

AI Solutions

SECTION B

1.What is AI?

Ans: AI is the field of science and engineering concerned with the conceptual understanding of what is commonly called intelligent behavior and with the creation of artifacts that exhibits such behavior. Artificial intelligence is intelligence demonstrated by machines, unlike the natural intelligence displayed by humans and animals.

AI has the following features:

- i) Thinking Humanly
- ii) Thinking Rationally
- iii) Acting Humanly
- iv) Acting Rationally

2.What is PEAS? Explain different agent types with their PEAS descriptions.

Ans: PEAS stands for Performance measure, Environment, Actuators, Sensors.

Task environments are problems to which rational agents are the solutions. The task environment is specified by PEAS.

Some of the different agent types with their PEAS description :-

Agent: Medical diagnosis system

- Performance measure: Healthy patient, minimize costs, lawsuits
- Environment: Patient, hospital, staff
- Actuators: Screen display (questions, tests, diagnoses, treatments, referrals)
- Sensors: Keyboard (entry of symptoms, findings, patient's answers)

Agent: Part-picking robot

- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors

Agent: Interactive English tutor

- Performance measure: Maximize student's score on test
- Environment: Set of students
- Actuators: Screen display (exercises, suggestions, corrections)
- Sensors: Keyboard

Some more :-

Agent Types with their PFEAS description				
Agent Type	Performance measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient reduced costs	Patient, hospital, staff	Display of questions, tests, diagnosis, treatments, referrals	Keyboard entry of symptoms, findings, patients' answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Paint-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery operations	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English Tutor	Student is score on test	Sets of students, testing agency	Display of exercises, suggestions, questions	Keyboard entry

3. Explain in detail the properties of Task Environments.

Ans :- Properties of Task Environments:

a) Fully observable vs Partially Observable:

- If an agent sensor can sense or access the complete state of an environment at each point of time then it is a fully observable environment, else it is partially observable.
- A fully observable environment is easy as there is no need to maintain the internal state to keep track of the history of the world.
- An agent with no sensors in all environments then such an environment is called as unobservable.

b) Deterministic vs Stochastic:

- If an agent's current state and selected action can completely determine the next state of the environment, then such an environment is called a deterministic environment.

- A stochastic environment is random in nature and cannot be determined completely by an agent.
- In a deterministic, fully observable environment, an agent does not need to worry about uncertainty.

c) Episodic vs Sequential:

- In an episodic environment, there is a series of one-shot actions, and only the current percept is required for the action.
- However, in Sequential environment, an agent requires memory of past actions to determine the next best actions.

d) Single-agent vs Multi-agent

- If only one agent is involved in an environment, and operating by itself then such an environment is called a single agent environment.
- However, if multiple agents are operating in an environment, then such an environment is called a multi-agent environment.
- The agent design problems in the multi-agent environment are different from single agent environment.

e) Static vs Dynamic:

- If the environment can change itself while an agent is deliberating then such environment is called a dynamic environment else it is called a static environment.
- Static environments are easy to deal because an agent does not need to continue looking at the world while deciding for an action.
- However for a dynamic environment, agents need to keep looking at the world at each action.
- Taxi driving is an example of a dynamic environment whereas Crossword puzzles are an example of a static environment.

f) Discrete vs Continuous:

- If in an environment there are a finite number of percepts and actions that can be performed within it, then such an environment is called a discrete environment else it is called continuous environment.
- A chess game comes under a discrete environment as there is a finite number of moves that can be performed.
- A self-driving car is an example of a continuous environment.

g) Known vs Unknown

- Known and unknown are not actually a feature of an environment, but it is an agent's state of knowledge to perform an action.
- In a known environment, the results for all actions are known to the agent. While in unknown environment, an agent needs to learn how it works in order to perform an action.
- It is quite possible for a known environment to be partially observable and an Unknown environment to be fully observable.

h) Accessible vs Inaccessible

- If an agent can obtain complete and accurate information about the state's environment, then such an environment is called an Accessible environment else it is called inaccessible.
- An empty room whose state can be defined by its temperature is an example of an accessible environment.
- Information about an event on earth is an example of Inaccessible environment.

4. Differentiate Informed & Uninformed search. Give examples.

<u>Roll 1828010</u>	
<u>Informed Search</u>	<u>Uninformed Search</u>
i) It uses knowledge for searching process. ii) It finds solution more quickly. iii) It is highly efficient iv) Cost is low. v) It consumes less time. vi) It provides the direction regarding the solution. vii) It is less lengthy while implementation viii) Greedy Search, A* Search, Graph Search are eg. of informed search.	i) It doesn't use knowledge for searching process. ii) It finds solution slow as compared to informed search iii) It is mandatory efficient iv) Cost is high. v) It consumes moderate time. vi) No suggestions is given regarding the solution in it. vii) It is more lengthy while implementation. viii) Depth first search, Breadth first Search

5.What is Greedy Best First Search? Explain with an example the different stages of Greedy Best First search.

Ans:- Greedy best-first search algorithm always selects the path which appears best at that moment. It is the combination of depth-first search and breadth-first search algorithms. It uses the heuristic function and search. Best-first search allows us to take the advantages of both algorithms. With the help of best-first search, at each step, we can choose the most promising node. In the best first search algorithm, we expand the node which is closest to the goal node and the closest cost is estimated by heuristic function, i.e.

$$1. f(n) = g(n) + h(n).$$

Where, $h(n)$ = estimated cost from node n to the goal.

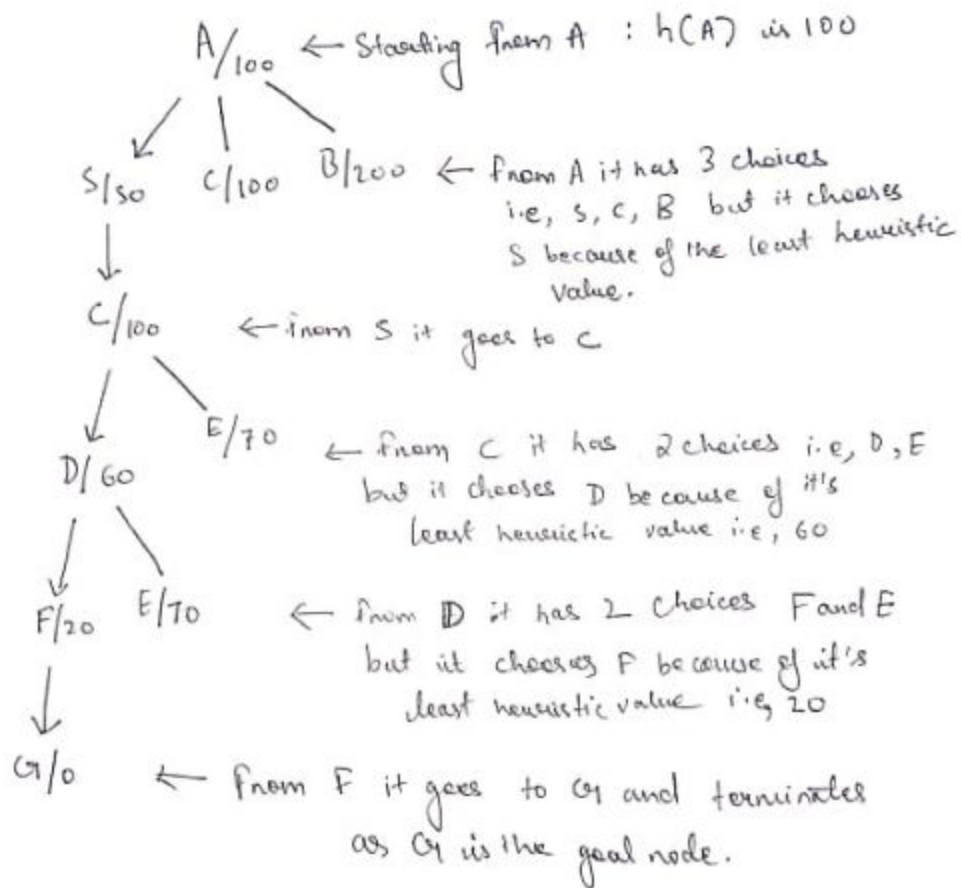
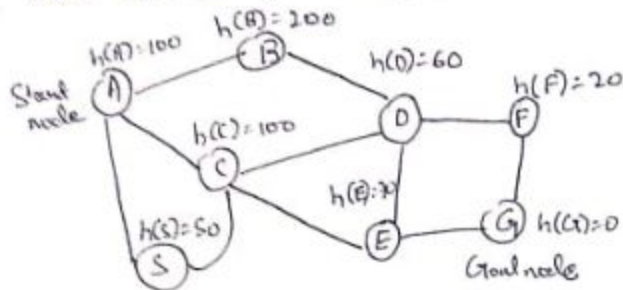
The greedy best first algorithm is implemented by the priority queue.

Example:

Roll 1825090

Example of Greedy BFS

Let's consider this graph with the following heuristic values:



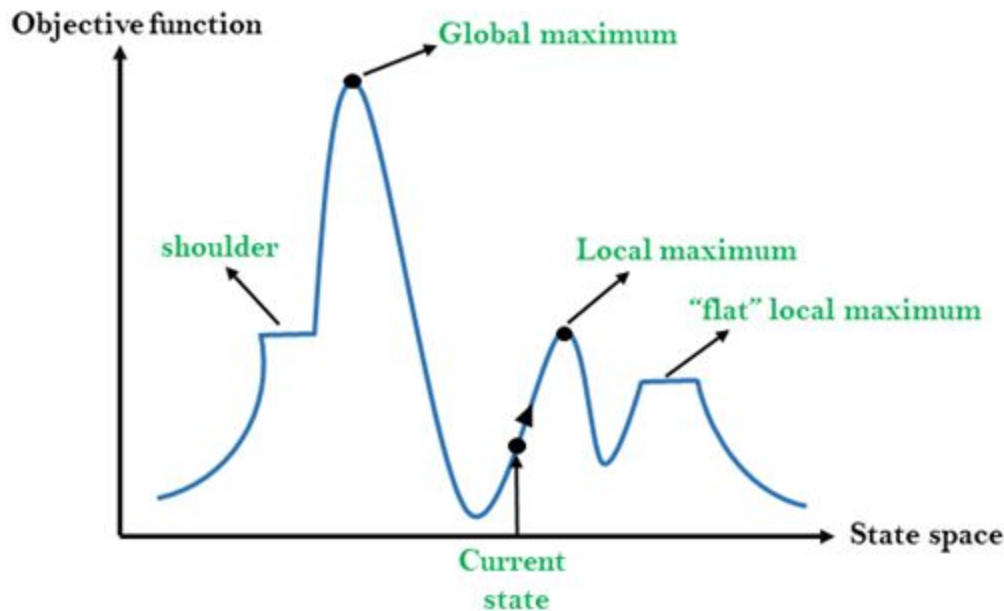
So, the final path is:

A → S → C → D → F → G
 (Start node) (Goal node)

6.Explain the following local search strategy with examples.

Hill climbing.

Ans:- <https://www.javatpoint.com/hill-climbing-algorithm-in-ai>



Local Maximum: Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.

Global Maximum: Global maximum is the best possible state of state space landscape. It has the highest value of objective function.

Current state: It is a state in a landscape diagram where an agent is currently present.

Flat local maximum: It is a flat space in the landscape where all the neighbor states of current states have the same value.

Shoulder: It is a plateau region which has an uphill edge.

7.Define constraint satisfaction problem (CSP). How CSP is formulated as a search problem? Explain with an example.

Ans:-

Roll 1828010

Constraint Satisfaction Problem (CSP)

Definition

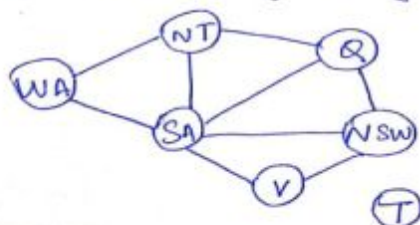
A constraint satisfaction problem (CSP) is a problem that requires its solution within some limitations or conditions also known as constraints.

Formulation

- A finite set of variables which stores the solution.
i.e, $(V = \{V_1, V_2, V_3, \dots, V_n\})$
- A set of discrete values known as domain from which the solution period is picked
i.e, $(D = \{D_1, D_2, D_3, \dots, D_n\})$
- A finite set of constraints
i.e, $(C = \{C_1, C_2, C_3, \dots, C_n\})$

Example: MAP Colouring Problem

Let's consider the following MAP:



For this MAP the CSP formulation will be the following:-

1. Variables:

$$V = \{WA, NT, Q, NSW, V, SA, T\}$$

2. Domains:

$$D = \{\text{red}, \text{Green}, \text{Blue}\}$$

3. Constraints:

Adjacent vertices should not have same colour:

$$C = \{SA \neq WA, SA \neq NT, SA \neq Q, SA \neq NSW, SA \neq V, \\ WA \neq NT, NT \neq Q, Q \neq NSW, NSW \neq V\}$$

* Here we use abbreviations: $SA \neq WA$ is a shorthand for $\{(SA, WA), (WA, SA)\}$, where $WA \neq NT, NT \neq$

$SA \neq WA$ can be fully enumerated in turn as:
 $\{(\text{red}, \text{green}), (\text{red}, \text{blue}), (\text{blue}, \text{green}), (\text{blue}, \text{red}), (\text{blue}, \text{green})\}$

* There are many possible solutions to this problem, such as:

$$\{WA = \text{red}, NT = \text{green}, Q = \text{red}, NSW = \text{green}, V = \text{red}, SA = \text{blue}, T = \text{red}\}$$

8. Illustrate the use of first-order logic to represent knowledge.

Ans:- First Order Logic is also called predicate calculus. It is an AI language representation that has: – Well-defined formal semantics and – Sound and complete inference rules.

It offers a formal approach to reasoning with a sound theoretical foundation. It provides flexibility in representing natural language. It is widely accepted by the workers in the AI field as one of the most useful representation methods.

It is used to represent knowledge in the following way:-

Identify the task Assemble the relevant knowledge Decide on a vocabulary of predicates, functions, and constants Encode general knowledge about the domain Encode a description of the specific problem instance Pose queries to the inference procedure and get answers Debug the knowledge base.

9. (a) Define the syntactic elements of first-Order logic (b) Illustrate the use of first-order logic to represent knowledge.

Ans:- Syntactic Elements of first order logic :-

- Connectives \neg , \Rightarrow , \wedge , \vee , \Leftrightarrow
- Equality =
- Constants : they are fixed value terms, belong to a given domain. Example:- KingJohn, 2, NUS,...
- Quantifiers \forall (universal), \exists (existential)
- Auxiliary symbols : like $()$, $[\]$, $\{ \}$ are used for punctuation.
- Variables : they are terms that can assume different values over a given domain. It is denoted by letters x, y, a, b,...
- Functions : function symbols defined over a domain map n elements ($n > 0$) to a single element of the domain. Here n is called the rank or degree of a function. Examples :- Sqrt, LeftLegOf,...
- terms : constant variables and functions are called terms
- Predicates : they denote relations or functional mapping from the elements of a domain to the values true or false. For example :- Brother, >, Like functions, predicates can have n ($n \geq 0$) terms as arguments. A 0-ary predicate is a proposition. i.e. propositions are constant predicates.

It is used to represent knowledge in the following way:-

Identify the task Assemble the relevant knowledge Decide on a vocabulary of predicates, functions, and constants Encode general knowledge about the domain Encode a description of the specific problem instance Pose queries to the inference procedure and get answers Debug the knowledge base.

10. Explain with algorithm and example : Minimax algorithm.

Ans:- Minimax Algorithm deals with the contingency problem.

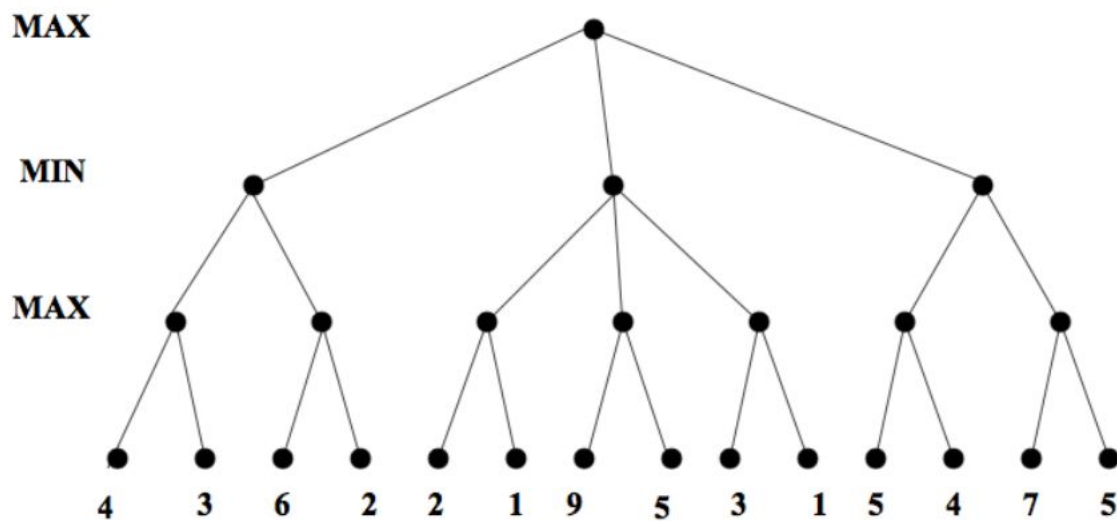
Assuming the opponent is always rational and always optimizes its behavior (opposite to us), we consider the best opponent's response then the minimax algorithm determines the best move.

Perfect play for deterministic games.

Idea: choose move to position with highest minimax value = best achievable payoff against best play

Example:-

Minimax. Example



8

11.Explain forward chaining with example.

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 starts 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 adds their conclusion to the known facts. This process repeats until the problem is solved.

For example -:

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

Prove that "Robert is criminal."

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

Facts Conversion into FOL:

o It is a crime for an American to sell weapons to hostile nations. (Let's say p, q, and r are variables)

American(p) \wedge weapon(q) \wedge sells(p, q, r) \wedge hostile(r) \rightarrow Criminal(p) ... (1)

o Country A has some missiles. $\exists p$ Owns(A, p) \wedge Missile(p). It can be written in two definite clauses by using Existential Instantiation, introducing new Constant T1.

Owns(A, T1) (2)

Missile(T1) (3)

o All of the missiles were sold to country A by Robert.

$\forall p$ Missiles(p) \wedge Owns(A, p) \rightarrow Sells(Robert, p, A) (4)

o Missiles are weapons.

Missile(p) \rightarrow Weapons(p) (5)

o Enemy of America is known as hostile.

Enemy(p, America) \rightarrow Hostile(p) (6)

o Country A is an enemy of America.

Enemy(A, America) (7)

o Robert is American

American(Robert). (8)

Forward chaining proof:

Step-1:

In the first step we will start with the known facts and will choose the sentences which do not have implications, such as: American(Robert), Enemy(A, America), Owns(A, T1), and Missile(T1). All these facts will be represented as below.



Step-2:

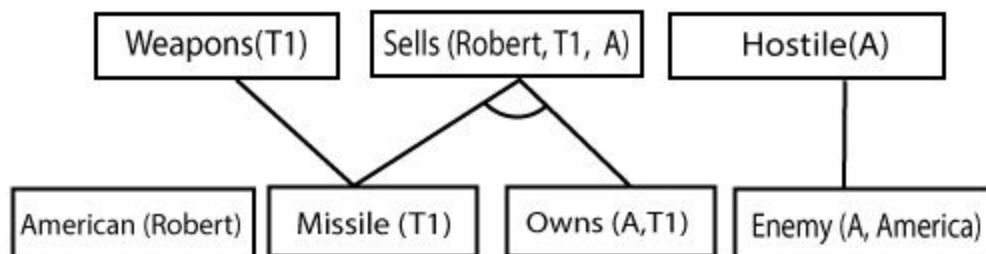
At the second step, we will see those facts which infer from available facts and with satisfied premises.

Rule-(1) does not satisfy premises, so it will not be added in the first iteration.

Rule-(2) and (3) are already added.

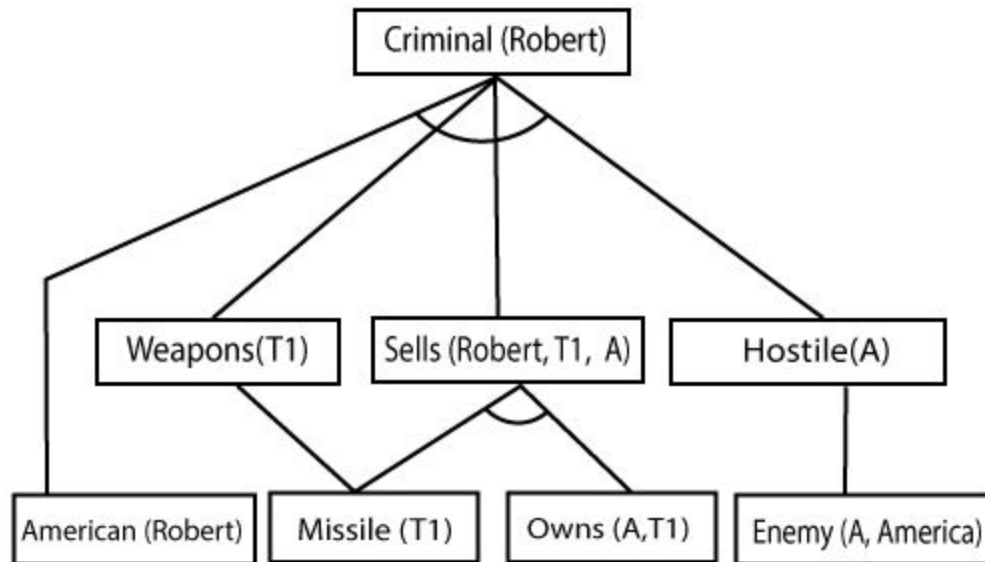
Rule-(4) satisfies the substitution $\{p/T1\}$, so Sells (Robert, T1, A) is added, which infers from the conjunction of Rule (2) and (3).

Rule-(6) is satisfied with the substitution (p/A) , so Hostile(A) is added and which infers from Rule-(7).



Step-3:

At step-3, as we can check Rule-(1) is satisfied with the substitution $\{p/Robert, q/T1, r/A\}$, so we can add Criminal(Robert) which infers all the available facts. And hence we reached our goal statement.



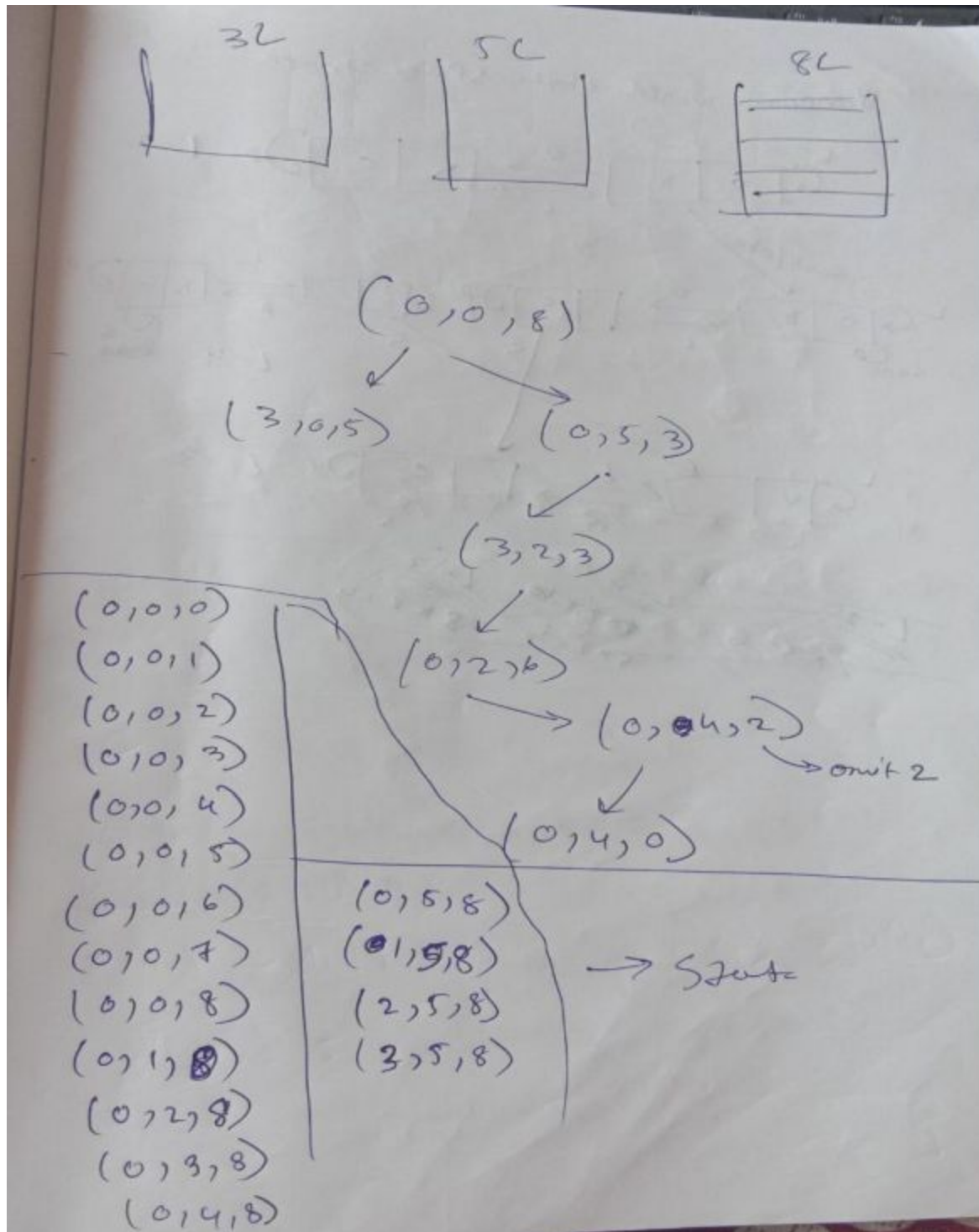
Hence it is proved that Robert is Criminal using a forward chaining approach.

12.(a) Suppose, there are 3 jugs of capacities 8, 5 and 3 litres respectively. There is no scale

on the jugs, so it's only their capacities that is known. Initially the 8 litre jug is full of water the other two jugs are empty. The water can be poured from one jug to another. The goal is to have exactly 4 litre of water in any of the jugs. The amount of the water in the other two jugs at the end is irrelevant.

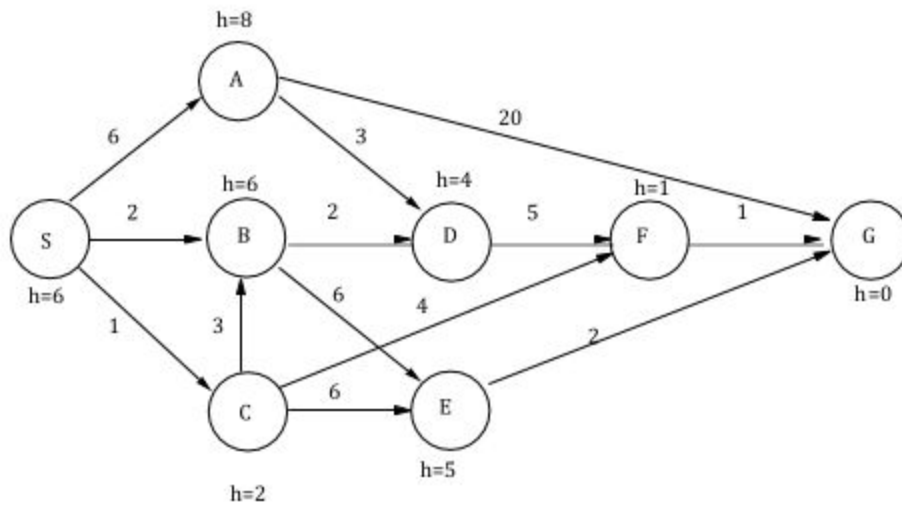
Formulate this problem as a state space search problem and draw the state space graph of this problem.

Ans:-



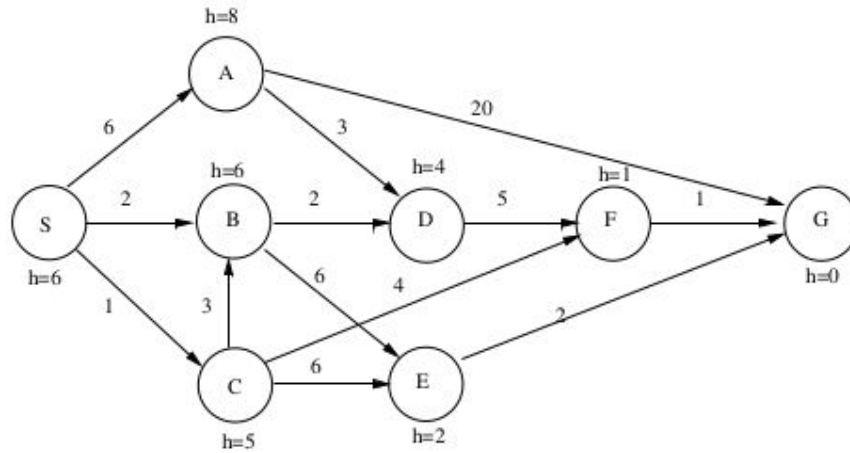
12.(b) Consider the search problem below with start state S and goal state G. The transition costs are next to the edges, and the heuristic values are above the states.
 (i) What is the path if the best first search algorithm is used to reach the goal?

- (ii) What is the path if depth first search is used? If a node has multiple successors, then expand the successors in increasing alphabetical order
- (iii) If A* algorithm is used, what is the path?
- (iv) Is the heuristic function in this problem admissible?



Ans:-

Consider the search problem below with start state S and goal state G . The transition costs are next to the edges, and the heuristic values are next to the states.



If we use Uniform-Cost Search:

(a) What is the final path for this search?

Answer: $S \rightarrow C \rightarrow F \rightarrow G$

If we use Depth First Search, and it terminates as soon as it reaches the goal state:

(b) What is the final path for this DFS search? If a node has multiple successors, then we always expand the successors in increasing alphabetical order.

Answer: $S \rightarrow A \rightarrow D \rightarrow F \rightarrow G$

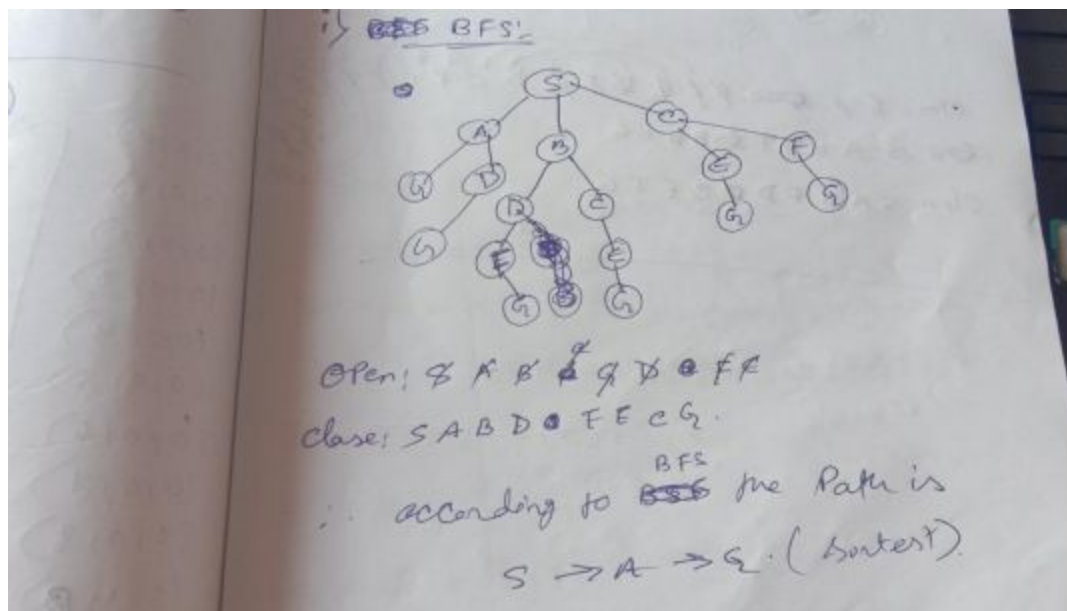
If we use A* search:

(c) What is the final path for this A* search?

Answer: $S \rightarrow C \rightarrow F \rightarrow G$

(d) Is the heuristic function in this example admissible?

Answer: Yes, since $h(s) \leq h^*(s)$ for all states s .



Open: ~~S~~ A B ~~D~~ E B A G E B A.

Close: S C F G.

ii) In Cases of DFS & expanding in increasing alphabetical order.

S A D F G.

∴ Solⁿ is S A D F G.

B
G
B
S C

iii) For the case of A^* Algorithm we will consider $g(n) \leftarrow$ path cost
 $h(n) \leftarrow$ heuristic cost

iii) Ans Algo:

$$S \rightarrow A \quad f(A) = (6+8) = 14$$

$$S \rightarrow A \rightarrow G \quad f(A) = (6+20+0) = 26$$

$$S \rightarrow A \rightarrow D \quad f(A) = (6+3+4) = 13$$

$$S \rightarrow B \quad f(B) = (2+6) = 8$$

$$S \rightarrow B \rightarrow D \quad f(D) = (2+2+4) = 8$$

$$S \rightarrow B \rightarrow D \rightarrow F \quad f(F) = (2+2+5+1) = 10$$

$$S \rightarrow B \rightarrow D \rightarrow F \rightarrow G \quad f(G) = (2+2+5+1+0) = 10$$

$$\cancel{S \rightarrow B \rightarrow D \rightarrow F \rightarrow G} \quad S \rightarrow B \rightarrow E \quad f(E) = (2+6+5) = 13$$

$$\cancel{S \rightarrow C \quad f(C) = (1+2) = 3}$$

$$S \rightarrow C \rightarrow$$

$$S \rightarrow B \rightarrow E \rightarrow G \\ = (2+6+2+0) \\ = 10$$

$$S \rightarrow C \quad f(C) = (1+2) = 3$$

$$S \rightarrow C \rightarrow B \quad f(B) = (1+3+6) = 10$$

$$S \rightarrow C \rightarrow B \rightarrow D \quad f(D) = (1+3+2+4) = 10$$

$$S \rightarrow C \rightarrow B \rightarrow D \rightarrow F \quad f(F) = (1+3+2+5+1) = 12$$

$$S \rightarrow C \rightarrow B \rightarrow D \rightarrow F \rightarrow G \quad f(G) = (1+3+2+5+1+0) = 12$$

$$S \rightarrow C \rightarrow E \quad f(E) = (1+6+5) = 12$$

$$S \rightarrow C \rightarrow E \rightarrow G \quad f(G) = (1+6+2+0) = 9$$

\therefore The Path is

$$S \rightarrow C \rightarrow E \rightarrow G$$

iii) As we know, if the $h(n) \leq h^*(n)$ then it is admissible.

In this case,

$$h(n) = 9, \text{ and } h^*(n) = 9$$

$$\therefore h(n) \leq h^*(n)$$

\therefore It is admissible.

14. Explain Task environment, State space and PEAS representation. Give the PEAS representation for the Vacuum-Cleaner world along with its State space diagram. Write its Agent function. Give some examples of the percept sequence and action mapping.

Ans:- Task environment - Designing intelligent agents. • an agent operates in a task environment: – task: the goal(s) the agent is trying to achieve. – environment: that part of the real world or a computational. system 'inhabited' by the agent.

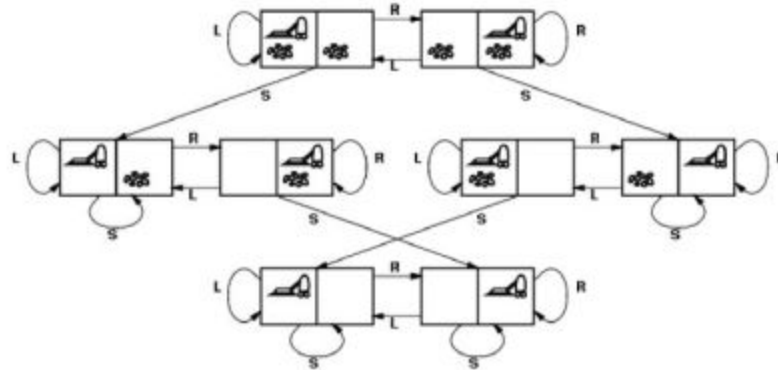
State space - State space search is a process used in the field of computer science, including artificial intelligence (AI), in which successive configurations or states of an instance are considered, with the intention of finding a goal state with a desired property.

PEAS- PEAS stands for Performance measure, Environment, Actuator, Sensor. ... Performance Measure: Performance measure is the unit to define the success of an agent. Performance varies with agents based on their different precepts.

PEAS representation of vacuum cleaner-:

- Performance: cleanness, efficiency: distance traveled to clean, battery life, security
- Environment: room, table, wood floor, carpet, different obstacles.
- Actuators: wheels, different brushes, vacuum extractor.
- Sensors: camera, dirt detection sensor, cliff sensor, bump sensors, infrared wall sensors.

VACUUM WORLD STATE SPACE GRAPH



- states? integer dirt and robot location
- actions? *Left, Right, Suck*
- goal test? no dirt at all locations
- path cost? 1 per action

Agent function of vacuum cleaner agent:

function Vacuum-Agent([location,status]) return an action

if status = Dirty then return Suck

else if location = A then return Right

else if location = B then return Left

Percept sequence is the complete history of everything the agent has ever perceived.
For example, vacuum cleaners precept that there is dirt present in the left block.

Action mapping: based on the percept sequence the vacuum agent performs cleaning action in the left block.

15. Describe various Consistencies for Constraint Propagation in CSPs. Write the Arc consistency algorithm with an example showing its use.

Ans:- There are different types of local consistency:

Node consistency

A single variable (a node in the CSP network) is node-consistent if all the values in the variable's domain satisfy the variable's unary constraint.

We say that a network is node-consistent if every variable in the network is node-consistent.

Arc consistency

A variable in a CSP is arc-consistent if every value in its domain satisfies the variable's binary constraints.

X_i is arc-consistent with respect to another variable X_j if for every value in the current domain D_i there is some value in the domain D_j that satisfies the binary constraint on the arc (X_i, X_j) .

A network is arc-consistent if every variable is arc-consistent with every other variable.

Arc consistency tightens down the domains (unary constraint) using the arcs (binary constraints).

Path consistency

Path consistency: A two-variable set $\{X_i, X_j\}$ is path-consistent with respect to a third variable X_m if, for every assignment $\{X_i = a, X_j = b\}$ consistent with the constraint on $\{X_i, X_j\}$, there is an assignment to X_m that satisfies the constraints on $\{X_i, X_m\}$ and $\{X_m, X_j\}$.

Path consistency tightens the binary constraints by using implicit constraints that are inferred by looking at triples of variables.

K-consistency


K-consistency: A CSP is k-consistent if, for any set of k-1 variables and for any consistent assignment to those variables, a consistent value can always be assigned to any kth variable.

1-consistency = node consistency; 2-consistency = arc consistency; 3-consistency = path consistency.

A CSP is strongly k-consistent if it is k-consistent and is also (k - 1)-consistent, (k - 2)-consistent, ... all the way down to 1-consistent.

A CSP with n nodes and making it strongly n-consistent, we are guaranteed to find a solution in time $O(n^2d)$. But an algorithm for establishing n-consistency must take time exponential in n. In the worst case, it also requires space that is exponential in n.

Arc algo-:

Consider the network of  There are three variables A, B, and C, each with domain {1,2,3,4}. The constraints are $A < B$ and $B < C$. In the constraint network, shown in [figure](#), there are four arcs:

$\langle A, A < B \rangle$

$\langle B, A < B \rangle$

$\langle B, B < C \rangle$

$\langle C, B < C \rangle$

. None of the arcs are arc consistent. The first arc is not arc consistent because for $A=4$ there is no corresponding value for B for which $A < B$. If 4 were removed from the domain of A, then it would be arc consistent. The second arc is not arc consistent because there is no corresponding value for A when $B=1$.

If an arc $\langle X, c \rangle$ is *not* arc consistent, there are some values of X for which there are no values for Y_1, \dots, Y_k for which the constraint holds. In this case, all values of X in D_X for which there are no

corresponding values for the other variables can be deleted from D_x to make the arc $\langle X, c \rangle$ consistent.

1: Procedure GAC(V, dom, C)

2: Inputs

3: V : a set of variables

4: dom : a function such that $dom(X)$ is the domain of variable X

5: C : set of constraints to be satisfied

6: Output

7: arc-consistent domains for each variable

8: Local

9: D_x is a set of values for each variable X

10: TDA is a set of arcs

11: for each variable X do

12: $D_x \leftarrow dom(X)$

13: $TDA \leftarrow \{ \langle X, c \rangle \mid c \in C \text{ and } X \in scope(c) \}$

14: while ($TDA \neq \{ \}$)

15: select $\langle X, c \rangle \in TDA$;

16: $TDA \leftarrow TDA \setminus \{ \langle X, c \rangle \}$;

17:

$ND_x \leftarrow \{ x \mid x \in D_x \text{ and some } \{ X=x, Y_1=y_1, \dots, Y_k=y_k \} \in c \text{ where } y_i \in D_{Y_i} \text{ for all } i \}$

18: if ($ND_x \neq D_x$) then

19: $TDA \leftarrow TDA \cup \{ \langle Z, c' \rangle \mid X \in scope(c'), c' \text{ is not } c, Z \in scope(c') \setminus \{ X \} \}$

20: $D_x \leftarrow ND_x$

21: return $\{ D_x \mid X \text{ is a variable} \}$

16. List and explain the methodologies that can be adopted to improve the Backtracking algorithm used to solve CSPs. Give appropriate examples for each.

Ans:- For Improving Backtracking efficiency general purpose methods can give huge gains in speed:-

The questions which arise in the process :-

- Which variable should be assigned next? -> Choose the variable with the most constraints on remaining variables

- In what order should its values be tried? -> Forward Checking -> Keep track of remaining legal values for unassigned variables and terminate search when any variable has no legal values. Variable ordering and value selection heuristics help significantly

- Can we detect inevitable failure early? -> Forward checking prevents assignments that guarantee later failure.

Examples :- PPT

17. Define Constraint Satisfaction Problems along with its components. Solve the following Cryptarithmic Problem, properly defining its variables, domains and constraints involved-

EAT

THAT

APPLE

Draw the constraint graph of the same.

Ans:-

18/28/18

> Constraint Satisfaction Problems are those Problems which satisfy a given Constraint or set of Constraints to solve a given Problem. following are the Components of the c.s.p.

$X = \langle x_1, x_2, x_3, \dots, x_n \rangle \equiv$ Set of variables

$D = \langle D_1, D_2, \dots, D_n \rangle \equiv$ Domain of variables

$R = \langle D_1 \times D_2 \times D_3 \dots \times D_n \rangle$

for the given Cryptarithmic Problem, following is the solution.

$$\begin{array}{r} \text{EAT} \\ + \text{THAT} \\ \hline \text{APPLE} \end{array}$$

So, here 'A' will be 1
& 'T' will be 9

$$\begin{array}{r} \text{So Now} \quad \text{E} \ 1 \ 9 \\ \quad \quad \text{9} \ 1 \ 9 \\ \hline \quad \quad 1 \ 1 \ 1 \ 8 \end{array}$$

Case, $E = 8$
 $A = 1$
 $T = 9$
 $L = 3$
 $H = 2$
 $P = 0$

$$\begin{array}{r} \therefore \quad 8 \ 1 \ 9 \\ + \quad 9 \ 2 \ 1 \ 9 \\ \hline \quad 1 \ 0 \ 0 \ 3 \ 8 \end{array}$$

18. Write the algorithm for Backtracking search in CSPs. What are its drawbacks ? What is Backjumping and how it is better than backtracking ? Explain by taking the example of 4-Queens problems.

Ans:-

```
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return RECURSIVE-BACKTRACKING({}, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(Variables[csp], assignment, csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
    if value is consistent with assignment according to Constraints[csp] then
      add { var = value } to assignment
      result ← RECURSIVE-BACKTRACKING(assignment, csp)
      if result ≠ failure then return result
      remove { var = value } from assignment
  return failure
```

Drawbacks of Backtracking:-

There are three major drawbacks of the backtracking are:- . One is thrashing, i.e., repeated failure due to the same reason. Thrashing occurs because the standard backtracking algorithm does not identify the real reason of the conflict, i.e., the conflicting variables.

The other drawback of backtracking is having to perform redundant work. Even if the conflicting values of variables are identified during the intelligent backtracking, they are not remembered for immediate detection of the same conflict in a subsequent computation.

Finally, the basic backtracking algorithm still detects the conflict too late as it is not able to detect the conflict before the conflict really occurs, i.e., after assigning the values to the all variables of the conflicting constraint.

Backjumping is a technique that reduces search space, therefore increasing efficiency. While backtracking always goes up one level in the search tree when all values for a variable have been tested, backjumping may go up more levels.

Backjumping Advantages over Backtracking :-

At every internal node, a set of variables is maintained.

When further backjumping from the node, the variable of the node is removed from this set, and the set is sent to the node that is the destination backjumping.

This algorithm works because the set maintained in a node collects all variables that are relevant to prove unsatisfiability in the leaves that are descendants of this node. Since sets of variables are only sent when retracing from nodes, the sets collected at nodes skipped by backjumping are automatically ignored.

Example:- 4- Queens Problem PPT

19. How the Local search can be applied to solve CSPs. Write and explain 'Min-Conflict Algorithm'. Give the 4-Queens problem formulation and solve it using local Search.

Ans:- Local search for CSPs:-

Local search algorithms for CSPs use a complete-state formulation: the initial state assigns a value to every variable, and the search changes the value of one variable at a time.

```
algorithm MIN-CONFLICTS is
  input: csp, A constraint satisfaction problem.
           max_steps, The number of steps allowed before giving up.
           current_state, An initial assignment of values for the
variables in the csp.
  output: A solution set of values for the variable or failure.

  for i ← 1 to max_steps do
    if current_state is a solution of csp then
      return current_state
    set var ← a randomly chosen variable from the set of conflicted
variables CONFLICTED[csp]
    set value ← the value v for var that minimizes
CONFLICTS(var, v, current_state, csp)
    set var ← value in current_state

  return failure
```

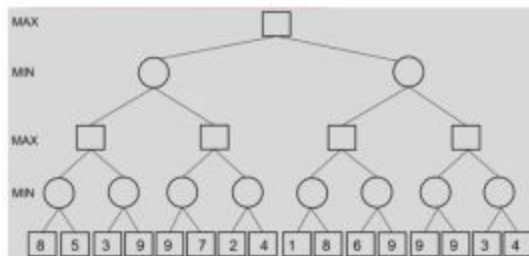
In computer science, the min conflicts algorithm is a search algorithm or heuristic method to solve constraint satisfaction problems (CSP).

Given an initial assignment of values to all the variables of a CSP, the algorithm randomly selects a variable from the set of variables with conflicts violating one or more constraints of the CSP.^[1] Then it assigns to this variable the value that minimizes the number of conflicts. If there is more than one value with a minimum number of conflicts, it chooses one randomly. This process of random variable selection and min-conflict value assignment is iterated until a solution is found or a pre-selected maximum number of iterations is reached.

4-Queens Problem using local Search:-

- States: 4 queens in 4 columns ($4^4 = 256$ states)
- Actions: move queen in column
- Goal test: no attacks
- Evaluation: $h(n)$ = number of attacks
- Given random initial state, can solve n-queens in almost constant time for arbitrary n with high probability (e.g., $n = 10,000,000$)

20. Write the Alpha-Beta Algorithm. Explain Alpha cut and Beta Cut briefly. Solve the following example and show how alpha-beta pruning helped in pruning the search tree.



Ans

2) a) α - β algorithm;

18/28/18

function minimax(node, depth, alpha, beta,
~~maximizing~~ player)

if depth = 0 or node is terminal node then
return static evaluation of node.

if Maximizing Player, then.

maxEva = $-\infty$.

for each child of node do

eva = minimax(child, depth-1, alpha, beta,
False)

maxEva = max(maxEva, eva)

alpha = max(alpha, maxEva)

if beta \leq alpha

break

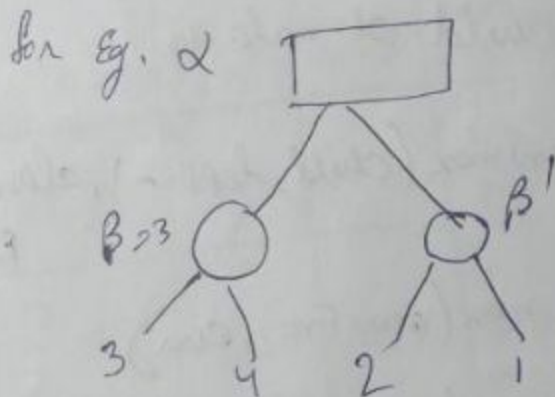
return maxEva.

else

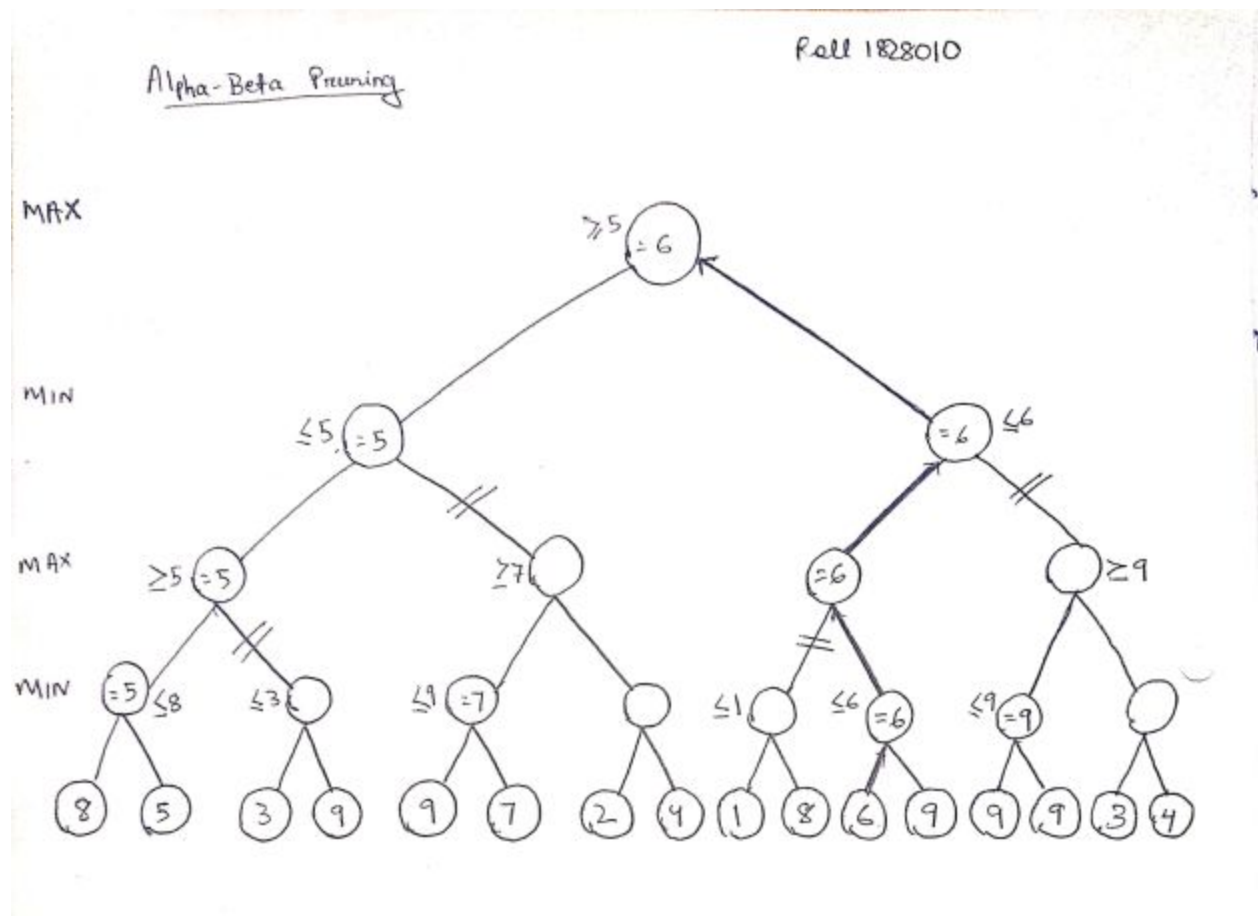
minEva = $+\infty$

if ($\beta \leq \alpha$)
 break
 return minVal.

b) The above α - β has a Problem
 i.e., its huge binding factor
 so there is an improvement to
 it this is what we call as α - β
 Pruning.



Starting from left most we have $\beta=3$
 & to fill α block we traverse other
 half & since first end node is less
 than $\beta=3$, so whatever be the
 case $\beta' \leq 3$. Thus we need not
 explore.



21. Explain the scenario of 'Imperfect Real Time' decisions. What methodologies could be adopted to deal with such situations? Explain by giving some examples of such Scenario.

Ans:- Imperfect, Real-Time Decisions and methodologies which can be used to deal such situations:-

- Searching through the whole (pruned) game tree is too inefficient for any realistic game •
- Moves must be made in a reasonable amount of time
- One has to cut off the generation of the game tree to some depth and the absolute terminal node values are replaced by heuristic estimates
- Game positions are rated according to how good they appear to be (with respect to reaching a goal state) •

A basic requirement for a heuristic evaluation function is that it orders the terminal states in the same way as the true utility function •

Of course, evaluation of game positions may not be too inefficient and the evaluation function should be strongly correlated with the actual chances of winning

22. Write the Minimax Algorithm and explain 'Evaluation function' and 'Optimal Strategy'. Take an example of Tic-tac-toe game and explain how evaluation function can

be used to find the next best move.

Ans:-

```
function minimax(node, depth, maximizingPlayer) is
  if depth = 0 or node is a terminal node then
    return the heuristic value of node
  if maximizingPlayer then
    value := -∞
    for each child of node do
      value := max(value, minimax(child, depth - 1, FALSE))
    return value
  else (* minimizing player *)
    value := +∞
    for each child of node do
      value := min(value, minimax(child, depth - 1, TRUE))
    return va
```

A minimax algorithm is a recursive algorithm for choosing the next move in an n-player game, usually a two-player game. A value is associated with each position or state of the game.

This value is computed by means of a position evaluation function and it indicates how good it would be for a player to reach that position. The player then makes the move that maximizes the minimum value of the position resulting from the opponent's possible following moves. If it is A's turn to move, A gives a value to each of their legal moves.

For non terminal leaf nodes at the maximum search depth, an evaluation function estimates a heuristic value for the node. The quality of this estimate and the search depth determine the quality and accuracy of the final minimax result.

Optimal Strategy for Minimax:-

Idea: choose move to position with highest minimax value = best achievable payoff against best play.

Example:- PPT- Tic Tac Toe for Optimal Function

23. What is a Logical Agent? What are its properties? Give the PEAS representation and characteristics feature for the 'Wumpus world' problem. Take an example and explain how the logical agent can solve the problem.

Ans:- Logical agents apply inference to a knowledge base to derive new information and make decisions

- Basic properties of logical agents:
 - syntax: formal structure of sentences
 - semantics: truth of sentences w.r.t models
 - entailment: necessary truth of one sentence given another
 - inference: deriving sentences from other sentences

- soundness: derivations produce only entailed sentences
- completeness: derivations can produce all entailed sentences

WUMPUS World PEAS Description:-

- Performance measure – gold +1000, death -1000 – -1 per step, -10 for using the arrow
- Environment – Squares adjacent to wumpus are smelly
 - Squares adjacent to pit are breezy
 - Glitter iff gold is in the same square
- Shooting kills wumpus if you are facing it
 - Shooting uses up the only arrow
 - Grabbing picks up gold if in same square
 - Releasing drops the gold in same square
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot
- Sensors: Stench, Breeze, Glitter, Bump, Scream

WUMPUS World Features:-

- Fully Observable No – only local perception
- Deterministic Yes – outcomes exactly specified
- Episodic No – sequential at the level of actions
- Static Yes – Wumpus and Pits do not move
- Discrete Yes
- Single-agent? Yes – Wumpus is essentially a natural feature

Example:-

- Marcus was a man. $\text{man}(\text{Marcus})$
- Marcus was a Pompeian. $\text{pompeian}(\text{Marcus})$
- All Pompeians were Romans. $\forall x [\text{Pompeian}(x) \Rightarrow \text{Roman}(x)]$
- Caesar was a ruler. $\text{ruler}(\text{Caesar})$
- All Romans were either loyal to Caesar or hated him. $\forall x [\text{Roman}(x) \Rightarrow \text{loyalto}(x, \text{Caesar}) \vee \text{hate}(x, \text{Caesar})]$

24. State and explain clearly, the Knowledge Engineering process in First order Logic. Explain the ‘Quantifiers’ used in FOL giving some examples.

Ans :- Knowledge Engineering in FOL :-

Identify the task Assemble the relevant knowledge

Decide on a vocabulary of predicates, functions, and constants

Encode general knowledge about the domain

Encode a description of the specific problem instance Pose queries to the inference procedure and get answers

Debug the knowledge base

Quantifiers used in FOL :-

Quantifiers:- \forall (universal), \exists (existential).

Properties of Quantifiers:-

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- $\exists x \forall y$ is not the same as $\forall y \exists x$
- $\exists x \forall y \text{ Loves}(x,y)$ – “There is a person who loves everyone in the world”
- $\forall y \exists x \text{ Loves}(x,y)$ – “Everyone in the world is loved by at least one person”
- Quantifier duality: each can be expressed using the other
- $\forall x \text{ Likes}(x, \text{IceCream}) \neg \exists x \neg \text{Likes}(x, \text{IceCream})$
- $\exists x \text{ Likes}(x, \text{Broccoli}) \neg \forall x \neg \text{Likes}(x, \text{Broccoli})$

Equality: $\text{term1} = \text{term2}$ is true under a given interpretation if and only if term1 and term2 refer to the same object

• E.g., $\text{Father}(\text{John}) = \text{Henry}$ The kinship domain (the domain of family relationships) : definition of Sibling in terms of Parent: $\forall x,y \text{ Sibling}(x,y) \Leftrightarrow [\neg(x = y) \wedge \exists m,f \neg (m = f) \wedge \text{Parent}(m,x) \wedge \text{Parent}(f,x) \wedge \text{Parent}(m,y) \wedge \text{Parent}(f, y)]$

25. Explain how the Planning problems can be represented. Explain the ‘Block’s World’ problem and how it can be solved using Total ordered planning.

Ans:-

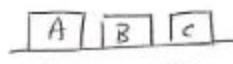
Blocks World Problem (Solved using Total Order Planning)

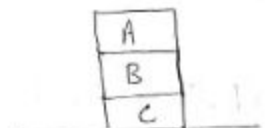
Example:


The block world problem consists of a set of cube shaped blocks sitting on a table.

A robot can pick up only one block at a time and move it to another position, either on the table or on top of another block.

Suppose the goal is to get block A ^{on} B and block B on C.


Initial state


Final/goal state


Intermediate state

Notations used

$On(b, x) \rightarrow$ Block b is on x , where x is either another block or the table.

$Clear(x) \rightarrow$ nothing on top of x

$move(b, x, y) \rightarrow$ moving block b from the top of x to the top of y

- The preconditions for this action is on other blocks be on top of b on y.

In STRIPS we can write:

- Action (move(b, x, y))

- PRECOND: $On(b, x) \wedge clear(b) \wedge clear(y)$

- EFFECT: $On(b, y) \wedge clear(x) \wedge \sim On(b, x) \wedge \sim clear(y)$

To Build a 3 block tower one solution is:-

\Rightarrow Init ($On(A, Table) \wedge On(B, Table) \wedge On(C, Table)$
 $\wedge Block(A) \wedge Block(B) \wedge Block(C) \wedge clear(A)$
 $\wedge clear(B) \wedge clear(C)$)

\Rightarrow Goal ($On(A, B) \wedge On(B, C)$)

\Rightarrow Action (move(b, x, y))

Then we can formulate the solution as the sequence

[move(B, Table, C), move(A, Table, B)]

The planning of the above is called totally ordered planning. It has only strict linear sequences of actions connecting the start state to the goal state - We can't decompose the problem into sub-problems.

26. Discuss various types of agents in AI. What are problem solving agents? Discuss about the states, percept sequence and working in case of 8 queen problems.

Ans :- Four basic types of agents in order of increasing sophistication:

- Simple reflex agents :-
 - It works by finding a rule whose condition matches the current situation (as defined by the percept) and then doing the action associated with that rule.
 - A set of condition ~ action rules / situation ~ action rules / productions / if~ then rules are defined. e.g if the car – in - front is braking then initiate – braking.
- Model-based reflex agents
 - :- • Model based agents can handle partially observable environments.
 - Its current state is stored inside the agent maintaining some kind of structure which describes the part of the world which can't be seen. This behavior requires information on how the world behaves and works. This additional information completes the “world view” model.
- Goal-based agents:-
 - Knowing about the current state of the environment is not always enough to decide what to do; additionally sort of goal information which describes situations that are desirable is also required. This allows the agent a way to choose among multiple possibilities , selecting the one which reaches a goal state.
 - Current state of the environment is always not enough.
 - The goal is another issue to achieve:-
 - Judgment of rationality / correctness
 - Actions chosen goals, based on:-
 - the current state
 - the current percep
- Utility-based agents:-
 - It is possible to define a measure of how desirable a particular state is. This measure can be obtained through the use of a utility function which maps a state to a measure of the utility of the state. So, utility function maps a state onto a real number, which describes the associated degree of happiness.
 - A complete specification of the utility function allows rational decisions.
- Goals alone are not enough
 - to generate high-quality behavior
 - E.g. meals in the Canteen, good or not ?
- Many action sequences the goals
 - some are better and some worse
 - If goal means success,
 - then utility means the degree of success (how successful it is)

27. Give a comparison between all the uninformed search algorithms and write their advantages and disadvantages. Write the algorithm for solving 8 queen problem using the hill climbing approach.

Ans:-

3.4.7 Comparing uninformed search strategies

Figure 3.21 compares search strategies in terms of the four evaluation criteria set forth in Section 3.3.2. This comparison is for tree-search versions. For graph searches, the main differences are that depth-first search is complete for finite state spaces and that the space and time complexities are bounded by the size of the state space.

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes ^a	Yes ^{a,b}	No	No	Yes ^a	Yes ^{a,d}
Time	$O(b^d)$	$O(b^{1+\lceil C^*/\epsilon \rceil})$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lceil C^*/\epsilon \rceil})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes ^c	Yes	No	No	Yes ^c	Yes ^{c,d}

Figure 3.21 Evaluation of tree-search strategies. b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: ^a complete if b is finite; ^b complete if step costs $\geq \epsilon$ for positive ϵ ; ^c optimal if step costs are all identical; ^d if both directions use breadth-first search.

Algorithm to solve 8 queens problem using hill climbing approach:-

1. Start with a random state(i.e, a random configuration of the board).
2. Scan through all possible neighbours of the current state and jump to the neighbour with the highest objective value, if found any. If there does not exist, a neighbour, with an objective strictly higher than the current state but there exists one with equal then jump to any random neighbour(escaping shoulder and/or local optimum).
3. Repeat step 2, until a state whose objective is strictly higher than all it's neighbour's objectives, is found and then go to step 4.
4. The state thus found after the local search is either the local optimum or the global optimum. There is no way of escaping local optima but adding a random neighbour

or a random restart each time a local optimum is encountered increases the chances of achieving global optimum(the solution to our problem).

5. Output the state and return.

28. What is backtracking and what are its benefits and disadvantages over brute force? Explain CSP with an MapColoring problem example?

Ans:- Depth-first search for CSPs with single-variable assignments is called backtracking search.

Backtracking search is the basic uninformed algorithm for CSPs.

Advantages

There are many advantages of backtracking.

1. It's very intuitive to code. Its almost like a small kid trying to solve the problem.
2. It is very easy to implement and contains less LOC . Most of the back tracking codes are generally few lines of recursive function code.

Disadvantages:

1. More optimal algorithms for the given problem may exist.
 1. Very time inefficient in lot of cases when branching factor is large
 2. Large space complexity because we are using recursion so function information is stored on stack.

Brute force search only takes the *explicit* constraints into account.

Backtracking on the other hand aims to optimize this process.

29. What are Stochastic games? List some Stochastic games. What is Alpha Beta pruning? Prove that alpha-beta pruning takes time $O(2^{m/2})$ with optimal move ordering, where m is the maximum depth of the game tree.

Ans:- In game theory, a stochastic game, introduced by Lloyd Shapley in the early 1950s, is a dynamic game with probabilistic transitions played by one or more players. The game is played in a sequence of stages. At the beginning of each stage the game is in some state. The players select actions and each player receives a payoff that depends on the current state and the chosen actions. The game then moves to a new random state whose distribution depends on the previous state and the actions chosen by the players. The procedure is repeated at the new state and play continues for a finite or infinite number of stages. The total payoff to a player is often taken to be the discounted sum of the stage payoffs or the limit inferior of the averages of the stage payoffs.

A stochastic (or Markov) game includes the following:

- 1) a finite set Q of states (games),
- 2) a set $N = \{1, \dots, n\}$ of agents,
- 3) For each agent i , a finite set A_i of possible actions
- 4) A transition probability function $P : Q \times A_1 \times \dots \times A_n \times Q \rightarrow [0, 1]$
 $P(q, a_1, \dots, a_n, q'') = \text{probability of transitioning to state } q'' \text{ if the action profile } (a_1, \dots, a_n) \text{ is used in state } q$
- 5) For each agent i , a real-valued payoff function $r_i : Q \times A_1 \times \dots \times A_n \rightarrow \mathbb{R}$

Proof :-

Q2

<https://www.studocu.com/en-us/document/university-of-oregon/introduction-to-artificial-intelligence/assignments/solution-2-past-exam-questions-on-computer-information-system/1052571/view>

30. Demonstrate the Wumpus world problem and its solution with the help of logical agents.

Ans :- The Wumpus world is a cave which has 4/4 rooms connected with passageways. So there are a total of 16 rooms which are connected with each other. We have a knowledge-based agent who will go forward in this world. The cave has a room with a beast which is called Wumpus, who eats anyone who enters the room. The Wumpus can be shot by the agent, but the agent has a single arrow. In the Wumpus world, there are some Pits rooms which are bottomless, and if an agent falls in Pits, then he will be stuck there forever. The exciting thing

with this cave is that in one room there is a possibility of finding a heap of gold. So the agent's goal is to find the gold and climb out the cave without falling into Pits or eaten by Wumpus. The agent will get a reward if he comes out with gold, and he will get a penalty if eaten by Wumpus or falls in the pit.

Solution with logical agent :- Exploring the Wumpus World:-

<https://www.javatpoint.com/the-wumpus-world-in-artificial-intelligence>

31. What is first order logic? Give examples of its syntax.

Differentiate between Propositional and First order inference. What is forward chaining?

Ans 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:

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

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

- a. Syntax
- b. Semantics

Eg- Consider the following three sentences:

- "Each animal is an organism"
- "All animals are organisms"
- "If it is an animal then it is an organism"

This can be formalised as: $\forall x(\text{Animal}(x) \rightarrow \text{Organism}(x))$

Observe the colour coding in the natural language sentences: 'each' and 'all' are different ways to say " \forall " and the 'is a' and 'are' in the first two sentences match the " \rightarrow ", and the combination of the " \forall " and " \rightarrow " in this particular construction can be put into natural language as 'if ... then'

Key differences between propositional and first-order logic-

1. Propositional Logic converts a complete sentence into a symbol and makes it logical whereas in First-Order Logic relation of a particular sentence will be made that involves relations, constants, functions, and constants.
2. The limitation of PL is that it does not represent any individual entities whereas FOL can easily represent the individual establishment that means if you are writing a single sentence then it can be easily represented in FOL.

3. PL does not signify or express the generalization, specialization or pattern for example 'QUANTIFIERS' cannot be used in PL but in FOL users can easily use quantifiers as it does express the generalization, specialization, and pattern.

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.

32. (a) If KANSAS + OHIO = OREGON Then find the value of $G + R + O + S + S$ (apply constraint satisfaction rules)
(b) HERE = COMES - SHE, (Assume $S = 8$) Find the value of $R + H + O$ (apply constraint satisfaction rules)

KANSAS
OHIO
OREGON

Clearly
 $K + \text{nothing} \neq K$
 $K + \text{nothing} = 0$
Thus 1 carry from previous slab
 $K + 1 = 0 \text{ (0=5)}$
 $K = 5 - 1 = 4$
 $A + \text{nothing} = R$
Thus 1 carry from previous slab
Generating carry to next slab
 $A = 9$
 $R = A + 1 (\text{carry}) = 0$

4 9 N S 9 S
5 H 1 S

S O E 9 5 N

If we look at 10's digit, $9 + 1 = 5$

- Case 1: $1 = 6$ (no carry from previous slab)
 $9 + 6 = 5$ (1 carry to next slab)
- Case 2: $1 = 5$ (1 carry from prev slab)
 $9 + 5 + 1 = 5$ (1 carry to next slab)

Since $0 = 5$, $1 = 6$

4 9 N S 9 S
5 H 6 S

S O E 9 5 N

$S + 5 = N$
 $S + 5 < 10$
 $S = \{0, 1, 2, 3, 4\}$
S can be 0 or 4 since $R = 0$ $K = 4$
S can't be ~~Case 1~~ $S + 5 = N$

KANSAS
OHIO
OREGON

Clearly

$$K + \text{nothing} \neq K$$

$$K + \text{Wolken} = 0$$

Thus I carry from previous slab

$$x + 1 = 0 \quad (0-5)$$

$$k = 5 - 1 = 4$$

$$A + \text{neutron} = R$$

Thus I carry from previous steps

generating array to unit slots

A - 9

$$R = A + 1(\text{carry}) = 0$$

4 9 N 5 9 S
S H 1 S

S O E 9 5 N

If we look at 10's digit, $9 + 1 = 5$

- Case 1 : 1-6 Cro carry for previous slab

$9 + 6 = 5$ (1 carry to next step)

- * Case 2: $1=5$ (1 cars for prev slots)

$$9 + 5 + 1 = 5 \text{ (1 carry to next slab)}$$

Since $0 \leq 5$, $1 \leq 6$

4 a n s a s

52146 S

50 E 95 N

$$S + S = N$$

$5 + 5 < 10$

$$S = \{0, 1, 2, 3, 4\}$$

S can be 0 or 4 since $R = 0$ $K = 4$

S can't be ~~less than~~ 1 $\Delta + S = N$

If ~~Gross~~ $S = 1$ $N = 6$

$1 = 6$ so S can't be 1

S can't be 4

~~$N = 9$~~ A will be 9, $A = 7$

By hit and trial method

$S = 2$ $N = 7$

4 9 7 2 9 2

5 4 6 5

5 6 5 4 5 7

$2 + 5 = E = 2$ (1 carry to next slot)

Not possible as $S = 2$

there must be 1 carry from previous slot

$7 + 3 + 1 = E \Rightarrow E = 2$

$R = 6$ $S = 2$ $K = 4$ $6 = 5$ $1 = 6$ $N = 7$ $A = 9$

Values possible: $\{1, 3, 8\}$

From hundreds place

$2 + 4 + 1$ (carry) = 9

$H = 8$ $A = 1$

Final Solution

4 9 7 2 9 2

5 8 6 5

5 0 3 1 5 7

GROSS = $1 + 0 + 5 + 2 + 2$
= 10

(b)

HERE
SHE

COMES

Step 1

$C = 1$ (as carry is there)

H
+ 0
—
CO

$H + 0 = H$ is not covered

The answer is ~~1~~ 0 which means carry from previous step

$H + 1 > 10$ results to CO

So $H = 9$ $O = H + 1 = 9 + 1 = 0$ (1 carry to next step)

HERE
SHE

COMES

Step 2

$S = 8$ (given)

$E + E = 8$ $E = 4$

$R + 9 = E$

after substituting E value $R + 9 = 4$

$R = 5$ (1 carry to next step)

$H = 9$ $E = 4$ $R = 5$ $S = 8$ $C = 1$ $O = 0$ $n = 3$

$R + H + O = 5 + 9 + 0 = 14$

33. Consider the following figure

A
B
C

D
E

Start State

A
D
B
C

E

Goal State

Use the heuristic function $h(n) = +1$, if the block is on the correct block/table, -1 , Otherwise Which type of problem do you face using hill climbing algorithms to reach the goal? Use a suitable heuristic function to avoid this problem to reach the goal. Show all the steps.

Ans:-

The handwritten solution illustrates a hill climbing search for a block stacking problem. It begins by defining the Start State and Goal State. The Start State has blocks A, B, and C on one table, and D and E on another. The Goal State has blocks A, D, B, and C on one table, and E on another. A heuristic function $h(n)$ is defined as $+1$ if a block is on the correct support, and -1 otherwise. A search tree is shown with three branches from the Start State. The first branch leads to a state with $h=3-2=1$. The second branch leads to a state with $h=1$. The third branch leads to a state with $h=3-2=1$. The text concludes that a local maxima of $h=1$ is reached instead of the global maxima, identifying this as a plateau problem. It then suggests using a Global Heuristic function to solve this problem, defining it as $+1$ for correct support structures and -1 for wrong ones.

Start State $h=1$

Goal State $h=6$

Here we get heuristic values of all 3 states as 1

And we reached a local maxima of $h=1$ instead of global maxima.

The problem faced here is called plateau. In plateau all neighbours have same value. Hence it is not possible to select the best direction.

To solve this problem we use Global Heuristic function:

i.e.,

- 1) For each block that has the correct support structure: $+1$ to every block in the support structure.
- 2) For each block that has a wrong support structure: -1 to every block in the support structure.

After applying label heuristic

-1	A		
+1	B	-1	D
+0	C	+0	E

Start State $h = -1$

+3	A		
+2	D		
+1	B		
+0	C	+0	E

Goal state

$h = 6$

			-1	A			
H	B		-1	D		+1	B
+0	C	-1	A	+0	E	+0	C

$h = -1$

			-1	A			
+1	B		+0			+0	D
+0	C		D	E	+0		

$h = 0$

			A	-1			
+1	B		D	-1			
+0	C		E	+0			

$h = -1$

-1	D		
-1	A		
+1	B		
+0	C		E

$h = -1$

	E	-1	
	A	-1	
	B	+1	
	C	+0	D

$h = -2$

+1	B		A	-1			
+0	C		D	E	+0		

$h = 0$

+1	B		-1	A	-1		
+0	C		D	E	+0		

$h = -1$

+2	D		
+1	B	+1	A
+0	C	+0	E

$h = 3 - 1 = 2$

+3	A		
+2	D		
+1	B		
+0	C		E

$h = 6$

(Goal state)

Several combinations possible

34. Write the Hill climbing search algorithm. Analyze the performance of the algorithm basing upon appropriate characteristics.

Ans **Algorithm for Simple Hill Climbing:**

Step 1: Evaluate the initial state, if it is goal state then return success and Stop.

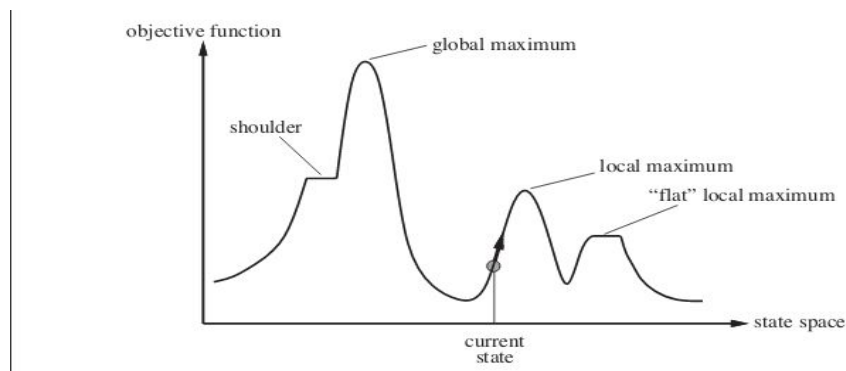
Step 2: Loop Until a solution is found or there is no new operator left to apply.

Step 3: Select and apply an operator to the current state.

Step 4: Check new state:

- a. If it is goal state, then return success and quit.
- b. Else if it is better than the current state then assign new state as a current state.
- c. Else if not better than the current state, then return to step2.

Step 5: Exit.



Different regions in the State Space Diagram

1. **Local maximum:** It is a state which is better than its neighboring state however there exists a state which is better than it(global maximum). This state is better because here the value of the objective function is higher than its neighbors.
2. **Global maximum :** It is the best possible state in the state space diagram. This is because in this state, objective function has highest value.

3. **Plateau/flat local maximum** : It is a flat region of state space where neighboring states have the same value.
4. **Ridge** : It is a region which is higher than its neighbours but itself has a slope. It is a special kind of local maximum.
5. **Current state** : The region of state space diagram where we are currently present during the search.
6. **Shoulder** : It is a plateau that has an uphill edge.

Problems in different regions in Hill climbing

Hill climbing cannot reach the optimal/best state(global maximum) if it enters any of the following regions :

1. **Local maximum** : At a local maximum all neighboring states have a values which is worse than the current state. Since hill-climbing uses a greedy approach, it will not move to the worse state and terminate itself. The process will end even though a better solution may exist.

To overcome local maximum problems : Utilize **backtracking technique**.

Maintain a list of visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path.

2. **Plateau** : On plateau all neighbors have the same value . Hence, it is not possible to select the best direction.

To overcome plateaus : Make a big jump. Randomly select a state far away from the current state. Chances are that we will land at a non-plateau region

3. **Ridge** : Any point on a ridge can look like a peak because movement in all possible directions is downward. Hence the algorithm stops when it reaches this state.

To overcome Ridge : In this kind of obstacle, use two or more rules before testing. It implies moving in several directions at once.

Complexity of Hill Climbing Technique

This technique does not suffer from space related issues, as it looks only at the current state. Previously explored paths are not stored.

For most of the problems in Random-restart Hill Climbing technique, an optimal solution can be achieved in polynomial time. However, for NP-Complete problems, computational time can be exponential based on the number of local maxima.

35. What is the importance of using contours for the A* search method?

Ans A* is complete and optimal on graphs that are locally finite where the heuristics are admissible and monotonic.

A* must be locally finite, because if there exist an infinite amount of nodes where the estimated path cost, $f(n)$, is less than the actual goal path cost then the algorithm could continue to explore these nodes without end, and it will be neither complete nor optimal.

How does monotonicity affect A*'s completeness? Because A* is monotonic, the path cost increases as the node gets further from the root. Contours can be drawn to show areas where the estimated path cost, the $f(n)$, for the nodes inside the areas are lower than or equal to the path cost for the outer bounds of the contours. These contours can be drawn as larger and larger areas that increase outwards as the $f(n)$ for the outer bound of these contours increases. The first solution found is optimal since it is the first band where the $f(n)$ for the contour is equal to the path cost for the goal. All the contours outside of this solution will have a higher f cost.

36. Suppose, there are 3 jugs of capacities 8, 5 and 3 litres respectively. There is no scale on the jugs, so it's only their capacities that are known. Initially the 8 litre jug is full of water, the other two jugs are empty. The water can be poured from one jug to another. The goal is to have exactly 4 litre of water in any of the jugs. The amount of the water in the other two jugs at the end is irrelevant. Formulate this problem as a state space search problem and draw the state space graph of this problem.

Ans Copy from 12(a). It is exactly the same.

37. Explain how to formally define a problem using constraint satisfaction problems? Provide the constraint propagation and backtracking process in relation to map coloring problem.

Ans:- ■ A constraint satisfaction problem (CSP) is defined by a set of variables X_1, X_2, \dots, X_n and a set of constraints C_1, C_2, \dots, C_m .

- Each variable X_i has a nonempty domain D_i of possible values.
- Each constraint C_i involves some subset of the variables and specifies the allowable combinations of values for that subset.
- An assignment of values to some or all of the variables that doesn't violate any constraints is called a consistent or legal assignment.

■ A complete assignment is that in which every variable is mentioned and if it satisfies all the constraints, then it is a solution.

Map Coloring Problem Constraint Propagation & Backtracking Process Example :- PPT

38. Formally define crypt-arithmetic problem , map colouring problem and N queen's problem as constraint satisfaction problem. Solve Map Colouring Problem using constraint Satisfaction problem.

Ans:- Cryptarithmic Problem is a type of constraint satisfaction problem where the game is about digits and its unique replacement either with alphabets or other symbols. In the cryptarithmic problem, the digits (0-9) get substituted by some possible alphabets or symbols. The task in cryptarithmic problem is to substitute each digit with an alphabet to get the result arithmetically correct.

We can perform all the arithmetic operations on a given cryptarithmic problem.

The rules or constraints on a cryptarithmic problem are as follows:

- There should be a unique digit to be replaced with a unique alphabet.
- The result should satisfy the predefined arithmetic rules, i.e., $2+2=4$, nothing else.
- Digits should be from 0-9 only.
- There should be only one carry forward, while performing the addition operation on a problem.
- The problem can be solved from both sides, i.e., left hand side (L.H.S), or right hand side (R.H.S)

Map Coloring problem is an outgrowth of the well-known four-colour map problem, which asks whether the countries on every map can be coloured by using just four colours in such a way that countries sharing an edge have different colours.

The eight queens puzzle is the problem of placing n chess queens on an $n \times n$ chessboard so that no two queens threaten each other; thus, a solution requires that no two queens share the same row, column, or diagonal.

Solution of Map Coloring Problem using CSP:- PPT

39. Formally define crypt-arithmetic problem , map colouring problem and N queen's problem as constraint satisfaction problem.

Ans:- Same question as above. (38)

40. What is the problem with informed search algorithms? Why informed search techniques in some condition are better than uninformed search techniques.

Ans:- Informed Search Algorithms with their disadvantages :-

1.) Best-first Search Algorithm (Greedy Search) Disadvantages:-

- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS.
- This algorithm is not optimal.

2.) A* Search Algorithm Disadvantages:-

- It does not always produce the shortest path as it is mostly based on heuristics and approximation.
- A* The search algorithm has some complexity issues.
- The main drawback of A* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

3.) Hill Climbing Search Disadvantages:-

- Problem: depending on initial state, can get stuck in local maxima.

In terms of efficiency informed search is better than the uninformed search. Uninformed search consumes more time and cost as it has no clue about the solution as compared to an informed search.

41. What do you understand by informed search techniques? Given A* Algorithm , explain how can you modify the A* algorithm to behave as Greedy Best First algorithm.

Ans:- Informed search algorithm contains an array of knowledge such as how far we are from the goal, path cost, how to reach the goal node, etc. This knowledge help agents to explore less to the search space and find more efficiently the goal node.

The informed search algorithm is more useful for large search spaces. Informed search algorithms use the idea of heuristic, so it is also called Heuristic search.

Heuristic is a function which is used in Informed Search, and it finds the most promising path. It takes the current state of the agent as its input and produces the estimation of how close the agent is from the goal. The heuristic method, however, might not always give the best solution, but it is guaranteed to find a good solution in reasonable time. Heuristic function estimates how close a state is to the goal. It is represented by $h(n)$, and it calculates the cost of an optimal path between the pair of states. The value of the heuristic function is always positive.

The only difference between Greedy BFS and A* BFS is in the evaluation function. For Greedy BFS the evaluation function is $f(n) = h(n)$ while for A* The evaluation function is $f(n) = g(n) + h(n)$. Essentially, since A* is more optimal of the two approaches as it also takes into consideration the total distance travelled so far i.e. $g(n)$.

42. What is the problem with informed search algorithms? Derive the time and space complexity of Iterative Depth First Search Algorithm?

Ans:- a. they don't use any problem-specific information to guide them.
b. Solutions take an infinite time to compute.

Disadvantages of Best First Search-

- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS.
- This algorithm is not optimal

Disadvantages of A* Search-

- It does not always produce the shortest path as it mostly based on heuristics and approximation.
- A* the search algorithm has some complexity issues.
- The main drawback of A* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

Time Complexity: If you can access each node in $O(1)$ time, then with branching factor of b and max depth of m , the total number of nodes in this tree would be worst case $= 1 + b + b^2 + \dots + b^{m-1}$. Using the formula for summing a geometric sequence (or even solving it ourselves) tells that this sums to $= (b^m - 1)/(b - 1)$, resulting in total time to visit each node proportional to b^m . Hence the complexity $= O(b^m)$.

On the other hand, if instead of using the branching factor and max depth you have the number of nodes n , then you can directly say that the complexity will be proportional to n or equal to $O(n)$.

Space Complexity: The length of longest path $= m$. For each node, you have to store its siblings so that when you have visited all the children, and you come back to a parent node, you can know which sibling to explore next. For m nodes down the path, you will have to store b nodes extra for each of the m nodes. That's how you get an $O(bm)$ space complexity.

43. What do you understand by soundness and completeness in inference mechanism? Provide the architecture of a knowledge based agent for partially observable environment.

Ans:- We should only be able to prove things that are true, and if they are true, we should be able to prove them. These two properties are called **soundness** and **completeness**.

A proof system is **sound** if everything that is provable is in fact true. In other words, if $\varphi_1, \dots, \varphi_n \vdash \psi$ then $\varphi_1, \dots, \varphi_n \models \psi$.

A proof system is **complete** if everything that is true has a proof. In other words, if $\varphi_1, \dots, \varphi_n \models \psi$ then $\varphi_1, \dots, \varphi_n \vdash \psi$.

Sketch of proof of soundness

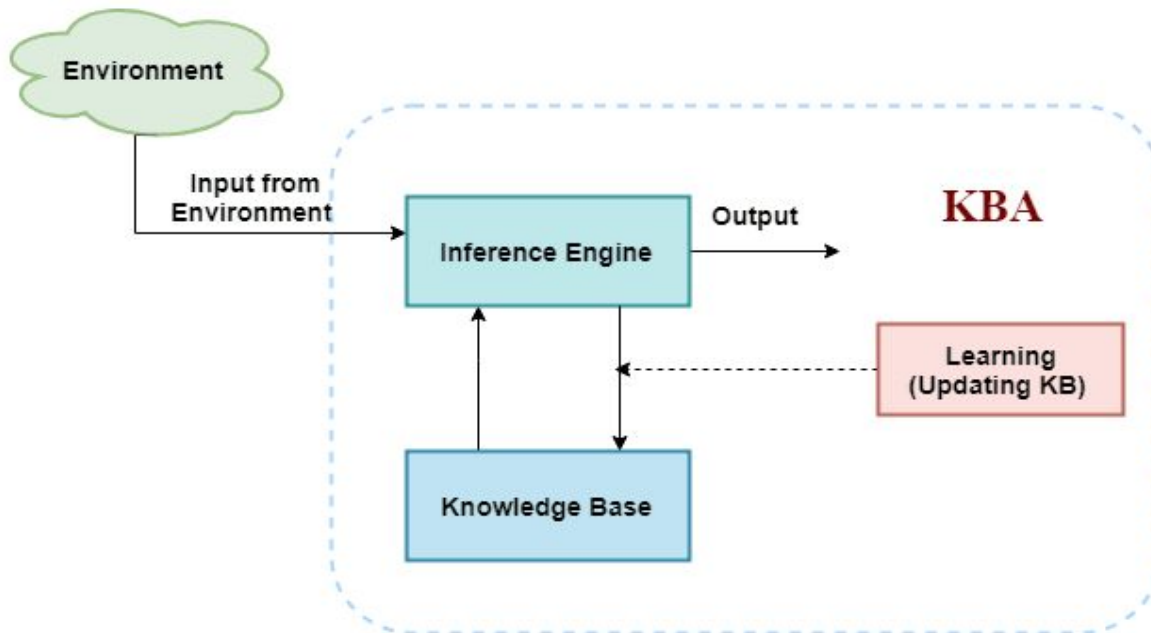
To prove that the set of natural deduction rules introduced in the previous lecture is sound with respect to the truth-table semantics given two lectures ago, we can use induction on the structure of proof trees.

Let X be the set of well-formed proofs. Then X is an inductively defined set; the set of rules of the proof system are the rules for constructing new elements of X from old.

Let $P(x)$ be the statement ``if x is a valid proof tree ending with $\varphi_1, \dots, \varphi_n \vdash \psi$ then $\varphi_1, \dots, \varphi_n \models \psi$ ". We can prove $\forall x \in X, P(x)$ by structural induction; we simply have to consider each inference rule; for the rules with subgoals above the line we can inductively assume entailment.

Sketch of proof of completeness

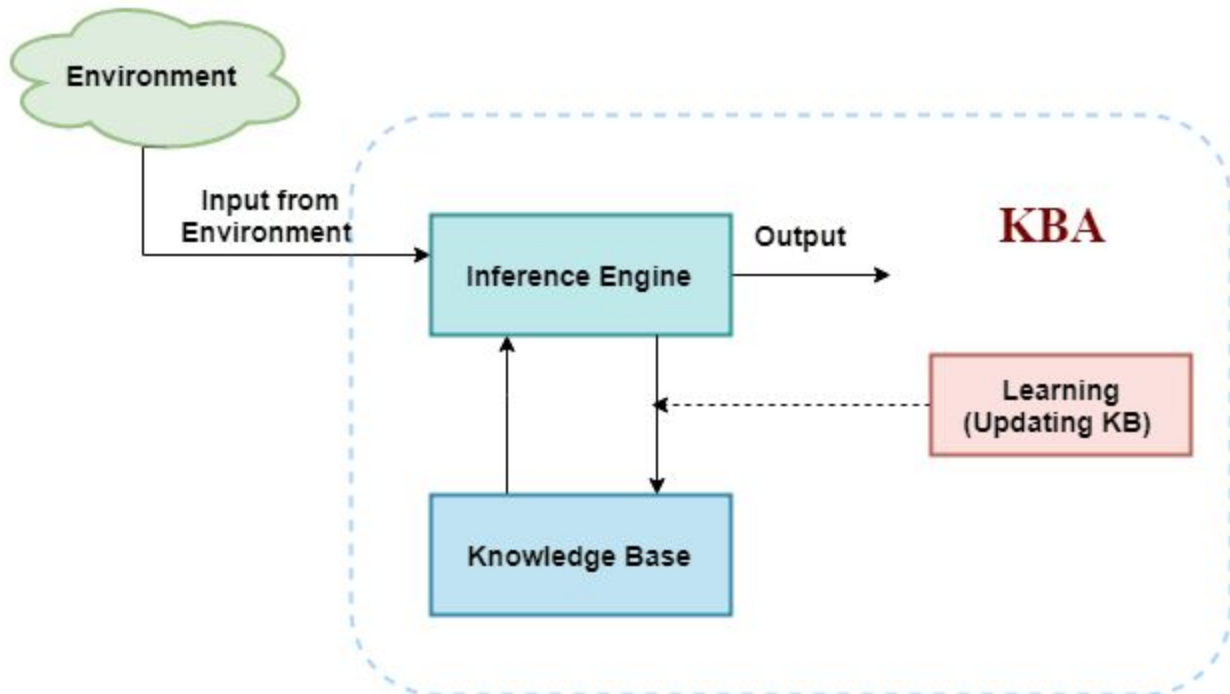
The idea behind proving completeness is that we can use the law of excluded middle and \vee introduction (as in the example proof from the previous lecture) to separate all of the rows of the truth table into separate subproofs; for the interpretations (rows) that satisfy the assumptions (and thus the conclusion) we can do a direct proof; for those that do not we can do a proof using reductio ad absurdum.



44. Provide the architecture of a knowledge based agent for partially observable environment. Elaborate how propositional logic can be used in designing knowledge based agent. You may take the example of “The Wumpus World”.

Ans:- A knowledge based agent is composed of a knowledge base and an inference mechanism to infer new sentences and use these sentences to decide what actions to take. • Example:- if it is raining then take an umbrella weather(raining) take(umbrella) premise/antecedent conclusion/consequent.

Architecture of Knowledge based agent for partially observable environment :-



Wumpus World Example:- PPT(slide 301)

45. How Resolution Algorithm is used for inference mechanism , provide the algorithm and elaborate using an example of your choice.

Ans:-

```

function PL-RESOLUTION( $KB, \alpha$ ) returns true or false
    clauses  $\leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg \alpha$ 
    new  $\leftarrow \{ \}$ 
    loop do
        for each  $C_i, C_j$  in clauses do
            resolvents  $\leftarrow$  PL-RESOLVE( $C_i, C_j$ )
            if resolvents contains the empty clause then return true
            new  $\leftarrow$  new  $\cup$  resolvents
        if new  $\subseteq$  clauses then return false
    clauses  $\leftarrow$  clauses  $\cup$  new

```

Resolution Algorithm Example Used for Inference Mechanism :- Resolution Wumpus Example - PPT slide(343-344)

46. Contrast Multiagent Planning with conditional and continuous planning. For a partially observable environment what type of planning do you suggest .

Ans:-

Multi-agent planning

So far we only discussed single-agent environments.

Other agents can simply be added to the model of the world:

- Poor performance since agents are not indifferent to other agents' intentions

In general two types of multi-agent environments:

- Cooperative
- Competitive

Continuous planning.

Agent persists indefinitely in an environment

- Phases of goal formulation, planning and acting

Execution monitoring + planner as one continuous process

Example: Blocks world

- Assume a fully observable environment
- Assume partially ordered plan



Conditional planning

Deal with uncertainty by checking the environment to see what is really happening.

Used in fully observable and nondeterministic environments:

- The outcome of an action is unknown.
- Conditional steps will check the state of the environment.
- How to construct a conditional plan?

We will choose conditional planning for a partially observable environment where we cannot track every state..

47. Given a partially observable environment what type of planning do you suggest . Justify by giving proper technical explanation.

Ans:- Conditional Planning can also take place in the Partially Observable Environments where we cannot keep a track on every state.

In vacuum cleaners e.g. if the dirt is at Right and the agent knows about Right, but not about Left. Then, in such cases Dirt might be left behind when the agent leaves a clean square. Initial state is also called a state set or a belief state.

Sensors play an important role in Conditional planning for partially observable environments. Automatic sensing can be useful; with automatic sensing an agent gets all the available precepts at every step.

48. Elaborate and explain using proper example the difference between inference using forward and backward chaining.

Ans:- • FC is data-driven, automatic, unconscious processing, – e.g., object recognition, routine decisions • May do lots of work that is irrelevant to the goal

- BC is goal-driven, appropriate for problem-solving, – e.g., Where are my keys? How do I get into a PhD program?
- Complexity of BC can be much less than linear in size of KB

Example:- Inference Based Example:-

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$

$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$

$W_{1,1} \vee W_{1,2} \vee \dots \vee W_{4,4}$

$\neg W_{1,1} \vee \neg W_{1,2} \vee \neg W_{1,3} \vee \neg W_{1,4}$

...

⇒ 64 distinct proposition symbols, 155 sentences

SECTION A -Short Answers and MCQ

1. What is Artificial Intelligence?

Ans: AI is the field of science and engineering concerned with the conceptual understanding of what is commonly called intelligent behavior and with the creation of artifacts that exhibits such behavior. Artificial intelligence is intelligence demonstrated by machines, unlike the natural intelligence displayed by humans and animals.

AI has the following features:

- i) Thinking Humanly
- ii) Thinking Rationally
- iii) Acting Humanly
- iv) Acting Rationally

2. What are the parameters on which the performance of an intelligent agent depend on ?

Ans Rationality of an agent depends on the following –

- The performance measures, which determine the degree of success.
- Agent's Percept Sequence till now.
- The agent's prior knowledge about the environment.
- The actions that the agent can carry out.

3. What is a heuristic function and what do you mean by an admissible heuristic function? Show the proof.

Ans A heuristic function $h(n)$ takes a node n and returns a non-negative real number that is an estimate of the cost of the least-cost path from node n to a goal node. The function $h(n)$ is an admissible heuristic if $h(n)$ is always less than or equal to the actual cost of a lowest-cost path from node n to a goal.

4. How can you increase the effectiveness of the alpha-beta pruning ?

Ans Ideal Ordering: In some cases of alpha beta pruning a lot of the nodes pruned by the algorithm. This is called Ideal ordering in pruning. In this case, the best move occurs on the left side of the tree. We apply DFS hence it first searches left of the tree and goes deep twice as a minimax algorithm in the same amount of time.

If mcq, ans depends on the nodes.

5. Give some examples of Constraint Satisfaction Problems "CSP"

Ans sudoku, cryptarithmic, crosswords, n-queen

6. What is a learning agent? Explain its Components.

Ans A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities. It starts with basic knowledge and then is able to act and adapt automatically through learning.

A learning agent has mainly four conceptual components, which are:

- a. Learning element: It is responsible for making improvements by learning from environment
- b. Critic: Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
- c. Performance element: It is responsible for selecting external action
- d. Problem generator: This component is responsible for suggesting actions that will lead to new and informative experiences.

7. Explain autonomous agent and omniscient agent with example

Ans Autonomous agents are software programs which respond to states and events in their environment independent from direct instruction by the user or owner of the agent, but acting on behalf and in the interest of the owner. examples include intelligent agents, **autonomous robots**, and various **software agents**, including **artificial life** agents, and many **computer viruses**.

An omniscient agent is an agent which knows the actual outcome of its action in advance. However, such agents are impossible in the real world.

8. What are the factors that a rational agent should depend on at any given time?

Ans What is rational at any given time depends on four things:

- - The performance measure that defines the degree of success.
- - Everything that the **agent** has perceived so far (the percept sequence)
- - What the **agent** knows about the environment.
- - The actions the **agent** can perform.

9. What are the main factors considered in designing the intelligent systems?

Ans In designing intelligent systems there are four main factors to consider

P Percepts – the inputs to our system

A Actions – the outputs of our system

G Goals – what the agent is expected to achieve

E Environment – what the agent is interacting with

10. Differentiate between a node and a state. Explain the components of a node.

Ans An AI problem, when formally defined, models the state of the world by using some variables, numbers, and symbols.

- 331 : Missionaries and Cannibals • [[_, _, X], [_, O, _], [_, _, _]] : Tic Tac Toe – A “State” is one possible way to set the variables.

A Node is a data structure that is part of a search tree. – You build nodes while you perform your search, and you can associate them with particular states of your AI problem that each is representing. – If you get to the same state by way of two different paths in a search tree, then you might have 2 nodes that represent the same state (i.e. they have the same setting of the variables).

11. Define the different ways we can evaluate an algorithm’s performance?

Ans a. Completeness

b. Optimality

c. Time and space complexity

12. What is bi-directional search? What are its pros and cons.

Ans Bidirectional search is a graph search algorithm which find smallest path form source to goal vertex. It runs two simultaneous search –

1. Forward search form source/initial vertex toward goal vertex
2. Backward search form goal/target vertex toward source vertex

Bidirectional search replaces a single search graph(which is likely to grow exponentially) with two smaller sub graphs – one starting from initial vertex and other starting from goal vertex. The search terminates when two graphs intersect.

Advantages- 1 The merit of bi-directional search is it's speed . Sum of the time taken by two searches(forward and backward) is much less than the $O(b^d)$ complexity 2. I requires less memory

Disadvantages- 1. Implementation of bidirectional search algorithm is difficult because additional logic must be included to decide which search tree to extend in each step. 2. One should know the goal state in advance 3. The algorithm must be too efficient to find the intersection of the two search trees 4. It is not always possible to search backward through the possible states.

13. Define consistent assignment and partial assignment with example.

Ans An assignment of values to some or all of the variables that doesn't violet any constraints is called a consistent or legal assignment.

A complete assignment is that in which every variable is mentioned and if it satisfies all the constraints, then it is a solution.

An assignment of a subset of the variables is called a partial assignment. Constraints only restrict the partial assignments over its scope

14. List out the varieties of constraints with Example

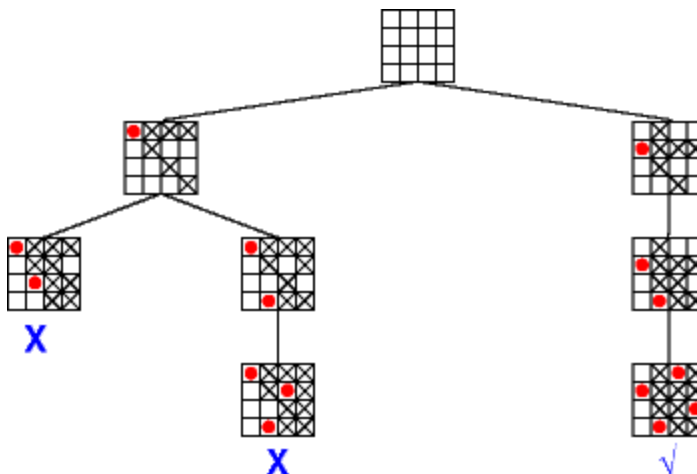
Ans **State constraints:** These constraints include physical constraints on the given state or on the states and make sure to forbid the states that are against maintenance goals.

- **Effect constraints:** These constraints are among the state variables and action variables at given time t , state variables with time $t + 1$ along with previous state.
- **Precondition constraints:** These constraints are between the state variable of given time t and specify the actions from a state.
- **Actions constraints:** They state which actions cannot co-occur.

Initial state constraints and goal constraints.

15. What is the use of forward checking? Give one example.

Ans Forward checking detects the inconsistency earlier than simple backtracking and thus it allows branches of the search tree that will lead to failure to be pruned earlier than with simple backtracking. This reduces the search tree and (hopefully) the overall amount of work done. But it should be noted that forward checking does more work when each assignment is added to the current partial solution. 4-queens problem and FC



16. Explain Conflict directed Backjumping and its advantages.

Ans:- A still more refined backjumping algorithm, sometimes able to achieve larger backjumps, is based on checking not only the common presence of two variables in the same constraint but also on whether the constraint actually caused inconsistency. In particular, this algorithm collects one of the violated constraints in every leaf. At every node, the highest index of a variable that is in one of the constraints collected at the leaves is a safe jump.

While the violated constraint chosen in each leaf does not affect the safeness of the resulting jump, choosing constraints of highest possible indices increases the highness of the jump. For this reason, conflict-based backjumping orders constraints in such a way constraints over lower indices variables are preferred over constraints on higher index variables.

17. Why is First order logic preferred over propositional logic ? Explain briefly.

Ans

1. Propositional Logic converts a complete sentence into a symbol and makes it logical whereas in First-Order Logic relation of a particular sentence will be made that involves relations, constants, functions, and constants.
2. The limitation of PL is that it does not represent any individual entities whereas FOL can easily represent the individual establishment that means if you are writing a single sentence then it can be easily represented in FOL.
3. PL does not signify or express the generalization, specialization or pattern for example 'QUANTIFIERS' cannot be used in PL but in FOL users can easily use quantifiers as it does express the generalization, specialization, and pattern.

18. Define Knowledge representation. What are the desired properties of a knowledge representation language?

Ans 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.

1. **Representational Accuracy:**

The KR system should have the ability to represent all kinds of required knowledge.

2. **Inferential Adequacy:**

The KR system should have the ability to manipulate the representational structures to produce new knowledge corresponding to existing structure.

3. **Inferential Efficiency:**

The ability to direct the inferential knowledge mechanism into the most productive directions by storing appropriate guides.

4. **Acquisitional efficiency-** The ability to acquire new knowledge easily using automatic methods.

19. What is inference? What are the properties of a good inference algorithm?

Ans:- In artificial intelligence, 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.**

20. Define object, predicate, relations and quantifiers with respect to the FOL.

Ans **objects** are the domain of discourse or the universe.

Relations: It can be unary relation such as: red, round, is adjacent, **or n-ary relation such as:** the sister of, brother of, has color, comes between.

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

A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimens in the universe of discourse.

21. What is Move-ordering and how it affects alpha-beta pruning?

Ans The effectiveness of alpha – beta pruning is based on the order in which node is examined. Move ordering plays an important role in alpha beta pruning.

There are two types of move ordering in Alpha beta pruning:

Worst Ordering: In some cases of alpha beta pruning none of the node pruned by the algorithm and works like standard minimax algorithms. This consumes a lot of time as because of alpha and beta factors and also does not give any effective results. This is called Worst ordering in pruning. In this case, the best move occurs on the right side of the tree.

Ideal Ordering: In some cases of alpha beta pruning a lot of the nodes pruned by the algorithm. This is called Ideal ordering in pruning. In this case, the best move occurs on the left side of the tree. We apply DFS hence it first searches left of the tree and goes deep twice as a minimax algorithm in the same amount of time.

22. What are Killer moves and what is Killer move heuristic?

Ans

Killer Heuristic, a dynamic, path-dependent move ordering technique. It considers moves that caused a **beta-cutoff** in a **sibling node** as **killer moves** and orders them high on the **list**. When a **node fails high**, a **quiet move** that causes a cutoff is stored in a table indexed by **ply**, typically containing two or three moves per ply. The replacement scheme ought to ensure that all the available slots contain different moves. These moves are called killer moves.

23. What is the Horizon effect in Real time games? How can they be handled?

Ans The Horizon Effect is caused by the **depth** limitation of the **search algorithm**, and became manifest when some negative event is inevitable but postponeable. Because only a partial game tree has been analyzed, it will appear to the system that the event can be avoided when in fact this is not the case.

Beside obligatory **quiescence search**, **extensions**, specially **check extensions** are designed to reduce horizon effects.

24. What is quiescence search? What is its use?

Ans Most chess programs, at the end of the main search perform a more limited **quiescence** search, containing fewer moves. The purpose of this search is to only **evaluate** "quiet" **positions**, or positions where there are no winning **tactical moves** to be made. This search is needed to avoid the **horizon effect**. Simply stopping your search when you reach the desired **depth** and then evaluate, is very dangerous.

25. How does Monte-Carlo simulation be helpful in Stochastic games?

Ans Monte Carlo simulations are used to model the probability of different outcomes in a process that cannot easily be predicted due to the intervention of **random variables**. It is a technique used to understand the impact of risk and uncertainty in prediction and forecasting models.

A Monte Carlo simulation can be used to tackle a range of problems in virtually every field such as finance, engineering, supply chain, and science. It is also referred to as a multiple probability simulation.

26. What are Stochastic games? Give one example. What are Chance nodes?

Ans A stochastic game is a collection of normal-form games that the agents play repeatedly. The particular game played at any time depends probabilistically on the previous game played and the actions of the agents in that game.

Eg The Big Match

A chance node, represented by a circle, shows the probabilities of certain results.

27. Explain Meta-reasoning. Where is it Used?

Ans Philosophers and cognitive scientists of many persuasions have long wondered what is unique to human intelligence. Although a number of ideas have been proposed, a common differentiator appears to be a pervasive capacity for thinking about ourselves in terms of who we are, how others see us, and in terms of where we have been and where we want to go. As humans, we continually think about ourselves and our strengths and weaknesses in order to manage both the private and public worlds within which we exist. But the Artificial Intelligence community has not only wondered about these phenomena; it has attempted to implement actual machines that mimic, simulate, and perhaps even replicate this same type of reasoning called metareasoning. It is used to train an AI using metacognition of which meta-reasoning is an integral part of.

28. What is Planning and what is a Solution to planning? List some classical planning environments.

Ans Planning is the task of finding a procedural course of action for a declaratively described system to reach its goals while optimizing overall performance measures. STRIPS (STanford Research Institute Problem Solver) a restrictive way to express states, actions and goals, but leads to more efficiency,
Classical Planning environment- Chess

29. Consider a uniform search tree T with branching factor b , depth of the solution d , and depth of the tree m . Let B and D be breadth-first and depth-first search implementations respectively that can be applied on T . What is the difference in the number of nodes expanded by B and D in the worst-case scenario when the goal state is at depth d ? Assume that the root node is at depth 0 .

Ans $B = b^d$ $D = b^m$ $B - D = b^d - b^m$

30. Describe a state space in which iterative deepening search performs much worse than depth first search.

Ans If your state space is like a linked list (i.e. only 1 child per node) and the goal state is the leaf, you will end up with exactly the situation you described.

With DFS, you will proceed along each child until you reach the leaf. If there are n nodes, the running time is $O(n)$.

With IDS, in the first iteration you will only visit the child of the root. In the second iteration, you will visit the root's child and its own child (depth = 2, visited 2 nodes). In the third iteration, you go to a depth of 3, visiting 3 nodes. Hence the total number of visits is $1 + 2 + \dots + n = O(n^2)$.

31. Briefly define and describe the significance of an admissible heuristic

Ans a **heuristic function** is said to be **admissible** if it never overestimates the cost of reaching the goal, i.e. the cost it estimates to reach the goal is not higher than the lowest possible cost from the current point in the path.

The function $h(n)$ is an admissible heuristic if $h(n)$ is always less than or equal to the actual cost of a lowest-cost path from node n to a goal

32. Prove or give a counterexample: Uniform Cost Search is a special case of A* search

Ans A* with $h(n) = 0$ (thus $f(n) = g(n)$) is uniform-cost; Contrariwise, A* with $g(n) = 0$ (thus $f(n) = h(n)$) is greedy

33. Prove that if a heuristic is consistent, it must be admissible. Construct an admissible heuristic that is not consistent

Ans let $k(n)$ be the cost of the cheapest path from n to the goal node. We will prove by induction on the number of steps to the goal (along the best path) that $h(n) \leq k(n)$.

Base Case: If there are 0 steps from n to the goal then n is a goal node, and $h(n) = 0 \leq k(n)$.

Inductive Step: Suppose that the best path from n to the goal has i steps, and call the next node along that path n' (let a be the action leading from n to n'). The best path from n' to the goal must have $i - 1$ steps, so by the induction hypothesis $h(n') \leq k(n')$. Using consistency we find that $h(n) \leq c(n, a, n') + h(n') \leq c(n, a, n') + k(n') = k(n)$.

constructing an admissible but inconsistent heuristic. We saw one in class. Here's another. Consider a search problem in which the states are nodes along a path $P = n_0, n_1 \dots n_m$ where n_0 is the start state, n_m is the goal state and for all i there is an action

from n_i to n_{i+1} of cost 1. The cheapest cost from any node n_i to the goal is then $k(n_i) = m - i$. Then define a heuristic function $h(n) = m - 2 \times \lceil i/2 \rceil$. Clearly this is admissible since for all states n_i , $h(n_i) \leq k(n_i)$. It is not consistent

34. Which are the basic requirements that an AI program should fulfill?

Ans The minimum requirement for AI is that an algorithm make decisions based on data, irrespective of the quality of the decisions.

35. State the fundamental goal of KR.

Ans Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which is 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 diagnosing 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.

36. List the 4 properties any KR system should possess

Ans

1. Representational Accuracy: KR system should have the ability to represent all kind of required knowledge.

2. Inferential Adequacy: KR system should have ability to manipulate the representational structures to produce new knowledge corresponding to existing structure.

3. Inferential Efficiency: The ability to direct the inferential knowledge mechanism into the most productive directions by storing appropriate guides.

4. Acquisitional efficiency- The ability to acquire new knowledge easily using automatic methods.

37. Give an example of relational knowledge.

Ans simple relational knowledge representation.

Player	Weight	Age
--------	--------	-----

Player1	65	23
Player2	58	18
Player3	75	24

38. Write the clausal form for the given English query “Does Jay breathe”?

Ans $P = \text{Jay breathes}$

Does Jay breathe? = Jay breathes AND Jay does not breathe

$P \wedge (\neg P)$

39. What is a heuristic function in problem solving searching algorithms?

Ans A Heuristic (or a heuristic function) takes a look at search algorithms. At each branching step, it evaluates the available information and makes a decision on which branch to follow. It does so by ranking alternatives. The Heuristic is any device that is often effective but will not guarantee work in every case.

40. What do you understand by state space representation?

Ans a **state-space representation** is a mathematical model of a physical system as a set of input, output and state variables related by first-order **differential equations** or **difference equations**. State variables are variables whose values evolve over time in a way that depends on the values they have at any given time and on the externally imposed values of input variables. Output variables' values depend on the values of the state variables.

41. Provide two uninformed search algorithms for problem solving?

Ans Uniform cost search, Bidirectional Search

42. Provide two informed search algorithms for problem solving?

Ans Best First Search, A* Search

43. What is an agent? Provide two applications of agents.

Ans An agent is anything that can be viewed as : perceiving its environment through **sensors** and acting upon that environment through **actuators**.

Vacuum cleaner robot, self-driving cars

44. What are utility based agents?

Ans The agents which are developed having their end uses as building blocks are called utility based agents. When there are multiple possible alternatives, then to decide which one is best, utility-based agents are used. They choose actions based on a **preference (utility)** for each state.

45. Enlist the four basic components of agent architecture?

Ans Sensors, Actuators, Agent Program, Environment

46. Why are Agents the “nuts and bolt” of A.I ?

Ans Artificial Intelligence system is often defined because of the study of the rational agent and its environment. The agents sense the environment through sensors and act on their environment through actuators.

An AI agent can have mental properties like knowledge, belief, intention, etc. A rational agent might be anything which makes decisions, as an individual, firm, machine, or software.

It acts with the simplest outcome after considering past and current percepts(agent's perceptual inputs at a given instance).

Multiple Choice

1. In which of the following situations might a blind search be acceptable?

c) small search space

2.What are the main goals of AI?

C. Both A and B

3.An Artificial Neural Network Is based on?

c)Cognitive Artificial Intelligence approach

4.Which of the following is not a type of agent in artificial intelligence?

D. target based

5.How many types are available in an uninformed search method?

c) 5

6.A search algorithm takes _____ as an input and returns _____ as an output.

b) Problem, solution

7.A* algorithm is based on _____

c) Best-First-Search

8.uniform-cost search expands the node n with the _____

a) Lowest path cost

9.What among the following Constitutes To The Incremental Formulation Of CSP?

d) All of the mentioned

10. Consider a problem of preparing a schedule for a class of student. What type of problem is this?

c) CSP

11. Adversarial search problems uses _____

a) Competitive Environment

12. What is called a Transposition table?

2. Hash table of previously seen positions

13. Translate the following sentence into FOL "Forever a, If a is a philosopher, then a is a scholar"

A. For all a, philosopher(a) scholar(a)

14. What is the condition of literals in variables?

b) Universally quantified

15. First Order Logic is also known as _____

d) All of the mentioned

16. Planning in partial order plan.

A. Relationships between the actions of the behavior are set prior to the actions

17. The initial state and the legal moves for each side define the _____ for

b. Game Tree

18. General algorithm applied on the game tree for making the decision of win/loss is _____.

D. MIN/MAX Algorithms

19. Which search is equal to minimax search but eliminates the branches that can't influence the final decision?

c. Alpha-beta pruning

20. Which of the following is identical to the closed list in Graph search?

c. Transposition table

21. Which function is used to calculate the feasibility of a whole game tree?

a. Evaluation function

22. What is called a transposition table?

b. Hash table of previously seen positions

23. Uncertainty arises in the Wumpus world because the agent's sensors give only

c) Partial & local Information

23. Any agent with no percepts can never fully satisfy its goals.

False

24 An environment that is fully observable for one agent can be partially observable for another agent. State T/F with justification.

True

25. If the environment does not change when the agent is deliberating on what action to do next, the environment is called

A. Static environment

26. The agent which internalises the the present state of the environment based on the previous state, previous action, current percepts and the model of the world is called a

B. Model Based reflex agent

27. The agent which maximises the expected performance given the percept sequence to date and prior knowledge is called a

C. Rational agent

28. The agent has very capable sensors which can sense the entire geography of the surroundings.

A. Fully observable

29 The agent's sucking action cleans the floor with a probability which is less than one.

B. Stochastic

30. The cooking agent should cook the food well in time, the food should neither be burned nor half-cooked, the taste should be good, the ingredients should be in the

A. Performance

31. utensils, LPG stove, refrigerator and other kitchenware.

A. Environment

32. The state of the food being cooked (particularly its temperature and water content) on the stove can change while the cooking agent is deliberating on what to do next.

A. Dynamic

33. Knowledge and reasoning also play a crucial role in dealing with _____ Environment.

b) Partially Observable

34. A) Knowledge base (KB) is consists of set of statements. B) Inference is deriving a new sentence from the KB. Choose the correct option.

a) A is true, B is true

35 Wumpus World is a classic problem, best example of _____

c) Reasoning with Knowledge

36 Inference algorithm is complete only if _____

d) It can derive any sentence that is an entailed version & It is truth preserving

37. Which search is equal to minimax search but eliminates the branches that can't influence the final decision?

c) Alpha-beta pruning

38. To which depth does the alpha-beta pruning can be applied?

d) Any depth

39. Which search is similar to minimax search?

b) Depth-first search

40. Which value is assigned to alpha and beta in the alpha-beta pruning?

d) Both Alpha = max & Beta = min

41. What is the expansion of PEAS in a task environment?

c) Performance, Environment, Actuators, Sensors

42. What kind of observing environments are present in artificial intelligence?

d) Both Partial & Fully

43. What kind of environment is a crossword puzzle?

a) Static

44. Where does the performance measure is included?

b) Task environment

45. Which algorithm will work backward from the goal to solve a problem?

b) Backward chaining

46. What will the backward chaining algorithm return?

c) Substitutes matching the query

47. Which problem can frequently occur in backward chaining algorithms?

d) Both Repeated states & Incompleteness

48. What is used in backward chaining algorithms?

c) Composition of substitution

49. Which closely resembles propositional definite clauses?

d) First-order definite clauses

50. Which condition is used to cease the growth of forward chaining?

c) No further inference

51. Which will be the instance of the class datalog knowledge bases?

b) No function symbols

52. Which will solve the conjuncts of the rule so that the total cost is minimized?

b) Conjunct ordering

53. Which is an example of global constraint ?

b. Alldiff

54. Minimax algorithm uses the property of

a. Depth First Search

55. In alpha-beta pruning, alpha is

d. maximum value found so far

56. Many game playing programs use _____ for opening and ending part of games.

c. table look-up