operands are true. In other words, it produces a value of true if at least one of its operands is false. The truth-table for $P \text{ NAND } q$ (also written as $P \uparrow q$ or $\bar{P}q$) is as follows

logical NAND

P	q	$P \uparrow q$
T	T	F
T	F	T
F	T	T
F	F	F

It is frequently useful to express a logical operation as a compound operation, that is, an operation that is built up of composed from other operations. Many such compositions are possible, depending on the operations that are taken as basic or "primitive" and the operations that are taken as composite or "derivative". In the case of logical NAND, it is clearly expressible as a compound of NOT and AND. The negation of a conjunction $\neg(P \wedge q)$, and the disjunction of negations $(\neg P) \vee (\neg q)$ can be tabulated as follows.

P	q	$P \wedge q$	$\neg(P \wedge q)$	$\neg P$	$\neg q$	$(\neg P) \vee (\neg q)$
T	T	T	F	F	F	F
T	F	F	T	F	T	T
F	T	F	T	T	F	T
F	F	F	T	T	T	T

Logical NOR

The logical NOR is an operation on two logical values, typically the values of two propositions, that produces a value of true if both of its operands are false. In other words, it produces a value of false if at least one of its operands is true. It is also known as the Peirce arrow after its inventor, Charles Sanders Peirce, and is a sole sufficient operator.

The truth table for $P \text{ NOR } q$ (also written as $P+q$ or $P \perp q$) is as follows:

logical NOR

P	q	$P+q$
T	T	F
T	F	F
F	T	F
F	F	T

The negation of a disjunction $\neg(P \vee q)$, and the conjunction of negations $(\neg P) \wedge (\neg q)$ can be tabulated as follows.

P	q	$P \vee q$	$\neg(P \vee q)$	$\neg P$	$\neg q$	$(\neg P) \wedge (\neg q)$
T	T	T	F	F	F	F
T	F	T	F	F	T	F
F	T	T	F	T	F	F
F	F	F	T	T	T	T

Inspection of the tabular derivations for NAND and NOR, under each assignment of logical values to the functional arguments p and q , produces the identical pattern of functional values for $\neg(P \wedge q)$ as for $(\neg P) \vee (\neg q)$, and for $\neg(P \vee q)$ as for $(\neg P) \wedge (\neg q)$.

The truth value of a compound proposition depends only on the value of components. writing F for "false" and T for "true", we can summarizing the meaning of the connectives in the following table.

P	q	$\neg P$	$P \wedge q$	$P \vee q$	$P \oplus q$	$P \rightarrow q$	$P \leftrightarrow q$
T	T	F	T	T	F	T	T
T	F	F	F	T	T	F	F
F	T	F	F	T	T	T	F
F	F	T	F	F	F	T	T

Note that \vee represents a non-exclusive or i.e., $P \vee q$ is true when any of p, q is true and also when both are true. On the other hand \oplus represents an exclusive or. i.e., $P \oplus q$ is true only when exactly one of p and q is true.

Well formed formulas

Not all strings can represent propositions of the predicate logic. those which produces a proposition when their symbols are interpreted must follow the rules given below, and they are called wffs (well-formed formulas) of the first order predicate logic.

rules for constructing wffs.

A predicate name followed by a list of variable such as $P(x, y)$, where P is predicate name, and x and y are variables, is called an atomic formula.

- 10
- (1) An atomic formula is a wff.
 - (2) If A is a wff, then $\neg A$ is also a wff.
 - (3) If A and B are wffs, then $(A \vee B)$, $(A \wedge B)$, $(A \rightarrow B)$ and $(A \leftrightarrow B)$.
 - (4) If A is wff and x is a many variable, then $(x)A$ and $(\exists x)A$ are wffs.
 - (5) Only those formulas obtained by using (1) to (4) wffs wffs are constructed using the following rules:
1. True and False are wffs.
 2. Each propositional constant, and each propositional variable (i.e., a variable representing propositions) are wffs.
 3. Each atomic formula (i.e., a specific predicate with variables) is a wff.
 4. If A , B and C are wffs, then so are $\neg A$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, and $(A \leftrightarrow B)$.
 5. If x is a variable, and A is a wff, then so are $\forall x A$ and $\exists x A$.

Example: "The capital of Virginia is Richmond." is a specific proposition. Hence it is a wff by Rule 2. Let B be a predicate name representing "being blue" and let x be a variable. Then $B(x)$ is an atomic formula meaning " x is blue", thus it is a wff by Rule 3 above. By applying Rule 5. to $B(x)$. $\forall x B(x)$ is a wff and so is $\exists x B(x)$. Then by applying Rule 4. to them $\forall x B(x) \wedge \exists x B(x)$ is seen to be a wff. Similarly if R is a predicate name representing "being round" then $R(x)$ is an atomic formula. Hence it is a wff. By applying rule 4 to $B(x)$ and $R(x)$, a wff $B(x) \wedge R(x)$ is obtained.

Tautology, contradiction, contingency:

A proposition is said to be a tautology if its truth value is T for many assignment of truth values to its components.

Example: The proposition $p \vee \neg p$ is a tautology.

A proposition is said to be a contradiction if its truth value is F for any assignment of truth values to its components.

Example: The proposition $p \wedge \neg p$ is a contradiction.

A proposition that is neither a tautology nor a contradiction is called a contingency.

$P \rightarrow P$	$P \vee \neg P$	$P \wedge \neg P$
T	F	T
T	F	T
F	T	T
F	T	T

Equivalence Implication:

We say that the statements r and s are logically equivalent if their truth tables are identical. For example the truth table of $\neg P \vee q$.

P	q	$\neg P \vee q$
T	T	T
T	F	F
F	T	T
F	F	T

shows the $\neg P \vee q$ is equivalent to $P \rightarrow q$. It is easily shown that the statements r and s are equivalent if and only if $r \leftrightarrow s$ is a tautology.

Predicates predicative logic:

A predicate or propositional function is a statement containing variables, for instance " $x+2=7$ ", " x is American", " $x < y$ ", "p is a prime number" are predicates. The truth value of the predicate depends on the value assigned to its variables. for instance if we replace x with 1 in the predicate " $x+2=7$ " we obtain " $1+2=7$ ", which is false. but if we replace it with 5 we get " $5+2=7$ ", which is true. we represent a predicate by a letter followed by the variables enclosed between parenthesis: $P(x)$, $\Theta(x,y)$, etc. An example for $P(x)$ is a value of x for which $P(x)$ is true. A counterexample is a value of x for which $P(x)$ is false. so, 5 is an example for " $x+2=7$ ", while 1 is a counterexample. Each variable in a predicate is assumed to belong to a universe of discourse. for instance in the predicate "n is an odd integer" ' n ' represents an integer, so the universe of discourse of n is the set of all integers. In " x is American" we may assume that x is a human being, so in this case the universe of discourse is the set of all human beings.

Free & Bound variables:

Let's now turn to a rather important topic: this distinction between free variable and bound variables.

Have a look at the following formula:

The first occurrence of x is free, whereas the second and third occurrences of x are bound, namely by the first occurrence of the quantifier \forall . The first and second occurrences of the variable y are also bound, namely by the second occurrence of the quantifier \forall .

Informally, the concept of a bound variable can be explained as follows: Recall that quantifications are generally of the form:

$\forall x \psi$ or $\exists x \psi$

$\exists x (\text{MAN}(x) \wedge (\forall x \text{WALKS}(x))) \wedge \text{HAPPY}(x)$

1. If x occurs within another, embedded, quantification $\forall x \psi$ or $\exists x \psi$, such as the x in $\text{WALKS}(x)$ in our example. Then we say that it is bound by the quantifier of this embedded quantification.

2. Otherwise, we say that it is bound by the top level quantifier.

Here's a full formal simultaneous definition of free and bound.

1. Any occurrence of any variable is free in any atomic formula.
2. No occurrence of any variable is bound in any atomic formula.
3. If an occurrence of any variable is bound in ψ or in ψ , then that some occurrence is bound in $\neg\psi$, $(\psi \rightarrow \psi)$, $(\psi \vee \psi)$, $(\psi \wedge \psi)$. Moreover, that some occurrence is bound in $\forall y \psi$ and $\exists y \psi$ as well, for any choice of variable y .

Rules of inference:-

The two rules of inference are called rules P and T

Rule P: A premise may be introduced at any point in the derivation.

Rule T: A formula may be introduced in a derivation if S is tautologically implied by any one or more of the preceding formulas in the derivation.

Modus ponens:

$$P \rightarrow q$$

$$\frac{P}{\therefore q}$$

Tautology $[P \wedge (P \rightarrow q)] \rightarrow q$

Modus ponens tollen

$$\rightarrow q$$

$$\frac{P \rightarrow q}{\therefore \neg P}$$

Tautology $[\neg q \wedge (P \rightarrow q)] \rightarrow \neg P$

Hypothetical syllogism

$$P \rightarrow q$$

$$q \rightarrow r$$

$$\therefore P \rightarrow r$$

Tautology $[(P \rightarrow q) \wedge (q \rightarrow r)] \rightarrow (P \rightarrow r)$

Disjunctive syllogism

$$P \vee q$$

$$\frac{\neg P}{\therefore q}$$

Tautology $((P \vee q) \wedge \neg p) \rightarrow q$

Addition

$$\frac{P}{\therefore P \vee q}$$

Tautology $P \rightarrow (P \vee q)$

Simplification

$$\frac{P \wedge q}{\therefore P}$$

Tautology $(P \wedge q) \rightarrow q$

Conjunction

$$\frac{\begin{matrix} P \\ q \end{matrix}}{\therefore P \wedge q}$$

Tautology $(\neg(P) \wedge (\neg q)) \rightarrow (P \wedge q)$

Resolution

$$P \vee q$$

$$\frac{\neg P}{\neg P \vee r}$$

$$\therefore q \vee r$$

Tautology $[(P \vee q) \wedge (\neg P \vee r)] \rightarrow (q \vee r)$

Equivalence laws.

Identity laws

$$P \wedge T = P$$

$$P \vee F = P$$

Domination Laws

$$P \wedge F = F$$

$$P \vee T = T$$

Idempotent Laws

$$P \wedge F = P$$

$$P \wedge P = P$$

Double Negation Law

$$\neg(\neg P) = P$$

Commutative Laws

$$P \wedge q = q \wedge P$$

$$P \vee q = q \vee P$$

Associative Laws

$$(P \wedge q) \wedge r = P \wedge (q \wedge r)$$

$$(P \vee q) \vee r = P \vee (q \vee r)$$

Distributive Laws

$$P \wedge (q \vee r) = (P \wedge q) \vee (P \wedge r)$$

$$P \vee (q \wedge r) = (P \vee q) \wedge (P \vee r)$$

De Morgan's Laws

$$\neg(P \wedge q) = \neg P \vee \neg q$$

$$\neg(P \vee q) = \neg P \wedge \neg q$$

De Morgan's Laws

Absorption Laws

$$P \wedge (P \vee q) = P$$

$$P \vee (P \wedge q) = P$$

Negation Laws

$$P \wedge \neg P = F$$

$$P \vee \neg P = T$$

consistency of premises:

consistency:-

A set of formulas H_1, H_2, \dots, H_m is said to be consistent if their conjunction has the truth value T for some assignment of the truth values to be atomic appearing in H_1, H_2, \dots, H_m .

inconsistency

If for every assignment of the truth values to the atomic variables, at least one of the formulas H_1, H_2, \dots, H_m is false, so that their conjunction is identically false, then the formulas H_1, H_2, \dots, H_m are called inconsistent.

A set of formulas H_1, H_2, \dots, H_m is inconsistent, if their conjunction implies a contradiction that is $H_1 \wedge H_2 \wedge \dots \wedge H_m \Rightarrow R \wedge \neg R$

Indirect method of proof

In order to show that a conclusion C follows logically from the premises H_1, H_2, \dots, H_m , we assume that C is false and consider $\neg C$ as an additional premise. If the new set of premises is inconsistent, so that they imply a contradiction, then the assumption that $\neg C$ is true does not hold simultaneously with $H_1 \wedge H_2 \wedge \dots \wedge H_m$ being true. Therefore, C is true whenever $H_1 \wedge H_2 \wedge \dots \wedge H_m$ is true. Thus, C follows logically from the premises H_1, H_2, \dots, H_m .

For example :- show that $\neg(P \wedge \theta)$ follows from $\neg P \wedge \neg \theta$.

solution.

{1} (1) $\neg(P \wedge \theta)$ P assumed premise

{1} (2) $P \wedge \theta$ T, (1) and EI

{1} (3) P T, (2) and II

{1} (4) $\neg P$ P

{2} (5) $\neg \theta$ T, (1) and II

{3} (6) $\neg P \wedge \neg \theta$ T, (3), (5) and Iq

Example 9 show that the following premises are inconsistent.

1. If jack misses many classes through illness, then he fails high school.
2. If jack fails high school, then he is uneducated
3. If jack reads a lot of books, then he is not uneducated.
4. Jack misses many classes through illness and reads a lot of books.

solution.

P: Jack misses many classes.

Q: Jack fails high school.

R: Jack reads a lot of books

S: Jack is uneducated,

The premises are $P \rightarrow Q$, $Q \rightarrow S$, $R \rightarrow \neg S$ and $P \wedge R$

{1} (1) $P \rightarrow Q$ P

{2} (2) $Q \rightarrow S$ P

{1,2} (3) $P \rightarrow S$ T, (1), (2) and 113

{4} (4) $R \rightarrow \neg S$ P

{4} (5) $S \rightarrow \neg R$ T, (4), and E18

{1,2,4} (6) $P \rightarrow \neg R$ T, (3), (5) and 113

{1,2,4} (7) $\neg P \vee \neg R$ T, (6) and E16.

{1,2,4} (8) $\neg(P \wedge R)$ T, (7) and E8

{9} (9) $P \wedge R$ P

{1,2,4,9} (10) $(P \wedge R) \wedge \neg(P \wedge R)$ T, (8), (9) and 19.

The rules above can be summed up in the following table. The "Tautology" column shows how to interpret the notation of a given rule.

proof of contradiction:

The "proof by contradiction" is also known as *reductio ad absurdum*, which is probably Latin for "reduce it to something absurd"

Here's the idea:

1. Assumes that a given proposition is untrue
2. Based on that assumption reach two conclusion that contradict each other,

This is based on a classical formal logic construction known as *Modus Tollens*: If p implies q and q is false, then p is false. In this case, q is proposition of the form (R and not R) which is always false. p is the negation of the fact that we are trying to prove and if the negation is not true then the original proposition must have been true. If computers are not "not stupid" then they are stupid.

Example:-

Let's prove that there is no largest prime number. Prime numbers are integers with no exact integer divisors except 1 and themselves.

1. To prove . "There is no largest prime number" by contradiction.
2. Assume: There is a largest prime number, call it p
3. Consider the number N that is one larger than the product of all of the primes smaller than or equal to p $N = 1 * 2 + 3 * 5 * 7 * 11 \dots * p + 1$. is it prime?

4. N is at least as big as $p+1$ and so is larger than p and so, by step 2, cannot be prime.

5. On the other hand, N has no prime factors between 1 and p because they would all leave a remainder of 1. It has no prime factors larger than p because step 2 says that there are no primes larger than p . So N has no prime factors and therefore must itself be prime.

We have reached a contradiction (N is not prime by step 4, and N is prime by step 5) and therefore our original assumption that there is a largest prime must be false.

Automatic Theorem proving:-

Automatic theorem proving (ATP) deals with the development of computer programs that show that some statement is a logical consequence of a set of statements. ATP systems are used in a wide variety of domains. For example, a mathematician might prove the conjecture the groups of order two are commutative, from the axioms of group; from those axioms prove that organizational death rates decrease with age; a hardware developer might validate the design of a circuit by proving a conjecture that describes a circuit's performance, given axioms that describe the circuit itself, or a frustrated teenager might formulate the jumbled faces of a Rubik's cube as a conjecture and prove

21

from axioms that describe legal changes to the cube's configuration, that the rub can be rearranged to the solution state. All of these are tasks that can be performed by an ATP system, given an appropriate formulation of the problem on axioms, hypotheses and a conjecture.

The language in which the conjecture, hypotheses, and axioms are written is a logic, often classical 1st order logic, but possibly a non-classical logic and possibly a higher order logic. These languages allow a precise formal statement of the necessary information, which can then be manipulated by an ATP system. This formality is the underlying strength of ATP: There is no ambiguity in the statement of the problem, as is often the case when using a natural language such as English. Users have to describe the problem at hand precisely and accurately, and this process in itself can lead to a clearer understanding of the problem domain. This is turn allows the user formulate their problem appropriately for submission to an ATP system.

The proofs produced by ATP systems describe how and why the conjecture follows from the axioms and hypotheses, in a manner that can be understood and agreed upon by everyone, even other computer programs. The proof output may not only be a convincing argument that the conjecture is a logical consequence of the axioms and hypotheses, it often also describes a process that may be implemented to solve problems. For example, in the Rubik's cube example mentioned above, the proof would describe the sequence of moves that need to be made in order to solve the puzzle.

ATP systems are enormously powerful computer programs, capable of solving immensely difficult problems. Because of this extreme capability, their application and operation sometimes needs to be guided by an expert in the domain of application, in order to solve problems in a reasonable amount of time. Thus ATP systems, despite the name, are often used by domain experts ~~by the system~~, or at a much higher level where the user determines intermediate lemmas to be proved on the way to the proof of a conjecture. There is often a synergistic relationship between ATP system users and the systems themselves.

- The system needs a precise description of the problem written in some logical form.
- the user is forced to think carefully about the problem in order to produce an appropriate formulation and hence acquires a deeper understanding of the problem.
- The system attempts to solve the problem.
- if successful the proof is a useful output,
- if unsuccessful the user can provide guidance, or try to prove some intermediate result, or examine the formulae to ensure that the problem is correctly described.
- and so the process iterates.

ATP is thus a technology very suited to situations where a clear thinking domain expert can interact with a powerful tool, to solve interesting and deep problems. potential ATP users need not be concerned that they need to write an ATP system themselves; there are many ATP systems readily available for use.