Knowledge-Based Agent in Artificial intelligence |||rd.

- o An intelligent agent needs **knowledge** about the real world for taking decisions and **reasoning** to act efficiently.
- o Knowledge-based agents are those agents who have the capability of **maintaining an internal state of knowledge, reason over that knowledge, update their knowledge after observations and take actions. These agents can represent the world with some formal representation and act intelligently**.
- o Knowledge-based agents are composed of two main parts:
  - o **Knowledge-base and**
  - o **Inference system**.

A knowledge-based agent must able to do the following:

- o An agent should be able to represent states, actions, etc.
- o An agent Should be able to incorporate new percepts
- o An agent can update the internal representation of the world
- o An agent can deduce the internal representation of the world
- o An agent can deduce appropriate actions.

The architecture of knowledge-based agent:
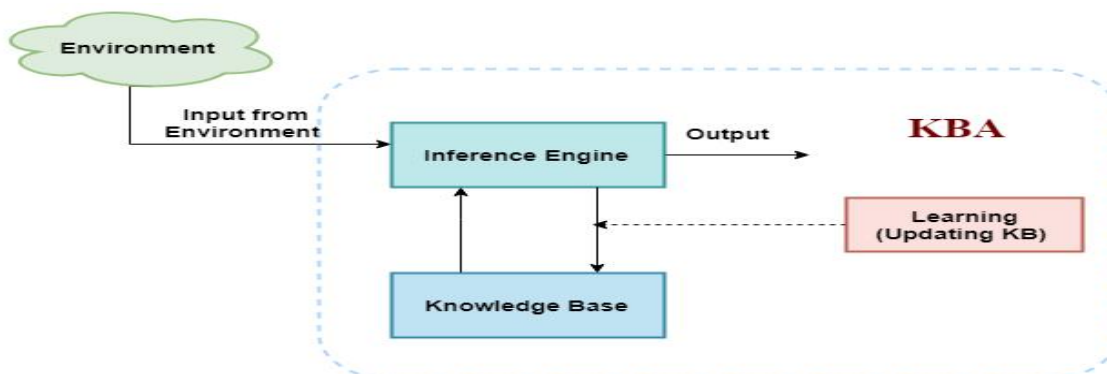
The above diagram is representing a generalized architecture for a knowledge-based agent. The knowledge-based agent (KBA) take input from the environment by perceiving the environment. The input is taken by the inference engine of the agent and which also communicate with KB to decide as per the knowledge store in KB. The learning element of KBA regularly updates the KB by learning new knowledge.

Knowledge base: Knowledge-base is a central component of a knowledge-based agent, it is also known as KB. It is a collection of sentences (here 'sentence' is a technical term and it is not identical to sentence in English). These sentences are expressed in a language which is called a knowledge representation language. The Knowledge-base of KBA stores fact about the world.

Why use a knowledge base?

Knowledge-base is required for updating knowledge for an agent to learn with experiences and take action as per the knowledge.

Inference system

Inference means deriving new sentences from old. Inference system allows us to add a new sentence to the knowledge base. A sentence is a proposition about the world. Inference system applies logical rules to the KB to deduce new information. Inference system generates new facts so that an agent can update the KB. An inference system works mainly in two rules which are given as:

- o **Forward chaining**
- o **Backward chaining**

What is knowledge representation?

Humans are best at understanding, reasoning, and interpreting knowledge. Human knows things, which is knowledge and as per their knowledge they perform various actions in the real world. **But how machines do all these things comes under knowledge representation and reasoning**. Hence we can describe Knowledge representation as following:

- o Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.

- o It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.

- o It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

What to Represent:

Following are the kind of knowledge which needs to be represented in AI systems:

- o **Object:** All the facts about objects in our world domain. E.g., Guitars contains strings, trumpets are brass instruments.
- o **Events:** Events are the actions which occur in our world.
- o **Performance:** It describe behavior which involves knowledge about how to do things.
- o **Meta-knowledge:** It is knowledge about what we know.
- o **Facts:** Facts are the truths about the real world and what we represent.
- o **Knowledge-Base:** The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the Sentences (Here, sentences are used as a technical term and not identical with the English language).

Types of knowledge

Following are the various types of knowledge:

1. Declarative Knowledge:

- o   Declarative knowledge is to know about something.
- o   It includes concepts, facts, and objects.
- o   It is also called descriptive knowledge and expressed in declarativesentences.
- o   It is simpler than procedural language.

2. Procedural Knowledge

- o   It is also known as imperative knowledge.
- o   Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- o   It can be directly applied to any task.
- o   It includes rules, strategies, procedures, agendas, etc.
- o   Procedural knowledge depends on the task on which it can be applied.

3. Meta-knowledge:

- o   Knowledge about the other types of knowledge is called Meta-knowledge.

4. Heuristic knowledge:

- o   Heuristic knowledge is representing knowledge of some experts in a filed or subject.
- o   Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

5. Structural knowledge:

- o   Structural knowledge is basic knowledge to problem-solving.
- o   It describes relationships between various concepts such as kind of, part of, and grouping of something.
- o   It describes the relationship that exists between concepts or objects.

Techniques of knowledge representation

There are mainly four ways of knowledge representation which are given as follows:

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules

1. Logical Representation

Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation. Logical representation means drawing a conclusion based on various conditions. This representation lays

down some important communication rules. It consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logics using syntax and semantics.

Syntax:

- o   Syntaxes are the rules which decide how we can construct legal sentences in the logic.
- o   It determines which symbol we can use in knowledge representation.
- o   How to write those symbols.

Semantics:

- o   Semantics are the rules by which we can interpret the sentence in the logic.
- o   Semantic also involves assigning a meaning to each sentence.

Logical representation can be categorised into mainly two logics:

a.   Propositional Logics

b.   Predicate logics

Note: We will discuss Prepositional Logics and Predicate logics in later chapters.

Advantages of logical representation:

1.   Logical representation enables us to do logical reasoning.

2.   Logical representation is the basis for the programming languages.

Disadvantages of logical Representation:

1.   Logical representations have some restrictions and are challenging to work with.

2.   Logical representation technique may not be very natural, and inference may not be so efficient.

Note: Do not be confused with logical representation and logical reasoning as logical representation is a representation language and reasoning is a process of thinking logically.

2. Semantic Network Representation

Semantic networks are alternative of predicate logic for knowledge representation. In Semantic networks, we can represent our knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects. Semantic networks are easy to understand and can be easily extended.
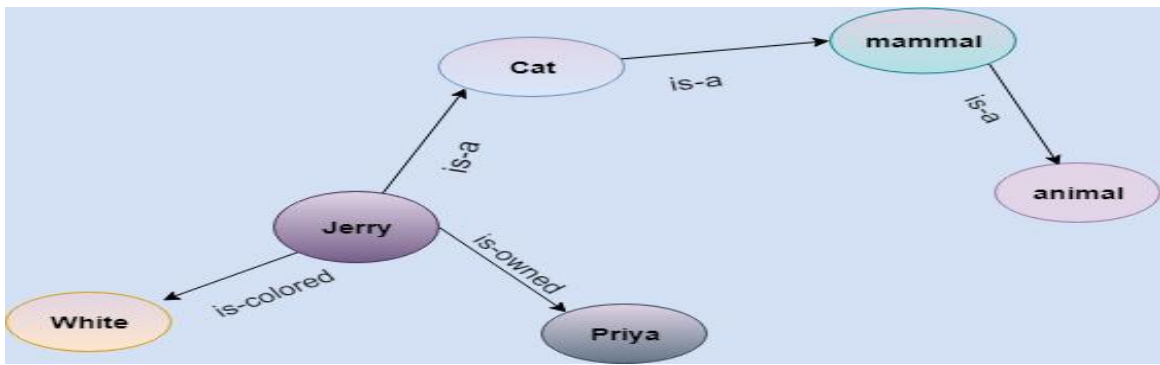
This representation consist of mainly two types of relations:

a.   IS-A relation (Inheritance)

b.   Kind-of-relation

Example: Following are some statements which we need to represent in the form of nodes and arcs.

Statements:

a.   Jerry is a cat.

b.   Jerry is a mammal

c.   Jerry is owned by Priya.

d.   Jerry is brown colored.

e.   All Mammals are animal.

In the above diagram, we have represented the different type of knowledge in the form of nodes and arcs. Each object is connected with another object by some relation.

Drawbacks in Semantic representation:

1. Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible in the worst case scenario that after traversing the entire tree, we find that the solution does not exist in this network.

2. Semantic networks try to model human-like memory (Which has 1015 neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.

3. These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.

4. Semantic networks do not have any standard definition for the link names.

5. These networks are not intelligent and depend on the creator of the system.

Advantages of Semantic network:

1. Semantic networks are a natural representation of knowledge.

2. Semantic networks convey meaning in a transparent manner.

3. These networks are simple and easily understandable.

3. Frame Representation

A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world. Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations. It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called facets.

Facets: The various aspects of a slot is known as **Facets**. Facets are features of frames which enable us to put constraints on the frames. Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values. A frame is also known as **slot-filter knowledge representation** in artificial intelligence.

Frames are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not much useful. Frames system consist of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base. The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Example: 1

Let's take an example of a frame for a book

Slots	Filters
Title	Artificial Intelligence
Genre	Computer Science

Author	Peter Norvig
Edition	Third Edition
Year	1996
Page	1152

Advantages of frame representation:

1. The frame knowledge representation makes the programming easier by grouping the related data.
2. The frame representation is comparably flexible and used by many applications in AI.
3. It is very easy to add slots for new attribute and relations.
4. It is easy to include default data and to search for missing values.
5. Frame representation is easy to understand and visualize.

Disadvantages of frame representation:

1. In frame system inference mechanism is not be easily processed.
2. Inference mechanism cannot be smoothly proceeded by frame representation.
3. Frame representation has a much generalized approach.

4. Production Rules

Production rules system consist of (**condition, action**) pairs which mean, "If condition then action". It has mainly three parts:

- o The set of production rules
- o Working Memory
- o The recognize-act-cycle

In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. And the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle.

The working memory contains the description of the current state of problems-solving and rule can write knowledge to the working memory. This knowledge match and may fire other rules.

If there is a new situation (state) generates, then multiple production rules will be fired together, this is called conflict set. In this situation, the agent needs to select a rule from these sets, and it is called a conflict resolution.

Example:

- o **IF (at bus stop AND bus arrives) THEN action (get into the bus)**
- o **IF (on the bus AND paid AND empty seat) THEN action (sit down).**
- o **IF (on bus AND unpaid) THEN action (pay charges).**
- o **IF (bus arrives at destination) THEN action (get down from the bus).**

Advantages of Production rule:

1. The production rules are expressed in natural language.
2. The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages of Production rule:

1. Production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses.

2. During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

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.

Syntax of propositional logic:

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

    a. **Atomic Propositions**

    b. **Compound propositions**

    o **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.

    o **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."

Limitations of Propositional logic:

    o We cannot represent relations like ALL, some, or none with propositional logic. Example:

        1. **All the girls are intelligent.**

        2. **Some apples are sweet.**

    o Propositional logic has limited expressive power.

    o In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

First-Order Logic/Predicate

In the topic of Propositional logic, we have seen that how to represent statements using propositional logic. But unfortunately, in propositional logic, we can only represent the facts, which are either true or false. PL is not sufficient to represent the complex sentences or natural language statements. The propositional logic has very limited expressive power. Consider the following sentence, which we cannot represent using PL logic.

- o **"Some humans are intelligent", or**
- o **"Sachin likes cricket."**

To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.

First-Order logic:

- o First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- o FOL is sufficiently expressive to represent the natural language statements in a concise way.
- o First-order logic is also known as **Predicate logic or First-order predicate logic**. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.
- o First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:

  - o **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
  - o **Relations: It can be unary relation such as:** red, round, is adjacent, **or n-any relation such as:** the sister of, brother of, has color, comes between
  - o **Function:** Father of, best friend, third inning of, end of, ...

- o As a natural language, first-order logic also has two main parts:

  1. **Syntax**
  2. **Semantics**

Syntax of First-Order logic:

The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Basic Elements of First-order logic:

Following are the basic elements of FOL syntax:

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

Atomic sentences:

- o Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
- o We can represent atomic sentences as **Predicate (term1, term2, ..., term n)**.

Example: Ravi and Ajay are brothers: => Brothers(Ravi, Ajay).
            Chinky is a cat: => cat (Chinky).

Complex Sentences:

- o   Complex sentences are made by combining atomic sentences using connectives.

First-order logic statements can be divided into two parts:

- o   **Subject:** Subject is the main part of the statement.

- o   **Predicate:** A predicate can be defined as a relation, which binds two atoms together in a statement.

Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.

Inference in First-Order Logic

Inference in First-Order Logic is used to deduce new facts or sentences from existing sentences. Before understanding the FOL inference rule, let's understand some basic terminologies used in FOL.

Substitution:

Substitution is a fundamental operation performed on terms and formulas. It occurs in all inference systems in first-order logic. The substitution is complex in the presence of quantifiers in FOL. If we write **F[a/x]**, so it refers to substitute a constant "**a**" in place of variable "**x**".

Note: First-order logic is capable of expressing facts about some or all objects in the universe.

Equality:

First-Order logic does not only use predicate and terms for making atomic sentences but also uses another way, which is equality in FOL. For this, we can use **equality symbols** which specify that the two terms refer to the same object.

Example: Brother (John) = Smith.

As in the above example, the object referred by the **Brother (John)** is similar to the object referred by **Smith**. The equality symbol can also be used with negation to represent that two terms are not the same objects.

Example: ¬(x=y) which is equivalent to x ≠y.

What is Unification?

- o   Unification is a process of making two different logical atomic expressions identical by finding a substitution. Unification depends on the substitution process.

- o   It takes two literals as input and makes them identical using substitution.

- o   Let Ψ_1 and Ψ_2 be two atomic sentences and Ψ_{12}, then it can be expressed as **UNIFY(Ψ_1, Ψ_2)**.

- o   **Example: Find the MGU for Unify{King(x), King(John)}**

Let Ψ_1 = King(x), Ψ_2 = King(John),

Substitution θ = {John/x} is a unifier for these atoms and applying this substitution, and both expressions will be identical.

- o   The UNIFY algorithm is used for unification, which takes two atomic sentences and returns a unifier for those sentences (If any exist).

- o   Unification is a key component of all first-order inference algorithms.

- o   It returns fail if the expressions do not match with each other.

- o   The substitution variables are called Most General Unifier or MGU.

E.g. Let's say there are two different expressions, **P(x, y), and P(a, f(z))**.

In this example, we need to make both above statements identical to each other. For this, we will perform the substitution.

            P(x, y)......... (i)
            P(a, f(z))......... (ii)

- o   Substitute x with a, and y with f(z) in the first expression, and it will be represented as **a/x** and f(z)/y.

- o   With both the substitutions, the first expression will be identical to the second expression and the substitution set will be: **[a/x, f(z)/y]**.

Conditions for Unification:

Following are some basic conditions for unification:

- o   Predicate symbol must be same, atoms or expression with different predicate symbol can never be unified.
- o   Number of Arguments in both expressions must be identical.
- o   Unification will fail if there are two similar variables present in the same expression.

Resolution in FOL

Resolution

Resolution is a theorem proving technique that proceeds by building refutation proofs, i.e., proofs by contradictions. It was invented by a Mathematician John Alan Robinson in the year 1965.

Resolution is used, if there are various statements are given, and we need to prove a conclusion of those statements. Unification is a key concept in proofs by resolutions. Resolution is a single inference rule which can efficiently operate on the **conjunctive normal form or clausal form**.

Clause: Disjunction of literals (an atomic sentence) is called a **clause**. It is also known as a unit clause.

Conjunctive Normal Form: A sentence represented as a conjunction of clauses is said to be **conjunctive normal form** or **CNF**.

Steps for Resolution:

1. Conversion of facts into first-order logic.
2. Convert FOL statements into CNF
3. Negate the statement which needs to prove (proof by contradiction)
4. Draw resolution graph (unification).

To better understand all the above steps, we will take an example in which we will apply resolution.

Example:

a. **John likes all kind of food.**
b. **Apple and vegetable are food**
c. **Anything anyone eats and not killed is food.**
d. **Anil eats peanuts and still alive**
e. **Harry eats everything that Anil eats.**
   Prove by resolution that:
f. **John likes peanuts.**

Step-1: Conversion of Facts into FOL

In the first step we will convert all the given statements into its first order logic.

a. ∀x: food(x) → likes(John, x)

b. food(Apple) ∧ food(vegetables)

c. ∀x ∀y: eats(x, y) ∧ ¬ killed(x) → food(y)

d. eats (Anil, Peanuts) ∧ alive(Anil).

e. ∀x : eats(Anil, x) → eats(Harry, x)

f. ∀x: ¬ killed(x) → alive(x) ⎤ added predicates.

g. ∀x: alive(x) →¬ killed(x) ⎦

h. likes(John, Peanuts)

Step-2: Conversion of FOL into CNF

In First order logic resolution, it is required to convert the FOL into CNF as CNF form makes easier for resolution proofs.

- o **Eliminate all implication (→) and rewrite**

    1. ∀x ¬ food(x) V likes(John, x)

    2. food(Apple) ∧ food(vegetables)

    3. ∀x ∀y ¬ [eats(x, y) ∧ ¬ killed(x)] V food(y)

    4. eats (Anil, Peanuts) ∧ alive(Anil)

    5. ∀x ¬ eats(Anil, x) V eats(Harry, x)

    6. ∀x¬ [¬ killed(x)] V alive(x)

    7. ∀x ¬ alive(x) V ¬ killed(x)

    8. likes(John, Peanuts).

- o **Move negation (¬)inwards and rewrite**

    9. ∀x ¬ food(x) V likes(John, x)

    10.   food(Apple) ∧ food(vegetables)

    11.   ∀x ∀y ¬ eats(x, y) V killed(x) V food(y)

    12.   eats (Anil, Peanuts) ∧ alive(Anil)

    13.   ∀x ¬ eats(Anil, x) V eats(Harry, x)

    14.   ∀x ¬killed(x)] V alive(x)

    15.   ∀x ¬ alive(x) V ¬ killed(x)

    16.   likes(John, Peanuts).

- o **Rename variables or standardize variables**

    1. ∀x ¬ food(x) V likes(John, x)

    2. food(Apple) ∧ food(vegetables)

    3. ∀y ∀z ¬ eats(y, z) V killed(y) V food(z)

    4. eats (Anil, Peanuts) ∧ alive(Anil)

5. ∀w¬ eats(Anil, w) V eats(Harry, w)

6. ∀g ¬killed(g)] V alive(g)

7. ∀k ¬ alive(k) V ¬ killed(k)

8. likes(John, Peanuts).

- o **Eliminate existential instantiation quantifier by elimination.**
  In this step, we will eliminate existential quantifier ∃, and this process is known as **Skolemization**. But in this example problem since there is no existential quantifier so all the statements will remain same in this step.
- o **Drop Universal quantifiers.**
  In this step we will drop all universal quantifier since all the statements are not implicitly quantified so we don't need it.

1. ¬ food(x) V likes(John, x)

2. food(Apple)

3. food(vegetables)

4. ¬ eats(y, z) V killed(y) V food(z)

5. eats (Anil, Peanuts)

6. alive(Anil)

7. ¬ eats(Anil, w) V eats(Harry, w)

8. killed(g) V alive(g)

9. ¬ alive(k) V ¬ killed(k)

10. likes(John, Peanuts).

Note: Statements "food(Apple) ∧ food(vegetables)" and "eats (Anil, Peanuts) ∧ alive(Anil)" can be written in two separate statements.

Distribute conjunction ∧ over disjunction ¬.
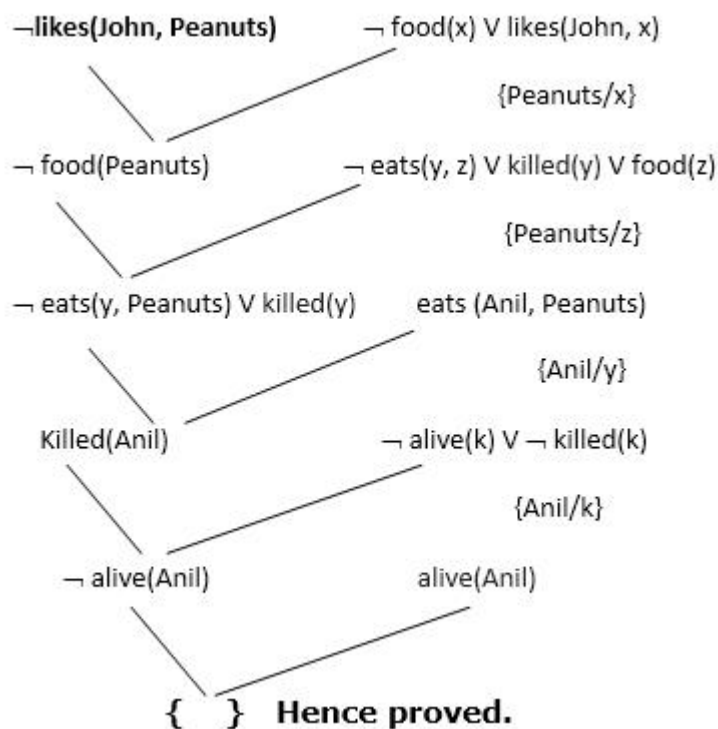This step will not make any change in this problem.

Step-3: Negate the statement to be proved

In this statement, we will apply negation to the conclusion statements, which will be written as ¬likes(John, Peanuts)

Step-4: Draw Resolution graph:

Now in this step, we will solve the problem by resolution tree using substitution. For the above problem, it will be given as follows:

−likes(John, Peanuts)　　　¬ food(x) V likes(John, x)

{Peanuts/x}

¬ food(Peanuts)　　　¬ eats(y, z) V killed(y) V food(z)

{Peanuts/z}

¬ eats(y, Peanuts) V killed(y)　　　eats (Anil, Peanuts)

{Anil/y}

Killed(Anil)　　　¬ alive(k) V ¬ killed(k)

{Anil/k}

¬ alive(Anil)　　　alive(Anil)

{　} **Hence proved.**

Hence the negation of the conclusion has been proved as a complete contradiction with the given set of statements.

Inference engine:

The inference engine is the component of the intelligent system in artificial intelligence, which applies logical rules to the knowledge base to infer new information from known facts. The first inference engine was part of the expert system. Inference engine commonly proceeds in two modes, which are:

a. **Forward chaining**

b. **Backward chaining**

Horn Clause and Definite clause:

Horn clause and definite clause are the forms of sentences, which enables knowledge base to use a more restricted and efficient inference algorithm. Logical inference algorithms use forward and backward chaining approaches, which require KB in the form of the **first-order definite clause**.

Definite clause: A clause which is a disjunction of literals with **exactly one positive literal** is known as a definite clause or strict horn clause.

Horn clause: A clause which is a disjunction of literals with **at most one positive literal** is known as horn clause. Hence all the definite clauses are horn clauses.

Example: (¬ p V ¬ q V k). It has only one positive literal k.

It is equivalent to $p \wedge q \rightarrow k$.

A. Forward Chaining

Forward chaining is also known as a forward deduction or forward reasoning method when using an inference engine. Forward chaining is a form of reasoning which start with atomic sentences in the knowledge base and applies inference rules (Modus Ponens) in the forward direction to extract more data until a goal is reached.

The Forward-chaining algorithm starts from known facts, triggers all rules whose premises are satisfied, and add their conclusion to the known facts. This process repeats until the problem is solved.

Properties of Forward-Chaining:

○ It is a down-up approach, as it moves from bottom to top.

○ It is a process of making a conclusion based on known facts or data, by starting from the initial state and reaches the goal state.

○ Forward-chaining approach is also called as data-driven as we reach to the goal using available data.

○ Forward -chaining approach is commonly used in the expert system, such as CLIPS, business, and production rule systems.

Consider the following famous example which we will use in both approaches:

Example:

"As per the law, it is a crime for an American to sell weapons to hostile nations. Country A, an enemy of America, has some missiles, and all the missiles were sold to it by Robert, who is an American citizen."
Prove that **"Robert is criminal."**

To solve the above problem, first, we will convert all the above facts into first-order definite clauses, and then we will use a forward-chaining algorithm to reach the goal.

B. Backward Chaining:

Backward-chaining is also known as a backward deduction or backward reasoning method when using an inference engine. A backward chaining algorithm is a form of reasoning, which starts with the goal and works backward, chaining through rules to find known facts that support the goal.

Properties of backward chaining:

- o   It is known as a top-down approach.

- o   Backward-chaining is based on modus ponens inference rule.

- o   In backward chaining, the goal is broken into sub-goal or sub-goals to prove the facts true.

- o   It is called a goal-driven approach, as a list of goals decides which rules are selected and used.

- o   Backward -chaining algorithm is used in game theory, automated theorem proving tools, inference engines, proof assistants, and various AI applications.

- o   The backward-chaining method mostly used a **depth-first search** strategy for proof.

Example:

In backward-chaining, we will use the same above example, and will rewrite all the rules.

- o   **American (p) $\wedge$ weapon(q) $\wedge$ sells (p, q, r) $\wedge$ hostile(r) → Criminal(p) ...(1)**
  Owns(A, T1)   ....(2)

- o   **Missile(T1)**

- o   **?p Missiles(p) $\wedge$ Owns (A, p) → Sells (Robert, p, A)   ......(4)**

- o   **Missile(p) → Weapons (p)   ....(5)**

- o   **Enemy(p, America) →Hostile(p)   ....(6)**

- o   **Enemy (A, America)   ....(7)**

- o   **American(Robert).   ....(8)**

Reasoning:

The reasoning is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. Or we can say, "**Reasoning is a way to infer facts from existing data**." It is a general process of thinking rationally, to find valid conclusions.

In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

Types of Reasoning

In artificial intelligence, reasoning can be divided into the following categories:

- o   Deductive reasoning

- o   Inductive reasoning

- o   Abductive reasoning

- o   Common Sense Reasoning

- o   Monotonic Reasoning

- o   Non-monotonic Reasoning

Probabilistic reasoning in Artificial intelligence

Uncertainty:

Till now, we have learned knowledge representation using first-order logic and propositional logic with certainty, which means we were sure about the predicates. With this knowledge representation, we might write A→B, which means if A is true then B is true, but consider a situation where we are not sure about whether A is true or not then we cannot express this statement, this situation is called uncertainty.

So to represent uncertain knowledge, where we are not sure about the predicates, we need uncertain reasoning or probabilistic reasoning.

Causes of uncertainty:

Following are some leading causes of uncertainty to occur in the real world.

1. Information occurred from unreliable sources.

2. Experimental Errors

3. Equipment fault

4. Temperature variation

5. Climate change.

Probabilistic reasoning:

Probabilistic reasoning is a way of knowledge representation where we apply the concept of probability to indicate the uncertainty in knowledge. In probabilistic reasoning, we combine probability theory with logic to handle the uncertainty.

We use probability in probabilistic reasoning because it provides a way to handle the uncertainty that is the result of someone's laziness and ignorance.

In the real world, there are lots of scenarios, where the certainty of something is not confirmed, such as "It will rain today," "behavior of someone for some situations," "A match between two teams or two players." These are probable sentences for which we can assume that it will happen but not sure about it, so here we use probabilistic reasoning.

Need of probabilistic reasoning in AI:

o   When there are unpredictable outcomes.

o   When specifications or possibilities of predicates becomes too large to handle.

o   When an unknown error occurs during an experiment.

In probabilistic reasoning, there are two ways to solve problems with uncertain knowledge:

o   **Bayes' rule**

o   **Bayesian Statistics**

Bayes' theorem in Artificial intelligence

Bayes' theorem:

Bayes' theorem is also known as **Bayes' rule, Bayes' law**, or **Bayesian reasoning**, which determines the probability of an event with uncertain knowledge.

In probability theory, it relates the conditional probability and marginal probabilities of two random events.

Bayes' theorem was named after the British mathematician **Thomas Bayes**. The **Bayesian inference** is an application of Bayes' theorem, which is fundamental to Bayesian statistics.

It is a way to calculate the value of P(B|A) with the knowledge of P(A|B).

Bayes' theorem allows updating the probability prediction of an event by observing new information of the real world.

Bayesian Belief Network in artificial intelligence

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

- o **Directed Acyclic Graph**
- o **Table of conditional probabilities.**

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:
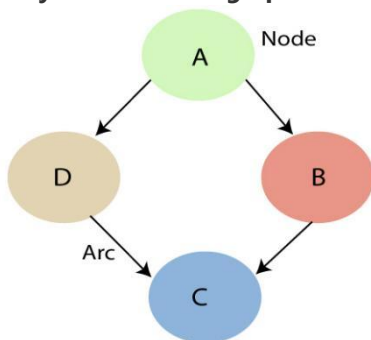
- o Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
- o **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.

  These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other

  - o **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
  - o **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
  - o **Node C is independent of node A.**

Note: The Bayesian network graph does not contain any cyclic graph. Hence, it is known as a directed acyclic graph or DAG.