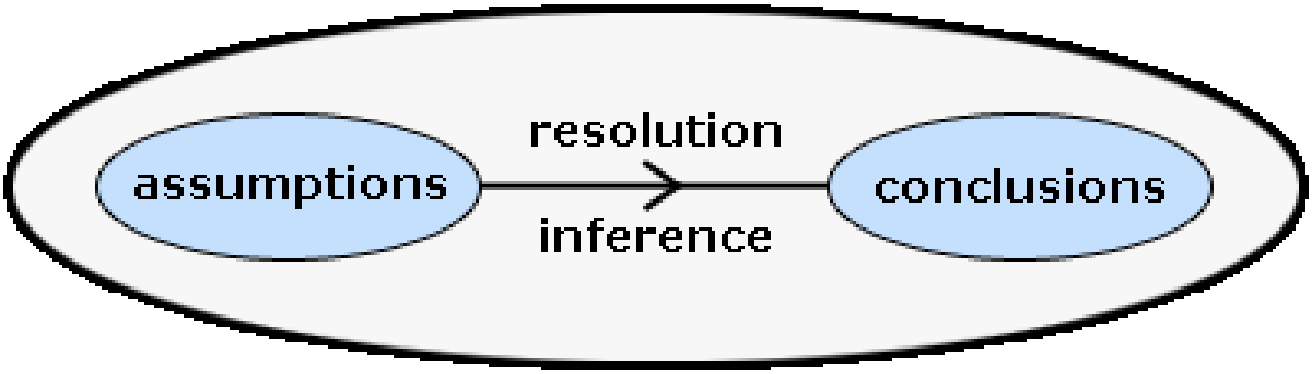
Unit – II

Logic Concepts and Logic Programming: Introduction, Propositional Calculus, Propositional Logic, Natural Deduction System, Resolution Refutation in Propositional Logic, Predicate Logic, Logic Programming. Representing Knowledge Using Rules: Logic programming, Procedural Vs Declarative knowledge, Forward Vs Backward Reasoning, Matching.

TOPIC: Introduction to Logic Concepts and Logic Programming:

Artificial Intelligence (AI) is the ability for an artificial machine to act intelligently. Logic Programming is a method that computer scientists are using to try to allow machines to reason because it is useful for knowledge representation. In logic programming, logic is used to represent knowledge and inference is used to manipulate it.



Prolog(PROgramming in LOGic), is a declarative programming language which is based on the ideas of logic programming.The idea of Prolog was to make logic look like a programming language and allow it to be controlled by a programmer to advance the research for theorem-proving.

propositional calculus and propositional logic in artificial intelligence:

Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions. A proposition is a declarative statement which is either true or false. It is a technique of knowledge representation in logical and mathematical form.

Example:

1. a) It is Sunday.
2. b) The Sun rises from West (False proposition)
3. c) 3+3= 7(False proposition)
4. d) 5 is a prime number.

Following are some basic facts about propositional logic:

- Propositional logic is also called Boolean logic as it works on 0 and 1.
- In propositional logic, we use symbolic variables to represent the logic, and we can use any symbol for a representing a proposition, such A, B, C, P, Q, R, etc.
- Propositions can be either true or false, but it cannot be both.
- Propositional logic consists of an object, relations or function, and **logical connectives**.
- These connectives are also called logical operators.
- The propositions and connectives are the basic elements of the propositional logic.
- Connectives can be said as a logical operator which connects two sentences.
- A proposition formula which is always true is called **tautology**.
- A proposition formula which is always false is called **Contradiction**.
- Statements which are questions, commands, or opinions are not propositions such as "**Where is Rohini**", "**How are you**", "**What is your name**", are not propositions.

Syntax of propositional logic:

The syntax of propositional logic defines the allowable sentences for the knowledge representation. There are two types of Propositions:

1.Atomic Propositions

2.Compound propositions

- **Atomic Proposition:** Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false.

Example:

1. a) 2+2 is 4, it is an atomic proposition as it is a **true** fact.
2. b) "The Sun is cold" is also a proposition as it is a **false** fact.
    - **Compound proposition:** Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives.

Example:

1. a) "It is raining today, and street is wet."
2. b) "Ankit is a doctor, and his clinic is in Mumbai."

Logical Connectives:

Logical connectives are used to connect two simpler propositions or representing a sentence logically. We can create compound propositions with the help of logical connectives. There are mainly five connectives, which are given as follows:

1. **Negation:** A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.
2. **Conjunction:** A sentence which has ∧ connective such as, **P ∧ Q** is called a conjunction.
   Example: Rohan is intelligent and hardworking. It can be written as,
   P=Rohanisintelligent,
   Q= Rohan is hardworking. → P∧ Q.
3. **Disjunction:** A sentence which has ∨ connective, such as **P ∨ Q**. is called disjunction, where P and Q are the propositions.
   Example: "Ritika is a doctor or Engineer",
   Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as **P ∨ Q**.
4. **Implication:** A sentence such as P → Q, is called an implication. Implications are also known as if-then rules. It can be represented as
   If it is raining, then the street is wet.
   Let P= It is raining, and Q= Street is wet, so it is represented as P → Q

5. **Biconditional:** A sentence such as **P⇔ Q is a Biconditional sentence, example If I am breathing, then I am alive**

   P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.

Following is the summarized table for Propositional Logic Connectives:

Connective symbols	Word	Technical term	Example
∧	AND	Conjunction	A ∧ B
∨	OR	Disjunction	A ∨ B
→	Implies	Implication	A → B
⇔	If and only if	Biconditional	A⇔ B
¬ or ~	Not	Negation	¬ A or ¬ B

Truth Table:

In propositional logic, we need to know the truth values of propositions in all possible scenarios. We can combine all the possible combination with logical connectives, and the representation of these combinations in a tabular format is called **Truth table**. Following are the truth table for all logical connectives:

For Negation:

P	¬ P
True	**False**
False	**True**

For Conjunction:

P	Q	P∧ Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**False**

For disjunction:

P	Q	P ∨ Q.
True	True	**True**
False	True	**True**
True	False	**True**
False	False	**False**

For Implication:

P	Q	P→ Q
True	True	**True**
True	False	**False**
False	True	**True**
False	False	**True**

For Biconditional:

P	Q	P⇔ Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**True**

Truth table with three propositions:

We can build a proposition composing three propositions P, Q, and R. This truth table is made-up of 8n Tuples as we have taken three proposition symbols.

P	Q	R	¬R	Pv Q	PvQ→¬R
True	True	True	False	True	False
True	True	False	True	True	True
True	False	True	False	True	False
True	False	False	True	True	True
False	True	True	False	True	False
False	True	False	True	True	True
False	False	True	False	False	True
False	False	False	True	False	True

Natural Deduction:

ND techniques were used as early as people did reasoning, it is unquestionable that the exact formulation of ND and the justification of its correctness was postponed until the 20th century.

Applications

There are three main fields of application of ND systems: practical, theoretical and philosophical.

ND systems became a standard tool of working logicians, mathematicians, and philosophers. At least in the Anglo-American tradition, ND systems prevail in teaching logic. They also had strong influence on the development of other types of non-axiomatic formal systems such as sequent calculi and tableau systems

Topic: Resolution Refutation in Propositional Logic

Resolution is one kind of proof technique that works this way - (i) select two clauses that contain conflicting terms (ii) combine those two clauses and (iii) cancel out the conflicting terms.

For example we have following **statements**,

(1) If it is a pleasant day you will do strawberry picking

(2) If you are doing strawberry picking you are happy.

Above statements can be written **in propositional logic** like this -

(1) strawberry_picking ← pleasant

(2) happy ← strawberry_picking

And again these statements can be written **in CNF** like this -

(1) (strawberry_picking ∨~pleasant) ∧

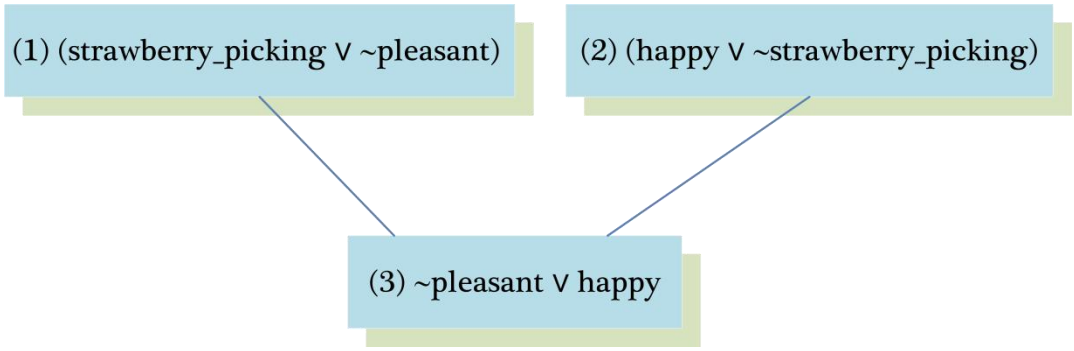
(2) (happy ∨~strawberry_picking)

By resolving these two clauses and cancelling out the conflicting terms 'strawberry_picking' and '~strawberry_picking', we can have one **newclause**,

(3) ~pleasant ∨ happy

How ? See the **figure on right**.

When we write above new clause in **infer or implies** form, we have 'pleasant → happy' or 'happy ← pleasant' i.e. If it is a pleasant day you are happy.



But sometimes from the collection of the statements we have, we want to know the answer of this question - "Is it possible to prove some other statements from what we actually know?" In order to prove this we need to make some inferences and those other statements can be shown true using **Refutation** proof method i.e. proof by contradiction using Resolution. So for the asked goal we will negate the goal and will add it to the given statements to prove the contradiction.

Let's see an **example** to understand how Resolution and Refutation work. In below example, **Part(I)** represents the English meanings for the clauses, **Part(II)** represents the propositional logic statements for given english sentences, **Part(III)** represents the Conjunctive Normal Form (CNF) of Part(II) and **Part(IV)** shows some other statements we want to **prove using Refutation proof method**.

Part(I) : English Sentences

(1) If it is sunny and warm day you will enjoy.

(2) If it is warm and pleasant day you will do strawberry picking

(3) If it is raining then no strawberry picking.

(4) If it is raining you will get wet.

(5) It is warm day

(6) It is raining

(7) It is sunny

Part(II) : Propositional Statements

(1) enjoy ← sunny ∧ warm

(2) strawberry_picking ← warm ∧ pleasant

(3) ~strawberry_picking ← raining

(4) wet ← raining

(5) warm

(6) raining

(7) sunny

Part(III) : CNF of Part(II)

(1) (enjoy ∨~sunny∨~warm) ∧

(2) (strawberry_picking ∨~warm∨~pleasant) ∧

(3) (~strawberry_picking ∨~raining) ∧

(4) (wet ∨~raining) ∧

(5) (warm) ∧

(6) (raining) ∧

(7) (sunny)

Why ∧ at the end of above statements?

Part(IV) : Other statements we want to prove by Refutation

(Goal 1) You are not doing strawberry picking.

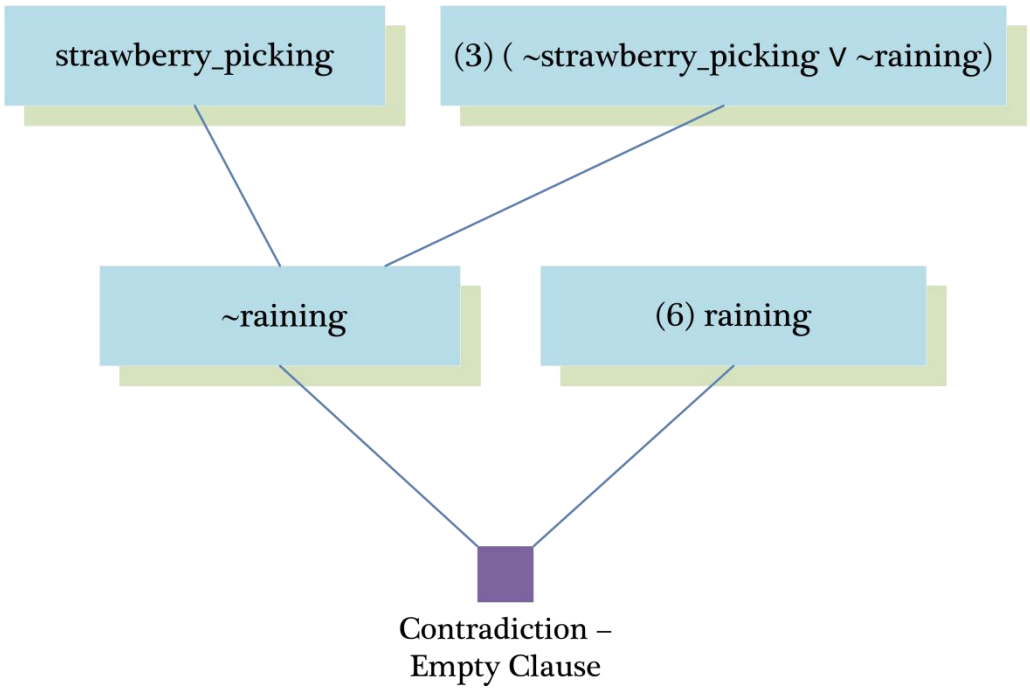
(Goal 2) You will enjoy.

(Goal 3) Try it yourself : You will get wet.

Goal 1 : You are not doing strawberry picking.

Prove : ~strawberry_picking

Assume : strawberry_picking (negate the goal and add it to given clauses).



Goal 2 : You will enjoy.

Prove : enjoy

Assume : ~enjoy (negate the goal and add it to given clauses

Predicate Logic:

Predicate Logic

Predicate Logic deals with predicates, which are propositions, consist of variables.\

A predicate is an expression of one or more variables determined on some specific domain. A predicate with variables can be made a proposition by either authorizing a value to the variable or by quantifying the variable.

The following are some examples of predicates.

- Consider E(x, y) denote "x = y"
- Consider X(a, b, c) denote "a + b + c = 0"
- Consider M(x, y) denote "x is married to y."

Quantifier:

The variable of predicates is quantified by quantifiers. There are two types of quantifier in predicate logic - Existential Quantifier and Universal Quantifier.

Existential Quantifier:

If p(x) is a proposition over the universe U. Then it is denoted as ∃x p(x) and read as "There exists at least one value in the universe of variable x such that p(x) is true. The quantifier ∃ is called the existential quantifier.

There are several ways to write a proposition, with an existential quantifier, i.e.,

(∃x∈A)p(x) or ∃x∈A such that p (x) or (∃x)p(x) or p(x) is true for some x ∈A.

Universal Quantifier:

If p(x) is a proposition over the universe U. Then it is denoted as ∀x,p(x) and read as "For every x∈U,p(x) is true." The quantifier ∀ is called the Universal Quantifier.

There are several ways to write a proposition, with a universal quantifier.

∀x∈A,p(x) or p(x), ∀x ∈A Or ∀x,p(x) or p(x) is true for all x ∈A.

Negation of Quantified Propositions:

When we negate a quantified proposition, i.e., when a universally quantified proposition is negated, we obtain an existentially quantified proposition,and when an existentially quantified proposition is negated, we obtain a universally quantified proposition.

The two rules for negation of quantified proposition are as follows. These are also called DeMorgan's Law.

Example: Negate each of the following propositions:

1.$\forall x\ p(x) \wedge \exists\ y\ q(y)$

Sol: $\sim.\forall x\ p(x) \wedge \exists\ y\ q(y))$
$\cong \sim \forall\ x\ p(x) \vee \sim \exists yq\ (y)$ $(\therefore \sim(p \wedge q) = \sim p \vee \sim q)$
$\cong \exists\ x\ \sim p(x) \vee \forall y \sim q(y)$

2. $(\exists x \in U)\ (x+6=25)$

Sol: $\sim(\ \exists\ x \in U)\ (x+6=25)$
$\cong \forall\ x \in U \sim\ (x+6)=25$
$\cong (\forall\ x \in U)\ (x+6) \neq 25$

3. $\sim(\ \exists\ x\ p(x) \vee \forall\ y\ q(y)$

Sol: $\sim(\ \exists\ x\ p(x) \vee \forall\ y\ q(y))$
$\cong \sim \exists\ x\ p(x) \wedge \sim \forall\ y\ q(y)$ $(\therefore \sim(p \vee q) = \sim p \wedge \sim q)$
$\cong \forall\ x \sim\ p(x) \wedge \exists y \sim q(y))$

Propositions with Multiple Quantifiers:

The proposition having more than one variable can be quantified with multiple quantifiers. The multiple universal quantifiers can be arranged in any order without altering the meaning of the resulting proposition. Also, the multiple existential quantifiers can be arranged in any order without altering the meaning of the proposition.

The proposition which contains both universal and existential quantifiers, the order of those quantifiers can't be exchanged without altering the meaning of the proposition, e.g., the proposition $\exists x\ \forall\ y\ p(x,y)$ means "There exists some x such that p (x, y) is true for every y."

Example: Write the negation for each of the following. Determine whether the resulting statement is true or false. Assume U = R.

1.$\forall\ x\ \exists\ m(x^2<m)$

Sol: Negation of $\forall\ x\ \exists\ m(x^2<m)$ is $\exists\ x\ \forall\ m\ (x^2 \geq m)$. The meaning of $\exists\ x\ \forall\ m\ (x^2 \geq m)$ is that there exists for some x such that $x^2 \geq m$, for every m. The statement is true as there is some greater x such that $x^2 \geq m$, for every m.

2. $\exists\ m \forall\ x(x^2<m)$

Sol: Negation of $\exists\ m\ \forall\ x\ (x^2<m)$ is $\forall\ m \exists x\ (x^2 \geq m)$. The meaning of $\forall\ m \exists x\ (x^2 \geq m)$ is that for every m, there exists for some x such that $x^2 \geq m$. The statement is true as for every m, there exists for some greater x such that x^2.

Topic: Difference between Procedural and Declarative Knowledge

Procedural Knowledge:

Procedural Knowledge also known as Interpretive knowledge, is the type of knowledge in which it clarifies how a particular thing can be accomplished. It is not so popular because it is generally not used.

It emphasize **how to do** something to solve a given problem.

Let's see it with an example:

```
var a=[1, 2, 3, 4, 5];

var b=[];

for(var i=0;i<a.length;i++)

{

  b.push(a[i]);

}

console.log(b);
```

Output is:

[1, 2, 3, 4, 5]

Declarative Knowledge:

Declarative Knowledge also known as Descriptive knowledge, is the type of knowledge which tells the basic knowledge about something and it is more popular than Procedural Knowledge.

It emphasize **what to do** something to solve a given problem.

Let's see it with an example:

```
var a=[1, 2, 3, 4, 5];

var b=a.map(function(number)

{

 return number*1});

console.log(b);
```

Output is:

[1, 2, 3, 4, 5]

In both example we can see that the output of a given problem is same because the only difference in that two methods to achieve the output or solution of problem.

Difference the Procedural and Declarative Knowledge:

S.NO	Procedural Knowledge	Declarative Knowledge
1.	It is also known as Interpretive knowledge.	It is also known as Descriptive knowledge.
2.	Procedural Knowledge means how a particular thing can be accomplished.	While Declarative Knowledge means basic knowledge about something.
3.	Procedural Knowledge is generally not used means it is not more popular.	Declarative Knowledge is more popular.
4.	Procedural Knowledge can't be easily communicate.	Declarative Knowledge can be easily communicate.
5.	Procedural Knowledge is generally process oriented in nature.	Declarative Knowledge is data oriented in nature.
6.	In Procedural Knowledge debugging and validation is not easy.	In Declarative Knowledge debugging and validation is easy.
7.	Procedural Knowledge is less effective in competitive programming.	Declarative Knowledge is more effective in competitive programming.

Topic: Forward Vs Backward Reasoning:In Artificial intelligence, the purpose of the search is to find the path through a problem space. There are two ways to pursue such a search that are forward and backward reasoning. The significant difference between both of them is that forward reasoning starts with the initial data towards the goal. Conversely, backward reasoning works in opposite fashion where the purpose is to determine the initial facts and information with the help of the given results.

BASIS FOR COMPARISON	FORWARD REASONING	BACKWARD REASONING
Basic	Data-driven	Goal driven
Begins with	New Data	Uncertain conclusion
Objective is to find the	Conclusion that must follow	Facts to support the conclusions
Type of approach	Opportunistic	Conservative
Flow	Incipient to consequence	Consequence to incipient

Definition of Forward Reasoning:

The solution of a problem generally includes the initial data and facts in order to arrive at the solution. These unknown facts and information is used to deduce the result. For example, while diagnosing a patient the doctor first check the symptoms and medical condition of the body such as temperature, blood pressure, pulse, eye colour, blood, etcetera. After that, the patient symptoms are analysed and compared against the predetermined symptoms. Then the doctor is

able to provide the medicines according to the symptoms of the patient. So, when a solution employs this manner of reasoning, it is known as **forward reasoning**.

Steps that are followed in the forward reasoning

The inference engine explores the knowledge base with the provided information for constraints whose precedence matches the given current state.

- In the first step, the system is given one or more than one constraints.
- Then the rules are searched in the knowledge base for each constraint. The rules that fulfil the condition are selected.
- Now each rule is able to produce new conditions from the conclusion of the invoked one. As a result, THEN part is again included in the existing one.
- The added conditions are processed again by repeating step 2. The process will end if there is no new conditions exist.

Definition of Backward Reasoning:

The **backward reasoning** is inverse of forward reasoning in which goal is analysed in order to deduce the rules, initial facts and data. We can understand the concept by the similar example given in the above definition, where the doctor is trying to diagnose the patient with the help of the inceptive data such as symptoms. However, in this case, the patient is experiencing a problem in his body, on the basis of which the doctor is going to prove the symptoms. This kind of reasoning comes under backward reasoning.

Steps that are followed in the backward reasoning

In this type of reasoning, the system chooses a goal state and reasons in the backward direction. Now, let's understand how does it happens and what steps are followed.

- Firstly, the goal state and the rules are selected where the goal state reside in the THEN part as the conclusion.

- From the IF part of the selected rule the subgoals are made to be satisfied for the goal state to be true.
- Set initial conditions important to satisfy all the subgoals.
- Verify whether the provided initial state matches with the established states. If it fulfils the condition then the goal is the solution otherwise other goal state is selected.

Key Differences Between Forward and Backward Reasoning in AI

1. The forward reasoning is data-driven approach while backward reasoning is a goal driven.
2. The process starts with new data and facts in the forward reasoning. Conversely, backward reasoning begins with the results.
3. Forward reasoning aims to determine the result followed by some sequences. On the other hand, backward reasoning emphasis on the acts that support the conclusion.
4. The forward reasoning is an opportunistic approach because it could produce different results. As against, in backward reasoning, a specific goal can only have certain predetermined initial data which makes it restricted.
5. The flow of the forward reasoning is from the antecedent to consequent while backward reasoning works in reverse order in which it starts from conclusion to incipient.

Conclusion

The production system structure of the search process facilitates in the interpretation of the forward and backward reasoning. The forward and backward reasoning are differentiated on the basis of their purpose and process, in which forward reasoning is directed by the initial data and intended to find the goal while the backward reasoning is governed by goal instead of the data and aims to discover the basic data and facts.

Topic:Matching: Intelligent matching is primarily used in maintaining and extracting databases, specifically those that are very large and complex in nature. It is generally implemented within database software, business intelligence solutions or a big data analytics application. It works by applying reasoning-based data matching techniques, which eventually deliver ideal or substantially related query results.

Some of the services intelligent matching provides include:

- The ability to scan each object for duplication within the target database
- The ability to remove duplicates within databases
- The ability to search and extract relevant information from big data repositories
- The ability to compare data, objects or files for similaritie