BE - SEMESTER–VII (NEW)
EXAMINATION – SUMMER 2022
Subject Code:3170716
Date:03/06/2022
Subject Name: Artificial Intelligence

Q.1

(a) Define AI. What are the task domains of AI? 03

ANSWER: Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions.

Task Domains of Artificial Intelligence (AI)

1.Mundane Tasks:

- Perception
- Vision
- Speech
- Natural Languages
- Understanding
- Generation
- Translation
- Common sense reasoning
- Robot Control

➢ Humans have been learning mundane (ordinary) tasks since their birth. They learn by perception, speaking, using language, and locomotives.

➢ For humans, the mundane tasks are easiest to learn. The same was considered true before trying to implement mundane tasks in machines.

➢ Earlier, all work of AI was concentrated in the mundane task domain.

2.Formal Tasks:

- Games: chess, checkers, etc
- Mathematics: Geometry, logic, Proving properties of programs

➢ Formal Tasks are the tasks that need deals with verification, theorem proving, deals with Math, Games, etc.

3.Expert Tasks:

- Engineering (Design, Fault finding, Manufacturing planning)
- Scientific Analysis
- Medical Diagnosis
- Financial Analysis

➢ Expert Tasks are those tasks, which involves scientific analysis, and different domain analysis, like Financial, healthcare, creative aspects, etc.

➢ Now researchers have understood that to solve mundane tasks, they need better and more efficient algorithms, and a much more knowledge base to help them tackle the problems they have set out to solve. And that is the reason that AI has shifted more on working with Expert Tasks, to enhance the capabilities of the AI system.

(b) Explain Water Jug problem with Sate Space Search method. 04

ANSWER: Problem Definition: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

1.Initial State:

➥ We will represent a state of the problem as a tuple (x, y), where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-gallon jug.

➥ Note that $0 \leq x \leq 4$, and $0 \leq y \leq 3$.

➥ Here the initial state is (0, 0). The goal state is (2, n) for any value of n.

2.Production Rules

Sr.	Current state	Next State	Descriptions
1	(x, y) if x < 4	(4,y)	Fill the 4 gallon jug
2	(x, y) if y <3	(x,3)	Fill the 3 gallon jug
3	(x, y) if x > 0	(x-d, y)	Pour some water out of the 4 gallon jug
4	(x, y) if y > 0	(x, y-d)	Pour some water out of the 3 gallon jug
5	(x, y) if x>0	(0, y)	Empty the 4 gallon jug
6	(x, y) if y >0	(x,0)	Empty the 3 gallon jug on the ground
7	(x, y) if x+y >= 4 and y > 0	(4, y-(4-x))	Pour water from the 3 gallon jug into the 4 gallon jug until the 4 gallon jug is full
8	(x, y) if x+y >= 3 and x>0	(x-(3-y), 3)	Pour water from the 4 gallon jug into the 3-gallon jug until the 3 gallon jug is full
9	(x, y) if x+y <=4 and y>0	(x+y, 0)	Pour all the water from the 3 gallon jug into the 4 gallon jug
10	(x, y) if x+y <= 3 and x>0	(0, x+y)	Pour all the water from the 4 gallon jug into the 3 gallon jug
11	(0,2)	(2,0)	Pour the 2 gallons from 3 gallon jug into the 4 gallon jug
12	(2,y)	(0,y)	Empty the 2 gallons in the 4 gallon jug on the ground

One of the possible solutions is given as:

3. Productions for the water jug problem

Gallons in the 4- gallon jug	Gallons in the 3- gallon jug	Rule applied
0	0	2
0	3	9
3	0	2
3	3	7
4	2	5 or 12
0	2	9 0r 11
2	0	--

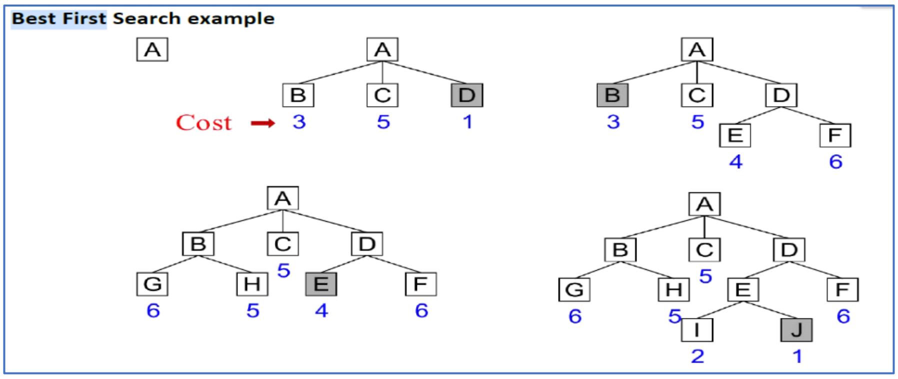
(c) Explain Best First Search with suitable example. 07

ANSWER: Best first search is a combination of DFS and BFS.

- In DFS, all the competing branches are not expanded and BFS never gets trapped on dead-end paths.
- So, if we combine both and continue following a single path until some competing path looks more promising, it is what best first search is.
- Best first search explores the most promising node using the heuristic value of the node.

➢ The algorithm is as follows:

1) Define a list, OPEN, which has the start node, s.
2) IF the list is empty, return failure.
3) Remove from the list the node n with the best score (the node where f is the minimum), and move it to a list, CLOSED.
4) Expand node n.
5) IF any successor to n is the goal node, return success and the solution (by tracing the path from the goal node to s).
6) FOR each successor node:
a) apply the evaluation function, f, to the node.
b) IF the node has not been in either list, add it to OPEN.
7) Continue the loop and go back to step-2.

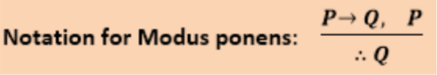


Q.2

(a)Define the following.

1. Modus Ponens 2. Horn Clause 3. Existential Quantifier 03

1. Modus Ponens: The Modus Ponens rule is one of the most important rules of inference, and it states that if P and P → Q is true, then we can infer that Q will be true. It can be represented as:



Notation for Modus ponens: $\frac{P \to Q,\quad P}{\therefore Q}$

2. Horn Clause: A Horn clause is a clause (a disjunction of literals) with at most one positive, i.e. unnegated, literal.

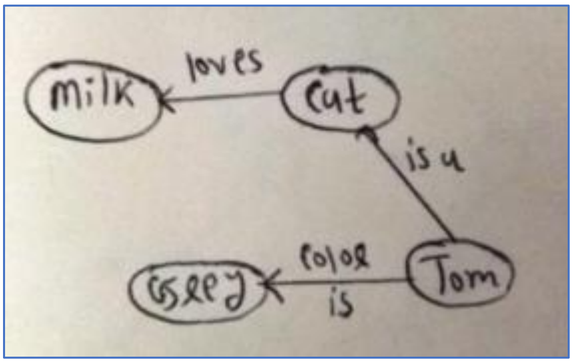
3.Existential Quantifier: Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something. It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

(b)Explain Semantic Net with example. 04

ANSWER: SEMANTIC NET

- A semantic net (or semantic network) is a knowledge representation technique used for propositional information.
- In semantic net, information is represented as a set of nodes connected to each other by set of labeled arcs, which represent relationships among the nodes.
- Semantic net allows us to perform inheritance reasoning as all members of a class will inherit all the properties of superclass.
- Semantic nets allow multiple inheritance.
- It allows a common form of inference known as inverse links. For example, we can have a HasMother link which is the inverse of MotherOf link.
- The inverse links make the job of inference algorithms much easier to answer queries such as who the mother of John is.

EX:- TOM IS A CAT.
CATS LOVE MILK.
TOM IS GREY IN COLOUR.



(c)Consider the following sentences:

1.Rita likes all kinds of food.

2.Apples are food.

3.Anything anyone eats and isn’t killed by is food.

4.Rahi eats peanuts and is still alive.

5.Tanvi eats everything Rahi eats.

i. Translate these sentences into formulas in predicate logic.

ii. Use resolution to answer the question, “What food does Tanvi eat?” 07

ANSWER:

(a) Translate these sentences into formulas in FOPC.

1.Rita likes all kinds of food.

->$\forall x: \text{Food}(x) \rightarrow \text{Likes}(\text{Rita}, x)$

2.Apples are food.

->$\text{Food}(\text{Apples})$

3.Anything anyone eats and isn't killed by is food.

->$\forall x \exists y\ \text{Eats}(y, x) \wedge \neg \text{KilledBy}(y, x) \rightarrow \text{Food}(x)$

4.Rahi eats peanuts and is still alive.

->$\text{Eats}(\text{Bill}, \text{Peanuts}) \wedge \neg \text{KilledBy}(\text{Bill}, \text{Peanuts})$

5.Tanvi eats everything Rahi eats.

->$\forall x\ \text{Eats}(\text{Rahi}, x) \rightarrow \text{Eats}(\text{Tanvi}, x)$

(b)Use resolution to answer the question, "What food does Tanvi eat?"

1.$\neg \text{Eats}(\text{Tanvi}, x)$ negated query

2.$\neg \text{Eats}(\text{Rahi}, x) \vee \text{Eats}(\text{Tanvi}, x)$

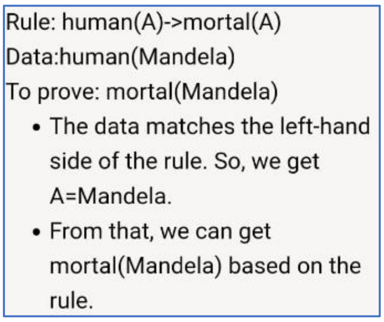
3.$\perp$ $\text{Eats}(\text{Rahi}, \text{Peanuts})$ $\neg \text{Eats}(\text{Rahi}, x) \vee \text{Eats}(\text{Tanvi}, x)$, unifying Peanuts/x

OR

(c)Explain Forward Reasoning and Backward Reasoning with example. 07

Answer: Forward Reasoning:

- Forward chaining starts from known facts and applies inference rule to extract more data unit it reaches to the goal.
- The Forward chaining algorithm starts from known facts, triggers all rules whose premises are satisfied, and add their conclusion to the known facts. This process repeats until the problem is solved.
- It is a bottom-up approach.
- Forward chaining is known as data-driven inference technique as we reach to the goal using the available data.
- Forward chaining is suitable for the planning, monitoring, control, and interpretation application.
- This technique is used by design expert system.



Rule: human(A)->mortal(A)

Data:human(Mandela)

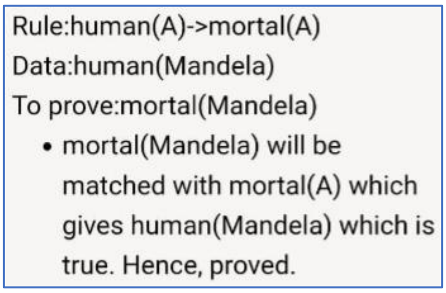
To prove: mortal(Mandela)

- The data matches the left-hand side of the rule. So, we get A=Mandela.
- From that, we can get mortal(Mandela) based on the rule.

Backward Reasoning:

- Backward chaining starts from the goal and works backward through inference rules to find the required facts that support the goal.
- A backward chaining algorithm is a form of reasoning, which starts with the goal and works backward, chaining through rules to find known facts that support the goal.
- It is a top-down approach.

- Backward chaining is known as goal-driven technique as we start from the goal and divide into sub-goal to extract the facts.
- Backward chaining is suitable for diagnostic, prescription, and debugging application.
- This technique is used by diagnostic expert system.



Rule:human(A)->mortal(A)
Data:human(Mandela)
To prove:mortal(Mandela)

- mortal(Mandela) will be matched with mortal(A) which gives human(Mandela) which is true. Hence, proved.

Q.3

(a)Explain Expert System Shell with example. 03

Answer: Expert System Shells:

An Expert system shell is a software development environment. It contains the basic components of expert systems. A shell is associated with a prescribed method for building applications by configuring and instantiating these components.

Shell components and description:

The generic components of a shell: the knowledge acquisition, the knowledge Base, the reasoning, the explanation and the user interface. The knowledge base and reasoning engine are the core components.

Example :

TMYCIN is an expert system shell modelled after the EMYCIN shell that was developed at Standford. It is especially useful for student exercises, although real expert systems have been written using it. 'TMYCIN' is written in common Lisp and it is fairly small (11 pages of code). It should run under any implementation of common Lisp.

(b)Explain MiniMax search procedure. 04

- The minimax search is a depth first and depth limited procedure.
- The idea is to start at the current position and use the plausible-move generator to generate the set of possible successor positions.
- Now we can apply the static evolution function to those positions and simply choose the best one.
- After doing so, we can back that value up to the starting position to represent our evolution of it.
- Here we assume that static evolution function returns larger values to indicate good situations for us.
- So, our goal is to maximize the value of the static evaluation function of the next board position.
- The opponents' goal is to minimize the value of the static evaluation function.
- The alternation of maximizing and minimizing at alternate ply when evaluations are to be pushed back up corresponds to the opposing strategies of the two players is called MINIMAX.
- It is recursive procedure that depends on two procedures:
- ***MOVEGEN(position, player)—*** The plausible-move generator, which returns a list of nodes representing the moves that can be made by Player in Position.
- ***STATIC(position, player)—*** static evaluation function, which returns a number representing the goodness of Position from the standpoint of Player.

(c)Explain Artificial Neural Network. 07

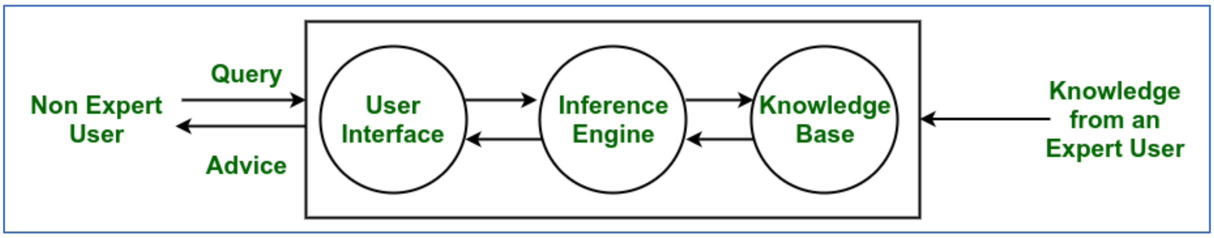
ANSWER:- Artificial Neural Network (ANN):-

- An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain process information.
- The brain is a highly complex, nonlinear, and parallel information processing system. It has the capability of organizing neurons so as to perform certain computations (e.g. pattern recognition, perception, and motor control) many times faster than the fastest digital computer. A brain has great structure and the ability to build up its own rules through what we usually refer to as experience.
- In its most general form, a neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest.
- The network is usually implemented using electronic components or simulated in software on a digital computer.
- To achieve good performance, neural networks employ a massive interconnection of simple computing cells referred to as neurons or processing units.
- A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use.
- It resembles the brain in two respects.
- Knowledge is acquired by the network through a learning process.
- Interneuron connection strengths known as synaptic weights are used to store the Knowledge.
- Neural networks are also referred to as neuro-computers, connectionist networks, parallel Distributed processors, etc.

OR

Q.3

(a)Draw and explain architecture of Expert System. 03



Architecture of an Expert System

- **Knowledge Base –**

The knowledge base represents facts and rules. It consists of knowledge in a particular domain as well as rules to solve a problem, procedures and intrinsic data relevant to the domain.

- **Inference Engine –**

The function of the inference engine is to fetch the relevant knowledge from the knowledge base, interpret it and to find a solution relevant to the user's problem. The inference engine acquires the rules from its knowledge base and applies them to the known facts to infer new facts. Inference engines can also include an explanation and debugging abilities.

- **Knowledge Acquisition and Learning Module –**

The function of this component is to allow the expert system to acquire more and more knowledge from various sources and store it in the knowledge base.

- **User Interface –**

This module makes it possible for a non-expert user to interact with the expert system and find a solution to the problem.

- **Explanation Module –**

This module helps the expert system to give the user an explanation about how the expert system reached a particular conclusion.

(b)Show the use of Alpha-Beta pruning. 04

Answer: Alpha-beta pruning:

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

->As we have seen in the minimax search algorithm that the number of game states it has to examine are exponential in depth of the tree. Since we cannot eliminate the exponent, but we can cut it to half.

->Hence there is a technique by which without checking each node of the game tree we can compute the correct minimax decision, and this technique is called **pruning**.

->This involves two threshold parameter Alpha and beta for future expansion, so it is called **alpha-beta pruning**. It is also called as **Alpha-Beta Algorithm**.

-> Alpha-beta pruning can be applied at any depth of a tree, and sometimes it not only prune the tree leaves but also entire sub-tree.

->The two-parameter can be defined as:

1. **Alpha:** The best (highest-value) choice we have found so far at any point along the path of Maximizer. The initial value of alpha is **-∞**.
2. **Beta:** The best (lowest-value) choice we have found so far at any point along the path of Minimizer. The initial value of beta is **+∞**.

->The Alpha-beta pruning to a standard minimax algorithm returns the same move as the standard algorithm does, but **it removes all the nodes which are not really affecting the final decision but making algorithm slow**.

->Hence by pruning these nodes, it makes the algorithm fast.

(c)Explain Backpropagation algorithm in Neural Network. 07

ANSWER: Backpropagation algorithm calculates the gradient of the error function.

- Backpropagation can be written as a function of the neural network.
- Backpropagation algorithms are a set of methods used to efficiently train artificial neural networks following a gradient descent approach which exploits the chain rule.
- The main features of Backpropagation are the iterative, recursive and efficient method through which it calculates the updated weight to improve the network until it is not able to perform the task for which it is being trained.

Two Types of Backpropagation Networks are:

- Static Back-propagation
- Recurrent Backpropagation

Static back-propagation:

- It is one kind of backpropagation network which produces a mapping of a static input for static output.
- It is useful to solve static classification issues like optical character recognition.

Recurrent Backpropagation:

- Recurrent Back propagation in data mining is fed forward until a fixed value is achieved.
- After that, the error is computed and propagated backward.

Advantages of Backpropagation are:

- Backpropagation is fast, simple and easy to program
- It has no parameters to tune apart from the numbers of input
- It is a flexible method as it does not require prior knowledge about the network
- It is a standard method that generally works well
- It does not need any special mention of the features of the function to be learned.

Q.4

(a)Briefly explain any one application of Natural Language Processing. 03

Answer: **Different applications of NLP are:**

- Sentiment Analysis
- Text Classification
- Chat bots & Virtual Assistants
- Information extraction
- Machine Translation
- Text Summarization
- Auto-Correct
- Speech Recognition

Text Classification:

->Text clarification is the process of categorizing the text into a group of words.

->By using NLP, text classification can automatically analyze text and then assign a set of predefined tags or categories based on its context.

->For e.g., Spam Detection is used to detect unwanted e-mails getting to a user's inbox.

(b)A bag I contains 4 white and 6 black balls while another Bag II contains 4 white and 3 black balls. One ball is drawn at random from one of the bags, and it is found to be black. Find the probability that it was drawn from Bag I. 04

ANSWER:

Let,
E1 = event of choosing the bag I
E2 = event of choosing the bag II
A = event of drawing a blackball
We have to Find P(E1/A) the probability of drawing Black ball from Bag I
Probability P(E1) = bag selected is bag **I = 1/2**
Probability P(E2) = bag selected is bag **II = 1/2**
Probability P(A|E1) = probability of drawing Black ball from Bag I
P(A|E1) = 6/10 = 3/5
Probability P(A|E2) = probability of drawing Black ball from Bag II
P(A|E2) = 3/7
Using Bayes Theorem:

$$P(E1|A) = \frac{P(A)\ P(A|E1)}{P(E1)\ P(A|E1) + P(E2)\ P(A|E1)}$$

$$= \frac{\frac{1}{2}*\frac{3}{5}}{\frac{1}{2}*\frac{3}{5}+\frac{1}{2}*\frac{3}{7}}$$

$$= \frac{\frac{3}{10}}{\frac{3}{10}+\frac{3}{14}}$$

$$= \frac{\frac{3}{10}}{\frac{21+15}{70}}$$

$$= \frac{\frac{3}{10}}{\frac{36}{70}}$$

$$= \frac{3}{10} * \frac{70}{36}$$

$$= \frac{7}{12}$$

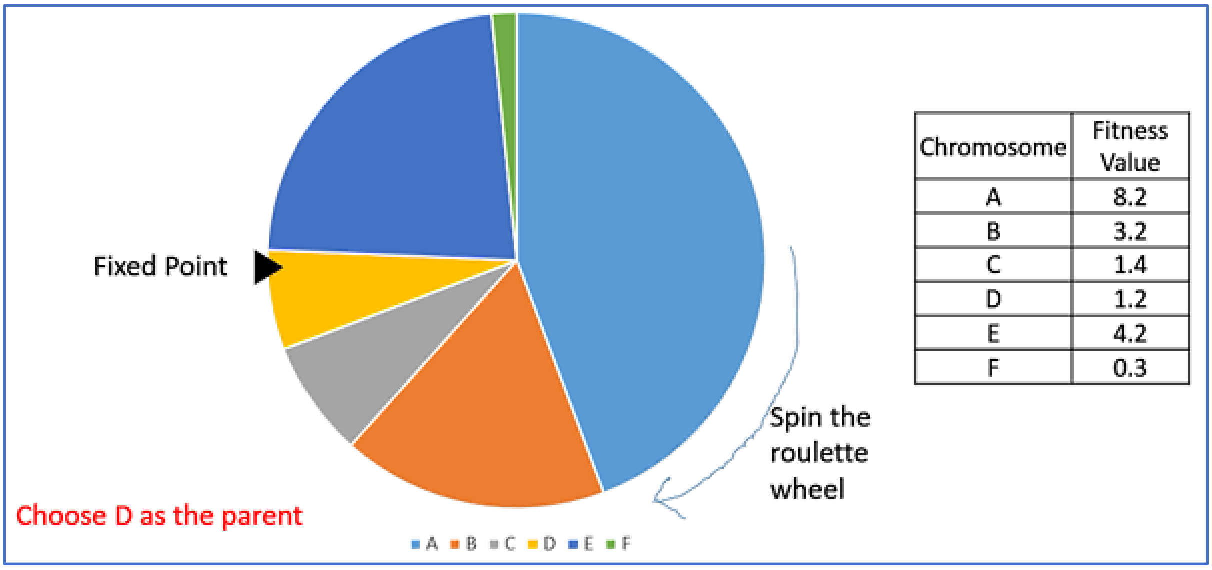
$$=\underline{0.583}$$

The probability of drawing Black ball from Bag I is 0.583

(c)Explain Roulette-Wheel selection method of genetic algorithm. 07

ANSWER: **Roulette Wheel Selection method:**

- Parent Selection in genetic algorithm is the process of selecting parents' which mate and recombine to create off-springs for the next generation.
- Fitness Proportionate Selection is one of the most popular ways of parent selection.
- In this every individual can become a parent with a probability which is proportional to its fitness.
- Roulette Wheel Selection method is one of the implementations of fitness proportionate selection.
- In a roulette wheel selection, the circular wheel is divided into n pies, where n is the number of individuals in the population.
- Each individual gets a portion of the circle which is proportional to its fitness value.
- A fixed point is chosen on the wheel circumference as shown and the wheel is rotated.
- The region of the wheel which comes in front of the fixed point is chosen as the parent.
- For the second parent, the same process is repeated.



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

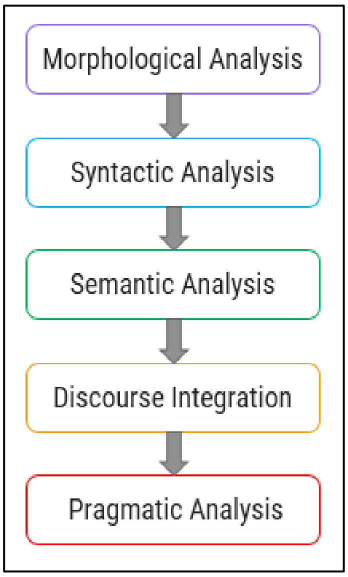
- It is clear that a fitter individual has a greater pie on the wheel and therefore a greater chance of landing in front of the fixed point when the wheel is rotated.
- Therefore, the probability of choosing an individual depends directly on its fitness.
- Implementation wise, we use the following steps –
    - Calculate S = the sum of a finesses.
    - Generate a random number between 0 and S.
    - Starting from the top of the population, keep adding the finesses to the partial sum P, till P<S.
    - The individual for which P exceeds S is the chosen individual.

OR

Q.4

(a)Enlist and describe different phases involved in Natural Language Processing. 03

ANSWER:



1. Morphological Analysis:
It involves identifying and analyzing the structure of words. It divides the whole text into paragraphs, sentences, and words.

2. Syntactic Analysis:
Syntactic Analysis is used to check grammar, word arrangements, and shows the relationship among the words. Linear sequence of words are transformed into structures that show how the words relate to each other.

3. Semantic Analysis:
Semantic analysis is concerned with the meaning representation. It mainly focuses on the literal meaning of words, phrases, and sentences.

4. Discourse Integration:
Discourse Integration is concerned with how immediately preceding sentence affects the interpretation of the next sentence. This depends upon the sentences that proceeds it and also invokes the meaning of the sentences that follow it.

5. Pragmatic Analysis:
Pragmatic is the fifth and last phase of NLP. It helps you to discover the intended effect by applying a set of rules that characterize cooperative dialogues.

For Example: "Open the door" is interpreted as a request instead of an order.

(b)The task is to recognize English alphabetical characters (F, E, X, Y, I, T) in an image processing system. Define two fuzzy sets I' and F' to represent the identification of characters I and F. 04

I'={(F, 0.4), (E, 0.3), (X, 0.1), (Y, 0.1), (I, 0.9), (T, 0.8)}

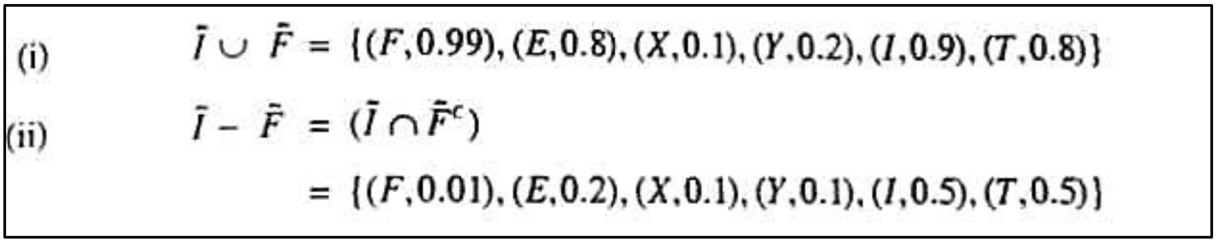
F'={(F, 0.99), (E, 0.8), (X, 0.1), (Y, 0.2), (I, 0.5), (T, 0.5)}

Find the following.

1. I' ∪ F'

2. I' – F'

Answer: Given below is the solution :



(i) $\tilde{I} \cup \tilde{F} = \{(F,0.99),(E,0.8),(X,0.1),(Y,0.2),(I,0.9),(T,0.8)\}$

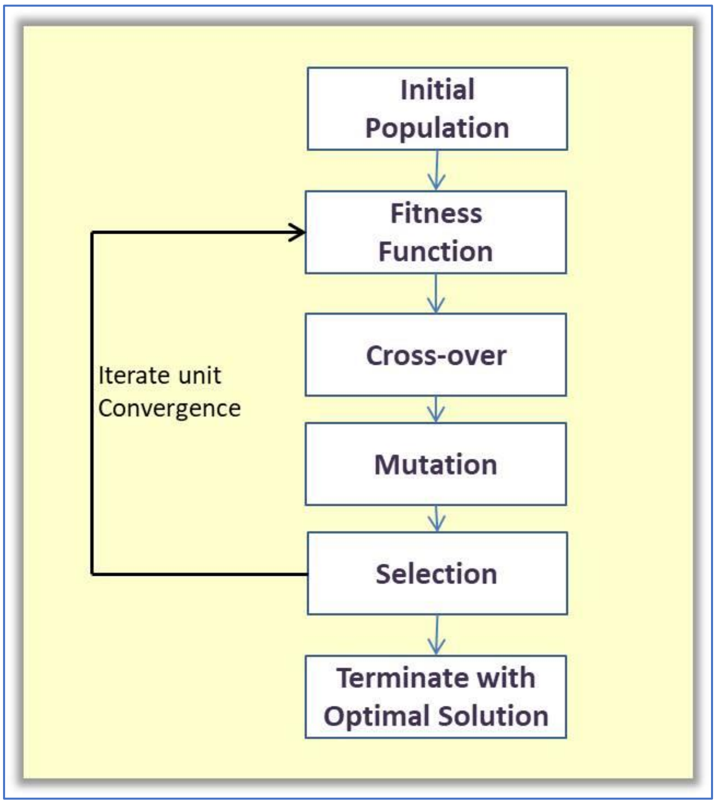
(ii) $\tilde{I} - \tilde{F} = (\tilde{I} \cap \tilde{F}^c)$

$= \{(F,0.01),(E,0.2),(X,0.1),(Y,0.1),(I,0.5),(T,0.5)\}$

(c)Describe the phases of genetic algorithm. 07

ANSWER:- Genetic algorithm basically involves five phases to solve the complex optimization problems, which are given as below:

- **Initialization**
- **Fitness Assignment**
- **Selection**
- **Reproduction**
- **Termination**



1.**Initial Population**

- This is the first phase of the process where an initial population is selected.
- The initial population is simply a set of individuals which are potential solution to the problem we want to solve.

- It could actually be made of thousands or even hundreds of thousands of individuals that may have been randomly selected or selected based on some specified criteria.

2.Fitness Function:

- This is a way of specifying the criteria for the 'goodness' of a given solution compared to others and to the problem being solved.
- Based on the fitness, members of the population are separated into two parts: those that are fit enough to be selected into the next generation and those that would be dropped.

3.Selection:

- Selection is simple the process of selecting the portion of the population that would pass into the next generation.
- The selection process is based on the fitness criteria from the preceding phase.
- Two pairs of individual members of the population are selected (like parents) based on their fitness.

4. Reproduction:

- In this step, the creation of a child occurs in the reproduction step.
- The two variation operators that are applied to the parent population in the reproduction phase are given below:

(a)Cross-Over: Cross-over is the most noteworthy & is also called recombination.

- It is a genetic operator that determines what information is passed from the two parents to the new offspring.
- Cross-over point is the point in within the genes of the parents at which the cross-over occurs.

(b)**Mutation**: The mutation operator inserts random genes in the offspring (new child) to maintain the diversity in the population.

- It can be done by flipping some bits in the chromosomes.
- Mutation helps in solving the issue of premature convergence and enhances diversification.

5. Termination:

- After the reproduction phase, a stopping criterion is applied as a base for termination.
- The algorithm terminates after the threshold fitness solution is reached.
- It will identify the final solution as the best solution in the population.

Q.5

(a)Explain planning problem. 03

ANSWER: - **Planning problem:**

- The planning problem in Artificial Intelligence is about sequence of actions that achieves a given goal when executed from a given initial world state.
- That is, given
  - a set of operator descriptions (defining the possible primitive actions by the agent),
  - an initial state description, and
  - a goal state description or predicate,
- Compute a plan, which is - a sequence of operator instances, such that executing them in the initial state will change the world to a state satisfying the goal-state description.
- Goals are usually specified as a conjunction of goals to be achieved

Planning:->Set of Actions.

Action:->Precondition:->For actions to performed there is need of some preconditions.
:->Effect-Effect is result of action performed.

(b)Explain limitations of Hill Climbing algorithm. 04

ANSWER:- **Limitations of Hill Climbing algorithm:** -
Hill climbing cannot reach the optimal/best state (global maximum) if it enters any of the following regions:
Local maximum: At a local maximum all neighbouring states have a value that is worse than the current state. Since hill-climbing uses a greedy approach, it will not move to the worse state and terminate itself. The process will end even though a better solution may exist.
To overcome the local maximum problem: Utilize the backtracking technique. Maintain a list of visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path.
Plateau: On the plateau, all neighbors have the same value. Hence, it is not possible to select the best direction.
To overcome plateaus: Make a big jump. Randomly select a state far away from the current state. Chances are that we will land in a non-plateau region.
Ridge: Any point on a ridge can look like a peak because movement in all possible directions is downward. Hence the algorithm stops when it reaches this state.
To overcome Ridge: In this kind of obstacle, use two or more rules before testing. It implies moving in several directions at once.

(c)Explain Cut and Fail predicates in Prolog. 07

ANSWER: **Fail:**

- If we want to force a rule to fail under certain conditions, then built in predicate called fail is used in Prolog.
- Predicate fail tells prolog interpreter to fail a particular goal and subsequently forces backtracking.
- All the sub goals defined after fail will never be executed.
- Hence, predicate fail, should always be used as the last sub goal in a rule.
- It is also to be noted, that rule containing fail predicate will not produce any solution.
- Ex- goal(x):- failure(x),!.goal(x).
- Failure(x) specifies the condition which makes goal(x) fail.

Cut:

- Sometimes it is desirable to selectively turn off backtracking
- Cut always succeeds but cannot be backtracked
- The cut effectively tells Prolog to freeze all the decisions made so far in this predicate. That is, if required to backtrack, it will automatically fail without trying other alternatives
- Performance is the main reason to use the cut.
- The Symbol of cut predicate is “!”.
- Ex- minimum(x,y,x):- x<=y,!.minimum(x,y,y):-x>y
- Cut is used to tell the interpreter to not to look for solutions when it finds one.

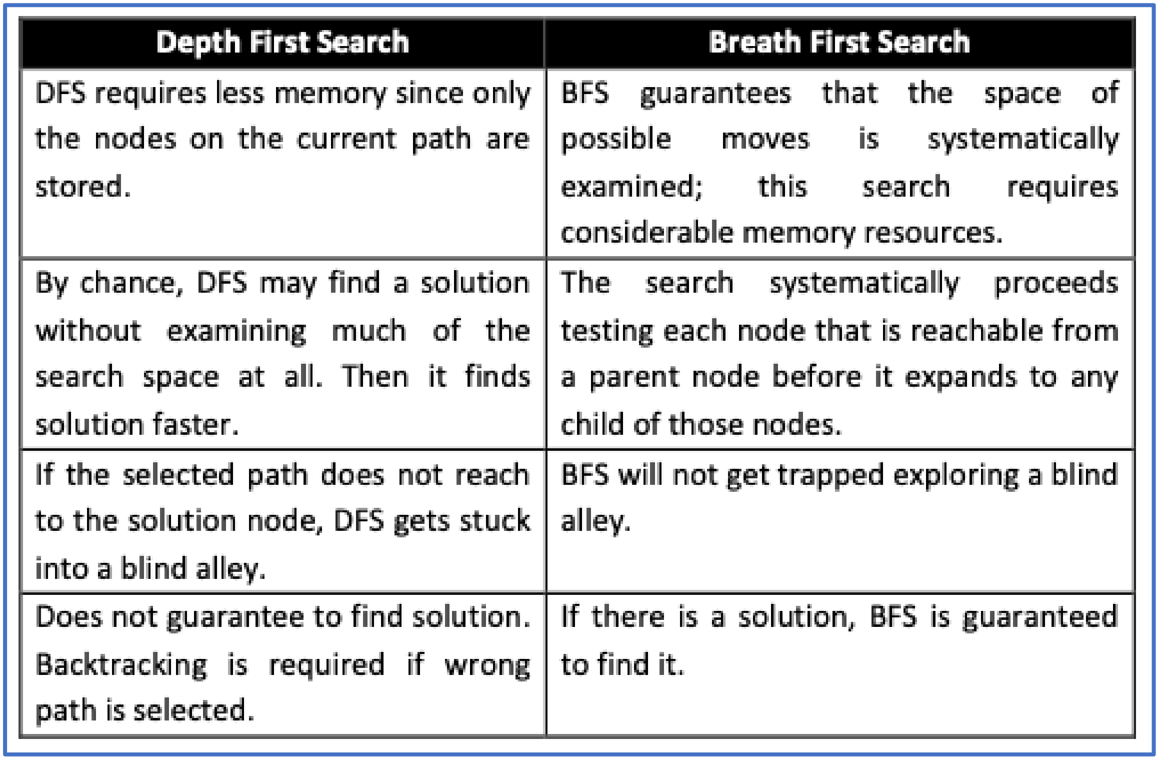
OR

Q.5

(a)Explain how planning is different from search procedure? 03

- Planning is the process of computing several steps of a problem-solving procedure before executing any of them
- This problem can be solved by search
- 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). Whereas in planning, states have structured representations (collections of properties) that are used by the planning algorithm.
- Planning only do predictions and just tells us the outcomes but searching need coding that's why it has 100 actions.

(b)Compare DFS and BFS. 04



Depth First Search	Breath First Search
DFS requires less memory since only the nodes on the current path are stored.	BFS guarantees that the space of possible moves is systematically examined; this search requires considerable memory resources.
By chance, DFS may find a solution without examining much of the search space at all. Then it finds solution faster.	The search systematically proceeds testing each node that is reachable from a parent node before it expands to any child of those nodes.
If the selected path does not reach to the solution node, DFS gets stuck into a blind alley.	BFS will not get trapped exploring a blind alley.
Does not guarantee to find solution. Backtracking is required if wrong path is selected.	If there is a solution, BFS is guaranteed to find it.

(c)Write a Prolog program to merge two sequentially ordered (ascending) lists into one ordered list. 07

ANSWER:-

```
merge([],[],[]).
   merge([X],[],[X]).
   merge([],[Y], [Y]).
   merge([X|List1],[Y|List2],[X|List]) :-
      X <= Y,!,
      merge(List1, [Y|List2], List).
merge([X]List1],[Y|List2],[Y|List]) :-
      merge([X|List1], List2,List).
```

BE - SEMESTER–VII (NEW)
EXAMINATION – WINTER 2021
Subject Code:3170716
Date:13/12/2021
Subject Name: Artificial Intelligence

Q.1
(a) What is a "control strategy" and what are its characteristics? 03

Ans: Control Strategy in Artificial Intelligence scenario is a technique or strategy, tells us about which rule has to be applied next while searching for the solution of a problem within problem space. It helps us to decide which rule has to apply next without getting stuck at any point.

Characteristics:

1.A good control strategy should cause motion.
2.The second requirement is that it should be systematic.
3.Finally, it should be efficient in order to find a good answer.

(b) Describe in brief how Min-Max search procedure works.

Ans: The minimax search is a depth first and depth limited procedure.

- The idea is to start at the current position and use the plausible-move generator to generate the set of possible successor positions.
- Now we can apply the static evolution function to those positions and simply choose the best one.
- After doing so, we can back that value up to the starting position to represent our evolution of it.
- Here we assume that static evolution function returns larger values to indicate good situations for us.
- So, our goal is to maximize the value of the static evaluation function of the next board position.
- The opponents' goal is to minimize the value of the static evaluation function.
- The alternation of maximizing and minimizing at alternate ply when evaluations are to be pushed back up corresponds to the opposing strategies of the two players is called MINIMAX.
- It is recursive procedure that depends on two procedures:
- ***MOVEGEN(position, player)***— The plausible-move generator, which returns a list of nodes representing the moves that can be made by Player in Position.
- ***STATIC(position, player)***— static evaluation function, which returns a number representing the goodness of Position from the standpoint of Player.

(c) Enlist and discuss major task domains of Artificial Intelligence.

ANS: Task Domains of Artificial Intelligence (AI)

1.Mundane Tasks:

- Perception
- Vision
- Speech
- Natural Languages
- Understanding
- Generation
- Translation
- Common sense reasoning
- Robot Control

- ➢ Humans have been learning mundane (ordinary) tasks since their birth. They learn by perception, speaking, using language, and locomotives.
- ➢ For humans, the mundane tasks are easiest to learn. The same was considered true before trying to implement mundane tasks in machines.
- ➢ Earlier, all work of AI was concentrated in the mundane task domain.

2.Formal Tasks:

- Games: chess, checkers, etc
- Mathematics: Geometry, logic, Proving properties of programs

- ➢ Formal Tasks are the tasks that need deals with verification, theorem proving, deals with Math, Games, etc.

3.Expert Tasks:

- Engineering (Design, Fault finding, Manufacturing planning)
- Scientific Analysis
- Medical Diagnosis
- Financial Analysis

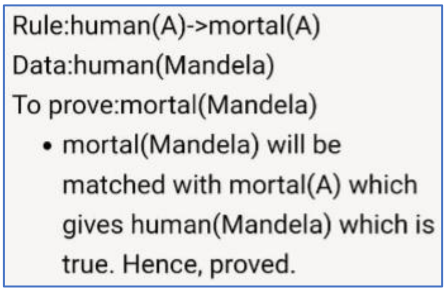
- ➢ Expert Tasks are those tasks, which involves scientific analysis, and different domain analysis, like Financial, healthcare, creative aspects, etc.
- ➢ Now researchers have understood that to solve mundane tasks, they need better and more efficient algorithms, and a much more knowledge base to help them tackle the problems they have set out to solve. And that is the reason that AI has shifted more on working with Expert Tasks, to enhance the capabilities of the AI system.

Q.2

(a) Discuss briefly backward reasoning with example. 0 3

ANS: Backward Reasoning:

- Backward chaining starts from the goal and works backward through inference rules to find the required facts that support the goal.
- A backward chaining algorithm is a form of reasoning, which starts with the goal and works backward, chaining through rules to find known facts that support the goal.
- It is a top-down approach.
- Backward chaining is known as goal-driven technique as we start from the goal and divide into sub-goal to extract the facts.
- Backward chaining is suitable for diagnostic, prescription, and debugging application.
- This technique is used by diagnostic expert system.



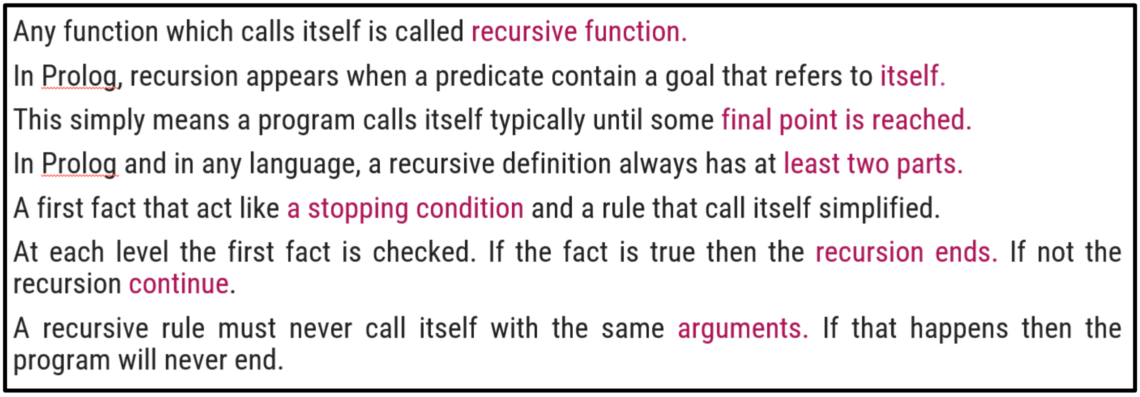
Rule:human(A)->mortal(A)

Data:human(Mandela)

To prove:mortal(Mandela)

- mortal(Mandela) will be matched with mortal(A) which gives human(Mandela) which is true. Hence, proved.

(b) Explain with example how recursive predicate is defined in Prolog? 04



Any function which calls itself is called recursive function.

In Prolog, recursion appears when a predicate contain a goal that refers to itself.

This simply means a program calls itself typically until some final point is reached.

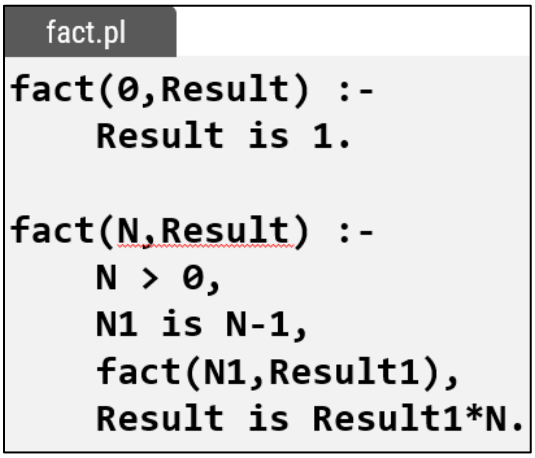
In Prolog and in any language, a recursive definition always has at least two parts.

A first fact that act like a stopping condition and a rule that call itself simplified.

At each level the first fact is checked. If the fact is true then the recursion ends. If not the recursion continue.

A recursive rule must never call itself with the same arguments. If that happens then the program will never end.

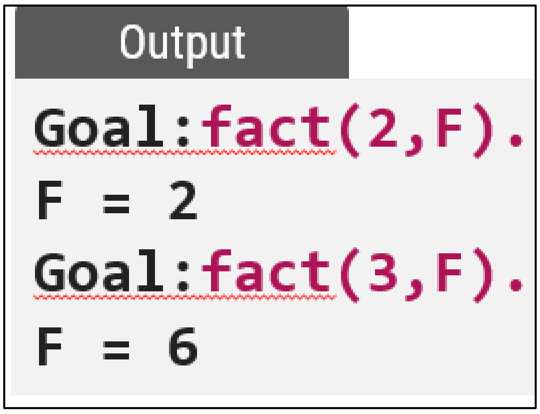
Find Factorial of a given no.



fact.pl

```
fact(0,Result) :-
    Result is 1.

fact(N,Result) :-
    N > 0,
    N1 is N-1,
    fact(N1,Result1),
    Result is Result1*N.
```



Output

```
Goal:fact(2,F).
F = 2
Goal:fact(3,F).
F = 6
```

(c) State Water Jug problem. Give its state space representation

Problem Definition: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

1.Initial State:

- We will represent a state of the problem as a tuple (x, y), where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-gallon jug.
- Note that 0 ≤ x ≤ 4, and 0 ≤ y ≤ 3.
- Here the initial state is (0, 0). The goal state is (2, n) for any value of n.

2.Production Rules

Sr.	Current state	Next State	Descriptions
1	(x, y) if x < 4	(4,y)	Fill the 4 gallon jug
2	(x, y) if y <3	(x,3)	Fill the 3 gallon jug
3	(x, y) if x > 0	(x-d, y)	Pour some water out of the 4 gallon jug
4	(x, y) if y > 0	(x, y-d)	Pour some water out of the 3 gallon jug
5	(x, y) if x>0	(0, y)	Empty the 4 gallon jug
6	(x, y) if y >0	(x,0)	Empty the 3 gallon jug on the ground
7	(x, y) if x+y >= 4 and y > 0	(4, y-(4-x))	Pour water from the 3 gallon jug into the 4 gallon jug until the 4 gallon jug is full
8	(x, y) if x+y >= 3 and x>0	(x-(3-y), 3)	Pour water from the 4 gallon jug into the 3-gallon jug until the 3 gallon jug is full
9	(x, y) if x+y <=4 and y>0	(x+y, 0)	Pour all the water from the 3 gallon jug into the 4 gallon jug
10	(x, y) if x+y <= 3 and x>0	(0, x+y)	Pour all the water from the 4 gallon jug into the 3 gallon jug
11	(0,2)	(2,0)	Pour the 2 gallons from 3 gallon jug into the 4 gallon jug
12	(2,y)	(0,y)	Empty the 2 gallons in the 4 gallon jug on the ground

One of the possible solutions is given as:

3. Productions for the water jug problem

Gallons in the 4- gallon jug	Gallons in the 3- gallon jug	Rule applied
0	0	2
0	3	9
3	0	2
3	3	7
4	2	5 or 12
0	2	9 0r 11
2	0	--

OR

(c) Explain A* algorithm. What happens if h' underestimates h and overestimates h? 07

Ans: A* algorithm uses following functions:

a) f': Heuristic function that estimates the merits of each node we generate. f' represents an estimate of the cost of getting from the initial state to a goal state along with the path that generated the current node. f' = g + h'

b) g: The function g is a measure of the cost of getting from initial state to the current node.

c) h': The function h' is an estimate of the additional cost of getting from the current node to a goal state.

The algorithm also uses the lists: OPEN and CLOSED
d) **OPEN -** nodes that have been generated and have had the heuristic function applied to them but which have not yet been examined. OPEN is actually a priority queue in which the elements with the highest priority are those with the most promising value of the heuristic function.
e) **CLOSED -** nodes that have already been examined. We need to keep these nodes in memory if we want to search a graph rather than a tree, since whenever a new node is generated; we need to check whether it has been generated before.

Step 1:

➥ Start with OPEN containing only initial node.

➥ Set that node's g value to 0, its h' value to whatever it is, and its f' value to h'+0 or h'.

➥ Set CLOSED to empty list.

Step 2: Until a goal node is found, repeat the following procedure:

➥ If there are no nodes on OPEN, report failure.
➥ Otherwise select the node on OPEN with the lowest f' value.
➥ Call it BESTNODE. Remove it from OPEN. Place it in CLOSED.
➥ See if the BESTNODE is a goal state. If so exit and report a solution.
➥ Otherwise, generate the successors of BESTNODE but do not set the BESTNODE to point to them yet.

Step 3: For each of the SUCCESSOR, do the following:
a. Set SUCCESSOR to point back to BESTNODE. These backwards links will make it possible to recover the path once a solution is found.
b. Compute g(SUCCESSOR) = g(BESTNODE) + the cost of getting from BESTNODE to SUCCESSOR
c. See if SUCCESSOR is the same as any node on OPEN. If so call the node OLD.
(i) Check whether it is cheaper to get to OLD via its current parent or to SUCESSOR via BESTNODE by comparing their g values.
(ii) If OLD is cheaper, then do nothing. If SUCCESSOR is cheaper then reset OLD's parent link to point to BESTNODE.
(iii) Record the new cheaper path in g(OLD) and update f'(OLD).
(iv) If SUCCESSOR was not on OPEN, see if it is on CLOSED. If so, call the node on CLOSED OLD and add OLD to the list of BESTNODE's successors.
(v) If SUCCESSOR was not already on either OPEN or CLOSED, then put it on OPEN and add it to the list of BESTNODE's successors.
Compute f'(SUCCESSOR) = g(SUCCESSOR) + h'(SUCCESSOR)

Q.3

(a) Explain local maxima, plateau and ridge in brief 03

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

To overcome the local maximum problem: Utilize the backtracking technique. Maintain a list of visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path.

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

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

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

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

(b) Discuss the approaches to knowledge representation.

Ans: **Different approaches to knowledge representation:**

- **Simple Relational Knowledge**

This is a relational method of storing facts which is among the simplest of the method. This method helps in storing facts where each fact regarding an object is providing in columns. This approach is prevalent in DBMS (database management systems).

- **Inheritable Knowledge**

Knowledge here is stored hierarchically. A well-structured hierarchy of classes is formed where data is stored, which provides the opportunity for inference. Here we can apply inheritance property, allowing us to have inheritable knowledge. This way, the relations between instance and class (aka instance relation) can be identified. Unlike Simple Relations, here, the objects are represented as nodes.

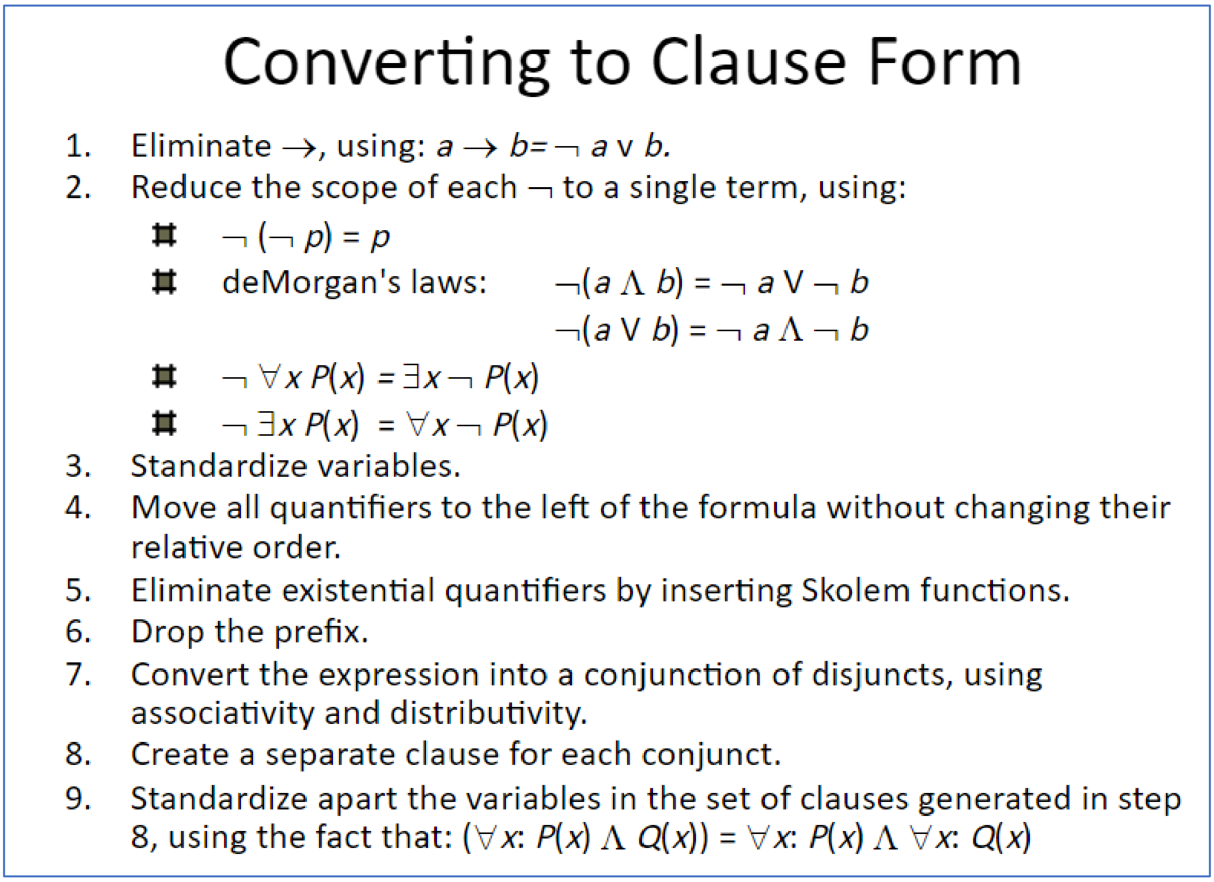
- **Inferential Knowledge**

In this method, logics are used. Being a very formal approach, facts can be retrieved with a high level of accuracy.

- **Procedural Knowledge**

This method uses programs and codes that use simple if-then rules. This is the way many programming languages such as LIST, Prolog save information. We may not use this method to represent all forms of knowledge, but domain-specific knowledge can very efficiently be stored in this manner.

(c)Demonstrate briefly steps to convert given wff into clause form 07



Converting to Clause Form

1. Eliminate $\rightarrow$, using: $a \rightarrow b = \neg a \vee b$.
2. Reduce the scope of each $\neg$ to a single term, using:
   - $\neg(\neg p) = p$
   - deMorgan's laws: $\neg(a \wedge b) = \neg a \vee \neg b$
$\neg(a \vee b) = \neg a \wedge \neg b$
   - $\neg \forall x\, P(x) = \exists x \neg P(x)$
   - $\neg \exists x\, P(x) = \forall x \neg P(x)$
3. Standardize variables.
4. Move all quantifiers to the left of the formula without changing their relative order.
5. Eliminate existential quantifiers by inserting Skolem functions.
6. Drop the prefix.
7. Convert the expression into a conjunction of disjuncts, using associativity and distributivity.
8. Create a separate clause for each conjunct.
9. Standardize apart the variables in the set of clauses generated in step 8, using the fact that: $(\forall x: P(x) \wedge Q(x)) = \forall x: P(x) \wedge \forall x: Q(x)$

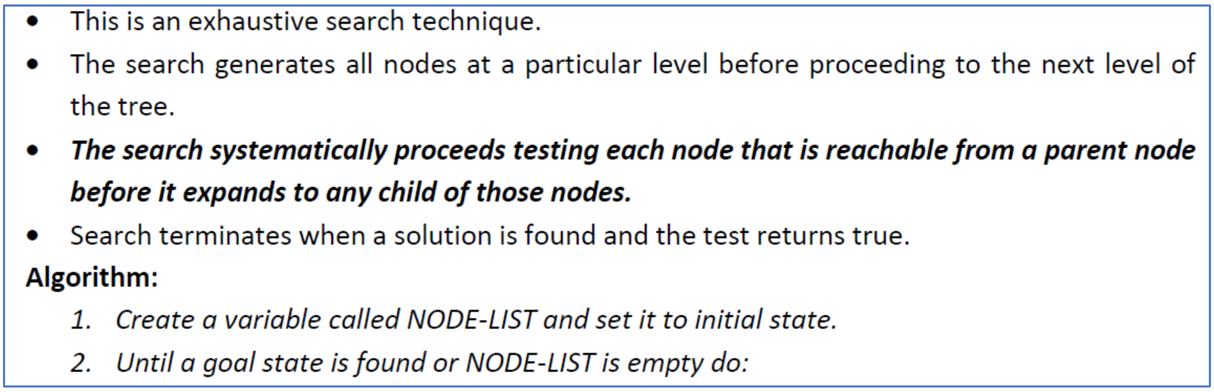
OR

Q.3

(a) Differentiate Informed & Uninformed search. Give examples. 03

Informed Search	**Uninformed Search**
It is also known as Heuristic Search.	It is also known as Blind Search.
It uses knowledge for the searching process.	It doesn't use knowledge for the searching process.
It finds a solution more quickly.	It finds solution slow as compared to an informed search.
It may or may not be complete.	It is always complete.
Cost is low.	Cost is high.
It consumes less time because of quick searching.	It consumes moderate time because of slow searching.
There is a direction given about the solution.	No suggestion is given regarding the solution in it.
It is less lengthy while implemented.	It is more lengthy while implemented.
It is more efficient as efficiency takes into account cost and performance. The incurred cost is less and speed of finding solutions is quick.	It is comparatively less efficient as incurred cost is more and the speed of finding the Breadth-First solution is slow.
Computational requirements are lessened.	Comparatively higher computational requirements.
Having a wide scope in terms of handling large search problems.	Solving a massive search task is challenging.
Examples: Greedy Search, A* Search, AO* Search, Hill Climbing Algorithm	Examples: Depth First Search (DFS), Breadth First Search (BFS), Branch and Bound

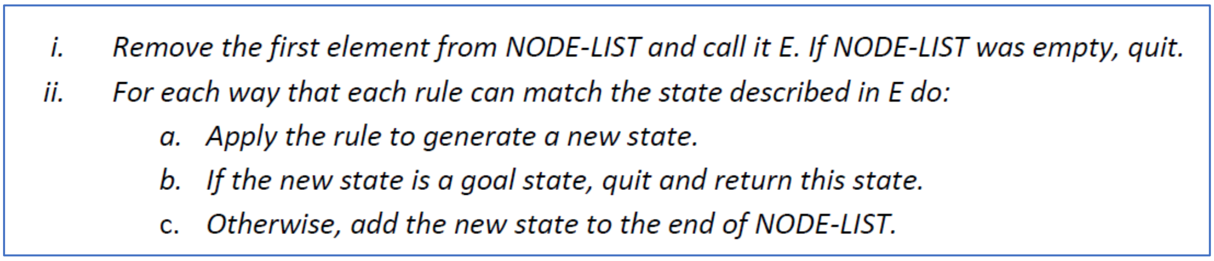
(b) Explain best first search algorithm. 04



- This is an exhaustive search technique.
- The search generates all nodes at a particular level before proceeding to the next level of the tree.
- ***The search systematically proceeds testing each node that is reachable from a parent node before it expands to any child of those nodes.***
- Search terminates when a solution is found and the test returns true.

Algorithm:

```
1. Create a variable called NODE-LIST and set it to initial state.
2. Until a goal state is found or NODE-LIST is empty do:
```



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

(c) Consider the following facts:

* Hemant only likes easy courses.
* Science courses are hard.
* All the courses in basketweaving department are easy.
* BK301 is a basketweaving course.

Use resolution to answer the question,” What course would Hemant like?”

ANS:

First, we will convert it into FOPL (First order predicate logic)

i."Hemant only likes easy courses.

- ➢ ∀x: easy(x) -> likes (Hemant, x)

ii. Science courses are hard.

- ➢ ∀x: science(x) -> ~easy(x)

iii. All the courses in basketweaving Department are easy.

- ➢ ∀x: basketweaving department (x) -> easy(x)

iv. BK301 is a basketweaving course ".

- ➢ basketweaving course (BK301)

❖ **The conclusion is encoded as likes(Hemant, x).**

• First we put our premises in the clause form and the negation of conclusion to our set of clauses (we use numbers in parentheses to number the clauses):

(1) ~easy(x) v likes(hemant,x)
(2) ~science(x) v ~easy(x)
(3) ~basketweaving course (x) v easy(x)
(4) basketweaving course (BK301)
(5) ~likes(hemant,x)

No.	Clauses	Notes
1	~easy(x) v likes(hemant,x)	P
2	~science(x) v ~easy(x)	P
3	~basketweaving course (x) v easy(x)	P
4	basketweaving course (BK301)	P
5	~ likes(hemant,x)	P
6	~easy(x)	1 & 5

7	~basketweaving course (x)	3 & 6
8	NIL	4&7, x = BK301

A resolution proof may be obtained by the following sequence of resolutions
(6) 1&5 yields resolvent ~easy(x).
(7) 3&6 yields resolvent ~basketweaving (x).
(8) 4&7 yields empty clause; the substitution x/BK301 is produced by the unification algorithm which says that the only wff of the form likes (hemant,x) which follows from the premises is likes(hemant, BK301). Thus, resolution gives us a way to find additional assumptions.

Q.4

(a) What is certainty factor? 03

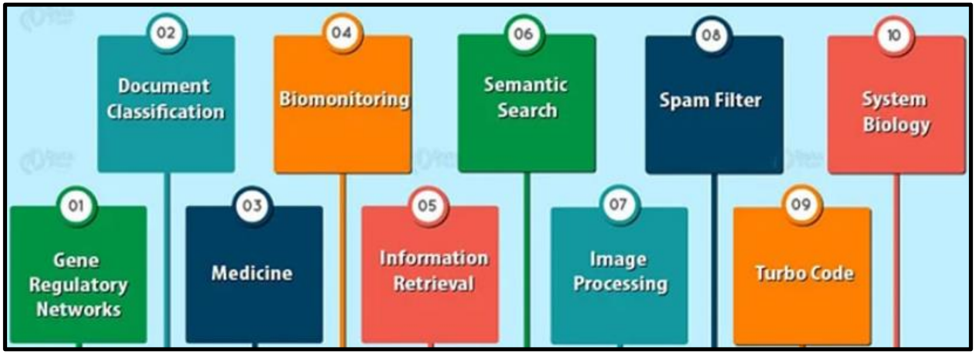
- **Certainty factor:** A certainty factor (CF) is a numerical value that expresses a degree of subjective belief that a particular item is true.
- Suppose that a certainty is defined to be a real number between -1.0 and +1.0, where 1.0 represents complete certainty that an item is true and -1.0 represents complete certainty that an item is false.
- Here a CF of 0.0 indicates that no information is available about either the truth or the falsity of an item.
- For example:
- IF has-spots(X) AND has-fever(X) THEN has-measles(X) CF 0.5
- Suppose, the certainty of has_spots (Raj) is 0.3, has_fever(Raj) is 0.8 and the certainty of the conjoined premise is min(0.3,0.8)=0.3.
- So, the certainty of conclusion will be 0.3*0.5-0.15.

(b) Discuss Bayesian network and its application. 04

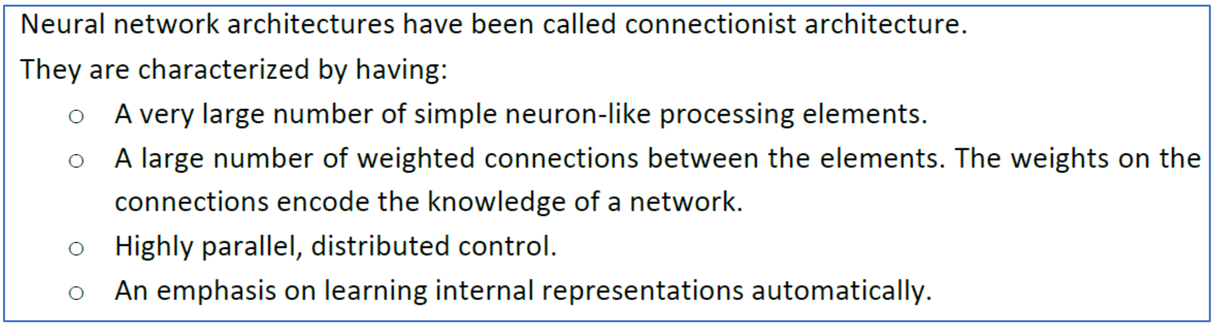
ANS: **Bayesian network**

- A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph.
- In a directed acyclic graph, each edge corresponds to a conditional dependency, and each node corresponds to a unique random variable.
- It is also called a Bayes network, belief network, decision network, or Bayesian model.
- Bayesian Network represents the dependency among events and assigning probabilities to them.
- Thus, ascertaining how probable or what is the change of occurrence of one event given the other.
- It can be used in various tasks including prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction, and decision making under uncertainty.

Applications of Bayesian N/W:



(c) Explain connectionist models. What is perceptron? What is the concept of back propagation for ANNs? 07



Neural network architectures have been called connectionist architecture.

They are characterized by having:

- A very large number of simple neuron-like processing elements.
- A large number of weighted connections between the elements. The weights on the connections encode the knowledge of a network.
- Highly parallel, distributed control.
- An emphasis on learning internal representations automatically.

- **The perceptron**, an invention of (1962) Rosenblatt, was one of the earliest neural network models.
  - It models a neuron by taking a weighted sum of its inputs and sending the output 1 if the sum is greater than some adjustable threshold value (otherwise it sends 0)
  - The perceptron model is also known as a single-layer neural network. This neural net contains only two layers: Input Layer and Output Layer
  - In this type of neural network, there are no hidden layers. It takes an input and calculates the weighted input for each node. Afterward, it uses an activation function (mostly a sigmoid function) for classification purposes.
  - As being supervised in nature, to calculate the error, there would be a comparison between the desired/target output and the actual output. If there is any difference found, then a change must be made to the weights of connection.

- **Backpropagation** in neural network is a short form for "backward propagation of errors."
  - It is a standard method of training artificial neural networks.
  - This method helps calculate the gradient of a loss function with respect to all the weights in the network.

OR

Q.4

(a) What do you mean by Expert Systems? List out its four applications. 03

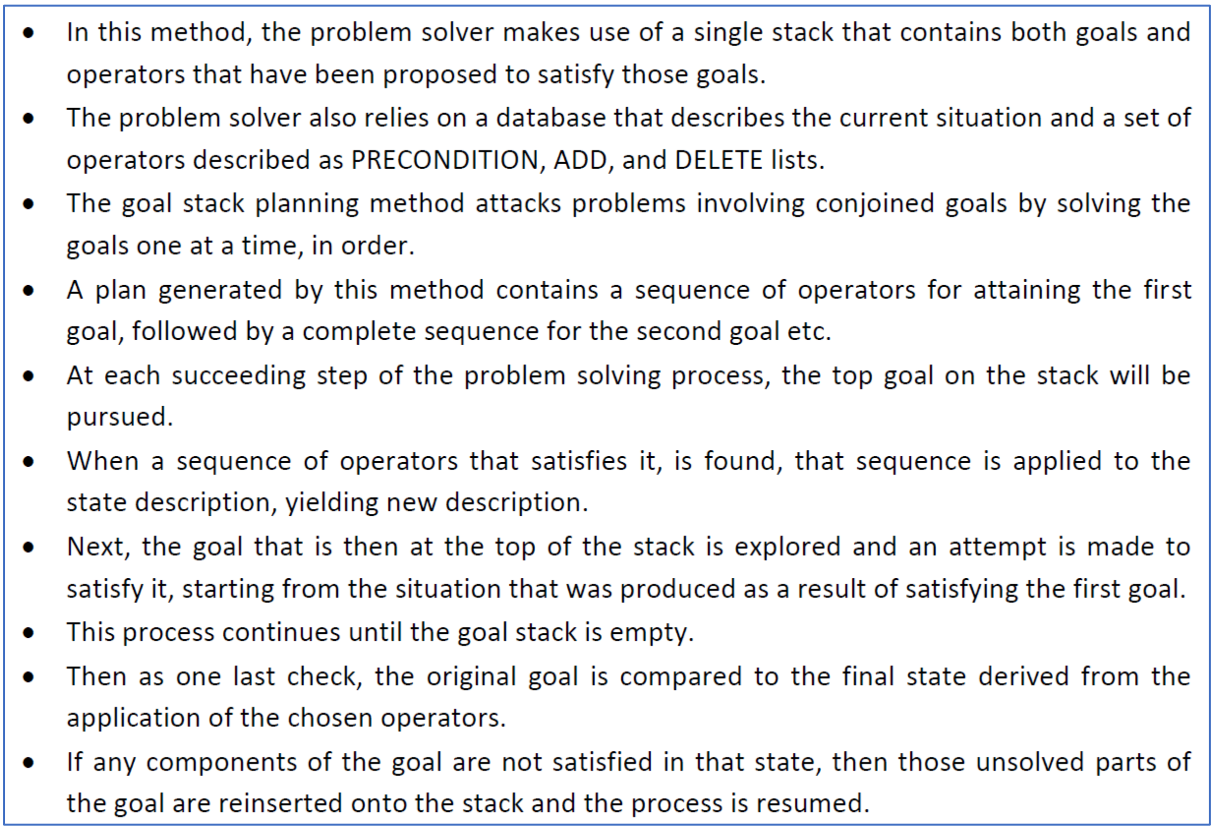
Ans: Expert System is an interactive and reliable computer-based decision-making system which uses both facts and heuristics to solve complex decision-making problems.

- It is considered at the highest level of human intelligence and expertise.
- The purpose of an expert system is to solve the most complex issues in a specific domain.
- The Expert System in AI can resolve many issues which generally would require a human expert.
- It is based on knowledge acquired from an expert.

Some popular Application of Expert System:

- Information management
- Hospitals and medical facilities
- Help desks management
- Employee performance evaluation
- Loan analysis
- Virus detection
- Useful for repair and maintenance projects
- Warehouse optimization
- Planning and scheduling
- The configuration of manufactured objects
- Financial decision-making Knowledge publishing
- Process monitoring and control
- Supervise the operation of the plant and controller
- Stock market trading
- Airline scheduling & cargo schedules

(b) Discuss Goal Stack planning 04



- In this method, the problem solver makes use of a single stack that contains both goals and operators that have been proposed to satisfy those goals.
- The problem solver also relies on a database that describes the current situation and a set of operators described as PRECONDITION, ADD, and DELETE lists.
- The goal stack planning method attacks problems involving conjoined goals by solving the goals one at a time, in order.
- A plan generated by this method contains a sequence of operators for attaining the first goal, followed by a complete sequence for the second goal etc.
- At each succeeding step of the problem solving process, the top goal on the stack will be pursued.
- When a sequence of operators that satisfies it, is found, that sequence is applied to the state description, yielding new description.
- Next, the goal that is then at the top of the stack is explored and an attempt is made to satisfy it, starting from the situation that was produced as a result of satisfying the first goal.
- This process continues until the goal stack is empty.
- Then as one last check, the original goal is compared to the final state derived from the application of the chosen operators.
- If any components of the goal are not satisfied in that state, then those unsolved parts of the goal are reinserted onto the stack and the process is resumed.

(c) Write about various defuzzification methods 07

ANS: Defuzzification:

Defuzzification is a method for the conversion of the fuzzy set (fuzzy output) to the crisp set or crisp output.

Methods of defuzzification are given below:

1.Center of sums Method.

2.Center of Gravity Method.

3.Center of Area Method.

4.Weighted Average Method.

5.Maxima Methods: First of Maxima (FOM), Last of Maxima (LOM), Mean of Maxima (MOM)

1.Center of sums Method.

- Center of sums is the fastest defuzzification method. This process involves the algebraic sum of individual output fuzzy sets. The only drawback of this method is that intersecting areas are added twice.

2.Center of Gravity Method.

- The basic principle in CoG method is to find the point x∗ where a vertical line would slice the aggregate into two equal masses. This method returns a precise value depending on the fuzzy set's center of gravity. The only drawback of this method is that it is Computationally intensive

3.Center of Area Method.

- Center of Largest Area (CoA) method is simple, computationally effective and widely used defuzzification.
- If the fuzzy set has two sub regions, then the center of gravity of the sub region with the largest area can be used to calculate the defuzzified value.

4.Weighted Average Method.

- Weighted average method is one of the simplest and widely used defuzzification technique, alternatively called as "Sugeno defuzzification" method.
- Formed by weighting each function in the output by its respective maximum membership value.
- This method is applicable to fuzzy sets with symmetrical output membership functions and yields results that are quite similar to the COA method. This method requires less computational power.

5.Maxima Methods: First of Maxima (FOM), Last of Maxima (LOM), Mean of Maxima (MOM)

- Maxima methods are quite simple but not as trivial as lambda cut methods. Maxima methods relies on the position of maximum membership of element at particular position in fuzzy set.
- The set of methods under maxima methods we will be discussing here are:
- *First of maxima (FoM): Determine the smallest value of the domain with maximized membership degree*
- *Last of maxima (LoM): Determine the largest value of the domain with maximized membership degree*

Mean of maxima (MoM): In order to find middle of maxima, we have to find the "middle" of elements with maximum membership value. This method is applicable to symmetric functions only.

Q.5

(a) Discuss cut and fail predicate in prolog 03

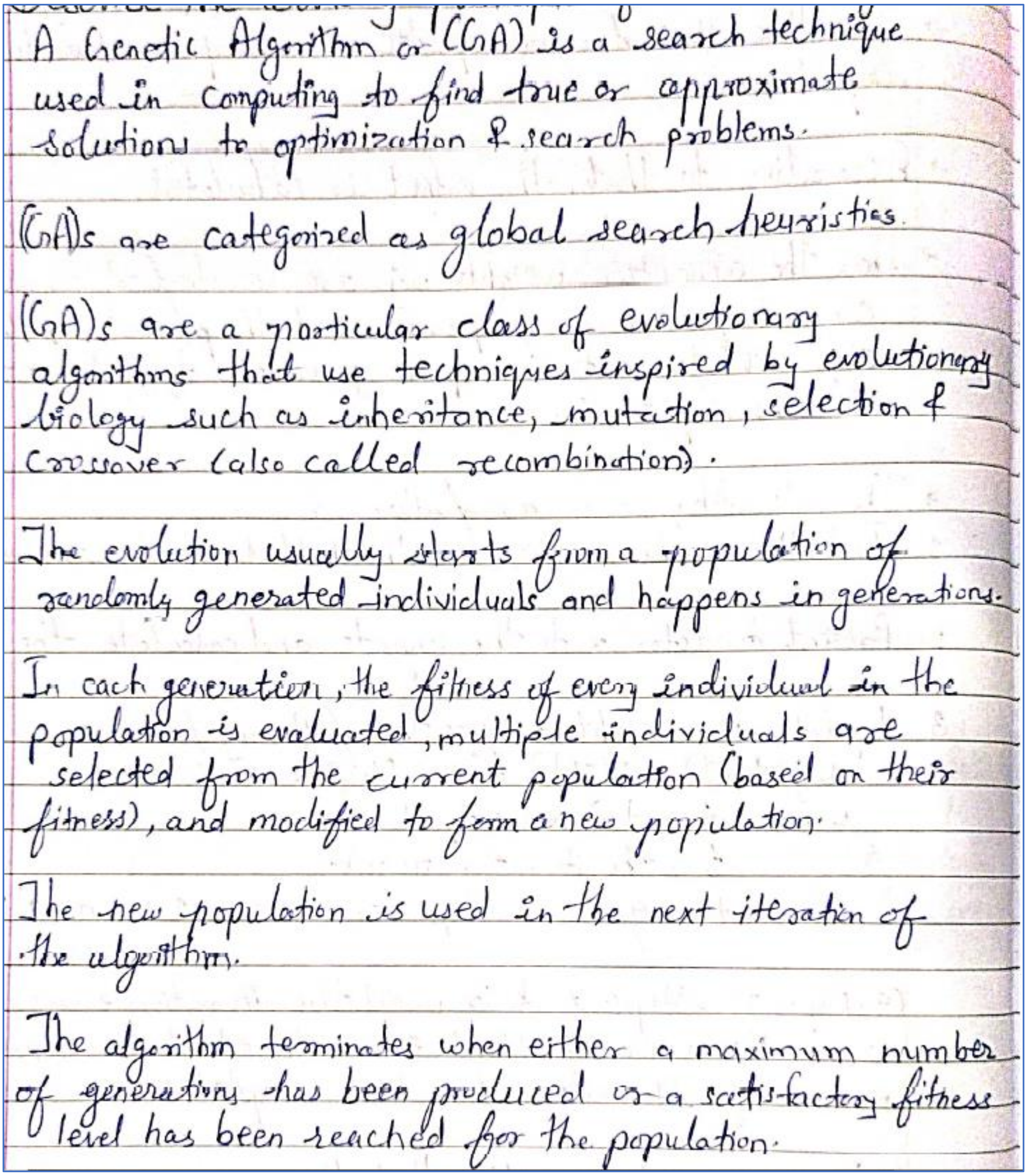
Fail:

- ➢ If we want to force a rule to fail under certain conditions, then built in predicate called fail is used in Prolog.
- ➢ Predicate fail tells prolog interpreter to fail a particular goal and subsequently forces backtracking.
- ➢ All the sub goals defined after fail will never be executed.

Cut:

- ➢ Sometimes it is desirable to selectively turn off backtracking
- ➢ Cut always succeeds but cannot be backtracked
- ➢ The cut effectively tells Prolog to freeze all the decisions made so far in this predicate. That is, if required to backtrack, it will automatically fail without trying other alternatives.

(b) Describe working principle of Genetic Algorithm 04



A Genetic Algorithm or (GA) is a search technique used in computing to find true or approximate solutions to optimization & search problems.

(GA)s are categorized as global search heuristics.

(GA)s are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection & crossover (also called recombination).

The evolution usually starts from a population of randomly generated individuals and happens in generations.

In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified to form a new population.

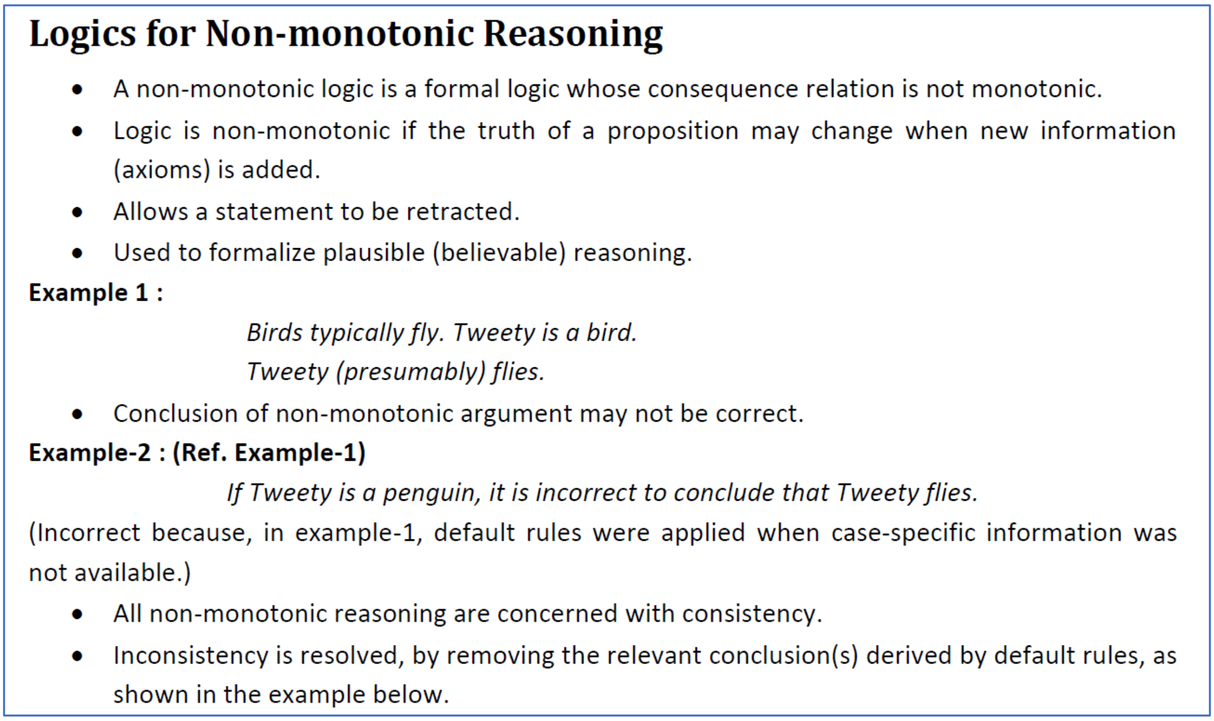
The new population is used in the next iteration of the algorithm.

The algorithm terminates when either a maximum number of generations has been produced or a satisfactory fitness level has been reached for the population.

(c) What is nonmonotonic reasoning? Explain logics for nonmonotonic reasoning. 07

Ans: Non-monotonic Reasoning:

- Non-monotonic Reasoning (NMR) is based on supplementing absolute truth with beliefs.
- These tentative beliefs are generally based on default assumptions that are made in light of lack of evidence.
- A non-monotonic reasoning (NMR) system tracks a set of tentative beliefs and revises those beliefs when knowledge is observed or derived.
- Human perceptions for various things in daily life, is a general example of non-monotonic reasoning.
- Human reasoning is not monotonic.



Logics for Non-monotonic Reasoning

- A non-monotonic logic is a formal logic whose consequence relation is not monotonic.
- Logic is non-monotonic if the truth of a proposition may change when new information (axioms) is added.
- Allows a statement to be retracted.
- Used to formalize plausible (believable) reasoning.

Example 1 :

Birds typically fly. Tweety is a bird.
Tweety (presumably) flies.

- Conclusion of non-monotonic argument may not be correct.

Example-2 : (Ref. Example-1)

If Tweety is a penguin, it is incorrect to conclude that Tweety flies.

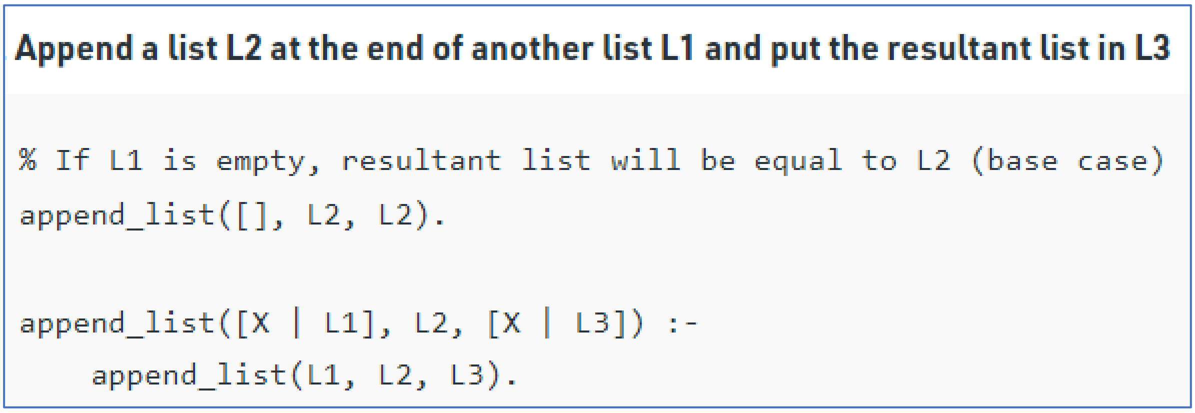
(Incorrect because, in example-1, default rules were applied when case-specific information was not available.)

- All non-monotonic reasoning are concerned with consistency.
- Inconsistency is resolved, by removing the relevant conclusion(s) derived by default rules, as shown in the example below.

OR

Q.5

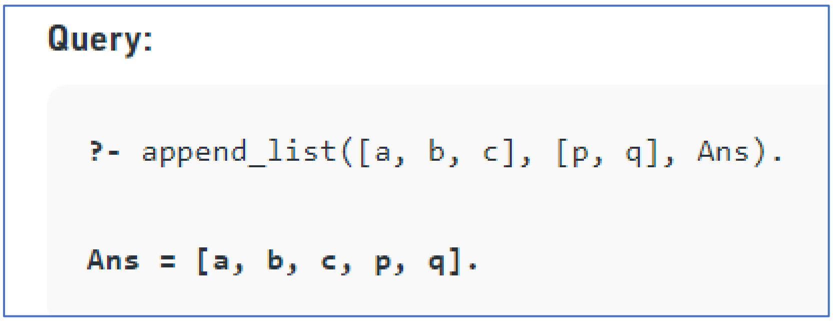
(a) Write a prolog program to append two given lists into third 03



Append a list L2 at the end of another list L1 and put the resultant list in L3

```
% If L1 is empty, resultant list will be equal to L2 (base case)
append_list([], L2, L2).

append_list([X | L1], L2, [X | L3]) :-
    append_list(L1, L2, L3).
```



Query:

```
?- append_list([a, b, c], [p, q], Ans).

Ans = [a, b, c, p, q].
```

(b) List out the genetic operators. Describe them briefly 04

ANS: GENETIC OPERATORS:

(I)Reproduction:

- ➢ It is usually the first operator applied on population.
- ➢ Chromosomes are selected from the population of parents to cross over and produce offspring.
- ➢ It is based on Darwin's evolution theory of "Survival of the fittest".
- ➢ Therefore, this operator is also known as 'Selection Operator'.

(II)Cross Over:

- After reproduction phase, population is enriched with better individuals.
- It makes clones of good strings but does not create new ones.
- Cross over operator is applied to the mating pool with a hope that it would create better strings.

(III)Mutation:

- After cross over, the strings are subjected to mutation.
- Mutation of a bit involves flipping it, changing 0 to 1 and vice-versa.

(c) Write a note on Natural Language Processing 07

ANS: Natural Language Processing:

- Natural language processing (NLP) is a field of Artificial Intelligence in which computers analyse, understand, and derive meaning from human language.
- The field focuses on communication between computers and humans in natural language and is about making computers understand and generate human language.
- Natural Language Processing (NLP) refers to communicating with an intelligent-systems using a natural language such as English.
- Processing of Natural Language is required when you want an intelligent system like robot to perform as per your instructions, when you want to hear decision from a dialogue based clinical expert system, etc.
- By utilizing NLP, developers can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, topic segmentation, etc.
- There are the following two components of NLP -
  - Natural Language Understanding (NLU): Natural Language Understanding (NLU) helps the machine to understand and analyse human language by extracting the metadata from content such as concepts, entities, keywords, emotion, relations, and semantic roles.
    - NLU mainly used in Business applications to understand the customer's problem in both spoken and written language.
  - Natural Language Generation (NLG): Natural Language Generation (NLG) acts as a translator that converts the computerized data into natural language representation. It mainly involves Text planning, Sentence planning, and Text Realization.
- Natural Language Processing (NLP) problem can be divided into two tasks:
  - **Processing written text:** using lexical, syntactic and semantic knowledge of the language as well as the required real-world information.
  - **Processing spoken language:** using all the information needed above plus additional knowledge about phonology as well as enough added information to handle the further ambiguities that arise in speech.
- Natural Language Processing has various steps given below:
  1. Morphological Analysis: scans the source code as a stream of characters and converts it into meaningful lexemes.
  2. Syntactic Analysis: used to check grammar, word arrangements, and shows the relationship among the words.
  3. Semantic Analysis: is concerned with the meaning representation.
  4. Discourse Integration: depends upon the sentences that proceeds it and also invokes the meaning of the sentences that follow it.
  5. Pragmatic Analysis: helps you to discover the intended effect by applying a set of rules that characterize cooperative dialogues.

For Example: "Open the door" is interpreted as a request instead of an order.