Date:
P. No:

ANUBHAV SINGH
IC-2K16-54
DX-1600875
MCA 9th Sem.
ARTIFICIAL INTELLIGENCE.

Q. No	Description	Page No.
1.	AI Technique & task domains	01-03
2.	Heuristic search & algo. search	03-06
3.	HST & steepest-ascent hill climb	07-10
4.	BFS & DFS	11-13
5.	SRK vs procedural knowledge	14-15
6.	Conversion of fact into CNF	16-19
7.	Conversion rules to CNF	20-21
8.	Convert rules into Prolog	22-25

Date:
P. No:

Q.No.	Description	Page No.
9.	Knowledge based system	26-27
10.	Conceptual dependency representation	28.
11.	Sementic net	29-30
12.	Fuzzy sets	31-33
13.	Short note	34-39
	Approachss to KR	34-35
	Restaurant Script	36
	Fail predicate	37
	Production rules of WJP	38-39

Q.1. What is an AI Technique? Explain various task domains of AI.

Ans. AI technique is a method that exploits knowledge that should be represented in such a way that:

- Knowledge capture generalization.
- It can be understood by people who must provide it.
- It can be easily modified to correct errors.
- It can be used in variety of situations.

In real world, the knowledge has some unwelcomed properties :-

- Its volume is huge,
- It is not well-organized,
- It keeps changing constantly.

AI techniques elevate the speed of execution of the complex program it is equipped with.

Task domains of AI :-

1. Mundane (ordinary) Tasks :

* Perception
  - Computer vision
  - Speech
  - Voice
* Natural Language Processing
  - Understanding
  - Language Generation
  - Language Translation
* Robotics
* Common sense
* Reasoning
* Planing

2. Formal Tasks :

* Games
  - Chess
  - Checkers
* Mathematics
  - Geometry
  - Logic
  - Providing properties of programs

3. Expert Tasks :

* Engineering
  - Design
  - Fault finding
  - Manufacturing planning
  - Monitoring
* Scientific Analysis
* Financial Analysis
* Medical Diagnosis
* Creativity

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 mundane. Later, more knowledge, complex knowledge representation & complicating algorithm. This is the reason AI work is more prospering in the Expert task domain.

Q. 2. Write the problem characteristics used in heuristic search in brief. How it is differ from algorithmic search?

Ans. Inorder to choose the most appropriate method for a particular problem, it is necessary to analyze the problem along several key dimensions:

- Is the problem decomposable into a set of independent smaller or easier subproblems?
- Can solution steps be ignored or at least undone if they prove unwise?
- Is the problem's universe predictable?
- Is a good solution to the problem obvious without comparision to all other possible solutions?

- Is the desired solution a state of the world or a path to a state?

- Is a large amount of knowledge absolutely required to solve the problem or is knowledge important only to constrain the search?

- Can a computer that is simply given the problem return the solution or will the solution of the problem require interaction between the computer and a person?

Difference b/w algorithmic & heuristic search.

→ An algorithm is a step wise procedure for solving a specific problem in a finite number of steps. The result of an algorithm is predictable & reproducible given the same parameters.

→ A heuristic is an educated guess which serves as a guide for subsequent explorations. Unlike an algorithm, the results of a heuristic are neither predictable and reproducible.

Q.3. Define the heuristic search technique. Explain steepest - ascent hill climbing method with an example.

Ans. A heuristic is a technique that improves the efficiency of a serach process, possibly by sacrificing claims of completeness.
Heuristic search are good to the extent that they point in generally interesting directions;
They are bad to the extent that they may miss points of interest to particular individuals.

There are two cateries of Heuristic Search Techniques:

- Direct Techniques: Also called blind search, this tech is not always possible because they require too much of time or large memory.

- Weak Techniques: These type of techniques are very effective if applied correctly on the right kind of Tasks.

Special purpose heuristics exploit domain specific knowledge.

STEEPEST - ASCENT HILL CLIMBING ALGORITHM:

Step 1: Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.

Step 2: Loop until a solution is found or until a complete iteration produces no change to current state:

a. Let SUCC be a state such that any possible successor of the current state will be better than SUCC.

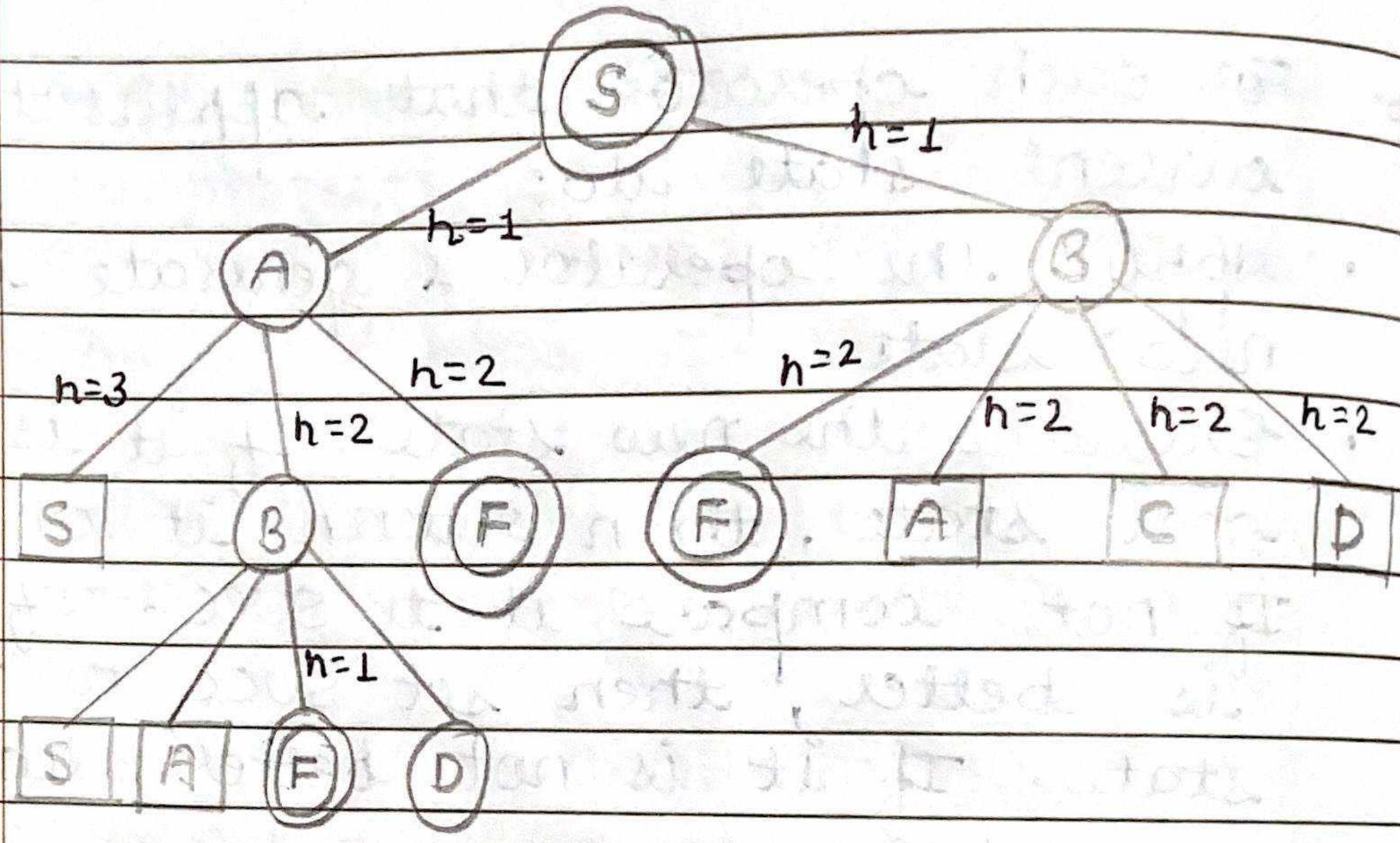
b. For each operator that applies to the current state do:

- Apply the operator & generate a new state.
- Evaluate the new state. If it is a goal state, then return it and quit. If not, compare it to SUCC. If it is better, then set SUCC to this state. If it is not better, leave SUCC alone.

c. If the SUCC is better than the current state, then set current state to SUCC

Step3: Exit

Ex:



0. current_node = S, successors (A=1, B=1)
1. current_node = A, successors (B=2, F=2, S=2)
2. current_node = B, successors (F=1, ...)
3. current_node = F: Goal Found!

→ Result is : S > A > B > F at cost = 4
But more optimal when backtrack and step 1 h(S) = 2 then

0. current_node = S, successors (B=1, A=1)
1. current_node = B, successors (F=2, A=2, C=2, D=2)
2. current_node = F, Goal Found!

→ Result is : S > B > F at cost = 3
We will get same cost at A node of left side : S > A > F cost = 3

Q.4. Write the algorithm for breadth first search (BFS) & depth first search (DFS).

Breadth first search (BFS):

1. Implement two new lists of nodes 'open' & 'closed'.

2. Start with 'open' containing just the initial state.

3. Until a goal is found or there are no nodes left on 'open' do:
   a. Pick the best node on 'open'.
   b. Generate its successors.
   c. For each successor do:
      i. If it has not been generated before, evalute it, add it to 'open', and record its parent.
      ii. If it has been generated before, change the parent if this new path is better than the previous one.

Algorithm for BFS:

- Start with queue = [initial-state] and found = FALSE.

- While queue not empty & not found do
  - Remove the first node N from queue.
  - If N is a goal state, then found = TRUE.
  - Find all the successor nodes of N, and put them on the end of the queue.

- Exit.

Algorithm for DFS:

- Start with stack = [initial-state] and found = FALSE.

- While stack not empty & not found do:
  - Remove the first node N from stack.
  - If N is a goal state, then found = TRUE.
  - Find all the successor nodes of N, and put them on the top of the stack.

- Exit.

5. Explain difference between simple relational knowledge & procedural knowledge with example.

Simple relational knowledge

- Provide very less inferential capabilities.
- May serve input to powerful inference engines.
- This approach of knowledge is famous in dB systems where the relationship b/w different entities is represented.
- Each fact about a set of the object is set out systematically in columns.

Ex:-

Player	Height	Weight	Handed
Peter	6-2	210	Left
Virat	5-10	180	Right
Smith	5-8	195	Right

Fails to infer "which RH player can best face a particular bowler".

Procedural knowledge :

- Represents "how to make it" rather than "what it is".
- May have inferential efficiency, but no inferential adequancy and acquisitional efficiency.
- In this knowledge, we can use various coding languages such as LISP language & prolog language.
- We can easily represent heuristic or domain-specific knowledge using this approach.

Ex: Writing a LISP or Prolog program.

→ To construct the exams timetable.

→ Assigning classrooms and benches according to rollno. and timetable to students.

Q.6. Consider the following axioms. Convert these facts in predicate logic & CNF form and prove that "Marcus not alive now" using resolution.

1. Marcus was a man.
man (Marcus)

2. Marcus was pompeian.
Pompeian (Marcus)

3. Marcus was born in 1890.
born (Marcus, 1890)

4. All men are martal.
$\forall x$: man $(x) \rightarrow$ martal (x)

5. All pompeians died when the volcano erupted in 1905.
$\forall$: Pompeian $(x) \rightarrow$ died $(x, 1905)$
erupted (volcano, 1905)

6. No mortal lives longer than 150 yrs.
$\forall x : \forall t_1 : \forall t_2 : mortal(x) \wedge born(x, t_1) \wedge gt(t_2 - t_1, 150) \rightarrow dead(x, t_2)$

7. It is now 2015.
$now = 2015$

8. Alive means not dead.
$\forall x : \forall t : [alive(x, t) \rightarrow \neg dead(x, t)] \wedge [\neg dead(x, t) \rightarrow alive(x, t)]$

9. If someone dies, then he is dead at all later times.
$\forall x : \forall t_1 : \forall t_2 : died(x, t_1) \wedge gt(t_2, t_1) \rightarrow dead(x, t_2)$

* Converting predicate logics to CNF form:

1. man (Marcus)
2. Pompeian (Marcus)
3. born (Marcus, 1890)
4. $\neg$man (x_1) $\vee$ mortal (x_1)

5a. $\neg$Pompeian (x_2) $\vee$ died (x_2, 1905)

5b. erupted (volcano, 1905)

6. $\neg$mortal (x_3) $\vee$ $\neg$born (x_3, t_1) $\vee$ $\neg$gt ($t_2 - t_1$, 150) $\vee$ dead (x_3, t_2)

7. now = 2015

8a. $\neg$alive (x_4, t_3) $\vee$ $\neg$dead (x_4, t_3)

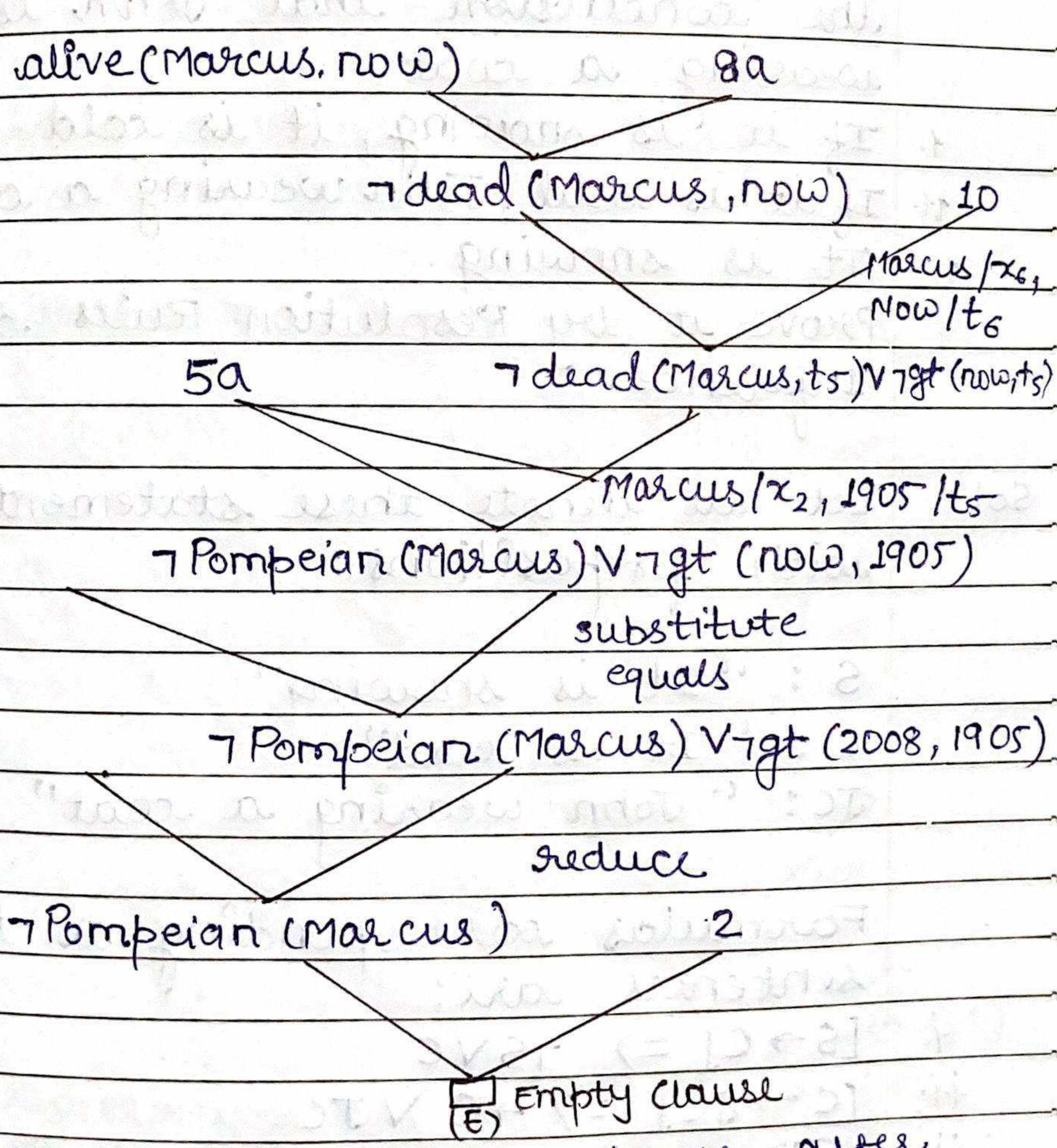
8b. dead (x_5, t_4) $\vee$ alive (x_5, t_4)

9. $\neg$died (x_6, t_5) $\vee$ $\neg$gt (t_6, t_5) $\vee$ dead (x_6, t_6)

To prove:

$\neg$alive (Marcus, now)

* Proving "Marcus is not alive now" using resolution:



So, we get (E) Empty clause after deduction. Therefore "Marcus is not alive" is proved.

Q.7. Write the conversion rules to CNF. The following arguments support the conclusion that John is wearing a coat.

i. If it is snowing, it is cold.
ii. If it is cold, John wearing a coat.
iii. It is snowing.

Prove it by Resolution Rules of Inference.

Sol:- Let us denote these statements with prepositions:

S : "It is snowing"
C : "It is cold"
JC : "John wearing a coat"

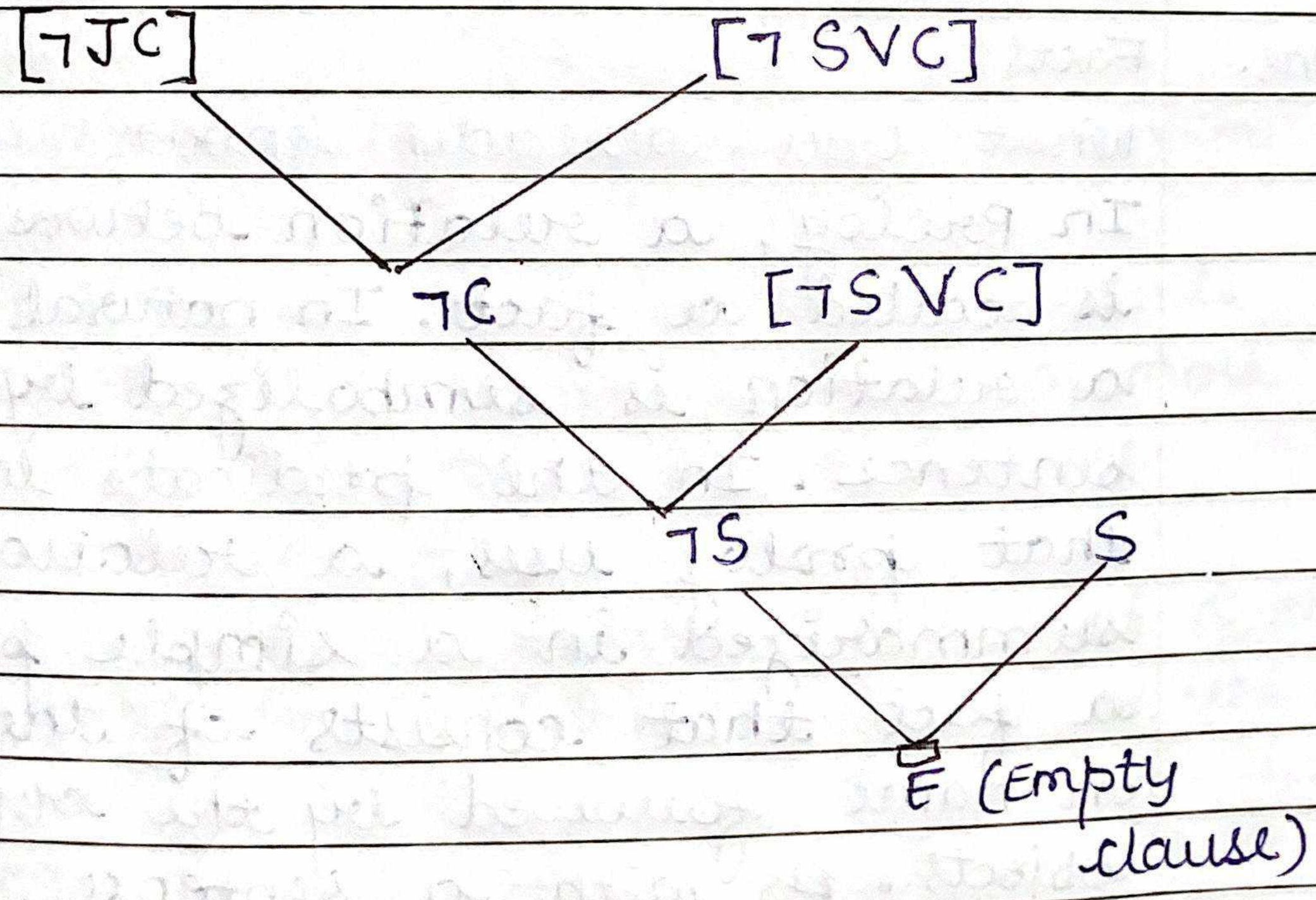
Formulas corresponding to the sentences are:

i. $[S \rightarrow C] \Rightarrow \neg S \vee C$
ii. $[C \rightarrow JC] \Rightarrow \neg C \vee JC$
iii. $[S] \Rightarrow S$

- To prove:
  John wearing coat
  JC

- Let's assume "John is not wearing coat" [¬JC]

Resolution Rules



Since an empty clause (E) has been deduced. So, we can say our our assumption is wrong & hence we have proved: "John wearing coat".

Q.8. State the role of facts and rules in knowledge representation with examples. Convert following rules in prolog.

a) A person is happy if he is rich & famous.
b) Ram likes anyone who likes cricket.
c) Ram likes anyone who likes themselves.
d) A hopelsss person is an unhappy person.
e) A healthy person eats fruits & drinks milk.

Ans. Facts:

What you already know.

In Prolog, a relation between objects is called a fact. In natural lang, a relation is symbolized by a sentence. In the predicate logic that prolog uses, a relation is summarized in a simple phrase a fact that consists of the relation name followed by the object or objects. As with a sentense, the fact ends with a period(.).

Ex: Adam likes Eva.
Eva likes Adam.
Adam likes dogs

* Conversion of facts into prolog :-
likes (Adam, Eva).
likes (Eva, Adam).
likes (Adam, dogs).

Rules :
What you can infer from given facts.
A rule is a conjunction that is known to be true if one or more other conclusion or facts are found to be true.
The if symbol separates the head of the rule from the body ; it can be the character ':-' or the word 'if'. The body is the set of conditions that must be true so that Prolog can prove that the head of the rule is true.

Ex: Ram & shyam are friends if they like each other.

```
friends (X, Y) :-
        likes (X, Y),
        likes (Y, X).
```

Conversion of rules in prolog:-

a. A person is happy if he is rich & famous

```
happy (X) :-
        rich (X),
        famous (X).
```

b. Ram likes anyone who likes cricket

```
likes (ram, X) :-
        likes (X, cricket).
```

c. Ram likes anyone who like themselves:

```
likes (ram, X) :-
        likes (X, X).
```

d. A hopeless person is an unhappy person.

unhappy (X) :-
hopeless.

e. A healthy person eats fruits and drink milk.

healthy (X) :-
eat (X, fruits),
drink (X, milk).

Q.9. What is a knowledge based system Draw & explain its architecture.

A knowledge based system is a computer program that uses AI to solve problems within a specialized domain that ordinarily requires human expertise.
Typical tasks for expert systems involves classification, diagosis, monitoring, design, scheduling, & planning for specialized tasks.
A knowledge based system is a more general than the expert system.

Knowledge based system architecture:
A KBS consists of:

- Interface
- Interface engine
- Knowledge Base

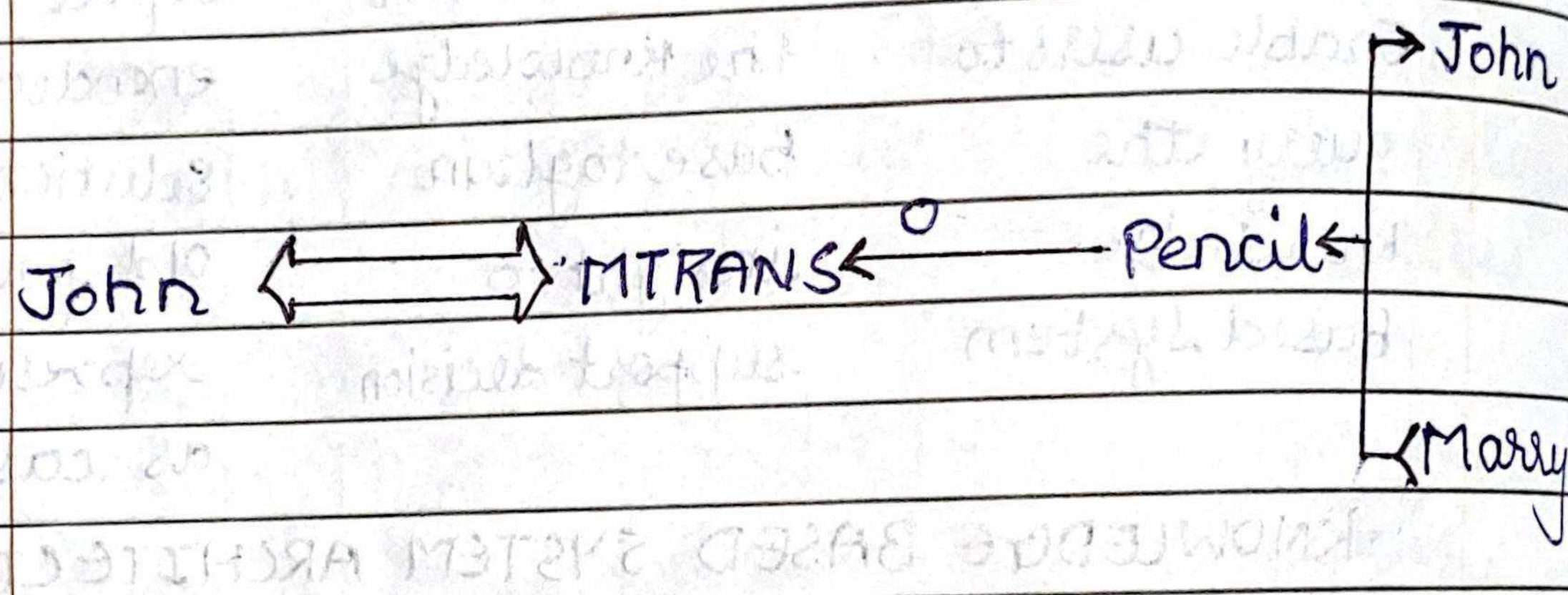
Interface	Interference engine	Knowledge base
Enable users to query the knowledge based system	Interacts with the knowledge base to glean insight to support decision	Expert know. encoded as rules Solution to old problems represented as cases

KNOWLEDGE BASED SYSTEM ARCHITECTURE

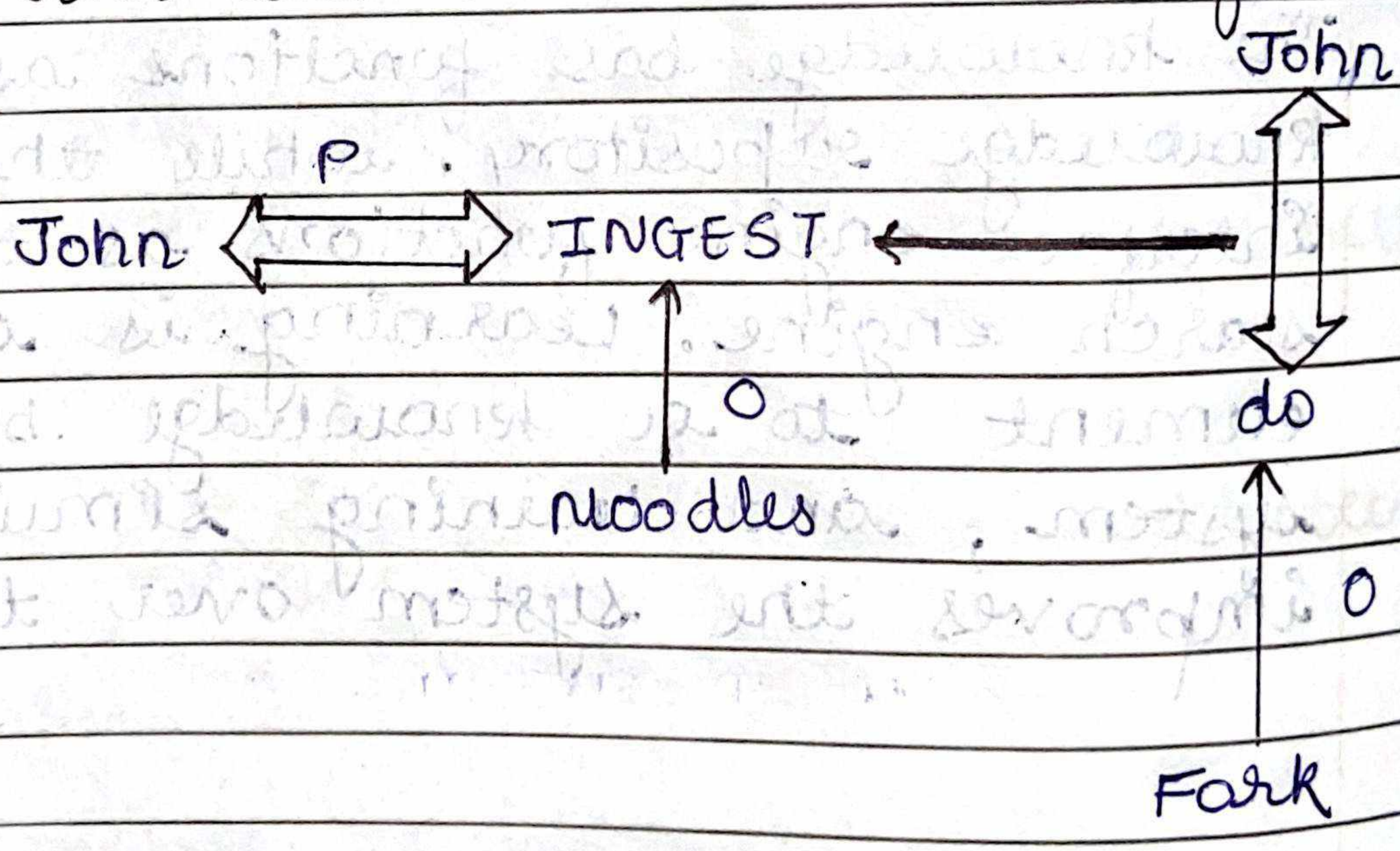
The knowledge base functions as the knowledge repository, while the interface engine functions as the search engine. Learning is a key element to a knowledge based system, and learning simulation improves the system over time.

Q.10. Show conceptual dependency representation of following:

a. John begged Mary for a pencil.



b. John ate noodles with a fork.



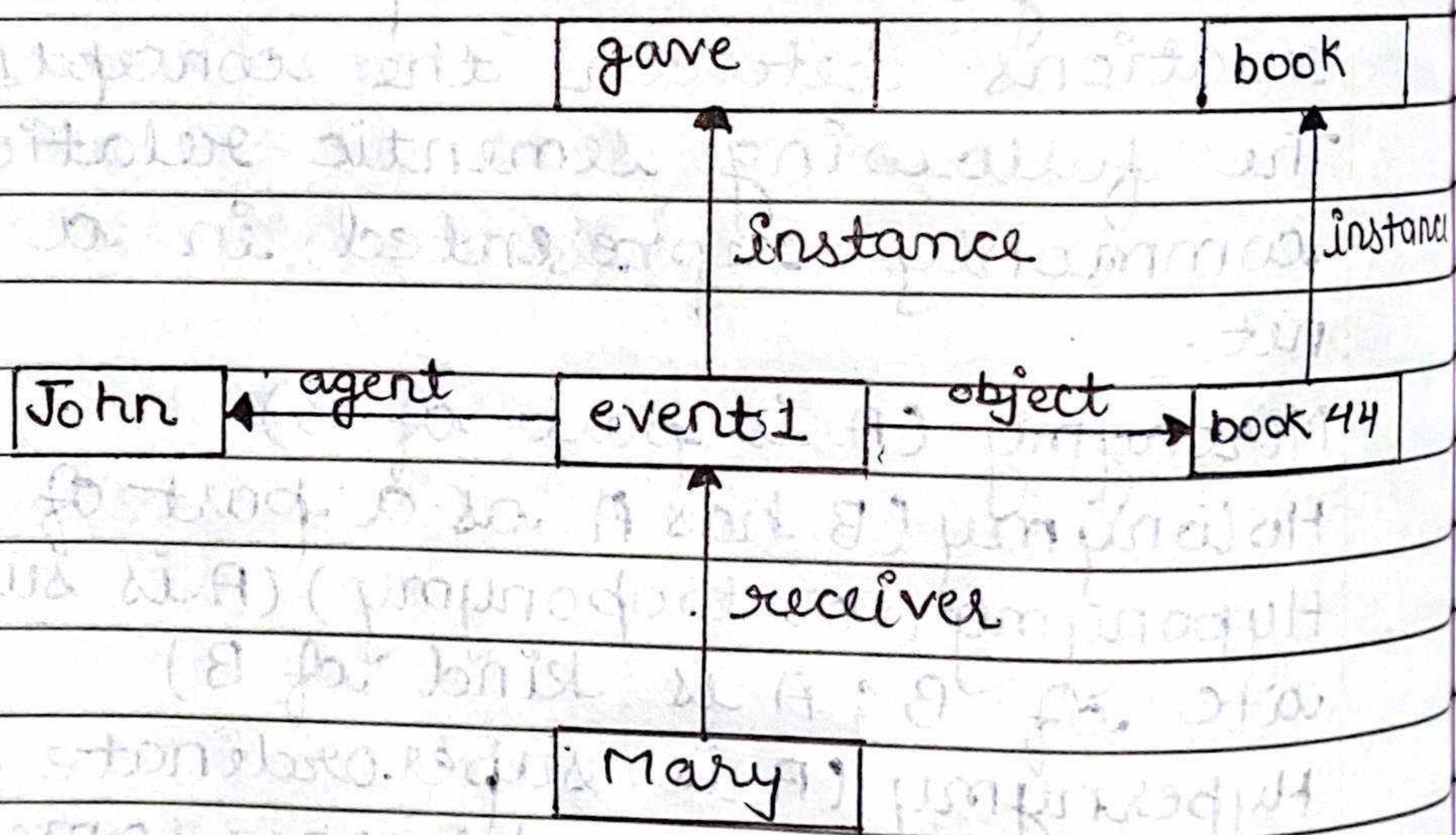
Q.11. Illustrate the use of sementic net in KR with an example.

Ans. A semantic network is often used as a form of KR. It is a directed graph consisting of vertices which represents concepts and edges which represents sementic relations between the concepts.
The following sementic relations are commonly represented in a sementic net.

Meronymy (A is part of B)
Holonymy (B has A as a part of itself)
Hyponymy (or troponymy) (A is subordinate of B; A is kind of B)
Hypernymy (A is superordinate of B)
Synonymy (A denotes the same as B)
Antonymy (A denotes the opposite of B).

The physical attributes of two person can be represented as in the following figure using a semantic net.

Ex:- John gave mary the book.
Here we have several aspects of an event.



Semantic network

Q.12. On the basis of 6 jobs, your task is to choose the job, that will give you the highest salary. Given the constraints that the job should be interesting and close to your home,

The fuzzy sets are:

Interest = {(1, 0.4), (2, 0.6), (3, 0.7), (4, 0.9), (5, 0.3), (6, 0.8)}

Drive = {(1, 0.1), (2, 0.9), (3, 0.6), (4, 1.0), (5, 0.2), (6, 0.3)}

Salary = {(1, 0.785), (2, 0.312), (3, 0.723), (4, 0.129), (5, 0.345), (6, 0.823)}

Highest membership shows highest interest, lowest drive & highest salary also draw graph.

Ans. Apply fuzzy AND or intersection operation for getting the result or decision.
Fuzzy AND operation is:

$$\mu(A \text{ and } B) = \min(\mu(A), \mu(B))$$

Where, $\mu(A)$ is degree of membership of set A
$\mu(B)$ is degree of membership of set B

Given fuzzy sets are
Interest = {(1,0.4), (2,0.6), (3,0.7), (4,0.9), (5,0.3), (6,0.8)}

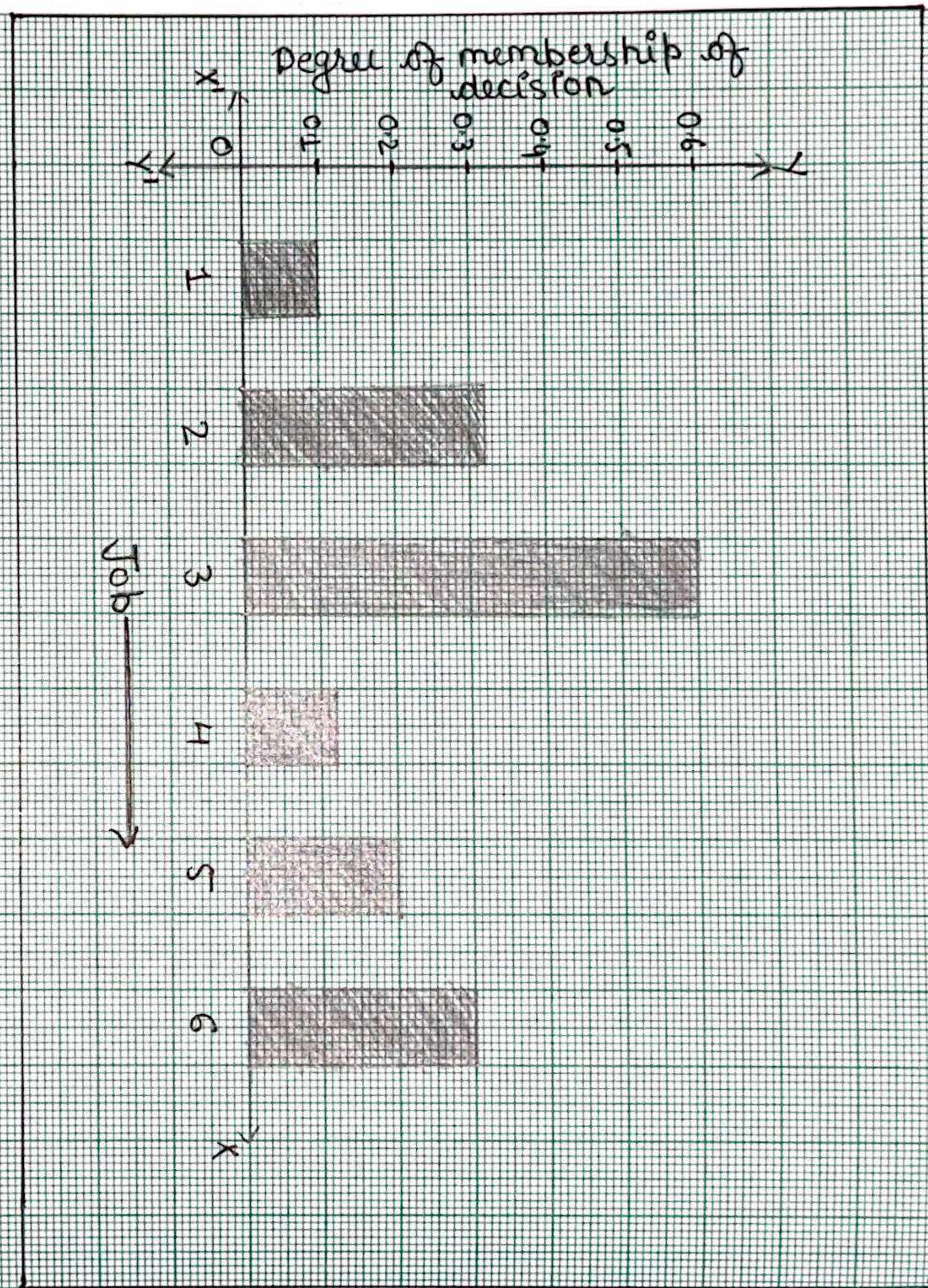
Highest degree of membership showing that job 4 is most interesting & lower degree of membership showing the lowest interested job as per given set job 5 is least interested. Now apply fuzzy operation for decision making.

Decision = Intersection [interest, drive, salary]

Result is:

Decision = {(1,0.1), (2,0.312), (3,0.6), (4,0.129), (5,0.2), (6,0.3)}

Decision set is showing that job 3 has highest member that means job 3 is more interesting, less drive & high salary.


Degree of membership of decision
0
0.1
0.2
0.3
0.4
0.5
0.6
1
2
3
4
5
6
Low

Q.13. Write short note on

a. Approaches to Knowledge representation.

There are mainly four approaches to KR, which are:

i. Simple relational knowledge:
Provides very week inferential capabilities. May serve as the input to powerful interference engine.

ii. Inheritable knowledge:
Objects are organized into classes and classes are organized in a generalization hierarchy.
Inheritance is a powerful form of inference, but not adequate.
This approch contains inheritable knowledge which shows a relation between instance & class, & it is called instance relation.

iii. Inferential knowledge:

Inferential knowledge approach represents knowledge in the form of formal logics.
This approach can be derived more facts.
It guaranteed correctness.

iv. Procedural knowledge:

Procedural knowledge approach uses small program and codes which describes how to do specific things, and how to proceed.
In this approach, one important rule is used which is If-then rule.
In this knowledge, we can use various coding languages such as LISP language & Prolog language.

b. Restaurant script.

Eat-at-Restaurant Script	
Props :	(Restaurant, Money, Food, Menu, Tables, chairs)
Roles :	(Hungry-persons, Wait-person, chef-person)
Point-of-view:	Hungry-person
Time-of-occurrence	(Times-of-operation of Restaurant
Place-of-occurence:	(Location of Restaurant)
Event Sequences:	First : Enter-Restaurent Script
	then : if (wait-to-be-seated
	sign or reservation) then get
	waiter's attention script
	then: please-be-seated script
	then: order food script
	then: eat-food-script unless
	(long-wait) when exit-
	restaurant-angry script
	then: pay-for-it script
	finally: leave-restaurant
	script

c) Fail predicate with example.

The fail predicate is provided by prolog. When it is called it causes the failure of the rule. And this will be forever, nothing can change the statement of this predicate. We can associate it to cut predicate. It allows us to include the negation rule. A typical use of fail is negation of predicate.

Ex: bachelor (P) :- male (P), not (married(P)).
male (henry).
male (tom).
married (tom).

?- not (married (Who)).
no

This is a fail predicate because variable binds Who = tom & so the negative goal fails.

d) Production rules for water-jug problem.

S.No.	Current state	Next state	Description
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 $x + y >= 3$ and $y > 0$	$(4, y-(4-x))$	Pour water from the 3 gallon jug into 4 until 4-gallon jug is full.

S.No	Current state	Next state	Description
8.	(x,y) if $x+y>=3$ and $x>0$	$(x-(3-y),3)$	Pour water from the 4 gallon jug into the 3 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 $x+y<=3$ & $x>0$	$(0,x+y)$	Pour all the water from the 4 gallon jug into the 3-gallon jug.
11.	$(0,2)$	$(2,0)$	Pour 2 gallons from 3 gallon jug into 4 gallon jug.
12.	$(2,y)$	$(0,y)$	Empty the 2 gallons in the 4-gallon jug on the ground.