Knowledge Representation

Inference in Propositional Logic, First order Predicate logic, **Resolution,** Logical Reasoning, Forward chaining, Backward chaining,

Knowledge representation techniques: Semantic networks and frames.

Resolution

Resolution method is an inference rule which is used in both Propositional as well as First-order Predicate Logic in different ways.

This method is basically used for proving the satisfiability of a **sentence(if it is possible to find an interpretation that makes the formula true**).

In resolution method, we use **Proof by Refutation** technique to prove the given statement.

The key idea for the resolution method is to use the knowledge base and negated goal to obtain null clause(which indicates contradiction).

Since the knowledge base itself is consistent, the contradiction must be introduced by a negated goal. As a result, we have to conclude that the original goal is true.

Resolution method is also called **Proof by Refutation**.

Resolution Method in Propositional Logic

In propositional logic, resolution method is the only inference rule which gives a new clause when two or more clauses are coupled together.

The idea of resolution is simple: if we know that

- p is true or q is true
- and we also know that p is false or r is true
- then it must be the case that q is true or r is true.

This line of reasoning is formalized in the Resolution Tautology:

```
(p OR q) AND (NOT p OR r) -> q OR r
```

The process followed to convert the propositional logic into resolution method contains the below steps:

- Convert the given axiom into clausal form
- Apply and proof the given goal using negation rule.
- Use those literals which are needed to prove.
- Solve the clauses together and achieve the goal.

Conjunctive Normal Form(CNF): In propositional logic, the resolution method is applied only to those clauses which are disjunction of literals.

$(A_1 \lor B_1) \land (A_2 \lor B_2) \land \ldots. \land (A_n \lor B_n)$

CNF can also be described as AND of ORS

Disjunctive Normal Form (DNF): This is a reverse approach of CNF.

$(A_1 \land B_1) \lor (A_2 \land B_2) \lor \ldots \lor (A_n \land B_n)$

In DNF, it is OR of ANDS, a sum of products

Example of propositional Logic

Consider the following Knowledge Base:

- The humidity is high or the sky is cloudy.
- If the sky is cloudy, then it will rain.
- If the humidity is high, then it is hot.
- It is not hot.

Goal: It will rain.

Solution: Let's construct propositions of the given sentences one by one:

1. Let, P: Humidity is high.

Q: Sky is cloudy.

It will be represented as **P V Q.**

2) Q: Sky is cloudy. ...**from(1)**

Let, R: It will rain.

It will be represented as b**Q → R.**

3) P: Humidity is high. ...**from(1)**

Let, S: It is hot.

It will be represented as **P → S.**

4) ¬S: It is not hot.

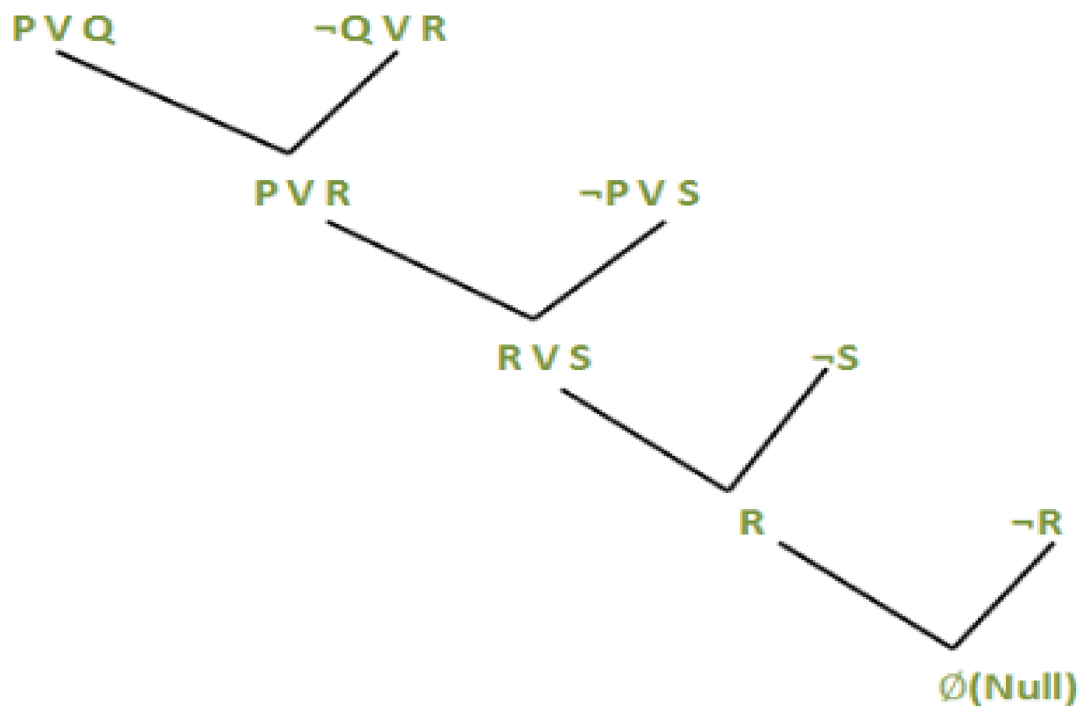
Applying resolution method:

In (2), Q → R will be converted as (¬Q V R)

In (3), P → S will be converted as (¬P V S)

Negation of Goal (¬R): It will not rain.

Finally, apply the rule as shown below:



After applying Proof by Refutation (Contradiction) on the goal, the problem is solved, and it has terminated with a **Null clause (Ø).** Hence, the goal is achieved. Thus, It is not raining.

Example

Prove R

1	P v Q
2	$P \rightarrow R$
3	$Q \rightarrow R$

Prove R

1	P v Q
2	P → R
3	Q → R

Step	Formula	Derivation
1	P v Q	Given
2	¬ P v R	Given
3	¬ Q v R	Given
4	¬ R	Negated conclusion
5	Q v R	1,2
6	¬ P	2,4
7	¬ Q	3,4
8	R	5,7
9	•	4,8

Prove R

1	$(P \rightarrow Q) \rightarrow Q$
2	$(P \rightarrow P) \rightarrow R$
3	$(R \rightarrow S) \rightarrow \neg(S \rightarrow Q)$

Resolution Proof Example

Prove R

1	$(P \rightarrow Q) \rightarrow Q$
2	$(P \rightarrow P) \rightarrow R$
3	$(R \rightarrow S) \rightarrow \neg(S \rightarrow Q)$

	$P \vee Q$	
	$P \vee R$	
	$\neg P \vee R$	
	$R \vee S$	
	$R \vee \neg Q$	
	$\neg S \vee \neg Q$	
	$\neg R$	Neg
	S	4,7
	$\neg Q$	6,8
	P	1,9
	R	3,10
	•	7,11

Resolution in FOPL

Resolution requires that all statements must be converted into a clausal form. We define a Clause as the disjunction of a number of literals.

Clausal Conversion Procedure

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

Step 2. Move all negations in to immediately precede an atom (use P in place of $\neg(\neg P)$, and DeMorgan's laws, $\exists x\ \neg F[x]$ in place of $\neg(\forall x)\ F[x]$ and $\forall x\ \neg F[x]$ in place of $\neg((\exists x)\ F[x])$.

Step 3. Rename variables, if necessary, so that all quantifiers have different variable assignments; that is, 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 different variable, say y, to give $\forall x\ (P(x) \rightarrow (\exists x\ (Q(y)))$.

Step 4. Skolemize by replacing all existentially quantified variables with Skolem functions as described above, and deleting the corresponding existential quantifiers.

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 set of such expressions is said to be in clausal form.

Skolemization, the replacement of existentially quantified variables with Skolem functions and deletion of the respective quantifiers, is then accomplished as follows:

1. If the first (leftmost) quantifier in an expression is an existential quantifier, replace all occurrences of the variable it quantifies with an arbitrary constant not appearing elsewhere in the expression and delete the quantifier. This same procedure should be followed for all other existential quantifiers not preceded by a universal quantifier, in each case, using different constants symbols in the substitution.

2. For each existential quantifier that is preceded by one or more universal quantifiers, replace all occurrences of the existentially quantified variable by a function symbol not appearing elsewhere in the expression. The arguments assigned to the function should match all the variables appearing in each universal quantifier which precedes the existential quantifier. This exisdtential quantifier should be deleted. The same process should be repeated for each remaining existential quantifier using a different function symbol and choosing function arguments that correspond to all universally quantified variables that precede the existentially quantified variable being replaced.

CASES

(i) $\exists x \forall y \forall z\, A \qquad \forall y \forall z\, A[c|x]$

Skolem constant

(ii) $\exists x \forall y \forall z \left(P(x,y) \to Q(x,z)\right)$ is

$$\forall y \forall z \left(P(c,y) \to Q(c,z)\right)$$

(iii) $\exists x_1 \cdots \exists x_k \forall y_1 \cdots \forall y_n\, A$ is

$$\forall y_1 \cdots \forall y_n\, A[c_1|x_1, \cdots c_k|x_k]$$

$c_1, c_2 \ldots c_k$ are Skolem constant.

Skolem Functions :-

(1) $\forall y \exists z\, P(y,z)$ is $\forall y\, P(y, f(y))$

where f is a new unary function called Skolem function.

(2) $\forall y \exists x_1 \cdots \exists x_k \forall y_1 \cdots \forall y_n A$ is

$\forall y \forall y_1 \cdots \forall y_n A[f_1(y)/x_1, \cdots f_k(y)/x_k]$,

$f_1 \cdots f_k$ are new Skolem functions.

(3) $\forall x \exists y \forall z \exists u\, A(x,y,z,u)$ is

$\forall x \forall z A[f(x)/y, g(x,z)/u]$

Unification

Any substitution that makes two or more expressions equal is called a unifier for the expressions.

Applying a substitution to an expression E produces an instance E’ of E where E’=Eβ

Example

(a) Marcus was a man.

(b) Marcus was a Roman.

(c) All men are people.

(d) Caesar was a ruler.

(e) All Romans were either loyal to Caesar or hated him (or both).

(f) Everyone is loyal to someone.

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

(h) Marcus tried to assassinate Caesar.

Prove: hate(Marcus, Caesar)

Convert to First Order Logic

(a) Marcus was a man.

(b) Marcus was a Roman.

(c) All men are people.

(d) Caesar was a ruler.

(e) All Romans were either loyal to Caesar or hated him (or both).

(f) Everyone is loyal to someone.

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

(h) Marcus tried to assassinate Caesar.

(a) man(marcus)

(b) roman(marcus)

(c) $\forall$X. man(X) $\rightarrow$ person(X)

(d) ruler(caesar)

(e) $\forall$X. roman(x) $\rightarrow$ loyal(X,caesar) $\vee$ hate(X,caesar)

(f) $\forall$X$\exists$Y. loyal(X,Y)

(g) $\forall$X$\forall$Y. person(X) $\wedge$ ruler(Y) tryassasin(X,Y) $\rightarrow$ $\neg$loyal(X,Y)

(h) tryassasin(marcus,caesar)

Negate!

$\neg$hate(X,caesar)

Convert to Clausal Form

(a) man(marcus)
(b) roman(marcus)
(c) $\forall X.\ man(X) \rightarrow person(X)$
    $(\neg man(X), person(X))$
(d) ruler(caesar)
(e) $\forall X.\ roman(X) \rightarrow loyal(X,caesar) \vee hate(X,caesar)$
    $(\neg roman(X), loyal(X,caesar), hate(X,caesar))$
(f) $\forall X \exists Y.\ loyal(X,Y)$
    $(loyal(X,f(X)))$
(g) $\forall X \forall Y.\ person(X) \wedge ruler(Y) \wedge tryassasin(X,Y) \rightarrow \neg loyal(X,Y)$
    $(\neg person(X), \neg ruler(Y), \neg tryassasin(X,Y), \neg loyal(X,Y))$
(h) tryassasin(marcus,caesar)

Negate!

$\neg hate(X,caesar)$

$\neg man(x) \vee person(x)$ $\quad$ $man(marcus)$ $\quad$ $x/marcus$

$person(marcus)$ $\quad$ $\neg person(marcus) \vee \neg ruler(Y)$
$\vee \neg tryass(marcus, Y)$
$\vee \neg loyal(marcus, Y)$

$Y/Caesar$

$\neg ruler(Y) \vee \neg tryass(marcus, Y) \vee \neg loyal(marcus, Y)$ $\quad$ $ruler(Caesar)$

$tryass(marcus, Caesar)$ $\quad$ $\neg tryass(marcus, caesar) \vee \neg loyal(marcus, Y)$

$\neg loyal(marcus, Caesar)$ $\quad$ [ ] $\quad$ $loyal(marcus, Caesar)$