3. **Reasoning:** It means to infer facts from existing data.

4.1 PROPOSITIONAL LOGIC AND ITS RESOLUTION

A logic is defined as a formal system in which the formulae or sentences have true or false values. In 1976, Robert Kowalski came up with an equation

Algorithm = Logic + Control

The **logic component** specifies the knowledge to be used in solving problems.

The **control component** determines the problem solving strategies by means of which that knowledge is used.

Please note that the logic component determines the meaning of the algorithm whereas the control component only affects its efficiency. In reasoning about truth values, we use a number of operators like

Operator	Meaning
$\wedge$	and
$\vee$	or
$\neg$	not
$\rightarrow$	implies
$\leftrightarrow$	iff (if and only if)

It is useful to represent these logical operators using truth table showing the possible values that can be generated by applying an operator to truth tables.

Let us now see various truth tables (for these operators).

1. **Not ($\neg$) operator:** It is a **unary operator** i.e. it is applied only to one variable

   Its truth table is as follows —

A	$\neg A$
true	false
false	true

2. **and ($\wedge$) operator**: It is a **binary operator,** i.e. it acts on two variables. It is also called as **conjunctive operator.** Its truth table is as follows:

A	B	$A \wedge B$
false	false	false
false	true	false
true	false	false
true	true	true

NOTE: that $A \wedge B$ is true if both A and B are true else it is False. A and B can be any statement or proposition.

3. **or ($\vee$) Operator:** It is again a **binary operator** and is also known as **disjunctive operator.**

   Its truth table is a follows –

A	B	$A \vee B$
false	false	false
false	true	true
true	false	true
true	true	true

So, $A \vee B$ is true for all cases except for $A = B =$ false. This table represents an **inclusive or operator**. An exclusive or would have been 'false' in the final row.

4. **Implies ($\rightarrow$) Operator:** Its truth table is as follows –

A	B	$A \rightarrow B$
false	false	true
false	true	true
true	false	false
true	true	true

This type of implication is also called as **material implication**. Also note that in statement,

$A \rightarrow B$

'A' is known as **antecedent.**

and 'B' is known as **consequent.**

$A \rightarrow B$ is real as 'A implies B' or 'if A then B' or 'if A is to B then B is true'

We can also construct complex truth tables like for $A \vee (B \vee C)$ the truth table is as follows—

A	B	C	$A \wedge (B \vee C)$
false	false	false	false
false	false	true	false
false	true	false	false
false	true	true	false
true	false	false	false
true	false	true	true
true	true	false	true
true	true	true	true

for $n = 3$ (i.e., 3 variables A, B and C) these are 8 outputs. **In general, for n inputs there will be 2^n outputs. Please note that the use of brackets is very important because $A \wedge (B \vee C) \neq (A \wedge B) \vee C$. Also note that to avoid ambiguity, these logical operators are assigned pre**

The order of precedence is as follows –

Highest precedence

$\neg$

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

Lowest precedenc

This means that $\neg A \vee B = (\neg A) \vee B$

Tip: Use braces whenever an expression becomes ambiguous.

5. **Tautology:** Consider the following truth table –

A	$AV \neg A$
false	true
true	true

This truth table has a property, that is, the value of expression is always true irrespective of the value of A. **Such an expression that is always true is known as tautology.** If A is a tautology, we write —

$$\models A$$

A logical expression that is tautology is often described as being valid. A **Valid expression** is defined as being one that is true under any interpretation. In other words, no matter what meanings and values we assign to variables in a valid expression, it will still be true. If an expression is false in any interpretation, it is described as being contradictory. The following expressions are contradictory:

(*a*) $A \wedge \neg A$

(*b*) $(A \vee \neg A) \rightarrow (A \wedge \neg A)$

It does not matter what A means in these expressions. The result cannot be time.

Some expressions are **satisfactory** but **not valid.** This means that they are true under some interpretation but not under all interpretations. The following expressions are satisfactory —

(*a*) $A \vee A$

(*b*) $(A \wedge B \vee \neg C) \rightarrow (D \wedge E)$

Please note that a contradictory expression is clearly not satisfactory and so is described as being unsatisfactory.

Tip: Logic is studied as knowledge representation language in A.I.

Logics are of different types —

(*a*) Propositional logic.

(*b*) Predicate logic.

(*c*) Temporal logic.

(*d*) Modal logic.

(*e*) Description logic. etc.

Out of these propositional logic and predicate logic are fundamental to all logic.

Propositional logic

Propositions are statements used in mathematics. A proposition or sentence is classified as declarative sentence whose value is either 'true' or 'false'.

For example. —

(*a*) The sky is blue.

(*b*) Snow is white.

(*c*) $4 * 4 = 16$

Propositions are of two types —

(*a*) Atomic propositions.

(*b*) Molecular propositions.

Atomic propositions are single propositions. The above given examples are all propositions.

Molecular propositions are formed by combining two or more atomic propositions.

For example. —

"Lata Mangeshkar is a singer and Rajiv Chopra is an author".

Propositions are 'sentences' either true or false but not both. A sentence is the smallest unit in propositional logic. If proposition is 'time', then truth value is 'time'. If proposition is 'false' then truth value is 'false'.

> **NOTE:** Propositional logic is also called as Propositional Calculus or Sentential Calculus or Boolean Algebra.

Please understand that propositional logic tells the ways of joining and/or modifying entire propositions, statements or sentences to form more complicated propositions, statement or sentences as well as the logical relationships and properties that are derived from the methods of combining or altering statements. Also note that the process used for performing manipulation of the symbols is according to some rules and laws.

A BNF (Backus Naur Form) grammar of Sentences in Propositional logic is as follows —

Sentence → Atomic sentence |complex sentence|

Atomic sentence → true |false| symbol

Symbol → $P|Q|R\ldots$

Complex sentence → ¬ sentence

| (Sentence ∧ Sentence)

| (Sentence □ Sentence)

| (Sentence → Sentence)

| (Sentence ↔ Sentence)

Some terminologies in Propositional Logic

1. **Statement:** A statement is a sentence that is TRUE or FALSE. These statements have some properties:–

   (*a*) **Satisfiability:** A statement is satisfiable if there is some interpretation for which it is true.

   (*b*) **Contradiction:** A sentence is contradictory (unsatisfiable) if there is no interpretation for which it is true. **For example,** Japan is capital of India.

   (*c*) **Validity:** A sentence is valid if it is true for every interpretation.

   For example, the sentence $P \vee \neg P$ is valid. Valid sentences are also called as tautologies.

   (*d*) **Logical Equivalence:** Two sentences are logically equivalent if they have the same truth table under every interpretation, written as $P \leftrightarrow Q$.

   For example, $P \wedge Q$ and $Q \wedge P$ are logically equivalent and this can be shown using truth tables given earlier.

2. **Connective or operator:** The connectives join simple statements into com... and joins compound into larger compound statements. ...nents

For example : 6 basic connectives and their symbols

Connectives	Symbols	Meaning
Assertion	P	'P is true'
Negation	$\neg P$, $\sim$, !, not	'P is false'
Conjunction	$p \wedge g$, AND, &	'Both p and q are true'
Disjunction	$p \vee q$, i, OR	'Either p is true or q is true or both'
Implication	$p \rightarrow q$, $\supset$, $\Rightarrow$, if then	'if p is true from q is also true' (i.e., p implies q)
Equivalence	$\leftrightarrow$, $\equiv$, $\Leftrightarrow$, if and only if	'P and q are either both true or both false'

NOTE: Both propositions and connective are the basic elements of propositional logic.

3. **Truth value:** The truth value of a statement is its TRUE or False value.

For example

(*a*) q is either TRUE or FALSE.

(*b*) $\sim q$ is either TRUE or FALSE.

(*c*) $q \rightarrow p$ is either TRUE or FALSE.

NOTE: Use 'T' or '1' to mean TRUE & 'F' or '0' to mean FALSE

For example.

p	q	$\neg p$	$\neg q$	$p \vee q$	$p \rightarrow q$
T	T	F	F	T	T
T	F	F	T	T	F
F	T	T	F	T	T
F	F	T	T	F	T

4. **Tautologies:** A proposition that is always true is called as a tautology.

For example: $(p \vee \neg p)$ is always true irrespective of the value of p.

5. **Contradictions:** A proposition that is always false is called as a contradiction.

For example: $(q \vee \neg g)$ is always false regardless of the truth value of the proposition.

6. **Contigencies:** A proposition is called as a contigency, if that proposition is neither a tautology nor a contradiction.

For example : $(p \vee q)$ is a contingency

7. **Antecedent and consequent:** In the conditional statements, $p \rightarrow q$, 'p' is the 'if-clause' and is called as **antecedent** whereas 'a' is the 'then clause' and is known as **consequent.**

8. **Argument:** Any argument can be expressed as a **compound statement. If we take all premises, conjoin them and make that conjunction the antecedent of a conditional and make the conclusion the consequent then this implication statement is called as the corresponding conditional of the argument.**

Please remember the following points here:

1. Every **argument** has a **corresponding condition.**
2. Every **implication statement** has a **corresponding argument.**

3. Because the corresponding condition of an argument is a statement, it is therefore either a **tautology or a contradiction or a contingency.**
4. An argument is **valid** "if and only if" its corresponding conditional is a **tautology.**
5. Two statements are **consistent** 'if and only if' their **conjunction is not a contradiction.**
6. Two statements are **logically equivalent** "if and only if" their truth table columns are identical or "if and only if" the statement of their equivalence using "≡" is a **tautology.**

NOTE: Truth tables are used to test validity tautology, contradiction, contingency. consistency and logical equivalence.

A logical system is defined in terms of its syntax, its semantics and a set of rules of deduction for proving.

An expression is referred to as a well formed formula (WFF) if it is constructed correctly according to the rules of the syntax of propositional calculus. We define a sentence recursively in terms of other sentences.

More formally **a WFF consists of atomic symbols joined with connectives.**

$\therefore P, P \wedge \neg P, P \wedge Q, P \vee Q, P \rightarrow Q, P \leftrightarrow Q$ are WFFs.

A WFF is defined recursively as —

1. An atom (say P) is a WFF.
2. If P is a WFF then $\neg p$ *is a WFF.*
3. If P and Q are WFF, then $(P \vee Q)$, $(P \wedge Q)$, $(P \rightarrow Q)$ and $(P \leftrightarrow Q)$ are WFF.
4. A string of symbols is a WFF if and only if it is obtained by using above rule 1 to rule 3

Also note the following points about WFF:

1. A WFF is not a proposition but if we substitute a proposition in place of a propositional variable, we get a proposition.

   For example : $\neg (P \vee Q) \wedge (Q \wedge R) \rightarrow (\neg Q \wedge R)$
2. The WFF is also simply called as a formula.
3. WFF are conveniently represented as **truth tables.**
4. A WFF for all true entries is called as a tautology.
5. Two WFFs, say α and β in propositional variables $p_1, p_2 \cdots p_n$ are equivalent if the formula $\alpha \rightarrow \beta$ is a tautology then, the equivalent statements are represented as $\alpha \equiv \beta$.
6. **Please understand that $\alpha \leftrightarrow \beta$ and $\alpha \equiv \beta$ are different as $\alpha \leftrightarrow \beta$ is a formula** whereas $\alpha \equiv \beta$ is not a formula but a relation between α and β.

The semantics (meaning) of the operators of propositional calculus can be defined in terms of truth tables. If our knowledge is represented in form of propositional logic then it is very easy for a computer to do reasoning on this knowledge. The set of inference rules and laws that are used for reasoning are given below:

Rule 1: Idempotency Rule

(*a*) $P \vee P = P$

(*b*) $P \wedge P = P$

Rule 2: Commutative Law

(a) $P \vee Q = Q \vee P$

(b) $P \wedge Q = Q \wedge P$

(c) $P \leftrightarrow Q = Q \leftrightarrow P$

Rule 3: Associative Law

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

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

Rule 4: Distributive Law

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

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

Rule 5: De Morgan's rule

(a) $\sim (P \vee Q) = \sim P \wedge \sim Q$

(b) $\sim (P \wedge Q) = \sim P \vee \sim Q$

Rule 6: Implication Removal

$P \leftrightarrow Q = \sim P \vee Q$

Rule 7: Biconditional elimination

$P \leftrightarrow Q = (P \rightarrow Q) \wedge (Q \rightarrow P)$

Rule 8: Absorption Law

(a) $P \vee (P \wedge Q) \equiv P$

(b) $P \wedge (P \vee Q) \equiv P$

Rule 9: Contrapositive

$P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$

Rule 10: Double negation

$P \equiv \neg (\neg P)$

Rule 11: Fundamental identities

(a) $P \vee \neg P \equiv T$

(b) $P \wedge \neg P \equiv F$

(c) $P \vee \neg T \equiv T$

(d) $P \vee T \equiv P$

(e) $P \vee F \equiv P$

(f) $P \vee F \equiv F$

(g) $(F \neg Q) \wedge (P \Rightarrow \neg Q) = \neg P$

(h) $P \Rightarrow Q \equiv (\neg P \vee Q)$

Rule 12: AND ($\wedge$) rule (Introduction)

Given P and Q. We deduce $P \wedge Q$ i.e.

$$\frac{P \quad Q}{P \wedge Q}$$

This follows from the definition of $\wedge$.

Rule 13: AND ($\wedge$) rule (Elimination)

Given P and Q. We deduce P and Q i.e. separately i.e.

(*a*) $P \wedge Q \rightarrow P$

(*b*) $P \wedge Q \rightarrow Q$

This also follows from the definition of $\wedge$.

Rule 14: OR ($\vee$) rule (Introduction)

Given P and Q, We deduce P and Q i.e. separately i.e.

(*a*) $A \rightarrow A \vee B$

(*b*) $B \rightarrow A \vee B$

i.e. from A we can deduce the disjunction of A which any expression.

Rule 15: Modus Ponens

From given two statements P and $P \rightarrow Q$, infer Q i.e.,

$$\frac{P \quad P \rightarrow Q}{\therefore \; Q}$$

NOTE: In implication form, this rule is written as

$$(P \wedge (P \rightarrow Q)) \rightarrow Q$$

For example.

Given: Rajiv is intelligent

and: Rajiv is intelligent $\rightarrow$ Rajiv is author.

Conclude: Rajiv is author.

Rule 16: Modus Tollens

From the given two statements, $\neg Q$ and $(P \rightarrow Q)$, infer $\neg P$ i.e.

$$\frac{\neg Q \quad P \rightarrow Q}{\therefore \; \neg P}$$

For example :

Given: Ajay is not a religious person.

and: Ajay goes to temple daily implies Ajay is a religious person

Conclude: Ajay does not go to temple daily.

NOTE: In implication form, this rule is written as

$$(\neg Q \wedge (P \rightarrow Q)) \rightarrow \neg P$$

Rule 17: Chain Rule/Hypothetical syllogism

From $(P \rightarrow Q)$ and $Q \rightarrow R)$, infer $(P \rightarrow R)$ i.e.

$$\frac{P \rightarrow Q \quad P \rightarrow Q}{\therefore \; (P \rightarrow R)}$$

For example :

Given: It is raining → must take umbrella

and: Must take umbrella → There is no sun

Conclude: There is no sun.

NOTE: In implication form, this rule is written as

$((P \rightarrow Q) \wedge (Q \rightarrow R)) \rightarrow (P \rightarrow R)$

Rule 18: Reductio Ad Absurdum

This rule states that if we assume that A is false ($\neg A$) and this leads to a contradiction ($\perp$) then we say that A is true. This is known as proof by contradiction. This is

$$\begin{array}{l} \neg A \\ \vdots \\ \perp \\ \hline \therefore A \end{array}$$

Here, the symbol $\perp$ is called as **falsumn** which is used to indicate an absurdity or a contradiction.

For example., $\perp$ can be deduced from $A \wedge \neg A$.

Rule 19: Disjunctive syllogism

From two given sentences, $\neg P$ and $(P \vee Q)$, infer Q i.e.,

$$\begin{array}{l} \neg P \\ P \vee Q \\ \hline \therefore Q \end{array}$$

NOTE: In implication form, this rule is written as

$(\neg P \wedge (P \vee Q)) \rightarrow Q$

Rule 20: Constructive and Destructive dilemma

From given two sentences

$$\begin{array}{l} (P \rightarrow Q) \wedge (R \rightarrow S) \\ \text{and} \qquad (P \vee R) \\ \hline \therefore (Q \vee S) \end{array}$$

(Constructive dilemma)

Also,

$$\begin{array}{l} (P \rightarrow Q) \wedge (R \rightarrow S) \\ \neg (P \vee R) \\ \hline \therefore (Q \vee S) \end{array}$$

(Destructive dilemma)

NOTE: In implication form, this rule is written as

$((P \rightarrow Q) \wedge (R \rightarrow S))\ (\neg Q \wedge S) \rightarrow (P \vee R)$.

We are in a position to solve some examples now.

EXAMPLE 1: Given P and Q, prove that

$((P \to Q) \to P) \to P$ **is tautologous.**

SOLUTION: We draw its truth table —

P	Q	$((P \to Q) \to P) \to P$		
T	T	T	T	T
T	F	F	T	T
F	T	T	F	T
F	F	F	F	T

Hence, $((P \to Q) \to P) \to P$ is tautologous.

EXAMPLE 2: Find the equivalent expression of

A **(&** $(A \vee B)$**) using truth table**

SOLUTION:

A	B	$(A \vee B) = C$ (say)	$A \,\&\, C$
t	t	t	t
f	f	f	f
t	f	t	t
f	t	t	f

$\therefore \quad A\,(\&\,(A \vee B)) = A$ $\qquad [\because AB' + AB = A\,(B' + B) = A]$

EXAMPLE 3: Prove that $(A \wedge B) \to A \vee B$**.**

SOLUTION: Given $A \wedge B$ (assumption)

$\to A$ (by $\wedge$ elimination)

$\to A \vee B$ (by $\vee$ introduction)

Hence Proved.

EXAMPLE 4: Prove that —

$((P \wedge Q) \to R) \vee (\sim Q \to \sim R)$ **is valid.**

SOLUTION: Let us denote this statement by Q

Q: $((P \wedge Q) \to R) \vee (\sim Q \to \sim R)$

We draw its truth table now:

P	Q	R	$P \wedge Q$	$P \wedge Q \to R$	$\sim Q \to \sim R$	Q
T	T	T	T	T	T	T
T	T	F	T	F	T	T
T	F	T	F	T	F	T
T	F	F	F	T	T	T
F	T	T	F	T	T	T
F	T	F	F	T	T	T
F	F	T	F	T	F	T
F	F	F	F	T	T	

$\because$ Q is true under all interpretations.

$\therefore$ It is a valid statement.

EXAMPLE 5: Show that the set of statements "I will be wet if it rains and I go out of the house. It is raining now. I go out of the house. I will not be wet" are inconsistent.

SOLUTION: W : I will be wet.

R : It rains.

G : I go out of the house.

$\therefore$ A set of formulae corresponding to given word problem is —

$\{(R \wedge G) \rightarrow W, R, \sim W\}$

It can be shown using truth table that

$((R \wedge G) \rightarrow W) \wedge R \wedge G \wedge \sim W$ is false under all interpretations and hence is inconsistent.

EXAMPLE 6: Determine whether each of following is

(a) Satisfiable. (b) Contradictory. (c) Valid.

$S_1 : (P \,\&\, Q) \vee \sim (P \,\&\, Q)$

$S_2 : (P \,\&\, Q) \rightarrow (R \vee \sim Q)$

$S_3 : (P \,\&\, Q) \rightarrow (R \vee \sim Q)$

$S_4 : (P \vee Q) \,\&\, (P \vee \sim Q) \vee P$?

SOLUTION: $S_1 : (P \,\&\, Q) \vee \sim (P \,\&\, Q)$

Its truth table is —

~~G : I go out of the house.~~

P	Q	$P \& Q = A$	$\sim(P \& Q) = B$	$A \vee B$
t	f	f	t	t
f	t	f	t	t
f	f	f	t	t
t	t	t	f	t

$\therefore$ It is satisfiable.

$\therefore$ It is not contradictory

$\therefore$ It is valid as sentence is true for every interpretation.

Consider $S_2 : (P \,\&\, Q) \rightarrow (R \vee \sim Q)$

P	Q	R	$P \& Q = X$	$\sim Q$	$R \vee \sim Q = Y$	$X \rightarrow Y$
f	f	f	f	t	t	f
f	f	t	f	t	t	f
f	t	f	f	f	f	t
f	t	t	f	f	t	f
t	f	f	f	t	t	f
t	f	t	f	t	t	f
t	t	t	t	f	t	t

$\therefore$ It is satisfiable t t f

It is not contradictory

It is not valid.

$S_3 : (P \& Q) \rightarrow (R \vee \sim Q)$

Its truth table is —

P	Q	R	$P \& Q = X$	$\sim Q$	$R \vee \sim Q = Y$	$X \rightarrow Y$
f	f	f	f	t	t	f
f	f	t	f	t	t	f
f	t	f	f	f	f	t
f	t	t	f	f	t	f
t	f	f	f	t	t	f
t	f	t	f	t	t	f
t	t	f	t	f	f	f
t	t	t	t	f	t	t

∴ It is satisfiable

It is not contradictory

It is not valid.

$S_4 : (P \vee Q) \& (P \vee \sim Q) \vee P$

P	Q	$P \vee Q = X$	$\sim Q$	$P \vee \sim Q = Y$	$Y \vee P = Z$	$X \& Z$
t	t	t	f	t	t	t
f	t	t	f	f	f	f
t	f	t	t	t	t	t
f	f	f	t	t	t	t

∴ It is satisfiable

It is not contradictory

It is not valid.

Normal Forms in Propositional Logic

There are two major normal forms of statements in propositional logic. They are Conjunctive Normal Form (CNF) and Disjunctive Normal Form (DNF).

Any formula can be converted into its normal form by substituting it by its equivalent formulae. Two formulas P and Q are equivalent if and only if the truth values of P and Q are same for all values of P and Q. We have already discussed some widely used equivalent formulae in table III.

A formula P is said to be in CNF if it is of the form

$$P = P_1 \wedge P_2 \wedge P_3, \ldots, \wedge P_n; n \geq 1$$

where each $P_1, P_2, P_3, \ldots, P_n$ is a disjunction of an atom or negation of an atom.

A formula P is said to be in DNF if it has the form

$$P = P_1 \vee P_2 \vee P_3, \ldots, \vee P_n; n \geq 1$$

where each $P_1, P_2, P_3, \ldots, P_n$ is a conjunction of an atom or negation of an atom.

Conversion Procedure to Normal Form

Step 1: Eliminate implication and biconditionals. We use the following laws

$$(P \rightarrow Q) = \neg P \vee Q$$

$$(P \leftrightarrow Q) = (P \rightarrow Q) \wedge (Q \rightarrow P)$$

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

Step 2: Reduce the NOT symbol by the formula $(\neg(\neg P)) = P$ and apply De Morgan's theorem to bring negations before the atoms.

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

Step 3: Use Distributive laws and other equivalent formula given in table III to obtain the normal form

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

Conjunctive Normal Form

The resolution rule applies only to disjunction of literals. So it would seem to be relevant only for knowledge bases and queries consisting of such disjunctions. How can then it lead to a complete inference procedure for all of propositional logic? **The answer is that every sentence of propositional logic is logically equivalent to a conjunction of disjunctions of literals. A sentence expressed as a conjunction of disjunctions of literals is said to be in a conjunctive normal form (CNF).** Every sentence can be transformed into a CNF sentence using the following steps making use of table 6.2

1. Eliminate $\leftrightarrow$ replacing $P \leftrightarrow Q$ with $(P \rightarrow Q) \wedge (Q \rightarrow P)$
2. Eliminate $\rightarrow$ replacing $P \rightarrow Q$ with $\neg P \vee Q$.
3. CNF requires $\neg$ should appear only in literals, so we move $\neg$ inwards by repreated application of following equivalences to Table 6.2

$$\neg(\neg P) \equiv \text{Double-negation elimination}$$
$$(P \wedge Q) \equiv (\neg P \wedge \neg Q) \text{ de Morgan}$$
$$\neg(P \vee Q) \equiv (\neg P \vee \neg Q) \text{ de Morgan}$$

4. We now apply distributivity law, distributive $\vee$ over $\wedge$. whenever the sentence contains nested $\vee$ over $\wedge$ operators applied to literals.

EXAMPLE 1: Convert $((P \rightarrow Q) \rightarrow R)$ into CNF

SOLUTION: $((P \rightarrow Q) \rightarrow R)$

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

Hence, $(P \vee R) \wedge (\neg Q \vee R)$ is the CNF of $((P \rightarrow Q) \rightarrow R)$

We are in a position to solve some examples now.

EXAMPLE 2: Convert $(P \rightarrow Q) \rightarrow R$ to CNF?

SOLUTION: Given: $(P \rightarrow Q) \rightarrow R$

$$= (\sim P \vee Q) \rightarrow R$$
$$= \sim(\sim P \vee Q) \vee R \qquad [\because P \rightarrow Q = \sim P \vee Q]$$
$$= (P \,\&\, \sim Q) \vee R \qquad \text{[De Morgan's law]}$$
$$= (P \vee R) \,\&\, (\sim Q \vee R) \qquad \text{[By distributive law]}$$

Example 3: Convert $\sim(P \& Q) \& (P \vee Q)$ to DNF?

Solution: $(\sim P \vee \sim Q) \& (P \vee Q)$ [De Morgan's law]

$= ((\sim P \vee \sim Q) \& P) \vee ((\sim P \vee \sim Q) \& Q)$ [By distributive law]

$= (\sim P \& P) \vee (\sim Q \& P) \vee (\sim P \& Q) \vee (\sim Q \& Q)$

Example 4: Convert $\sim(P \vee \sim Q) \& (R \to S)$ to DNF?

Solution: $(\sim P \& Q) \& (\sim R \vee S)$ [De Morgan's law]

$= \sim(\sim((\sim P \& Q) \& (\sim R \vee S)))$ [Negative 2 times]

$= \sim((P \vee \sim Q) \vee (R \& \sim S)$

Example 5: Convert $P \to ((Q \& R) \leftrightarrow S)$ to DNF?

Solution: $F \to G = \sim F \vee G$

$\therefore$ We have —

$= (\sim P) \vee ((Q \& R) \to S$

$= (\sim P) \vee (\sim (Q \& R) \vee S) \& (\sim S \vee (Q \& R))$ [$\because P \leftrightarrow Q = (\sim P \vee Q) \& (\sim Q \vee P)$]

$= (\sim P) \vee (((\sim Q \vee \sim R) \vee S) \& ((\sim S \vee Q) \& (\sim S \vee R)))$ [Distributive law?]

$= (\sim P) \vee ((\sim Q \vee \sim R) \vee S) \& ((\sim S \vee Q) \& (\sim S \vee R)))$

Example 6: Convert $(\sim P \& Q) \vee (P \& \sim Q) \& S$ to CNF?

Solution: $(\sim P \& Q) \vee P \& (\sim Q \& S)$

$= ((\sim P \vee P) \& (Q \vee P) \& (\sim Q \& S))$ (Associativity)

[Distributive law]

Resolution in Propositional Logic [By Robinson]

Resolution is the process of producing proofs by refutation or contradiction. It is a procedure used for **theorem proving.** It is the rule of **inference. The procedure of resolution is as follows:**

1. Assume that the negation of the theorem which we want to prove is time.
2. Show that the axioms and the assumed negation of the theorem together cannot be true.
3. Conclude that the assumed negation of the theorem cannot be true because it leads to contradiction.
4. Conclude that the theorem must be true as the assumed negation of the theorem is not true.

Resolution is applicable only on the facts represented in clausal form. A clause is a special formula expressed as disjunction of literals. If a clause contains only one literal then it is called as a **visit clause. Please note that CNF representation of a formula is of the form** $(C_1 \wedge C_2 \wedge \ldots \wedge C_n)$ **where each** $C_k (1 \le k \le n)$ **is a clause**

If two clauses C_1 and C_2 contain a complementary pair of literals $\{L, \sim L\}$ then th
may be resolved together by deleting L from C_1 and $\sim L$ from C_2 and constructing a ne
the disjunction of the remaining literals in C_1 and C_2. The new clause th erate
resolvent of C_1 and C_2. Here, C_1 and C_2 are called **parents** of resolv

contain more than one set of complementary pair of literals. All complementary pair of literals are removed and resolvent is constructed, by the disjunction of the remaining literals in the parents.

For example : Consider two clauses

C_1 with literal $L1$ and $\sim L2$

&

C_2 with literal $\sim L1$ and $L2$

i.e., $C_1 = L1 \vee \sim L2 \vee C_1'$

&

$C_2 = \sim L1 \vee \sim L2 \vee C_2'$

where C_1' and C_2' are disjunction of remaining literals in C_1 and C_2 respectively

After resolving C_1 and C_2 we get

An inverted binary tree is generated with the last node of the binary tree to be a **resolvent. This is known as a Resolution tree. Resolved literals are underlined for more clarity.**

For example : Consider two clauses as

$$C_1 = P \vee Q \vee \sim R$$

&

$$C_2 = \sim Q \vee W$$

$\therefore$ $P \vee \underline{Q} \vee \sim R$ $\qquad$ $\underline{\sim Q} \vee W$

$$P \vee \sim R \vee W$$

$\therefore$ Resolvent $(P \vee Q \vee \sim R, \sim Q \vee W) = P \vee \sim R \vee \underline{W}$

Now, consider another example with three clauses —

$$C_1 = P \vee Q \vee R$$

$$C_2 = \sim Q \vee W$$

and

$$C_3 = \sim P \vee \sim W$$

$\therefore$ $P \vee \underline{Q} \vee R$. $\qquad$ $\sim \underline{Q} \vee W$.

$\underline{P} \vee \sim R \vee \underline{W}$ $\qquad$ $\sim \underline{P} \vee \sim \underline{W}$.

R

∴ Resolvent $R = (C_1, C_2, C_3)$

Please note that the order of clauses taken for resolution is immaterial, we have to identify two clauses with complementary literals and resolve them. Take the current resolvent and try to find another clause yet not been resolved having complementary literal proceed till all the clauses have been resolved. *Also note that the resolved clauses are not used in resolution process further.*

In a similar manner, in the big system, where there are multiple clauses, all pairs of resolvable clauses are identified and resolved.

Resolution Rule: The resolution of clauses

$(A \vee B)$ and $(\neg B \vee C)$ is $A \vee C$.

Resolution works on the principle of **refutation,** i.e., it presumes that the statement to be proved is not true and then it proves that whatever we have assumed is not true and hence the original statement is true. Now let us resolve the following clause —

$\{(\neg A, B)\ (\neg A, \neg B, C), A \neg C\}$

We start by resolving the first clause with the second clause, thus eliminating B and $\neg B$ to get

$\{(\neg A, C), A, \neg C\}$

$= \{C, \neg C\}$

$= \perp$ (i.e., falsum).

∴ This resolution has resulted in a **falsum** meaning that the original clauses were inconsistent. Thus, we have refutated the original clauses using resolution.

Refutation (RR)

∴ We say that

$\{(\neg A, B), (\neg A, \neg B, C), A \neg C\} = \perp$

after *RR*.

This is how refutation helps in logical inferencing.

Let us now write an algorithm for Resolution using Propositional Logic —

Algorithm : Resolve (P, F)

Step 1: Convert all propositions of F to clausal form.

Step 2: Negate P and convert the result to clausal form. Add it to the set of clauses obtained in step 1.

Step 3: Repeat until either of the contradictions is found or no progress can be made.

(*a*) Select two clauses (called as parent clauses).

(*b*) Resolve them together to create the resolvent.

(*c*) If the resolvent is the empty clause, then a contradiction has been found.

If it is not, then add it to the set of clauses available.

Application of Resolution

Resolution cannot be used to solve **combinational search** problems b… whether the solution to such a problem exists or not.

For example.: **Consider a three coloring problem "Given a map, it is p… using three colors such that no two countries that are next t…**

We can represent this search in form of clauses and … resolution. This method tells us that if a solution exists but … **are in a position to solve some problems now.**

EXAMPLE 1: "If it is hot, then it is humid. If it is humid then it will rain. It is hot." Show that "It will rain".

SOLUTION: Let us denote these statements

H : It is humid

R : It will rain.

Q : It is hot.

∴ Formulae corresponding to given sentences are —

If it is hot, then it is humid i.e., $Q \to H \cong \sim Q \vee H$

If it is humid, then it will rain i.e., $H \to R \cong \sim H \vee R$

Construct a set, S, of clauses from above set of statements as given below —

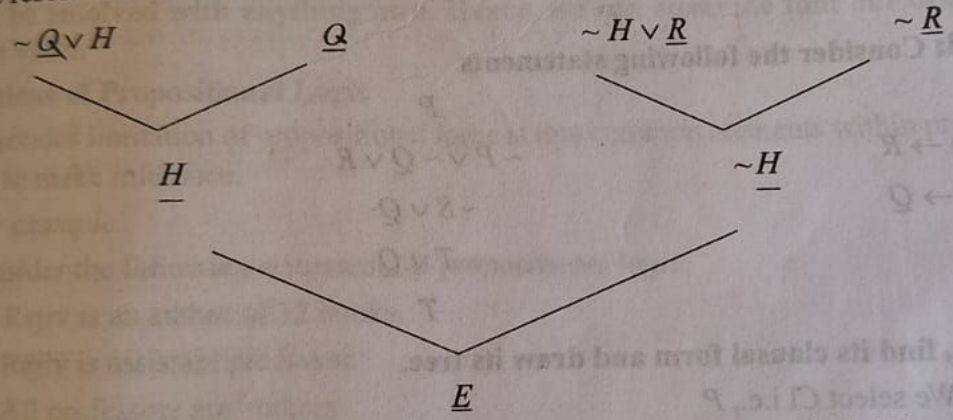
$$S = \{\sim Q \vee H, \sim H \vee R, Q\}$$

Include negation of "It will rain (say R)" to the set, S.

∴
$$S' = \{\sim Q \vee H, \sim H \vee R, Q, \sim R\}$$

Now, we can show that S' *is* unsatisfiable, i.e., from S' one can deduce empty clause, then we can infer or conclude that 'It will rain'.

∴ The resolution tree is —



∴ an empty clause (E) is deduced from a set, S', "It will rain" is concluded from S.

EXAMPLE 2: Consider the following statements —

$$A \to B$$
$$B \to C$$
$$C \to D$$
$$D \to EVF$$
$$A \to F.$$

Using resolution, find its clausal from and draw its tree.

SOLUTION: Firstly, we negate the conclusion

$A \to F$, i.e., we get $\neg (A \to F)$.

Next, $D \to E \vee F$

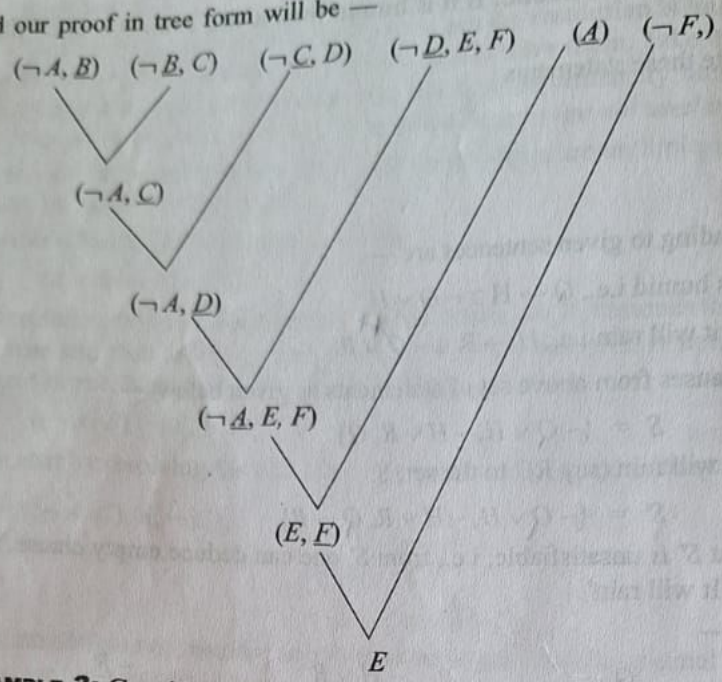
$$\equiv \neg D \vee (E \vee F)$$

& $\neg (A \to F) \equiv \neg (\neg A \vee F)$

$$\equiv A \wedge \neg F$$

∴ Our clauses are, $S = \{(\neg A, B), (\neg B, C), (\neg C, D), (\neg D, E, F), (A \neg F)\}$

and our proof in tree form will be —



EXAMPLE 3: Consider the following statements

1. P
2. $(P \wedge Q) \rightarrow R$
3. $(S \vee T) \rightarrow Q$
4. P
5. T

$$\sim P \vee \sim Q \vee R$$
$$\sim S \vee Q$$
$$\sim T \vee Q$$
$$T$$

Using resolution, find its clausal form and draw its tree.

SOLUTION: We select $C1$ i.e., P and then negate $\sim R$.

Now select ($C2$) clauses as

$\sim P \vee \sim Q \vee R$.

$\therefore$ We have two clauses as—

$C1 : \sim R$

$C2 : \sim P \vee \sim Q \vee R$

$\therefore$ Resolvent will be $\sim P \vee \sim Q$.

Here, we have proved R. We should always select those pairs that can be resolved very easily. Now, prove P. We negate $\sim P$ to P.

$C1 : \sim P \vee \sim Q$

$C2 : P$

Resolvent is $\sim Q$.

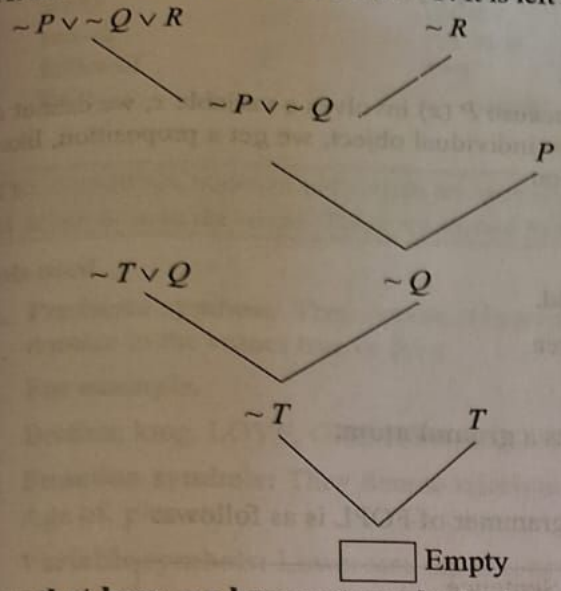
Now prove T. Select $\sim T \vee R$

$C1 : \sim Q$

$C2 : \sim T \vee Q$

$\therefore$ Resolvent is $\sim T$.

At last we have $C1$ as T and $C2$ as $\sim T$. It is left empty which is a contradiction.



Note that here, we have not reached falsum as we are left with a single clause {E} which cannot be resolved with anything now. Hence, we can conclude that our original conclusion was not valid.

Limitations of Propositional Logic

A very serious limitation of propositional logic is that common elements within propositions cannot be used to make inference.

For example.,

Consider the following statements in propositional logic.

S_1 : Rajiv is an author of 12 books.

S_2 : Rajiv is assistant professor.

S_3 : All professors are authors

Now, from S_1, S_2 and S_3 we cannot conclude that all professors are authors because this method of representation fails in capturing the relationship between any individual. Propositional logic lacks in expressiveness which is very vital for good KR. Propositional logic is too coarse to easily describe properties of objects. We cannot make generalized statements.

So, we go to predicate logic.

4.2 PREDICATE LOGIC & ITS RESOLUTION

We are interested in a form of predicate logic called as **first order predicate logic.** FOPL allows the quantified variables to refer to object domain and not to predicates or functions. If the quantification is over first order predicates and functions then it becomes **second order predicate.** Similarly, third order predicate allows quantification over predicates and function of second order and so on. A.I. researchers have found that FOPL is most suitable for KR.

Every complete sentence contains two parts — **a subject** and a **predicate.**

Subject: The subject is what (or whom) the sentence is about.

Predicate: It tells something about the subject.

For example. In the sentence, 'Rajiv is an author' subject is 'Rajiv' and predicate is 'author'.

Sentences involving the predicates that describe the property of objects are denoted by $P(x)$ where

P — The predicate,

x — is a variable denoting any object.

Here, $P(x)$ is not a proposition. This is because $P(x)$ involves a variable x, we cannot assign a truth value to $P(x)$. But if we replace x by an individual object, we get a proposition, like if we replace x by "Rajiv" in $P(x)$ we get a proposition.

Characteristics of predicate logic

1. Logical inferencing is allowed.
2. More accurate *KR* of facts of real world.
3. Program designing is its application area.
4. Better theoretical foundation.
5. A predicate with no variable is called as a **ground atom.**

Syntax of FOPL

A complete description of syntax from the grammar of FOPL is as follows:

Sentence → Atomic Sentence
| (Sentence Connective Sentence)
| (Quantifier variable … Sentence)
| (¬sentence)

Atomic Sentence → Predicate (Term…) | Term = Term

Term → Function (Term…)
| Constant
| Variable

Connective → ⇒ | ∧ | ∨ | ⇒

Quantifier → ∀ | ∃

Constant → $a \mid X_1 \mid$ John | …

Variable → $a \mid x \mid s \mid$

Predicate → Before | HasColour | Raining | …

Function → Mother | LeftLeg

NOTE: Predicate calculus symbols are irreducible syntactic elements just as we have tokens in programming language.

Rules for predicate calculus symbols —

1. Set of letters (uppercase or lowercase) is allowed.
2. Set of digits (0 to 9) is allowed.
3. Underscore (_) is allowed.

But,

1. Blanks and non-alphanumeric characters cannot be used.
2. Special characters like $, *, #, /, are not allowed.

For example.

Valid symbols	Invalid symbols
white	2black
son-of	Ab % b
father-of	**3
Rajiv	— parrot

NOTE: Parentheses, commas and periods are used to construct well formed structure but do not denote objects or relations in the world. These are known as **improper symbols.**

Symbols used

1. **Predicate symbols:** They denote relations or functional mapping from the elements of a domain to the values true or false

   For example.

   Brother, king, LOVE, GREATER, EQUAL, MARRIES etc.

2. **Function symbols:** They denote relations defined on a domain. **For example.** Father of, Age of, plus etc.

3. **Variable symbols:** Lowercase unsubscripted or subscripted letters like x, y, z, t, u, v etc.

   They can assume different values over a given domain.

4. **Constants:** They are fixed value terms. They are individual symbols which are names of objects like john, rajiv, 1, 2, 3, a, b, c etc.

5. **Quantifiers:** They are of two types —

   (a) $\exists$ (**or existential quantifier**) where ($\exists x$) means for some x or there is no x

   (b) $\forall$ (**or universal quantifier**) where ($\forall x$) means for all x.

   We will discuss about it a bit later.

6. **Logical Operators / Connectives:** FOPL uses the same five connectives of PL i.e.

   $\sim$ (not), $\wedge$ (and), $\vee$ (or), $\rightarrow$ (implication) and $\leftrightarrow$ (equivalence).

Please note that function and predicate symbols take a specified number of arguments. If they take n arguments then it is called as n-place function and predicate respectively. Also note that constants, variables and functions are referred to as terms.

Predicates are referred to as atomic formulas or atoms.

Precedence of operators & quantifiers

$\sim$ ((Higher priority)
$\forall$
$\exists$
$\wedge$
$\vee$
$\rightarrow$
$\leftrightarrow$ (Lower priority)

Quantifiers $\{\forall, \exists\}$ have same priorities as $\sim$ and parentheses have highest priorities.

Using quantifiers

(a) **Universal quantifier ($\forall$)**

It is used to represent the phrase **'for all'**. It says that something is true for all possible values of a variable.

For example. (*a*) John loves everyone

In predicate calculus, we write it as

$(\forall x)$ *LOVE* (john, x)

(*b*) for all $n, x^n = x \cdot x \cdot x \ldots x$-times is written as $\forall x\ \forall n\ Q(x, n)$ where

(*b*) **Existential quantifier ($\exists x$):**

It is used to represent the Phrase '**there exist.**' It indicates that the variable is time for at least one interpretation.

For example. (*a*) "There exists x such that $x^3 = 8$." is written as —

$\exists x$: R(x). Where R(x) is $x^3 = 8$.

(*b*) Lord Ram has a crown on his head.

$\exists$x Crown (x) $\wedge$ On head (x, Ram)

NOTE: ($\forall$) uses $\rightarrow$ connective whereas $\exists$ uses $\wedge$ connective

(*c*) **Nested quantifiers:**

We can use both $\forall$ and $\exists$ quantifiers separately.

For example. 'Brothers are siblings' can be written as —

$\forall x\ \forall y$ Brother (x, y) $\rightarrow$ Sibling (x, y)

or $\forall x, y$ sibling (x, y) $\leftrightarrow$ sibling (y, x)

Example 2: 'Everybody loves somebody' can be written as —

$\forall x\ fy$ loves (x, y)

or $Q(x, n) = x^n$.

Connection between $\forall$ and $\exists$

The two quantifiers are actually intimately connected to each other, through negation. Saying that everyone dislikes garlic is the same as saying that there does not exist someone who likes the garlic:

$\forall x \neg$ likes (x Garlic) $= \neg \exists x$ likes (x, Garlic)

We can go one step further: "Everyone likes ice cream" means that there is no one who does not like ice cream:

$\forall x$ likes (x, Ice Cream) $= \neg \exists x$ likes (x, Ice Cream)

Because $\forall$ is really a conjunction over the universe of objects and $\exists$ is a disjunction, it should not be surprising that they obey de Morgan's rules. The de Morgan rules for quantified and unquantified sentences are:

1. $\forall x \neg P \equiv \neg \exists x P$
2. $\neg \forall x P \equiv \exists x \neg P$
3. $\forall x P \equiv \neg \exists x \neg P$
4. $\exists x P \equiv \neg \forall x \neg P$
5. $\neg P \wedge \neg Q \equiv \neg (P \vee Q)$
6. $\neg (P \wedge Q) \equiv (\neg P \vee \neg Q)$
7. $P \wedge Q \equiv \neg (\neg P \vee \neg Q)$
8. $P \vee Q \equiv \neg (\neg P \wedge \neg Q)$

Thus, we do not really need both $\forall$ and $\exists$ as we do not really need both $\wedge$ and $\vee$.

The range over which a variable is valid is called its scope.

For example. $(\forall x)$ (MAN (x) $\rightarrow$ Mortal (x))

Here, scope of ($\forall x$) is (MAN (x) $\rightarrow$ Mortal (x))

An occurrence of a variable x in a formula is said to be **bound** if and only if the occurrence is within the scope of a quantifier employing the variable x, else a varia[illegible] said to be **free.**

For example. In $(\forall x)\, P(x, y)$

x is bound

and y is free

Example 2: In $(\exists x)(\forall y)\, P(x, y) \wedge Q(x)$

x and y are bound in $P(x, y)$ x is free in $Q(x)$ *as* $Q(x)$ is outside the scope of $(\exists x)$ and $(\forall y)$.

A formula is said to be **closed** if there are no free occurrence of any variable in it.

For example. $(\exists x)(\forall y)(P(x, y) \wedge Q(x))$ is a closed formula

whereas $(\forall y)\, P(x, y)$ is not.

We are in a position to solve some problems now.

EXAMPLE 1: Represent the following facts in predicate logic —

(*a*) Snow is white.

(*b*) Marcus was a man.

(*c*) Marcus was a Pompeian.

(*d*) All Pompeians were Romans.

(*e*) Caesar was a ruler.

(*f*) All Romans were either loyal to Caesar or hated him.

(*g*) Everyone is legal to someone.

(*h*) Marcus tried to assassinate Caesar.

(*i*) If it rains then sky will be cloudy.

(*j*) If you will not work hard, you will fail.

SOLUTION: (*a*) Snow (white).

(*b*) Man (Marcus).

(*c*) Pompeian (Marcus)

(*d*) $\forall x$: Pompeian $(x) \rightarrow$ Roman (x)

(*e*) Ruler (Caesar)

(*f*) $\forall x$: Roman $(x) \rightarrow$ loyal to $(x,$ Caesar$) \vee$ hate $x,$ (Caesar)

(*g*) $\forall x\, \exists y$ loyalto (x, y)

(*h*) tryassassinate (Marcus, Caesar).

(*i*) raining $\rightarrow$ cloudy (sky)

(*j*) $\neg$ Workhard $\rightarrow$ fail

EXAMPLE 2: Write the predicate forms of the following —

(*a*) All indoor games are easy.

(*b*) Rajiv only likes cricket game.

(*c*) All dogs are mammals.

(*d*) Roses are red.

(*e*) John is father of Bob.

SOLUTION: (*a*) $\forall x$ indoor game $(x) \rightarrow$ easy (x)

(*b*) likes (rajiv, cricket)

(*c*) $\forall x$ dog $(x) \rightarrow$ Mammal (x)

(*d*) red (roses)

(*e*) father (john, bob).

EXAMPLE 3: Given formulas S_1 and S_2 below. Show that $Q(a)$ is a logical consequence of the two –

$S_1 : (\forall x)\,(P(x) \rightarrow Q(x))$

$S_2 : P(a)$

SOLUTION: As $(\forall x)\,(P(x \rightarrow Q(x))$

Now, $x = a$ in S_2

i.e., $Q(a)$ is a logical consequence of the two statements S_1 and S_2.

EXAMPLE 4: Translate the following statements in FOPL—

(*i*) All empolyees earning more than Rs. 60,000 per year pay income tax.

(*ii*) No employee of University earns more than the VC.

(*iii*) Only old people get sick.

(*iv*) All men are people.

(*v*) All Pompeians were Romans.

(*vi*) All Romans were either loyal to Caesar or hated him.

(*vii*) People only try to assassinate rulers they are not loyal to.

(*viii*) Everyone is loyal to someone.

(*ix*) It is now 2013.

(*x*) Mohan has exactly two friends.

(*xi*) American vegetarians do not take milk but Indian vegetarians do.

(*xii*) For every natural number, there is a natural number greater than it.

(*xiii*) Nothing beautiful is evil.

(*xiv*) Some highly qualified scientists are unemployed in Russia but in America all qualified scientists are employed.

SOLUTION: (*a*) For all x, if x is an employee and x earns more than 60,000, x pay tax.

Let employee be EMPLOYEE and pay tax be TAX.

Then, we get—

$\forall x$: ((EMPLOYEE (x) & GE $(i(x), 60{,}000) \rightarrow$ TAX(x)).

(*b*) Let employees $= x$

and $VC = y$

For all x and for all y, if x is employee and y is VC then x cannot earn more than y.

$\therefore$ $\forall xy\,((E(x)\ \&\ P(y) \rightarrow \sim GE(i(x), i(y))$.

(*c*) There exists some x, if x is people and x is old,

Let people be $P(x)$,

Old be $O(x)$,

Sick be $S(x)$

$\therefore$ We get—

$\exists x$: $(P(x)\ \&\ O(x) \rightarrow S(x))$

(*d*) For all x, if x is men, x is people.

Let Men → MEN

People → PEOPLE

$\therefore \quad \forall x$: MEN $(x) \rightarrow$ PEOPLE(x)

(*e*) For all x, if x is Pompeians, x were Romans.

$\forall x$: Pompeian $(x) \rightarrow$ Roman (x)

(*f*) For all x, if x were Romans, x were loyal to Caesar or x hated him.

$\therefore \quad \forall x$: Roman $(x) \rightarrow$ LOYAL $(x,$ Caesar$) \vee$ HATE $(x,$ Caesar$)$

(*g*) $\forall xy$: PEOPLE$(x) \wedge$ RULER$(y) \wedge$ ASSASSINATE $(x, y) \rightarrow \sim$ LOYAL, To (x, y)

(*h*) For all x, x is loyal to y.

$\forall x : \exists y :$ Loyal (x, y)

(*i*) now = 2013

(*j*) Let $E(x)$ where x is a person,

$M(x)$ where x is a Mohan.

$F(x)$ x has friend.

a for 2 as constant

$\therefore \quad \forall x\,(M(x) \rightarrow F(a, x))$

(*k*) Let $E(x) - x$ is vegetarian.

$A(x) - x$ is American.

$I(x) - x$ is Indian.

$M(x) - x$ takes milk.

$\sim M(x) - x$ do not take milk.

$\therefore \quad \exists x\,(E(x)\ \&\ A(x) \rightarrow \sim M(x))$

$\exists x\,(E(x)\ \&\ I(x) \rightarrow M(x))$

(*l*) For all x and y, if x is natural number and y is natural number, y is greater than x.

Let $N(x)$ be x as natural number,

$N(y)$, y is a natural number.

$G(x, y)$, x is greater than y.

$\forall x, y\,(N(x)\ \&\ N(y) \rightarrow a\,(y, x))$

(*m*) For every x, if x is nothing beautiful, x is evil.

$\forall x\,(\sim B(x) \rightarrow E(x))$.

This can also be written as—

For every x, if x is nothing and x is not beautiful, x is evil.

$\forall x : (\sim B(x) \rightarrow N(x) \rightarrow E(x))$

where $B(x)$, x is beautiful. $N(x)$, x is nothing.

$E(x)$, x is evil.

(*n*) Let us denote the following—

$S(x)$: x is highly qualified scientist

$A(x)$: x is American (scientist in America)

$R(x)$: x is Russian (scientist in Russia)

$E(x)$: x is employed.

$\sim E(x)$: x is not employed

$\therefore \quad \forall x : (S(x) \,\&\, A(x) \rightarrow E(x))$

$\exists x \,(S(x) \,\&\, R(x) \rightarrow E(x))$

EXAMPLE 5: Convert into FOPL statements:

(*i*) If x is connected to y by z, y is connected to x by z.

(*ii*) If you can get to y from z and you can get to z from y, you can get to z from x.

(*iii*) If x is connected to town y by highway z and bicycles are allowed on z, you can get to y by bike from x.

(*iv*) Town A is connected to Town B by Road 1.

(*v*) Apples are food.

(*vi*) Bill eats peanuts and is still alive.

(*vii*) Bill is brother of SUE.

(*viii*) All Pompeians died when volcano erupted in 79 A.D.

(*ix*) All men are mortal.

(*x*) Town–A and Town–E are not connected by Road 3.

(*xi*) Bikes are allowed on road-3, road-4 and road-5.

SOLUTION: (*i*) $\forall x, y, z$: (CONNECTED $(x, y, z) \rightarrow$ CONNECTED (y, x, z))

(*ii*) $\forall x, y, z$: (GETTO (z, y) and GETTO $(y, z) \rightarrow$ GETTO (x, z))

(*iii*) $\forall x, y, z$ (CONNECTED (x, y, z) and BIKE $(z) \rightarrow$ GETTO (x, y))

(*iv*) $\forall a, b$ road-1 CONNECTED $(a, b,$ road 1)

(*v*) For all x, if x is apple, x is food.

$A(x) : x$ is apple.

$F(x) : x$ is food.

$\therefore \quad \forall x : (A(x) \rightarrow F(x))$

(*vi*) $B(x) : x$ is bill.

$P(x) : x$ eats peanuts.

$A(x) : x$ is alive.

$\therefore \quad \forall x : (B(x) \rightarrow P(x) \,\&\, A(x))$

(*vii*) Brother (Bill, Sue)

(*viii*) Erupted (volcano, 79) $\wedge \forall x$ [Pompeian $(x) \rightarrow$ died $(x, 79)$]

(*ix*) For all x if x is men, x is mortal.

(*x*) If x is town A and y is town B, x and y are not connected by road-3.

$\therefore \quad \forall x, y, z : \sim$ CONNECTED (x, y, z) or

$\forall x, y, z$: (Town (x) & Town $(y) \rightarrow \sim$ CONNECTED (x, y, z)

(*xi*) $\forall x$: BIKE (x)

$\forall y$: BIKE (y)

$\forall z$: BIKE (z).

Inferencing in predicate logic

The rules of inference for the propositional formulas are applicable to predicate calculus. So propositional formulas are also predicate formulas. We have to just replace propositional variables by predicate variables.

For example.

1. $\exists x(P(x) \vee Q(x)) \equiv \exists x\, P(x) \vee \exists x\, Q(x)$
2. $\exists x(P \vee Q(x)) \equiv P \vee (\exists x\, Q(x))$
3. $\forall x(P(x) \wedge Q(x)) \equiv \forall x\, P(x) \wedge \forall x\, Q(x)$
4. $\forall x(P \wedge Q(x)) \equiv P \wedge (\forall x\, Q(x))$
5. $\neg(\exists x\, P(x)) \equiv \forall x \neg(p(x))$
6. $\forall x\, P(x) \Rightarrow \exists x\, P(x)$
7. $\neg(\exists x\, P(x)) \equiv \exists x \neg P(x))$

We are in a position to solve some problems now.

Q.1. Represent the following in symbol logic:

(*a*) All that glitters is not gold.

(*b*) Any person who is respected by every person is a king.

(*c*) God helps those who help themselves.

(*d*) Jack and Jill went up the hill.

(*e*) Ram likes Sita.

Ans. (*a*) $\neg \forall x$ Glitters $(x) \rightarrow$ Gold

(*b*) $\exists x \forall y$ King (x)

(*c*) $\forall x$ helps (God, helps (x, x))

(*d*) together (went up (jack), went up (jill)) $\rightarrow$ hill

(*e*) likes (Ram, Sita)

Resolution in Predicate Logic

In predicate logic, we need to compare the arguments of the literals also. In predicate logic, the existential and universal quantifiers are used. We do **skolemisation for that skolemisation is removing existential quantifiers and replacing the corresponding variable by a constant or a function.** A constant or a function is called as **skolem constant or function** respectively.

Resolution Steps

It requires that all statements be converted into a nonmalized clausal form. Its procedure is given below:

Step 1: Eliminate all implication and equivalency connectives (use $\sim P \vee Q$ in place of $P \rightarrow Q$ and $(\sim P \vee Q)$ and $(\sim Q \vee P)$ in place of $(P \leftrightarrow Q)$

Step 2: Move all negations in, to the immediately preceding atom. Use P in place of $\sim(\sim P)$ and De Morgans's Law, $\exists x \sim F[x]$ in place of $\sim(\forall x)F(x)$ and $\forall x \sim F[x]$ in place of $\sim(\exists x)(F[x])$.

Step 3: Rename Variables, if necessary, so that all quantifiers have different variable assignments, i.e., rename variables so that variables bound by one quantifier are not the same as variables bound by a different quantifier.

For example.: In the expression $\forall x(P(x)) \rightarrow (\exists x(Q(x)))$

rename the second "**dummy**" variable x which is bound by the existential quantifier to be a different variable, say y, to give $\forall x(P(x) \rightarrow (\exists y\, Q(y)))$.

Step 4: **The process of eliminating the existential quantifiers through a substitution process requires that all such variables be replaced by something called "Skolem functions" which are the arbitrary functions which can always assume a correct value required for an existentially quantified variable. In this 4th step we skolemize by replacing all existentially quantified variables with skolem functions.**

Step 5: Move all universal quantifiers to the left of the expression and put the expression on the right into CNF.

Step 6: Eliminate all universal quantifiers and conjunctions since they are retained implicitly. The resulting expressions are 'clauses' and the set of such expressions is said to be in clausal form.

EXAMPLE. Convert the expression: $\exists x\, \forall y\, (\forall z\, P\,(f\,(x), y, z)$

into clausal form: $\rightarrow (\exists u\, Q\,(x, u)$ and $\exists v\, R\,(y, v)))$

SOLUTION.

Step 1: Applying step (1):

$$\exists x\, \forall y\, (\sim (\forall z)\, P\,(f(x), y, z) \vee (\exists u\, Q\,(x, u) \text{ and } (\exists v)\, R\,(y, v)))\ (\exists v)\, R\,(y, v))))$$

$[\because\ F \rightarrow G = \sim F \vee G]$

Step 2: Applying step (2) we get —

$$\exists x \forall y\, (\exists z \sim P\,(f\,(x), y, z) \vee (\exists u\, Q\,(x, u) \text{ and } (\exists v)\, R\,(y, v)))\ (\exists v)\, R\,(y, v)))$$

Step 3: Step (3) is not required.

Step 4: $\forall y\, (\sim P\,(f\,(a), y, g\,(y)) \vee (Q\,(a, h(y)) \text{ and } R\,(y, I\,(y)))$

[$\because$ u exists as an existential quantifier, x **exists as an existential quantifier and so does v.** So, in this step, we need to remove them and this can be done by replacing x with a, u with $h(y)$ and v with $I(x)$.]

Step 5: After applying step 5, the result is

$$\underbrace{\forall y\, ((\sim P(f\,(a), y, g\,(y)) \vee Q(a, h\,(y))}_{F_1} \text{ and } (\sim P\,(f(a), y\, g(y)) \vee R(y, I\,(y)))$$

Step 6: After application of step 6 we obtain:

$$\sim P\,(f\,(a), y, g\,(y)) \vee Q\,(a, h\,(y))$$

$$\sim P\,(f\,(a), y, g\,(y) \vee R\,(y, I\,(y))$$

Thus, given FOPL sentence has been converted into clausal form.

Inference Rules in FOPL

Like *PL*, a key inference rule in FOPL is modus porens. *For example*.:

Assertion: *LION* (*leo*)

Implication: $\forall x$ *LION* $(x) \rightarrow$ *FE ROCIOUS* (x)

Conclusion: *Ferocious* (*leo*)

Representation using rules

Rules can be considered as a subset of predicate logic. They have become a popular representation scheme for expert systems (also called as rule-based systems). They were f... in general problem solver (GPS) system in early (1970s). **Rules hav... mponent p... referred**

to as the antecedent, premise, condition or situation and a RHS known as the consequent conclusion, action or response. LHS is also known as the if part and the RHS as the then part of the rule. Some rules also include an also part. Examples of such rules are:

IF : The temperature is greater than 95°C.

THEN: Open the relief valve.

These rules are stored in KB. The interpreter or inference engine inspects the LHS of each rule in KB and replaces the contents of working memory by the RHS of the rule. **We are in a position to solve some examples now.**

Q. 1. Resolve the following example wff into its clausal form.

$$(\forall x)\{P(x) \rightarrow \{(\forall y)[P(y) \rightarrow P(f(x, y))] \wedge \sim (\forall y)[Q(x, y) \rightarrow p(y)]\}\}$$

Ans. S1: Eliminate implication symbols — all occurrences of the $\rightarrow$ symbol in a wff are eliminated by making the substitution $\sim x | \vee x 2$ for $X1 \rightarrow X2$ throughout the wff. So, this substitution yields:

$$(\forall x)\{\sim P(x) \vee \{(\forall y)[\sim P(y) \vee P(f(x, y))] \wedge \sim (\forall y)[\sim Q(x, y) \vee p(y)]\}\}$$

S2: Reduce Scopes of negation symbols — We want each negation symbol, ~, to apply to at most one atomic formula. By making use of de Morgan's laws we get:

$$(\forall x)\{\sim P(x) \vee \{(\forall y)[\sim P(y) \vee P(f(x, y))] \wedge (\forall y)[Q(x, y) \wedge \sim p(y)]\}\}$$

S3: Standardize variables—Instead of writing, $(\forall x)[P(x) \rightarrow (\exists x)\, Q(x)]$, we write

$$(\forall x)[P(x) \rightarrow (\exists y)\, Q(y)]$$

i.e., one type of dummy variable should be used with each type of quantifier. So in our wff, x-variable (dummy viz.) is used with universal ($\forall$) quantifier whereas y is used with Existential quantifier ($\exists$) . Standardizing our example wff yields:

$$(\forall x)\{\sim P(x) \vee \{(\forall y)[\sim P(y) \vee P(f(x, y))] \wedge (\exists w)[Q(x, w) \wedge \sim P(w)]\}\}$$

S4: Eliminate existential quantifiers—Consider the wff : $(\forall y)\,[(\exists x)\, P(x, y)]$ which might be read as "for all y, there exists an x (possibly depending on y) such that $P(x, y)$." Note that because existential quantifier is within the scope of a universal quantifier, we allow the possiblity that value x that exists might depend on the value of y. Let this dependence be explicitly defined by some function $P(x, y)$. which maps each value of y into x that "exists". Such a function is called a **skolem function**. If we use the skolem function in place of the x that exists, we can eliminate the existential quantifier 'all' together and write $(\forall y)\, P[g(y), y]$.

> **NOTE.** If the existential quantifier being eliminated is not within the scope of any universal quantifiers, we use a skolem function of no arguments, which is just a constant. Thus, $(\exists x)\, P(x)$ becomes $P(A)$ where the constant symbol A is used to refer to the entity that we know exists.

Eliminating all existentially quantified variables from given (there is just one) in our wff yields:

$$(\forall x)\{\sim P(x) \vee \{\forall y)[\sim P(y) \vee P(f(x, y))] \vee [Q(x, g(x)) \wedge \sim P(g(x))]\}\}$$

where $\boxed{w = g(x)}$; $g(x)$ is a skolem function.

S5: Convert to prenex form—We may now move all of the universal quantifiers to the front of the wff. The resulting wff is said to be in **prenex form**. So we get:

$$(\forall x)\,(\forall y)\,\{\sim P(x) \vee \{[\sim P(y) \vee P(f(x, y))] \wedge [Q(x, g(a)) \wedge \sim P(g(x))]\}$$

S6: Put the matrix in CNF — We may put any matrix into CNF by repeatedly using one of the distributive rules, i.e.,

$$\times 1 \vee (\times 2 \wedge \times 3) \text{ by } (\times 1 \vee \times 2) \wedge (\times 1 \vee \times 3).$$

So, our equation wff becomes:

$$(\forall x)(\forall y)\{[\sim P(x) \vee \sim P(y) \vee P(f(x, y))] \wedge [\sim P(x) \vee Q(x, g(x))] \wedge [\sim P(x) \vee p(g(x))]\}$$

S7: Eliminate universal quantifiers — We get:

$$\sim P(x) \vee \sim P(y) \vee P(f(x, y)) \wedge [\sim p(x) \vee Q(x, g(x))] \wedge [\sim P(x) \vee \sim P(g(x))]$$

S8: Eliminate $\wedge$ symbols — We get

$$\left.\begin{array}{l} \sim P(x) \vee \sim P(y) \vee P[f(x, y)], \\ \sim P(x) \vee Q[x, g(x)], \\ \sim P(x) \vee \sim P[g(x)] \end{array}\right\} \text{ is required clausal form}$$

Q. 2. Show the validity of the following sentences. "All men are mortal. John is a man. Therefore John is mortal."

$$(\forall x)\{P(x) \rightarrow \{(\forall y)[P(y) \rightarrow P(f(x, y))] \wedge \sim (\forall y)[Q(x, y) \rightarrow p(y)]\}\}$$

Ans. Say,

MAN (x) — x is a man.

MORTAL (x) — x is mortal.

In *FOPL*, we have

$$((\forall x)(\text{MAN}(x) \rightarrow \text{MORTAL}(x)) \wedge \text{MAN (John)}) \rightarrow \text{MORTAL (John)}$$

Convert it to Skolem form —

$$\propto : \sim ((\text{MAN}(x) \vee \text{MORTAL}(x)) \wedge \text{MAN (John)}) \vee \text{MORTAL (John)}$$

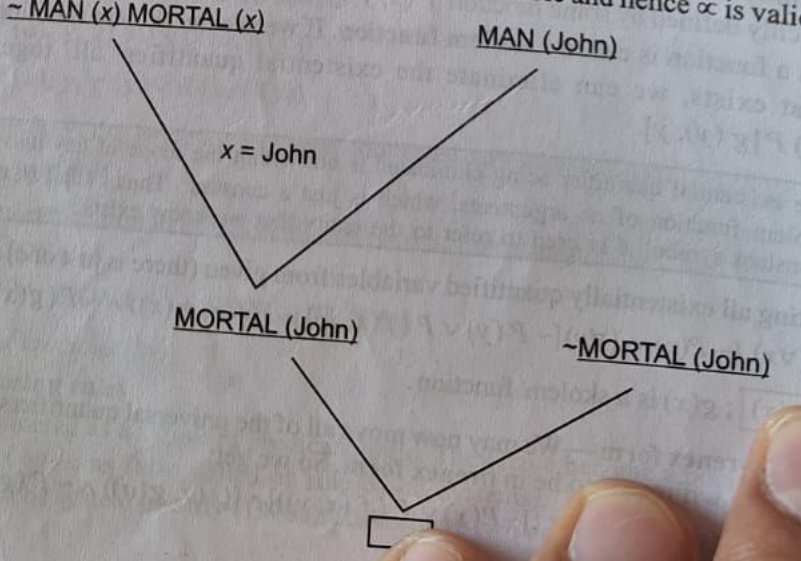
If we show that $\sim\propto$ is unsatisfiable then $\propto$ is valid.

$$\sim \propto : (\sim \text{MAN}(x) \vee \text{MORTAL}(x)) \wedge \text{MAN (John)} \wedge \sim \text{MORTAL (John)}$$

$\therefore$ A set of clauses, S, that can be obtained from $\sim\propto$ is —

$$S = \{\sim \text{MAN}(x) \vee \text{MORTAL}(x), \text{MAN (John)}, \sim \text{MORTAL (John)}\}$$

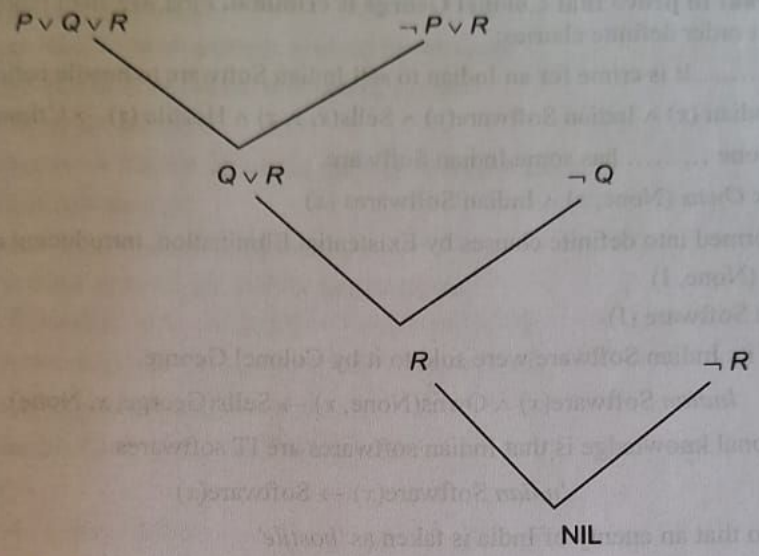
If we show that S is unsatisfiable then $\sim\propto$ is also unsatisfiable and hence $\propto$ is valid. In tree form,



Q. 3. Given the following predicates, show how resolution process can be applied to resolve —

$T: P \vee Q \vee R \quad U: \sim P \vee R \quad V: \sim Q \quad W: \sim R$

Ans. The resolution tree is —



Q. 4. Consider the following axioms —

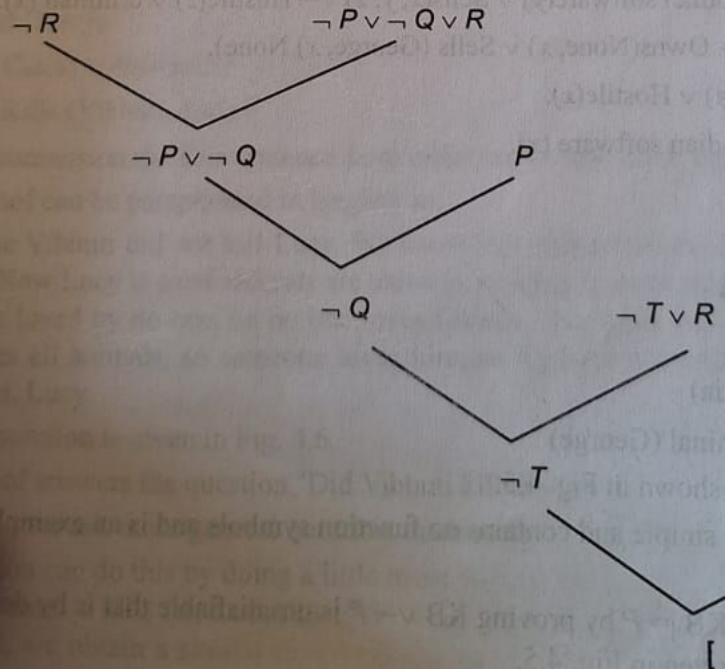
$P, (P \wedge Q) \rightarrow R, (S \vee T) \rightarrow Q, T$

Prove that *R* is true by resolution.

Ans. We negate the goal, i.e., we negate *R*, i.e, $\neg R$. We apply resolution now —

We convert the given axioms into clausal form —

1. P 2. $\neg P \vee \neg Q \vee R$ 3. $\neg S \vee Q$ 4. $\neg T \vee Q$ 5. T



Q. 5. Consider the following problem:

The law says that it is crime for an Indian to sell Indian Software to hostile nations. The country, named None, an enemy of India has some Indian software and all its software were sold to it by Colonel George who is an Indian. Give its resolution proof.

Ans. We want to prove that Colonel George is criminal. First, we shall represent these facts as first order definite clauses:

1. .....It is crime for an Indian to sell Indian Software to hostile nations.

   Indian $(x) \wedge$ Indian Software$(y) \wedge$ Sells$(x, y, z) \wedge$ Hostile $(z) \rightarrow$ Criminal (x)

2. None ..... has some Indian Software.

   $\exists x$ *Owns* (None, x) $\wedge$ Indian Softwares (x)

which is transformed into definite clauses by Existential Elimination, introducing new constant I

Owns (None, I)

Indian Software (I)

3. All of its Indian Software were sold to it by Colonel George.

   Indian Software$(x) \wedge$ Owns(None, x) $\rightarrow$ Sells(George, x, None)

   Additional knowledge is that Indian softwares are IT softwares

   Indian Software$(x) \rightarrow$ Software(x)

   and also that an enemy of India is taken as '*hostile*'

   Enemy$(x$, India$) \rightarrow$ Hostile(x)

4. *George who is an Indian* .....

   Indian (George)

5. The country None, an enemy of India .....

   Enemy (None, India)

In CNF they are:

(*a*) $\neg$ Indian$(x) \vee \neg$ Indian software$(y) \vee$ Sells$(x, y, z) \vee \neg$ Hostile$(z) \vee$ criminal (x).

(*b*) $\neg$ Software$(x) \vee \neg$ Owns(None, x) $\vee$ Sells (George, x) None).

(*c*) $\neg$ Enemy (x, India) $\vee$ Hostile(x).

(*d*) $\neg$ Missile $(x) \vee$ Indian software (x).

(*e*) Owns (None, I).

(*f*) Software (I).

(*g*) Criminal (George)

(*h*) Indian (George)

(*i*) Enemy (None, India)

and the negated goal $\neg$ criminal (George)

The resolution proof is shown in Fig. 4.5.

This knowledge base is simple and contains no function symbols and is an example of Datalog knowledge base.

Resolution proves that KB $\models P$ by proving KB $\vee \neg P$ is unsatisfiable that is by deriving empty clause. Resolution proof is given in Fig. 4.5.

We may notice the "single spine" beginning with the goal clause, resolving against clauses from the knowledge base until the empty clause is generated. This is characteristic of resolution on Horn clause knowledge bases.

This example makes use of Skolemisation and involves clauses which are not definite clauses. This results in a somewhat complex structure. The problem in English is as follows:

To prove Vibhuti Killed Lucy

1. Everyone who loves all animals is loved by someone.
2. Everyone who kills an animal is loved by no one.
3. Gaurav loves all animals.
4. Either Gaurav or Vibhuti killed the cat who is named Lucy.
5. Did Vibhuti kill the cat?

Goal, G to prove that Vibhuti killed Lucy.

Conversion is First-order Logic and the negated goal.

(*a*) $\forall x\,[\forall y\,\text{Animal}(y) \rightarrow \text{Loves}\,(x,y)] \rightarrow [\exists y\,\text{Loves}\,(y, x)]$.

(*b*) $\forall x\,[\exists y\,\text{Animal}\,(y) \wedge \text{Kills}\,(x, y)] \rightarrow [\forall z \neg \text{Loves}\,(z, x)]$.

(*c*) $\forall x\,\text{Animal}\,(y) \rightarrow \text{Loves}\,(\text{Gaurav}, x)$.

(*d*) $\text{Kills}\,(\text{Gaurav}, \text{Lucy}) \vee \text{Kills}\,(\text{Vibhuti}, \text{Lucy})$

(*e*) $\text{Cat}(\text{Lucy})$

(*f*) $\forall x\,\text{Cat}(x) \rightarrow \text{Animal}\,(x)$.

(*g*) $\text{Kills}\,(\text{Vibhuti}, \text{Lucy})$

and conversion to CNF to each sentence gives

(*a*) $\text{Animal}\,(F(x)) \vee \text{Loves}\,(G(x), x)$

(*b*) $\neg\,\text{Loves}\,(x, F(x)) \vee \text{Loves}\,(G(x), x)$

(*c*) $\neg\,\text{Animal}\,(y) \vee \neg\,\text{Kills}(x, y) \vee \neg\,\text{Loves}(z, x)$

(*d*) $\neg\,\text{Animal}\,(x) \vee \neg\,\text{Loves}\,(\text{Gaurav}, x)$

(*e*) $\text{Kills}(\text{Gaurav}, \text{Lucy}) \vee \text{Kills}\,(\text{Vibhuti}, \text{Lucy})$

(*f*) $\text{Cat}\,(\text{Lucy})$

(*g*) $\neg\,\text{Cat}(x) \vee \text{Animal}(x)$

(*h*) $\neg\,\text{Kills}\,(\text{Vibhuti}, \text{Lucy})$

After conversion the first sentence from order conversion gives two sentences.

The proof can be paraphrased in English as:

Suppose Vibhuti did not kill Lucy. We know that either Gaurav or Vibhuti did; thus Gaurav must have. Now Lucy is a cat and cats are animals, so Lucy is an animal. Because anyone who kills an animal is loved by no one, so no one loves Gaurav. But in the knowledge base it is given that Gaurav loves all animals, so someone loves him, so we have a contradiction. Therefore, Vibhuti killed the cat, Lucy.

The Resolution is given in Fig. 4.6.

The proof answers the question "Did Vibhuti kill the cat?"

But often we ask more general question, such as "who killed the cat?"

Resolution can do this by doing a little more work. The goal is ∃w kills (w, Lucy) which when negated becomes ¬ kills (w, Lucy) in CNF. Repeating the proof in the above Fig., with the new ne... goal, we obtain a similar proof tree, the substitution then becomes $w/_{vibhuti}$ in one of the