DATE : / /
PAGE :

a) Date and Time of examination: 04/04/2022 ; 9:00AM

b) Examination Roll number : 2123474057

c) Name of the Program : M.Sc. Computer Science

d) Semester / Year : I / 1st year

e) Unique Paper code (UPC) : 223411102

f) Title of the Paper : Artificial Intelligence

Answer 3:

(a) There are mainly four ways of knowledge representation which are :-

1. Logical Representation.
2. Semantic Network Representation.
3. Frame Representation.
4. Production Rules (or Rule-based systems).

1. Logical Representation

→ A logic is a formal language, with precisely defined syntax and semantics, which supports sound inference. Independent of domain of application.

→ Different logics exist, which allows us to represent different kinds of things and which allow more or less efficient inference.

- Propositional logic, predicate logic, temporal logic, description logic.

Example :- "Everything in the garden is lovely".

$$\forall X \text{ in } (X, \text{garden}) \rightarrow \text{lovely}(X)$$

Here, in (X, garden): X in garden and lovely(X): X is lovely.

2. Semantic Network Representation

→ In Semantic networks, we can represent our knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects.

2123474705 7

DATE :
PAGE :

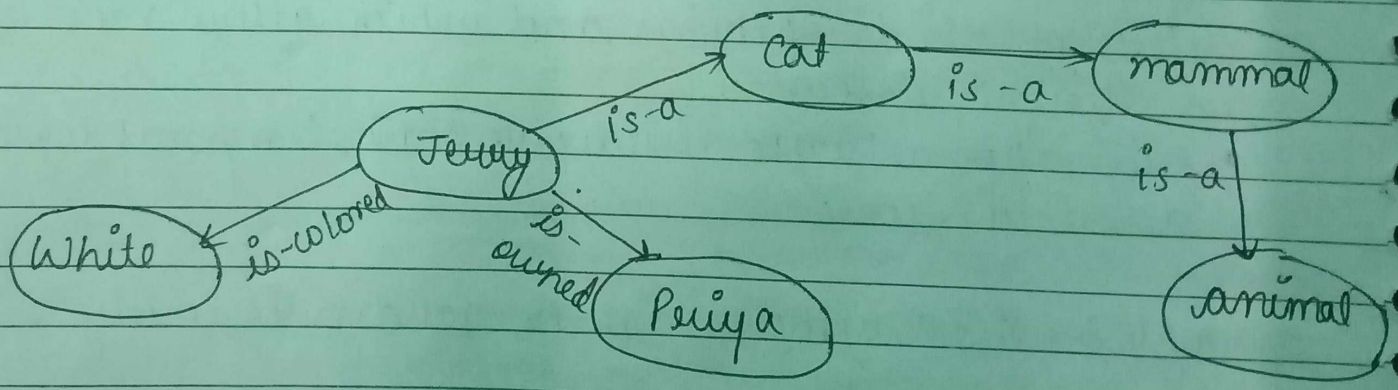
→ Semantic networks can categorize the object in different forms and can also link those objects. They are easy to understand and can be easily extended.

→ This representation consists of mainly two types of relations.

a) IS-A relation (Inheritance)
b) kind-of-relation

Example :- Here are some statements which we need to represent as semantic network.

a. Jerry is a cat.
b. Jerry is a mammal
c. Jerry is owned by Priya
d. Jerry is brown colored.
e. All mammals are animal.




## 3. Frame Representation

→ A frame is a record like structure which consists of a collection of attributes and it values to describe an entity in the world. It consists of a collection of slots and slot values. These slots may be of any type and size. Slots have names and value which

21234747057

DATE: / /
PAGE:

are called facts.

→ In a frame, knowledge about an object or event can be stored together in the knowledge base.

Example:-

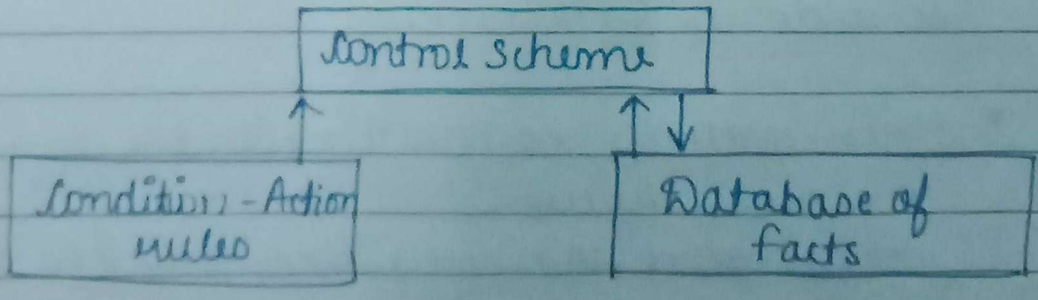
mammal :
subclass : animal
elephant :
subclass : mammal
size : large
haspart : trunk

Nellie :
instance : elephant
likes : apples.

4. Rule-Based Systems

→ Rule-based systems are based on rules that say what to DO, given various conditions.

IF < this is the case > THEN < do this >




Rule based system architecture

DATE : / /
PAGE :

→ Two main kinds of rule-based systems are-

a) forward chaining - which starts with the facts, and sees what rules apply (and hence what should be done) given the facts.

b) Backward chaining - starts with something to find out, and looks for rules that will help in answering it.

Example :- IF (at bus stop AND bus arrives)
THEN action (get into the bus)

(b) i) !(cut) — • It is used to prevent unwanted backtracking by controlling which subgoals can be reduced.

• Used to guarantee termination or control execution order.

• It succeeds when called, but fails the parent goal when an attempt is made to redo its backtracking.

• This includes committing to the clause containing the cut.
⇒ the goal can only succeed if the clause succeeds.

DATE: / /
PAGE:

example - max(X, Y, X) :- X >= Y, !.
max(X, Y, Y) :- X < Y.

Here it backtracks correctly as well as trims unnecessary choices.

ii) !, fail :- Cut together with fail is a kind of negation. This denotes the statements like A = ~B as B :- A, !, fail.

example - Not equal to

different(X, X) :- !, fail.
different(_, _).

OR

not(G) :- call(G), !, fail.
not(_).

not(G) fails if G succeeds.
not(G) succeeds if G fails.

2123474705 7

DATE: / /
PAGE:

## (c) Alpha-beta Pruning

→ Alpha-beta pruning is a modified version of the minimax algorithm. It is an optimization technique for the minimax algorithm.

→ In the minimax algorithm, the number of game states it has to examine are exponential in depth of the tree. Since, we cannot eliminate the exponent, but we can cut it to half. So, we the technique called pruning by which without checking each node of the game tree we can compute the correct minimax decision. This involves two threshold parameter alpha and beta for future expansion, so it is called apha-beta pruning.

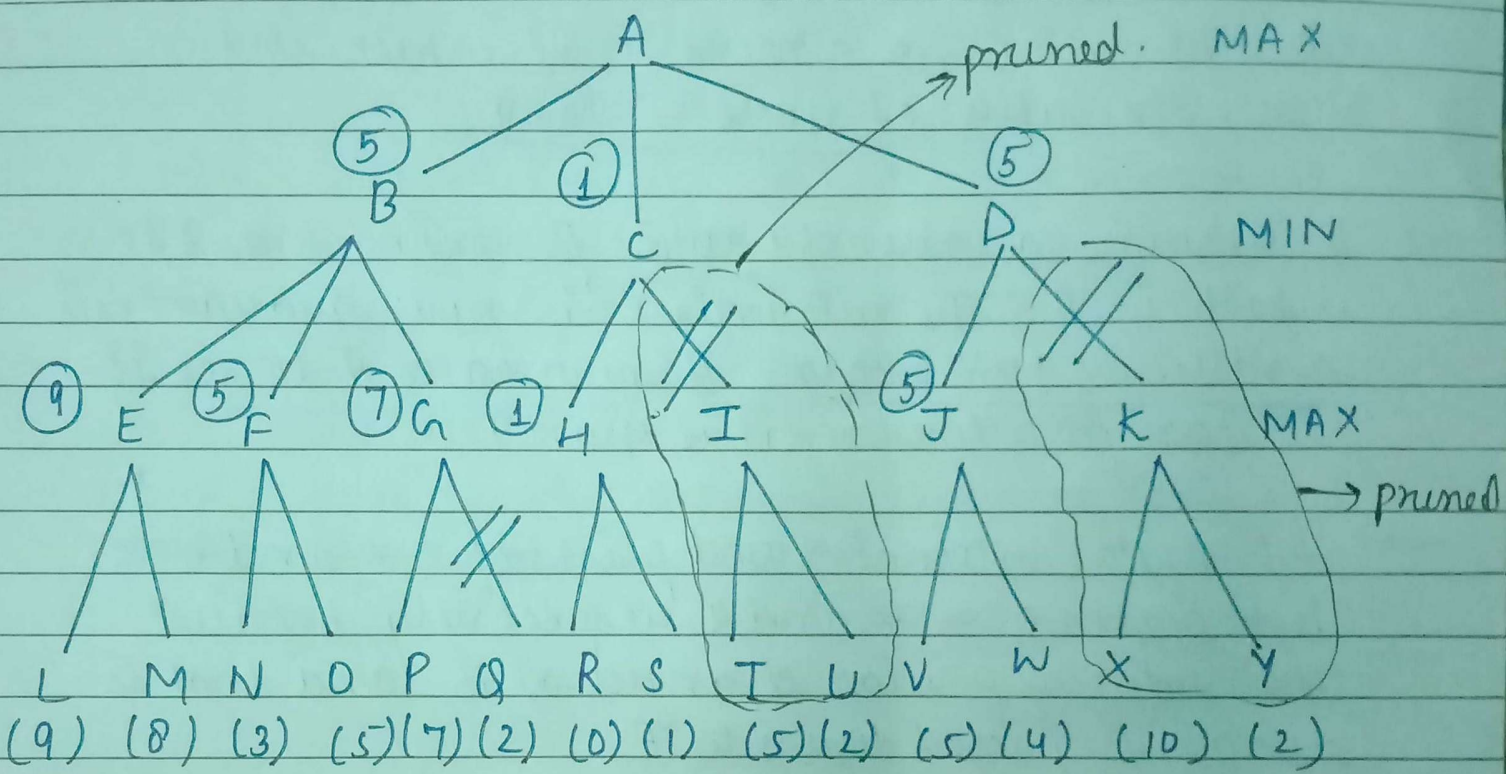
Alpha ($\alpha$) :- The best (highest-value) choice we have found so far at any point along the path of Maximizer. The initial value of $\alpha$ is $-\infty$.

Beta ($\beta$) :- The best (lowest-value) choice we have found so far at any point along the path of Minimizer. The initial value of beta is $+\infty$.

→ Alpha-beta pruning removes all the nodes which are not really affecting the final decision but making algorithm slow. Hence by pruning these nodes, it makes the algorithm fast.
So, it it advantageous to use $\alpha$-$\beta$ pruning procedure

when the depth of the tree is more and also the branching is more in order to make the algorithm fast.

Apha-beta pruning procedure on the given game tree -




→ The initial call starts from A. The value of α here is $-\infty$ and the value of beta is $+\infty$. These values are passed down to subsequent nodes in the tree. At A the maximizer must choose max of B, C and D. So, A calls of B first.

→ At B, it is the minimizer must choose min E, F and G and hence calls E First

→ At E, it looks at its left child which is a leaf node. This node returns a value of 9. Now the value of α at E is max ($-\infty$, 9) which is 9.

→ To decide whether its worth looking at its right node or not, it checks the condition $\beta <= \alpha$. This is false since $\beta = +\infty$ and $\alpha = 9$. So, it continues the search.

→ E now look at its right child which returns a value 8. At E, $\alpha = \max(9, 8)$ which is 9. Now, the value of node E is 9.

→ E returns a value of 9 to B, $\beta = \min(+\infty, 9)$ which is 9. The minimizer is now guaranteed a value of 9 or lesser. B now calls F to see if it can get a lower value than 9.

→ At F the value of $\alpha$ and $\beta$ is not $-\infty$ and $+\infty$ but instead $-\infty$ and 9 respectively, because the value of $\beta$ was changed at B and that is what B passed down to F.

→ Now F looks at its left child which is 3. So now we check the cond. $\beta <= \alpha$. This is false since $\beta = 9$ and $\alpha = 3$. So, it continues the search.

→ F looks at its right child which returns a value of 5. At F, $\alpha = \max(3, 5)$ which is 5. Now, the value of node F is 5.

→ At B, $\beta = \min(9, 5)$ which is 5.

→ B now calls C to see if it can get a better (i.e lower) value than 5.

2123474705 7

DATE: / /
PAGE:

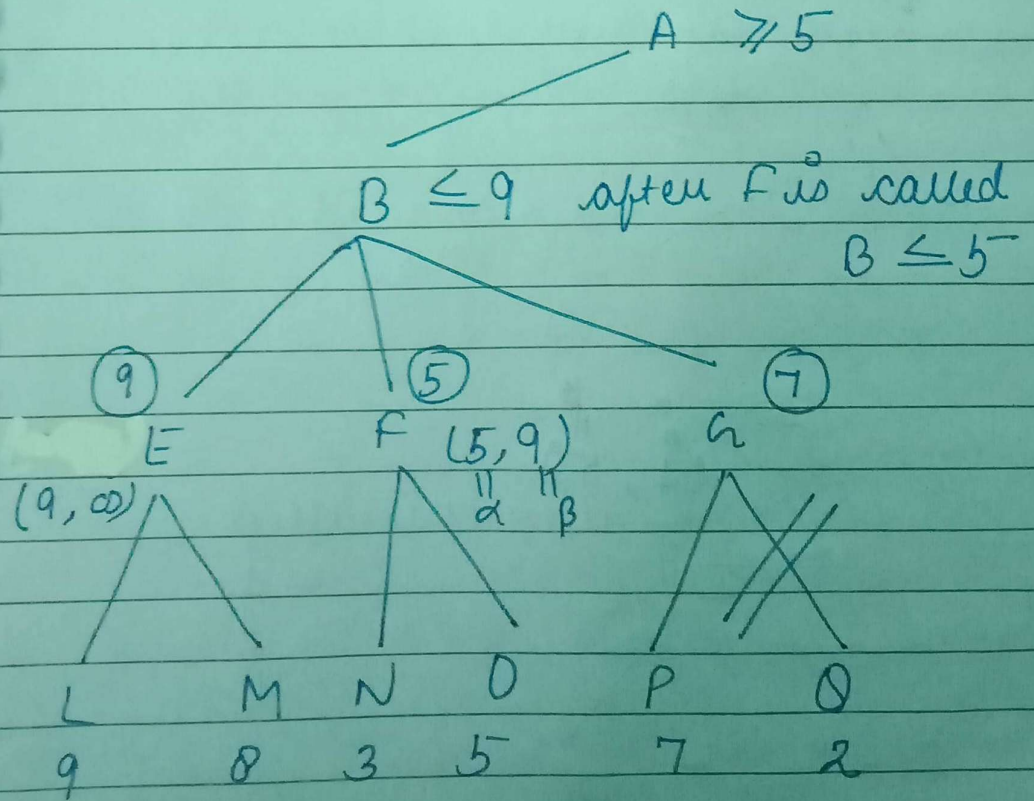
⇒ At G, $\alpha = -\infty$ and $\beta = 5$

→ Now, G looks at its left child which is 7 so, $\beta <=$ alpha is true since $5 < 7$. Hence it breaks and G returns a value of 7 to B, as the value of node G is 7.

→ At B, $\beta = \min(5, 7)$ which is 5.

→ The value of node B is 5.

→ B returns 5 to A. At A, $\alpha = \max(-\infty, 5)$ which is 5. Now maximizer is guaranteed a value of 5 or greater. A now calls C to see if it can get a higher value than 5.



DATE: / /
PAGE:

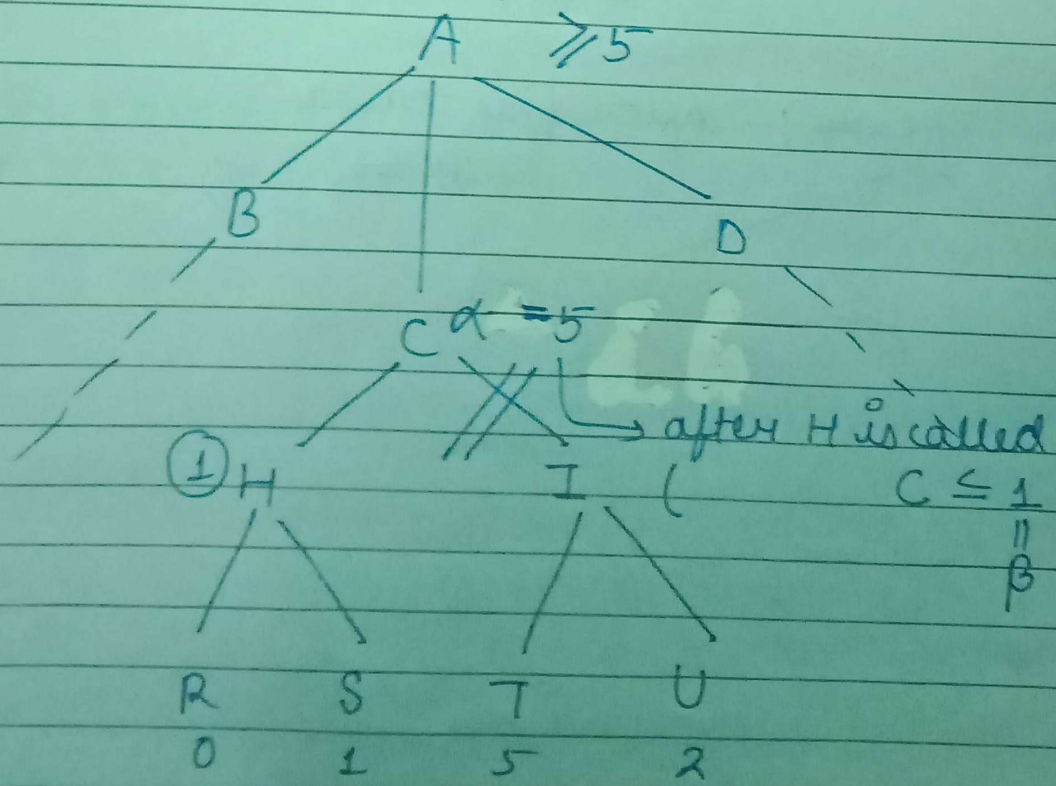
→ At C, $\alpha = 5$ and $\beta = +\infty$. C calls H.

→ At H, $\alpha = 5$ and $\beta = +\infty$. H looks at its left child which is 0 $\alpha = \max(5, 0)$ which is 5.

→ H looks at it's right child which is 1, $\alpha$ still remains 5. The value of node H is 1.

→ H returns a value of 1 to C. At C, $\beta = \min(+\infty, 1)$ which is 1. The condition $\beta <= \alpha$ becomes true as $1 < 5$, it breaks and it does not have to compute the entire subtree of I. The value of node C is 1.

→ C now returns a value of 1 to A. Therefore the best value of A is max (5, 1) which is 5.



→ Now, A again calls D to see if it can get higher value than 5.

→ At D, $\alpha = 5$ and $\beta = +\infty$, D calls J.

→ At J, $\alpha = 5$ and $\beta = +\infty$. J looks at it's left child which is 5. $\alpha = \max(5, 5)$ which is 5 only.

→ We continue the search as $\beta <= \alpha$ is false. J calls its right child which is 4. So, $\alpha = \max(5, 4)$ which is 5. The value of node J is 5.

→ J returns a value of 5 to D. At D, $\alpha = 5$, $\beta = \min(\infty, 5)$ which is 5.

→ At D, $\beta <= \alpha$ i.e $5 <= 5$ is true, and the search is not continued and the entire subtree K is pruned.

So, the nodes that need not to be examined are Q, I, T, U, K, X, Y.