SLOT:D2+TD2

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

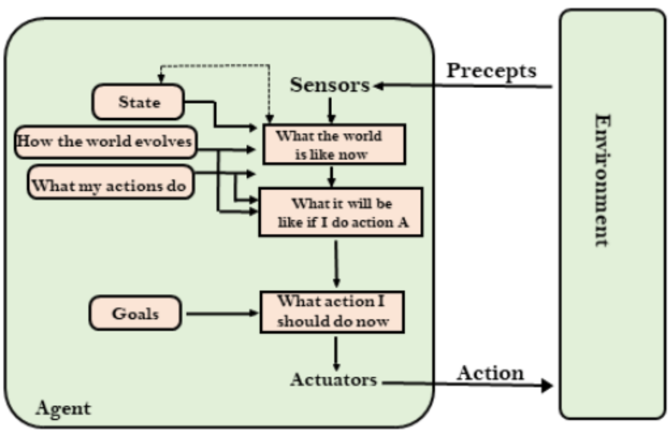
SCHOOL OF MECHANICAL ENGINEERING
CONTINUOUS ASSESSMENT TEST - I
WINTER SEMESTER 2024-2025

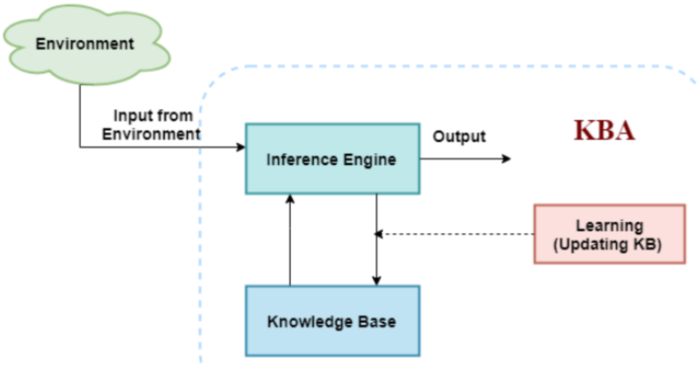
Programme Name & Branch : B. Tech – BMA/BME/BMM

Course Code and Course Name : BMEE407L - Artificial Intelligence

General instruction(s):

- Answer All Questions
- M - Max mark; CO – Course Outcome; BL – Blooms Taxonomy Level (1 – Remember, 2 – Understand, 3 – Apply, 4 – Analyse, 5 – Evaluate, 6 – Create)
- Course Outcomes
  1. Translate the characteristics of artificial intelligence and its sub-sets.
  2. Implement appropriate algorithm for problem solving by searching.
  3. Construct the logical agents and familiar in the application of fuzzy in AI

Q. No	Question	M	CO	BL
1.	**List and explain the different types of AI agents. Describe Goal-based agents and their functioning. Provide an example of a Goal-based agent in action.** The knowledge of the current state environment is not always sufficient to decide for an agent to what to do.  The agent needs to know its goal which describes desirable situations. Goal-based agents expand the capabilities of the model-based agent by having the "goal" information. They choose an action, so that they can achieve the goal.	10	1	2

	These agents may have to consider a long sequence of possible actions before deciding whether the goal is achieved or not. Such considerations of different scenario are called searching and planning, which makes an agent proactive.			
2.	**Explain the structure of a Knowledge-Based System. Describe its key components and their functions.**  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. 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	10	1	1

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:

Forward chaining

Backward chaining

Operations Performed by KBA

Following are three operations which are performed by KBA in order to show the intelligent behavior:

TELL: This operation tells the knowledge base what it perceives from the environment.

ASK: This operation asks the knowledge base what action it should perform.

Perform: It performs the selected action.

A generic knowledge-based agent:

Following is the structure outline of a generic knowledge-based agents program:

The knowledge-based agent takes percept as input and returns an action as output. The agent maintains the knowledge base, KB, and it initially has some background knowledge of the real world. It also has a counter to indicate the time for the whole process, and this counter is initialized with zero.

Each time when the function is called, it performs its three operations:

Firstly it TELLs the KB what it perceives.

Secondly, it asks KB what action it should take

Third agent program TELLS the KB that which action was chosen.

The MAKE-PERCEPT-SENTENCE generates a sentence as setting that the agent perceived the given percept at the given time.

The MAKE-ACTION-QUERY generates a sentence to ask which action should be done at the current time.

MAKE-ACTION-SENTENCE generates a sentence which asserts that the chosen action was executed.

Various levels of knowledge-based agent:

A knowledge-based agent can be viewed at different levels which are given below:

1. Knowledge level

Knowledge level is the first level of knowledge-based agent, and in this level, we need to specify what the agent knows, and what the agent goals are. With these specifications, we can fix its behavior. For example, suppose an automated taxi agent needs to go from a station A to station B, and he knows the way from A to B, so this comes at the knowledge level.

2. Logical level:

At this level, we understand that how the knowledge representation of knowledge is stored. At this level, sentences are encoded into different logics. At the logical level, an encoding of knowledge into logical sentences occurs. At the logical level we can expect to the automated taxi agent to reach to the destination B.

3. Implementation level:

This is the physical representation of logic and knowledge. At the implementation level agent perform actions as per logical and knowledge level. At this level, an automated taxi agent actually implement his knowledge and logic so that he can reach to the destination.

Approaches to designing a knowledge-based agent:

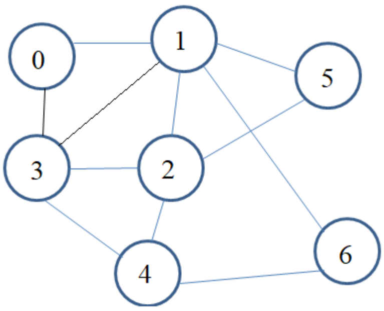
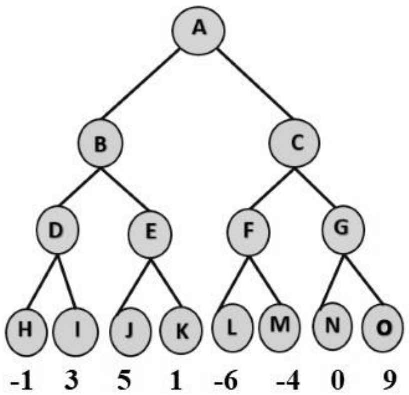
There are mainly two approaches to build a knowledge-based agent:

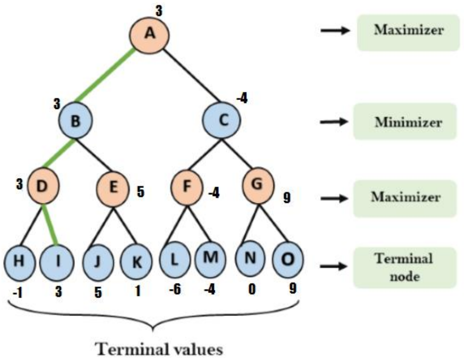
1. Declarative approach: We can create a knowledge-based agent by initializing with an empty knowledge base and telling the agent all the sentences with which we want to start with. This approach is called Declarative approach.

2. Procedural approach: In the procedural approach, we directly encode desired behavior as a program code. Which means we just need to write a program that already encodes the desired behavior or agent.

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF MECHANICAL ENGINEERING
CONTINUOUS ASSESSMENT TEST - I
WINTER SEMESTER 2024-2025

3.	**Using a Breadth-First Search (BFS) algorithm, describe the traversal path for the following graph when the starting node is 0. Explain the steps taken to explore the graph and the order in which the nodes are visited.**  Queue: 0 1 3 2 5 6 7 Result: 0 1 3 2 5 6 4	10	2	4
4.	**Given the following game tree, apply the Minimax search algorithm to determine the optimal move at the root. Show the process of evaluating the tree, including the calculation of the minimax values at each node, and describe the optimal decision based on the minimax values.** 	10	2	5



Pseudo-code for MinMax Algorithm:

```
function minimax(node, depth, maximizingPlayer) is
if depth ==0 or node is a terminal node then
return static evaluation of node

if MaximizingPlayer then      // for Maximizer Player
maxEva= -infinity
 for each child of node do
 eva= minimax(child, depth-1, false)
maxEva= max(maxEva,eva)       //gives Maximum of the values
return maxEva

else                        // for Minimizer player
 minEva= +infinity
 for each child of node do
 eva= minimax(child, depth-1, true)
 minEva= min(minEva, eva)       //gives minimum of the values
 return minEva
```

5.	Explain the concept of First-Order Logic (FOL). Provide an example of how FOL is used to represent knowledge. Illustrate the process of reasoning with a simple example, showing how facts and rules are applied to infer new information. 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	10	3	3

	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 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 $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$ Equality == Quantifier			

∀ , ∃

Atomic sentences:

- 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.
- 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).

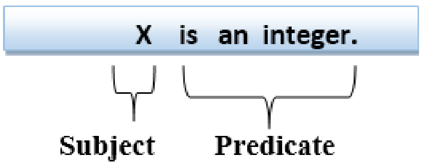
Complex Sentences:

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

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

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



Quantifiers in First-order logic:

- A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:
  - Universal Quantifier, (for all, everyone, everything)
  - Existential quantifier, (for some, at least one).

Universal Quantifier:
Universal quantifier is a symbol of logical representation, which specifies that

the statement within its range is true for everything or every instance of a particular thing.

The Universal quantifier is represented by a symbol ∀, which resembles an inverted A.

Note: In universal quantifier we use implication "→".

If x is a variable, then ∀x is read as:

- For all x
- For each x
- For every x.

Example:
All man drink coffee.

Let a variable x which refers to a cat so all x can be represented in UOD as below:

∀x man(x) → drink (x, coffee).

It will be read as: There are all x where x is a man who drink coffee.

Existential Quantifier:

Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something.

It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

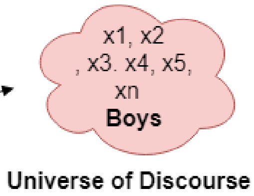
Note: In Existential quantifier we always use AND or Conjunction symbol (∧).

If x is a variable, then existential quantifier will be ∃x or ∃(x). And it will be read as:

- There exists a 'x.'
- For some 'x.'
- For at least one 'x.'

Example:
Some boys are intelligent.

- **x1 is intelligent**
  V
- **x2 is intelligentV**
- **x3 is intelligent**
  V
- .
- .
  V
- **xn is intelligent**



So in short-hand notation, we can write it as:

∃ x: boys(x) ∧ intelligent(x)

It will be read as: There are some x where x is a boy who is intelligent.

*****