2 Knowledge Representations:

The process of acquisition of knowledge could be carryout manually or automatically. In manual mode, knowledge receives from one or more domain experts. Whereas in automatic mode, a machine learning system is used for autonomous learning and refining knowledge from the external world. Now knowledge after acquisition can be represented by various tools and techniques such:

- ***Logic Representation***
    - *Propositional Calculus.*
    - *Predicate Calculus.*
- ***Production Rules Systems.***
- ***Networks Representation***
    - *Semantic Nets*

2.1 PROPOSITIONAL CALCULUS

2.1.1 Syntax of Propositional Calculus:

The propositional calculus and predicate calculus are first of all languages. Using their words, phrases, and sentences, we can represent and reason about properties and relationships in the world. The first step in describing a language is to produce the pieces that make it up: its set of symbols.

DEFINITION: Propositional Calculus Symbols

- The symbols of propositional calculus are: {P, Q, R, S, …}
- Truth symbols: **{True, false}**
- Connectives: **{∧ ,∨ , ¬ , → , ≡}**

Propositional symbols denote ***propositions***, or statements about the world that may be either true or false, such as "***the car is red***" or "***water is wet***." Propositions are denoted by uppercase letters near the end of the English alphabet Sentences in the propositional calculus are fanned from these atomic symbols according to the following rules:

DEFINITIONS: Propositional Calculus Sentence

- Every propositional symbol and truth symbol is a sentence.

  For example: **true, P, Q,** and **R** are sentences**.**
- The ***negation*** of a sentence is a sentence.

  For example: ¬**P** and ¬**false** are sentences.

- The ***conjunction***, **AND**, of two sentences is a sentence.

  For example: **P** ∧ ¬**P** is a sentence.
- The ***disjunction***, **OR** of two sentences is a sentence.

  For example: **P** ∨ ¬**P** is a sentence.
- The ***implication*** of one sentence from another is a sentence.

  For example: **P** → **Q** is a sentence.
- The ***equivalence*** of two sentences is a sentence.

  For example: **P** ∨ **Q** ≡ **R** is a sentence.
- Legal sentences are also called ***well-formed formulas*** or ***WFFs.***

In expressions of the form **P** ∧ **Q, P** and **Q** are called the ***conjuncts***. In **P** ∨ **Q**, **P** and **Q** are referred to as ***disjuncts***. In an implication, **P** → **Q, P** is the ***premise*** and ***Q***, the ***conclusion*** or ***consequent.***

In propositional calculus sentences, the symbols **( )** and **[ ]** are used to group symbols into sub-expressions and so to control their order of evaluation and meaning.

For Example: (P ∨ **Q)** ≡ **R** is quite different from **P** ∨ **(Q** ≡ **R)** as can be demonstrated using truth tables. An expression is a sentence, or well-formed formula, of the propositional calculus if and only if it can be formed of legal symbols through some sequence of these rules.

For Example: ((P ∧ **Q)** → **R)** ≡ ¬**P** ∨ ¬**Q** ∨ **R**

is a well-formed sentence in the propositional calculus because:

P, Q, and **R** are propositions and thus sentences.

P ∧ **Q**, the conjunction of two sentences, is a sentence.

(P ∧ **Q)** → **R**, the implication of a sentence for another, is a sentence.

¬**P** and ¬**Q**, the negations of sentences, are sentences.

¬**P** ∨ ¬**Q** the disjunction of two sentences, is a sentence.

¬**P** ∨ ¬**Q** ∨ **R**, the disjunction of two sentences, is a sentence.

((P ∧ **Q)** → **R)** ≡ ¬**P** ∨ ¬**Q** ∨ **R**, the equivalence of two sentences, is a sentence.

2.1.2 The Semantics of the Propositional Calculus:

In this section we formally define the ***semantics*** or "**meaning**" of these sentences. Because **AI** programs must reason with their representational structures, it is important to demonstrate that the truth of their conclusions depends only on the truth of their initial knowledge, i.e., that logical errors are not introduced by the inference procedures. A precise treatment of semantics is essential to this goal.

A proposition symbol corresponds to a statement about the world. For example, **P** may denote the statement "***it is raining***" or **Q**, the statement "***I live in a brown house."*** A proposition may be either true or false, given some state of the world. The truth value assignment to propositional sentences is called an ***interpretation***, an assertion about their truth in some ***possible world***.

Formally, an interpretation is a mapping from the propositional symbols into the set **{T, F}**. As mentioned in the previous section, the symbols true and false are part of the set well-formed sentences of the propositional calculus; i.e., they are distinct from the truth value assigned to a sentence. To enforce this distinction, the symbols **T** and **F** are used for truth value assignment.

Each possible mapping of truth value onto propositions corresponds to a possible world of interpretation. For example, if **P** denotes the proposition "***it is raining***" and **Q** denotes "***I am at work***" then the set of propositions {**P, Q**} has four different functional mappings into the truth values {**T, F**}. These mappings correspond to four different interpretations.

DEFINITION: PROPOSITIONAL CALCULUS SEMANTICS

An ***interpretation*** of a set of propositions is the assignment of a truth value, either **T** or **F**, to each propositional symbol. The symbol **True** is always assigned **T**, and the symbol **False** is assigned **F**.

The interpretation or truth value for sentences is determined by:

- The truth assignment of ***negation***, **¬P**, where **P** is any propositional symbol, is **F** if the assignment to **P** is **T**, and **T** if the assignment to **P** is **F**.
- The truth assignment of ***conjunction***, ∧, is **T** only when both **conjuncts** have truth value **T**; otherwise it is **F**.

- The truth assignment of ***disjunction***, ∨, is **F** only when both **disjuncts** have truth value **F**; otherwise it is **T**.
- The truth assignment *of* ***implication,→***, is **F** only when the premise or symbol before the implication is **T** and the truth value of the consequent or symbol after the implication is **F**; otherwise it is **T**.
- The truth assignment of ***equivalence***, ≡, is **T** only when both expressions have the same truth assignment for all possible interpretations; otherwise it is **F**.

The truth assignments of compound propositions are often described by ***truth tables***. A truth table lists all possible truth value assignments to the atomic propositions of an expression and gives the truth value of the expression for each assignment. Thus, a truth table enumerates all possible worlds of interpretation that may be given to an expression.

For Example, the truth table for **P ∧ Q**, **Fig.(2.1)** , lists truth values for each possible truth assignment of the operands. **P ∧ Q** is true only when both **P** and **Q** are both **T**. **Or (∨), not(¬), implies (→)**, and **equivalence (≡)** are defined in a similar fashion. The construction of these truth tables is left as an exercise.

Two expressions in the propositional calculus are equivalent if they have the same value under all truth value assignments. This equivalence may be demonstrated using truth tables. For example, a proof of the equivalence of **P → Q** and **¬P ∨ Q** is given by the truth table **Fig.(2.2).**

By demonstrating that two different sentences in the propositional calculus have identical truth tables, we can prove the following equivalences. For propositional expressions **P, Q,** and **R**:

¬ (¬P) ≡ P

P → Q ≡ ¬P ∨ Q

The contrapositive law: (P → Q) ≡ (Q → P)

De Morgan's law: ¬(P ∨ Q) ≡ (¬P ∧ ¬Q) and ¬(P ∧ Q) ≡ (¬P ∨ ¬Q)

The commutative laws: (P ∧ Q) ≡ (Q ∧ P) and (P ∨ Q) ≡ (Q ∨ P)

The associative law: ((P ∧ Q) ∧ R) ≡ (P ∧ (Q ∧ R))

The associative law: ((P ∨ Q) ∨ R) ≡ (P ∨ (Q ∨ R))

The distributive law: $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$

The distributive law: $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$

Identities such as these can be used to change propositional calculus expressions into a syntactically different but logically equivalent form. These identities may be used instead of truth tables to prove that two expressions are equivalent: find a series of identities that transform one expression into the other.

The ability to change a logical expression into a different form with equivalent truth values is also important when using ***inference rules*** (**modus ponens, and resolution**) that require expressions to be in a specific form.

P	Q	P ∧ Q
T	T	T
T	F	F
F	T	F
F	F	F

Fig. (2.1) : Truth table for AND ∧ operator

P	Q	¬P	¬P∨Q	P→Q	(¬P∨Q) = (P→Q)
T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

Fig. (2.2) : Truth table demonstrating the equivalence of (¬P∨Q) = (P→Q)

2.1.3 Theorem Proving by Propositional Logic:

We present here two techniques for logical theorem proving in propositional logic. These are : ***(1) Semantic methods***, ***(2) Syntactic methods*** of theorem proving.

(1) Semantic Method for Theorem Proving:

The following notation will be used to represent a symbolic theorem, stating that conclusion "**C**" follows from a set of premises $\mathbf{p_1, p_2, ..., p_n}$

$$\mathbf{p_1, p_2, ..., p_n \Rightarrow C}$$

In this technique, we first construct a truth table representing the relationship of $\mathbf{p_1}$ through $\mathbf{p_n}$ with "**c**". Then we test the validity of the theorem by checking whether both the ***forward and backward chaining methods***, to be presented shortly, hold good. The concept can be best illustrated with an example.

Example 1: Let us redefine p_1= ***the-sky-is-cloudy***, p_2 = ***it-will-rain*** and $p_3 \equiv p_1 \rightarrow p_2$ to be three propositions. We now form a truth table of p_1, p_2 and p_3 , and then attempt to check whether forward and backward chaining holds good for the following theorem: $p_1, p_3 \Rightarrow p_2$,

We have $p_3 \equiv p_1 \rightarrow p2 \equiv \neg p_1 \vee p_2$

p_1	p_2	p_3 $(\neg p_1 \vee p_2)$
0	0	1
0	1	1
1	0	0
1	1	1

Fig.(2.3): Truth Table of p_1, p_2, p_3

Forward chaining: *When all the **premises** are **true**, check whether the **conclusion** is **true**. Under this circumstance, we say that forward chaining holds good.*

In this example, when p_1 and p_3 are true, check if p_2 is true. Note that in the last row of the truth table, $p_1 = 1$, $p_3 = 1$ yield $p_2 = 1$. So, forward chaining holds good. Now, we test for backward chaining.

Backward chaining**:** *When all the **consequences** are **false**, check whether at least one of the **premises** is **false**.*

In this example p_2=0 in the first and third row. Note that when p_2=0, then p_1=0, in the first row and p_3=0 in the third row. So, backward chaining holds good. As forward and backward chaining both are satisfied together, the theorem: $p_1, p_3 \Rightarrow p_2$ also holds good.

Example 2: Show that for **example** 1, $p_2, p_3 \Rightarrow p_1$. It is clear from the truth **table 2** that when p_1=0, then p_2=0 (first row) and $p_3 = 1$ (first row), backward chaining holds good. But when $p_2 = p_3 = 1$, p_1=0 (second row), forward chaining fails. Therefore, the theorem does not hold good.