T.Y. BSC. (IT): Sem. V (Elective-I)

Artificial Intelligence

Mumbai University Examination Paper Solution: Nov-19

Q.P. Code:754321

Q1. Attempt any three of the following **[15]**

Q1(a) Elaborate artificial intelligence with suitable example along with it's application.

Ans. John Maccarthy as invented the word Artificial Intelligence in year 1956.AI is define as science and engineering of making. Ai indicates that machine are able to take some decision in order to take right decision. AI system works like human brain.

Application of Ai are as follows:

i. Robotic Vehicles= A driver less robotic car or a autonomous car name "Stalee" was develop using AI.

ii. Speech recognition = It recognize the speech of an authenticated user and allows the service while block the service to an unauthorized user.

iii. autonomous planning and Scheduling= NASA develop an autonomous planning and scheduling for scheduling of different operations for a space craft.

iv. Game Playing= IBM DEEP BLUE become 1st computer program to detect the world chess champion GARRY KASPAROE.

v. Spam fighting = Each day the learning algorithm classify over a billon message as spam saving recipient from having to waste time deleting unwanted mail.

vi. Logistic Planning= Logistic planning is being used in different industry for performing planning and scheduling different activities.

vii. Robotics= The iRobot corporation has sold over two million robotics vacuum cleaners.

viii. Machine Translation = It uses AI technology to translate any language into an English language allowing an English people to communicate efficiently.

Other application of Ai are as follows:

AI in shopping

AI in education

AI in marketing

AI in hospitals

AI in entertainment. Industry

AI in Military Application

Q1(b) Discuss the historical evaluation of Artificial Intelligence. **(5)**

Ans.1. 1943-Greek mythology talos: Talos was giant animated bronze warrior program to guard the Island.

2. 1950-Allan Twining: Allan twining publish reach paper in which he started about possibility of creating machine that can think.

3. 1951-Game AI: Cristopher wrote a cheeker's program using AI and Dietrich wrote chess program using AI

4. 1956-Both of AI: In the year 1956 john macchathy invented first term Artificial intelligence.

5. 1959-First Ai lab: MIT ai lab was first setup in year 1959 and the research on AI begin.

6. 1960-General motors robot: General motors robot in 1960 introduce robot's for assembly line for industrial applications.

7. 1961-First chatbox: In year 1961 the 1st chat box was develop called as "eliza".

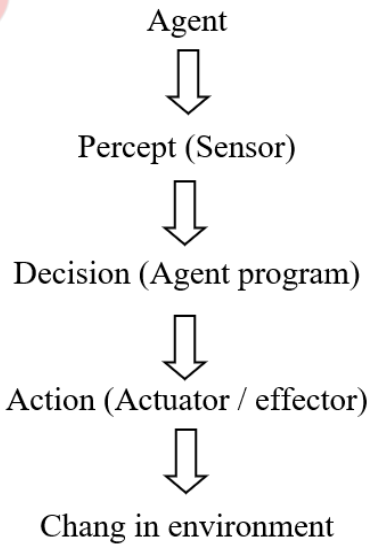
8. 1997-IBM DEEP BLUE: IBM DEEP BLUE defeated world chess champion "Garry Kasparov" in year 1997.

9. 2005-DARPA Grand challenge: Stanfordracingteam develop at autonomous robotics car which won the DARPA Grand challenge.

10. 2011-IBM Watson: IBM Watson is question and answering system which defected the worlds greatest genius people.

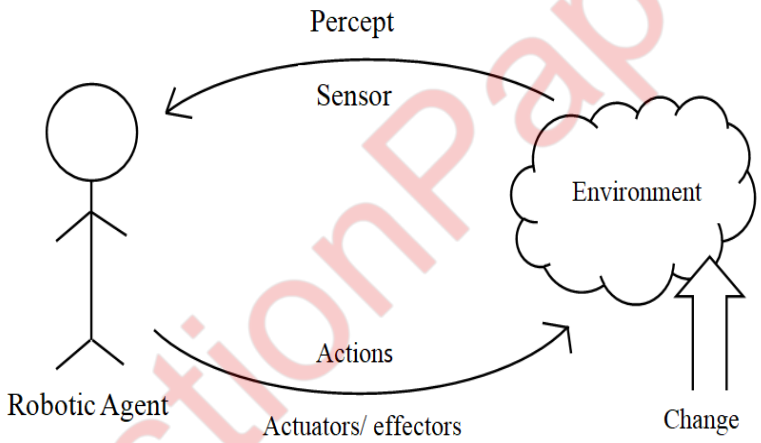
Q1(c)State the relationship between agent and environment. **(5)**

Ans.



- An agent is something which mean percept environment using different sensors and act upon the environment using actuators and effectors.

- The human agent percept the environment using eyes, nose, etc. while robotic agent percept using different sensors
- The human agent performs different actions using organ like arm, leg. While robotic agent performs different actions using actuators / effectors.
- The robotic agent percepts environment using different sensor while make decision using agent program want to perform different action using actuator /effect on which will bring change in environment.
- Agent function is the description of all the function.
- The agent function provide mapping between percent sequences to the desire action.
- The agent program is computer program that implements the agent function in suitable computer language and it is responsible for decision making.



Q1(d) Explain the concept of Rationality. **(5)**

Ans. 1. A rationality agent is one which acts to perform the right decision in order to achieve the best outcome.

2. A rational agent is one that does the right thing, conceptually speaking every entry in the table, for the agent function is filled out correctly.

3. A general rule says that it is better to design the performance measures according to what one actually wants in the environment rather than according to how one thinks the agent should behave.

4. The agent behaviors is based on following points:

- To achieve the high performance.

- To achieve the optimized result.
- To behave rationality.

5. What is rationality at any given time depends upon the four things as follows.
   - The performance measure that defines the criteria of success.
   - The agents prior knowledge of the environment.
   - The action that the agent can perform.
   - The agents percept sequence to the data.
6. These are four different types of agent which are as follows.
   i. Simple reflex agent
   ii. Model based agent
   iii. Goal based agent
   iv. Utility based agent

Q1(e) Explain types of environment. **(5)**

Ans. Following are different types of environment:

1. Fully observable v/s Partially observable:
   — The agent sensors can have access to complete state of environment of any given time or not which decides whether it is fully observable or partially observable.
   — The fully observable environment, agent is able to gather all necessary information required to take the action and don't have to keep records of internal states. e.g. 8 puzzle problem, chess, world black problem.
   — In partially observable, the environment is not even completely at any given point of time. E.g. Automated car driver's system which cannot predict the thinking of drivers driving the car.
2. Single agent v/s multiagent:
   — It indicates the number of agents acting in the environment and whether the agent is operating in its own or in collaborations decides whether it is single agent environment or multiagent environment.
   — E.g. Single agent: vacuum cleanness working on its own for home is.
   — E.g. Multi agent: autonomous car working in collaboration with respect to other cars.
3. Deterministic v/s stochastic:

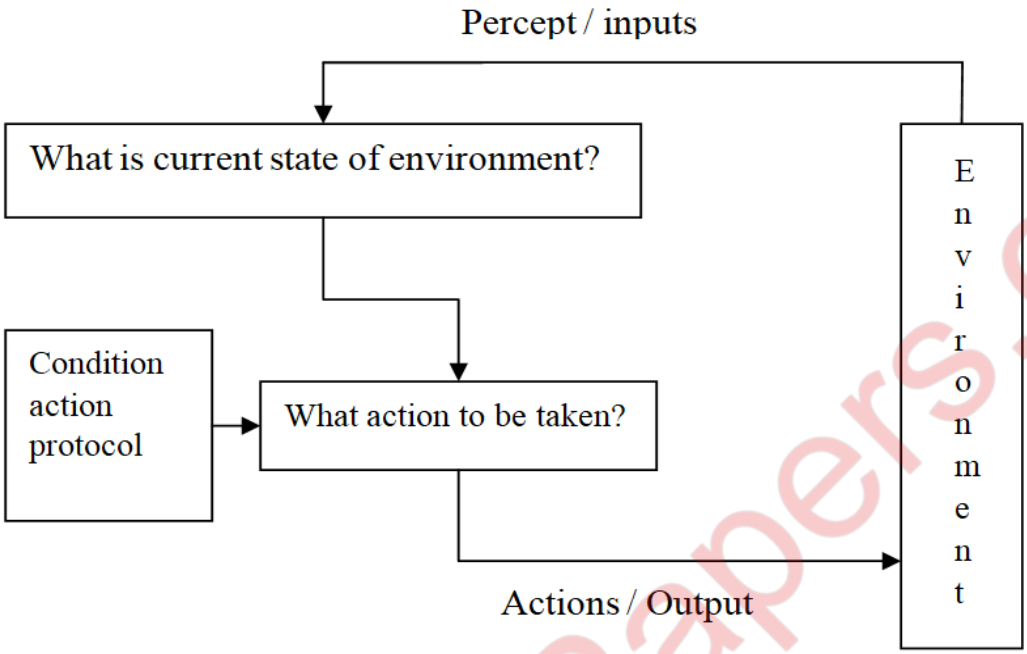
— An environment is said to be deterministic when the next state of the environment can be determined by the previous state. Eg. 8 puzzle problem, chess, etc.

— Stochastic environment means the next state cannot be merely determined by the previous state which requires the probability. Eg. Autonomous car.

4. Episodic v/s Sequential:
   — In episodic environment, each of the agent action is divide into episodes.
   — The current incident is different from the previous incident and there are no dependencies. E.g. Pick & place robot.
   — In sequential environment, the previous decision can effect the future decision. E.g. Checkers & game.
5. Static v/s Dynamic:
   — The static environment indicates that the environment remains unchanged while the agent is performing the task. E.g. vacuum cleaner for home use.
   — The dynamic environment indicates this environment changes while the agent is performing any task. E.g. Autonomous car.
6. Discrete v/s Continuous:
   — In discrete environment, there are distinct and clearly define inputs to the robotics agent.
   — The input is in discontinuous manner with respect to time. E.g. Chess.
   — In continuous environment there is no distinct or clearly defined inputs to the robotics agents.
   — The inputs to the agents are in continuous manners with respect to time. E.g. Input received by the sensors of the autonomous car.
7. Known v/s unknown:
   — In known environment, the agent is already aware of the environment whether it has to perform task.
   — Here, the agents perform the action without gaining knowledge. E.g. Chess, tic-tocker, ludo, etc.
   — In unknown environment, the agent is now aware of the environment whether it is going to perform the task.
   — Here, agent perform action by gaining the knowledge.

Q1(f) Explain Reflex agents with state. **(5)**

Ans.



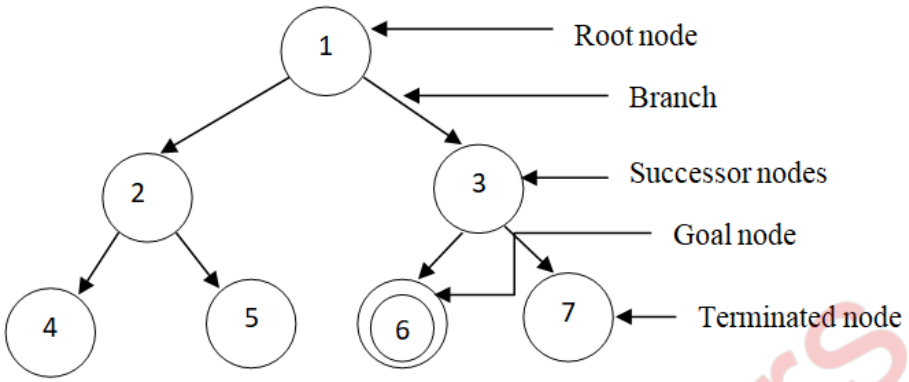
- In simplex reflex agent, an agent performs the action based on the current state/ input only by ignoring all the previous state of agent of the environment is called as simplex reflex agent.
- Simple reflex agent is totally uncomplicated type of agent.
- The simplex reflect agent functions is based on the situation and its corresponding action.
- The agent reflects on condition action protocol for performing any action.
- If the condition is true, then the matching action is performed without considering previous history.
- E.g.: robotic vacuum cleaner for the home use.

Percepts	Actions
Dirt	Suck
Clean	Un suck

Q2. Attempt any three of the following **[15]**

Q2.(a) Write procedure for tree search. **(5)**

Ans.



1. Function tree- Search(problem, fringe) returns a solution, or failure
2. fringe<- Insert(Make-node(Initial-State[problem]),fringe)
3. loop do
4. if fringe is empty then return failure
5. node<- Remove- Front(fringe)
6. if goal- Test(problem,State(node))then return node
7. fringe<- InsertAll(Expand(node,problem),fringe)
8. function Expand(node,problem) returns a set of nodes
9. successors<- the empty set for each action, result in successor-Fn(problem,State[node])do
10. s<- a new node
11. Parent-Node[s]<- node;Action[s]<-action;state[s]<-result
12. Path-cost[s]<-Path-cost[node]+ step-cost(State[node],action,result)
13. Depth[s]<-Depth[node]+1
14. Add s to successors
15. Return successors

Q2.(b) Explain the algorithm for breadth first search algorithm. **(5)**

Ans. BFS is simple strategy in which rot node is expanded 1st then all the successor of root node are expanded next. Then they are become successors. In bsf the shallowest unexpanded node is selected for the expansion. Bsf is also called as a fifo technique which mean first in first out. BSF algorithm are as follows:

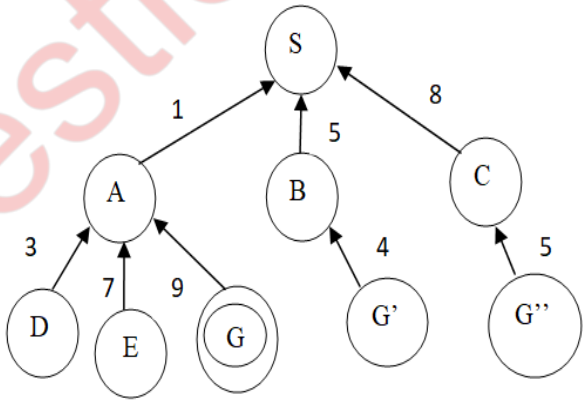
- Create a variable called NODE-LIST and set it to initial state.

- Until a goal state is found or NODE-LIST is empty do:
  a. Remove the first element from NODE-LIST and call it E. if NODE-LIST was empty, quit.
  b. For each way that each rule can match the state describe in E do:
     i. Apply the rule to generate a new state.
     ii. If the new state is goal state, quit and return this state
     iii. Otherwise, add the new state to the end of NODE-LIST.

Exp. Node	Open list	Closed list
	{S}	{}
S	{A B C}	{S}
A	{B C D E G}	{S A}
B	{C D E G G'}	{S A B}
C	{D E G G' G"}	{S A B C}
D	{E G G' G"}	{S A B C D}
E	{G G' G"}	{S A B C D E}
g	{G' G"}	{S A B C D E}

Solution path found is S A G<- this G also has cost 10

Number of nodes expanded(including goal node)=7



Q2.(c) Give the outline of uniform cost search algorithm. (5)

Ans. All step cost are equal BSF is optimal because it always expands the shallowest unexpanded node. Instead of expanding the shallowest node uniform cost search expand node with the lowest path cost. UCS(uniform cost search) does not care about the number of steps a path has but only about their total cost.

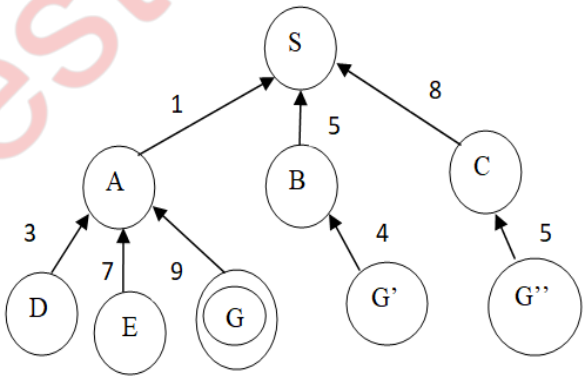
- Create a variable called NODE-LIST and set it to initial state.

- Until a goal state is found or NODE-LIST is empty do:
  a. Remove the first element from NODE-LIST and call it E. if NODE-LIST was empty, quit.
  b. For each way that each rule can match the state describe in E do:
     i. Apply the rule to generate a new state.
     ii. If the new state is goal state, quit and return this state
     iii. Otherwise, add the new state to the end of NODE-LIST.

Exp. Node	Open list	Closed list
	{S}	{}
S	{A B C}	{S}
A	{B C D E G}	{S A}
B	{C D E G G'}	{S A B}
C	{D E G G' G"}	{S A B C}
D	{E G G' G"}	{S A B C D}
E	{G G' G"}	{S A B C D E}
g	{G' G"}	{S A B C D E}

Solution path found is S A G<- this G also has cost 10

Number of nodes expanded(including goal node)=7

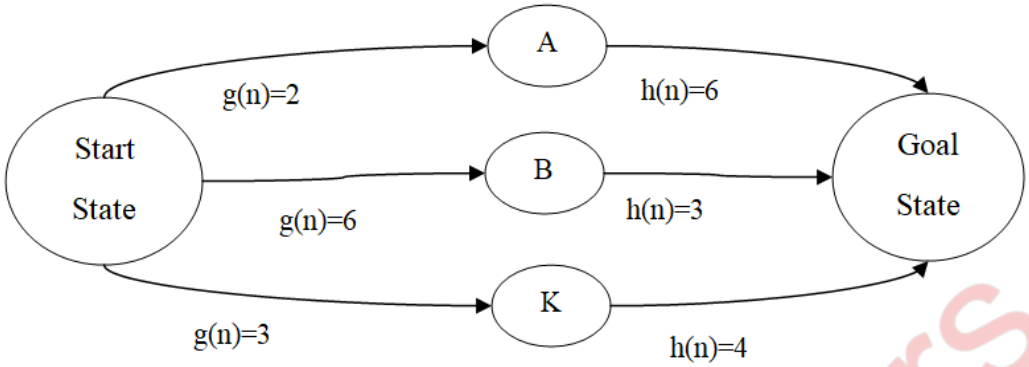


Q2.(d) Explain A* algorithm for the shortest path. (5)

Ans. A* search is the commonly known from of best first search. It uses heuristic f^n and cost to reach to the node n from the start state g(n). it has the combine features of UCS and greedy best first search by which it solve the problem efficiently. A8 search algorithm finds the shortest

path through search space using heuristic f^n. This search algorithm expands less search tree and provides optimal result faster.

Its evaluation function given as: F(n)=g(n)+h(n)



A* search algorithm,

i. Place starting node in open list.
ii. Check if open list is empty or not.
iii. If list is empty then return failure and stop.
iv. Select node from OPEN list which has smallest value of execution function f(n)=g(n)+h(n)
v. If node n is goal node then return success and stop.
vi. Otherwise expand node n and generate all of its successor and put it into the CLOSE list.

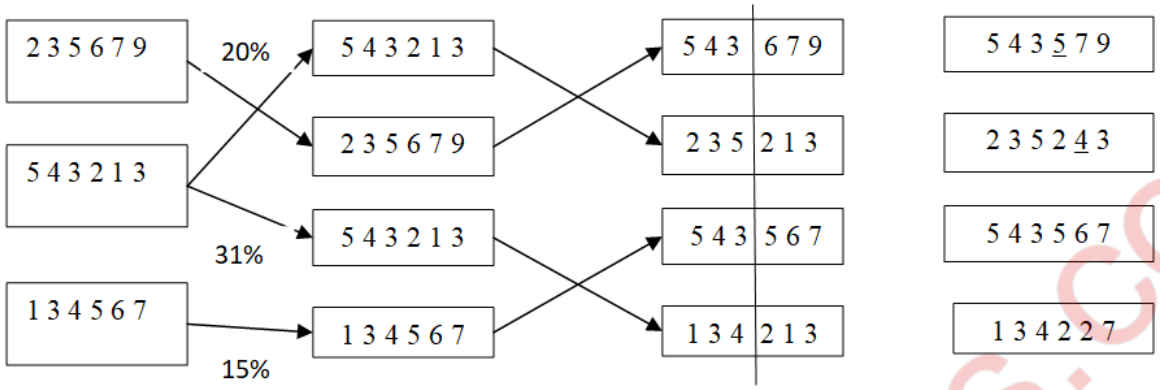
Q2.(e) Give the outline of hill climbing algorithm. (5)

Ans. Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing values to find peak of the mountain or best solution to the problem. It determinates when it reaches a peak values where no neighbor has a higher value. Hill climbing algorithm is a technique which is used to optimize the mathematical problem. One of the widely discussed example of hill where we need to minimize distance travelled by salesman. Hill climbing algorithm:

```
Functio HILL-CLIMBING(problem)return a state that is a local minimum
Current<- MAKE-NODE(problem.INITIAL-STATE)
loop do
        neighbour<- a highest-valued successor of current
        if neighbour. VALUE ≤ current.VALUE.then return current.STATE
        current<-neighbour
```

Q2(f) Explain the working mechanism of Genetic Algorithm. **(5)**

Ans:



- A genetic algorithm is a stochastic hill climbing search in which a large population of state is maintained.
- New states are generate by mutation and by crossover which combines pairs of states from the population.
- Genetic algorithm:
  - i. Generate a random population of chromosomes.
  - ii. Evaluate the fitness of each chromosomes in the population.
  - iii. Create new population by repeating the following steps until new population is complete select 2 parent chromosomes from a population as per their fitness with crossover probability. Crossover the parents to from new offspring as a child with a mutation probability, muted a new offspring in a new population.
  - iv. If the end condition is satisfied, stop and return the best solution.

Q3. Attempt any three of the following **[15]**

Q3(a) What is alpha-beta pruning? Explain the function of alpha-beta pruning. **(5)**

Ans. Alpha beta pruning is a modifies version of min-max algorithm.

- It is an optimization technique for the minmax algorithm.
- There is technique by which without checking each node of the game tree we can complete the correct minmax decision and this technique is called as pruning.
- This involves two threshold parameters alpha and beta for future expansions. so it is called as a alpha beta pruning.
- It is also called as alpha beta algorithm.
- Alpha beta pruning can be applied at any depth of a tree and sometimes it not only prime the tree levees but also the entire sub tree.

- The two parameters involved are as follows:
  - Alpha(α):
    - The alpha is the best highest value choice found at any point algorithm the path of maximize.
    - The initial value of alpha is -∞.

      α = -∞
  - Beta(β):
    - It is best lowest value choice found at any point along the path of minimize.
    - The initial value of beta is ∞.

      β = ∞
- the alpha beta pruning to a standard minmax algorithm returns the same move as the standard algorithm does but it removes all the nodes which are not really affecting the final decision but making the algorithm allow.
- Hence by pruning these nodes, it makes the algorithm fast.
- The main condition required for alpha beta pruning is α > β
- The max player will only update value of alpha.
- The min player will only update value of beta.
- While backtracking the tree, the nodes value will be passed to upper nodes instead of value of alpha beta.
- Steps involved for solving alpha beta problem:
  a. The initial value of alpha and beta are as follows:

     α = -∞, β = ∞
  b. The alpha value is updated at the max node.
  c. The beta value is updated at the min node.
  d. When the value of alpha and beta travels downwards it remains the same value.
  e. When the value travel upwards the value is forwarded as per the node whether it is more node as the min node.

Q3(b) Give the outline of min max algorithm. **(5)**

Ans. The minmax algorithm perform a complete depth-first exploration of the game tree. If the maximum depth of the tree is m and there are b legal moves at each point, then the time complexity of the minmax algorithm is moves at each point, then the time complexity of the minmax algorithm is O(bm). The space complexity is O(bm). For an algorithm that generates all actions at once, or O(m) for an algorithm that generates actions one at a time. Choose move to position with highest minmax value= best achievable payoff against best play.

function Minmax – Distance(state) returns an action

inputs: state, current state in game

return the a in Action(state) maximizing Min-Value (Result (a, state))

function Max – Value(state) returns a utility value

if Terminal – Test(state) then return Utility(state)\

v= -∞

for a, s in Successor(state) do v Max (v, Min – Value(s))

return v

function Min – Value(state) returns a utility value

if Terminal – Test(state) then return Utility(state)

v= ∞

for a, s in Successor(state) do v Min (v, Max – Value(s))

return v

Q3(c) Give note on card games. **(5)**

Ans. After the cards are distributed to players, there is no randomness. Every player can choose which cards he plays but no random process occurs as when distributing the cards at the beginning of the game. There is restriction about the cards which can be played when a card was already played. The player which wins the trick, plays then first. E.g. Player 1 plays a card, Player 2 plays a card and wins. Then Player 2 plays a card and then Player 1 plays.

Following are steps to play card game:

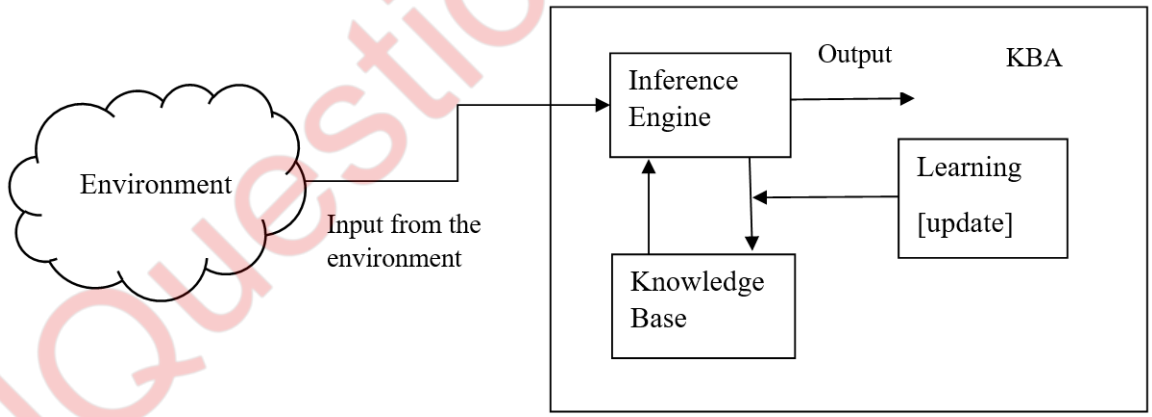
- Determine the probabilities of each card being in a given hand. You know with certainty which cards are in your hand and which cards have been played. Determine the probability of all other cards based on cards that have been played and possibly a

player's bid if there's bidding involved. To start, you could just use a naive and equal probability that a card is in some player's hand.

- Now, run through as many "virtual" games as you can. Simulate playing a card from your hand and then determine your opponents' responses using the rules of the game and your probabilities. For each virtual game, use your probabilities to assign cards to a player and then quickly simulate the game. Assume each player will play to the best of their ability. You know all the cards in your virtual game so you can make each player play perfectly.
- When you have a solid sampling (or you run out of time), pick the legal move that gave you the best outcome most often.
- Once you get something working, you can add all sorts of enriched strategies. For instance, vary your probabilities based on a player's historic plays, vary probabilities based on a player's style (passive, cautious, aggressive), or even consider the effects of specific players playing together.

Q3(d) What is knowledge-based agent? Explain its role and importance. (5)

Ans.

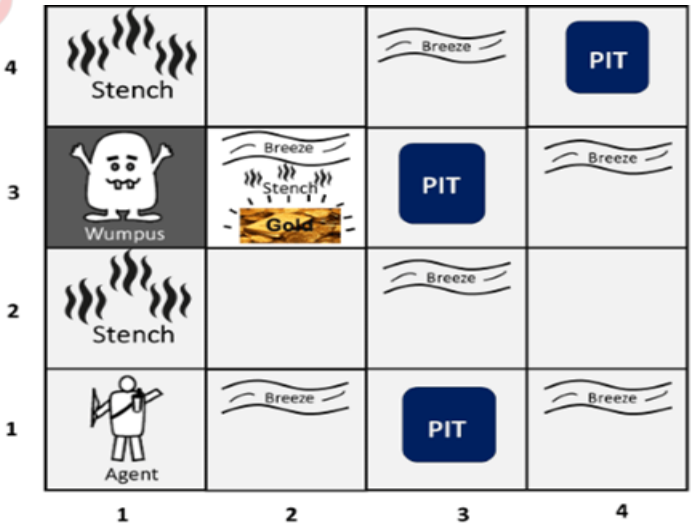


- An intelligent agent needs knowledge about the real world for taking decision and reasoning to add efficiently.
- Knowledge based agent are those agents who has capability of maintaining an interval state of knowledge reason over that knowledge update action.
- Input taken by the inference engine of the agent which also communicate with knowledge based.

- The learning element of knowledge-based agent regularly updates the knowledge based by learning new knowledge.
- Components of knowledge base agent [KBA]:
    a. Knowledge base [KB]=
        — Knowledge base is central component of kb agent. It is also known as KB.
        — It is collection of sentences where sentences are expressed in language which is also called as knowledge representation language.
        — The knowledge based of KBA stores the fact about the world.
        — KB required for updating knowledge for an agent to learn with experience and take action as per the knowledge.
    b. Inference Engine=
        — Inference means deriving new sentence from old sentence.
        — Inference system allows to add new sentence to KB.
        — A sentence is a proposition about the world.
        — Inference system applies logical rules to the knowledge base to derive new information.
        — Inference system generates new facts so than an agent can update the knowledge based.

Q3(e) Write note on Wumpus world problem. **(5)**

Ans.



- The Wumpus world problem is simple example to illustrate the KBA and to represent the knowledge Representation.
- The Wumpus world is a cave which has 4*4 rooms connected with passage waves, so there are total 16 rooms which are connected with each other.
- We have KBA who will go forward in this world.
- The cave has room with a beast which is called as Wumpus who can eat anyone, who enters into the room.
- The Wumpus can be shoot by the agent but the agent has single arrow.
- It the Wumpus world there are some pit rooms which are bottom less and if agent falls in pit then he will be stuck their forever.
- The exiting think with this cave is that, in one room there is possibility of finding a gold so the agent goal is to find the gold and climb out the cave without fallen into pit or eaten by Wumpus.
- The room adjacent to the Wumpus room are smelly which result in stench.
- The room adjacent to pit has a breeze so if the agent reaches near to pit then he will pursue the breeze.
- There will be glitter in the room if and only if the room has gold.
- The Wumpus can be killed by agent if the agent is facing to it and Wumpus will emit the horrible screen which can be hurt anywhere in cave.
- The Wumpus world is partially observable because the agent can only pursue the close environment such as an adjacent room.
- Wumpus world is deterministic and static in nature.

Q3(f) Give outline of resolution algorithm. (5)

Ans. The process of forming an inferred clause or resolving from the parent clauses is called resolution. It is a general rule of inference that combines substitution, modus pones and various types of syllogism. It is also a syntactic inference procedure which determines if the set is unsatisfiable. There are various types of resolutions such as binary, unit resulting, unary and linear input resolution. In general, the process of resolution is given into two steps:

1. Two parent clauses that contain the literal, once in the positive form and once in the negative form are selected.

2. The new inferred clause is formed by OR' ing together with all the literal form the parent clause except form the cancelled pair.

These steps are applied iteratively till nil is achieved in the refutation graph for the unsatisfiable set. The refutation is the process for showing the unsatisfiable of the set of clauses.

Q4. Attempt any three of the following **[15]**

Q4(a) What are Predicates? Explain its syntax and semantics. **(5)**

Ans. A predicate is a function that tests for some condition involving its arguments and returns nil if the condition is false, or some non-nil value if the condition is true. One may think of a predicate as producing a Boolean value, where nil stands for false and anything else stands for true.

- We have seen how to represent the statement using propositional logic but unfortunately in propositional logic we can only represent fact which can be either true or false.
- Propositional logic is not sufficient to represent the complex sentences or natural language statement.
- The propositional logic has very limited expressive power. E.g. some people likes cricket. Such statement cannot be represented by propositional logic. To represent this statement, we require some powerful logic such as FOL/ Predicate.
- Predicate is sufficiently expressive to represent the natural language statements in concise way.
- Syntax:
  - It indicates the structure or the format of the data.
  - Syntax of predicate determines which collection of symbol is logical representation or expression.
  - The basic syntax are as follows:
    - Constant= 1, 2, A, john, etc.
    - Variable= a, b, x, p.
    - Predicates= brother of, father of, sister of
    - Function= sqrt(), add(), sub(), mul(), div().
    - Connectives=^, ⇔, =>.
- Semantics:

— It indicates the meaning of data and it helps to infer new logical statement.

— It contains equality [==] or it indicates quantifiers ∀, ∃.

Q4(b) what are Quantifiers? Explain types with syntax and example. (5)

Ans. A quantifiers is language element which generates quantification and quantification specifies in universe of discos. There are two types of quantifiers,

I. Universal Quantifier:

— It is symbol of logical representation which specifies that the statement within its range is true for everything or every instance of particular think.

— It is used to represent for following cases such as for all x, for each x, for every x.

— The universal quantifier represented by symbol [∀] which indicates an invited 'A'.

— The main connective for the universal quantifier is "->" [implication].

— E.g. all boys like cricket.

∀ x: boy(x) -> like (x, cricket)

II. Existential Quantifier:

— Existential quantifier is the type of quantifier which express the statements within its scope its true for at least one instance or some think.

— It is denoted by logical operator "∃" which indicates an invented "E".

— The main connective which is used for the existential quantifier is "A" [and]/ conjunction.

— E.g. some boys like cricket.

∃ x: boys (x) ^ like (x, cricket)

Q4(c) Convert following into predicate form. (5)

1) Virat is software engineer.

Ans. Virat(software engineer)

2) All vehicles have wheels.

Ans. For-all(x): vehicles(x) -> wheel(x)

3) Some -one speaks some language in class.

Ans. ∃ x ∃ y: person(x) v language(y)-> speaks(x, y)

4) Everybody loves somebody sometimes.

Ans. (for all(x) (exists(y) -> loves sometime(x, y)))

5) All software engineer develops software.

Ans. For-all(x):software(x) -> software engineer(y)

Q4(d) Explain process of knowledge engineering. **(5)**

Ans. Knowledge Engineering,

The process of constructing knowledge based in FOL is called as knowledge engineering. In the someone who investigate particular domain learns important concept of that domain and generates a formal representation of the objects is known as knowledge engineering.

Following steps are follows for the knowledge engineering process:

1. Identify the task:
   - — the knowledge engineer must delineate the range of questions that the knowledge base will support and the kinds of fact that will be available for each specific problem instance.
   - — This step is analogous to the PEAS process for designing agents.
2. Assemble the relevant knowledge:
   - — The knowledge engineer might already be an expert in the domain, or might need to work with real expert to extract what they know- a process called knowledge acquisition
   - — At this stage, the scope of the knowledge is not represented formally.
   - — The idea is to understand the scope of the knowledge base, as determined by the task, and to understand how the domain actually works.
3. Decide on vocabulary:
   - — That is, translate the important domain-level concepts into logic-level names.
   - — This involves many questions of knowledge-engineering style.
   - — Like programming style. This can have a significant impact on the eventual success of the project.
4. Encode general- knowledge about the domain:
   - — The knowledge engineer writes down the axioms for all the vocabulary terms.

— This pins down the meaning of the terms, enabling the expert to check the content.

— Often, this step reveals misconceptions or gaps in the vocabulary that must be fixed by returning to step 3 and iterating through the process.

5. Encode description of problem instance:
   — Of the ontology is well throughout, this step will easy.
   — It will involve writing simple automatic sentence about instances of concepts that are already part of the ontology.
6. Pose queries to the interface procedure and get the answer:
   — This is where the reward is: we can let the interface procedure operate on the axioms and problem=specific facts to derive the facts we are interested in knowing.
   — Thus, we avoid the need for writing an application-specific solution algorithm.
7. Debug the knowledge base:
   — In this step debugging of the knowledge based takes place.
   — This is last step of complete process; in this step we will try to debug the issue of knowledge.

Q4(e) What is unification? Explain the process of unification. (5)

Ans. Unification is a process of making two different logical atomic expression identical by finding substitution.

- Unification depends on the substitution process.
- Let, Ψ_1 and Ψ_2 = Two atomic sentences

  σ= unifier

  Such that, $\Psi_1 \sigma = \Psi_2 \sigma$ can be expressed as UNIFY (Ψ_1, Ψ_2)
- E.g. find MGU [Most General unifier] for unify(king(x), king (john))

  Let, Ψ_1 = king(x), Ψ_2 = king(john)

  Substitution: θ = $\frac{\text{john}}{\text{x}}$

  unifier[MGU]= $\frac{\text{john}}{\text{x}}$
- E.g. Ψ_1 =p(x, y), Ψ_2 =(p(a, F(z)))

  θ_1 = a/x θ_2 =f(x)/y

- Unification takes two literals as input and makes them identical using substitution.
- The unify algorithm is used for unification which takes two atomic sentences and return unifier for those sentences.
- Unification is key component of all first order inference algorithm.
- It returns fail if the expression does not match with each other.
- The substitution variable is called MGU.
- Implementation of unification algorithm:
    - ✓ Initialize the unification algorithm.
    - ✓ Recursively unify the atomic sentences.
    - ✓ Check for identical expression.
    - ✓ Of one expression variable "v" and other is term "t" which does not contain variable "v" then substitute t/v in existing substitution.
    - ✓ Add t/v to the substitution set list.
    - ✓ If both the expressions are functions then function name must be similar and the number f argument must be same in both the expression.

Q4(f) Give outline of simple forward Channing algorithm. (5)

Ans. 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.

E.g. If It is Raining, we will take the umbrella.

Data= if it is raining.

Goal/ Decision: we will take umbrella.

Properties of Forward-Chaining:

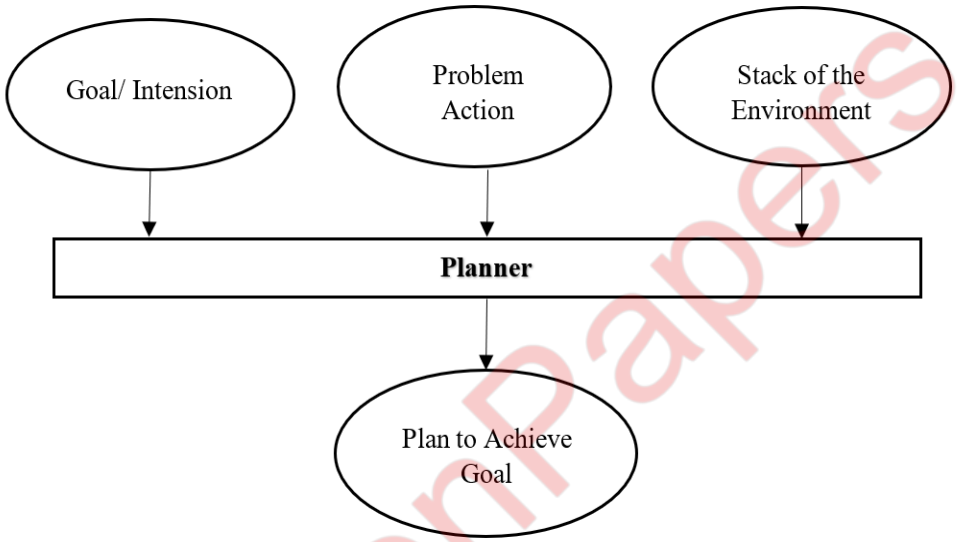
- It is a down-up approach, as it moves from bottom to top.
- It is a process of making a conclusion based on known facts or data, by starting from the initial state and reaches the goal state.

- 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.

Q5. Attempt any three of the following **[15]**

Q5(a) What is planning? what is need of planning. **(5)**

Ans.

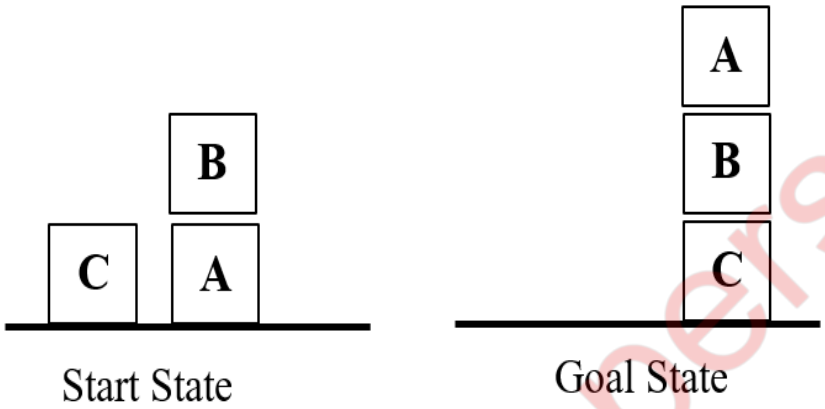


- Planning is artificial intelligence can be defined as a problem that needs decision making by intelligent system to accomplish the given target.
- There is one more definition of planning which says that planning is an activity where agent has to come up with sequence of action to accomplish the target.
- Here we have information about the initial status of agent, goal condition of agent and set of action an agent can take.
- Aim of agent is to find the proper sequence of action which will lead from start to goal state and produce an efficient solution.
- A planning agent that interacts with environment using sensors and actuators.
- When task comes to agent it has to decide the sequence of actions to be taken and then accordingly execute action.
- Planning Problem,
  - — The states of an agent correspond to the probable surrounding environment while action and goal of an agent are specified based on logical formulation.

— To achieve any goal an agent has to answer few questions like what will be the effect of its action, how it will affect the upcoming action.

Q5(b) Explain block world problem for following start state and end state. (5)

Ans.



- In the block world, if the goal is to build a tower A on B, which in turn is on C, which in turn is on the table, as in the figure, them the subgoals are serializable bottom to top :if we first achieve C on table, we will never have to undo it while we are achieving subgoals.
- A planner that uses the bottom-to-top can solve any problem in the block's world without backtracking (although it might not always find the shortest plan).

Q5(c) Write note on planning graph. (5)

Ans.

Planning Graph Level

Set of literals

- All literals which are true at that point of time.
- Depends on executed actions at previous time sleep.

Set of actions

- All actions which meet prerequisite at the point of time.
- Depends upon which literates are true.

- Planning graph is a special data structure which is used to get better accuracy. It is directed graph and is useful to accomplish improved heuristic estimates.
- Any of the search technique can make use of planning graph. Also, GRAPHPLAN can be used to extract a solution directly.
- Planning graph works only for propositional problems without variable.
- Similarly, in case of planning graph there are series of levels which match to time ladle in the plan. every level has set of literals and a set of actions.
- Level 0 is the initial state of planning graph.
- Properties of planning graph:
  - If goal is absent from last level then goal cannot be achieved.
  - If there exist a path to goal the goal is present in the last level.
  - If goal is present in last then there may not exist any path.

Q5(d) What are events? Explain its importance. (5)

Ans. We can represent situation of real world by specifying date, time, place, related people and many other related objects. But there are many accessions of having continuous actions such as filling a bucket of water, solving a puzzle, etc.

Situation calculus can only specify condition at the start of the action and at the end of actions; but it cannot represent what happed during the action was taking place. Considering the same example, situation calculus will specify that, bucket was empty at the start of action and at the end bucket is full.

One more limitation of situation calculus is that it cannot represent simultaneous action. E.g. writing assignment while watching tv program. To handle such things, we have event calculus.

Event calculus is based on time points instead of only start state and end state. Event can be described as instance of the event category. Event calculus consist of event and fluent. Fluent are the object to represents fact but do not specify its truthfulness.

Following is set of event representation:

T(f, t)	Fluent f is true at time.
Happens(e, i)	Events e happens over the time interval.
Initiates(e, f, t)	Event e causes fluent f to start to hold at time t.

Terminates(e, f, t)	Event e cause fluent f to cease to hold at time t.
Clipped(f, i)	Fluent f ceases to be true at some point during time interval i.
Restored(f, i)	Fluent f becomes true sometime during time interval.

Knowledge is the most important aspect of knowledge base agent. They use the available knowledge and deduce new set of knowledge based on that. As in case of human, if we need some information, we ask a question to the authorized person, who knows the matter we need.

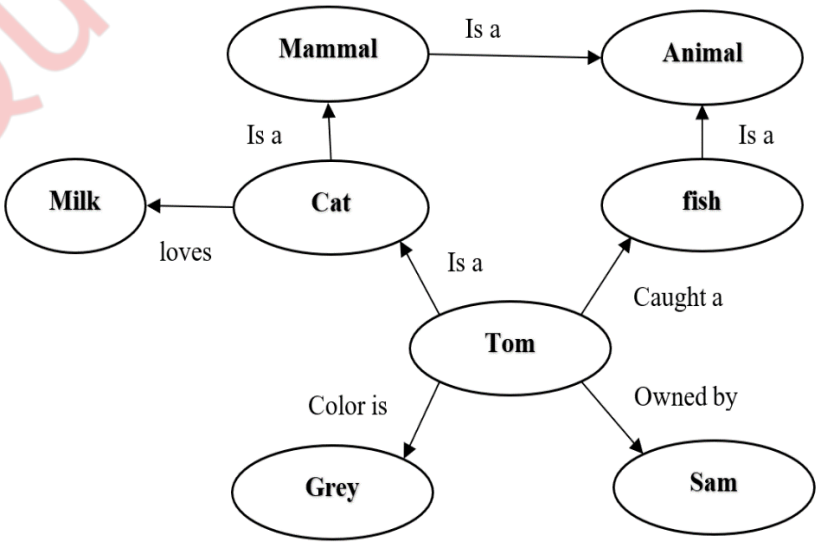
Similarly, the knowledge base agent also needs to know what information it has in order to deduce new one. It needs to know about the inference process, so that it gets the answer wants to have.

This initiates the need to know the objects in someone's head called as mental objects and the process to manipulate the object is called as mental events. Using mental objects and events, agent can reason about beliefs of agents.

Mental objects for agents can show their attitudes such as knows, wants, believes, intends, informs. These attitudes cannot be used as a normal predicate. Hence, we need to reify them.

Q5(e) Write note on semantic network. **(5)**

Ans. Semantic nets provide graphical representation of knowledgebase and efficient algorithms provides inferences from the mental objects on the basis of category membership and decides subset superset relationships among the categories.



- A semantic net or semantic network is a knowledge representation technique used for propositional information, so it is also called a propositional net. In semantic networks the knowledge is represented as objects and relationships between objects.
- They are two dimensional representations of knowledge. It conveys meaning. Relationships provide the basic structure for organizing knowledge.
- It uses graphical notations to draw the networks. Mathematically a semantic net can be defined as a labeled directed graph. As nodes are associated with other nodes semantic nets are also referred to as associative nets.
- Semantic nets consist of nodes, links and link labels. Nodes of the graph denote object while the links indicate relations among the objects.
- Nodes can appear as circles, ellipses, or rectangle to represent objects such as physical objects, concepts or situations. Links are drawn as arrows to express the relationships between objects, and link labels specify specifications of relationships.
- The two nodes connected to each other via a link are related to each other. The relationships can be of two types: "IS-A" relationship or "HAS" relationship. IS-A relationship stands for one object being "part of" the other related object And “HAS” relationship indicates one object “consists of” the other related object.

Q5(f) Write note on Truth maintenance system. (5)

Ans. A truth maintenance system maintains consistency between old believed knowledge and current believed knowledge in the knowledge base (KB) through revision. If the current believed statements contradict the knowledge in the KB, then the KB is updated with the new knowledge.

It may happen that the same data will again be believed, and the previous knowledge will be required in the KB. If the previous data are not present, but may be required for new inference. But if the previous knowledge was in the KB, then no retracing of the same knowledge is needed.

The use of TMS avoids such retracing; it keeps track of the contradictory data with the help of a dependency record. This record reflects the retractions and additions which makes the inference engine (IE) aware of its current belief set.

Each statement having at least one valid justification is made a part of the current belief set. When a contradiction is found, the statement(s) responsible for the contradiction are

identified and the records are appropriately updated. This process is called dependency-directed backtracking.

The TMS algorithm maintains the records in the form of a dependency network. Each node in the network is an entry in the KB (a premise, antecedent, or inference rule etc.) Each arc of the network represents the inference steps through which the node was derived.

A premise is a fundamental belief which is assumed to be true. They do not need justifications. The set of premises are the basis from which justifications for all other nodes will be derived.

There are two types of justification for a node. They are:

1. Support list [SL]
2. Conditional proof (CP)

Many kinds of truth maintenance systems exist. Two major types are single-context and multi-context truth maintenance. In single context systems, consistency is maintained among all facts in memory (KB) and relates to the notion of consistency found in classical logic. Multi-context systems support Para consistency by allowing consistency to be relevant to a subset of facts in memory, a context, according to the history of logical inference.