AI – ANSWERS

Assignment - 1

Q.1] Explain Importance Of AI & List out problems Of AI ?

Importance of Artificial Intelligence:

• Machine learning is the area of artificial intelligence where machines are responsible for completing daily tasks and are believed to be smarter than humans.

• They are known to learn, adapt and perform much faster than humans and are programmed to do so.

• Robotics and integration with IoT devices have taken machines to think and work to a new level where they out-perform humans in their cognitive abilities and smarts.

• Artificial Intelligence's importance and subsequent components have been known for a long time. They are being seen as tools and techniques to make this world better.

• Its importance lies in making our life easier. These technologies are a great asset to humans and are programmed to minimize human effort as much as possible.

-They can operate in an automated fashion. Therefore, manual intervention is the last thing that can be sought or seen during the operation of parts involving this technology.

• These machines speed up your tasks and processes with guaranteed accuracy and precision, making them a useful and valuable tool.

-Apart from making the world an error-free place with their simple and everyday techniques, these technologies and applications are not only related to our ordinary and everyday life.

- It is affecting and holds importance for other domains as well.

AI Problems -

• Intelligence does not imply perfect understanding; every intelligent being has limited perception, memory and computation.

• Many points on the spectrum of intelligence versus cost are viable, from insects to humans.

• AI seeks to understand the computations required from intelligent behavior and to produce computer systems that exhibit intelligence.

• Aspects of intelligence studied by AI include perception, communicational using human languages, reasoning, planning, learning and memory.

• What are the underlying assumptions about intelligence?

• What kinds of techniques will be useful for solving AI problems?

• At what level human intelligence can be modelled?

• When will it be realized when an intelligent program has been built?

Q.2] Discuss Different Application Of Ai ?

1] AI in Astronomy –

- Artificial Intelligence can be very useful to solve complex universe problems.

- AI technology can be helpful for understanding the universe such as how it works, origin, etc.

2] AI in Healthcare –

• In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry.

• Healthcare Industries are applying AI to make a better and faster diagnosis than humans.

-AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization.

3] AI in Gaming –

• AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.

4] AI in Finance –

• AI and finance industries are the best matches for each other.

-The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

5] AI in Data Security –

• The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world.

-AI can be used to make your data more safe and secure.

- Some examples such as AEG bot, AI2 Platform, are used to determine software bug and cyber-attacks in a better way.

6] AI in Social Media –

• Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way.

-AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

7] AI in Travel & Transport –

• AI is becoming highly demanding for travel industries.

• AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers.

• Travel industries are using AI-powered chatbots which can make humanlike interaction with customers for better and fast response.

8] AI in Automotive Industry –

• Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.

• Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

9] AI in Robotics –

• Artificial Intelligence has a remarkable role in Robotics.

Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI, we can create intelligent robots which can perform tasks with their own experiences without pre-programmed.

• Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

10] AI in Entertainment –

• We are currently using some AI based applications in our daily life with some entertainment services such as Netflix or Amazon. With the help of ML/AI algorithms, these services show the recommendations for programs or shows.

11] AI in Agriculture –

• Agriculture is an area which requires various resources, labor, money, and time for best result.

- Now a day's agriculture is becoming digital, and AI is emerging in this field.

- Agriculture is applying AI as agriculture robotics, solid and crop monitoring, predictive analysis.

• AI in agriculture can be very helpful for farmers.

Q.3] Describe State Space Representation using Diagram?

State space representation –

• A state space is a way to mathematically represent a problem by defining all the possible states in which the problem can be

• This is used in search algorithms to represent the initial state, goal state, and current state of the problem.

• Each state in the state space is represented using a set of variables.

Features of State Space Search –

• State space search has several features that make it an effective problemsolving technique in Artificial Intelligence.

Exhaustiveness –

-State space search explores all possible states of a problem to find a solution.

Completeness –

- If a solution exists, state space search will find it.

Optimality –

-Searching through a state space results in an optimal solution.

Uninformed and Informed Search –

-State space search in artificial intelligence can be classified as uninformed if it provides additional information about the problem.

- In contrast, informed search uses additional information, such as heuristics, to guide the search process.

• State space Representation involves defining an INITIAL STATE and a GOAL STATE and then determining a sequence of actions, called states, to follow.

• **State –**

-A state can be an Initial State, a Goal State, or any other possible state that can be generated by applying rules between them.

• **Space –**

-In an AI problem, space refers to the exhaustive collection of all conceivable states.

Search -

-This technique moves from the beginning state to the desired state by applying good rules while traversing the space of all possible states.

• **Search Tree –**

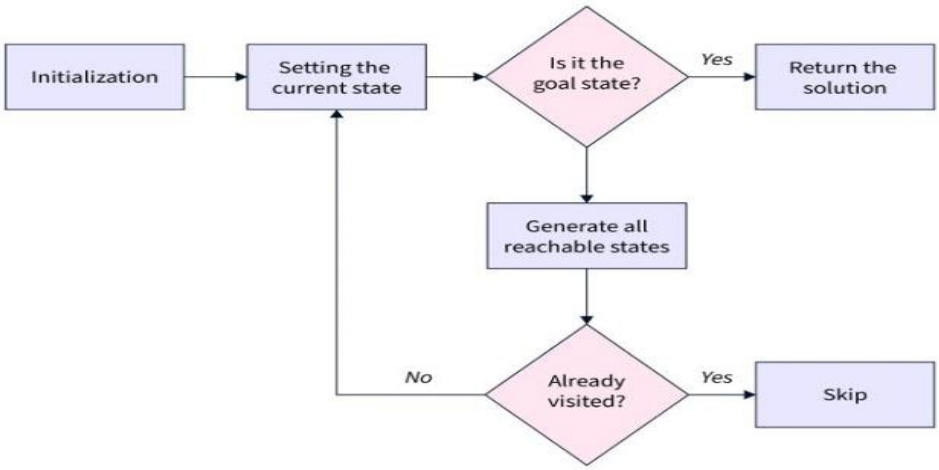
-To visualize the search issue, a search tree is used, which is a tree-like structure that represents the problem.

- The initial state is represented by the root node of the search tree, which is the starting point of the tree.

• **Transition Model –**

-This describes what each action does, while Path Cost assigns a cost value to each path, an activity sequence that connects the beginning node to the end node.

- The optimal option has the lowest cost among all alternatives.



Q.4] Explain & State Problems reduction Representation ?

- Problem reduction is a technique used in artificial intelligence to simplify complex problems by breaking them down into smaller, more manageable sub-problems.

• This approach is particularly useful when dealing with problems that are difficult to solve directly or require a significant amount of computational resources.

• In problem reduction, the main problem is divided into smaller subproblems, known as subgoals, that can be tackled individually.

• This technique is often used in problem-solving algorithms, search algorithms, and task planning.

THE PROCESS OF PROBLEM REDUCTION: The problem reduction approach involves the following steps:

• Identifying the main problem –

-The first step in problem reduction is to identify the main problem that needs to be solved.

• Breaking down the problem –

-Once the main problem is identified, it is broken down into smaller sub-problems.

- This is done by identifying the key components or factors that contribute to the main problem.

• **Solving the sub-problems –**

-Each sub-problem is solved individually using appropriate techniques or algorithms.

• **Combining the solutions –**

-Finally, the solutions to the sub-problems are combined to obtain the solution to the main problem.

BENEFITS OF PROBLEM REDUCTION: The problem reduction approach offers several benefits in artificial intelligence:

• **Complex problems become more manageable –**

-By breaking down a complex problem into smaller sub-problems, AI systems can focus their resources and efforts on solving each sub-problem individually, making the overall problem more manageable.

ADVANTAGES OF PROBLEM REDUCTION

• **EFFICIENCY** –

-By breaking down a complex problem into smaller and more manageable sub problems, the overall computational complexity is reduced.

- This allows AI systems to process and solve problems faster, saving time and computational resources.

• **SCALABILITY** –

-Problem reduction allows AI systems to scale and handle larger and more complex problem domains.

- By decomposing a problem into smaller components, it becomes easier to analyze and solve each component independently.

- This scalability is particularly beneficial in domains with vast amounts of data or complex decision-making processes.

• **MODULARITY –**

-Problem reduction promotes modularity by breaking down a problem into smaller modules or sub problems.

- This modular structure enables developers to work on individual components separately, making the development process more manageable and promoting code reusability.

-Modularity also improves system reliability, as errors in one module are less likely to affect the overall system.

• **FLEXIBILITY –**

-Another advantage of problem reduction is its flexibility.

-AI systems using this approach can handle different problem types and adapt to various problem-solving techniques.

- This flexibility allows developers to apply problem reduction in a wide range of scenarios and domains, making it a versatile technique in artificial intelligence.

Examples of problem reduction in AI -

1] Image recognition –

- To identify objects in an image, the problem can be reduced by segmenting the image into regions, extracting features from each region, and classifying them.

- By breaking down the problem into these smaller steps, AI models can accurately recognize objects and perform tasks like object detection or image categorization.

Q.5] Explain Production System ?

PRODUCTION SYSTEMS -

• Production systems provide appropriate structures for performing and describing search processes.

A production system has four basic components as

• A set of rules each consisting of a left side that determines the applicability of the rule and a right side that describes the operation to be performed if the rule is applied.

• A database of current facts established during the process of inference.

• A control strategy that specifies the order in which the rules will be compared with facts in the database and also specifies how to resolve conflicts in selection of several rules or selection of more facts.

• A rule firing module.

• The production rules operate on the knowledge database.

-Each rule has a precondition—that is, either satisfied or not by the knowledge database.

• If the precondition is satisfied, the rule can be applied

• Application of the rule changes the knowledge database.

-The control system chooses which applicable rule should be applied and ceases computation when a termination condition on the knowledge database is satisfied.

Assignment - 2

Q.1] Explain Brute Force search algorithm With suitable example ?

Brute Force Search –

-In AI, brute-force search is a method of problem solving in which all possible solutions are systematically checked for correctness.

- It is also known as exhaustive search or complete search.

- Brute-force is considered an exhaustive search and is not a quick option, but rather serves as the heuristic alternative to more sophisticated approaches.

Example:

• **Closest Pair -**

Problem statement - To find out the two closest points in a set of n points in the two-dimensional cartesian plane.

-There is n number of scenarios where this problem arises.

- A real-life example would be in an air traffic control system where you have to monitor the planes flying near each other and find out the safest minimum distance these planes should maintain.

• The brute force algorithm computes the distance between every distinct set of points and returns the point's indexes for which the space is the smallest.

• Brute Force solves this problem with the time complexity of [O(n2)], where n is the number of points.

• the pseudo-code uses the brute force algorithm to find the closest point.

Algorithm BruthForceClosestPonints(P)

Algorithm BruteForceClosestPoints(P)
    // P is list of points
    $dmin \leftarrow \infty$
    for $i \leftarrow 1$ **to** n-1 **do**
        for $j \leftarrow i$+1 **to** n **do**
            $d \leftarrow \text{sqrt}((x_i-x_j)^2 + (y_i-y_j)^2)$
            if $d < dmin$ **then**
                $dmin \leftarrow d$; $index1 \leftarrow i$; $index2 \leftarrow j$
    return $index1$, $index2$

Q.2] Explain Depth First search technique With suitable example ?

• search all the vertices of a tree data structure or a graph.

• The depth-first search (DFS) algorithm starts with the initial node of graph G and goes deeper until we find the goal node or the node with no children.

The process of implementing the DFS is similar to the BFS algorithm:

• The step by step process to implement the DFS traversal is given as follows –

• First, create a stack with the total number of vertices in the graph.

• Now, choose any vertex as the starting point of traversal, and push that vertex into the stack.

• After that, push a non-visited vertex (adjacent to the vertex on the top of the stack) to the top of the stack

. • Now, repeat steps 3 and 4 until no vertices are left to visit from the vertex on the stack's top.

• If no vertex is left, go back and pop a vertex from the stack. • Repeat steps 2, 3, and 4 until the stack is empty.

Algorithm:

• Step 1: SET STATUS = 1 (ready state) for each node in G

• Step 2: Push the starting node A on the stack and set its STATUS = 2 (waiting state)

• Step 3: Repeat Steps 4 and 5 until STACK is empty

• Step 4: Pop the top node N. Process it and set its STATUS = 3 (processed state)

• Step 5: Push on the stack all the neighbors of N that are in the ready state (whose STATUS = 1) and set their STATUS = 2 (waiting state)

• Step 6: EXIT

```
DFS(G,v)   ( v is the vertex where the search starts )
   Stack S := {};   ( start with an empty stack )
   for each vertex u, set visited[u] := false;
   push S, v;
   while (S is not empty) do
     u := pop S;
     if (not visited[u]) then
       visited[u] := true;
       for each unvisited neighbour w of uu
         push S, w;
     end if
   end while
 END DFS()
```

Q.3] Explain breadth First search technique With suitable example ?

breadth-first search -

- explores all the vertices in a graph at the current depth before moving on to the vertices at the next depth level.
- It starts at a specified vertex and visits all its neighbors before moving on to the next level of neighbors.
- BFS is commonly used in algorithms for pathfinding, connected components, and shortest path problems in graphs.
- **Algorithm for the BFS -**
- **Initialization**:

-Enqueue the starting node into a queue and mark it as visited.(Enqueue: Adds an item from the back of the queue.)

- **Exploration:** While the queue is not empty:

o Dequeue a node from the queue and visit it (e.g., print its value).(Dequeue: Removes an item from the front of the queue.)

o For each unvisited neighbor of the dequeued node:

♣ Enqueue the neighbor into the queue.

♣ Mark the neighbor as visited.

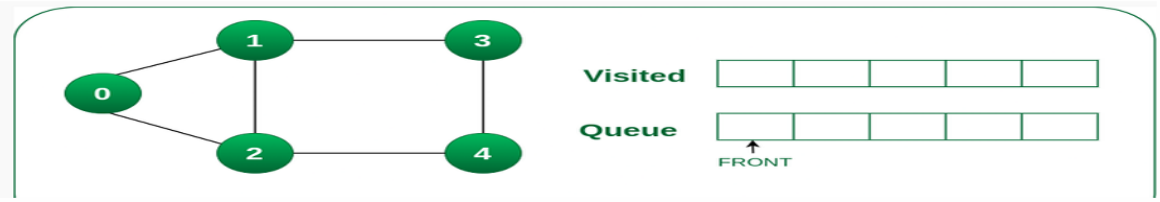
How Does the BFS Algorithm Work?

• Starting from the root, all the nodes at a particular level are visited first and then the nodes of the next level are traversed till all the nodes are visited.

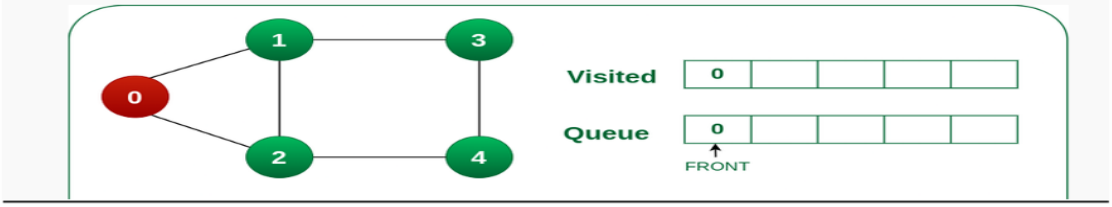
• To do this a queue is used. All the adjacent unvisited nodes of the current level are pushed into the queue and the nodes of the current level are marked visited and popped from the queue.

• **Example:**

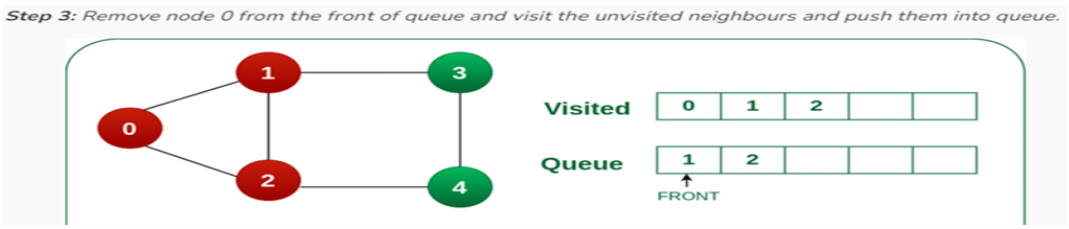
step1*:* Initially queue and visited arrays are empty

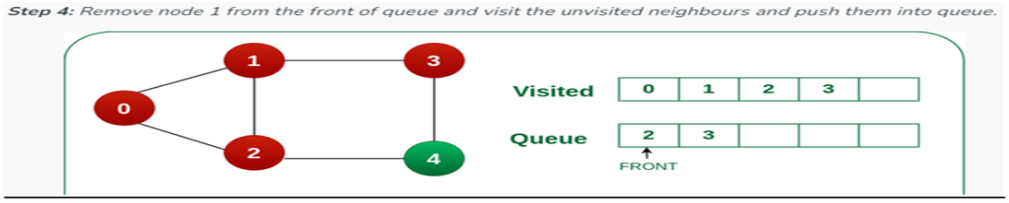


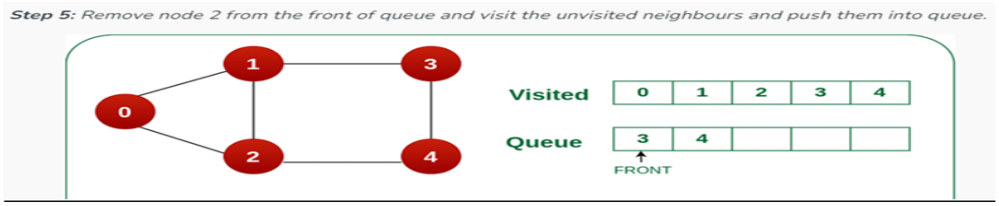
Step2: Push node 0 into queue and mark it visited.

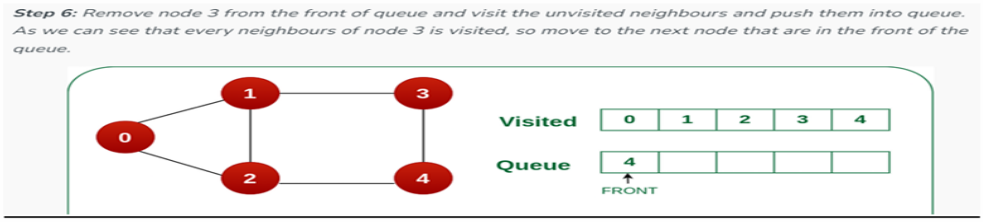


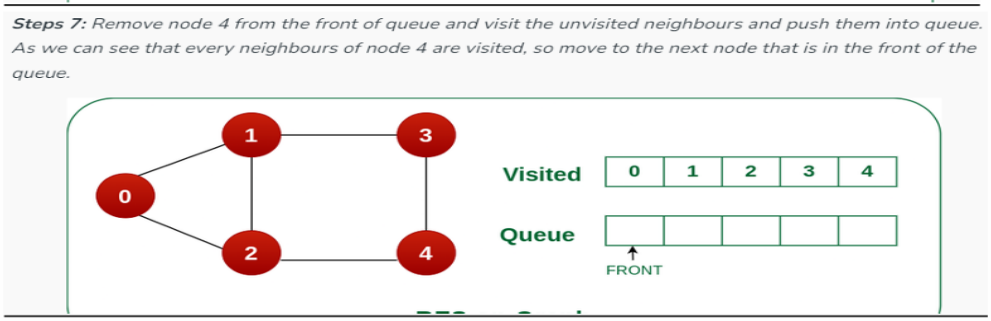
Step 3: Remove node 0 from the front of queue and visit the unvisited neighbours and push them into queue.











Q.4] Explain hill climbing algorithm ?

• Hill climbing is a simple optimization algorithm used in Artificial Intelligence (AI) to find the best possible solution for a given problem.

• It belongs to the family of local search algorithms and is often used in optimization problems where the goal is to find the best solution from a set of possible solutions.

• Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem

• It terminates when it reaches a peak value where no neighbor has a higher value.

• Hill climbing algorithm is a technique which is used for optimizing the mathematical problems.

Features of Hill Climbing -

Following are some main features of Hill Climbing Algorithm:

• **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.

• **Greedy approach**: Hill-climbing algorithm search moves in the direction which optimizes the cost.

• **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

Types of Hill Climbing Algorithm:

• **Simple hill Climbing –**

-It only evaluates the neighbour node state at a time and selects the first one which optimizes current cost and set it as a current state.

- It only checks it's one successor state, and if it finds better than the current state, then move else be in the same state.

• **Steepest-Ascent hill-climbing –**

-The steepest-Ascent algorithm is a variation of simple hill climbing algorithm.

- This algorithm examines all the neighboring nodes of the current state and selects one neighbor node which is closest to the goal state.

- This algorithm consumes more time as it searches for multiple neighbors.

• Stochastic hill Climbing:

• Stochastic hill climbing does not examine for all its neighbor before moving.

• Rather, this search algorithm selects one neighbor node at random and decides whether to choose it as a current state or examine another state.

Q.5] Explain A* Algorithm & beam search Algorithm With suitable example ?

A* Algorithm -

• The A* algorithm or A star algorithm in AI is a powerful pathfinding algorithm that efficiently finds the shortest path in a graph while considering both the actual cost incurred so far and an estimate of the remaining cost.

• **Here are core components of A* algorithm in AI with example -**

1] Open Set –

- The open set is a collection of nodes that are candidates for evaluation. Initially, it contains only the starting node.

-As the algorithm progresses, nodes are added or removed from the open set based on their estimated total cost (usually denoted as "f-score").

- The node with the lowest f-score is selected for evaluation next.

2] Closed Set –

- The closed set contains nodes that have already been evaluated.

- Once a node is evaluated, it is moved from the open set to the closed set.

- This prevents revisiting nodes and ensures that the algorithm explores the graph efficiently.

3] Cost Function –

- The A* algorithm uses a cost function that assigns a cost (often referred to as "g(n)") to each node based on the cost of reaching that node from the starting point.

- Additionally, it calculates a heuristic cost estimate (often referred to as "h(n)") from that node to the goal node.

- The f-score of a node is the sum of its actual cost (g(n)) and the estimated cost to reach the goal (h(n)).

- The node with the lowest f-score is prioritized for evaluation.

• **beam search :**

• Optimized version of Best first search

• Heuristic search algorithm

• Explores a graph by expanding most promising node in a limited set

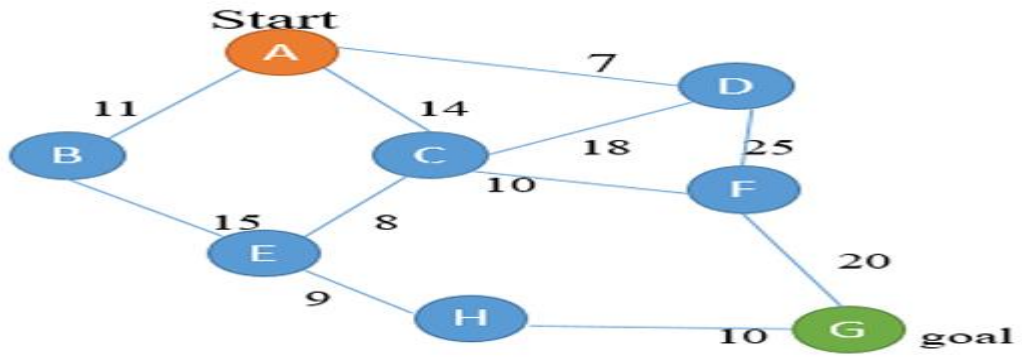
• reduces memory requirement.

• greedy approach

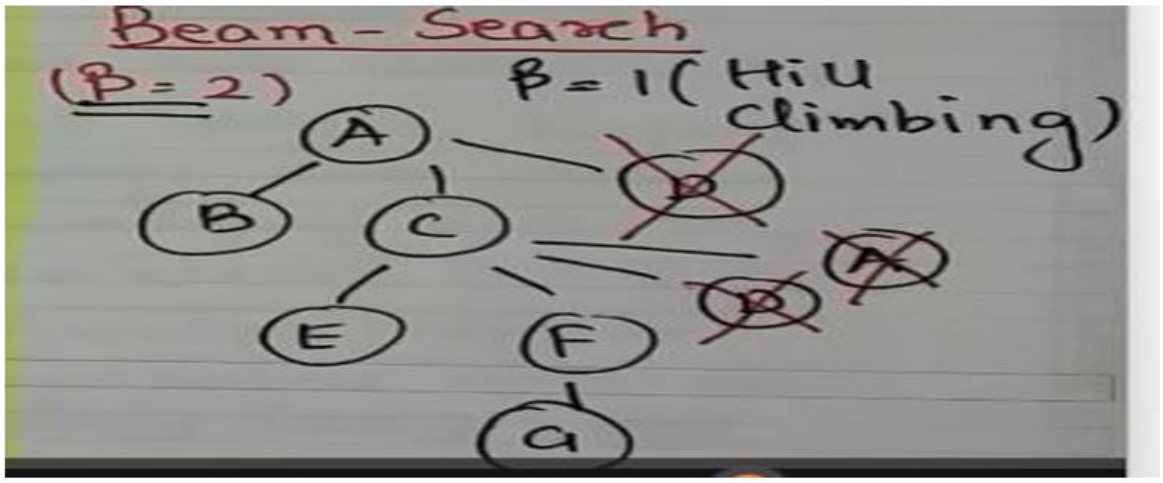
Components of Beam Search

- A beam search takes three components as its input:
- 1. The problem usually represented as graph and contains a set of nodes in which one or more of the nodes represents a goal.
- 2. The set of heuristic rules for pruning: are rules specific to the problem domain and prune unfavorable nodes from memory regarding the problem domain.
- 3. A memory with a limited available capacity
- The memory is where the ***"beam"*** is stored, memory is full, and a node is to be added to the beam, the most costly node will be deleted, such that the memory limit is not exceeded.

Example:



Heuristic values: A- G=40 B- G=32 C- G=25 D- G=35 E- G=19 F- G=17 H- G=10 G- G=0



Assignment - 3

Q.1] Discuss knowledge representation issues ?

Knowledge representation issues -

• 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:

-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.

- 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.

- 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.

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

• **Object**: All the facts about objects in our world domain. E.g., Guitars contains strings, trumpets are brass instruments.

• **Events**: Events are the actions which occur in our world.

• **Performance**: It describe behavior which involves knowledge about how to do things.

• **Meta-knowledge**: It is knowledge about what we know.

• **Facts**: Facts are the truths about the real world and what we represent.

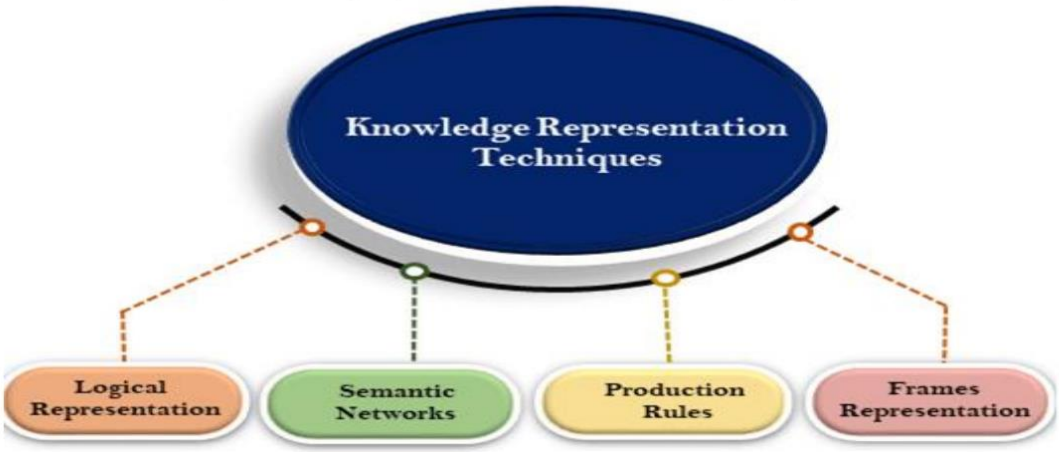
• **Knowledge-Base**: The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the Sentences (Here, sentences are used as a technical term and not identical with the English language).

• Knowledge: Knowledge is awareness or familiarity gained by experiences of facts, data, and situations.

Q.2] Explain knowledge representation mechanism ?

- **knowledge representation mechanism:**

There are mainly four ways (techniques) of knowledge representation:



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

Advantages of logical representation:

• Logical representation enables us to do logical reasoning.

• Logical representation is the basis for the programming languages.

Disadvantages of logical Representation:

• Logical representations have some restrictions and are challenging to work with

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

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. T

• his 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.

Advantages of Semantic network:

• Semantic networks are a natural representation of knowledge.

• Semantic networks convey meaning in a transparent manner.

• 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.

Production Rules:

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

• The set of production rules

• Working Memory

• 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.

Advantages of Production rule:

• The production rules are expressed in natural language.

• The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages of Production rule:

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

Q.3] Explain Rules For Knowledge representation ?

Rules for Knowledge Representation:

• One way to represent knowledge is by using rules that express what must happen or what does happen when certain conditions are met.

• Rules are usually expressed in the form of IF . . . THEN . . . statements, such as: IF A THEN B This can be considered to have a similar logical meaning as the following: A→B

• A is called the antecedent and B is the consequent in this statement.

• In expressing rules, the consequent usually takes the form of an action or a conclusion.

• In other words, the purpose of a rule is usually to tell a system (such as an expert system) what to do in certain circumstances, or what conclusions to draw from a set of inputs about the current situation.

• In general, a rule can have more than one antecedent, usually combined either by AND or by OR (logically the same as the operators ∧ and ∨).

• Similarly, a rule may have more than one consequent, which usually suggests that there are multiple actions to be taken.

• In general, the antecedent of a rule compares an object with a possible value, using an operator.

• For example, suitable antecedents in a rule might be

• IF x > 3

• IF name is “Bob”

• IF weather is cold

• Here, the objects being considered are x, name, and weather; the operators are “>” and “is”, and the values are 3, “Bob,” and cold.

Q.4] Write short note on a horn clause ?

A Horn clause:

• is a clause (a disjunction of literals) with at most one positive, i.e.

unnegated, literal.

• A clause with at most one positive (unnegated) literal is called a Horn

Clause.

Types of Horn Clauses :

• **Definite clause** / Strict Horn clause : Has exactly one positive literal.

• **Unit clause** : Definite clause with no negative literals.

.**Goal clause** : Horn clause without a positive literal.

• Horn Clausal form :

(i) The Horn clause,

NOT(P1) OR NOT(P2) OR ... OR NOT(Pn) OR Q

• Can be transformed into the clause :

P1 AND P2 AND … AND Pn => Q

• which is written in Datalog as the following rules.

Q :- P1, P2, ..., Pn.

• The above Datalog Rule, is hence a Horn clause.

• Meaning : If the predicates P1 AND P2 AND … AND Pn, are all true for a particular binding to their variable arguments, then Q is also true and can hence be inferred.

(ii) The Horn clause,

NOT(P1) OR NOT(P2) OR ... OR NOT (Pn)

• Can be transformed into

P1 AND P2 AND … AND Pn =>

• which is written in Datalog as follows :

P1, P2, ..., Pn.

• The above Datalog expression can be considered as an integrity constraint, where all the predicates must be true to satisfy the query.

• Horn Clauses:

• The definite clause language does not allow a contradiction to be stated.

However, a simple expansion of the language can allow proof by contradiction.

• An integrity constraint is a clause of the form

• false ←a 1 ∧...∧ak.

• where the ai are atoms and false is a special atom that is false in all interpretations.

• A Horn clause is either a definite clause or an integrity constraint. That is, a Horn clause has either false or a normal atom as its head.

• Integrity constraints allow the system to prove that some conjunction of atoms is false in all models of a knowledge base - that is, to prove disjunctions of negations of atoms.

Q.5] List type of learning and explain it ?

. Type of learning:

1. Supervised Learning:

• An AI model that identifies patterns from data sets

• Supervised learning is a method of teaching AI that builds off of inductive learning but uses a little less structure.

• It involves providing an AI with labeled training data.

• For example, let's say you give an AI a set of images correctly labeled as either "cat" or "dog".

• By looking at these images, the AI would start to see patterns in the "cat" images that distinguish them from the "dog" images and vice versa.

• Eventually, after it's learned everything it can from these images, the AI would then be able to correctly identify any image of any cat or dog.

• The only drawback to supervised learning is that it requires an existing

dataset.

2. Unsupervised Learning:

• An AI model that develops its own definitions

• That's where unsupervised learning comes into play.

• Unsupervised learning also teaches AI to look for patterns in data, but unlike supervised learning, it doesn't provide training data.

• In an unsupervised model, you would give the AI images of cars, buses, and bicycles, without labeling them as such.

• This way, the AI would have to make deductions about these images and come up with its own categories. When given a new image, the AI will try to classify it into one of these categories, but it will not know what that category represents, such as "car" or "bicycle."

• It would be forced to come up with its own definitions.

3. Semi-Supervised Learning:

 An AI model that identifies connection between data sets

 Semi-supervised learning was created to address a common issue in unsupervised learning, which is aimlessness.

 In theory, unsupervised learning is great, but it often gives the AI too little data to work with, resulting in minimal progress.

 Semi-supervised learning involves giving the AI a small set of labeled data and a larger set of unlabeled data.

4. Reinforcement Learning:

An AI model that relies on trial and error

• Reinforcement learning is a type of machine learning that involves teaching AI through trial and error.

• Take the familiar lab mouse trying to find the cheese at the end of a maze.

• On a first attempt, the mouse may struggle to even make it to the end.

• But each time it's placed in the maze it becomes more and more proficient at the maze, until eventually, it can make consistently perfect runs.

Assignment - 4

Q.1] Explain propositional logic & predicate logic ?

Propositional logic:

• 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:

a) It is Sunday.
b) The Sun rises from West (False proposition)
c) 3+3= 7(False proposition)
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**, and it is also called a valid sentence.

• A proposition formula which is always false is called **Contradiction**.

• A proposition formula which has both true and false values

• Statements which are questions, commands, or opinions are not propositions such as "**Where is XYZ**", "**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:

• **Atomic Propositions**

• **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:**

a) 2+2 is 4, it is an atomic proposition as it is a **true** fact.
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:**

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

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

Predicate Logic:

• Predicate Logic deals with predicates, which are propositions, consist of variables.
Predicate Logic - Definition

• 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.

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."

Q.2] Illustrate Inference rules & various type of inference rule with the help of example ?

inference rules:
• we need intelligent computers which can create new logic from old logic or by evidence, **so generating the conclusions from evidence and facts is termed as Inference**.

Inference rules:
• Inference rules are the templates for generating valid arguments.

• Inference rules are applied to derive proofs in artificial intelligence, and the proof is a sequence of the conclusion that leads to the desired goal.

• In inference rules, the implication among all the connectives plays an important role.

• **Following are some terminologies related to inference rules:**

• **Implication:** It is one of the logical connectives which can be represented as P → Q. It is a Boolean expression.
• **Converse:** The converse of implication, which means the right-hand side proposition goes to the left-hand side and vice-versa. It can be written as Q → P

• **Contrapositive:** The negation of converse is termed as contrapositive, and it can be represented as ¬ Q → ¬ P.

• **Inverse:** The negation of implication is called inverse. It can be represented as ¬ P → ¬ Q.
• 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.

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."

Types of Inference rules:

1] Modus Ponens -

-The Modus Ponens rule is one of the most important rules of inference, and it states that if P and P → Q is true, then we can infer that Q will be true.
It can be represented as:

Example:
Statement-1: "If I am sleepy then I go to bed" ==> P→ Q(implies)
Statement-2: "I am sleepy" ==> P
Conclusion: "I go to bed." ==> Q.
Hence, we can say that, if P→ Q is true and P is true then Q will be true.
Proof by Truth table:

P	Q	$P \rightarrow Q$
0	**0**	**0**
0	**1**	**1**
1	**0**	**0**
1	**1**	**1** ←

2] Modus Tollens –

• The Modus Tollens rule state that if P→ Q is true and **¬ Q is true, then ¬ P** will also true.

• It can be represented as:

Notation for Modus Tollens: $\frac{P \rightarrow Q,\ \sim Q}{\sim P}$

Statement-1: "If I am sleepy then I go to bed" ==> P→ Q **Statement-2:** "I do not go to the bed."==> ~Q **Statement-3:** Which infers that "**I am not sleepy**" => ~P
• Proof by Truth table:

P	Q	$\sim P$	$\sim Q$	$P \rightarrow Q$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1

3] Hypothetical Syllogism –

-The Hypothetical Syllogism rule state that if P→R is true whenever P→Q is true, and Q→R is true.
It can be represented as the following notation:

• **Example:**

• **Statement-1:** If you have my home key then you can unlock my home. **P→Q**
Statement-2: If you can unlock my home then you can take my money. **Q→R**
Conclusion: If you have my home key then you can take my money. **P→R**

Proof by truth table:

P	Q	R	$P \rightarrow Q$	$Q \rightarrow R$	$P \rightarrow R$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	1

4] Disjunctive Syllogism –

• The Disjunctive syllogism rule state that if P∨Q is true, and ¬P is true, then Q will be true.

• It can be represented as:

Notation of Disjunctive syllogism: $\frac{P \vee Q,\ \neg P}{Q}$

• **Example:**

• **Statement-1:** Today is Sunday or Monday. ==>P∨Q **Statement-2:** Today is not Sunday. ==> ¬P **Conclusion:** Today is Monday. ==> Q

Proof by truth-table:

P	Q	$\neg P$	$P \vee Q$
0	0	1	0
0	1	1	1
1	0	0	1
1	1	0	1

5] Addition –

-The Addition rule is one the common inference rule, and it states that If P is true, then PVQ will be true.

Example:

Notation of Addition: $\frac{P}{P \vee Q}$

Statement: I have a vanilla ice-cream. ==> P **Statement-2:** I have Chocolate ice-cream. **Conclusion:** I have vanilla or chocolate ice-cream. ==> (PVQ)
Proof by Truth-Table:

P	Q	$P \vee Q$
0	0	0
1	0	1
0	1	1
1	1	1

6] Simplification –

• The simplification rule state that if **P∧ Q** is true, then **Q or P** will also be true.

• It can be represented as:

Notation of Simplification rule: $\frac{P \wedge Q}{Q}$ Or $\frac{P \wedge Q}{P}$

Proof by Truth-Table:

P	Q	$P \wedge Q$
0	0	0
1	0	0
0	1	0
1	1	1

7] Resolution –

-The Resolution rule state that if PVQ and ¬ P∧R is true, then QVR will also be true.

It can be represented as:

Notation of Resolution $\frac{P \vee Q,\ \neg P \wedge R}{Q \vee R}$

P	¬ P	Q	R	$P \vee Q$	¬ P∧R	$Q \vee R$	
0	1	0	0	0	0	0	
0	1	0	1	0	0	1	
0	1	1	0	1	1	1	←
0	1	1	1	1	1	1	←
1	0	0	0	1	0	0	
1	0	0	1	1	0	1	
1	0	1	0	1	0	1	
1	0	1	1	1	0	1	←

Q.3] Demonstrate first order predicate logic with its syntax and symmetric ?

First-Order Logic –

• 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.

"Some humans are intelligent", or
"Sachin likes cricket."
above statements, PL logic is not sufficient, so we required some more logic, **such as first-order logic.**

First-Order logic -

• First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.

• FOL is sufficiently expressive to represent the natural language statements in a concise way.

• 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.
• 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:

Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus ,.....
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
Function: Father of, best friend, third inning of, end of, ...
• As a natural language, first-order logic also has two main parts:
• **Syntax**

• **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 FOL syntax:**

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

Q.4] illustrate first order predicate logic & Quantifiers first order predicate logic ?

- Let's illustrate first-order predicate logic (FOPL) and quantifiers with a simple example involving animals. Suppose we have a domain of animals, and two predicates: $(x)P(x)$ which denotes "x is a mammal" and $Q(x)Q(x)$ which denotes "x can fly".

We'll use the following symbols -

$\forall\forall$: Universal quantifier (for all)

$\exists\exists$: Existential quantifier (there exists)

$\wedge\wedge$: Conjunction (and)

$\rightarrow\rightarrow$: Implication

Here are some statements we can make using FOPL and quantifiers:

Universal Quantification:

$\forall x(x)\forall xP(x)$: "All animals are mammals."

$\forall(P(x)\rightarrow Q(x))\forall x(P(x)\rightarrow Q(x))$: "If an animal is a mammal, then it can fly."

Existential Quantification:

$\exists x(x)\exists xP(x)$: "There exists an animal that is a mammal."

$\exists(P(x)\wedge Q(x))\exists x(P(x)\wedge Q(x))$: "There exists an animal that is a mammal and can fly."

Combined Quantifiers:

$\forall x\exists(P(x)\wedge Q(y))\forall x\exists y(P(x)\wedge Q(y))$: "For every animal, there exists an animal that is a mammal and can fly."

$\exists x\forall(P(y)\rightarrow Q(x))\exists x\forall y(P(y)\rightarrow Q(x))$: "There exists an animal such that if any animal is a mammal, then that animal can fly."

Assignment - 5

Q.1] Explain concept of reasoning in AI with the help of different types of reasoning ?

- Reasoning in Artificial Intelligence refers to the process by which AI systems analyze information, make inferences, and draw conclusions to solve problems or make decisions.

-It is a fundamental cognitive function that enables machines to mimic human thought processes and exhibit intelligent behavior.

Types of reasoning's –

1] Deductive reasoning –

• Deductive reasoning is deducing new information from logically related known information.

-It is the form of valid reasoning, which means the argument's conclusion must be true when the premises are true.

• Deductive reasoning is a type of propositional logic in AI, and it requires various rules and facts.

• In deductive reasoning, the truth of the premises guarantees the truth of the conclusion.

• Deductive reasoning mostly starts from the general premises to the specific conclusion,

Example: Premise-1: All the human eats veggies Premise-2: Suresh is human.

2] Inductive Reasoning -

• Inductive reasoning is a form of reasoning to arrive at a conclusion using limited sets of facts by the process of generalization.

-It starts with the series of specific facts or data and reaches to a general statement or conclusion.

• Inductive reasoning is a type of propositional logic, which is also known as cause-effect reasoning or bottom-up reasoning.

• In inductive reasoning, we use historical data or various premises to generate a generic rule, for which premises support the conclusion.

• In inductive reasoning, premises provide probable supports to the conclusion, so the truth of premises does not guarantee the truth of the conclusion.

Example: Premise: All of the pigeons we have seen in the zoo are white.

3] Abductive reasoning -

• Abductive reasoning is a form of logical reasoning which starts with single or multiple observations then seeks to find the most likely explanation or conclusion for the observation.

• Abductive reasoning is an extension of deductive reasoning, but in abductive reasoning, the premises do not guarantee the conclusion.

Example: • Implication: Cricket ground is wet if it is raining

4] Common Sense Reasoning -

• Common sense reasoning is an informal form of reasoning, which can be gained through experiences.

• Common Sense reasoning simulates the human ability to make presumptions about events which occurs on every day.

-It relies on good judgment rather than exact logic and operates on heuristic knowledge and heuristic rules.

Example: 1. One person can be at one place at a time.

5] Monotonic Reasoning –

• In monotonic reasoning, once the conclusion is taken, then it will remain the same even if we add some other information to existing information in our knowledge base.

• In monotonic reasoning, adding knowledge does not decrease the set of prepositions that can be derived.

• To solve monotonic problems, we can derive the valid conclusion from the available facts only, and it will not be affected by new facts.

• Monotonic reasoning is not useful for the real-time systems, as in real time, facts get changed, so we cannot use monotonic reasoning.

• Monotonic reasoning is used in conventional reasoning systems, and a logic-based system is monotonic.

Example: • Earth revolves around the Sun.

6] Non-monotonic Reasoning -

• In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base.

• Logic will be said as non-monotonic if some conclusions can be invalidated by adding more knowledge into our knowledge base.

• Non-monotonic reasoning deals with incomplete and uncertain models.
Example: Let suppose the knowledge base contains the following knowledge: • Birds can fly • Penguins cannot fly • Pitty is a bird

Q.2] illustrate the truth maintenance system & its Working in expert system ?

Truth Maintenance System -

• A truth maintenance system is a knowledge representation and reasoning system designed to manage inconsistent and uncertain information.

• It helps maintain a consistent and coherent view of the world by tracking the beliefs and assumptions used to derive conclusions.

• It is used to detect and resolve conflicts between different beliefs and assumptions.

• In other words, it's a way of ensuring that the AI system is aware of all the knowledge it possesses and that this knowledge is valid and up-to-date.

• The TMS works by keeping a record of all the beliefs or facts in the knowledge base, along with the reasoning that led to them.

• When new information is added to the knowledge base, the TMS checks to see if it is consistent with the existing beliefs.

• If it is not, the TMS identifies the conflicting beliefs and tries to resolve the inconsistency by removing or modifying one or more of them. This process is called "belief revision."

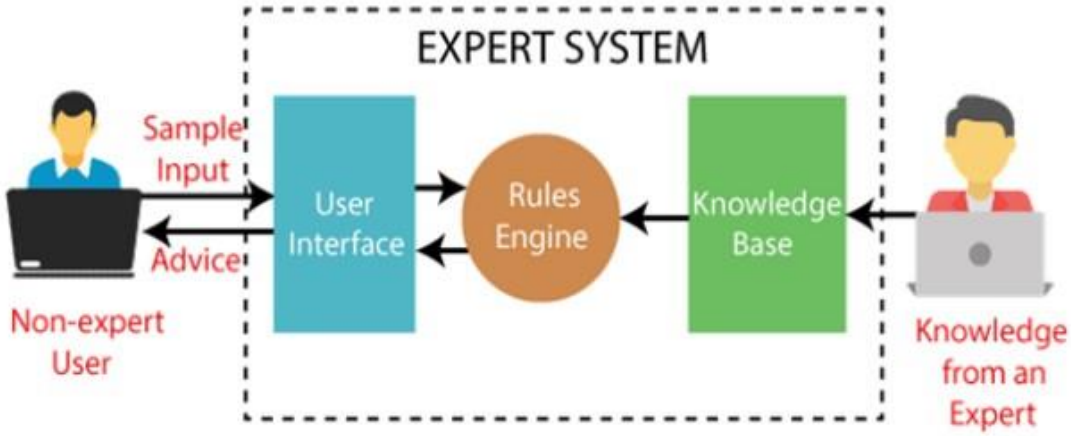
• The TMS also keeps track of the dependencies between different beliefs.

• For example, if one belief is based on another belief, the TMS will ensure that the latter belief is still valid before accepting the former. This helps to maintain the integrity of the knowledge base.

• In practical terms, a TMS is often used in expert systems or decisionmaking systems, where the accuracy and reliability of the information are critical.

• It can also be used in natural language processing, where it helps to ensure that the AI system is interpreting language correctly and consistently.

Working of an expert system –



Q.3] Explain probabilistic reasoning ?

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.

• 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 –

• When there are unpredictable outcomes.

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

• When an unknown error occurs during an experiment.

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

• Bayes' rule

• Bayesian Statistics

- before understanding probabilistic reasoning, let's understand some common terms:

• **Probability**

-Probability can be defined as a chance that an uncertain event will occur.

- It is the numerical measure of the likelihood that an event will occur.

- The value of probability always remains between 0 and 1 that represent ideal uncertainties.

$0 \leq P(A) \leq 1$, where P(A) is the probability of an event A.

P(A) = 0, indicates total uncertainty in an event A.

P(A) =1, indicates total certainty in an event A.

We can find the probability of an uncertain event by using the below formula.

P(¬A) = probability of a not happening event. P(¬A) + P(A) = 1.

• **Event**: Each possible outcome of a variable is called an event.

• **Sample space**: The collection of all possible events is called sample space.

• **Random variables**: Random variables are used to represent the events and objects in the real world.

• **Prior probability**: The prior probability of an event is probability computed before observing new information.

• **Posterior Probability**: The probability that is calculated after all evidence or information has taken into account. It is a combination of prior probability and new information.

Q.4] Differences between DFS & BFS ?

Difference Between BFS and DFS:

Parameters	BFS	DFS
Stands for	BFS stands for Breadth First Search.	DFS stands for Depth First Search.

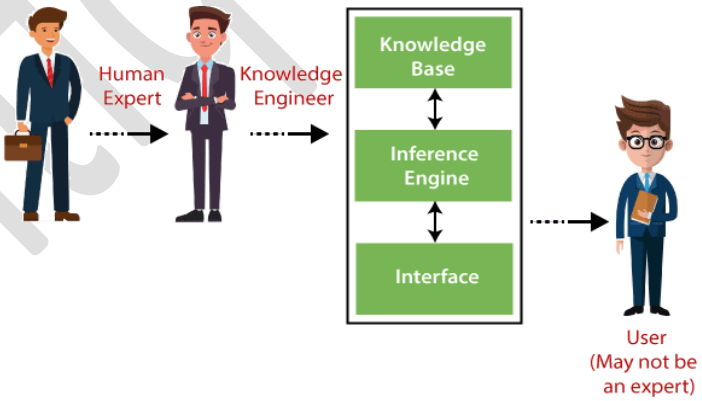
Parameters	BFS	DFS
Data Structure	BFS(Breadth First Search) uses Queue data structure for finding the shortest path.	DFS(Depth First Search) uses Stack data structure.
Definition	BFS is a traversal approach in which we first walk through all nodes on the same level before moving on to the next level.	DFS is also a traversal approach in which the traverse begins at the root node and proceeds through the nodes as far as possible until we reach the node with no unvisited nearby nodes.
Conceptual Difference	BFS builds the tree level by level.	DFS builds the tree sub-tree by sub-tree.
Approach used	It works on the concept of FIFO (First In First Out).	It works on the concept of LIFO (Last In First Out).
Suitable for	BFS is more suitable for searching vertices closer to the given source.	DFS is more suitable when there are solutions away from source.
Applications	BFS is used in various applications such as bipartite graphs, shortest paths, etc.	DFS is used in various applications such as acyclic graphs and finding strongly connected components etc.

Assignment – 6

Q.1] Explain the expert System & various components with the help of block Diagram ?

- An expert system is a computer program that is designed to solve complex problems and to provide decision-making ability like a human expert.
- An expert system works by pulling information from its database and then using rules to make decisions based on what the user asks.
- It solves the most complex issue as an expert by extracting the knowledge stored in its knowledge base.
- The system helps in decision making for compsex problems using both facts and heuristics like a human expert.
- it contains the expert knowledge of a specific domain and can solve any complex problem of that particular domain.
- These systems are designed for a specific domain, such as medicine, science, etc.
- The more knowledge stored in the KB, the more that system improves its performance.
- One of the common examples of an ES is a suggestion of spelling errors while typing in the Google search box.

Componets of expert System:-



1. **User Interface**:

- With the help of a user interface, the expert system interacts with the user, takes queries as an input in a readable format, and passes it to the inference engine
- After getting the response from the inference engine, it displays the output to the user.

2. Inference Engine(Rules of Engine) :

- The inference engine is known as the brain of the expert system as it is the main processing unit of the system.
- the user's queries, applies rules, and draws conclusions based on the knowledge stored in the Knowledge Base.

There are two types of inference engine:

• **Deterministic Inference engine:** The conclusions drawn from this type of inference engine are assumed to be true. It is based on facts and rules.

• **Probabilistic Inference engine:** This type of inference engine contains uncertainty in conclusions, and based on the probability.

3. **Knowledge Base (KB**):

• The knowledgebase is a type of storage that stores knowledge acquired from the different experts of the particular domain.

It is considered as big storage of knowledge.

-The more the knowledge base, the more precise will be the Expert System.

• It is similar to a database that contains information and rules of a particular domain or subject.

• One can also view the knowledge base as collections of objects and their attributes.

- Such as a Lion is an object and its attributes are it is a mammal, it is not a domestic animal, etc.

Q.2] Write Short note

1] Characteristics of Expert System -

- **High Performance**: The expert system provides high performance for solving any type of complex problem of a specific domain with high efficiency and accuracy.

- **Understandable:** It responds in a way that can be easily understandable by the user. It can take input in human language and provides the output in the same way.
- **Reliable:** It is much reliable for generating an efficient and accurate output.
- **Highly responsive**: ES provides the result for any complex query within a very short period of time

2] Knowledge engineering -

- Knowledge engineering is a branch of artificial intelligence (AI) that develops rules that are applied to data to imitate the thought process of a human who is an expert on a specific topic.
- This field of artificial intelligence attempts to emulate the judgment and behaviour of a human expert in a particular field.
- Knowledge engineering focused on the transfer process; transferring the expertise of a problem-solving human into a program that could take the same data and make the same conclusions.

Process of Knowledge engineering:-

1. **Task Identification:-**
   - This is the first step in the knowledge engineering process where the task to be performed is defined.
   - Figure out what problem you want the computer to solve.
   - Make sure it's a real problem and that experts understand it well.
2. **Acquisition of Knowledge:-**
   - Once the problem is well defined then the next step Gather information about the problem.
   - For some problems, collect standard data.
3. **Prepare a road map:-**
   - Once the goal and knowledge base are available the next step is to get the roadmap ready by breaking the goal down into small steps .
   - Use what experts know to plan how the computer should make decisions
   - Think about all the different ways the problem could be solved.
4. **Encode :-**
   - Now it’s time to convert this knowledge into computer language.

- Here the knowledge is using different functions as well as in some cases, for a specific task, the algorithm is used to create a model.
- Expert Train and test the model using lots of data to make sure it works well.

5. Evaluation and Debugging:-

- In the process of creating an expert system, at each step, the model should be evaluated and debugged and then added to workflow.
- Once all small tasks are evaluated, they are assembled to create one whole expert system.
- This system is again evaluated on similar problems and Debugged if any issue is there.