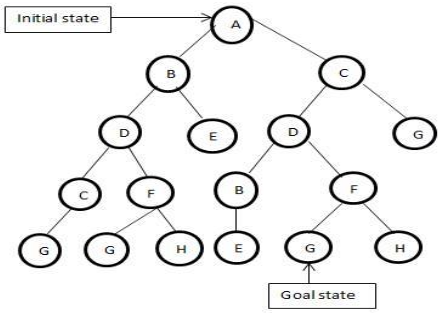
Table of Contents

Assignment: 01 – 10 (Questions)

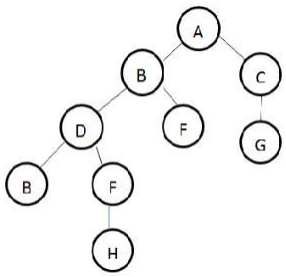
Assignment 1 Questions:

1. Write the algorithm for Basic Search.
2. Write the algorithm for Breadth-First Search.
3. Write the algorithm for Depth-First Search.
4. Write the algorithm for Graph Search.
5. Write the algorithm for Depth Limited Search.
6. Write the differences between Uniform-Cost Search and Greedy Search algorithm.
7. Find out the open and close list for the diagram below using Breadth-first and Depth-first Search algorithm.



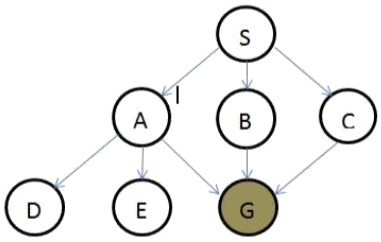
Assignment 2 questions

1. "Can machines think? "_ Explain your reasons both for agreement and disagreement.
2. Can machine be comparable to human beings on behaving terms?
3. What your opinion on the statement –"Machine will have minds."
4. Is it possible for machine to be intelligent?
5. Prove with an example- Search tree is exponentially larger than the graph.
6. Find out the open and close list for the following diagram Using BFS and DFS.



7. Consider the following graph and its table for Greedy, A*, Uniform Cost Search.

N	g(n)	h(n)	f(n)	A*(n)
S	0	8	8	9
A	1	8	9	9
B	5	4	9	4
C	8	3	11	5
D	4	∞	∞	∞
E	8	∞	∞	∞
G	9	0	9	0



Which is the optimal search algorithm for it?

8. Write down the algorithm for BFS.
9. What is framing?
10. Differentiate between implicit and explicit state space.
11. Differentiate between abductive and deductive reasoning.

Assignment 3 Questions:

1. What is Admissible Heuristic function?-Explain briefly.
2. What is Evaluation Function?-Explain briefly.
3. Describe the two heuristic function related to 8 puzzle problem.
4. Write down the Completeness, Time Complexity, Space Complexity, Optimality of the following:

    BFS

    DFS

    DFID

    Bidirectional Search

Assignment 4 Questions:

1. Write down the MIN-MAX Algorithm.
2. Write down the Alpha-Beta Pruning Algorithm.
3. Write down the Alpha-Beta Pruning Algorithm with figure for MAX and MIN state

Assignment 5 Questions:

1. Translate the following propositional logics.
    a. It rains in July.
    b. The book is not costly.
    c. If it rains today and one does not carry umbrella, then he will be drenched.
    d. If triangle is equilateral then it is isosceles.
    e. ABC is an equilateral triangle.
2. If P is true and Q is true then write down the truth table for the following:
    a. $P \rightarrow Q$
    b. $(\neg P \vee Q) \rightarrow Q$
    c. $(\neg P \vee Q) \rightarrow P$
    d. $(P \vee \neg P)$
    e. $(P \wedge \neg P) \vee Q$

3. Write down the De Morgan's Law and prove it.
4. Prove Modus Ponens.

   A: If triangle is Equilateral.
   B: It is isosceles
5. Consider the following Sentences
   1. Mammals drink milk.
   2. Man is mortal.
   3. Man is mammals.
   4. Tom is a man.

   a. Tom drinks milk.
   b. Tom is mortal.

Assignment 6 questions:

1. What are the differences between Propositional Logic (PL) and First-Order Logic (FOL)?
2. Why we need an extension of PL?
3. What are the essential difference betweenx Quantifier and x Quantifier?
4. Translate in FOL:
   a. All dogs are faithful.
   b. Mammals drink milk.
   c. Man is mortal.
   d. Man is mammals.
   e. All birds cannot fly.
   f. At least one planet has life on it.
   g. Some dogs bark
   h. All dogs have four legs
   i. No dogs purr.
   j. Fathers are male parents with children.
   k. All barking dogs are irritating.
5. Show the similarity of PL and FOL and the extra power of FOL.
6. What is horn sentence?
7. Define Skoleminazation.

Assignment 7 questions:

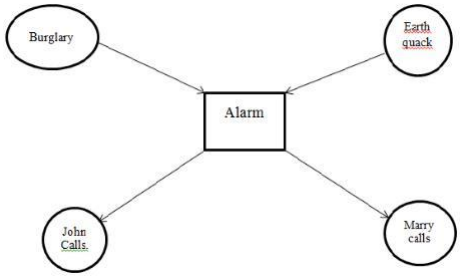
1. What is the major difference in resolution in PL and resolution in FOL?
2. Show the architecture of rule based system.
3. Define backtracking.
4. Define conflict resolution.
5. Write down the forward chaining algorithm.
6. Solve the problem for both forward and backward chaining. R1: IF hot AND
   smoke THEN add fire.
   R2: IF alarm beeps THEN add smoke.

R3: IF fires THEN add switch-sprinkles

Assignment 8 Questions:

1. Explain the limitations of bay's theorem.
2. Explain the concept of priori and post-priori with example.
3. State and prove bay's theorem.
4. Show how rules can be represented as conditional probability.
5. Write different types of inference in Belief Network.

6. Consider the following diagram:



What is the probability that the alarm has sounded at neither for burglary nor for earthquake has occurred and both marry and john calls. Where_

Probability of burglary, P(B)=0.001

Probability of earthquake, P(E)=0.002.

Probability of Marry calls, P(M|A)=0.70(true) and 0.01(false). Probability of John calls, P(J|A)=0.90(true) and 0.05(false)

Probability on different cases both for Burglary and Earthquake.

B	E	P(A| B,E)
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

Assignment 9 & 10 questions:

1. What is frame problem?
2. What is ramification problem?
3. Do you consider planning as search problem? If yes, then justify it.
4. Explain how planning system differ from classical search techniques.
5. Formulate the block world planning problem.
6. Find the similarity between basic search algorithm and forward planner algorithm.
7. Define “Sussman Anamoly”.
8. Explain why BS is much more effective than PS.
9. Prove why strip planner is unsound? How can you make it sound?
10. Describe via some example for which progressive planning is appropriate.
11. Describe via some example for where BR is more efficient.
12. What is flaw?
13. What is plan refinement? Elaborate with an example.
14. Describe the strips language for representing states goal and operations within a physical system.
15. Briefly explain why planning is difficult to cast as a state space search algorithm.
16. Represent information to differentiate between traditional computing and neural computing.
17. What is spike celestial impulse?
18. What is ANN?
19. What are the learning tasks?
20. Write down the Benefits of NN?
21. Write down the limitations of NN?
22. What is parsing?
23. What will be the parse tree of the following sentences:
24. What is ‘Anaphora’? Explain with example.
25. Explain why context becoming so important in NLP.
26. What are the procedures of learning in NLP?
27. Diagrammatically show a model of cross language information retrieval.

Answers : (Assignment: 01 – 10)

Assignment – 01 : Solution

1. **Write the algorithm for Basic Search.**
   Ans:
   Basic Search Algorithm:
   Let fringe be a list containing the initial state
   Loop
   if fringe is empty return failure
   Node → remove first (fringe)
   if Node is a goal
   then return the path from initial state to Node
   else generate all successors of node
   merge the newly generated nodes into fringe
   End
2. **Write the algorithm for Breadth-First Search.**
   Ans:
   Breadth-First Search:
   1. Place the starting node s on the queue
   2. If the queue is empty, return failure and stop
   3. If the first element on the queue is a goal node
   g, return success and stop. Otherwise
   4. Remove and expand the first element from the
   queue and place all the children at the end of
   the queue in any order
   5. Return to Step 2.

   Breadth-First Search Algorithm:
   Let fringe be a list containing the initial state
   Loop
   if fringe is empty return failure
   Node → remove first (fringe)
   if Node is a goal
   then return the path from initial state to Node
   else generate all successors of node
   merge the newly generated nodes into fringe
   [strategy: at the back of the fringe]
   End

   Note: Shallowest node expand first

3. **Write the algorithm for Depth-First Search.**
   Ans:
   Depth-first search:

1. Always expands the leaf node with greatest depth.
2. Last-in-first-out (LIFO) queue.

Depth-first search Algorithm:

```
Let fringe be a list containing the initial state
Loop
        if fringe is empty return failure
        Node → remove first (fringe)
        if Node is a goal
                then return the path from initial state to Node
        else expand deepest node first (DFS)
                        generate all successors of node (BFS)
                        if depth of node = limit return cutoff (DLS)
                merge the newly generated nodes into fringe
End
Note: strategy: add generated nodes to the front of the
```

4. **Write the algorithm for Graph Search.**

Ans:

Graph Search Algorithm:

```
Let fringe be a list containing the initial state
Let closed be initially empty
Loop
        if fringe is empty return failure
        Node → remove first (fringe)
        if Node is a goal
                then return the path from initial state to Node
        else put node in CLOSED
                generate all successors of node S
                for all nodes m in S
                        if m not in CLOSED
                        merge m into fringe
End Loop
```

5. **Write the algorithm for Depth Limited Search.**

Ans:

Depth Limited Search:

1. Expand deepest node first until depth limit reached.
2. Implementation:
   -Queueing-Fn: Append new nodes to end of queue.

Depth Limited Search Algorithm:

```
Let fringe be a list containing the initial state
Loop
        if fringe is empty return failure
        Node → remove first (fringe)
        if Node is a goal
                then return the path from initial state to Node
        else generate all successors of node
                merge the newly generated nodes into fringe
```

End

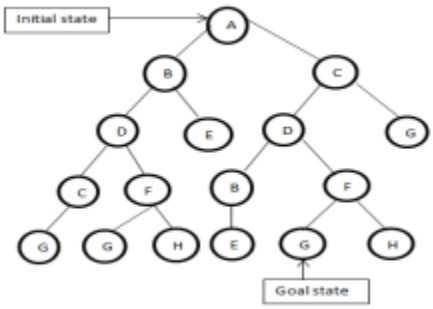
6. Write the differences between Uniform-Cost Search and Greedy Search algorithm.

Ans:

Uniform Cost Search Algorithm	Greedy Search Algorithm
Uniform-Cost pick the lowest total cost from the entire tree.	Greedy Search doesn't go back up the tree - it picks the lowest value and commits to that.
Move to goal seeing minimum backward cost.	Move to goal seeing minimum forward cost.

7) Find out the open and close list for the diagram below using Breadth-first and Depth-first Search algorithm.

Ans:



Breadth First Search:

exp. node	OPEN list	CLOSED list
	{A}	{}
A	{B C}	{A}
B	{C D E}	{A B}
C	{D E D G}	{A B C}
D	{E D G C F}	{A B C D}
E	{D G C F}	{A B C D E}
D	{G C F B F}	{A B C D E D}
G	{C F B F}	{A B C D E D G}
C	{F B F G}	{A B C D E D G C}
F	{B F G G H}	{A B C D E D G C F}
B	{F G G H E}	{A B C D E D G C F B}
F	{G G H E G H}	{A B C D E D G C F B F}
G	{G H D G H}	{A B C D E D G C F B F G}
G	{H E G H}	{A B C D E D G C F B F G G}
H	{E G H}	{A B C D E D G C F B F GG H}
E	{G H}	{A B C D E D G C F B F G G H E}
G	{H}	{A B C D E D G C F B F G G H E}

Depth First Search:

exp. node	OPEN list	CLOSED list
	{A}	{}
A	{B C}	{A}
B	{D E C}	{A B}
D	{C F C}	{A B D}
C	{G F C}	{A B D C}
G	{F C}	{A B D C G}
F	{G H C}	{A B D C G F}
G	{H C}	{A B D C G F G}
H	{C}	{A B D C G F G H}
C	{D G}	{A B D C G F G H C}
D	{B F G}	{A B D C G F G H C D}
B	{E F G }	{A B D C G F G H C D B}
E	{F G}	{A B D C G F G H C D B E}
F	{G H}	{A B D C G F G H C D B E F}
G	{H}	{A B D C G F G H C D B E F G}

Assignment – 02 : Solution

2.Can machines be comparable to human beings on behaving terms?

Ans. Yes,machines can be comparable to human beings on behaving terms.
Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.
AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.
While exploiting the power of the computer systems, the curiosity of human, lead him to wonder, “Can a machine think and behave like humans do?”
Thus, the development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans.

3.What is your opinion on "Machines will have minds"?

Ans. No engineer or chemist claims to be able to
produce a material which is indistinguishable from the human skin. It is possible that at some time this might be done, but even supposing this invention available we should feel there was little point in trying to make a "thinking machine" more human by dressing it up in such artificial flesh. The form in which we have set the problem reflects this fact in

the condition which prevents the interrogator from seeing or touching the other competitors, or hearing -their voices. Some other advantages of the proposed criterion may be shown up by specimen questions and answers. Thus:
Q: Please write me a sonnet on the subject of the Forth Bridge.
A : Count me out on this one. I never could write poetry.
Q: Add 34957 to 70764.
A: (Pause about 30 seconds and then give as answer) 105621.
Q: Do you play chess?
A: Yes.
Q: I have K at my K1, and no other pieces. You have only K at K6 and R at R1. It is your move. What do you play?
A: (After a pause of 15 seconds) R-R8 mate.
The question and answer method seems to be suitable for introducing almost any one of the fields of human endeavour that we wish to include. We do not wish to penalise the machine for its inability to shine in beauty competitions, nor to penalise a man for losing in a race against an aeroplane. The conditions of our game make these disabilities irrelevant. The "witnesses" can brag, if they consider it advisable, as much as they please about their charms, strength or heroism, but the interrogator cannot demand practical demonstrations.
The game may perhaps be criticised on the ground that the odds are weighted too heavily against the machine. If the man were to try and pretend to be the machine he would clearly make a very poor showing. He would be given away at once by slowness and inaccuracy in arithmetic. May not machines carry out something which ought to be described as thinking but which is very different from what a man does? This objection is a very strong one, but at least we can say that if, nevertheless, a machine can be constructed to play the imitation game satisfactorily, we need not be troubled by this objection.
It might be urged that when playing the "imitation game" the best strategy for the machine may possibly be something other than imitation of the behaviour of a man. This may be, but I think it is unlikely that there is any great effect of this kind. In any case there is no intention to investigate here the theory of the game, and it will be assumed that the best strategy is to try to provide answers that would naturally be given by a man.

4.Is it possible for machine to be intellegent?

Ans. Signs of intelligence-
Learn or understand from experience
• Make sense out of ambiguous or contradictory messages
• Respond quickly and successfully to new situations
• Use reasoning to solve problems
• Understand and infer in ordinary, rational ways
• Apply knowledge to manipulate the environment
• Think and reason
• Recognize the relative importance of different elements in a situation
Machines cannot be intellegent.

Assignment – 03 : Solution

1. What is Admissible Heuristic function?-Explain briefly.

Answer:

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.

An admissible heuristic is used to estimate the cost of reaching the goal state in an informed search algorithm . In order for a heuristic to be admissible to the search problem, the estimated cost must always be lower than or equal to the actual cost of reaching the goal state. The search algorithm uses the admissible heuristic to find an estimated optimal path to the goal state from the current node.

An admissible heuristic can be derived from relaxed version of the problem, or by information from pattern databases that store exact solutions to subproblems of the problem, or by using inductive learning methods.

If an admissible heuristic is adapted in an algorithm , then this algorithm would eventually find an optimal solution to the goal. A well-known and easy-to-understand example is breadth-first search . In the case of BFS, the heuristic evaluation function equals to for each node , which is obviously less than the actual cost (thus underestimated). This heuristic would cause the BFS algorithm to search literally all the possible paths and eventually find the optimal solution.

2. What is Evaluation Function?-Explain briefly.

Answer:

An evaluation function returns an estimate of the expected utility of the game from a given position, just as the heuristic functions of Chapter 3 return an estimate of the distance to the goal. The idea of an estimator was not new when Shannon proposed it. For centuries,chess players (and aficionados of other games) have developed ways of judging the value of a position because humans are even more limited in the amount of search they can do than are computer programs. It should be clear that the performance of a game-playing program depends strongly on the quality of its evaluation function. An inaccurate evaluation function will guide an agent toward positions that turn out to be lost. How exactly do we design good evaluation functions?

First, the evaluation function should order the terminal states in the same way as the true utility function: states that are wins must evaluate better than draws, which in turn must be better than losses. Otherwise, an agent using the evaluation function might err even if it can see ahead all the way to the end of the game. Second, the computation must not take too long! (The whole point is to search faster.) Third, for nonterminal states, the evaluation function should be strongly correlated with the actual chances of winning.

Most evaluation functions work by calculating various features of the state—for example, in chess, we would have features for the number of white pawns, black pawns, white queens, black queens, and so on. The features, taken together, define various categories or equivalence classes of states: the states in each category have the same values for all the features. For example, one category contains all two-pawn vs. one-pawn endgames. Any given category, generally speaking, will contain some states that lead to wins, some that lead to draws, and some that lead to losses. The evaluation function cannot know which states are which, but it can return a single value that reflects the proportion of states with each outcome. For example, suppose our experience suggests that 72% of the states encountered in the two-pawns vs. one-pawn category lead to a win (utility +1); 20% to a loss (0), and 8% to a draw (1/2). Then a reasonable evaluation for states in EXPECTED VALUE the category is the expected value: $(0.72 \times +1) + (0.20 \times 0)$

+ (0.08 × 1/2) = 0.76. In principle, the expected value can be determined for each category, resulting in an evaluation function that works for any state. As with terminal states, the evaluation function need not return actual expected values as long as the ordering of the states is the same.

3.Describe the two heuristic function related to 8 puzzle problem.

Answer:

The two heuristic function related to 8 puzzle problem are:

- Hamming distance
- Manhattan distance

Hamming distance:

The Hamming distance is the total number of misplaced tiles. It is clear that this heuristic is admissible since the total number of moves to order the tiles correctly is at least the number of misplaced tiles (each tile not in place must be moved at least once). The cost (number of moves) to the goal (an ordered puzzle) is at least the Hamming distance of the puzzle.

Manhattan distance:

Manhattan distance is the sum of the distances of the tiles from their goal positions. Because tiles cannot move along diagonals, the distance we will count is the sum of the horizontal and vertical distances. This is also called the city block distance. It is also admissible because all any move can do is move one tile one step closer to the goal.

4.Write down the Completeness, Time Complexity, Space Complexity, Optimality of the following:

BFS
DFS
DFID
Bidirectional Search

Answer:

BFS:
Completeness : Yes (if b is finite).
Time complexity: 1+b+b^2+b^3+...+b^d= O(b^d).
Space complexity: O(b^d).
Optimality: Yes,if uniform cost not optimal in general.

DFS:
Completeness : No,in infinite depth spacesin loops etc.
Time complexity: O(b^m),very bad where m is much larger than d.
Space complexity: O(bm). Only linear.
Optimality: No.

DFID:
Completeness: Yes.
Time complexity: (d+1)b^0+db+(d-1)b^2+...+b^d= O(b^d).
Space complexity: O(bd). Only linear.
Optimality: Yes, if step cost= 1.

Bidirectional Search:
Completeness: Yes.
Time complexity: O(b^d/2).
Space complexity: O(b^d/2).
Optimality:Yes

Assignment – 04 : Solution

1. Write down the MINIMAX algorithm.

Ans:

```
function MINIMAX-DECISION(state) returns an action
        return arg max MIN-VALUE(RESULT(state,a))
    function MAX-VALUE(state) returns a utility value
        if TERMINAL-TEST(state) then return UTILITY(state)
        for each a in ACTIONS(state) do
            v ← MAX(v,MIN-VALUE(RESULT(s,a)))
        return v
    function MIN-VALUE(state) returns a utility value
        if TERMINAL-TEST(state) then return UTILITY(state)
        for each a in ACTIONS(state) do
            v ← MIN(v,MAX-VALUE(RESULT(s,a)))
        return v
```

2. Write down the Alpha-Beta Pruning algorithm.

Ans:

```
function ALPHA-BETA-SEARCH(state) returns an action
        v ← MAX-VALUE(state)
        return the action in ACTIONS(state) with value v
    function MAX-VALUE(state) returns a utility value
        if TERMINAL-TEST(state) then return UTILITY(state)
        for each a in ACTIONS(state) do
            v ← MAX(v,MIN-VALUE(RESULT(s,a)))
        return v
    function MIN-VALUE(state) returns a utility value
        if TERMINAL-TEST(state) then return UTILITY(state)
        for each a in ACTIONS(state) do
            v ← MIN(v,MAX-VALUE(RESULT(s,a)))
        return v
```

3. Write down the Alpha-Beta Pruning algorithmwithfigure for MAX and MIN state.

Ans:

An example of Alpha-Beta Pruning algorithm with figure for MAX and MIN state is shown below:

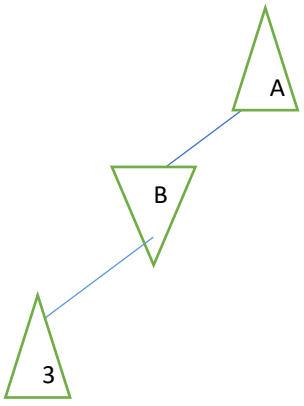


Fig: (a)

The first leaf below B has the value 3. Hence, B, which is a MIN node, has a value of at most 3.

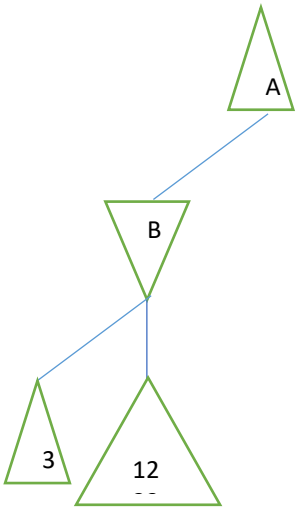


Fig: (b)

The second leaf below B has a value of 12.MIN would avoid this move. So the value of B is still at most 3.

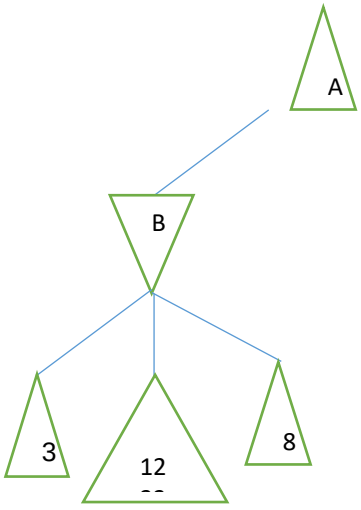


Fig: (c)

The third leaf below B has a value of 8. So the value of B is exactly 3. Now we can infer that the value of the root is at least 3, because MAX has a choice worth 3 at the root.

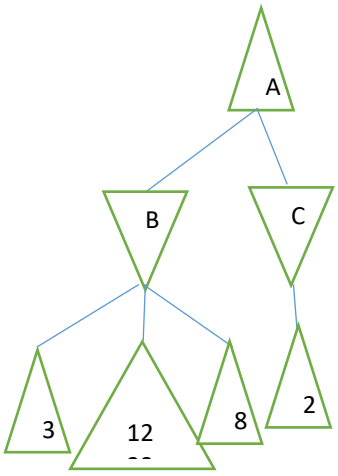


Fig: (d)

The first leaf below C has the value 2. Hence, C, which is a MIN node, has a value of at most 2. But B is worth 3. So MAX would never choose C. Therefore, there is no point in looking at the other successor states of C. This is an example of alpha-beta pruning.

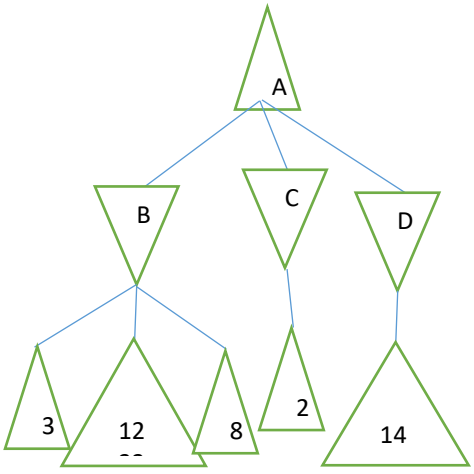


Fig: (e)

The first leaf below D has the value 14. So D is worth at most 14. This is still higher than MAX's best alternative(i.e., 3). So we need to keep exploring D's successor states. We now have bounds on all of the successors of the root. So the root's value is also at most 14.

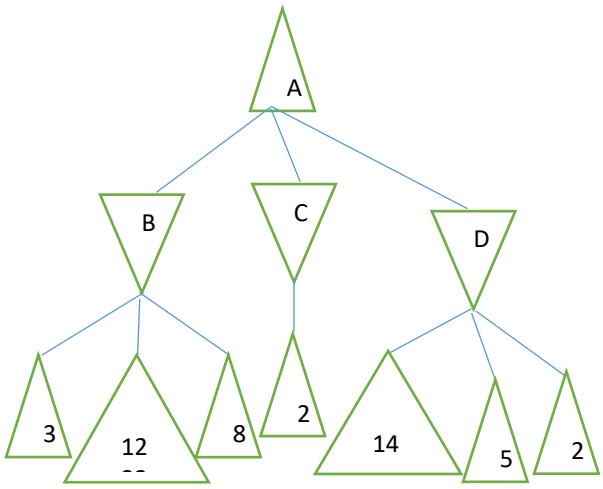


Fig: (f)

The second successor of D is worth 5 and the third successor is worth 2. So now D is worth exactly 2. MAX's decision at the root is to move to B, giving a value of 3.

Assignment – 05 : Solution

1. Translate the following propositional logics.

a) **It rains in July.**
b) **The book is not costly.**
c) **If it rains today and one does not carry umbrella, then he will be drenched.**
d) **If triangle is equilateral then it is isosceles.**
e) **ABC is an equilateral triangle.**

Answer:

a) rains(July): It rains in July.
b) ¬costly(book).
c) P = rains(today) : It rains today.
   Q = carry(umbrella, x): x carries umbrella.
   R = drenched(x): x is drenched.
   Answer is: $(P \wedge \neg Q) \rightarrow R$.
d) P = triangle(x): x is a triangle.
   Q = equilateral(x): x is equilateral.
   R = isosceles(x): x is isosceles.
   Answer is: $(P \wedge Q) \rightarrow R$.
e) triangle(ABC) ^ equilateral(ABC).

2. If P is true and Q is true then write down the truth table for the following:

a) **P→Q**
b) **(¬P$^{\vee}$Q)→Q**
c) **(¬P$^{\vee}$Q)→P**
d) **(P$^{\vee}$¬P)**
e) **(P^¬P)$^{\vee}$Q**

Answer:

a) Truth table for P→Q:

P	**Q**	**P→Q**
T	T	T
T	F	F
F	T	T
F	F	T

b) Truth table for (¬P∨Q)→Q:

P	**¬P**	**Q**	**¬P ∨Q**	**(¬P ∨Q)→Q**
T	F	T	T	T
T	F	F	F	T
F	T	T	T	T
F	T	F	T	F

c) Truth table for (¬P∨Q)→P:

P	**¬P**	**Q**	**¬P ∨Q**	**(¬P ∨Q)→P**
T	F	T	T	T
T	F	F	F	T
F	T	T	T	F
F	T	F	T	F

d) Truth table for (P ∨ ¬P):

P	**¬P**	**P∨¬P**
T	F	T
F	T	T

e) Truth table for (P^¬P)∨Q:

P	**¬P**	**Q**	**P^¬P**	**(P^¬P)∨Q**
T	F	T	F	T
T	F	F	F	F
F	T	T	F	T
F	T	F	F	F

3. Write down the De Morgan's Law and prove it.

Answer:

The De Morgan's Law:

I. ¬(P∨Q) = ¬P ^ ¬Q

Proof using truth table:

P	**¬P**	**Q**	**¬Q**	**P∨Q**	**¬(P∨Q)**	**¬P ∧ ¬Q**
T	F	T	F	T	F	F
T	F	F	T	T	F	F
F	T	T	F	T	F	F
F	T	F	T	F	T	T

II. ¬(P^Q) = ¬P ∨ ¬Q

Proof using truth table:

P	**¬P**	**Q**	**¬Q**	**P^Q**	**¬(P^Q)**	**¬P ∨ ¬Q**
T	F	T	F	T	F	F
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	T	F	T	F	T	T

4. Prove Modus Ponens.

OR

Prove that, (A∧¬A)∨B = B, where

A: If triangle is Equilateral.

B: It is isosceles.

Answer:

Modus Ponens: If **A** and **B** are two simple propositions defined on universe of discourse **X** then the compund proposition tropology

(A^(A→B))→B is called **Modus Ponens.**

Proof:

(A^(A→B))→B

- (A∧(¬A∨B))→B [Implication]
- ((A^¬A) ∨ (A ∧ B))→B [Distributibility]
- (**F** ∨ (A^B))→B [Law of Excluded Middle]
- (A^B)→B [Identity]
- ¬(A^B)∨B [Implication]
- (¬A ∨ ¬B)∨B [De Morgan's Law]
- ¬A ∨ (¬B∨B) [Associativity]
- ¬A ∨ **T** [Law of Excluded Middle]
- **T** [Identity]

5. Consider the following Sentences

1. **Mammals drink milk.**
2. **Man is mortal.**
3. **Man is mammals.**
4. **Tom is a man.**

Using modus ponens and resolution, Prove that,

(a).Tom drinks milk.

(b). Tom is mortal.

Answer:

drinks(x, y): x drinks y.
mortal(x): x is mortal.
mamal(x): x is a mamal.
man(x): x is a man.

Now,

mamal(Tom)→drinks(Tom, Milk) … (I)
man(Tom) →mortal(Tom) … (II)
man(Tom) →mamal(Tom) … (III)
man(Tom) … (IV)

Again,

1. ¬mamal(Tom) ˅ drinks(Tom, Milk)
2. ¬man(Tom) ˅ mortal(Tom)
3. ¬man(Tom) ˅ mamal(Tom)
4. man(Tom)

a) Tom drinks milk.

Proof using modus ponens:

Applying modus ponens in (III) and (IV): mamal(Tom) … (V)

Applying modus ponens in (I) and (V):

drinks(Tom, Milk) [proved]

Proof using resolution:

Try to prove ¬drinks(Tom, Milk) in contradiction.

¬drinks(Tom, Milk) ¬mamal(Tom) ∨ drinks(Tom, Milk)

¬mamal(Tom) ¬man(Tom) ∨ mamal(Tom)

¬man(Tom) man(Tom)

NULL CLAUSE

So, ¬drinks(Tom, Milk) couldn't be proved, therefore it is proved that,

drinks(Tom, Milk) [proved].

b) Tom is mortal.

Proof using modus ponens:

Applying modus ponens in (II) and (IV): mortal(Tom) [proved].

Proof using resolution:

Try to prove ¬mortal(Tom) in contradiction.

¬mortal(Tom) ¬man(Tom) ∨ mortal(Tom)

¬man(Tom) man(Tom)

NULL CLAUSE

So, ¬mortal(Tom) couldn't be proved, therefore it is proved that,

mortal(Tom) [proved].

Assignment – 06 : Solution

1. What are the differences between Propositional Logic (PL) and First-Order Logic (FOL)?

Answer: Difference between PL and FOL:

Propositional Logic	First Order Logic
1. PL is formal logical system that represents logical relations among sentences.	1. FOL is the symbolized reasoning in which each sentence is broken down into a subject and a predicate.
2. It is also called sentential logic/calculus.	2. It is also callded first order predicate logic (FOPL) or FOP Calculus.
3. It represents only facts which are true or false.	3. It can represent objects, facts and relations between objects and facts which are true or false.
4. An interpretation is simply an assignment of truth value to the atoms.	4. There are variables so we can do more that that.
5. Doesn't have quantifiers, functions, relation symbols.	5. It has quantifiers, functions and relation symbols.
6. It is weak, cannot talk directly about properties and relations between individuals.	6. It is expressive enough to represent any kind of information.

2. Why do we need an extension of PL?

Answer: We need an extention of PL because,

PL is formal logical system represents logical relations among sentences or propositions. It can represent only facts which are true or false but it cannot represent about objects and the relations of objects with the facts. In PL there is no variable that is why an interpretation is simply an assignment of truth value to the atom. PL also have no quantifiers, functions and relational symbol. It is a weak language that cannot talk directly about the properties of individuals and the relations between them.

To recover all of those limitations of PL we need an extension of PL. First Order Logic (FOL) is an extenstion of PL.

3. What are the essential differences between ∀x Quantifier and ∃x Quantifier?

Answer:

The essential differences between ∀x Quantifier and ∃x Quantifier:

∀x Quantifier	**∃x Quantifier**
1. ∀x means "for all x".	1. ∃x means "for some x".
2. ∀x **P(x)** is true if and only if for all the possible values of "x" P(x) is true.	2. ∃x **P(x)** is true if and only if there exists one or more "x" for which P(x) is true.
3. To prove this true every possible values of "x" must be considered.	3. To prove this true only a sinlge example is enough.
4. To prove it false only a single counter example is enough.	4. To prove it false every possible values of "x" must be considered.
5. Example: $\forall_{x \in R}(x^2 \geq 0)$**:** Square of all real numbers are positive.	5. Example: $\exists_{x \in R}(x^3 < 0)$**:** Cube of some real number is negative.

4. Translate in FOL:

a. All dogs are faithful.
b. Mammals drink milk.
c. Man is mortal.
d. Man is mammals.
e. All birds cannot fly.
f. At least one planet has life on it.
g. Some dogs bark
h. All dogs have four legs
i. No dogs purr.
j. Fathers are male parents with children.
k. All barking dogs are irritating.

Answer:

a. faithful(x): x is faithful.
dog(x): x is a dog.
$\forall_x$(dog(x) →faithful(x))

b. mamal(x): x is a mamal.
drink(x, y): x drinks y.
$\forall_x$(mamal(x) →drinks(x, Milk))

c. man(x): x is a man.
mortal(x): x is mortal.
$\forall_x$(man(x) →mortal(x))

d. $\forall x(man(x) \rightarrow mamal(x))$

e. bird(x): x is a bird.
fly(x): x can fly.
$\neg \forall x(bird(x) \rightarrow fly(x))$

f. planet(x): x is a planet.
has_life(x): x has life on it.
$\exists x(planet(x) \wedge has\_life(x))$

g. bark(x): x barks.
$\exists x(dog(x) \wedge bark(x))$

h. has_legs(x, y): x has y legs.
$\forall x(dog(x) \rightarrow has\_legs(x, 4))$

i. purr(x): x purrs.
$\forall x(dog(x) \rightarrow \neg purr(x))$

j. father(x, y): x is father of y.
parent(x, y): x is parent of y.
male(x): x is male.
has_child(x): x has child.
$\forall x \exists y(father(x, y) \rightarrow parent(x, y) \wedge male(x) \wedge has\_child(x))$

k. irritating(x): x is irritating.
$\forall x(dog(x) \wedge bark(x) \rightarrow irritating(x))$

5. Show the similarity of PL and FOL and the extra power of

FOL. Answer: Similarities between PL and FOL:

i. Both PL & FOL are used for knowledge representation.
ii. FOL is a proper extension of the PL that defines which statements are provable.
iii. FOL is the generalization of PL. If PL is defined with a suitable set of axioms and then single rule of inference modus ponens, then the FOL can be defined by appending to the propositional logic several axioms and the inference rule called "Universal Generalization".
iv. Unlike the PL, FOL is undecidable, but Moodic FOL and Bernays –sckonfinkel class of FOL formulas are decidable.
v. Syntactically, FOL has the same connectives as PL.
vi. FOL & PL, both systems are known to be consistent e.g. by exhibiting models in which axioms are satisfied.

Extra Power of FOL:

i. FOL has variables & quantifiers, functions and relational symbols. FOL is expressive enough to represent any kind of information. Syntax of FOL can be

determined in terms of __

- Terms
- Predicates
- Quantifiers

i. FOL can capture the sentence in a natural way. But PL cannot do this. FOL is used to model the world in term of __

- Objects
- Properties
- Classes
- Relations
- Functions
- Predicates

So, FOL has the extra power than the PL.

6. What is horn sentence?

Answer:

A sentence is called a horn sentence if it satisfies the following properties:

i. It is an atomic sentence.
ii. Or, It is an implication with a conjunction of atomic sentences on the left and a single atom on the right.
iii. It does not have any existential quantifier.

Examples of horn sentences:

a) perfect_square (36)
b) $\forall_{x,y}$ (perfect _sq (x)$\wedge$prime (y)$\wedge$ divides (x,y)$\rightarrow$divides (x, square (y))

7. Define Skoleminazation.

Answer:

Every first-order formula may be converted into Skolem normal form while not changing its satisfiability via a process called Skolemization (sometimes spelled Skolemnization). The resulting formula is not necessarily equivalent to the original one, but is equisatisfiable with it: it is satisfiable if and only if the original one is satisfiable.

It is a method of removing the existential quantifiers from a first order logic sentence. If the existential quantifier is outside any universal quantifier, a skolem constant is introduced e.g. $\exists$y prime(p)$\Rightarrow$prime(p), where p is a Skolem constant
Otherwise a skolem function is introduced.

For example, $\forall_x\exists_y$prime (y)$\wedge$divides(x,y) $\Rightarrow\forall_x$prime(PD(x))$\wedge$divides(x, PD(x)), where PD(x) is a Skolem function.

Assignment – 07 : Solution

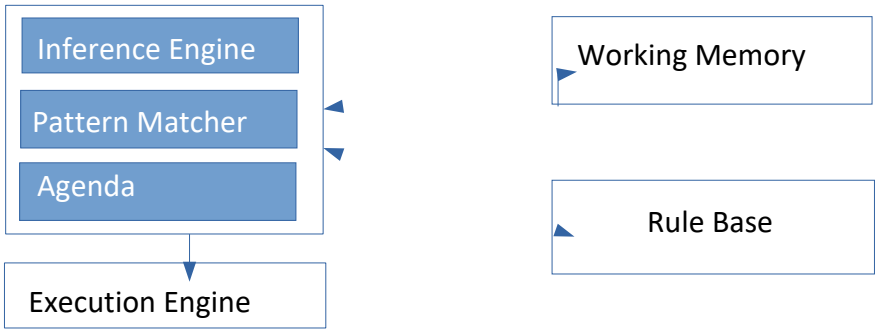
1. What is the major difference in resolution in PL and resolution in FOL?

Answer: Difference between Resolution in PL and Resolution in FOL:

Resolution in PL	Resolution in FOL
1. Resolution rule is single valid inference rule.	1. Resolution rule can be generalized in FOL.
2. PL can deal with only a few number of propositions.	2. FOL allows us to express and infer arguments in infinite models.
3. The expression power is weak, cannot express the universal truth.	3. FOL is generalized form of PL.
4. PL has no variable.	4. Variables and quantifiers are added.
5. Syntax of PL defines only the allowable sentences in a simple way.	5. Syntax of FOL can be determined in terms of term-predicate Quantifier.

2. Show the architecture of rule based system.

Answer:
Here is the architecture of a typical rule based system_



3. Define backtracking.

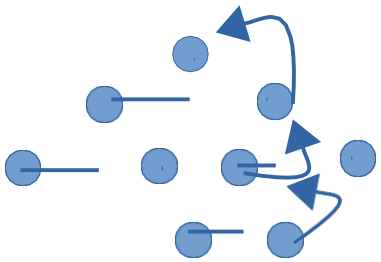
Answer:

Backtracking is a general algorithm for finding all (or some) solutions to some computational problems, notably constraint satisfaction problems, that incrementally builds candidates to the solution.

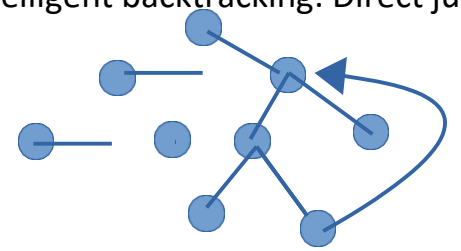
Performance of backtracking can be improved by using a number of techniques such as variable ordering, value ordering, back jumping and forward checking.

There are two types of backtracking strategies_

1) Chronological backtracking: One level up backtracking



2) Intelligent backtracking: Direct jump backtracking_



4. Define conflict resolution.

Answer:

Conflict resolution strategies are used in production systems in AI, such as in rule based expert systems to help in choosing which production rule to fire.

There are many strategies for conflict resolution. Some of them are given below:-

a) FCFS
b) Specify Ordering
c) Fire All (Fired Rules)
d) Heuristic Measure
e) LIFO
f) Prioritization
g) Line of reasoning
h) Data driven search space (given data, facts)
i) Goal driven search space (goal to fact).

5. Write down the forward chaining algorithm.

Answer:

function FOL-FC_Ask(kB, a): returns a substitution or false.
Input: kB, the knowledge base, a set of first order definite clauses. Local variables: new, the new sentences inferred on each iteration.

Repeat until new is
    empty new ← {}
    for each rule in kB do
        $(P_1 \wedge ... \wedge P_n \rightarrow q) \leftarrow$ STANDARDIZE-VARIABLES
        for each Q such that SUBST(Q, $P_1 \wedge ... \wedge P_n$) = SUBST(Q, $P'_1 \wedge ... \wedge P'_n$)
    for some $P'_1 \wedge ... \wedge P'_n$ in kB
        $q' \leftarrow$ SUBST(Q, q)
        if q' doesn't unify with some sentence already in kB or new then add q' to new
        $\phi \leftarrow$ UNIFY(q', a)
        if ϕ is not fit then return
    ϕ add new to kB
return false

6. Solve the problem for both forward and backward chaining.

- **R1: IF hot AND smoke THEN add fire.**
- **R2: IF alarm beeps THEN add smoke.**
- **R3: IF fires THEN add switch-sprinkles**

DB of facts
Alarm beeps
Hot

Answer:

Using forward chaining:

Rule R2 Fired :

DB of facts
Alarm beeps
Hot
Smoke

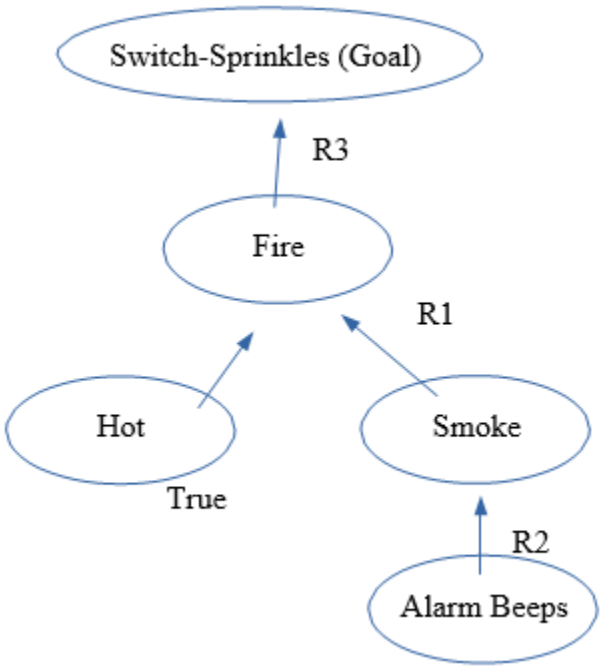
Rule R1 Fired:

DB of facts
Alarm beeps
Hot
Smoke
Fire

Rule R3 Fired:

DB of facts
Alarm beeps
Hot
Smoke
Fire
Switch-Sprinkles (Goal)

Using backward chaining:



Assignment – 08 : Solution

1. **Explain the limitations of bay's theorem.**

   In parameter estimation, you are trying to determine the most likely value of a parameter in some model, given some data set. In very broad-brush terms, when we approach this with Bayesian parameter estimation, we identify in Bayes' Law $P(B|A) = \frac{P(A|B)P(B)}{P(A)}$ the quantity A with some experimental results and the quantity B with some parameterized model. P(A|B) is then identified with how we expect the data to vary depending on the parameters, which is typically known. P(B) is called the "prior'': it represents prior knowledge about the parameter. The quantity related to P(A) comes from the data. What you are then after is P(B|A), the "posterior'', related to "probability of the parameter given the data". So to determine P(B|A), We have to have some reasonable distribution to assign to the prior. If we don't really know anything about the parameter we are after, it can be tricky to decide what to use for the prior, and the answer can be sensitive to what you choose. Bayes' Law is a tremendously powerful tool in many statistical contexts, in particular when a sensible prior can be determined, because it gives a highly effective way to incorporate knowledge you already have. However, when it is hard to choose a sensible prior, the method can give dubious results.

2. **Explain the concept of priori and posteriori with example.**

   Example of A Priori Probability:

   For example, consider flipping a coin. A fair coin has two different sides and each time you flip it has an equal chance of landing on either side, regardless of the previous toss outcome. The a priori probability of landing on the "heads" side of the coin is 50%. Another example is how the price of a share can change. Its price can increase, decrease or remain the same. Therefore, according to a priori probability, we can assume that there is a 1-in-3, or 33%, chance of one of the outcomes occurring (all else remaining equal).

   Example of A Posterior Probability:

   As a simple example to envision posterior probability, suppose there are three acres of land with labels A, B and C. One acre has reserves of oil below its surface, while the other two do not. The prior probability of oil in acre C is one-third, or 33%. A drilling test is conducted on acre B, and the results indicate that no oil is present at the location. With acre B eliminated, the posterior probability of acre C containing oil becomes 0.5, or 50%.

3. State and prove Bayes' theorem

Bayes' theorem relates the conditional and marginal probabilities of stochastic events A and B:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Each term in Bayes' theorem has a conventional name:

- P(A) is the prior probability or marginal probability of A. It is "prior" in the sense that it does not take into account any information about B.
- P(A|B) is the conditional probability of A, given B. It is also called the posterior probability because it is derived from or depends upon the specified value of B.
- P(B|A) is the conditional probability of B given A.
- P(B) is the prior or marginal probability of B, and acts as a normalizing constant.

Proof:

The probability of two events A and B happening, P(A∩B), is the probability of A, P(A), times the probability of B given that A has occurred, P(B|A).

$$P(A \cap B) = P(A)P(B|A) \quad \text{..... (1)}$$

On the other hand, the probability of A and B is also equal to the probability of B times the probability of A given B.

$$P(A \cap B) = P(B)P(A|B) \quad \text{…….. (2)}$$

Equating the two yields:

$$P(B)P(A|B) = P(A)P(B|A) \quad \text{……..(3)}$$

and thus

$$P(A|B) = P(A)\frac{P(B|A)}{P(B)}$$

……...…….(4)

This equation, known as Bayes Theorem is the basis of statistical inference.

4. Show how rules can be represented as conditional probability.

Rules for conditional probabilities: The function PB(A) = P(A|B), considered as a function of A, with the conditioning event B fixed, satisfies the Kolmogorov
axioms, and any rules derived from these axioms, in particular:

- Addition formula for conditional probabilities: P(A1 [A2 [· · · |B) = P(A1|B) + P(A2|B) + · · · for mutually disjoint events A1,A2, ...
- Complement rule for conditional probabilities: P(Ac | B) = 1 − P(A|B)

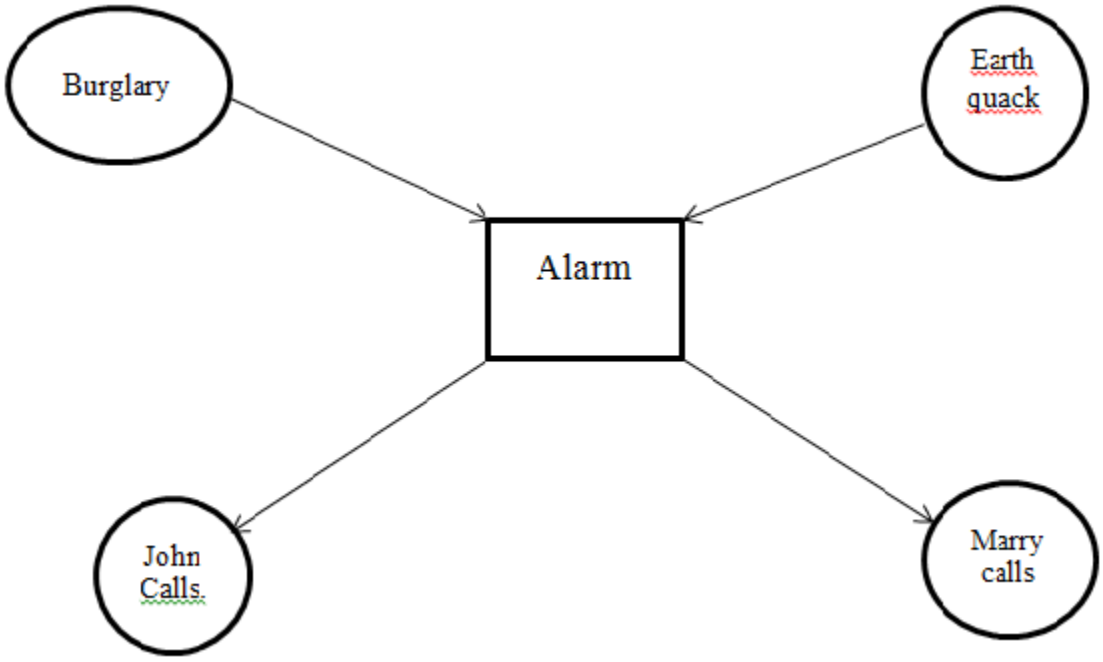
- Multiplication rule (or chain rule) for conditional probabilities: P(AB) = P(A|B)P(B). More generally, for any events A1, . . . ,An, P(A1A2 . . .An) = P(A1)P(A2|A1)P(A3|A1A2) . . . P(An|A1A2 . . .An−1).

5. Write different types of inference in Belief Network

Different types of inference in Belief Network:

1. Diagnostic inference (from effects to causes)
2. Causal inference (from causes to effects)
3. Intercausal inference (between causes of a common effect)
4. Mixed inference (combination of the above)

6. Consider the following diagram:



What is the probability that the alarm has sounded at neither for burglary nor for earthquake has occurred and both marry and john calls. Where_

- Probability of burglary, P(B)=0.001
- Probability of earthquake, P(E)=0.002.
- Probability of Marry calls, P(M|A)=0.70(true) and 0.01(false).
- Probability of John calls, P(J|A)=0.90(true) and 0.05(false)
- Probability on different cases both for Burglary and Earthquake.

B	E	P(A| B,E)
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

$$P(!B \cap !E \cap A \cap J \cap M) = P(J|A) * P(M|A) * P(A|\ !E \cap !B) * P(!E) * P(!B)$$
$$= 0.9 * 0.7 * 0.001 * (1 - 0.002) * (1\text{-}0.001)$$
$$= 0.628$$

Assignment – 09&10 : Solution

1. What is frame problem?

In the confined world of a robot, surroundings are not static. Many varying forces or actions can cause changes or modifications to it. The problem of forcing a robot to adapt to these changes is the basis of the frame problem in artificial intelligence. Information in the knowledge base and the robot's conclusions combine to form the input for what the robot's subsequent action should be. A good selection from its facts can be made by discarding or ignoring irrelevant facts and ridding of results that could have negative side effects.

The frame problem within artificial intelligence concerns the application of knowledge about the past to draw inferences about the future. It requires distinguishing those properties that change across time against a background of those properties that do not, which thus constitute a frame. From the point of view of philosophy it appears to be a special case of the problem of induction, which requests justification for drawing inferences about the future based on knowledge of the past. David Hume, in particular, suggested that one's expectations about the future are no more than habits of the mind and doubted that knowledge relating the future to the past was possible.

2. What is ramification problem?

In artificial intelligence, especially knowledge based systems, the ramification problem is concerned with the indirect consequences of an action. It might also be posed as how to represent what happens implicitly due to an action or how to control the secondary and tertiary effects of an action.

This problem describes how an action can cause deviations within its environment. For example, a robotic arm has been given the task of picking up a brick and placing it on its side in a different location. If the brick has been knocked over, what can the robot do to rectify the problem? Will it still know which side should be facing up without the ability of human sight? Should these deviations be examined individually each time an action has taken place?

This is the basis of ramification problem.

3. Do you consider planning as search problem? If yes, then justify it.

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. Automated planners find the transformations to apply in each given state out of the possible transformations for that state. On the other hand, search is the systematic examination of states to find path from the start/root state to the goal state. The set of possible states, together with operators defining their connectivity constitute the search space. The output of a search algorithm is a solution, that is, a path from the initial state to a state that satisfies the goal test.

Yes, I consider planning to be a search problem, or at least, a planning problem can be reduced to a search problem.

Search is an almost totally generic construct: there is a space of possibilities, and you want to locate one, but you have to find it by inspecting a subset. There are all manner of details as to the specific search problem and particular search algorithm. Pretty much any problem can be posed as a search problem (what's the space of possibilities, and how do you tell which is a desired one).

Planning is a particular kind of search, it is a search through the space of action sequences (or more generally, partial orders) for a plan that satisfies some criteria. That doesn't mean it has to be implemented as a search, but the problem can be described that way.

4. Explain how planning system differ from classical search techniques.

The main difference between search and planning is the representation of states. In search, states are represented as a single entity (which may be quite a complex object, but its internal structure is not used by the search algorithm). In planning, states have structured representations (collections of properties) which are used by the planning algorithm.

5. Formulate the block world planning problem.

The blocks world is one of the most famous planning domains in artificial intelligence. In this problem, a set of wooden blocks of various shapes and colors sitting on a table are imagined. The goal is to build one or more vertical stacks of blocks. The catch is that only one block may be moved at a time: it may either be placed on the table or placed atop another block. Because of this, any blocks that are, at a given time, under another block cannot be moved. Moreover, some kinds of blocks cannot have other blocks stacked on top of them.

6. Find the similarity between basic search algorithm and forward planner algorithm.

A deterministic plan is a sequence of actions to achieve a goal from a given starting state. A deterministic planner is a problem solver that can produce a plan. The input to a planner is an initial world description, a specification of the actions available to the agent, and a goal description. The specification of the actions includes their preconditions and their effects.
One of the simplest planning strategies is to treat the planning problem as a path planning problem in the state-space graph. In a state-space graph, nodes are states, and arcs correspond to actions from one state to another. The arcs coming out of a state s correspond to all of the legal actions that can be carried out in that state. That is, for each state s, there is an arc for each action a whose precondition holds in state s, and where the resulting state does not violate a maintenance goal. A plan is a path from the initial state to a state that satisfies the achievement goal.
A forward planner searches the state-space graph from the initial state looking for a state that satisfies a goal description.
Basic search algorithms consists of informed search (greedy, A*) and uninformed search (BFS, DFS). Both search algorithms and forward planner algorithms have a definite goal of finding optimal solution to a problem.

7. Define “Sussman Anomaly”.

The Sussman anomaly is a problem in artificial intelligence, first described by Gerald Sussman, that illustrates a weakness of non interleaved planning algorithms.
In the problem, three blocks (labeled A, B, and C) rest on a table. The agent must stack the blocks such that A is atop B, which in turn is atop C. However, it may only move one block at a time. The problem starts with B on the table, C atop A, and A on the table.
However, non interleaved planners typically separate the goal (stack A atop B atop C) into subgoals, such as:

1. get A atop B
2. get B atop C

Suppose the planner starts by pursuing Goal 1. The straightforward solution is to move C out of the way, then move A atop B. But while this sequence accomplishes Goal 1, the agent cannot now pursue Goal 2 without undoing Goal 1, since both A and B must be moved atop C:
If instead the planner starts with Goal 2, the most efficient solution is to move B. But again, the planner cannot pursue Goal 1 without undoing Goal 2:
The problem was first identified by Sussman as a part of his PhD research. Most modern planning systems can handle this anomaly, but it is still useful for explaining why planning is non-trivial.

8. Explain why BS is much more effective than PS.
Planning is the process of computing several steps of a problem-solving procedure before executing any of them. The main difference between search and planning is the representation of states.
In search, states are represented as a single entity (which may be quite a complex object, but its internal structure is not used by the search algorithm)
In planning, states have structured representations (collections of properties) which are used by the planning algorithm.
BS or Basic Search is effective than Progressive Search (PS) because BS can be applied everywhere automatically, but PS will require an intelligent being (human) to plan it for the search algorithm before it can take control.

9. Prove why STRIP planner is unsound? How can you make it sound?
STRIPS planner is a Divide and conquer algorithm. The procedure is, to create a plan to achieve a conjunction of goals, create a plan to achieve one goal, and then create a plan to achieve the rest of the goals.
To achieve a list of goals:
- ➢ choose one of them to achieve.
- ➢ If it is not already achieved
  - ➢ choose an action that makes the goal true
  - ➢ achieve the preconditions of the action
  - ➢ carry out the action
- ➢ achieve the rest of the goals.

The STRIPS algorithm, as presented, is unsound. Achieving one subgoal may undo already achieved subgoals.
There are Two ideas to make STRIPS sound:-

1. Protect subgoals so that, once achieved, until they are needed, they cannot be undone. Let remove return different choices.
2. Reachieve subgoals that have been undone. Protecting subgoals makes STRIPS incomplete. Re-achieving subgoals finds longer plans than necessary.

10. Describe via some example for which progression planning is appropriate.
Answer: As planning problems can be mapped as planning problem maps, we can use heuristic search algorithm to solve this planning problems. Using forward search to solve this sort of problems is known as Progression planning. In this state-space search approach, we have to

explore every possible state to proceed from a start state to a goal state. Although, this is considered inefficient because of the searching of irrelevant actions but also it can take a long time for large state-space which most of planning problems has. But with heuristics with planning, a relaxed problem and applying constraints can solve the problem with much less complexity which makes progression planning feasible.
To consider a scenario for which progression planning is appropriate, lets consider a shop with 3 jars with different shapes of candies in it. The owner wants to mix the candies with such amounts and balance that it will look appealing to the customer. Generally, there is no defined goal state of this problem. So, to solve this problem the owner has to gradually work his way around in every possible way the candies can be put in the jars. A rather meticulous but tiresome work but with heuristic planning we can put on some relaxed condition such as all jars must have 20 candies at least thus abstracting the problem sets and they must have mixed colors to a ratio of 1:2:3 thus ignoring the pre-condition.
For these small planning space-states problems, progression search can be used with a bit help of the heuristic analysis of the states-space by us, the human beings.

11. Describe via some example for where BR is more efficient.
Answer: BR (Backward regression) uses backward search to solve a planning problem. To avoid problems with forward search, it performs the search of the state-space only with the relevant states to the goal state. This helps reducing unnecessary searches done through out the state-space but sometimes it can miss a feasible solution just because it abandons plans-state based on the goal state.
Let's consider a situation, where we have to buy a book. To find that book we can visit suppose 50 shops in a city and thus try to find the book in those shops. This is obviously using forward search and would cost a lot of time. To perform the backward approach, we first have to know which shops carries the categories of books we want. And thus, reducing the visits to unnecessary shops. This can also be used in many other problems such as rearranging some books on a shelf. To know the final state, we can sort out the book in the optimal way and reduce unnecessary moving of the books.

12. **What is Flaw?**
Answer: A flaw is anything that keeps the partial plan from being a solution. Partial plan is partially ordered structures of plans. And if a flaw is detected then an action is added to the partial plan to solve the problem. It can also be fixed by removing the pre-conditions. We try to make the least commitment at each state to remove the flaws.

13. **What is plan refinement? Elaborate with an example.**
Answer: Plan refinement means detecting flaws in the partially sorted plans and adding nodes or actions to solve the flaws to gradually advance towards the goal state.
To make partially sorted plans, we start with the empty plan containing nothing but the initial state and the goal state. After that we need to add an action to solve the "Flaw" of solving the problem.
Let's consider the scenario, where a flat tire needs to be changed with the spare tire.
So, we get the initial state with (Flat tire, Axle) and (Spare tire, Trunk) and the goal state is (Spare tire, Axle). To solve this, we need to add one action of PUT ON (Spare tire, Axle). This is a partially sorted plan. But to solve the flaws within this, we have to take out the spare tire out of

the trunk. Thus, we have to add another action REMOVE (Spare tire, Trunk). By adding the actions on each stage, we can devise a sorted plan to go from initial state to the goal state.

14. **Describe the strips language for representing states goal and operations within a physical system.**

Answer: The Stanford Research Institute Problem Solver, known by its acronym STRIPS, is an automated planner developed by Richard Fikes and Nils Nilsson in 1971 at SRI International.

A STRIPS instance is composed of:

- An initial state
- The specification of the goal states – situations which the planner is trying to reach
- A set of actions. For each action, the following are included:
    - preconditions (what must be established before the action is performed)
    - postconditions (what is established after the action is performed).

Mathematically, a STRIPS instance is a quadruple <P, O, I, G>, in which each component has the following meaning:

1. P is a set of conditions
2. O is a set of operators (i.e., actions); each operator is itself a quadruple <A, B, C,D> , each element being a set of conditions. These four sets specify, in order, which conditions must be true for the action to be executable, which ones must be false, which ones are made true by the action and which ones are made false
3. I is the initial state, given as the set of conditions that are initially true (all others are assumed false)
4. G is the specification of the goal state; this is given as a pair <N, M>, which specify which conditions are true and false, respectively, in order for a state to be considered a goal state.

A plan for such a planning instance is a sequence of operators that can be executed from the initial state and that leads to a goal state.

15. **Briefly explain why planning is difficult to cast as a state space search algorithm.**

Answer: Intelligent agency involves controlling the evolution of external environments in desirable ways. Planning provides a way in which the agent can maximize its chances of effecting this control. A planning may include many possibilities and entities that needs to be modified. To cast them into a state space means we have to find out every possible state for these entities and the variables with the controls. For a large amount of entities and variables it is impossible to find out all the states and cast them. Although many languages and techniques are used to ease these problems but to cast every possible state into the state space search is quite impossible. And to perform a search on the state space without the adequate amount of states won't perform any optimal result. So, to cast the planning, we need a heuristic analysis

of the state space by a more diligent system such a human being to reshape the state space to perform the search algorithm.

16. **Represent information to differentiate between traditional computing and neural computing.**

Topic	Traditional computing	Neural computing
1. Processing	Processing is sequential for traditional computers.	Parallel processing is done with neural computing.
2. Multiprocessor	Traditional computing although says to have multiprocessor but won't do multiprocessing.	Neural computing is multiprocessor friendly system.
3. Functioning ways	Traditional computers functions with a set of rules and calculations	Neural computers can function via images, pictures and concepts.
4. Learning ways	It learns from rules.	It can learn by doing something and then learning from it.
5. Self-programming	Traditional computers must learn by doing different sequences or steps in an algorithm.	Neural networks are continuously adaptable by truly altering their own programming
6. Speed	Speed depends on either big processors or the tedious, error-prone idea of parallel processors.	Neural networks require the use of multiple chips custom-built for the application

17. What is spike celestial impulse?

Spiking neural networks (SNNs) are artificial neural network models that more closely mimic natural neural networks. In addition to neuronal and synaptic state, SNNs also incorporate the concept of time into their operating model. The idea is that neurons in the SNN do not fire at each propagation cycle (as it happens with typical multi-layer perceptron networks), but rather fire only when a membrane potential – an intrinsic quality of the neuron related to its membrane electrical charge – reaches a specific value. When a neuron fires, it generates a signal which travels to other neurons which, in turn, increase or decrease their potentials in accordance with this signal.

In the context of spiking neural networks, the current activation level (modeled as some differential equation) is normally considered to be the neuron's state, with incoming spikes pushing this value higher, and then either firing or decaying over time. Various *coding methods* exist for interpreting the outgoing s*pike train* as a real-value number, either relying on the frequency of spikes, or the timing between spikes, to encode information.

18. What is ANN?

An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output. ANNs are considered nonlinear statistical data modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.

19. What are the learning tasks?

Following are top 8 most common machine learning tasks that one could come across most frequently while solving an advanced analytics problem:

1. **Feature Selection**: Feature selection is one of the critical tasks which would be used when building machine learning models. Feature selection is important because selecting right features would not only help build models of higher accuracy but also help achieve objectives related to building simpler models, reduce overfitting etc. The following are some of the techniques which could be used for feature selection:
    - Filter methods which helps in selecting features based on the outcomes of statistical tests. The following are some of the statistical tests which are used:
        - Pearson's correlation
        - Linear discriminant analysis (LDA)
        - Analysis of Variance (ANOVA)
        - Chi-square tests
    - Wrapper methods which helps in feature selection by using a subset of features and determining the model accuracy. The following are some of the algorithms used:
        - Forward selection
        - Backward elimination
        - Recursive feature elimination
    - Regularization techniques which penalizes one or more features appropriately to come up with most important features. The following are some of the algorithms used:
        - LASSO (L1) regularization
        - Ridge (L2) regularization
2. **Regression**: Regression tasks mainly deal with estimation of numerical values (**continuous variables**). Some of the examples include estimation of housing price, product price, stock price etc. Some of the following ML methods could be used for solving regressions problems:
    - Kernel regression *(Higher accuracy)*
    - Gaussian process regression *(Higher accuracy)*
    - Regression trees
    - Linear regression
    - Support vector regression
    - LASSO
3. **Classification**: Classification tasks is simply related with predicting a category of a data (**discrete variables**). One of the most common example is predicting whether or not an email if spam or ham. Some of the common use cases could be found in the area of healthcare such as whether a person is suffering from a particular disease or not. It also has its application in financial use cases such as determining whether a transaction is fraud or not. The ML methods such as following could be applied to solve classification tasks:
    - Kernel discriminant analysis *(Higher accuracy)*
    - K-Nearest Neighbors *(Higher accuracy)*
    - Artificial neural networks (ANN) *(Higher accuracy)*

- Support vector machine (SVM) *(Higher accuracy)*
- Random forests *(Higher accuracy)*
- Decision trees
- Boosted trees
- Logistic regression
- naive Bayes
- Deep learning

4. **Clustering**: Clustering tasks are all about finding natural groupings of data and a label associated with each of these groupings (clusters). Some of the common example includes customer segmentation, product features identification for product roadmap. Some of the following are common ML methods:
    - Mean-shift *(Higher accuracy)*
    - Hierarchical clustering
    - K-means
    - Topic models
5. **Multivariate querying**: Multivariate querying is about querying or finding similar objects. Some of the following ML methods could be used for such problems:
    - Nearest neighbors
    - Range search
    - Farthest neighbors
6. **Density estimation**: Density estimation problems are related with finding likelihood or frequency of objects. In probability and statistics, density estimation is the construction of an estimate, based on observed data, of an unobservable underlying probability density function. Some of the following ML methods could be used for solving density estimation tasks:
    - Kernel density estimation *(Higher accuracy)*
    - Mixture of Gaussians
    - Density estimation tree
7. **Dimension reduction**: As per Wikipedia page on Dimension reduction , Dimension reduction is the process of reducing the number of random variables under consideration, and can be divided into feature selection and feature extraction. Following are some of ML methods that could be used for dimension reduction:
    - Manifold learning/KPCA *(Higher accuracy)*
    - Principal component analysis
    - Independent component analysis
    - Gaussian graphical models
    - Non-negative matrix factorization
    - Compressed sensing
8. **Testing and matching**: Testing and matching tasks relates to comparing data sets. Following are some of the methods that could be used for such kind of problems:
    - Minimum spanning tree
    - Bipartite cross-matching

- N-point correlation

20. Write down the Benefits of NN?

Benefits of NN is given below :

1. A neural network can perform tasks that a linear program can't

2. When an element of the neural network fails, it can continue without any problem by their parallel nature.

3. A neural network learns and doesn't need to be reprogrammed.

4. It works even in the presence of noise with good quality output.

Write down the limitations of NN?

21. Write down the limitations of NN?

Limitations of NN is given below:

1. High cost:

    i) Each neuron in the neural network can be considered as logistic regression.

    ii) Training the entire neural network is to train all the interconnected logistic regressions.

2. Difficult to train as the number of hidden layers increases:

    i) Recall the logistic regression is trained by gradient descent.

    ii) In backpropagation, gradient is progressively more dilute.

3. Stuck in local optima

    i) The objective function of the neural network is usually not convex.

    ii) The random initialization does not guarantee starting from the proximity of global optima.

4. Solution:

    Deep learning/ learning multiple levels of representation.

22. What is parsing?

Parsing is the formal analysis by a computer of a sentence or other string of words into its constituents, resulting in a parse tree showing their syntactic relation to each other, which may also contain semantic and other information.

23. What will be the parse tree of the following sentences:

There's no sentence available. So there won't be any parse tree.

24. What is 'Anaphora'? Explain with example.

Anaphora is a figure of speech in which words repeat at the beginning of successive clauses, phrases, or sentences. For example, Martin Luther King's famous "I Have a Dream" speech

contains anaphora: "So let freedom ring from the prodigious hilltops of New Hampshire. Let freedom ring from the mighty mountains of New York. Let freedom ring from the heightening Alleghenies of Pennsylvania…"

25. Explain why context becoming so important in NLP.

- Large volumes of textual data

Natural language processing helps computers communicate with humans in their own language and scales other language-related tasks. For example, NLP makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which parts are important.

- Structuring a highly unstructured data source

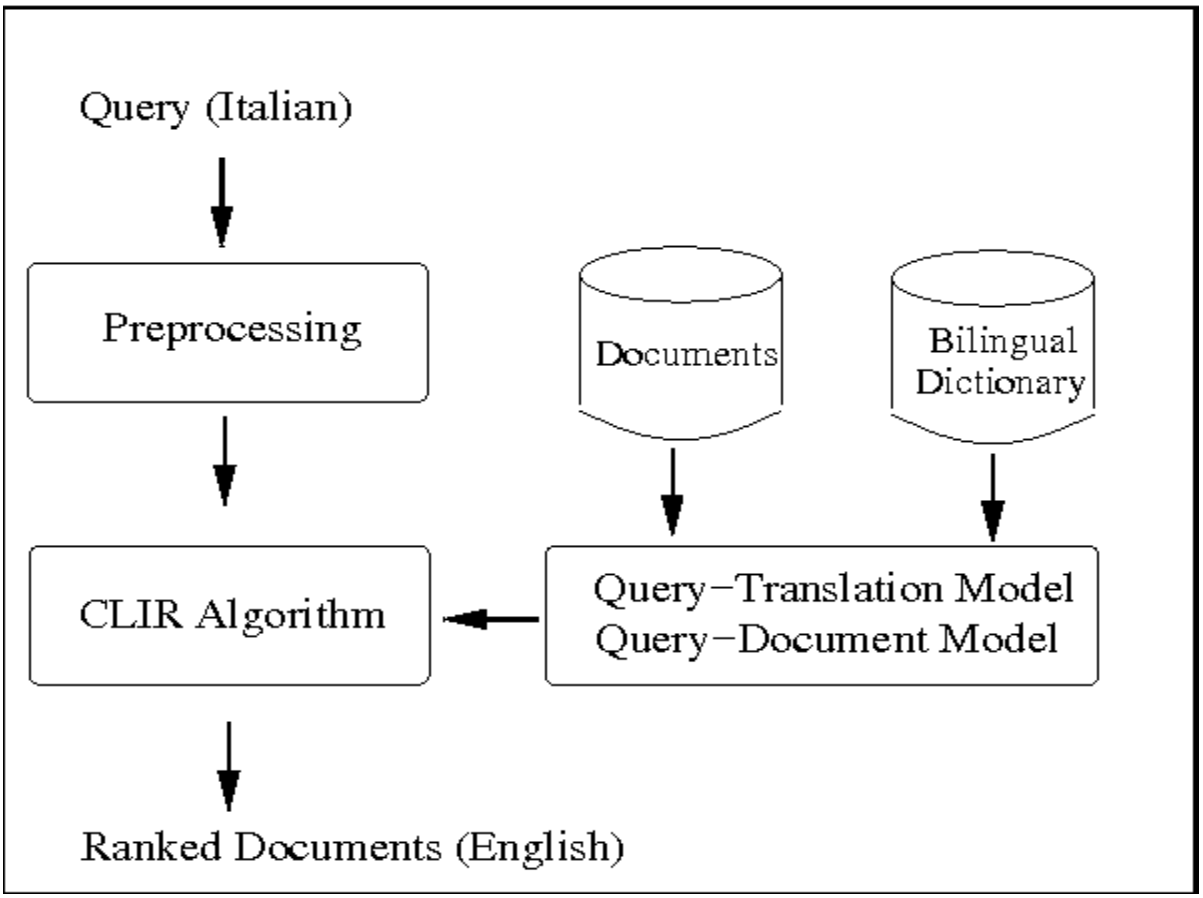
Human language is astoundingly complex and diverse. Not only are there hundreds of languages and dialects, but within each language is a unique set of grammar and syntax rules, terms and slang. When we write, we often misspell or abbreviate words, or omit punctuation. When we speak, we have regional accents, and we mumble, stutter and borrow terms from other languages. While supervised and unsupervised learning, and specifically deep learning, are now widely used for modeling human language, there's also a need for syntactic and semantic understanding and domain expertise that are not necessarily present in these machine learning approaches. NLP is important because it helps resolve ambiguity in language and adds useful numeric structure to the data for many downstream applications, such as speech recognition or text analytics.

26. What are the procedures of learning in NLP?

The procedures are given below:

Step 1: Sentence Segmentation
Step 2: Word Tokenization
Step 3: Predicting Parts of Speech for Each Token
Step 4: Text Lemmatization
Step 5: Identifying Stop Words
Step 6: Dependency Parsing
Step 6b: Finding Noun Phrases
Step 7: Named Entity Recognition (NER)
Step 8: Reference Resolution

27.Diagrammatically show a model of cross language information retrieval.



Chapter 1 – 9 (Questions)

Chapter 1

1. What are the different approaches in defining AI?
2. Suppose you design a machine to pass the Turing test. What are the capabilities such a machine must have?

Chapter 2

1. What is sub assumption agent?
2. What is learning agent?
3. Compare and contrast different agent architectures?
4. Find out about the Mars rover.
   a) What are the percepts for this agent?
b) Characterize the operating environment?
c) What are the actions the agent can take?
d) How can one evaluate the performance of the agent?
e) What sort of agent architecture do you think is most suitable for this agent?
5. Answer the same questions as above for an Internet shopping agent.
6. Define agent.
7. What is rational agent?
8. What is bounded rationality?
9. What is autonomous agent?
10. Describe the salient features of an agent.

Problem formulation

Means choosing a relevant set of states to consider and a feasible set of operators for moving from one state to another.

Search is the process of imagining sequences of operators applied to the initial state and checking which sequence reaches a goal state.

Plan – path – sequence of action

Action – change the state of the agent

Chapter 3

You have three jugs measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. You need to measure out exactly one gallon.

Initial state: All three jugs are empty
Goal test: Some jug contains exactly one gallon

How you can think of it as state-space problem
Give the initial state, goal test, successor function, and cost function for each of the following.

Choose a formulation (state-space formulation) that is precise enough to be implemented.

1.You have to colour a planar map using only four colours, in such a way that no two adjacent regions have the same colour.

2. In the traveling salesperson problem (TSP) there is a map involving N cities some of which are connected by roads. The aim is to find the shortest tour that starts fromk a city visits all the cities exactly once and comes back to the starting city.

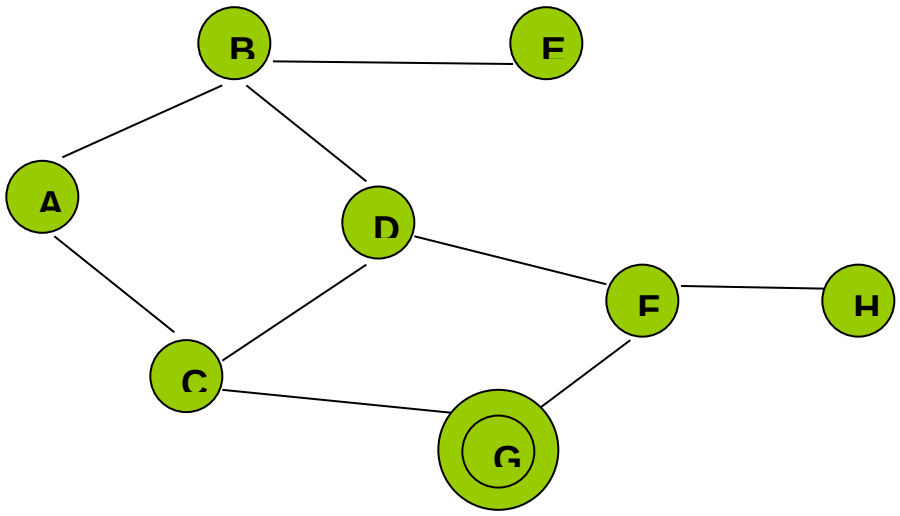
3. Missionaries & Cannibals problem: 3 missionaries & 3 cannibals are on one side of the river. 1 boat carries 2. Missionaries must never be outnumbered by cannibals. Give a plan for all to cross the river.

What is Hamiltonian path?

4. Given a full 5-gallon jug and an empty 2-gallon jug, the goal is to fill the 2-gallon jug with exactly one gallon of water. You may use the following state space formulation.

- State (x,y), where x is the number of gallons of water in the 5-gallon jug and y is number of gallon in the 2-gallon jug
- Initial state = (5,0)
- Goal state = (“, 1), where “ means any amount. Create the search tree. Discuss which search strategy is appropriate for this problem.

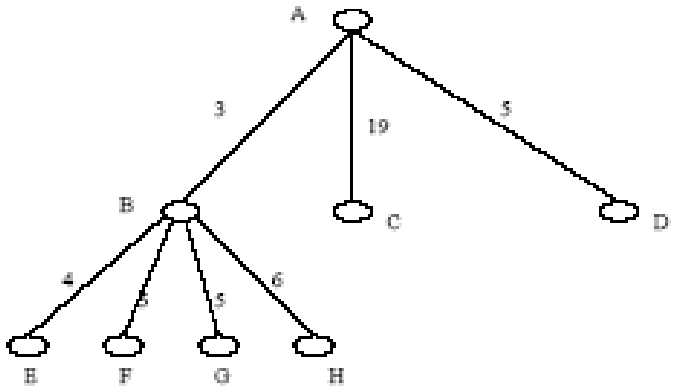
5. Consider the following graph. Starting from state A execute DFS. The goal node is G. Show the order in which the nodes are expanded. Assume that the alphabetically smaller nodes is expanded first to break ties.



Run iterative deepening search on the same graph.

6. The following figure shows a partially expanded search tree. Each arc is

labelled with the cost of the corresponding operator, and the nodes are labelled by alphabets (A-H). Assume node B has been fully expanded as shown and consider D as the goal state.



For each of the following search strategies

(i) breadth-first

(ii) depth-first

(iii) iterative deepening search

briefly describe the strategy and show which node (in the figure above) will be expanded next in each case. A conclusion on the performance of the three search strategies for this specific problem will follow this.

Chapter -4

1. Suppose you have the following search space:

a) Draw the state space of this problem.

b) Assume that the initial state is A and the goal state is G. Show how each of the following search strategies would create a search tree to find a path from the initial state to the goal state.

Uniform cost

Greedy search

A* search

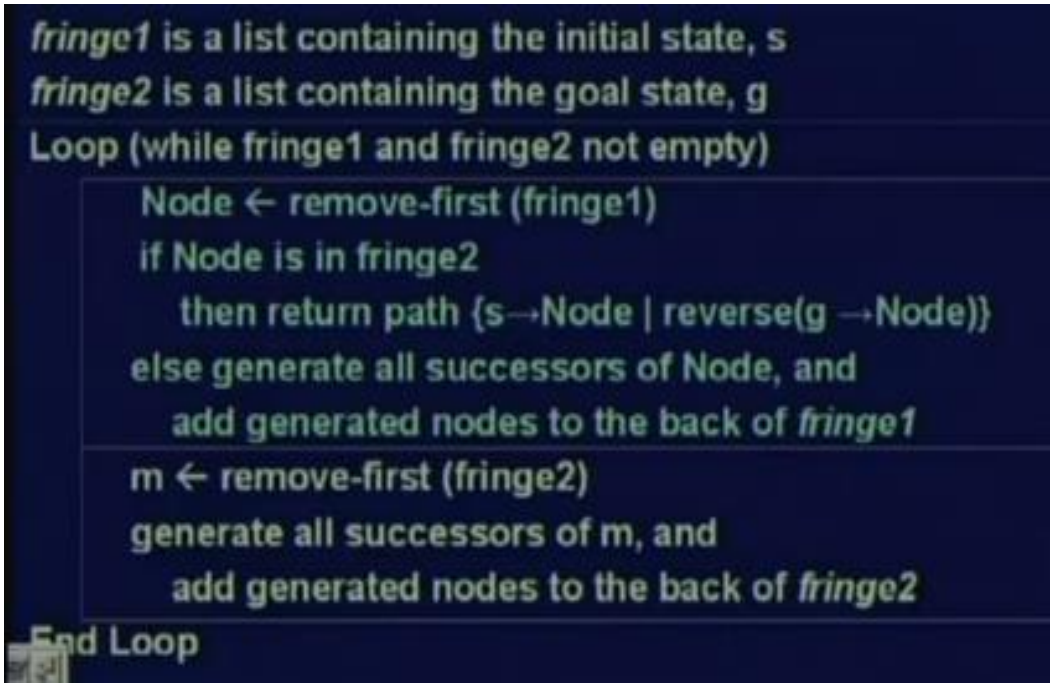
c) At each step of the search algorithm, show which node is being expanded, and the content of fringe. Also report the eventual algorithm, and the solution cost.

State	Next	Cost
A	B	4
A	C	1
B	D	3
B	E	8
C	C	0
C	D	2
C	F	6

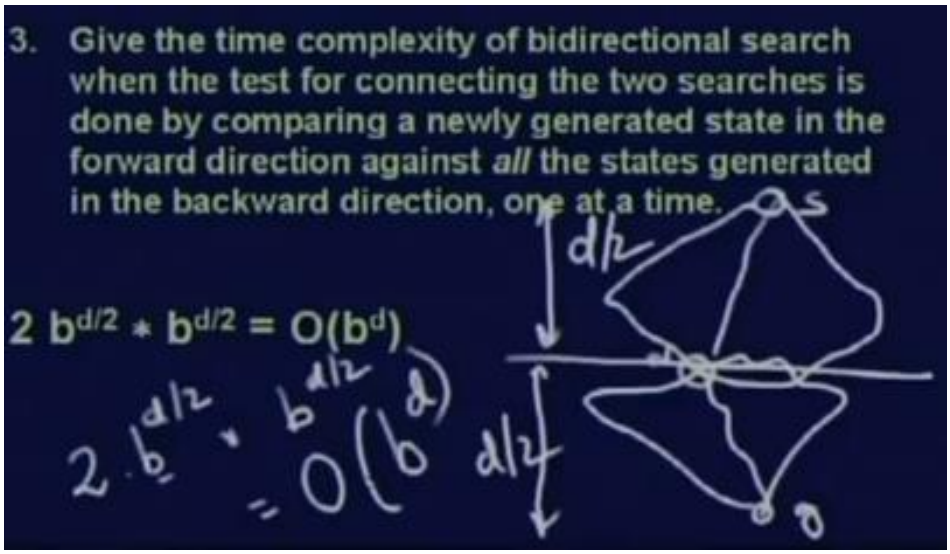
D	C	2
D	E	4
E	G	2
F	G	8

State	h
A	8
B	8
C	6
D	5
E	1
F	4
G	0

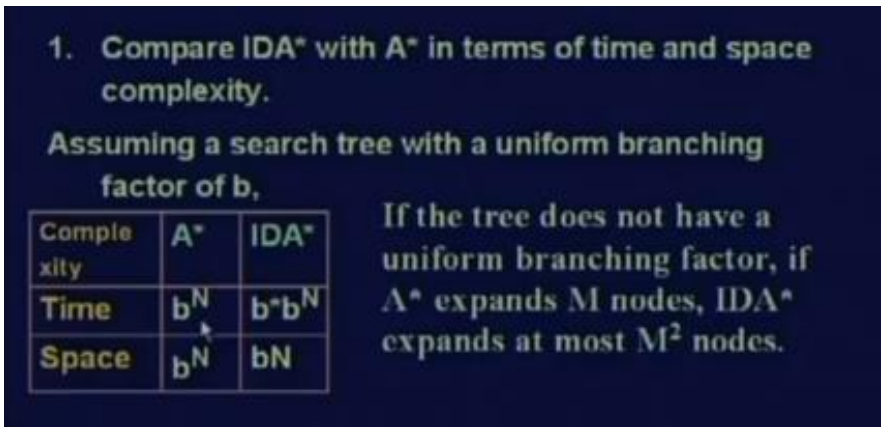
2. Write out the algorithm for bidirectional search, using pseudo-code. Assume that each search is a breadth-first search, and that the forward and backward searches alternate expanding one node at a time.



3.Give the time complexity of bidirectional search when the test for connecting the two searches is done by comparing a newly generated state in the forward direction against all the states generated in the backward direction, one at a time.



4. Apply IDA* on a given search problem
5. Compare IDA* with A* in terms of time and space complexity.



6. Is hill climbing guaranteed to find a solution to the n-queens problem?

7. Is simulated annealing guaranteed to find the optimum solution of an optimization problem like TSP?
8. What is iterative improvement algorithm? Do you consider hill-climbing is one of the classes of such algorithm? If yes, justify your answer.
9. Prove the optimality of A*

Chapter -5

Game Playing

Minimax (sometimes **minmax**) is a decision rule used in decision theory, game theory, statistics and philosophy for *mini*mizing the possible loss for a worst case (maximum loss) scenario. Alternatively, it can be thought of as maximizing the minimum gain (**maximin**). Originally formulated for two-player zero-sum game theory, covering both the cases where players take alternate moves and those where they make simultaneous moves, it has also been extended to more complex games and to general decision making in the presence of uncertainty.

What is zero-sum game?

Write the on the performance of alpha-beta pruning algorithm.

Discuss when best case and worst case occur

Discuss Quiscence, horizontal effect, transposition Table, special purpose hardware

M, P, Nim

There are p piles of m matches

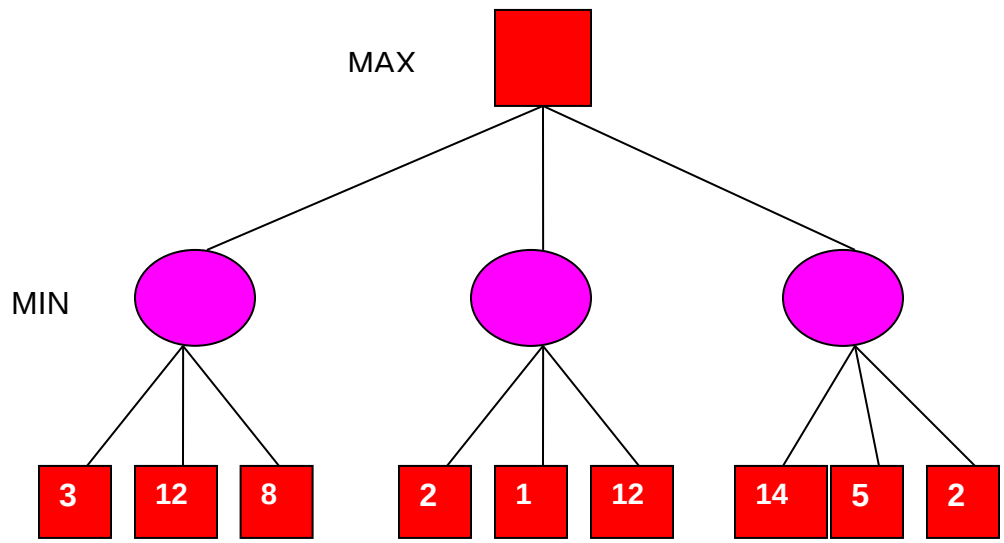
To move a player removes any number of matches from exactly one pile

The winning player is the one who removes the final match

1. Consider the game of 2,2 Nim
2. Draw the game tree
3. Mark on this tree the values at the terminal nodes (+1 if it is a win for MAX, -1 if it is a win for MIN)
4. Apply minimax on the game tree and mark on the tree the backed up value on the root node. Use these values to find the best starting move.
5. Can MAX force a win in this game?
6. Apply alpha beta pruning on this game tree

You are given this game tree. You are going to apply minimax with alpha-beta pruning on this game tree and identify the best possible move for MAX

Apply alpha-beta pruning on the following game tree.



The tree will be drawn during class.
Suppose the game tree for a particular problem has a branching factor of b. If you do a p-ply look ahead and then apply mini-max on this game tree, how many nodes will you expand per move?
Consider mini-max search on a game tree with p ply search, and a mini-max search on the same game tree with a q ply search. If q>p, which of these is guaranteed to find a better strategy for the MAX player?

Uncertainty

1.Taking account of the example below, explain the concept of uncertainty.
:" The doorbell rang at 12'0 clock in the midnight.

- Was someone there at the door?
- Did Karim wakeup?"

Explain why we can't 100% certain of ringing doorbell because of someone at the door and Karim wakeup because of ringing of doorbell.

2. What is abductive reasoning?
3. What is deductive reasoning? Explain why it is not an example of sound reasoning and hence considered as the source of uncertainty.
4. What is inductive reasoning? Explain why it is not an example of sound reasoning and hence considered as the source of uncertainty.
4. Discuss different sources of uncertainty?
5. Explain propagation of uncertainty.
6. What is default reasoning? Elaborate it with an example.
7. How uncertainty occurs due to default reasoning, elaborate with an example.
8. Show how uncertainty arises from conflicting information.
9. Explain why Bayes' theorem forms the basis of probabilistic reasoning.
10. Differentiate between union probability of two independent events and compound probability of two independent events.
11. What is the difference between dependent event and independent event? Elaborate your answer with suitable example.
12. What is conditional probability?
13. What are the basic axioms of probability?
14. What is causality? Can you demonstrate causality between dependent event or independent event and how?
15. Explain why we need the extension of Baye's theorem. What is extended Baye's theorem?
16. What is the problem with probability?
17. What is Bayesian approach?
18. What is d-seperation? Explain its rule in reducing computation in Bayesian Belief Network.
19. Write the steps for constructing BN.
20. Why we consider polytree in BN?
21. What is conditional independence?
22. Write the different types of inference procedures in BN
23. Diagrammetically compare and contrast inference procedures in BN

Answers : Chapter 1 – 9

Chapter 1

1. What are the different approaches in defining AI?

Answer :

Acting humanly

When a computer acts like a human, it best reflects the Turing test, in which the computer succeeds when differentiation between the computer and a human isn't possible. We see it employed for technologies such as natural language processing, knowledge representation, automated reasoning, and machine.

The Total Turing Test does include physical contact in the form of perceptual ability interrogation, which means that the computer must also employ both computer vision and robotics to succeed. Modern techniques include the idea of achieving the goal rather than mimicking humans completely. For example, the Wright Brothers didn't succeed in creating an airplane by precisely copying the flight of birds; rather, the birds provided ideas that led to aerodynamics that eventually led to human flight. The goal is to fly. Both birds and humans achieve this goal, but they use different approaches.

Thinking humanly

When a computer thinks as a human, it performs tasks that require intelligence from a human to succeed, such as driving a car. There are some methods of determining how humans think, which the cognitive modeling approach defines. This model relies on three techniques:

- **Introspection:** Detecting and documenting the techniques used to achieve goals by monitoring one's own thought processes.
- **Psychological testing:** Observing a person's behavior and adding it to a database of similar behaviors from other persons given a similar set of circumstances, goals, resources, and environmental conditions (among other things).
- **Brain imaging:** Monitoring brain activity directly through various mechanical means, such as Computerized Axial Tomography (CAT), Positron Emission Tomography (PET), Magnetic Resonance Imaging (MRI), and Magnetoencephalography (MEG).

After creating a model, a program is writtenthat simulates the model. The amount of variability among human thought processes and the difficulty of accurately representing these thought processes as part of a program, the results are experimental at best.

Thinking rationally

Studying how humans think using some standard enables the creation of guidelines that describe typical human behaviors. A person is considered rational when following these behaviors within certain levels of deviation. A computer that thinks rationally relies on the recorded behaviors to create a guide as to how to interact with an environment based on the data at hand. The goal of this approach is to solve problems logically, when possible. In many cases, this approach would

enable the creation of a baseline technique for solving a problem, which would then be modified to actually solve the problem.

Acting rationally
Studying how humans act in given situations under specific constraints enables to determine which techniques are both efficient and effective. A computer that acts rationally relies on the recorded actions to interact with an environment based on conditions, environmental factors, and existing data. As with rational thought, rational acts depend on a solution in principle, which may not prove useful in practice. However, rational acts do provide a baseline upon which a computer can begin negotiating the successful completion of a goal.

2. Suppose you design a machine to pass the Turing test. What are the capabilities such a machine must have?

Answer :

Here are the four features that would be required for a machine to pass the Turing test.

- **Natural Language Processing -** To be able to communicate in a commonly agreed language.
- **Knowledge representation -** To store the known information.
- **Automated Reasoning -** To use the stored information to answer questions and to draw new conclusions.
- **Machine Learning –** To adapt to new circumstances and to detect and extrapolate patterns.

Chapter 2

1. What is sub assumption agent?

Answer :

An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.
The sub assumption agent is the part of an agent that assumes the obligation to promote or close transactions for the principal, receiving an agreed commission.

2. What is learning agent?

Answer :

A **learning agent** is a tool in AI that is capable of learning from its experiences. It starts with some basic knowledge and is then able to act and adapt **autonomously**, through learning, to improve its own performance. Unlike intelligent agents that act on information provided by a programmer, learning agents are able to perform tasks, analyze performance and look for new ways to improve on those tasks.

3. Compare and contrast different agent architectures?

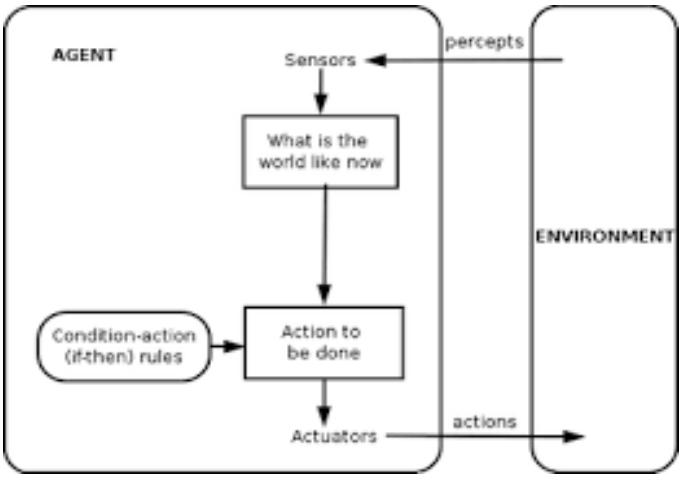
Answer :

Simple reflex agent	**Model based reflex agent**	**Goal based agent**	**Utility based agent**
Simple reflex agents act only on the basis of the current percept, ignoring the rest of the percept history.	Its current state is stored inside the agent maintaining some kind of structure which describes the part of the world which cannot be seen.	Goal based agents further expand on the capabilities of the model based agents, by using "goal" information.	It is possible to define a measure of how desirable a particular state is

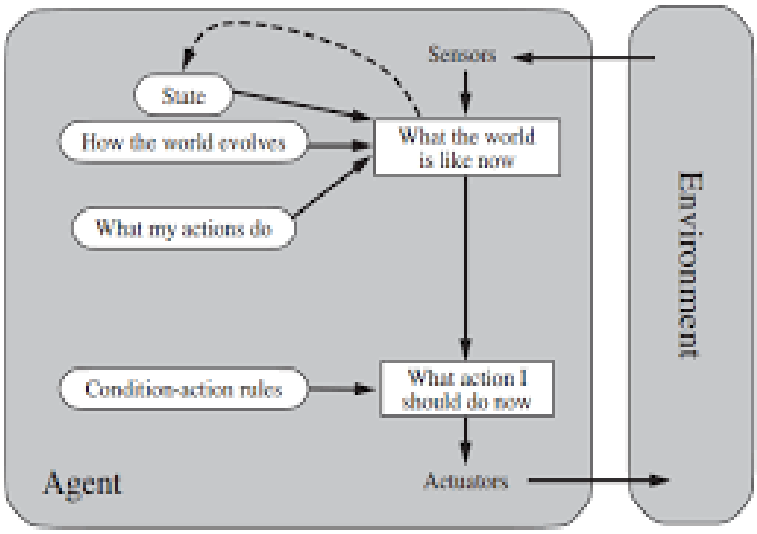
The agent function is based on the condition-action rule: if condition then action.	A model based agent can handle a partially observable environment.	This allows the agent a way to choose among multiple possibilities, selecting the one which reaches a goal state.	The term utility, can be used to describe how “happy” the agent is.
Succeeds when the environment is fully observable.	It keeps track of the current state of the world using an internal model (Environment).	It considers the future action. It uses goal information to select between possible actions in the current state.	This measure can be obtained through the use of utility function which maps a state to a measure of the utility of the state.
Some reflex agents can also contain information on their current state which allows them to disregard condition.	This knowledge about “how the world evolves” is called a model of the world, hence the name “model based agent”.	Goal based agents distinguish between goal states and non-goal states. Typically investigated in search and planning research.	A more general performance measure should allow a comparison of different world states according to exactly how happy they would make the agent.

They are stateless devices which do not have memory of past world states	Have internal states which is used to keep track of past states of the world	More flexible since knowledge is represented explicitly and can be manipulated	Allows decisions comparing choice between conflicting goals, and choice between likelihood of success and importance of goal

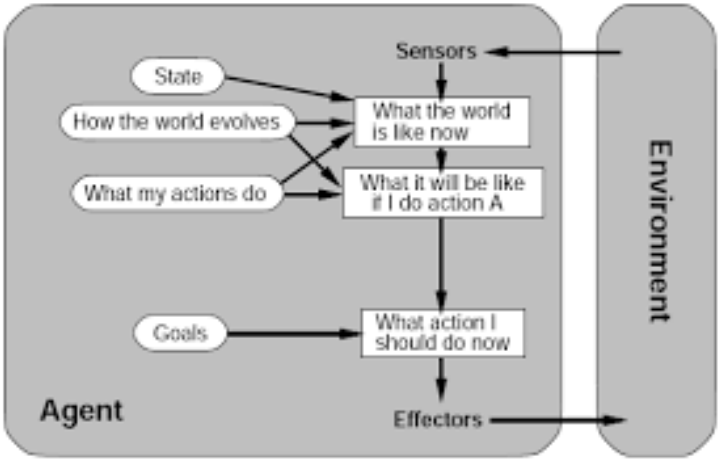
Simple reflex agent



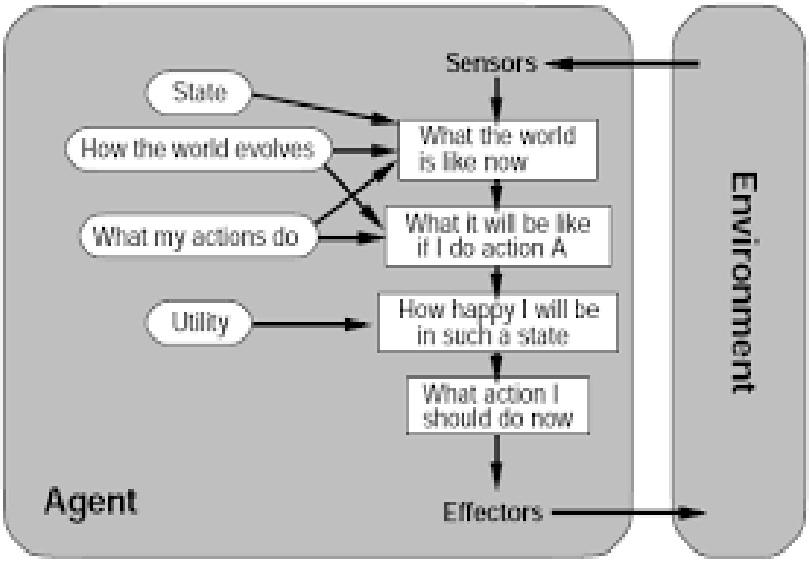
Model based reflex agent



Goal based agent



Utility based agent



4)Find out about the Mars rover.

a) What are the percepts for this agent ?

Ans:

Spirit's sensor include
* panoramic and microscopic cameras,
* a radio receiver,
* spectrometers for studying rock samples including an alpha particle x-ray spectrometer, M" ossbauer spectrometer, and miniature thermal emission spectrometer.

b) Characterize the operating environment

Ans:

The environment (the Martian surface)

- partially observable
- non-deterministic
- sequential
- dynamic
- continuous and
- may be single-agent.

If a rover must cooperate with its mother ship or other rovers, or if mischievous Martians tamper with its progress, then the environment gains additional agents.

c) What are the actions the agent can take ?

Ans:

The rover Spirit has

- motor-driven wheels for locomotion
- along with a robotic arm to bring sensors close to interesting rocks and a
- rock abrasion tool (RAT) capable of efficiently drilling 45mm holes in hard volcanic rock.
- Spirit also has a radio transmitter for communication.

d) How can one evaluate the performance of the agent?

Ans:

Performance measure : **A Mars rover** may be tasked with

- maximizing the distance or variety of terrain it traverses,
- or with collecting as many samples as possible,
- or with finding life (for which it receives 1 point if it succeeds, and 0 points if it fails).

Criteria such as maximizing lifetime or minimizing power consumption are (at best) derived from more fundamental goals; e.g., if it crashes or runs out of power in the field, then it can't explore.

e) What sort of agent architecture do you think is most suitable for this agent ?

Ans:

Model-based reflex agent for low level navigation;
for route planning, experimentation etc. some combination of goal-based, and utility-based would be needed.

5) Internet shopping agent

Ans:

a)Performance measure:
price, quality, appropriateness, efficiency
b)Environment:
current and future WWW sites, vendors, shippers
c)Actuators:
display to user, follow URL, fill in form
d)Sensors:
HTML pages (text, graphics, scripts)
e) Goal-based agents would be needed.

6) Define an agent.

Ans:

An agent is anything that can be viewed as perceiving its environment through sensors and executing actions using actuators.

7)What is rational agent?

Ans:

An agent can be described as anything that is seen as

- perceiving its environment through sensors
- acting upon it through actuators

Rational Agents :

An agent which acts in a way that is expected to maximize to its performance measure, given the evidence provided by what it perceived and whatever built-in knowledge it has.The performance measure defines the criterion of success for an agent.

Such agents are also known as Rational Agents. The rationality of the agent is measured by its performance measure, the prior knowledge it has, the environment it can perceive and actions it can perform.

For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure,given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.

8)What is bounded rationality?

Ans:

Bounded rationality is the idea that in decision-making, rationality of individuals is limited by the information they have, the cognitive limitations of their minds, and the finite amount of time they have to make a decision.

In AI, the idea of bounded rationality floated around among several discussion groups interested in the general topic of resource-bounded rationality in the late 198Os, par- ticularly those at Rockwell (organized by Michael Fehling) and Stanford (organized by Michael Bratman). The term “bounded rationality” seems to have been originated by Eric Horvitz, who defined it informally as “the optimization of computational utility given a set of assumptions about expected problems and constraints on resources”.

9)What is autonomous agent?

Ans:

Autonomous agent research is a domain situated at the forefront of artificial intelligence. It was argued that genuine intelligence can emerge only in embodied, situated cognitive agents. It is a highly interdisciplinary research area, connecting results from theoreticalcognitive science, neural networks, evolutionary computation, neuroscience,and engineering. Besides its scientific importance, there are also important applications of this domain in the development of robots used in industry,defense and entertainment.

An autonomous agent is an intelligent agent operating on an owner's behalf but without any interference of that ownership entity.In artificial intelligence, an intelligent agent (IA) is an autonomous entity which acts, directing its activity towards achieving goals (i.e. it is an agent), upon an environment using observation through sensors and consequent actuators (i.e. it is intelligent).

10)Describe the salient features of an agent.

Ans:

Features of Agents:

1. Mobility: Intelligent agents engaged in eCommerce travel from computer to computer, across different system architecture and platforms and gather information until search parameters are exhausted.

2. Goal oriented: Intelligent agent has the ability to accept the user statement of goals and carry out the task delegated to it. It can move around from one machine to another and can act proactively in response to their environment, and can exhibit goal-directed behavior by taking the initiative.

3. Independent: Intelligent agents function on its own without human intervention and must have the ability to make decisions and to initiate action without direct human supervision. They communicate independently with repositories of information and other agents and accomplish the objectives and tasks on the behalf of the user.

4. Intelligent: Intelligent Agents are able to crawl for data more intelligently. They have intelligence to reason things out based on the existing knowledge of its user and environment and on past experiences. Intelligent agents follow preset rules to evaluate conditions in the external environment.

5. Reduces net traffic: Agents can communicate and cooperate with other agents quickly. This enables them to perform tasks, such as information searches quickly and efficiently and reduces network traffic.

6. Multiple tasks: An intelligent agent can perform multiple tasks simultaneously. It relieves the human user from monotonous clerical work.

Chapter 03

Question: You have three jugs measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. You need to measure out exactly one gallon.

Initial state: All three jugs are empty
Goal test: Some jug contains exactly one gallon

How you can think of it as state-space problem
Give the initial state, goal test, successor function, and cost function for each of the following.

Answer:
State space problem is a process to finding a goal with desired property by considering successive states of an instance are considered. Here, problems are modeled as a state space (a set of states) that a problem can be in. We can consider a problem to be state space problem if we can define the initial state, goal test, successor function and cost function of that problem.

For the given problem initial state, goal test, successor function and cost function can be considered as follows:

Initial state: All three jugs are empty
Goal test: Some jug contains exactly one gallon
successor function: Fill jug, Empty jug, Transfer to.
Cost function: Cost to fill and transfer water form jugs.

Choose a formulation (state-space formulation) that is precise enough to be implemented.

1.You have to color a planar map using only four colors, in such a way that no two adjacent regions have the same color.

Answer:
Initial state: No region colored
Goal test: I) All region colored
II) No two adjacent regions are colored same
successor function: Choose color, color region
Cost function: complexity of computation.

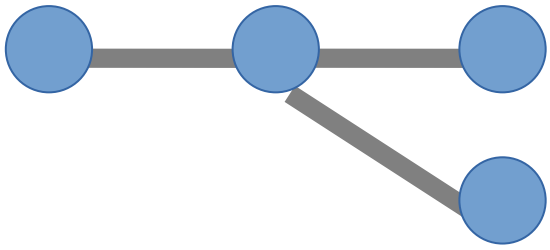
2. In the traveling salesperson problem (TSP) there is a map involving N cities some of which are connected by roads. The aim is to find the shortest tour that starts from a city visits all the cities exactly once and comes back to the starting city.

Answer:
Initial state: In a city, none other visited.
Goal test: In the starting city, all other visited exactly once.
successor function: visit to
Cost function: Routing cost.

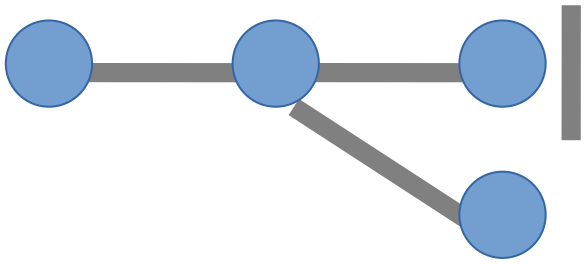
3. Missionaries & Cannibals problem: 3 missionaries & 3 cannibals are on one side of the river. 1 boat carries 2. Missionaries must never be outnumbered by cannibals. Give a plan for all to cross the river.

Answer:
Initial state: Missionaries and Cannibals are on one side of river.
Goal test: Missionaries and Cannibals are on the other side of river.
successor function: Take on boat.
Cost function: Number of boat crossing the river.

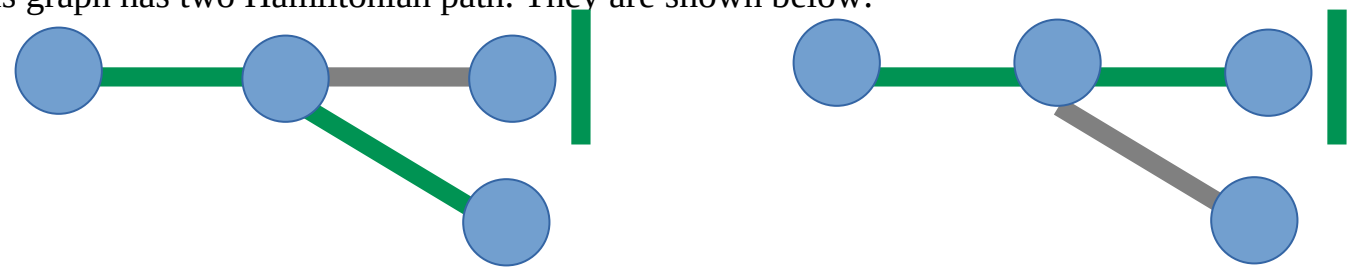
What is Hamiltonian path?
Answer: Hamiltonian Path is a path in a graph that visits each vertex exactly once. The problem is to check whether a graph contains a Hamiltonian path is NP-complete, so is the problem of finding all the Hamiltonian Paths in a graph.



This graph has no Hamiltonian path.



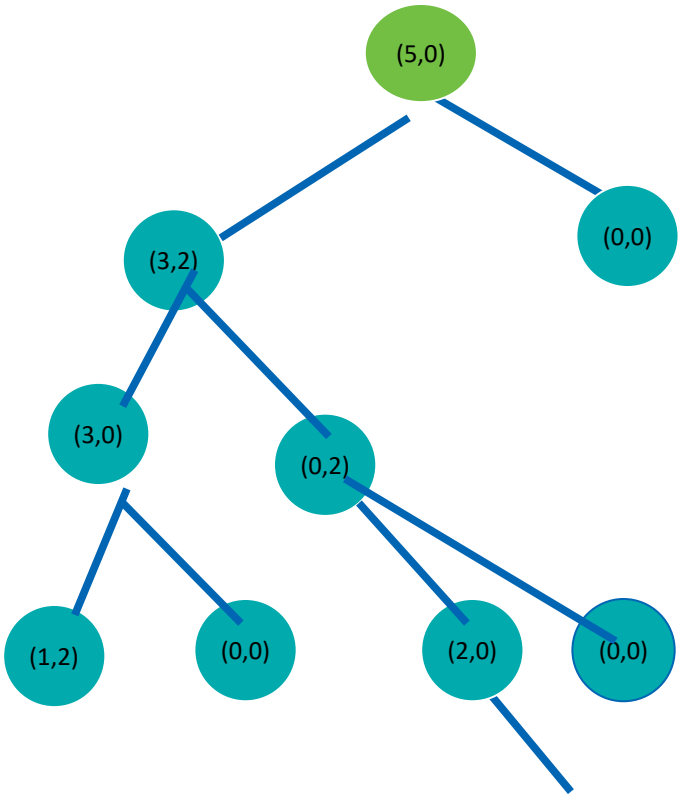
This graph has two Hamiltonian path. They are shown below:

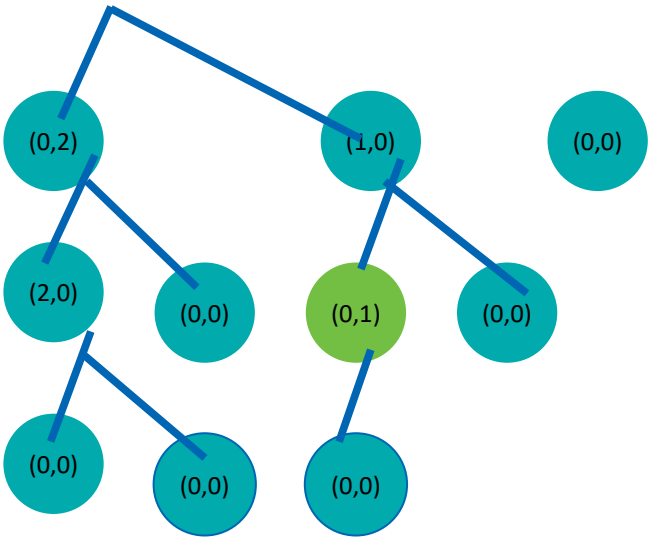


4. Given a full 5-gallon jug and an empty 2-gallon jug, the goal is to fill the 2-gallon jug with exactly one gallon of water. You may use the following state space formulation.

- **State (x,y), where x is the number of gallons of water in the 5-gallon jug and y is number of gallon in the 2-gallon jug**
- **Initial state = (5,0)**
- **Goal state = (“, 1), where “ means any amount. Create the search tree. Discuss which search strategy is appropriate for this problem.**

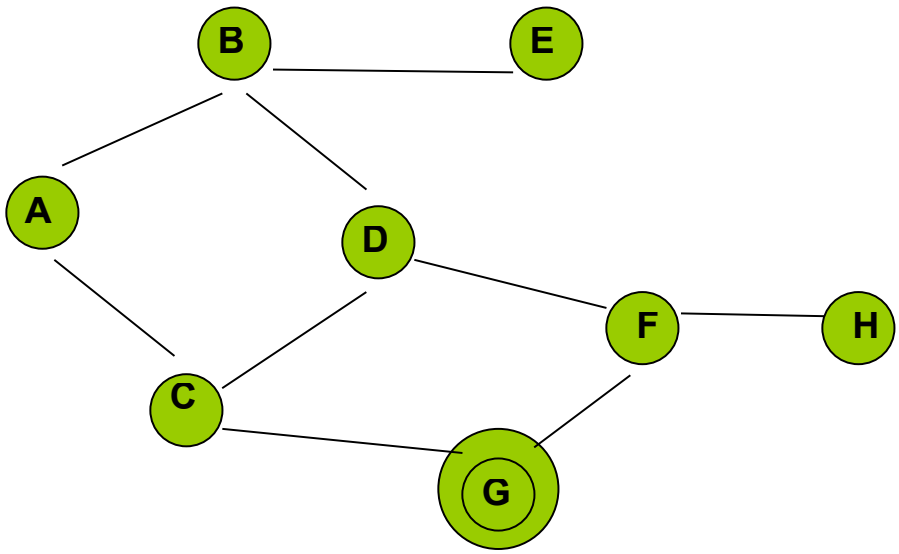
Answer:





As we can see, the solution lays in the deepest node, in the leaf of the tree. Besides, the graph has a very low depth. The highest depth is seven. For this reason depth-first search stagy will be appropriate here.

5. Consider the following graph. Starting from state A execute DFS. The goal node is G. Show the order in which the nodes are expanded. Assume that the alphabetically smaller nodes is expanded first to break ties.
Run iterative deepening search on the same graph.



Answer: The traversal of DFS in the graph will be as follows:

Expanded node	Open node	Closed node
-	A	-
A	B,C	A
B	D,E,C	A,B
D	F,E,C	A,B,D
C	F,E,G	A,B,D,C
G	F,E	A,B,D,C,G

So, the order of DFS traversal will be A,B,D,C,G.

By applying iterative deepening search we will get traversal as follows:

Limit	Order
0	A
1	A,B,C
2	A,B,D,E,C,G
3	A,B,D,C,G,F,E
4	A,B,D,C,G,F,H,E

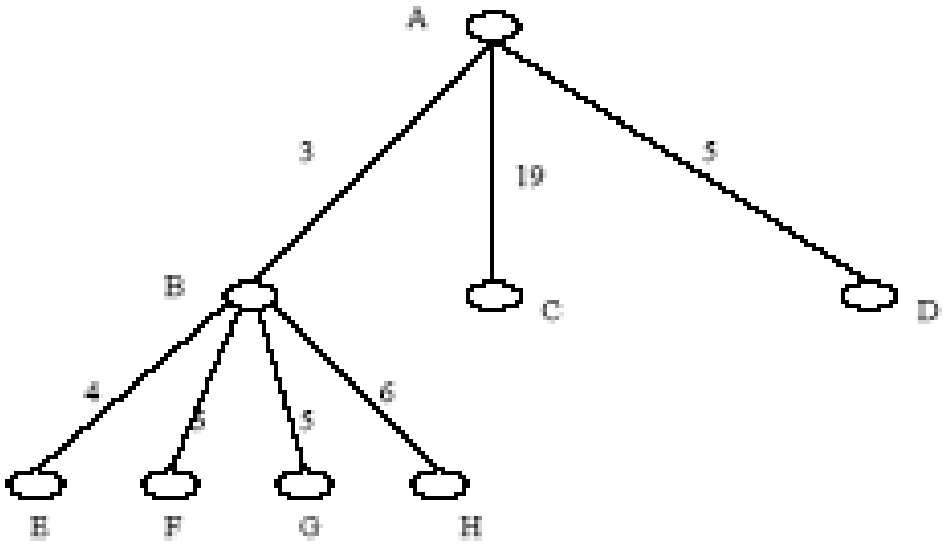
So, to travel all the nodes, we may set the depth limit to 4. But to reach in the goal node, 2 is enough.

6. The following figure shows a partially expanded search tree. Each arc is

labeled with the cost of the corresponding operator, and the nodes are labeled

by alphabets (A-H). Assume node B has been fully expanded as shown and

consider D as the goal state.



For each of the following search strategies

(i) breadth-first

(ii) depth-first

(iii) iterative deepening search

briefly describe the strategy and show which node (in the figure above) will be expanded next in each case. A conclusion on the performance of the three search strategies for this specific problem will follow this.

Answer: For breadth-first search, the nodes will be expanded as follows:

Expanded node	Open node	Closed node
-	A	-
A	B,C,D	A
B	C,D,E,F,G,H	A,B
C	D,E,F,G,H	A,B,C
D	E,F,G,H	A,B,C,D

So, the path is A,B,C,D.
The path cost will be (3+19+5)=27

For depth-first search, the nodes will be expanded as follows:

Expanded node	Open node	Closed node
-	A	-
A	B,C,D	A
B	E,F,G,H,C,D	A,B
E	F,G,H,C,D	A,B,E
F	G,H,C,D	A,B,E,F
G	H,C,D	A,B,E,F,G
H	C,D	A,B,E,F,G,H
C	D	A,B,E,F,G,H,C
D	-	A,B,E,F,G,H,C,D

So, the path is A,B,E,F,G,H,C,D.
The path cost will be (3+4+5+5+6+19+5)=49

For iterative deepening search, the nodes will be expanded as follows:

For limit=0

Expanded node	Open node	Closed node
-	A	-
A	-	A

For limit=1

Expanded node	Open node	Closed node
-	A	-
A	B,C,D	A
B	C,D	A,B

Expanded node	Open node	Closed node
C	D	A,B,C
D	-	A,B,C,D

Path cost=(3+19+5)=27

For limit=2

Expanded node	Open node	Closed node
-	A	-
A	B,C,D	A
B	E,F,G,H,C,D	A,B
E	F,G,H,C,D	A,B,E
F	G,H,C,D	A,B,E,F
G	H,C,D	A,B,E,F,G
H	C,D	A,B,E,F,G,H
C	D	A,B,E,F,G,H,C
D	-	A,B,E,F,G,H,C,D

Path cost = (3+4+5+5+6+19+5)=49

So, the goal node can be found when limit is greater than 0. However, the goal node can be found easily at depth limit=1. At limit=1 the iterative deepening search works as like as breadth-first search.

Chapter 04

1. Suppose you have the following search space:

a) Draw the state space of this problem.

b) Assume that the initial state is A and the goal state is G. Show how each of the following search strategies would create a search tree to find a path from the initial state to the goal state.

Uniform cost

Greedy search

A* search

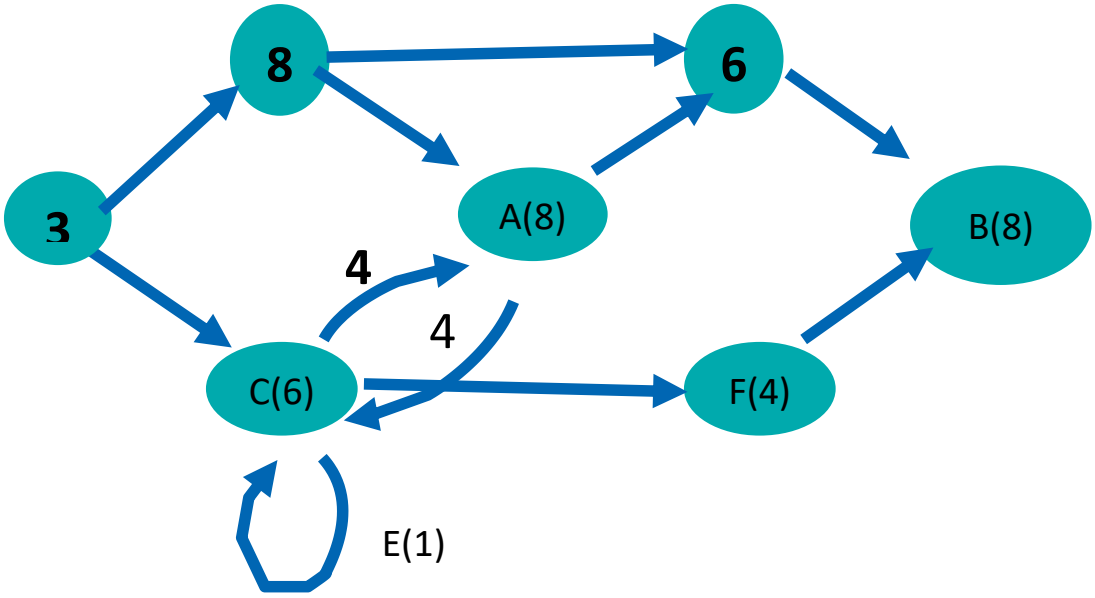
c) At each step of the search algorithm, show which node is being expanded, and the content of fringe. Also report the eventual algorithm, and the solution cost.

State	Next	Cost
A	B	4
A	C	1
B	D	3
B	E	8
C	C	0
C	D	2
C	F	6
D	C	2
D	E	4
E	G	2
F	G	8

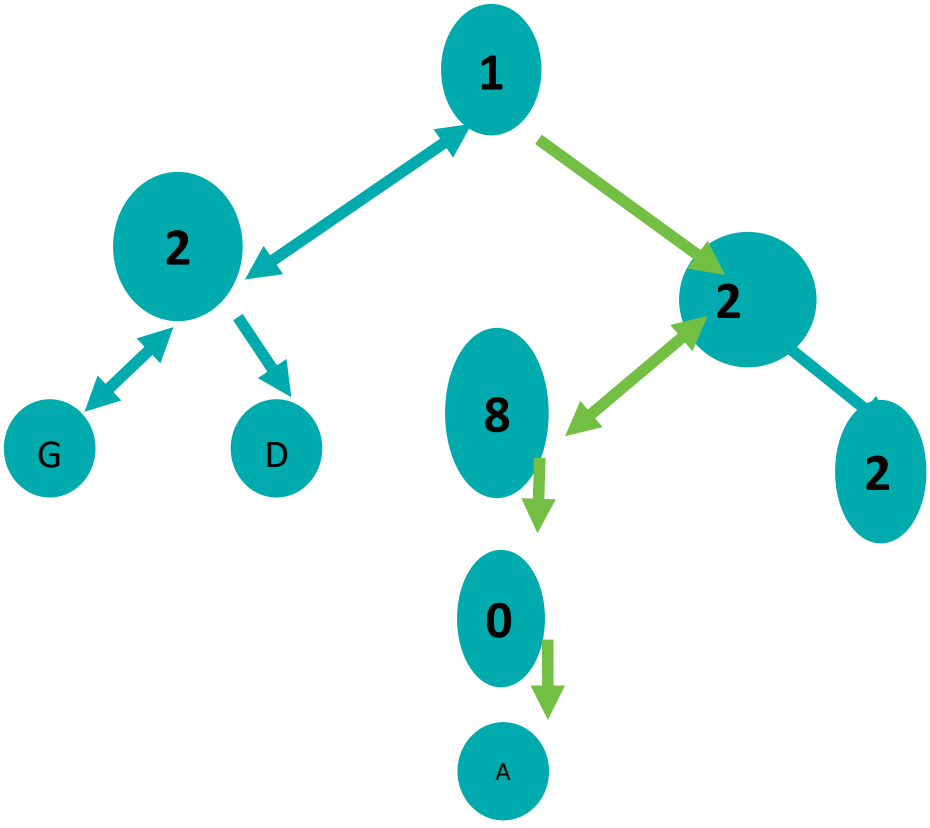
State	h
A	8
B	8
C	6
D	5
E	1
F	4
G	0

Answer:

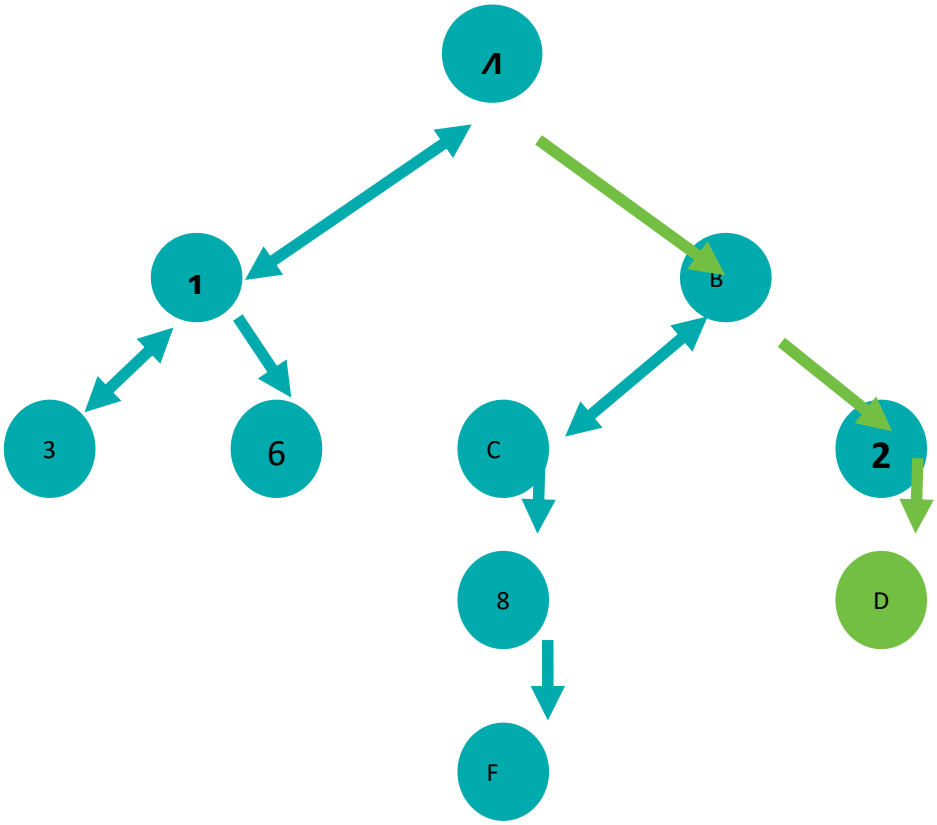
a) The state space is drawn below:



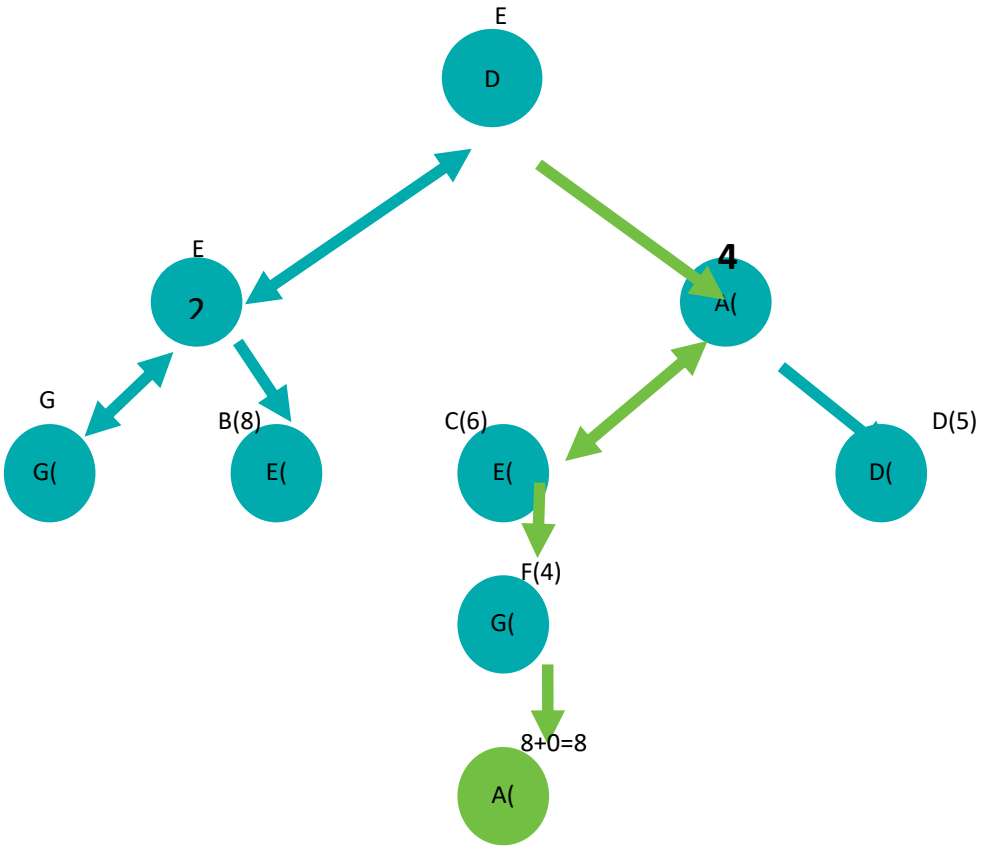
b) Search tree for Uniform cost search:



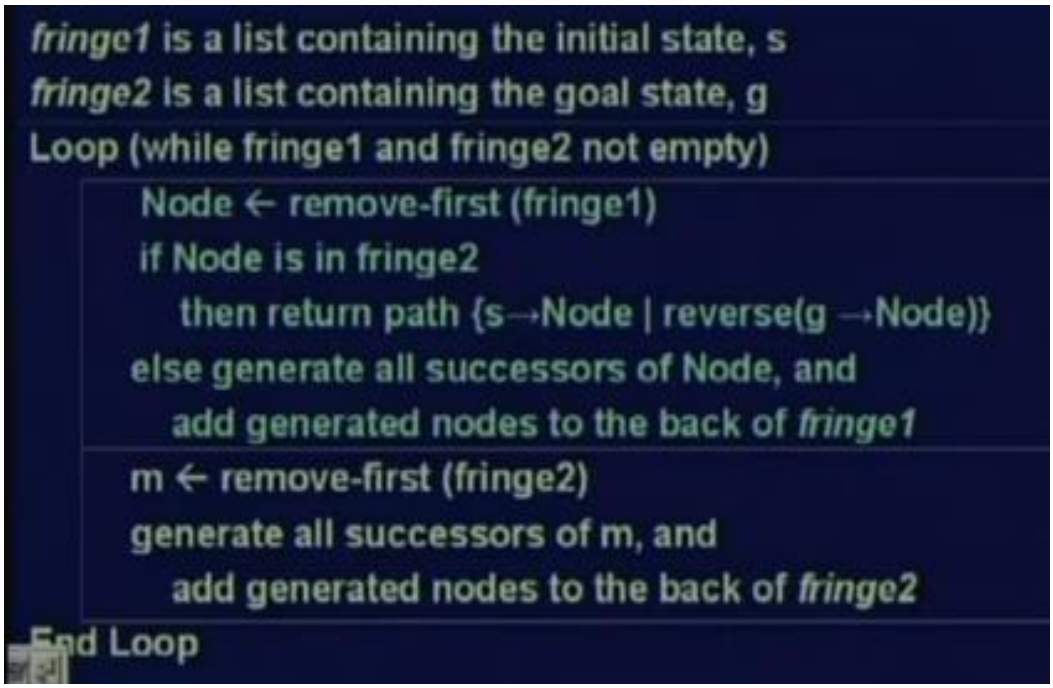
Search tree for greedy search:



Search tree for A* search:



2. Write out the algorithm for bidirectional search, using pseudo-code. Assume that each search is a breadth-first search, and that the forward and backward searches alternate expanding one node at a time.



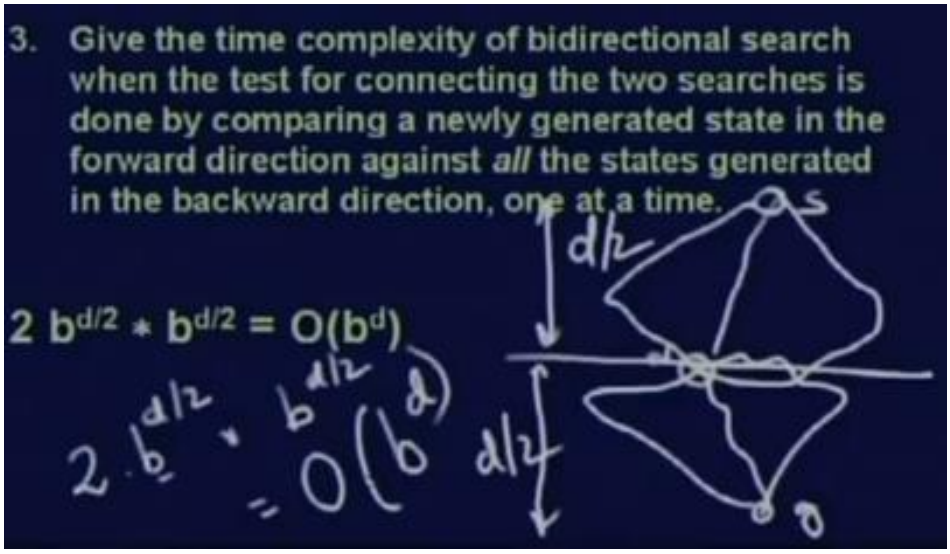
Answer: The desired algorithm can be as follows:

function BIDIRECTIONAL-SEARCH(problem) **returns** a solution, or failure
node1← a node with STATE=problem.INITIAL-STATE, PATH-COST=0,
node2← a node with STATE=problem.GOAL-STATE
if problem.GOAL-TEST(node.STATE) **then return** SOLUTION(node)
frontier1← a FIFO queue with node1 as the only element
frontier2← a FIFO queue with node2 as the only element
explored← an empty set
Loop do
if EMPTY(frontier1 AND frontier2) then return failure
node1← POP(frontier1)
node2←POP(frontier2)
add node1.STATE to explored
add node2.STATE to explored
for each action in problem.ACTIONS(node1.STATE and node2.STATE) **do**
child← CHILD-NODE(problem, node, action)
if child.STATE is not in explored or frontier **then**
frontier1←insert(child, frontier1)

else if (child.STATE is in frontier1 and frontier2) **return** SOLUTION(child)

END

3.Give the time complexity of bidirectional search when the test for connecting the two searches is done by comparing a newly generated state in the forward direction against all the states generated in the backward direction, one at a time.



Answer: For the total depth of the path d, the forward and backward direction search will have d/2 depth. If every state has b successors (imagining a uniform tree) the forward and backward search each will have the following comparison:

$b+b^2+b^3+\ldots\ldots+b^{d/2}=O(b^{d/2})$

So, a newly generated state will have a time complexity of $O(b^{d/2})$.
Therefore, the total time complexity will be $O(b^{d/2}+b^{d/2})$ or $O(b^{d/2})$

5. Compare IDA* with A* in terms of time and space complexity.

Answer: Assuming a uniform search tree with a branching factor of B and depth N, we get the following time and space complexity:

Complexity	**A***	**IDA***
Time	B^N	B^{N+1}
Space	B^N	BN

6. Is hill climbing guaranteed to find a solution to the n-queens problem?

Answer: No, Hill climbing search doesn't always guarantee to find a solution to the n-queens problem. The steepest ascent version of Hill climbing search gets stack 86% of the time, solving only 14% of problem instances. It works quickly, taking just 4 steps on average when it succeeds and 3 when it gets stack. Although it is not a bad performance; measuring 8^8 or almost 17 million states, it doesn't granteee the solution at all. Restricting the sideways move raises the percentage of problem instances solved by hill climbing from 14% to 94%. But in this case the algorithm takes 21 steps in average for each successful instance and 64 for each failure.

The only version of hill climbing that guarantees the solution of n-queens problem is **Random-restart hill climbing**. It is actually a series of hill climbing searches. If the probability of success is p for all the hill climbing searches, then it takes restart at most 1/p times.

7. Is simulated annealing guaranteed to find the optimum solution of an optimization problem like TSP?

Answer: Yes, Simulated annealing guarantees to find the optimum solution of an optimization problem. Simulated annealing is an algorithm that combines hill climbing with random walk. Instead of picking the best move like hill climbing, it takes a random move. If the move improves the situation, it accepts the move. Otherwise the algorithm accepts the move with some probability less than 1. The probability decreases exponentially with the badness of the move. If the badness lowers the probability slowly, the algorithm likely to find the global optimum with probability approaching 1.

8. What is iterative improvement algorithm? Do you consider hill-climbing is one of the classes of such algorithm? If yes, justify your answer.

Answer: Iterative improvement algorithm is a class of algorithms that approaches to a solution of a problem in iterative approach. In the cases where the path is not important, rather the solution is, iterative method is a good choice.

Yes, Hill climbing is one of the iterative improvement algorithm. Hill climbing starts with a very small amount of problem specification and gradually increase to the main goal. It is simply a loop that continually moves in the direction of increasing value. It doesn't forms any search tree. It only works by the records of the states and value of the objective function.

For example, in n-queen problem, hill climbing starts with 1 queen and gradually approaches towards the ultimate problem. So, we can say hill climbing to be a iterative improvement algorithm.

9. Prove the optimality of A*

Answer: A* has the following properties:
I) The tree search version of A* is optimal if h(n) is admissible.
II) The graph-search version is optimal if h(n) monotonic.
The second of these two claims are proved below:
1st Step: (Establishing if h(n) is consistent, then the values of f(n) along any path are nondecreasing.)
The proof follows directly form the definition of consistency.
Let n' is a successor of n; the g(n')= g(n) + c(n,a,n') for some action a, and we have

$$f(n')=g(n')+ h(n')= g(n')= g(n) + c(n,a,n')+ h(n') \geq g(n) + h(n) = f(n)$$

2nd Step: (proving whenever A* selects a node for expansion, the optimal path to that node has been found.)
Were this not the case, there would havve to be another forntier node n' on the optimal path form start node to n, by the graph separation property; because f is nondecreasing along any path, n' would have lower f-cost than n and would have beeen selected first.

From the two preceeding observations, it follows that sequnce of nodes expanded by A* using GRAPH-SEARCH is in nondecreasing order of f(n). Hence, the first goal node selectd for expansion must be an optimal solution because f is the true cost for goal nodes and all later goal nodes will be at least as expensive. Hence, the optimality of A* is proved.

Chapter 05

Game Playing

1. What is zero sum game?

Ans: Zero-sum is a situation in game theory in which one person's gain is equivalent to another's loss, so the net change in wealth or benefit is zero. A zero-sum game may have as few as two players, or millions of participants.

Zero-sum games are found in game theory, but are less common than non-zero sum games. Poker and gambling are popular examples of zero-sum games since the sum of the amounts won by some players equals the combined losses of the others. Games like chess and tennis, where there is one winner and one loser, are also zero-sum games.

2. Write on the performance of alpha-beta pruning algorithm.

Ans: If we apply alpha-beta pruning to a standard minimax algorithm, it returns the same move as the standard one, but it removes (prunes) all the nodes that are possibly not affecting the final decision.In fact, it is an updated version of minimax algorithm. In this method, the search is cut-offed by exploring less number of nodes. By exploring less number of nodes, the performance increases and saves time ,too. When a best path is found, the other nodes are ot explored, rather they are pruned.

The benefit of alpha–beta pruning lies in the fact that branches of the search tree can be eliminated. This way, the search time can be limited to the 'more promising' subtree, and a deeper search can be performed in the same time. Like its predecessor, it belongs to the branch and bound class of algorithms. The optimization reduces the effective depth to slightly more than half that of simple minimax if the nodes are evaluated in an optimal or near optimal order (best choice for side on move ordered first at each node).

3. Discuss when best case and worst case occur

Ans: The complexity of minimax algorithm is O(b^d), where b is branching factor and d is the depth or ply. When b or d is small, it is easy for minimax algorithm to traverse and gives us best answer. Example: In Tic-Tac-Toe, Nim games. But, in chess, there are 35 moves possible and game length = 100. So, search space = (35)^100 is difficult to traverse, exact solution completely infeasible. Then, it is the worst case.

4. **Discuss Quiescence, horizontal effect, transposition Table, special purpose Hardware.**

Ans: **Horizon Effect:** A game search that stops at a fixed depth has a problem: If a tactical action is in progress at the end of the variation, the evaluation function may give unreliable answers. In a chess search, for example, the white's last move may be queen takes knight, and the

evaluation function will report that white is up a knight. If the search looked one move deeper, it would see that black has the reply pawn takes queen, and realize that black is winning.

One name for this problem is the *horizon effect.*

Quiescence Search: A quiescence search is an additional search, starting at the leaf nodes of the main search, which tries to solve this problem of Horizon effect. In chess, quiescence searches usually include all capture moves, so that tactical exchanges don't mess up the evaluation. In principle, quiescence searches should include any move which may destabilize the evaluation function—if there is such a move, the position is not quiescent.

Not all search algorithms require a quiescence search. A smart selective search might, in effect, do its own quiescence searching as a natural part of its operation. But simpleminded depth-limited search algorithms, the kind that provide strong play in many games, usually need a quiescence search to avoid errors (though of course it depends on the game). It's logical to try to get the best of both worlds by combining a dumb search with a smart selective search for quiescence.

Transposition Table: Game-playing programs work by analyzing millions of positions that could arise in the next few moves of the game. Typically, these programs employ strategies resembling depth-first search, which means that they do not keep track of all the positions analyzed so far. In many games, it is possible to reach a given position in more than one way. These are called transpositions.[1] In chess, for example, the sequence of moves ***1.*** *d4 Nf6* ***2.*** *c4 g6* has 4 possible transpositions, since either player may swap their move order. In general, after *n* moves, an upper limit on the possible transpositions is $(n!)^2$. Although many of these are illegal move sequences, it is still likely that the program will end up analyzing the same position several times.

To avoid this problem, transposition tables are used. Such a table is a hash table of each of the positions analyzed so far up to a certain depth. On encountering a new position, the program checks the table to see whether the position has already been analyzed; this can be done quickly, in amortized constant time. If so, the table contains the value that was previously assigned to this position; this value is used directly. If not, the value is computed, and the new position is entered into the hash table.

Special purpose Hardware: An AI accelerator is a class of microprocessor[1] or computer system[2] designed as hardware acceleration for artificial intelligence applications, especially artificial neural networks, machine vision and machine learning. Typical applications include algorithms for robotics, internet of things and other data-intensive or sensor-driven tasks.[3] They are often manycore designs and generally focus on low-precision arithmetic, novel dataflow architectures or in-memory computing capability.[4] A number of vendor-specific terms exist for devices in this category, and it is an emerging technology without a dominant design. AI accelerators can be found in many devices such as smartphones, tablets, and computers all around the world. See the heading titled ¨Examples" for more examples.

Game of NIM

1) The game of 2,2 Nim.

   Let's,
   Max's move is denoted by rectangle shape,
   Min's move is denoted by oval shape

2) Game tree for 2,2 Nim

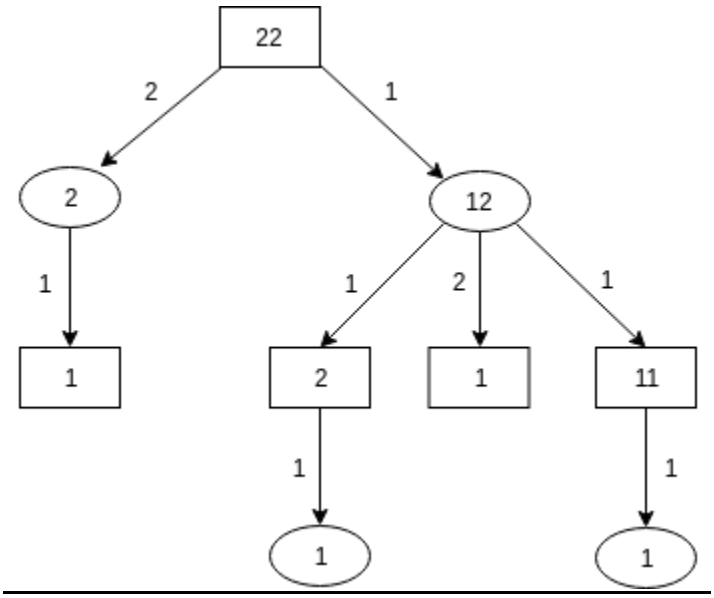


Figure: Game tree for 2,2 Nim game

3) Marking the terminal nodes:

   (+1) denotes a win for Max,
   (-1) denotes a win for Min

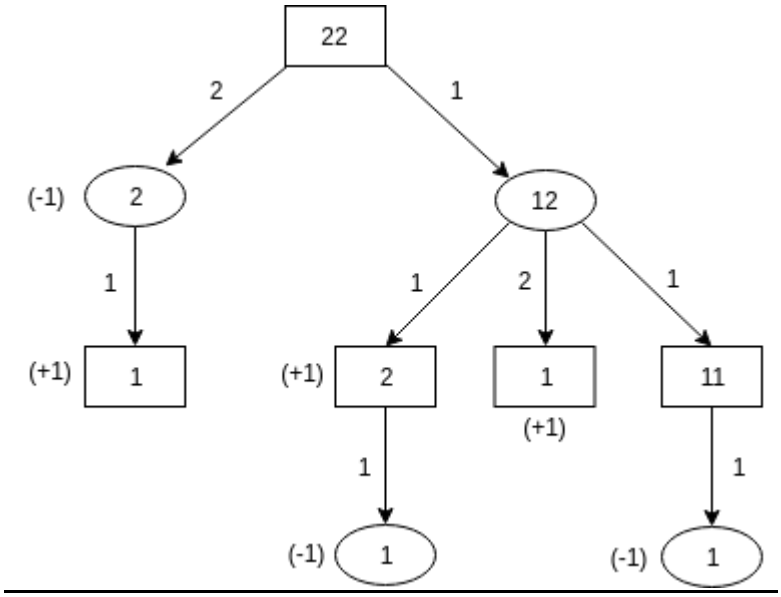


Figure: Game tree with terminal nodes value for 2,2 Nim game

4) Initial values of the terminal node of the tree:

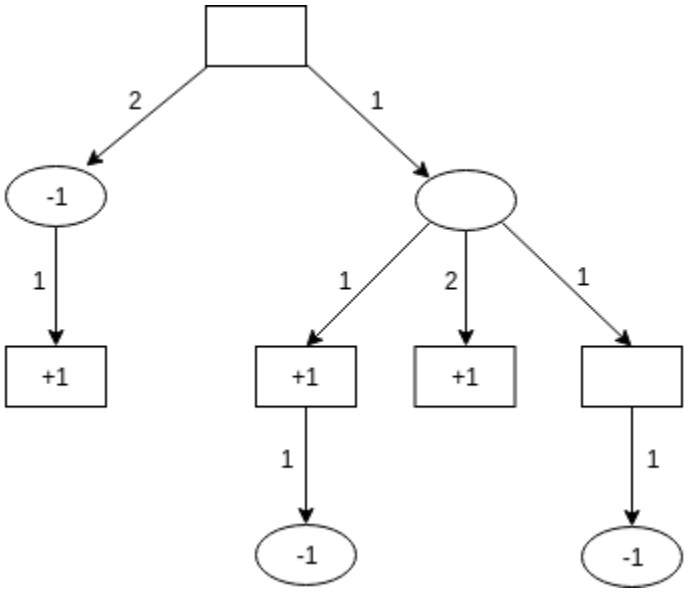


Figure: Initial values of the terminal node of the game tree

After applying Minimax algorithm on this tree:

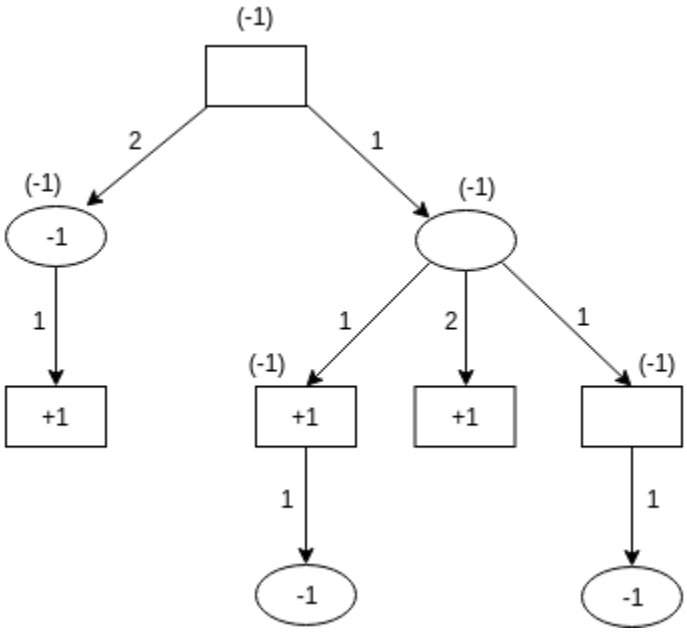


Figure: Minimax algorithm on 2,2 Nim game tree

5) From the tree above, we can see "-1" can be achieved using three leaf of the tree. That means Min can force a win in this game. But Max is totally dependent on Min. Max can only win this game if Min become foolish while removing match. So Max can't force a win in this game.

6) Initial values of the terminal node of the tree:

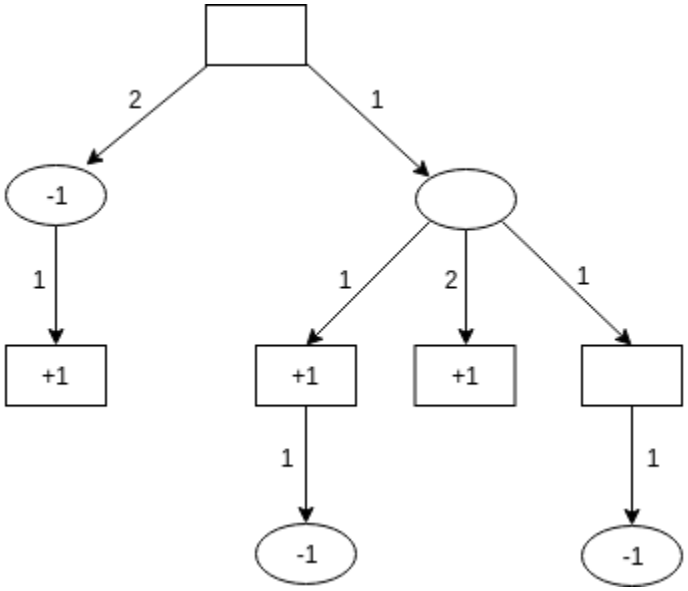


Figure: Initial values of the terminal node of the game tree

After applying alpha beta pruning algorithm on this tree:

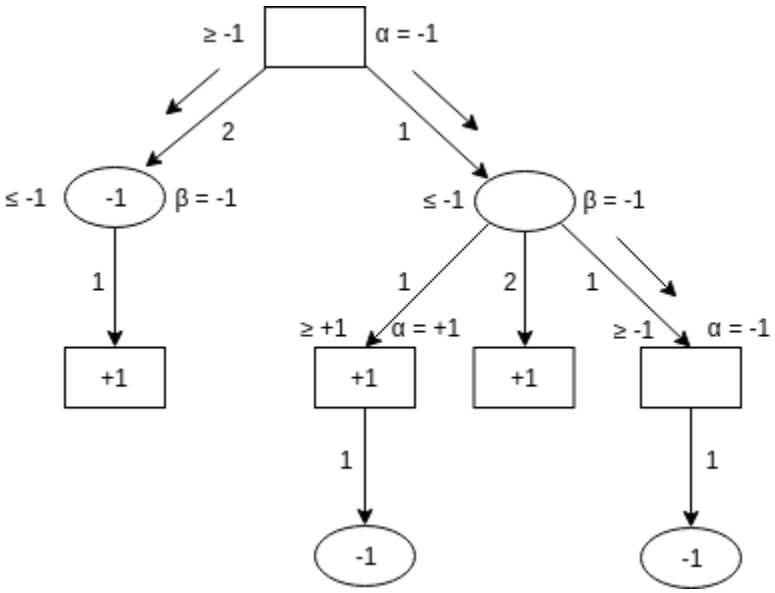


Figure: Alpha beta pruning algorithm on 2,2 Nim game tree

Uncertainty

Q-1: Taking account of the example below, explain the concept of uncertainty.
:" The doorbell rang at 12'0 clock in the midnight.

- Was someone there at the door?
- Did Karim wakeup?"

Explain why we can't 100% certain of ringing doorbell because of someone at the door and Karim wakeup because of ringing of doorbell.

Answer: There can be two propositional logic:

Prop1: AtDoor(x) → DoorBell
Prop2: DoorBell → wake(Karim)

Given doorbell can we say that AtDoor(x) → DoorBell?
--No, the doorbell might start ringing due to some other reason.
Given doorbell can we say that Doorbell → wake(Karim)
--Karim may not wake up even if the bell rings.

Reasons of this uncertainty :

1.Data:

Missing data, unreliable, ambiguous, imprecise representation , inconsistence, subjective, derived from defaults.

2. Expert Knowledge:

--inconsistency between different experts
--plausibility: best guess of expert
--quality: Causal knowledge-deep understanding
Statistical association-observations

3. Knowledge Representation:

--Restricted model of the real system
--Limited expressiveness of the representation mechanism

4. Inference Process:

--Deductive: The derived result is formally correct but wrong in the real system
--Inductive: New conclusions are not well founded.
: Unsound reasoning methods.

Because of these reasons, we can't 100% certain of ringing doorbell because of someone at the door and Karim wakeup because of ringing of doorbell.

Q-2: What is abductive reasoning?

Answer: **Abductive reasoning** (also called abduction, abductive inference, or retroduction) is a form of logical inference which starts with an observation then seeks to find the simplest and most likely explanation. In abductive reasoning, unlike in deductive reasoning, the premises do not guarantee the conclusion. One can understand abductive reasoning as inference to the best explanation, although not all uses of the terms abduction and inference to the best explanation are exactly equivalent.

In the 1990s, as computing power grew, the fields of law, computer science, and artificial intelligence research spurred renewed interest in the subject of abduction. Diagnostic expert systems frequently employ abduction.

Q-3: What is deductive reasoning? Explain why it is not an example of sound reasoning and hence considered as the source of uncertainty.

Deductive Reasoning: Deductive reasoning or logical deduction is the process of reasoning from one or more statements (premises) to reach a logically certain conclusion.

Deductive reasoning goes in the same direction as that of the conditional and links premises with conclusions. If all premises are true, the terms are clear and the rules of deductive logic are followed, then the conclusion reached is necessarily true.

Why it's not an example of sound reasoning and hence considered as the source of uncertainty:

Deductive: The derived result is formally correct but wrong in the real system.

An argument is "**sound**" if it is *valid* and the premises are true.

It is possible to have a deductive argument that is logically *valid* but is not *sound*. Fallacious arguments often take that form.

The following is an example of an argument that is "valid", but not "sound":

1. Everyone who eats carrots is a quarterback.
2. John eats carrots.
3. Therefore, John is a quarterback.

The example's first premise is false – there are people who eat carrots who are not quarterbacks – but the conclusion would necessarily be true, if the premises were true. In other words, it is impossible for the premises to be true and the conclusion false. Therefore, the argument is "valid", but not "sound". False generalizations – such as "Everyone who eats carrots is a quarterback" – are often used to make unsound arguments. The fact that there are some people who eat carrots but are not quarterbacks in the real world which proves the flaw of the argument. So, it's an uncertain reasoning and source of uncertainty.

Q-4: What is inductive reasoning? Explain why it is not an example of sound reasoning and hence considered as the source of uncertainty.

Inductive Reasoning: Inductive reasoning is a form of argument that—in contrast to deductive reasoning—allows for the possibility that a conclusion can be false, even if all of the premises are true. Instead of being valid or invalid, inductive arguments are either strong or weak, according to how probable it is that the conclusion is true. We may call an inductive argument plausible, probable, reasonable, justified or strong, but never certain or necessary. Logic affords no bridge from the probable to the certain.

Why it's not an example of sound reasoning and hence considered as the source of uncertainty:

Inductive: New conclusions are not well founded.
: Unsound reasoning methods.

We can use inductive reasoning in math to find patterns, or in empirical science to draw inferences from observations, or in philosophy to deduce likely truths from other likely truths. No matter how we use inductive reasoning, it is the same basic idea.

Inductive reasoning is determining what is possible given the data we have (which by extension can help us deduce what isn't possible using deduction). The more useful data we have, the more certain we can be about our conclusion.

If we say:

1. Most Greeks had beards.
2. Plato was a Greek.
3. Therefore, Plato likely had a beard.

That would be inductive reasoning. The conclusion didn't necessarily follow from the premises, it was just likely.

No matter what example we use, the basics are the same, inductive reasoning involves the inferring of *B* from *A* where *B* does not necessarily follow from *A*. In other words, induction involves inferring an inference (conclusion) from a premise or set of premises (statement or set of statements) in cases where the inference doesn't necessary follow from the premise(s). Since it doesn't necessarily follow, we therefore have to state qualifiers like confidence and/or likelihood to make our conclusions themselves are factual (f we say "therefore, Plato had a bread – period"… we are essentially lying, even though our data suggested it)

The takeaway is: Meanwhile, even if we aren't dealing with probability, if say we are only dealing with two specific observations, if the inference doesn't necessarily follow from the premise, it is induction.

Due to the inherent uncertainty of induction, we can state simply that **inductive reasoning is any reasoning that deals with probability rather than certainty in its inference** (regardless of the argument form used or the structure of the argument). So, it's a source of uncertainty.

Q-4: Discuss different sources of uncertainty?
Answer:

- **Implications may be weak**
  - Domain experts and knowledge engineers must establish concrete correlations between IF and THEN parts of the rules. Therefore expert systems need to have the ability to handle vague associations, for example by excepting the degree of correlations as numerical certainty factors.

- **Imprecise language like often, rarely, sometimes**
  - Our natural language is ambiguous and imprecise. As a result it can be difficult to express knowledge in the precise IF-THEN form of production rules. Expert systems need quantified results.

- **Uncertain data**
  - Missing data, unreliable, ambiguous, imprecise representation, inconsistent, subjective, derived from defaults, noisy…

- **Uncertain knowledge**
  - Multiple causes lead to multiple effects
  - Incomplete knowledge of causality in the domain
  - Probabilistic/stochastic effects
  - Precise information may be too complete
  - Too many antecedents or consequents

- **Uncertain knowledge representation**
  - Restricted model of the real system
  - Limited expressiveness of the representation mechanism
  - We may not know or guess all the possible antecedents or consequents
  - Experts often provide conflicting information

- **Inference process**
  - Derived result is formally correct, but wrong in the real world
  - New conclusions are not well-founded (inductive reasoning)
  - Incomplete, default reasoning methods

Q-5: Explain propagation of uncertainty.
Answer:
Uncertainty refers to situation involving imperfect or unknown information. It applies to predictions of future events. Propagation of uncertainty is an important aspect of uncertainty analysis. It is a method that transmits the uncertainties of independent variables through an equation to estimate the uncertainty of the final calculation. In absence of interdependencies propagation of uncertain knowledge increase the uncertainty of the conclusion. Propagation of uncertainty means chaining of multiple inference rules. For example-

If
1. The patient has cough
2. The patient has a high WBC count
3. The patient has fever

Then
The patient has pneumonia

Q-6: What is default reasoning? Elaborate it with an example.
Answer:
Default Reasoning: When giving information, we don't want to enumerate all of the exceptions, even if we could think of them all. In default reasoning, we specify general knowledge and modularly add exceptions. The general knowledge is used for cases we don't know are exceptional. Classical logic is monotonic: If g logically follows from A, it also follows from any superset of A. Default reasoning is non-monotonic: When we add that something is exceptional, we can't conclude what we could before.
Example:
Default reasoning can be modeled using
- H is normality assumptions
- F states what follows from the assumptions

An explanation of g gives an argument for g.

A reader of newsgroups may have a default: “Articles about AI are generally interesting”.
H = {interest AI}, where Interesting AI means X is interesting if it is about AI.
With facts: Interesting ← About AI ∧ interest AI.
About AI.
{interest AI} is an explanation for interesting.
We can have exceptions to defaults: false ← Interesting ∧ Uninteresting.
Suppose an article is about AI but is Uninteresting:
Interesting ← About AI ∧ interest AI.
About AI .
Uninteresting.
We cannot explain interesting even though everything we know about the previous we also know about this case.

Q-7: How uncertainty occurs due to default reasoning, elaborate with an example.
Answer:

Generally speaking, to develop a system that reasons with uncertainty means to provide the following:

- a semantic explanation about the origin and nature of the uncertainty
- a way to represent uncertainty in a formal language
- a set of inference rules that derive uncertain conclusions
- an efficient memory-control mechanism for uncertainty management

A reasoning system is monotonic if the truthfulness of a conclusion does not change when new information is added to the system — the set of theorem can only monotonically grows when new axioms are added. In contrast, in a system doing non-monotonic the set of conclusions may either grow or shrink when new information is obtained.

Non-monotonic logics are used to formalize plausible reasoning, such as the following inference step:

Birds typically fly.
Sparky is a bird.

Sparky (presumably) flies.

Such reasoning is characteristic of commonsense reasoning, where default rules are applied when case-specific information is not available. This is also known as default reasoning.

The conclusion of non-monotonic argument may turn out to be wrong. For example, if Sparky is a penguin, it is incorrect to conclude that Sparky flies. Default reasoning often requires jumping to a conclusion and subsequently retracting that conclusion as further information becomes available.

All systems of default reasoning are concerned with the issue of consistency. Inconsistency is resolved by removing the relevant conclusion derived previously by default rules. Simply speaking, the truth value of propositions in a non-monotonic logic can be classified into the following types:

1. Facts that are definitely true, such as "Sparky is a bird"
2. Default rules that are normally true, such as "Birds fly"
3. Tentative conclusions that are presumably true, such as "Sparky flies"

When an inconsistency is recognized, only the truth value of the last type is changed.

Q-8: Show how uncertainty arises from conflicting information?
Ans: There are many sources from that uncertainty may arises .Uncertainty may arises when experts provide conflicting information. That is, individual experts in decision making often disagree with each other. For example, in foraging decision male and female prefer different types of forage.

Q-9: Explain why Bayes theorem forms the basis of probabilistic reasoning?
Ans: Bayesian probability is the interpretation of the concept of probability .The Bayesian interpretation probability is the extension of propositional logic which enables reasoning with hypotheses. That is the proposition whose truth or falsity is uncertain. From the Bayesian view, it is assumed that a probability is assigned to a hypothesis. Under some frequent inference, a hypothesis is tested without being assigned a probability. Thus Bayes theorem forms the basis of probabilistic reasoning.

Q-10: What is the difference between dependent event and independent event? Elaborate your answer with suitable example.
Ans. The differences between dependent event and independent event are given below:

Dependent Event	**Independent Event**
1. Events that affect each other in any way are called dependent event.	**1**. Events that don't affect each other in any way are called dependent event.
2. It is known as conditional probabilities	**2**. It is known as compound probabilities
3. For two dependent events A and B $P(A/B) = \frac{P(A \cap B)}{P(B)}$ $P(B) \neq 0$	**3**. For two independent events A and B $P(A \cap B) = P(A) * P(B)$
4. The occurrence of one event does have the effect on other events	**4.** The occurrence of one event doesn't have the effect on the probability of second event.
5. Example: A boy choses a sock from drawer of socks. Then choses the second sock without replacing the first one.	**5. Example:** When a coin is tossed for two times, the event of getting head in the first throw and the event of getting head in the second throw, are independent events.

Q-11: Differentiate between union probability of two independent events and compound probability of two independent events.

Ans:

Union probability: The probability of the occurrence of the two events A and B denoted as p(AUB),probability that event A and/or event B occurs. This is also known as the probability of union of A.

Compound probability: Compound probability describes independent events.

- Don't affect each other in any way.

It is the joint probability of two independent events.

Q-12: What is conditional probability?

In probability theory, **conditional probability** is a measure of the probability of an event (some particular situation occurring) given that another event has occurred. If the event of interest is *A* and the event *B* is known or assumed to have occurred, "the conditional probability of *A* given *B*", or "the probability of *A* under the condition *B*", is usually written as P(*A* | *B*), or sometimes $P_B(A)$ or P(*A* / *B*).

For example, the probability that any given person has a cough on any given day may be only 5%. But if we know or assume that the person has a cold, then they are much more likely to be coughing. The conditional probability of coughing by the unwell might be 75%, then: P(Cough) = 5%; P(Cough | Sick) = 75%.

Q-13: What are the basic axioms of probability?

In order to understand the axioms for probability, we must first discuss some basic definitions. We suppose that we have a set of outcomes called the sample space *S.* This sample space can be thought of as the universal set for the situation that we are studying. The sample space is comprised of subsets called events E_1, E_2,…, E_n.

We also assume that there is a way of assigning a probability to any event *E*. This can be thought of as a function that has a set for an input, and a real number as an output. The probability of the event *E* is denoted by *P*(*E*).

- **Axiom One**:

  The first axiom of probability is that the probability of any event is a nonnegative real number. This means that the smallest that a probability can ever be is zero and that it cannot be infinite. The set of numbers that we may use are real numbers. This refers to both rational numbers, also known as fractions, and irrational numbers that cannot be written as fractions.

One thing to note is that this axiom says nothing about how large the probability of an event can be. The axiom does eliminate the possibility of negative probabilities. It reflects the notion that smallest probability, reserved for impossible events, is zero.

- **Axiom Two:**

  The second axiom of probability is that the probability of the entire sample space is one. Symbolically we write $P(S) = 1$. Implicit in this axiom is the notion that the sample space is everything possible for our probability experiment and that there are no events outside of the sample space.

  By itself, this axiom does not set an upper limit on the probabilities of events that are not the entire sample space. It does reflect that something with absolute certainty has a probability of 100%.

- **Axiom Three:**

  The third axiom of probability deals with mutually exclusive events.
  If E_1 and E_2 are mutually exclusive, meaning that they have an empty intersection and we use U to denote the union, then $P(E_1 \text{ U } E_2) = P(E_1) + P(E_2)$.

  The axiom actually covers the situation with several (even count ably infinite) events, every pair of which are mutually exclusive. As long as this occurs, the probability of the union of the events is the same as the sum of the probabilities:

  $P(E_1 \text{ U } E_2 \text{ U } \ldots \text{ U } E_n) = P(E_1) + P(E_2) + \ldots + E_n$

  Although this third axiom might not appear that useful, we will see that combined with the other two axioms it is quite powerful indeed.

Q-14: What is causality? Can you demonstrate causality between dependent event or independent even and how?

Causality indicates a relationship between two events where one event is affected by the other. In **statistics**, when the value of one event, or variable, increases or decreases as a result of other events, it is said there is **causation**.

Causality is the area of statistics that is commonly misunderstood and misused by people in the mistaken belief that because the data shows a correlation that there is necessarily an underlying causal relationship.

The use of a controlled study is the most effective way of establishing causality between events. In a controlled study, the sample or population is split in two, with both groups being comparable in almost every way. The two groups then receive different treatments, and the outcomes of each group are assessed.

For example, in medical research, one group may receive a placebo while the other group is given a new type of medication. If the two groups have noticeably different outcomes, the different experiences may have caused the different outcomes.

Due to ethical reasons, there are limits to the use of controlled studies; it would not be appropriate to use two comparable groups and have one of them undergo a harmful activity while the other does not. To overcome this situation, observational studies are often used to investigate correlation and causation for the population of interest. The studies can look at the groups' behaviors and outcomes and observe any changes over time.

The objective of these studies is to provide statistical information to add to the other sources of information that would be required for the process of establishing whether or not causality exists between two variables.

Q-15: Explain why we need the extension of Bayes theorem. What is extended Bayes theorem?

We need the extension of Bayes theorem for using Bayes theorem on four or more events.

Extended Bayes Theorem: We can extend Bayes theorem taking into consideration more probability events. Suppose that the events $B_1,...,B_n$ are conditionally independent given A. Let $\sim A$ denote the complement of A. Then:

$$P(A|B_1,...,B_n) = P(B_1,...,B_n|A) * P(A) / P(B_1,...,B_n)$$

$$= [P(B_1|A) * ... * P(B_n|A) * P(A)] / [P(B_1|A) * ... * P(B_n|A) * P(A) + P(B_1|\sim A) * ... * P(B_n|\sim A) * P(\sim A)]$$

Q-16: What is problem with probability?

Ans: Chances of selecting specific class of samples only

If a surveyor is appointed to survey about any data relating to family members, there is likely chances that s/he will develop a trend of starting to number from the eldest member to the youngest and numbers will be only increasing or decreasing only. In this case, only oldest or the latest generations will be taken as samples.

Redundant and monotonous work

As the surveyor is asked to do a repetitive job to assign the numbers and to take the information, there is likely chances that the surveyor suffers from monotony and the effectiveness of the system will be reduced.

Q-17: What is bayesian approach?

Ans: Bayesian approach is a method of statistical inference in which Bayes' theorem is used to update the probability for a hypothesis as more evidence or information becomes available. Bayesian inference is an important technique in statistics, and especially in mathematical statistics. Bayesian updating is particularly important in the dynamic analysis of a sequence of data. Bayesian inference has found application in a wide range of activities, including science, engineering, philosophy, medicine, sport, and law. In the philosophy of decision theory, Bayesian inference is closely related to subjective probability, often called "Bayesian probability". Bayesian inference derives the posterior probability as a consequence of

two antecedents: a prior probability and a "likelihood function" derived from a statistical model for the observed data. Bayesian inference computes the posterior probability according to Bayes' theorem:

P(H|E)= P(E|H)*P(H)/P(E)

Q-18: What is d-separation?Explain its rule in reducing computation bayesian belief network.

Ans : Two variables (nodes) in the network are D-Separated if the information is "blocked" between the two nodes by some evidence about the nodes in the middle. D-separation is not equivalent to conditional independence. The D-separation of X and Y given Z *implies* the following conditional independence:

P(X,Y|Z)=P(X|Z)P(Y|Z).P(X,Y|Z)=P(X|Z)P(Y|Z).

However D-seperation is a concept that applies specifically to graphical models. You can talk about conditional independence in any context involving random variables.
The statement is talking about something different. What the statement is saying is that just because two nodes X and Y are not D-seperated given some subset of nodes Z, that doesn't mean there doesn't exist some probability distribution which factorizes over GG for which X and Y *are* conditionally independent given Z. In fact there will *always* exist such a distribution.

The Bayes net assumption says:
"Each variable is conditionally independent of its non-descendants, given its parents."
It's certainly possible to reason about independence using this statement, but we can use d-separation as a more formal procedure for determining independence.
Then we follow this procedure:
1. Draw the ancestral graph.
Construct the "ancestral graph" of all variables mentioned in the probability expression. This is a reduced version of the original net, consisting only of the variables mentioned and all of their ancestors (parents, parents' parents, etc.)
2. "Moralize" the ancestral graph by "marrying" the parents.
For each pair of variables with a common child, draw an undirected edge (line) between them. (If a variable has more than two parents, draw lines between every pair of parents.)
3. "Disorient" the graph by replacing the directed edges (arrows) with undirected edges (lines).
4. Delete the givens and their edges.
If the independence question had any given variables, erase those variables from the graph and erase all of their connections, too. Note that "given variables" as used here refers to the question "Are A and B conditionally independent, given D and F?", not the equation "P(A|BDF) =? P(A|DF)", and thus does not include B.
5. Read the answer off the graph.
If the variables are disconnected in this graph, they are guaranteed to be independent.
If the variables are connected in this graph, they are not guaranteed to be independent.*
Note that "are connected" means "have a path between them," so if we have a path X-Y-Z, X and Z are considered to be connected, even if there's no edge between them.
If one or both of the variables are missing (because they were givens, and were therefore deleted), they are independent.

* We can say “the variables are dependent, as far as the Bayes net is concerned” or “the Bayes net does not require the variables to be independent,” but we cannot guarantee dependency using d-separation alone, because the variables can still be numerically independent (e.g. if P(A|B) and P(A) happen to be equal for all values of A and B).

Q-19: Write the steps for constructing BN.

Answer: A Bayesian Network consists of a directed acyclic graph (DAG) where each node is associated with a random variable X_i , a domain for each variable X_i and a set of conditional probability distribution for each node X_i give its parents in the graph

$$P(X_i | \text{ Parents}(X_i))$$

Constructing Bayesian Networks:

The steps for constructing bayesian network are given below:

1. choose an ordering of variable $X_1,\ldots,X_n$
2. for i=1 to n

- add X_i to the network
- select parents from $X_1,\ldots,X_{i-1}$ such that

$P(X_i \mid \text{Parents}(X_i)) = P(X_i \mid X_1,\ldots,X_{i-1})$

This choice of parents guarantees:

$P(X_1, \ldots, X_n) = \pi_{i=1} P(X_i \mid X_1, \ldots, X_{i-1})$ (chain rule) $= \pi_{i=1} P(X_i \mid \text{Parents}(X_i))$ (by construction)

Q-20: Why we consider polytree in BN?

Answer: According to graph theory, a polytree is a directed acyclic graph whose underlying graph is tree. If we replace its directed edges with undirected edges, we obtain an undirected graph that is both connected and acyclic.

A Bayesian network is a probabilistic graphical model that represents a set of variables and their conditional dependencies via directed acyclic graph. Bayesian networks are ideal for taking an event that occurred and predicting the likelihood that any one of several possible known causes was the contributing factor.

Polytree in BN:

- The nodes are random variables.
- Arrows connect pairs of nodes.
- An arrow from node X to node Y means X has a direct influence on Y
- Easy for a domain expert to determine these relationships
- The absence/presence pf arrows will be made mode precise later on

Q-21: What is conditional independence?

Answer: In probability theory, two random events A and B are conditionally independent given a third event C precisely if the occurrence of A and the occurrence of B are independent events in their conditional probability distribution given C. In other words, A and B re conditionally independent given C if and only if given knowledge that C occurs, knowledge of whether A occurs provides no information on the likelihood of B occurring, and the knowledge of whether B occurs provides no information on the likelihood of A occurring.

The notation of a Bayes Network implies

$$P(E|C,A)=P(E|C)$$

If it is known that C did or did not happen, it doesn't matter whether or not *A* happened. So at this point we have:

$$P(E,C|A)=P(E|C)\cdot P(C|A)$$

Q-22: Write the different types of inference procedures.

Answer: An inference procedure G = {$G_{k,z}(.)$} assigns a cumulative distribution function $G_{k,z}(.)$, called an estimate, to every sample (k,z) such that the estimate $G_{k,z^\wedge}$ first order dominates the estimate $G_{k,z}(.)$ whenever $z^\wedge>z$.

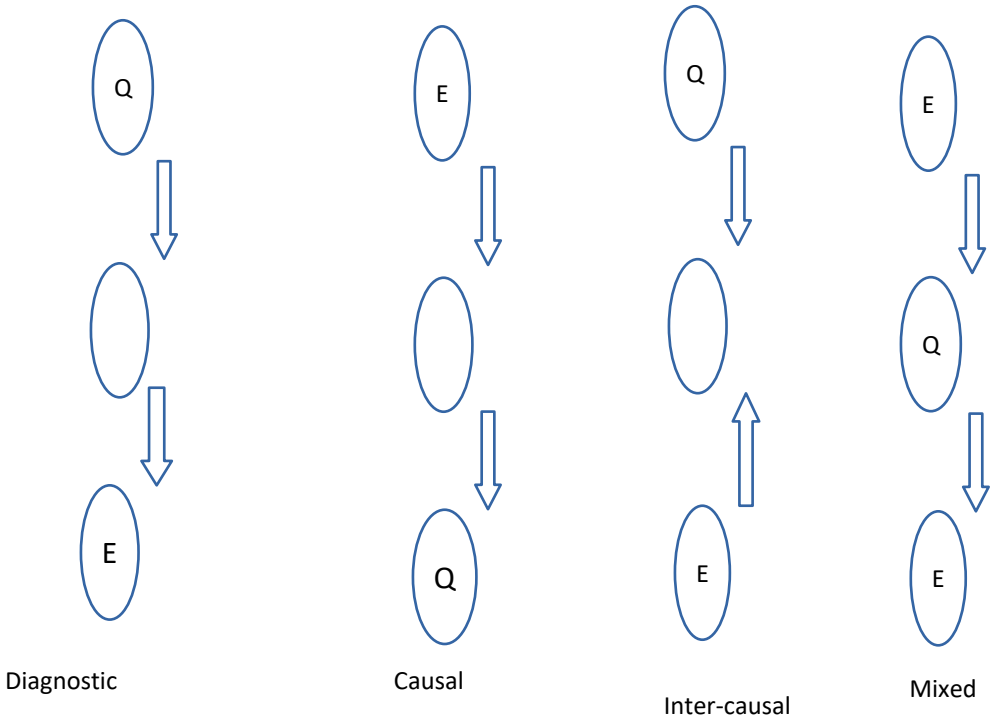
The different types of inference procedure are:

- Diagnostic inference
- Causal inference
- Intercausal inference
- Mixed inferences
-

Q-23: Diagrammatically compare and contrast inference procedures in BN.

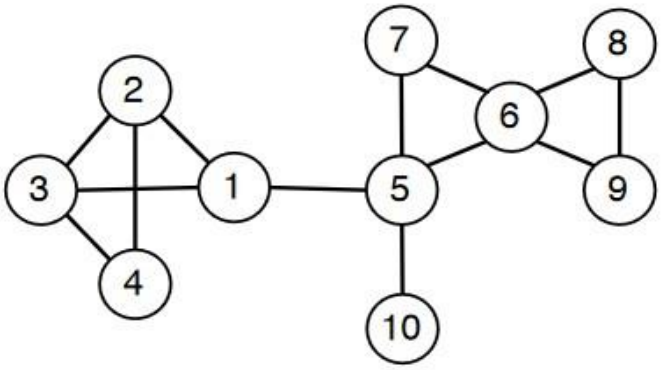
Answer: Diagrammetically comparing and contrasting inference procedurea in BN are given below:

- Diagnostic inference (from effects to causes): given that John calls, the probability of Burglary is P(B|J)
- Causal inference (from causes to effects): given Burglary, the probability of John calls is P(J|B) and Mary calls P(M|B)
- Intercausal inference(between causes of a common event): given alarm, the probability of burglary is P(B|A); again now given that earthquake, the probability of burglary is P(B|A^E)
- Mixed inference(some causes and some effects known): given John calls and earthquake, the probability of alarm is P(A|J^¬E)

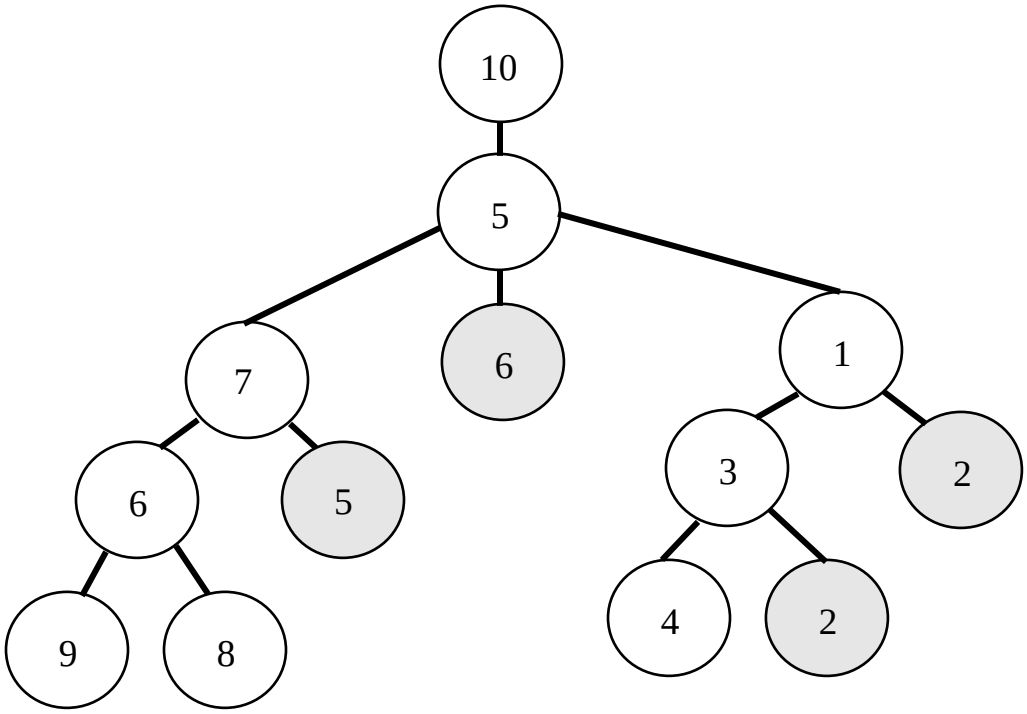


DFS.pdf

1. Show the DFS tree on the above graph. Use the vertices in decreasing order.
   Ans:

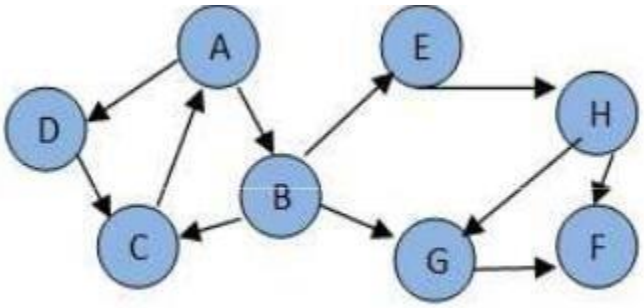


The order of expansion: **10, 5, 7, 6, 9, 8, 1, 3, 4, 2**

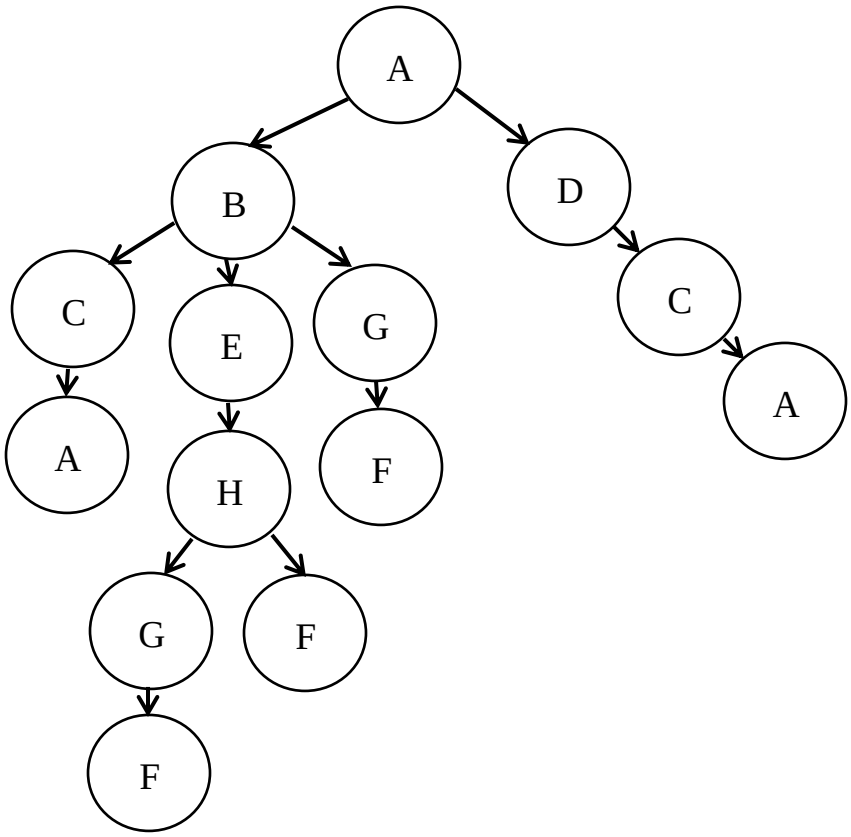


2. a) In what order will the nodes be visited using a Breadth First Search?

   b) In what order will the nodes be visited using a Depth First Search?



Ans:
The tree generated from the given graph is:



Lets take **A** as start and **G** as goal

a) By using Breadth First Search:

Expand Node	**Open Node**	**Closed Node**
	{A}	{ }
A	{B, D}	{A}
B	{D, C, E, G}	{A, B}
D	{C, E, G, C'}	{A, B, D}
C	{E, G, C', A'}	{A, B, D, C}

Expand Node	Open Node	Closed Node
E	{G, C', A', H}	{A, B, D, C, E}
G	{C', A', H, F}	{A, B, D, C, E, G}

Order of Visit: **A, B, G**

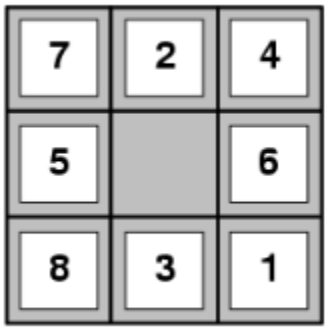
b) By using Depth First Search:

Expand Node	Open Node	Closed Node
	A	{ }
A	{B, D}	{A}
B	{C, E, G', D}	{A, B}
C	{A', E, G', D}	{A, B, C}
A'	{E, G', D}	{A, B, C, A'}
E	{H, G', D}	{A, B, C, A', E}
H	{G, F, G', D}	{A, B, C, A', E, H}
G	{F', F, G', D}	{A, B, C, A', E, H, G}

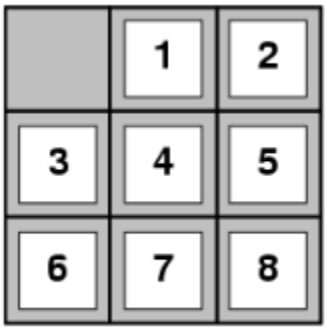
Order of Visit: **A, B, E, H, G**

Informed and Uninformed Search.pdf

1. Prove that the Manhattan Distance heuristic for 8-puzzle is admissible Here,
   - Tiles cannot move along diagonals, so each tile has to move at least d(n) steps to its goal
   - Any move can only move one tile at a time



Start State



Goal State

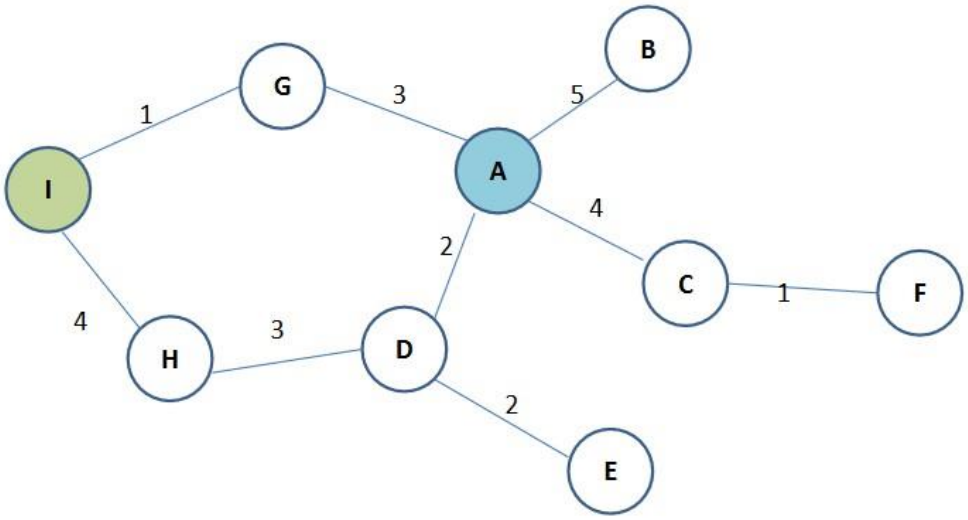
Ans:

Tiles	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Manhattan Distance	3	1	2	2	2	3	3	2

Total Manhattan Distance: 3+1+2+2+2+3+3+2 = 18
The actual solution for this 8 puzzle is 26 steps long. Here we solved it in 18 steps which is less than the number of actual steps.

So, it is proven that the Manhattan Distance heuristic for 8-puzzle is admissible.

2. Using Uniform Cost Search, Find the Goal: path A → I



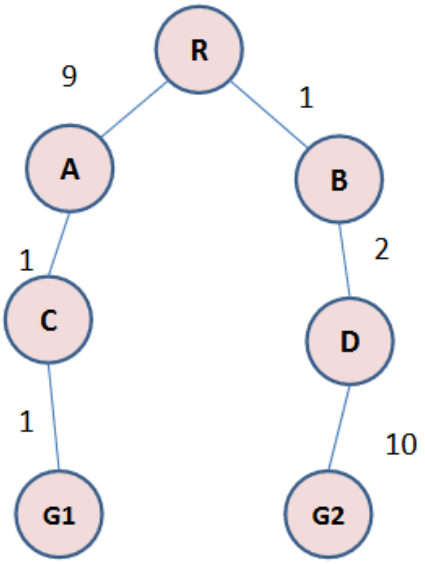
Ans:
Using Uniform Cost Search,

Expand Node	**Node List**
	A(0)
A	D(2), G(3), C(4), B(5)
D	G(3), C(4), E(4), H(5), B(5)
G	C(4), E(4), I(4), H(5), B(5)

The Path is: **A, G, I**
Path Cost is 4.

3. Using A* Search, prove that, f(n) = g(n) + h(n)
   Where,
   f(n) – estimated total cost of path through n to goal
   g(n) –cost so far to reach n

h(n) –estimated cost from n to goal



Ans:

Suppose R is start state and G1 is goal state.

Node	g(n)	h(n)	f(n)
R	0	11	11
A	9	2	11
B	1	inf	inf
C	10	1	11
D	3	inf	inf
G1	11	0	11
G2	13	inf	inf

Using A* we get,

Expand Node	**Node List**
	R(0)
R	A(11), B(inf)
A	C(11), B(inf)
C	G(11), B(inf)
G1	B(inf)

The path is **R, A, C, G1**

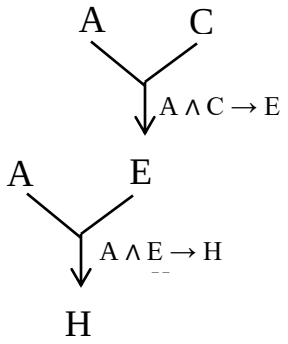
From the first table we can see that f(n) is always the sum of g(n) and h(n).

Forward Chaining.pdf

1. Use Forward Chaining to solve the following problem: Given:
   A
   B
   C
   $A \wedge B \rightarrow D$
   $B \wedge D \rightarrow F$
   $F \rightarrow G$
   $A \wedge E \rightarrow H$
   $A \wedge C \rightarrow E$

   Is H true?
   Draw a tree to illustrate the search for a proof.

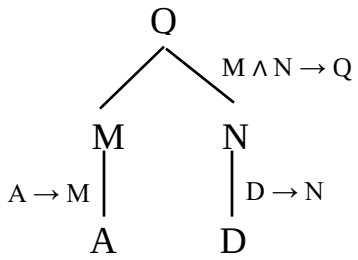
Ans:
In our KB we have A B C which are True. By using $A \wedge C \rightarrow E$ and $A \wedge E \rightarrow H$ we can prove H is true.



2. Use Backward Chaining on the following KB to prove Q:
   $P \rightarrow Q$
   $E \rightarrow B$
   $R \rightarrow Q$
   $M \wedge N \rightarrow Q$
   $A \wedge B \rightarrow P$
   $A \rightarrow M$
   $C \rightarrow M$
   $D \rightarrow N$
   D
   A

Draw a tree to illustrate the search for a proof. Mark the nodes that are satisfied in this KB. What is the proof of Q? (Write explicitly a sequence of steps to obtain Q)

Ans:



Steps to solve:

Rules	Unknown	KB
	Q	A, D
M ∧ N → Q	Q, M, N	A, D
A → M	Q, N	A, D, M
D → N		A, D, M, N, Q

Planning.pdf

1. Consider the case of a game for three players, who we will call A, B, and C. In such a game, we require an evaluation function that gives us 3 values, indicating how good the position is for player A, B, or C, respectively. So, if game position X has evaluation (1,3,5) and position Y has evaluation (4,3,2), A will prefer Y, B will treat them the same, and C will prefer X.

   For each of the following internal nodes, give its evaluation based on the evaluation values for its children, depending on whose turn it is to play. For example, in a two player game, you may have:
   (a) A to play. Children: (3,1), (8,12), (-2,7)
   (b) A to play. Children: (1,2,3), (5,6,1), (1,2,0)
   (c) B to play. Children: (1,2,3), (5,6,1), (1,2,0)
   (d) C to play. Children: (1,2,3), (5,6,1), (1,2,0)
   (e) A to play. Children: (1,2,3), (-1,-2,-3), (66,1,44)

Ans:

(a) A will prefer (8,12)
(b) A will prefer (5,6,1)

(c) B will treat all of them same.
(d) C will prefer (1,2,0)
(e) A will prefer (66,1,44)

Bayesian Network.pdf

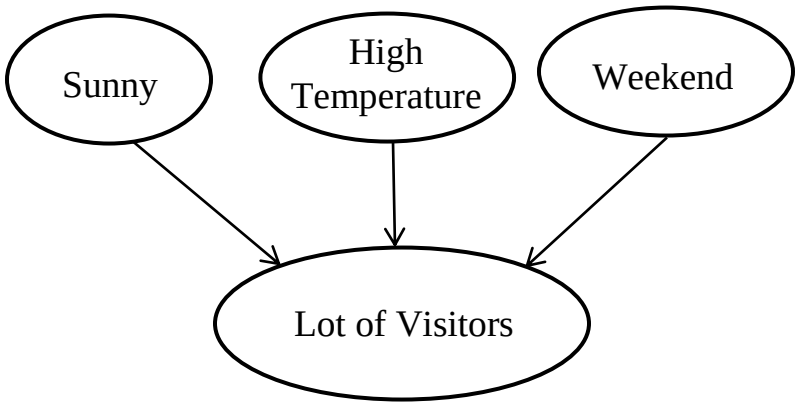
1.

	Feature 1 Sunny?	Feature 2 High Temperature?	Feature 3 Weekend?	Class Lots of Visitors?
Day 1	yes	yes	yes	yes
Day 2	yes	no	yes	yes
Day 3	no	yes	no	yes
Day 4	yes	yes	no	yes
Day 5	yes	yes	no	yes
Day 6	yes	no	no	no
Day 7	no data since you were on business travel			
Day 8	no	no	yes	no

a) Show the Bayesian network that will form a Naive Bayesian Learner, so that the Naïve Bayesian Learner learn from it.
b) What's the probability that the learned Bayesian network will predict that the number of visitors on a sunny and hot weekend day?

Ans:

a)



b)

Let, X= (Sunny = Yes, High Temperature = Yes, Weekend= Yes)

P(Lots of Visitor = Yes) = $\frac{5}{7}$

P(Lots of Visitor = No) = $\frac{2}{7}$

P(Sunny = Yes | Lots of Visitor = Yes) = $\frac{4}{5}$

P(Sunny = Yes | Lots of Visitor = No) = $\frac{1}{2}$

P(High Temperature = Yes | Lots of Visitor = Yes) = $\frac{4}{5}$

P(Sunny = Yes | Lots of Visitor = No) = $\frac{0}{2}$

To avoid zero probability, using laplasian correction,

P(Sunny = Yes | Lots of Visitor = No) = $\frac{1}{2+2} = \frac{1}{4}$

P(Weekend= Yes | Lots of Visitor = Yes) = $\frac{3}{5}$

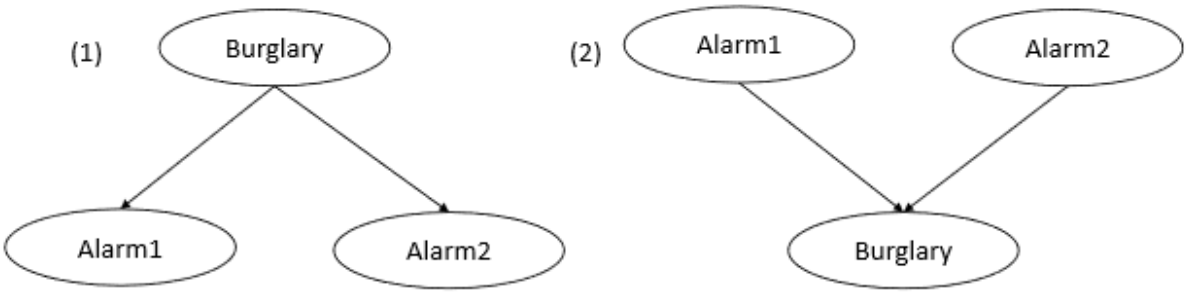
P(Weekend = Yes | Lots of Visitor = No) = $\frac{1}{2}$

P(X | Lots of Visitor = Yes) = $\frac{4}{5}\times\frac{4}{5}\times\frac{3}{5}\times\frac{5}{7}$ = 0.183

P(X | Lots of Visitor = No) = $\frac{1}{2}\times\frac{1}{4}\times\frac{1}{2}\times\frac{2}{7}$ = 0.0178

So, the learned Bayesian network will predict lot of visitors on a sunny and hot weekend day YES 91% time and NO 9% times.

2. Which one of the two Bayesian networks given below makes independence assumptions that are true? Explain all of your reasoning.
   Here,
   Alarm1 = the first alarm system rings,
   Alarm2 = the second alarm system rings
   Burglary = a burglary is in progress

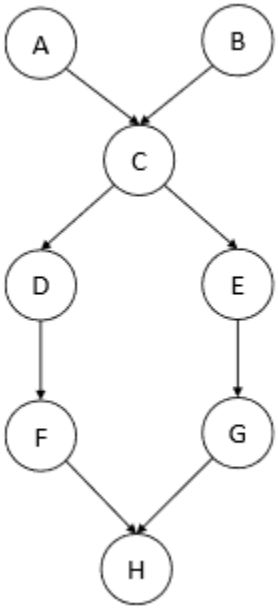


Ans:

Bayesian Network1 gives independence assumptions that are true.

Because, in network1 the Alarm1 and Alarm2 is independent given burglary. So, whenever there is a burglary either Alarm1 or Alarm2 or both will be sounded. But in case of network2 Alarm1 and Alarm2 are independent of Burglary. So, in case of Burglary, the alarms may not sound.

3.



a) Are D and E necessarily independent given evidence about both A and B?
b) Are A and C necessarily independent given evidence about D?
c) Are A and H necessarily independent given evidence about C?

Ans:
a) No. Because they are connected through C.
b) No. Because they are directly connected.
c) Yes. Because they are not connected.

4. Construct an example where a naive Bayesian learner predicts for a feature vector that the predicted class must be true with probability 1 when, in reality, it is false.

Ans:

Feature 1	Feature 2	Class
Yes	Yes	Yes
Yes	No	Yes
No	Yes	Yes

In this example naïve learner will learn only the class “YES”. So, when predicting, it will always predict yes with probability 1.For example in case of X = (Feature 1 = No, Feature 2 = No) the network will predict Class = Yes but actually the class is No.

Assignment on Constraint Satisfaction Problem

1. Below is the web link of the paper entitled "A belief rule based expert system to assess suspicion of acute coronary syndrome (ACS) under uncertainty" published in the journal of Soft Computing of Springer. Download and read the paper carefully. Identify the section where various constraints are mentioned to develop the optimal learning model. Investigate these constraints and submit a report.
https://link.springer.com/article/10.1007%2Fs00500-017-2732-2

Report:
The section under which the various constraints have been discussed is Section 4.2, *BRBES to evaluate ACS suspicion*. Here under the subsection 4.2.5, *BRBES training model*, the different constraints have been discussed.
The variables on which the constraints apply are Rule Weights(θ_k) of L rules, antecedent attribute weights (δ_k) of the L rules, consequent belief degrees [β_{jk} (j = 1, 2, 3, k =1,.., L)] which are 3L in number and Severity Scores of Consequnt Reference Values [$\mu(O_j)(j=1,...,3)$] which are 3 in number. So we have a total of (L+L+3L+3)= 5L+3 variables. Here L is the total number of Rules.
The constraints applied are as follows:

1) Severity scores of three reference levels $\mu(O_j)(j=1,...,3)$:

a. $1 \geq \mu(O_j)(j = 1, . . ., 3) \geq 0$; [This is a unary inequality constraint on $\mu(O_1(High))$, $\mu(O_2(Medium))$ and $\mu(O_3(Low))$ each].

b. $\mu(O_1(High)) \geq \mu(O_2(Medium)) \geq \mu(O_3(Low))$; [This is a higher order ternary inequality constraint on $\mu(O_1(High))$, $\mu(O_2(Medium))$ and $\mu(O_3(Low))$].

2) Rules Weights θ_k (k=1,…,L):

$1 \geq \theta_k(k=1,\ldots,L) \geq 0.01$ [This is a unary inequality constraint on each of the L rule weights]

3) Antecedent Attribute Weights δ_k (k=1,…,L):

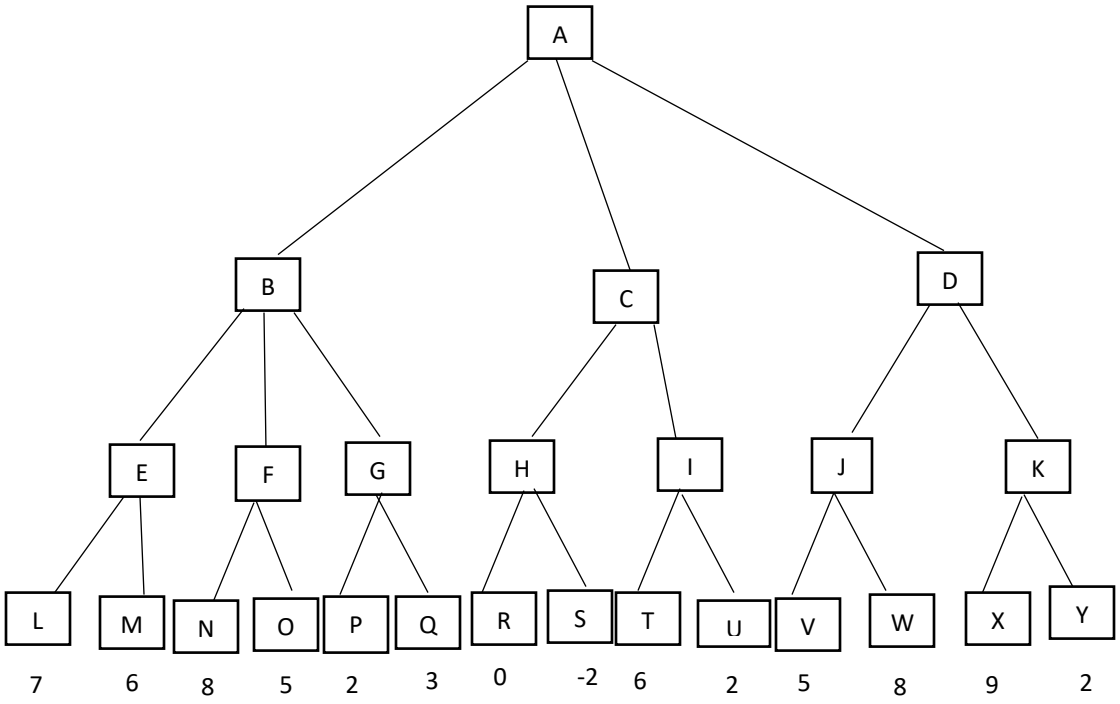
$1 \geq \delta_k \geq 0$ [This is a unary inequality constraint on each of the L Antecedent Attribute weights]

4) Consequent Belief Degrees β_{jk} (j=1,2,3,k=1,..,L):

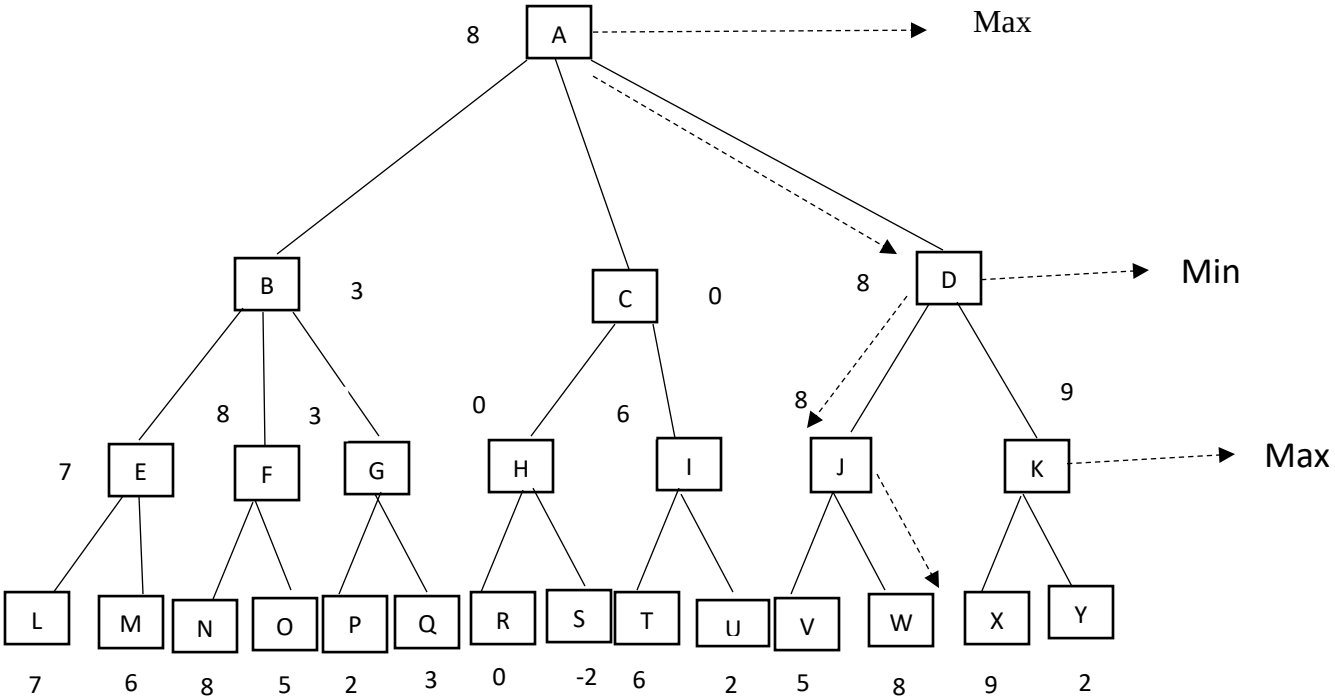
a. $1 \geq \beta_{jk}$ (j=1,2,3,k=1,..,L) ≥ 0; [This is a unary inequality constraint on each of the 3L Antecedent Attribute weights]

b. $\Sigma^3_{j=1}\beta_{jk}(j=1,2,3,k=1,..,L) \geq 0$ [This is a higher order ternary inequality constraint on Consequent Belief Degrees β_{1k}, β_{2k} and β_{3k} where k = 1,…,L]

1. Consider the following game tree in which static scores are all form of the first players point of view



Suppose the first player is the maximizing player. What move should be chosen?
Answer:



From above figure dotted arrow shows the path A→D→J→W. Which A player should choose.

2. What nodes would not need to be examined using the alpha beta pruning procedure?

B & C nodes would not need to be examined using the alpha beta pruning procedure.

3. Why does the search in game playing programs always proceed forward from the current position rather than backward from the goal state?

Because there may be lots of "goal"(winning) states. Only one current state.

4. Is the minimax procedure a depth-first or breadth first search procedure?

MinMax procedure is a depth first search. Minimax is better implemented as a depth-first search, which requires only a linear amount of memory in relation to tree depth. The structure used for this search is a stack, either through recursive function calls or a direct stack based implementation.

5: Answer

If the number of different distinguishable world situations is sufficiently small, a graph representing all possible action and situation can be stored explicitly.
So if number of possible situation is sufficiently small/limited in a game, we can store each time a new situation is generated. We can also check if the situation was generated and evaluated before as they are saved in graph.
But if the number of possible situation is not sufficiently small we can not store each situation in graph because we don't have enough space to store. It is not a good idea in that case.
So we can say if number of possible situation is sufficiently small/limited in a game so that we can store the situations in our memory then it is a good idea to search graph and check each time a new situation is generated if it was generated and evaluated before.
Modifying minmax procedure:

Normally on minmax procedure, we use bfs/dfs or heuristic search technique for searching and there is a termination condition to stop searching when it seems not necessary, Because searching all possible situation is not possible as the number of possible situation is very high. After search terminates we estimate the best possible move for current player from the search tree. We estimate the best move from leaf nodes by applying 'static evaluation function' .
Now on modifying minmax function when we encounter any situation we check if the situation is new, if it is new then we save the situation by mapping or some other procedure and for every situation we save best possible move after estimating .
We can check if the situation is new as situations are saved.
If the situation was generated before then we don't proceed further because we have already calculated the value.
So we can see on our modified minmax approach we don't calculate any situation twice. But the limitation is the number of possible situation must be limited as we have to save each generated situation on memory.

The modified algorithm has been given below:

Algorithm:

function map(state) **returns** mapped value of a state
return mapped_value //This should be calculated

function get(state) **returns** a value
return result[map(state)]
function calculated(state) **returns** boolean value
if result[map(state)] return true
return false

function MINIMAX-DECISION(state) **returns** an action
return argmaxa ∈ ACTIONS(s) MIN-VALUE(RESULT(state, a))

function MAX-VALUE(state) **returns** a utility value
if calculated(state)]**return** get(state)
if TERMINAL-TEST(state) **then return** UTILITY(state)
v ←−∞
for each a **in** ACTIONS(state) **do**
v ←MAX(v, MIN-VALUE(RESULT(s, a)))
result[map(state)]=v;
return v

function MIN-VALUE(state) **returns** a utility value
if calculated(state) **return** get(state)
if TERMINAL-TEST(state) **then return** UTILITY(state)
v←∞
for each a **in** ACTIONS(state) **do**
v ←MIN(v, MAX-VALUE(RESULT(s, a)))
result[map(state)]=v;
return v

6: Answer
Minmax procedure for more than two player.
Normally on two player minmax procedure, we use bfs/dfs or heuristic search technique for searching and there is a termination condition to stop searching when it seems not necessary, Because searching all possible situation is not possible as the number of possible situation is very high.
After search terminates we estimate the best possible move for current player from the search tree. We estimate the best move from leaf nodes by applying 'static evaluation function' .
Here one player wants maximum valu but other wants minimum.
Now let consider a game where there are three player and first player wants maximum and named as max1, second player wants minimum and named as min, third player wants maximum value named as max2.
A ply of ply-depth k in this game consists of nodes of 3k, 3k+1, 3k+2 depths of nodes where 3k th node means it is max1's move next, 3k+1 means it is min's move next and 3k+2 means it is max's move next.

Here max1 and max2 want to maximize the value where min wants to minimize.
Now we use we use bfs/dfs or heuristic search technique and things are quite same as two player. On each player's turn they want to maximize(max1 and max2 want this) or minimize(min wants this).
After search terminates we estimate the best possible move for current player from the search tree. We estimate the best move from leaf nodes by applying 'static evaluation function' .
The algorithm for a 3 player game is given below on minmax procedure.
Here to calculate value for max1,min and max2 we have funcion MAX-VALUE1,MIN-VALUE and MAX-VALUE2.
The MINIMAX-DECESION returns the best action for max1. The modified algorithm is as follows:**function** MINIMAX-DECISION(state) **returns** an action
return argmax
a ∈ ACTIONS(s) MIN-VALUE(RESULT(state, a))

function MAX-VALUE1(state) **returns** a utility value
if TERMINAL-TEST(state) **then return** UTILITY(state)
v ←−∞
for each a **in** ACTIONS(state) **do**
v ←MAX(v, MIN-VALUE(RESULT(s, a)))

function MAX-VALUE2(state) **returns** a utility value
if TERMINAL-TEST(state) **then return** UTILITY(state)
v ←−∞
for each a **in** ACTIONS(state) **do**
v ←MAX(v, MAX_VALUE1(RESULT(s, a)))
return v

function MIN-VALUE(state) **returns** a utility value
if TERMINAL-TEST(state) **then return** UTILITY(state)
v←∞
for each a **in** ACTIONS(state) **do**
v ←MIN(v, MAX-VALUE2(RESULT(s, a)))
return v

When it comes two 4 player game let the players are max1,min1,max2,min2 respectively where max1 and max2 want to get maximum value and min1 and min2 want to get minimum value.
A ply of ply-depth k in this game consists of nodes of 4k, 4k+1, 4k+2,4k+3 depths of nodes where 4k th node means it is max1's move next, 4k+1 means it is min1's move next and 4k+2 means it is max2's move next, 4k+3 means min2's move next.
Now we use we use bfs/dfs or heuristic search technique and things are quite same as two player. On each player's turn they want to maximize(max1 and max2 want this) or minimize(min wants this).
After search terminates we estimate the best possible move for current player from the search tree. We estimate the best move from leaf nodes by applying 'static evaluation function' .
The algorithm for a 4 player game is given below on minmax procedure.
Here to calculate value for max1,min1 and max2,min2 we have funcion MAX-VALUE1,MIN-VALUE1 and MAX-VALUE2,MIN-VALUE2.
The MINIMAX-DECESION returns the best action for max1.
The Algorithm is as follows:

```
function MINIMAX-DECISION(state) returns an action
return argmax
a ∈ ACTIONS(s) MIN-VALUE1(RESULT(state, a))

function MAX-VALUE1(state) returns a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
v ←−∞
for each a in ACTIONS(state) do
v ←MAX(v, MIN-VALUE1(RESULT(s, a)))

function MAX-VALUE2(state) returns a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
v ←−∞
for each a in ACTIONS(state) do
v ←MAX(v, MIN-VALUE2((RESULT(s, a)))
return v

function MIN-VALUE1(state) returns a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
v←∞
for each a in ACTIONS(state) do
v ←MIN(v, MAX-VALUE2((RESULT(s, a)))
return v

function MIN-VALUE2(state) returns a utility value
if TERMINAL-TEST(state) then return UTILITY(state)
v←∞
for each a in ACTIONS(state) do
v ←MIN(v, MAX-VALUE1(RESULT(s, a)))
return v
```