Course Title : Introduction to Artificial Intelligence
Course no : CSC-304 |355
Credit hours : 3

Full Marks : 60+20+20
Pass Marks: 24+8+8

Contents:

UNIT 1: Introduction to Artificial Intelligence

9 page Artificial Intelligence and related fields, brief history of AI, applications of AI, Definition and Importance of knowledge and learning, Agent and its type and performance measures.

UNIT 2: Problem solving

5 page Problem definition, problem as a state space search, problem formulation, problem types: Toy problems, Real world problems, Well-defined problems, Constraint satisfaction problem (Basic concept & examples), Production systems (Definition, Architecture, examples).

UNIT 3: Search Techniques

7 page Uniformed search techniques: depth first search, breadth first search, depth limit search, Iterative deepening search, Bidirectional search, & search strategy comparison. Informed search techniques: Greedy best first search, A* search, Hill climbing search, simulated annealing, Game playing, Adversarial search techniques- mini-max procedure, alpha beta pruning.

Book :- E.Rich and Knight, Artificial Intelligence, McGraw Hill

UNIT 4: Knowledge Representation, Inferential reasoning

12 page

Formal logic connectives, truth table, syntax, semantics, tautology, validity, well-formed formula, propositional logic, Inference with PL: Resolution, Backward chaining & Forward chaining, predicate logic (FOPL), quantification, inference with FOPL by converting into PL (Existential & Universal instantiation), Directly with FOPL (Unification & lifting, resolution, backward chaining, forward chaining), Rule based deduction system, Statistical reasoning - Probability & Bayes theorem & casual networks, reasoning in belief network.

UNIT 5: Structured Knowledge Representation

13 page

Representation and mappings, Approaches to knowledge representation, Issues in knowledge representation, Semantic nets, Frames, Conceptual dependencies and scripts (Rich and knight)

UNIT 6: Machine Learning

6 page

Concepts of learning, learning from examples, explanation based learning, learning by analogy, learning by simulating evolution, learning by training neural nets, learning by training perceptions.

UNIT 7: Applications of Artificial Intelligence

13 page

Expert system (Architecture, Expert system development process), Neural Network (Mathematical model, get realization, Network structure), natural language processing (steps of NLP parsing), Basic concepts of Machine vision.

UNIT 1:
Introduction to Artificial Intelligence

Intelligence:

A very general capacity among the other things that (differently) involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience. It is not simply a book learning, a narrow academic skills, or test-taking smarts. Furthermore, "it reflects a broader and deeper capacity for comprehending our surroundings - making sense or figuring out what to do": from Mainstream Science of Intelligence (1994).

In otherwords, the ability to learn complex ideas to adapt effectively to the environment, to learn from the experience to engage in various forms of reasoning and to overcome obstacles by taking thought. Although these individual difference can be substantial they are never entirely consistent. A given person's intellectual performance will vary on different ocassion in entirely different domain and judge by different criteria. Therefore, intelligence is ability to reason, understand, create, learn from experience, plan and execute complex task.

Artificial:

Made as a copy something like natural. Therefore AI is defined as giving machines ability to perform task normally associated with human intelligence.

According to Barr and Feignbaum, "Artificial Intelligence is a part of computer science that deals with designing intelligence computer system i.e. system that exhibits the characteristics associated with intelligence in human behaviours."

What AI is not?

AI is not a creation and study of conventional computer system. AI is not study of mind, nor body, nor of language but some overlaps occurs with these fields.

Artificial Intelligence:

Artificial Intelligence of machine is the branch of computer science that aims to design intelligent agents which is a program which perceives its environment and takes action that maximizes the chance of success. With AI it comes the issue of deduction, reasoning, problem solving, knowledge representation, planning, learing natural language processing.

In other words, AI is a branch of science which deals with helping machines to find solutions to complex problem in more human ~~fa~~ life fashion. This generally involves borrowing characteristics from human intelligence and applying them in an algorithm in a computer friendly way. A more or less flexible or efficient approach can be taken depending on the requirement established which influence how artificial the intelligent being appears.

The definitions can be divided into two dimensions: the top dimension is concerned with thought processes and reasoning whereas bottom dimension addresses the behaviour.

logical manner.

Systems that thinks humanly	System that thinks rationally
i) The exciting new effort to make computers think i.e. machine with minds, in the full or literal sense. (Haugeland, 1985)	i) The study of mental faculties through the use of computational models. (Charniak and Mc. Dermott, 1985).
ii) The automation of activities that is associated with human thinking; activities such as decision-making, problem solving, learning. (Bellman, 1978).	ii) The study of computation that make it possible to perceive, reason and act. (Winston, 1992).

System that act like humans	System that acts rationally.
i) The art of creating machines that perform functions that require intelligence when performed by people (Kurzweil, 1990)	i) Computational Intelligence is the study of the design of intelligent agents. (Poole et al, 1998)
ii) The study of how to make computer do things at which at the moment, people are better (Rich & knight, 1991)	ii) AI is concerned with intelligent behaviour in artifacts (Nilsson, 1998).

The top dimension is concerned with throught processes and reasoning, whereas bottom dimension is concerned with behaviour. The definition on the left measures the success in terms of fidelity of human performance, where as definition on the right measures an ideal concept of intelligence, which is called rationality.

Acting Humanly: The Turing Test Approach

The Turing Test, proposed by Alan Turing (1950) was designed to provide a satisfactory operational definition of intelligence. The computer passes the test if a human interrogator after posing some written question, cannot tell whether the written response come from human or not.

To pass a Turing test, a computer must have following capabilities:

→ Natural Language Processing: Must be able to communicate in English successfully.
→ Knowledge representation: To store what it knows and hears.
→ Automated reasoning: Answer the questions based on the stored information.
→ Machine learning: Must be able to adapt in new circumstances.

Turing test avoid the physical interaction with human interrogator. Physical simulation of human being is not necessary for testing the intelligence.

The total Turing test includes video signals and manipulation capability so that the interrogator can test the subject's perceptual abilities and object manipulation ability. To pass the total Turing test computer must have following additional capabilities:

→ Computer Vision: To perceive objects

→ Robotics: To manipulate objects and move.

Thinking Humanly: Cognitive modeling approach

Make the ~~mak~~ machines with mind. Cognition means the action or process of acquiring knowledge and understanding through thought, experience and senses. To make a machine that think like human brain, scientific theories of internal brain activities (cognitive model) are required. Two ways of doing this is:

- Predicting and testing human behaviour (cognitive science)
- Identification from neurological data (Cognitive neuroscience)

Once we have precise theory of mind, it is possible to express the theory as a computer program. But unfortunately until up to now there is no precise theory about thinking process of human brain. Therefore it is not possible to make the machines that think like human brain.

Thinking Rationally : The laws of thought approach

The Greek philosopher Aristotle was one of the first attempt to codify the right thinking that is irrefutable reasoning process. He gave syllogisms that always yielded correct conclusion when correct premises are given.

For example:

Ram is a man
Man is mortal
⇒ Ram is mortal.

These laws of thought were supposed to govern the operation of mind.
This study initiated the field of logic. The logicist tradition in AI hopes to create intelligent systems using logic programming.

Problems:

- It is not easy to take informal knowledge and state in the formal terms required by logical notation, particularly when knowledge is not 100% certain.
- Solving problem principally is different from doing it in practice. Even problems with certain dozens of fact may exhaust the computational resources of any computer unless it has some guidance as which reasoning step to try first.

Acting Rationally : The rational Agent approach

Agent is something that acts. Computer agent is expected to have following attributes:

→ Autonomous control
→ Perceiving their environment
→ Persisting over a prolonged period of time
→ Adapting to change
→ And capable of taking on another's goal.

Rational behaviour- means doing the right thing.
Right thing - is that which is expected to maximize goal achievement, given the available information.
Rational Agent - is one that acts so as to achieve the best outcome.

In the laws of thought approach to AI, the emphasis is given to correct inferences. Making correct inferences is sometimes part of being a rational agent because one way to act rationally is to reason logically to the conclusion and act on that conclusion. On the otherhand, there are also some ways of acting rationally that cannot be said to involve inference. For example; recoiling from a hot stove is a reflex action that is usually more successful than a slower action taken after careful deliberation.

Scanned with CamScanner

Characteristics of AI :

1. Symbolic Reasoning :- AI machine should not only deal with numerical calculation. It must also have reasoning capacity of object represented by symbols and their properties and their relationships.

2. knowledge:- AI general principles are stored in the program and that program must be used for their respective circumstances.

3. Search :- AI machine must have a method of finding a solution to a problem when no direct method exists.

4. Flexible Control:- Processing direction can be changed by changing facts in the environment.

Applications of AI :-

1. Game playing :
You can buy machines that can play master level chess for a few hundred dollars. There are some AI in them, but they play well against people mainly through brute force computation - looking at hundreds of thousands of positions. To beat a world champion by brute force and known reliable heuristics requires being able to look at 200 million positions per second.

2. Speech recognition:

In the 1990's, computer speech recognition reached a practical level for limited purposes. Thus United Airlines has replaced its keyboard tree for flight information by a system using speech recognition of flight number and city names. It is quite convenient. On the other hand, while it is possible to instruct some computers using speech but most users have gone back to the keyboard and the mouse.

3. Understanding natural language:

Just getting a sequence of words into a computer is not enough. Parsing sentences is not enough either. The computer has to be provided with an understanding of the domain the text is about, and this is presently possible only for very limited domains.

4. Computer vision:

The world is composed of 3-dimensional objects, but the inputs to the human eye and computers' TV cameras are two dimensional. Some useful programs can work solely in two dimensions, but full computer vision requires partial 3-dimensional that is not just a set of two-dimensional views. At present, there are only limited ways of representing three-dimensional information directly, and they are not as good as what human evidently use.

5. Expert Systems:

A "knowledge engineer" interviews experts in a certain domain and tries to embody their knowledge in a computer program for carrying out some task. How well this works depends on whether the intellectual mechanisms required for the task are within the present state of AI. When this turned out not to be so, there were many disappointing results. For example: MYCIN in 1974, which diagnosed bacterial infections of the blood and suggested treatments. It did better than medical students or practicing doctors, provided its limitations were observed.

6. Heuristic Classification:

One of the most feasible kinds of expert system given the present knowledge of AI is to put some information in one of a fixed set of categories using several sources of information. An example is advising whether to accept a proposed credit card purchase. Information is available about the owner of the credit card, his record of payment and also about the item he is buying and about the establishment from which he is buying it (e.g., about whether there have been previous credit card frauds at this establishment).

Knowledge

- Knowledge is a theoretical or practical understanding of subject or particular domain. knowledge is also the sum of what is currently known. The body of truth, information and principles acquired by mankind.
- knowledge is information combined with experience, context, interpretation and the reflection. It is a high value form of information that is ready to apply to decision and actions.
- knowledge consists of information that has been:
  a) interpreted b) categorized c) experienced and revised
- In general, knowledge is more than just data, it consists of facts, ideas, beliefs, rules, relationship, etc.
- knowledge is more important in AI for making intelligent machines.

Key issues confronting the designer of AI system are:

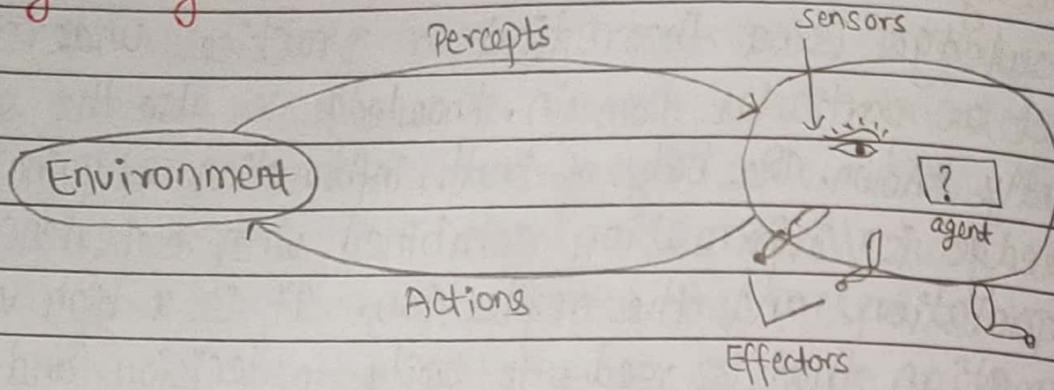
- knowledge acquisition: Gathering the knowledge from the problem domain to solve the AI problem.
- Knowledge representation: Expressing the identified knowledge into some knowledge representation language such as propositional logic, predicate logic, etc.
- knowledge manipulation: Knowledge is manipulated to draw conclusion from knowledge based.

Note: The main importance of knowledge is learning.

Agent:- An agent is anything that can perceive its environment through its sensors and act upon that environment through actuators.

(12)

Intelligent Agent:



An intelligent agent perceives its environment via sensors and act rationally upon that environment with its effectors. Hence an agent get percept one at a time and map this percept sequence to actions.

Properties of the agent:

i) Autonomous
ii) Interacts with other agent plus the environment
iii) Reactive to the environment
iv) Goal directed
v) Should have ability to learn
vi) Should have ability to adjust in changing environment.

For human:
sensor - eye, ear, skin, tongue, nose, etc
percepts - object in the visual field
effectors - hand, limbs, etc.
actions - lift finger, turn left, walk, run, etc.

A more specific example: automated taxi driving system.
environment - urban street, traffic, freeways, etc.
goal - maintain safety, reach destination, obeys law, provide passenger comfort

actions - horn, steering, accelerate
percept - speedometer, engine sensor, GPS, etc.

Percept: The agent perceptual inputs at any given instance.

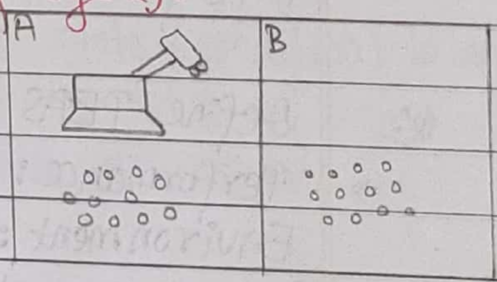
Percept sequence: The complete history of everything the agent has ever perceived.

Agent function: The agent function is a mathematical concept that maps percept sequence to actions.

$$f : P^* \longrightarrow A$$

The agent function will internally represented by the agent program.

The vacuum cleaner world (Example of agent):



Environment: square A and square B
Percepts : Location and content
eg: [A, Dirty]
Actions: Left, Right, suck, no-op.

Percept sequence:	Actions:
[A, Clean]	Right
[A, Dirty]	Suck
[B, Clean]	Left
[B, Dirty]	Suck
⋮	⋮

Concept of rational agent:

A rational agent is one that does the right thing.
Right thing: A right thing is one that will cause the agent to be most successful.

Therefore, we need some way to measure success of an agent. Performance measures are the criterian for success of an agent behaviour. For example: performance measure of a vacuum cleaner agent could be amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise (error) generated.

Environment: To design a rational agent, we must specify its task environment. Task environment means PEAS description of the environment:
i) Performance ii) Environment iii) Actuators iv) Sensors

Q: Define PEAS of the automated Taxi driver.
→ Performance: safety, destination, profits, legally, comfort, etc.
Environment: urban street, traffic, free ways, etc.
Actuators: steering, accelerating, brake, horn, etc.
Sensors: Speedometer, engine sensors, GPS, etc.

Types of Agent:

Agent = architecture + program

1) Simple Reflex Agent
2) Model Based Reflex Agent
3) Goal Based Agent
4) Utility Based Agent
5) Learning Agent.

Scanned with CamScanner

1) Simple Reflex Agent :

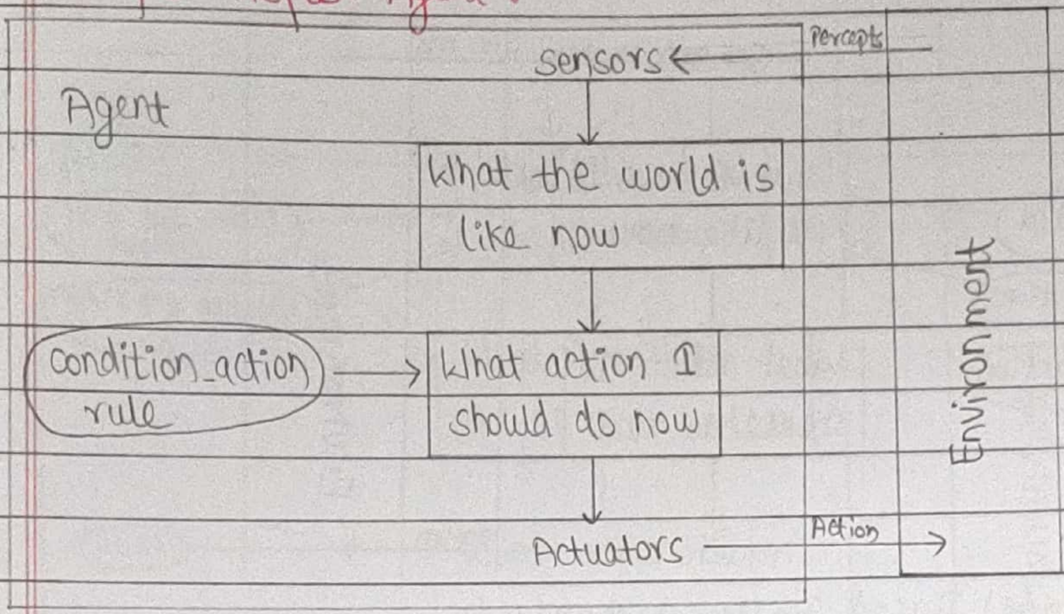


Fig:- Schematic diagram of a simple reflex agent.

→ They choose actions only based on the current percept (situation).
→ Their environment is completely observable.
→ Condition_action rule: It is a rule that maps state (condition) to an action. For eg:- if (condition)
then (action)

<u>Example</u>:
Vacuum cleaner is an example of simple reflex agent. If the current square is dirty then suck, otherwise move to the other square.

2) Model Based Reflex Agent:

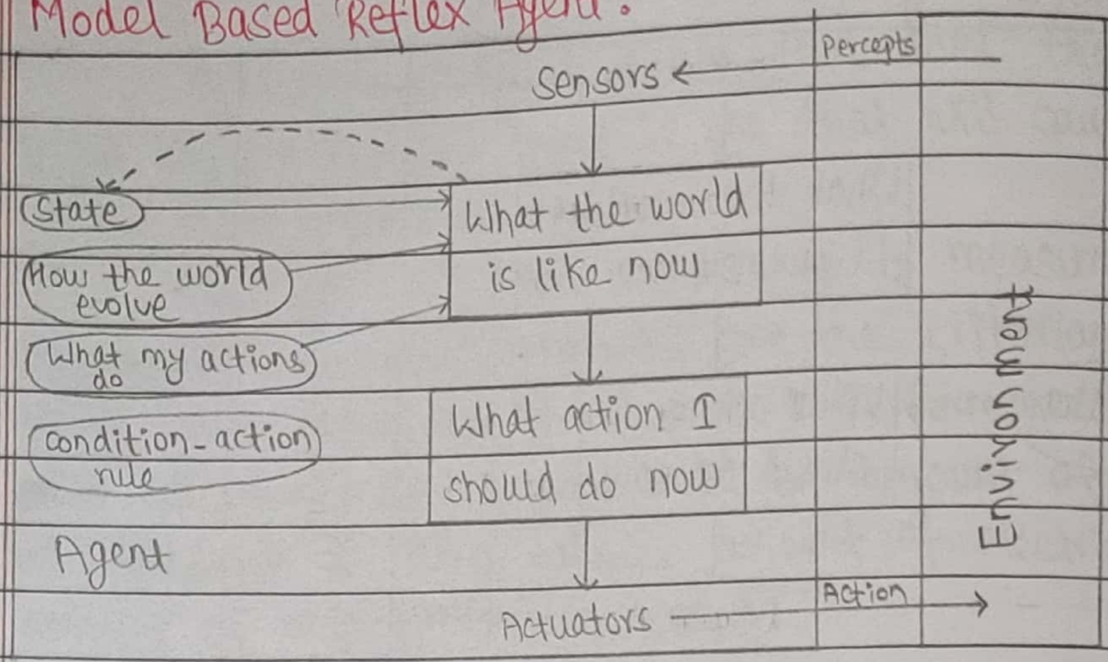


Fig:- Model Based Reflex Agent

It keeps track of the current state of the world using an internal model. It then chooses an action in the same way as the simple reflex agent. A model based agent can handle partially observable environment. The knowledge about how the world works is called the model of the world (internal model). A model based reflex agent should maintain some sort of internal model that depends on the percept history and thereby reflects at least some of the unobserved aspects of the current state.

For example: Marse lander after picking up its first sample, it stores this in the internal state of world around it so when it come across the second same sample it passes it by and saves space for other sample.

3) Goal Based Agent:

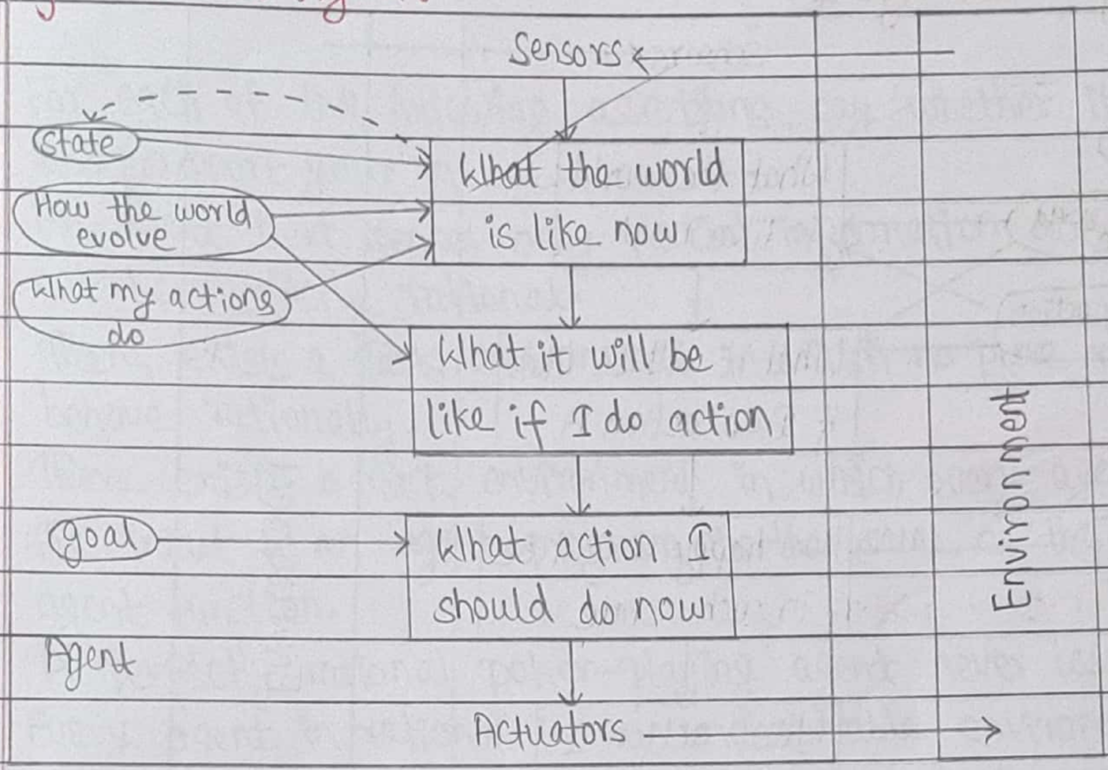


Fig:- Goal Based Agent

Goal Based Agent further expand on the capabilities of the model based agent by using goal information. It keeps track of the world state as well as a set of the world it is trying to achieve and choose an action that will lead to the achievement of its goal. Goal information describes situations that are desirable. This allows the agent a way to choose among multiple possibilities selecting the one which research a goal state.

For example:

An agent who want to travel from A city to E city choosing shortest path.

Goal :- Travel from A to E in shortest path, goals can be achieved by problem solving which states, what action can help us achieving these goals.

4) Utility Based Agent:

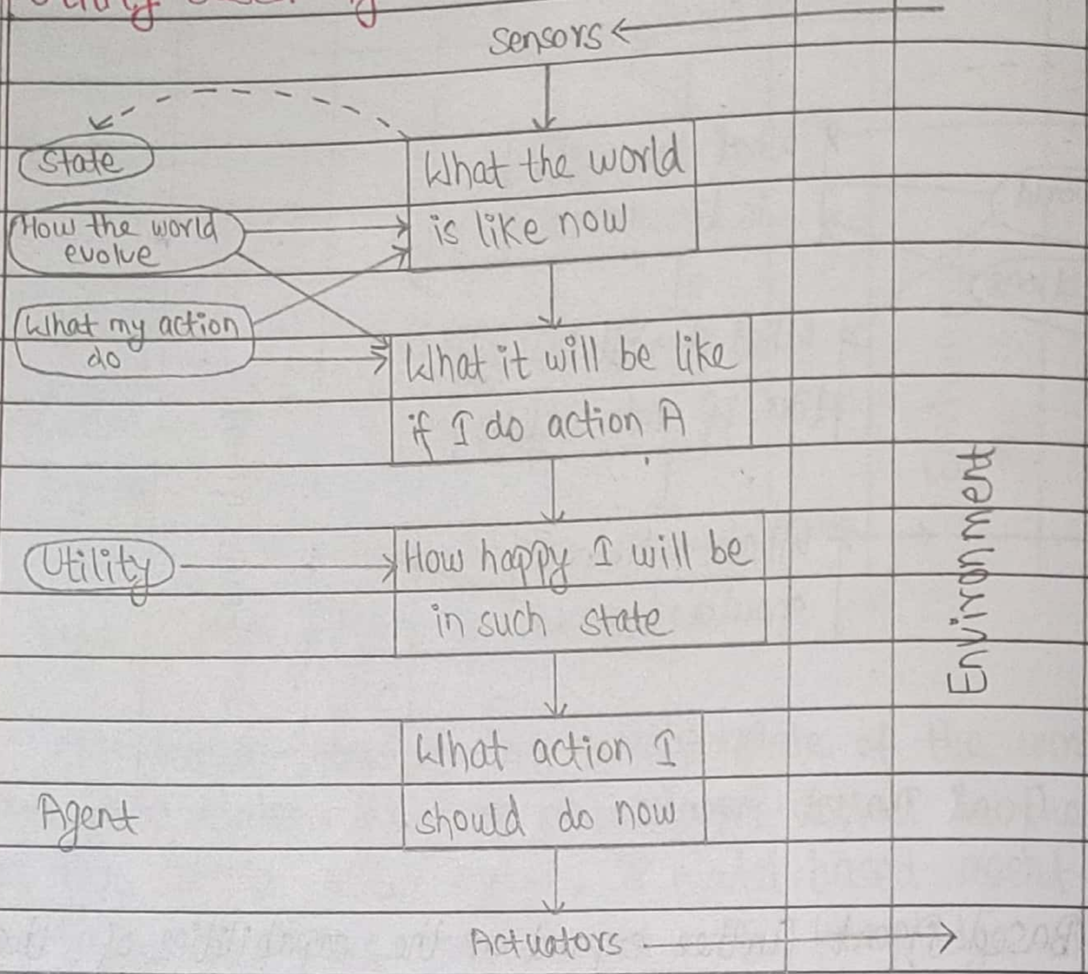


Fig:- Utility Based Agent

It uses a model of the world along with utility function that measure its performance among state of the world. They ~~Need~~ choose the action that leads to the best expected utility where expected utility is computed by averaging overall possible outcomes states waited by the probability of the outcome.

Goal based agent only distinguished between goal state and non-goal state. It is possible to define a performance measure of how desirable a particular state is. This measure can be obtained through the use of utility function.

Q. For each of the following assertions, say whether it is true or false and support your answer.

1) An agent that senses only partial information about the state can not be perfectly rational.
2) There exists a Task environment in which no pure reflex agent can behave rationally.
3) There exists a Task environment in which every agent is rational.
4) The input to an agent program is the same as the input to the agent function.
5) A perfectly rational poker-playing agent never loses.
6) Every agent in rational in an unobservable environment.
7) It is possible for a given agent to be perfectly rational in two distinct task environment.

UNIT 2:
Problem Solving

Problem Solving:

Problem solving, particularly in artificial intelligence, may be characterized as a systematic search through a range of possible actions in order to reach some predefined goal or solution. Problem-solving methods divide into special purpose and general purpose. A special-purpose method is tailor-made for a particular problem and often exploits very specific features of the situation in which the problem is embedded. In contrast, a general purpose method is applicable to a wide variety of problems. One general-purpose technique used in AI is means-end analysis — a step-by-step, or incremental, reduction of the difference between the current state and the final goal.

Four general steps in problem solving:

i) Goal formulation:
- What are the successful world states.

ii) Problem formulation:
- What actions and states to consider given the goal

iii) Search:
- Determine the possible sequence of actions that lead to the states of known values and then choosing the best sequence.

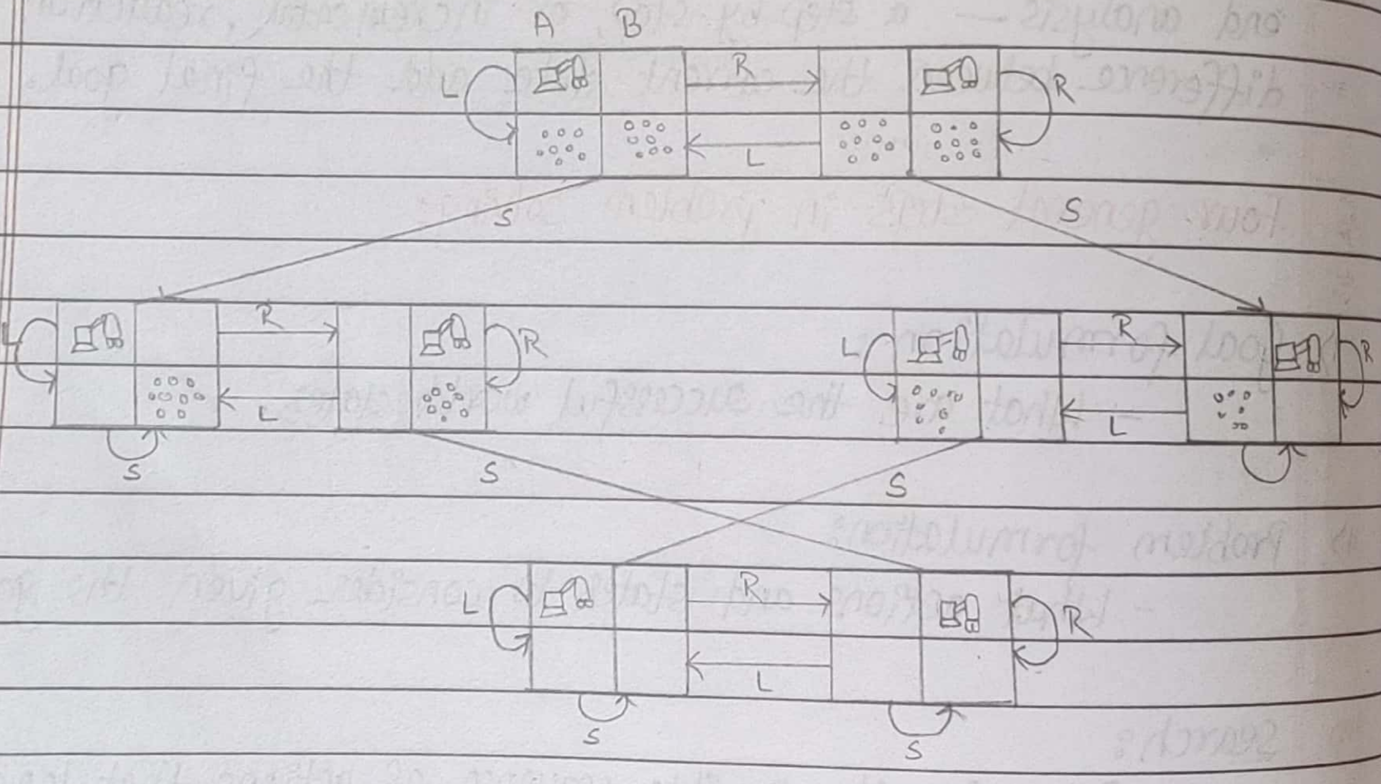
iv) Execute:
- Give the solution perform the actions.

Problem formulation:

A problem is defined by:-
- → An initial state : State from which agent start.
- → Successor function : Description of possible actions available to the agent.
- → Goal test : Determine whether the given state is goal state or not.
- → Path cost : Sum of cost of each path from initial state to the given state

State space Representation : set of state for a given particular problem

$n \times 2^n$

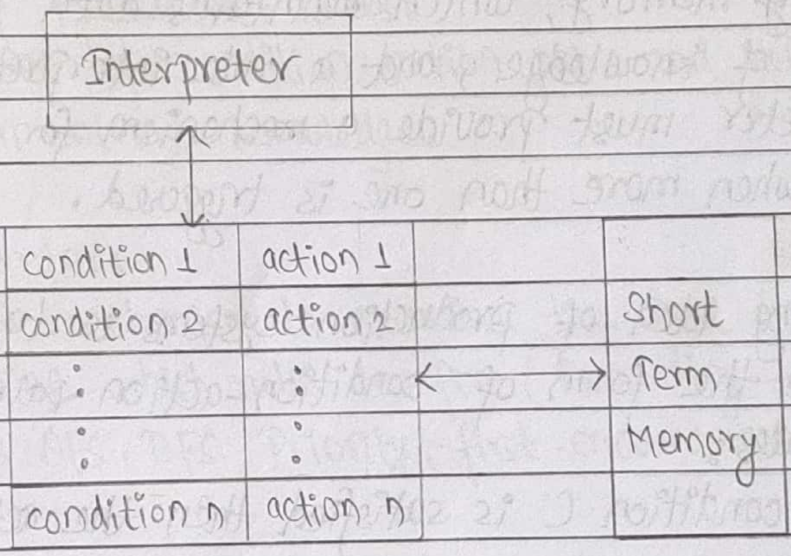


- The set of all possible states for a given problem is known as state space representation. The state spaces is commonly defined as directed graph in which each node is a state and each arc represents the application of an operator transforming a state to a successor state.

  A solution is a path from the initial state to a goal state.

<u>States</u>: two locations with or without dirt : i.e. $2 \times 2^2 = 8$ states
<u>Initial state</u> : any state can be initial state.
<u>Actions</u> : {Left, Right, Suck}
<u>Goal test</u> : Check whether squares are clean.
<u>Path cost</u> : Number of actions to reach goal.

Production System: (Production rule system)



Rule Base

A production system (or production rule system) is a computer program typically used to provide some form of artificial intelligence, which consits primarily of a set of rules about behaviour. These rules, termed productions, are a basic representation found useful in automated planning, expert systems and action selection. A production system provides the mechanism necessary to execute productions in order to achieve some goal for the system.

Productions consist of two parts: a sensory precondition (or "IF" statement) and an action (or "THEN"). If a production's precondition matches the current state of the world, then production is said to be triggered. If a production's action is executed, it is said to have fired. A production system also contains a database, sometimes called working memory, which maintains data about current state of knowledge, and a rule interpreter. The rule interpreter must provide a mechanism for prioritizing productions when more than one is triggered.

The underlying idea of production systems is to represent knowledge in the form of condition-action pairs called production rules.

If the condition C is satisfied then the action A is appropriate.

Types of production rules

Situation-action rules:

If it is raining then open the umbrella.

Inference rules:

If Cesar is a man then Cesar is a person.

Production system is also called ruled-based system.

Architecture of Production System:

Short Term Memory:

- Contains the description of the current state.

Set of Production Rules:

- Set of condition-action pairs and defines a single chunk of problem solving knowledge.

Interpreter:

- A mechanism to examine the short term memory and to determine which rules to fire (According to some strategies such as DFS, BFS, Priority, first-encounter, etc).

{ Block diagram } - drawn before.

The execution of a production system can be defined as a series of recognize-act cycles:
Match – memory contain matched against condition of production rules, this produces a subset of production called conflict set. Conflict resolution – one of the production in the conflict set is then selected, Apply the rule.

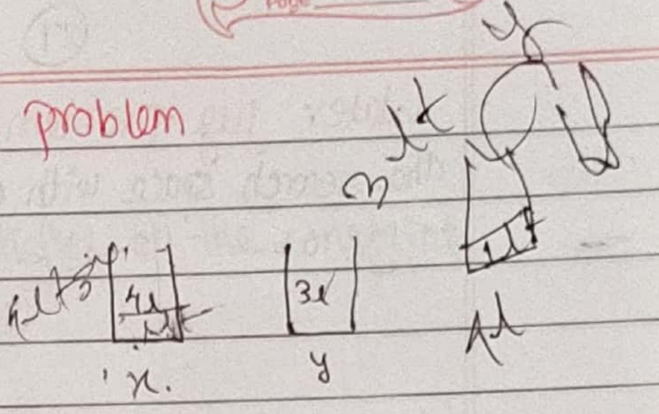
Consider an example:
Problem : Sorting a string composed of letters a, b & c.
Short Term Memory : cbaca
Production set :

1. ba → ab
2. ca → ac
3. cb → bc

Interpreter : Choose one rule according to some strategy.

Iteration	Memory	Conflict set	Rule fired
0	cbaca	1,2,3	1
1	cabca	2	2
2	acbca	2,3	2
3	acbac	1,3	1
4	acabc	2	2
5	aacbc	3	3
6	aabcc	ϕ	halt.

Production system : The water jug problem



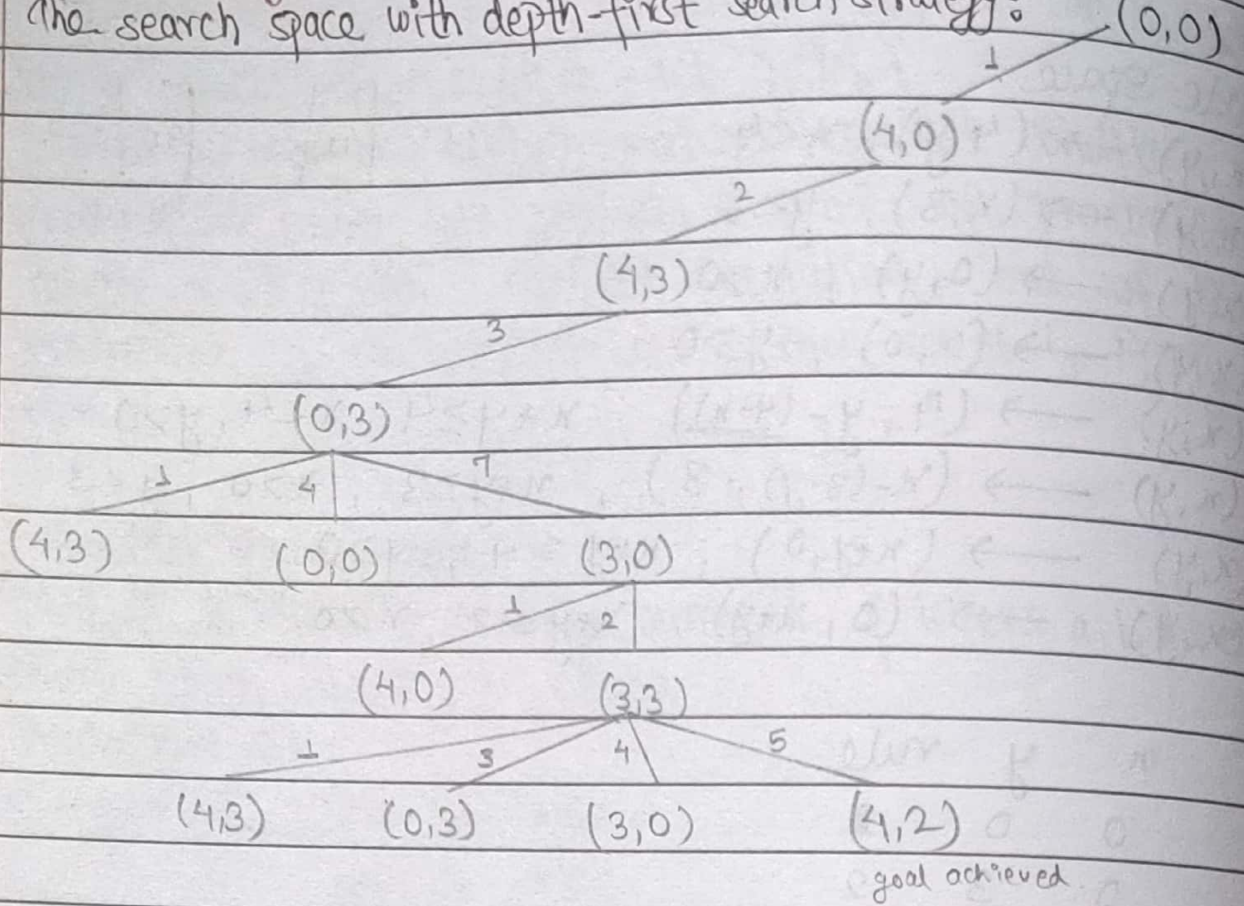
State space

1. $(x,y) \rightarrow (4,y)$, $x<4$
2. $(x,y) \rightarrow (x,3)$, $y<3$
3. $(x,y) \rightarrow (0,y)$, $x>0$
4. $(x,y) \rightarrow (x,0)$, $y>0$
5. $(x,y) \rightarrow (4, y-(4-x))$, $x+y \geq 4$, $x<4$, $y>0$
6. $(x,y) \rightarrow (x-(3-y), 3)$, $x+y \geq 3$, $x>0$, $y<3$
7. $(x,y) \rightarrow (x+y, 0)$, $x+y \leq 4$, $y>0$
8. $(x,y) \rightarrow (0, x+y)$, $x+y \leq 3$, $x>0$. Empty the 4-gallon jug into 3-gallon jug.

x	y	rule	
0	0	2	
0	3	2	
3	0	7	
3	3	2	
4	2	5	final goal.
0	2	3	
2	0	7	

Water jug problem representation:

The search space with depth-first search strategy:



generalized algorithm hudaina.

Constraint Satisfaction Problem (CSP):

- A set of variables: $\{x_1, x_2, \ldots, x_n\}$
- A set of constraints: $\{C_1, C_2, \ldots C_k\}$
- Non-empty set of domains: $\{D_1, D_2, \ldots, D_n\}$

(X, D, C)

variables domains constraints (rule)

$C : \langle t, R \rangle$

$t \rightarrow$ a set of tuple of variables

$R \rightarrow$ set of tuples of values

$v : X \longrightarrow D$

$(v(x_1), v(x_2), \ldots, v(x_n)) \in R$

- A solution is an evaluation that satisfies all the constraints.

Example:

N - Queen Problem

4x4

Variable : q_1, q_2, q_3, q_4
Domain : 1, 2, 3, 4
Constraints : no two queens in the same column or same diagonal

Place N queens on a N×N chess board so that queen can't attack any other queen.

- No queen can attack any other queen.
- Given any two queens Q_i and Q_j they cannot attack each other.
- Now we translate each of these individual conditions into a separate constraint.

	q_1		
			q_2
q_3			
		q_4	

Variable : q_1, q_2, q_3, q_4
Domain : 1, 2, 3, 4
Constraints : no two queens in the same column or same diagonal.

C

Searching
Application:

Search problems are a part of large numbers of real world problem.

VLSI layout:- Position millions of component and connections on a chip to minimize area.

Path Planning :

Robot Navigation: Special case of route finding for robot with no specific roots or connections, where state space and action space are potentially infinite. ~~There are~~

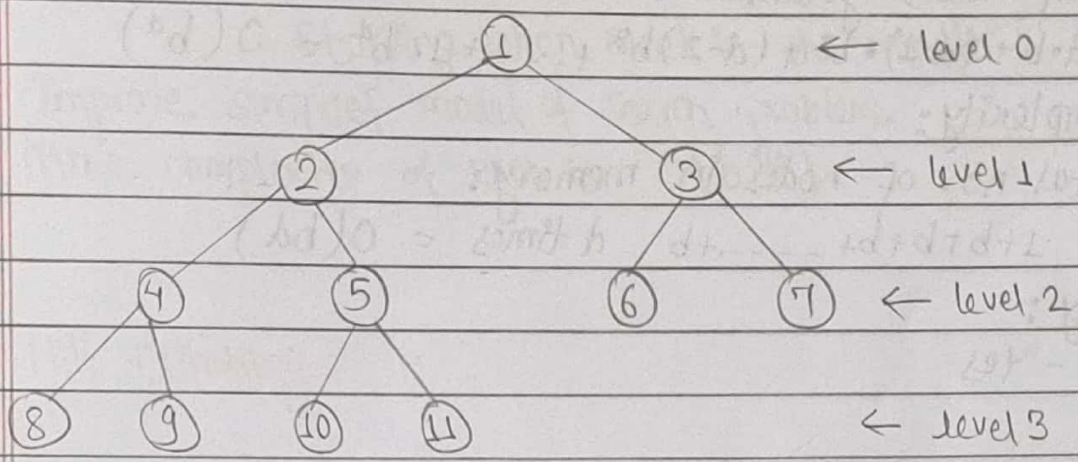
There are two broad categories classes of search methods:

1) Uninformed Search (Blind search): is a searching technique which have no additional information about the distance from the current state to the goal.

ii) Informed search: is an another technique which have additional information about the estimate distance from the current state to the goal.

Iterative Deepening Search (IDS):-

- Combination of BFS + DFS
- Advantage of BFS : It will find a goal node at any way.
- Disadvantage of BFS : It consume very much ~~money~~ memory.
- Advantage of DFS : It consume less memory.



Level	IDS
0	1
1	1,2,3
2	1,2,4,5,3,6,7
3	1,2,4,8,9,5,10,11,3,6,7

ID Search evaluation:

- Completeness :
  - YES (no infinite paths)
- Time complexity:
  * costly : repeated states
  * node generation:
    level d : once

level $d-1$: 2

level $d-2$: 3

....

level 2 : $d-1$

level 1 : d

- Total no. of nodes generated:

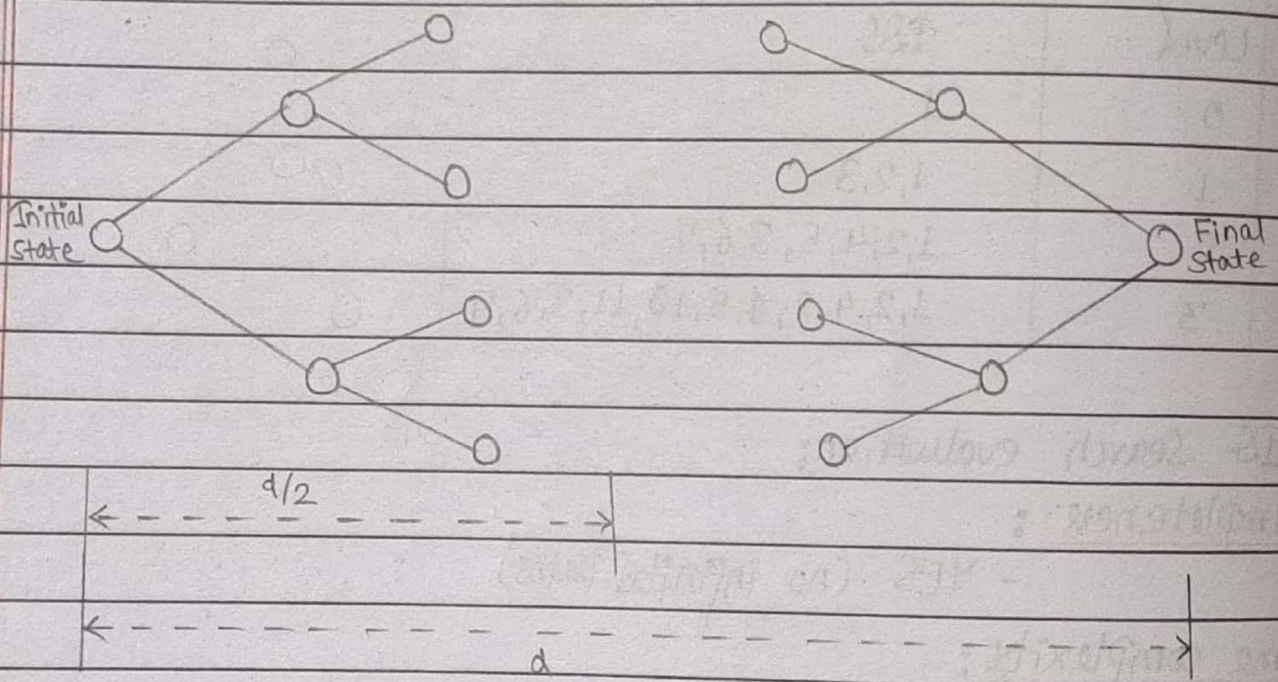
  $$d \cdot b + (d-1) \cdot b^2 + (d-2) b^3 + \dots + 1 \cdot b^d = O(b^d)$$

- Space complexity:
  - Total no. of nodes in memory:

    $$1 + b + b + b + \dots + b \quad d \text{ times} = O(bd)$$

- Optimality:
  - Yes

Bidirectional search:



- Graph search algorithm find shortest path from initial vertex to a goal vertex.
- It runs two simultaneous searches. One forward from initial state and one backward from goal.
- Stopping criteria:
  Stopping when meet in the middle.
- Improve simplief model of search problem.
- Time complexity of each is $O(b^{d/2})$.

Hill Climbing:

Algorithm

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

Step 2: Loop until the solution is found or there are no new operators (particular nodes or states) left.

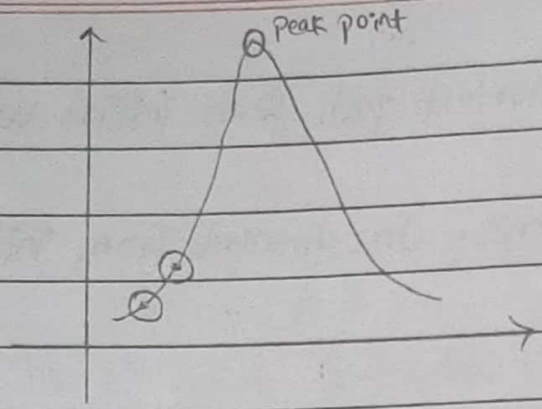
Step 3: Select and apply new operators.

Step 4: Evaluate new state.

i) If it is a goal state, then quit.

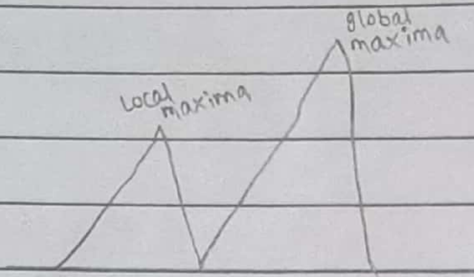
ii) If it is better than current state then make it new current state.

iii) If it is not better than current then go to step 2.

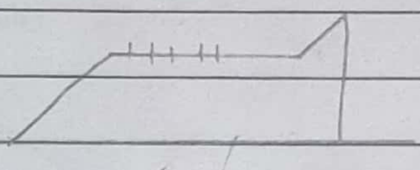


Limitation:

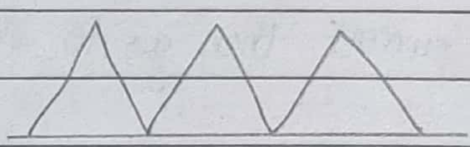
1) Local maxima



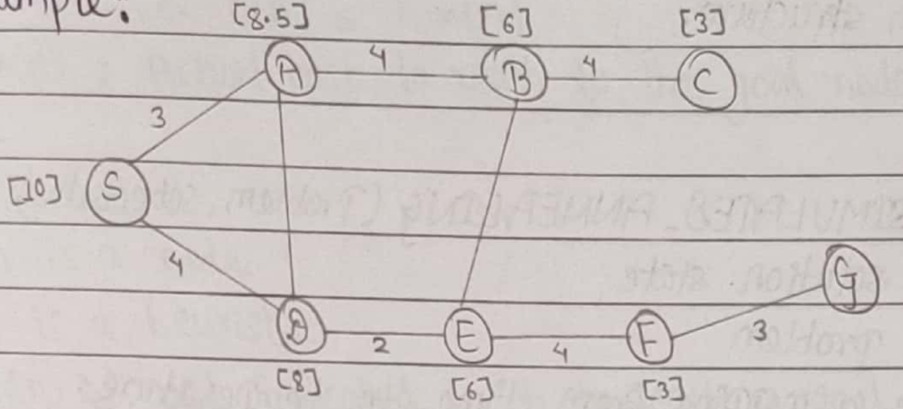
2) Plateau

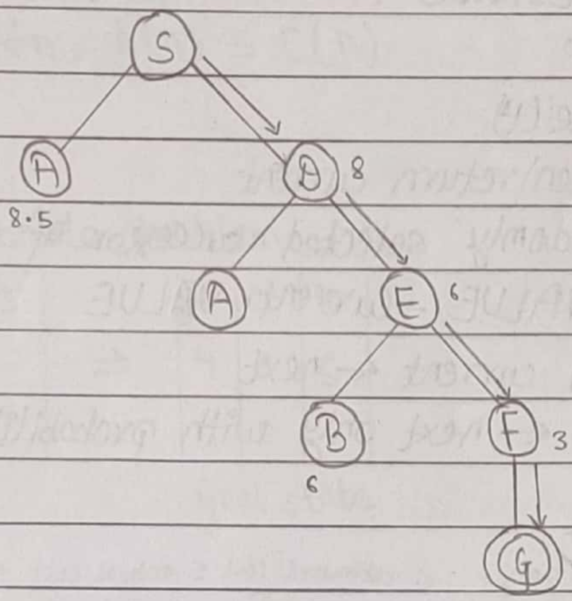


3) Ridge



Example:





Simulated Annealing (SA):

- Hill climbing without downhill is incomplete, because it can get stuck on a local maxima.
- Pure random walk is complete but inefficient.
- Combination of hill climbing and random walk yields both completeness and efficiency.
- Simulated Annealing is such as algorithm
- Annealing is the process in which material is heated and slowly cooled

into a uniform structure.

Pseudocode:

```
Function SIMULATED_ANNEALING (Problem, schedule)
    return solution state
    inputs: problem
    schedule: a mapping from time to temperature.

current ← MAKE_NODE (Problem. INITIAL_STATE)
    for t=1 to ∞
    T ← schedule (t)
    if T = 0, then return current
    next ← a randomly selected successor of current.
    ΔE ← next.VALUE - current.VALUE
    if ΔE >0 then current ← next
    else current ← next only with probability e^(ΔE/T).
```

Actual cost vanda kam huna paryo i.e. estimated cost ≤ actual cost huna paryo

Admissible Heuristic:

- The purpose of heuristic function is to guide the search process in the most profitable path among all that are available.
- A heuristic function at a node n is an estimate of the optimum cost from the current node to a goal. It is denoted by h(n).

  h(n): estimated cost of the cheapest path from node n to a goal node.
- A heuristic function is said to be admissible heuristic if it never overestimates the cost to reach to the goal.

i.e. $h(n) \leq h*(n)$

$h*(n)$: Actual cost to reach to the goal node from node n.

Formulation

- n is a node
- h is a heuristic
- h(n) is cost indicated by h to reach a goal from n.
- C(n) is the actual cost to reach a goal from n.
- h is admissible if

$$\forall n,\ h(n) \leq C(n).$$

Example:

8-puzzle problem.

1	2	3
4	8	-
7	6	5

Start state

$\Rightarrow$

1	2	3
4	5	6
7	8	

Goal state

1) {(1,2,3), (4,8,0), (7,6,5)}
2) {(1,2,3), (4,8,5), (7,6,0)}
3) {(1,2,3), (4,8,5), (7,0,6)}
4) {(1,2,3), (4,0,5), (7,8,6)}
5) {(1,2,3), (4,5,0), (7,8,6)}
6) {(1,2,3), (4,5,6), (7,8,0)}

Example:

7	2	4
5	-	6
8	3	1

Start state

-	1	2
3	4	5
6	7	8

Goal state

$h_1(S)$: Total no. of moves = 8

$h_2(S)$: Sum of distance of the tiles from their goal position (not diagonal).

$= 3+1+2+2+2+3+3+2 = 18$

$h_n(S) = \max\{h_1(S), h_2(S)\} = 18$

A* search :-

Properties:

- Consistency (Monotonicity)

A heuristic is consistent if:

$h(n) \leq d(n, n') + h(n')$ [Note: Suppose n' is successor of n.

where,

$h(n)$ = Estimated cost to reach to the goal node from node n.

$d(n, n')$ = actual cost from n to n'.

Prove: If $h(n)$ is consistent, then the values of $f(n)$ along the path are non-decreasing.

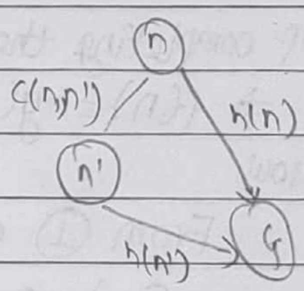
$f(n')$ = estimated distance from start to goal through n'.
= actual distance from start to n ($g(n)$) + step cost from n to n' ($cost(n, n')$) + estimated distance from n' to goal ($h(n')$).

$= g(n) + cost(n,n') + h(n')$ {By consistency: $cost(n,n') + h(n') \geq h(n)$}
$\geq g(n) + h(n)$
$= f(n)$

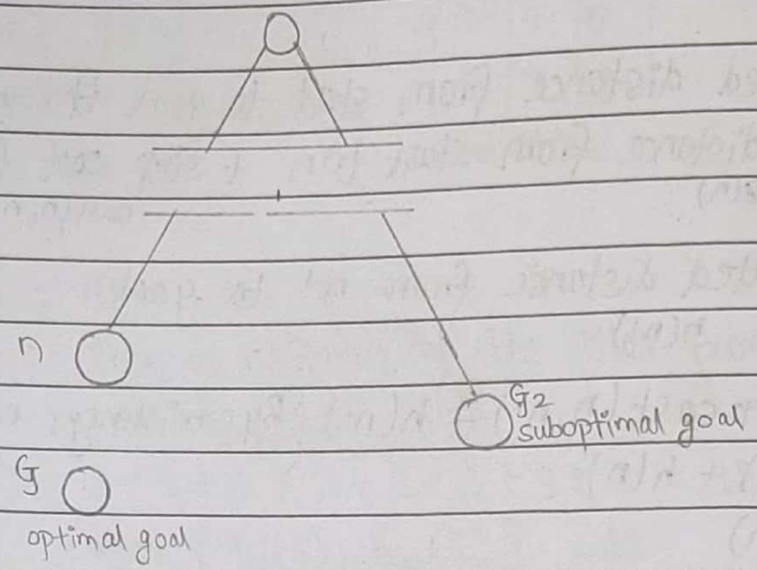
$\therefore f(n') \geq f(n)$. So, $f(n)$ never decreases along a path.

Example:-

S
⋮
n $f(n) = g(n) + h(n) = 7$
$c(n,n') = 1$ decrease
n' $f(n') = g(n') + h(n') = 6$
⋮
G



Prove that: A* search gives optimal solution when the heuristic is admissible.



C^* : cost of the optimal goal

$f(G_2) = h(G_2) + g(G_2)$
$f(G_2) = g(G_2)$ since $h(G_2) = 0$.
$f(G_2) > C^*$ —— (1)

Again,
Since $h(n)$ is admissible, it does not overestimates the cost of completing the solution path.
$\Rightarrow f(n) = g(n) + h(n) \leq C^*$ —— (2)

Now,
From (1) and (2)
$f(n) \leq C^* < f(G_2)$

Since, $f(G_2) > f(n)$, A* will never select for expansion. Thus, A* gives up optimal solution when heuristic function is admissible.

Game Search

An exemplary game: Tic-tac-toe

- There are two players denoted by X and O.
- They are alternatively writing their letter in one of the 9 cells of a 3 by 3 board.
- The winner is one who succeeds in writing three letters in line.
- The game begins with an empty an empty board. It ends in a win for one player and a loss for the other, or possibly in a draw.
- Terminal states are those representing a win for X, loss for X, or a draw.

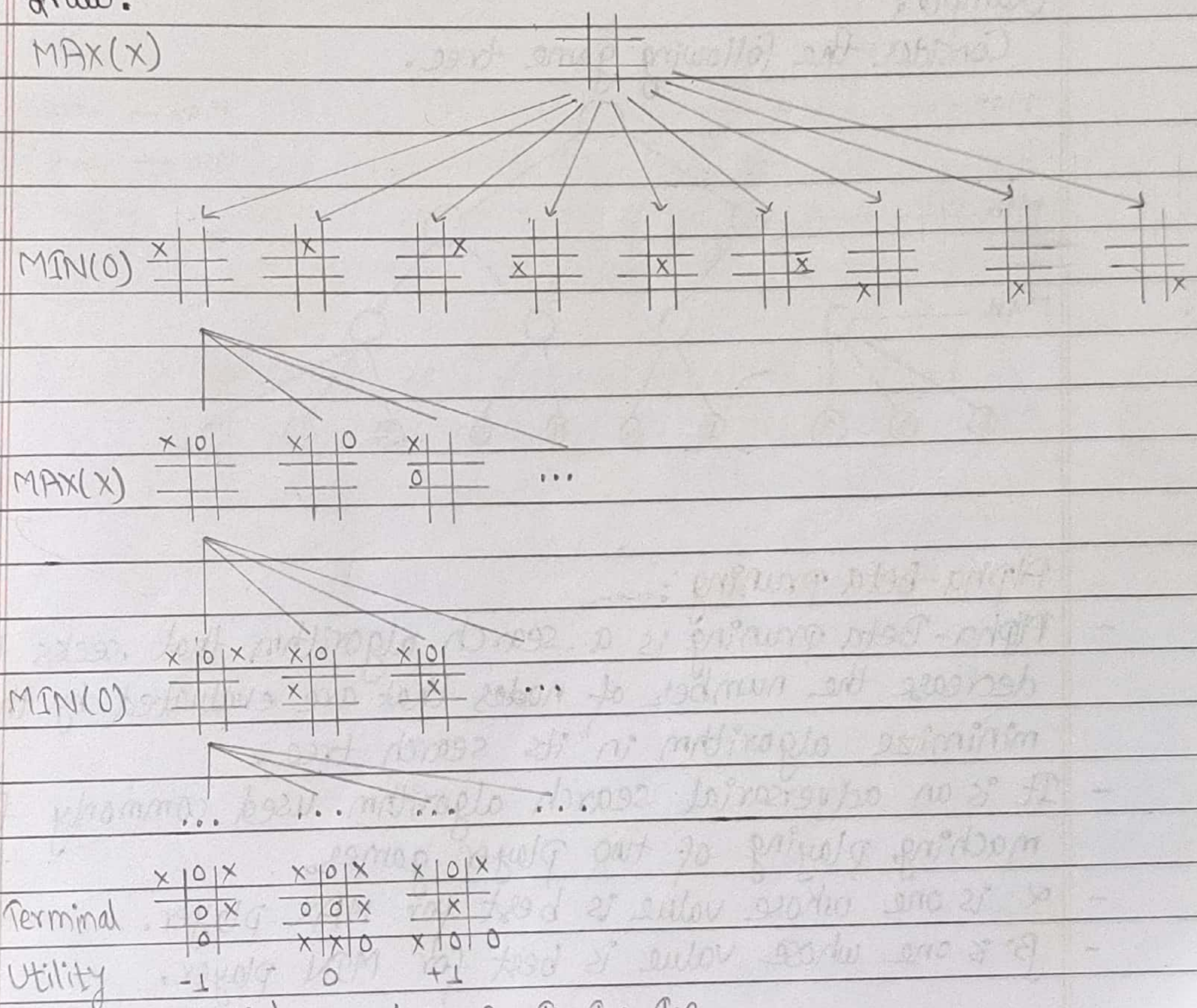


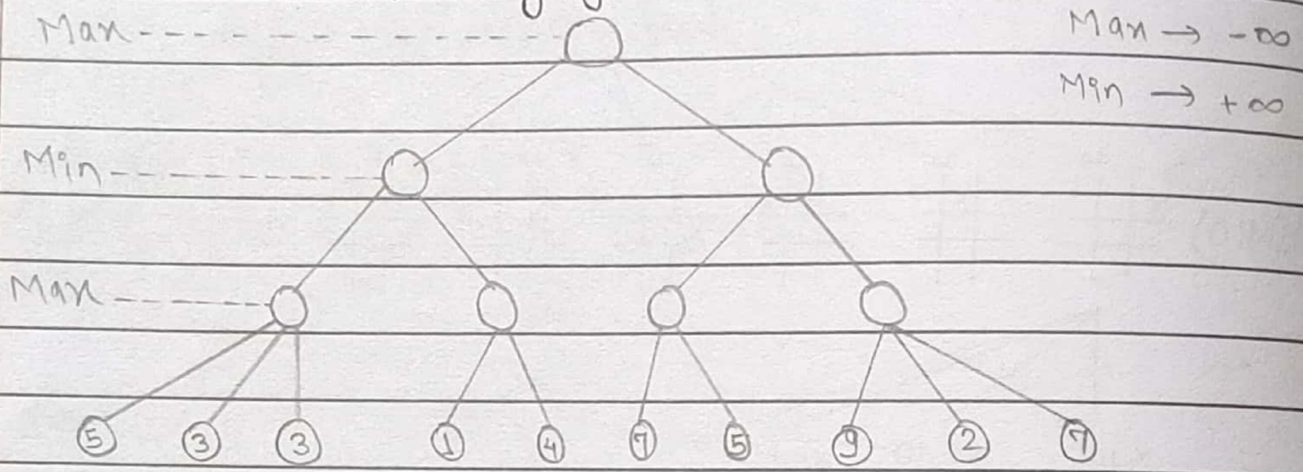
Fig: Partial game tree for Tic-Tac-Toe.

Minimax Algorithm

- Minimax is a procedure used for minimizing the possible loss while maximizing the potential gain.
- Originally formulated for two players game like chess, checkers, tic-tac-toe
- Consider two player games, the players are reffered to as MAX (the player) and MIN (the opponent).
- MAX is the player trying maximize its score and MIN is the opponent trying to minimize MAX's score.

Example:

Consider the following game tree.



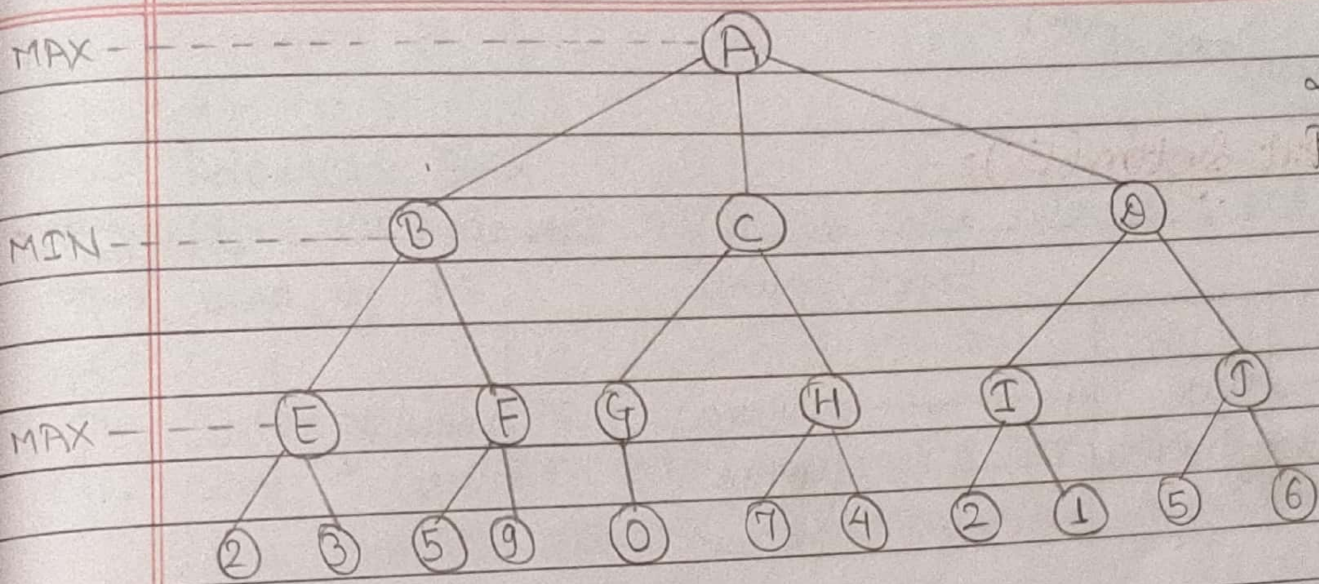
Alpha-beta pruning :-

- Alpha-Beta pruning is a search algorithm that seeks to decrease the number of nodes that are evaluated by the minimax algorithm in its search tree.
- It is an adversarial search algorithm used commonly for maching playing of two player games.
- α is one whose value is best for MAX player.
- β is one whose value is best for MIN player.

Condition: $\alpha \geq \beta$

$\alpha(MAX) = -\infty$

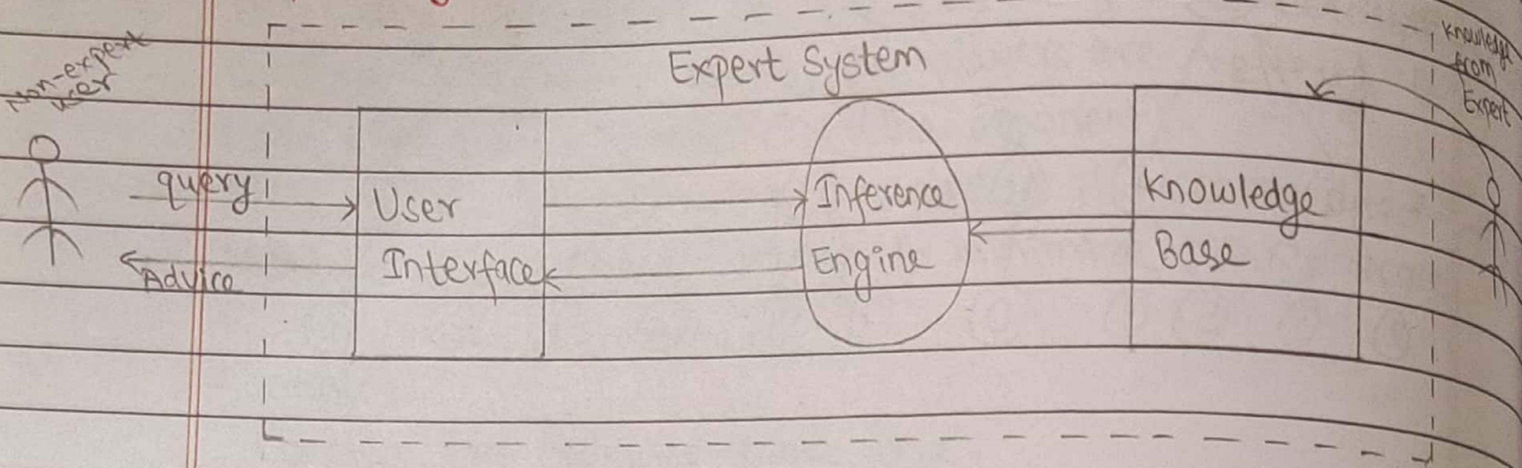
$\beta(MIN) = +\infty$



Application of Artificial Intelligence

(Knowledge Base System)
OR

Expert System (ES):-



- The ES are computer applications developed solve complex problems in a particular domain, at the level of extra-ordinary human intelligence and experties.

- An ES is a computer system that emulates the decision-making capabilities of human expert.

Characteristics of ES:

- High performance
- Understandable
- Reliable
- Highly Responsive

Components of ES:

- Knowledge Base
    - Store all relevent information, data, rules, cases and relationship used by ES.

- Inference Engine
    - Seeks information and relationship from the knowledge base and provides answers, predictions, and suggestions in the way a human expert.

- User Interface
    - The component of ES that communicate with the user.
    - bidirectional.

Capabilities of ES:

- Interpreting results
- Predicting results
- Explaining
- Deriving a solution
- Justifying the conclusion.

Applications:

- Design domain : camera lens design, automobile design
- Medical domain : Diagnosis systems to deduce cause of diseases from observed data.
- Monitoring system : Comparing data with observed system.

- Knowledge domain: Find out fault in vehicles, computers.
- Finance / Commerce: Detection of possible fraud, stock market trading, Airline scheduling, cargo scheduling

Benefits:
- Availability: They are easily available due to mass production of software.
- Speed: They offer great speed.
- Less error rate: Error is low as compared to human error.
- Reducing risk: They can work in the environment dangerous to human.
- Steady Response: They work steadily without getting emotional, tensed.

Disadvantages:
- Cost to buy and setup the system.
- ES have no common sense.
- Not widely used for tested.

Why Expert System:
- The problem cannot be specified in terms of well defined algorithms.
- When the task is hazardous
- There is scarcity of experts in the area.

Expert System development Team

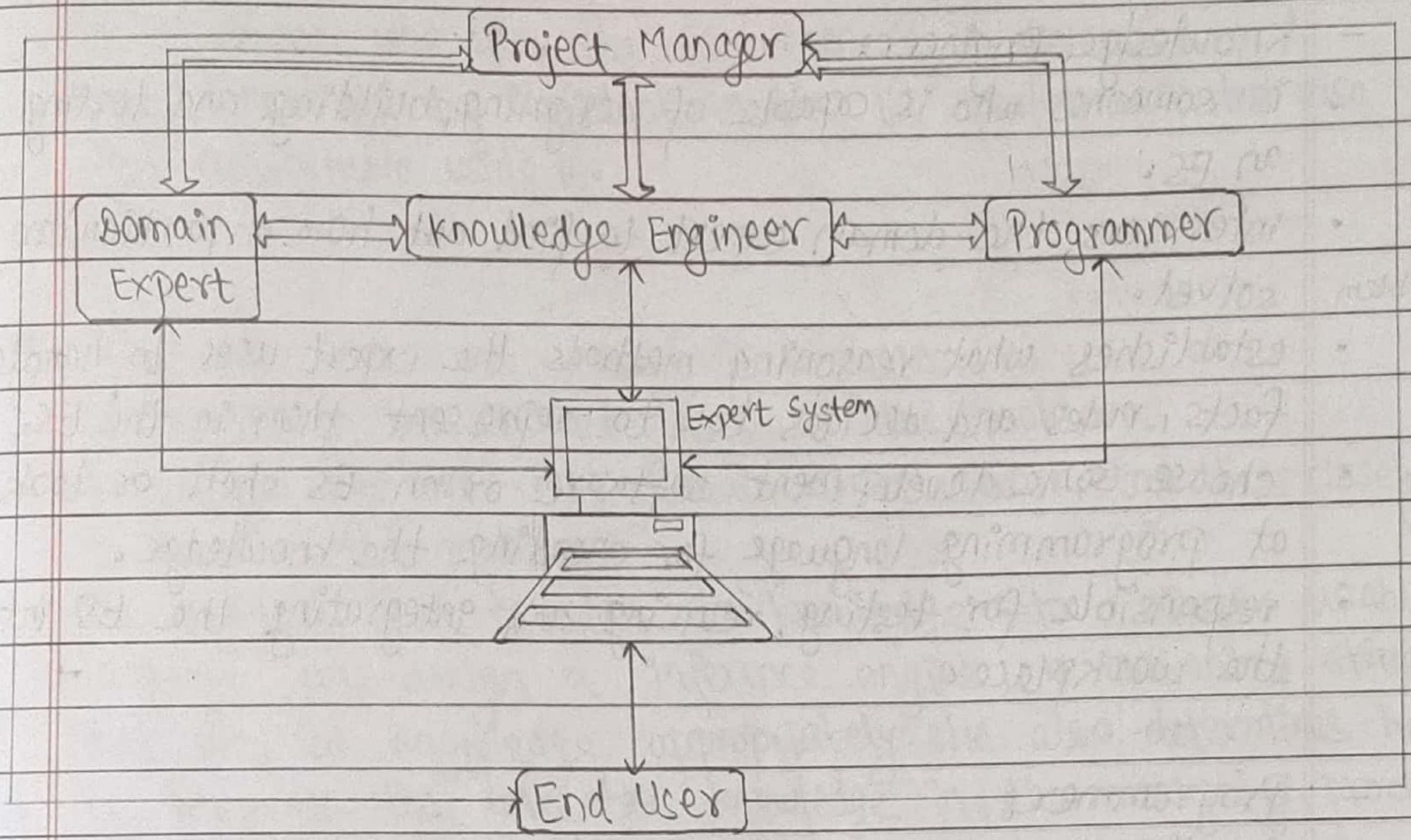


Fig:- The main players of the expert system development team.

There are five players of the expert system (ES) development team.

- Domain Expert:
  - is a knowledgeable and skilled person capable of solving problems in a specific area or domain.
  - the person's experties is to capture in the ES.
  - could be more than one expert that contribute to the ES.
  - the expert must be able to communicate his or her knowledge, be willing to participate in the ES development and commit a substantial amount of time to the project.

classmate
Date
Page

- is the most important person in the ES development team.

- Knowledge Engineer:
  - is someone who is capable of designing, building and testing an ES.
  - interviews the domain expert to find out how a particular problem solved.
  - establishes what reasoning methods the expert uses to handle facts, rules and decides how to represent them in the ES.
  - choose some development software or an ES shell or look at programming language for encoding the knowledge.
  - responsible for testing, revising and integrating the ES into the workplace.

- Programmer:
  - is the person responsible for the actual programming, describing the domain knowledge in terms of that a computer can understand.
  - needs have the skill in symbolic programming in such AI language such as prolog, LISP, etc.
  - should also know conventional programming language like C, Pascal, FORTRAN, etc.

- Project Manager:
  - is the leader of ES development team, responsible for keeping the project on track.
  - makes sure that all the deliverables and milestones are met, interacts with the expert knowledge engineer, programmer and end user.

- End user:
  - often called the user
  - is a person who uses the ES when it is developed.
  - must not only be confident in the ES performance but also feel comfortable using it.

2069

How can you construct Expert System?

- Determining the characteristics of the problem.
- Knowledge engineer and domain experts work together closely to describe the problem.
- The engineer then translate the knowledge into computer usable language, and design a inference engine, a reasoning structure that uses the knowledge appropriately. He also determines how to integrate the use of uncertain knowledge in the reasoning process and what ~~kno~~ kind of explanation would be useful together to the end user.
- When expert system is implemented, it may be
  - the inference engine is not just right.
  - form a representation of knowledge is awkward.
- An expert system is judged to be entirely successful when it operates on the level of a human expert.

NLP (Natural Language Processing):-

- Language: representing your understanding.
- Natural language: Language spoken by people, e.g. English, Nepali, Spanish, etc as opposed to artificial language, like C++, Java, etc.
- NLP is the branch of computer science focused on developing system that allow computer to communicate with people using everyday language.
- NLP is related to human-computer interaction.
- NLP is sub-field of AI. Devoted to make computers "understand" statements written in human languages.

Categories:

Phonology: Study of speech sound.
Morphology: Study of meaningful components of words.
Syntax: Study of structural relationship between words.
Semantics: Study of meaning.

Why NLP?

- Computers "see" text in english the same way we have seen in previous text.
- People have no trouble understanding language
  - Common sense knowledge
  - Reasoning capacity
  - Experience
- Computers have
  - no common sense knowledge
  - no reasoning capacity

That's why we need NLP.

Scanned with CamScanner

Components of NLP:

1) Natural Language Understanding

* Taking some spoken or typed sentence and working out what it means.

or,

What does the written / spoken sentence means?

* different level of analysis required
  - Morphological analysis
  - Syntatic analysis
  - Semantic analysis
  - Discourse analysis

* A language can be very ambiguous (different meaning of same sentence).

i) Lexical ambiguity: It means word level ambiguity { 'noun' or 'verb' }
eg: Orange ⇒ (colour, fruit)

ii) Syntatical ambiguity: It is about parsing the ambiguity (grammatical ambiguity)
eg: I can write → aux. verb, verb or noun.

iii) Referential ambiguity: Meaning is not well referred from the sentence.
eg: MEERA went GEETA and said 'I am hungry'.
- Here, who is hungry is not clear from the sentence.

2) **Natural Language Generation**

- Taking some formal representation of what you want to say and working out a way to express it in a natural (human) language. eg: English.
- The process included in this are:
  - *. **Text planning** :- It includes the extracting knowledge from knowledge base.
  - *. **Sentence planning** :- This includes selection of correct words and forming sentence which follow the grammer.
  - *. **Text Realization** :- Mapping the planned sentence into reality.

2014

Steps in NLP (Natural Language Processing):

Morphological or Lexical Analysis

↓

Syntatic Analysis

↓

Semantic Analysis

↓

Discouser Analysis

↓

Pragmatic Analysis

i) Lexical Analysis:-
- It deals with the recognition and identification of structure of the sentences, phrases, words.
- It divides the paragraphs into sentences, phrases, words.

ii) Syntatical Analysis:-
- Here the sentences are parsed as noun, verbs, adjectives and other part of sentences.
- Here the grammer of the sentences is analyzed inorder to get the relationship among different words in a sentence.

iii) Semantic Analysis:-
- Here the actual meaning of the sentence is extracted from the words used.
- It checks whether the sentence makes any meaning.
- Sentences which has no meaning is rejected.
  Eg: "Colourless green ideas" has no meaning.

iv) Discosure Analysis:-
- Here the meaning of sentence is verified with sentence before it.
  Eg: "John wanted it", the word 'it' depends upon John.

v) Pragmatic Analysis:-
- Here the sentences are re-interpreted to verify the correctness of meaning in given context.
- Here, the real world knowledge of language is required.
  Eg: "Do you know what time it is?"

Basic Parsing Techniques:

- The first task for any NLP-based system is to read (or to parse) the text.
- Before the meaning of a sentence can be determined, the meaning of its constituent parts must be established.
- The process of determining the syntactical structure of a sentence is known as parsing.
- Parsing depends on these components of a language.
  - i) Lexicon
  - ii) Categorization
  - iii) Grammer rules.

i) Lexicon:

stench | breeze | glitter | nothing | ....
is | see | smell | shoot | feel | ....
right | left | east | south | ....
here | there | nearby | ....
Me | you | I | ....
John | Mery | Boston | ....
a | an | the
to | in | or | ....
and | or | but | ....
0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

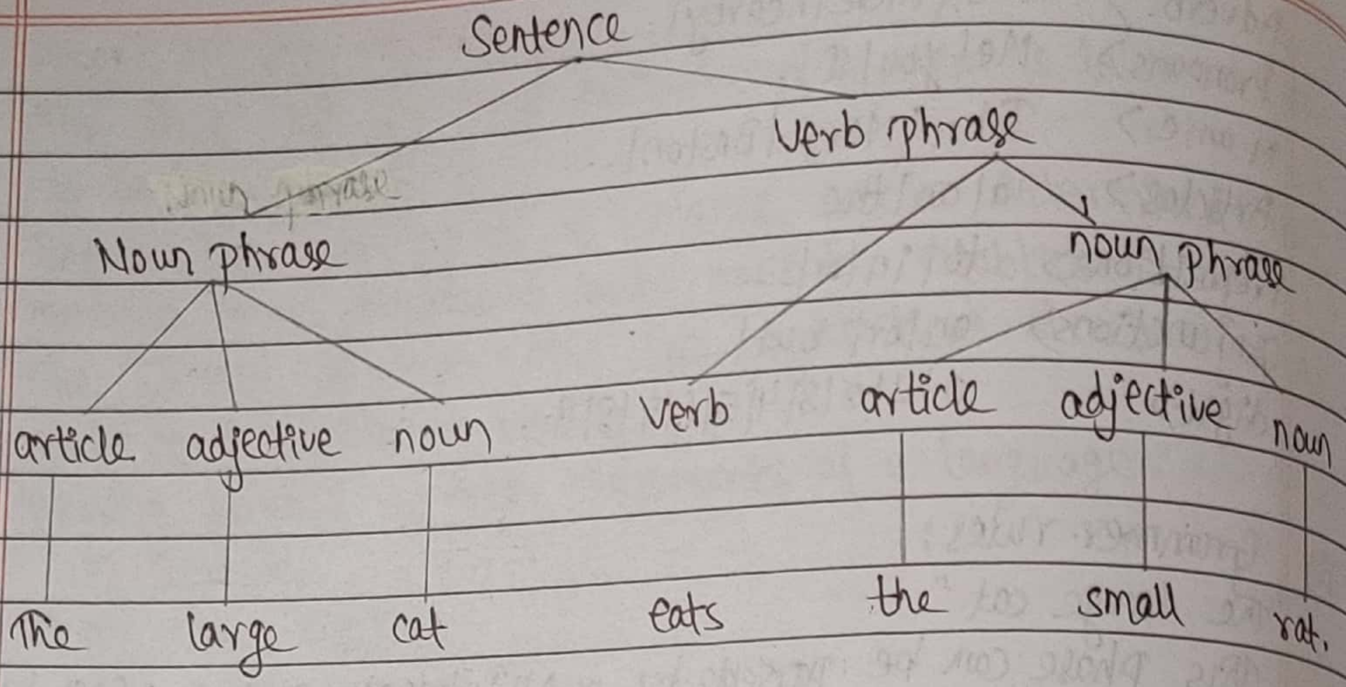
ii) Categorization:

Noun stench | breeze | glitter | nothing | ....
Verb is | see | smell | shoot | feel | ....
Adjective right | left | east | south | ....

Adverb > here | there | nearby | ...
Pronouns > Me | you | I | ...
Name > John | Mery | Boston | ...
Articles > a | an | the
Prepositions > to | in | or | ...
Conjunctions > an | or | but | ...
digit > 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9.

iii) Grammer rules:
- "The large cat"
- This phase can be parsed by a NLP-based system if it has a grammer like
  Noun Phrase (NP) → Determiner + Adjective + Verb
  (Article)
- If your system finds a phrase or sentence that has a pattern not mentioned in its set of grammer rule it won't be able to parse them.
- Parsing is the process of using grammer rules to determine whether a sentence is legal and to obtain a syntatic tree.

Syntax Tree
" The large cat eats the small rat."
[Note: Sentence ⇒ Noun Phrase (NP), Verb Phrase (VP)
NP ⇒ Article, adjective, noun
VP ⇒ Verb, NP



computer vision use garera manufacture tira goods haru products garne

Machine Vision :-

- The automatic extraction of information from digital images.
- A digital image is a representation of 2-D images as a finite set of digital values, called picture elements or pixels.
- Machine vision is the ability of a computer to "see".
- Machine vision is the application of computer vision to industry and manufacturing.

Application:

*. Semiconductor chips
*. automobiles, etc.

- A machine vision systems have two primary hardware elements : the camera, which serve as the eye of the system and a computer video analyzer.

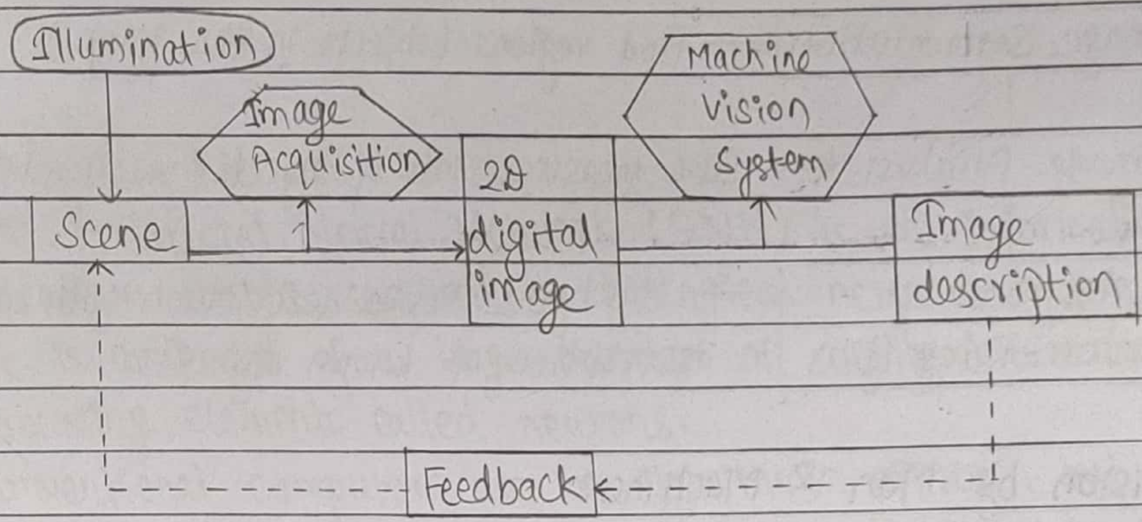
- A typical machine vision system will consists of the following components :

i) One or more digital camera or analog camera.

ii) I/O hardware or communication links.

iii) A synchronizing sensor for part detection to trigger images acquisition.

iv) A program to process images and detect relevant features.



The goal of machine vision system is to compute a meaningful description of the scene (eg: Object).

Machine Vision Stages:

Image Acquisition ← Analog to digital converter

↓

Image Processing / Image Enchancement / Image Restoration ← Remove noise/pattern, improve contrast

↓

Image Segmentation ← Find regions (objects in the image)

↓

Image Analysis ← Take measurements of objects / relationship

↓

Model Matching / Pattern Recognition ← Match the above description with similar description of known objects.

Vision by Man & Machine:

- What is the mechanism of human vision.
  - * Can a machine do the same thing?
- Human and machines have different
  - → software
  - → hardware

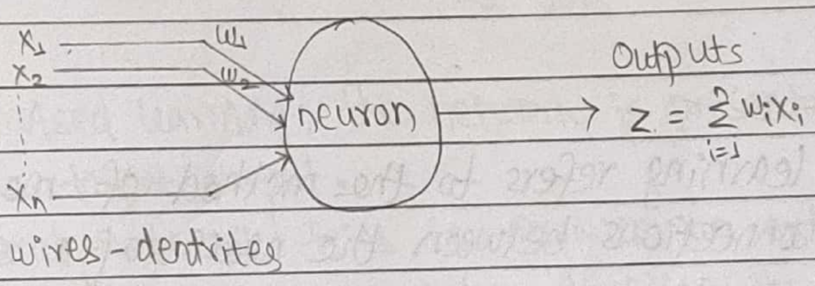
- Human "Hardware"
  - * Photoreceptors take measurements of light signals.
    - about 10^6 photoreceptors (sense organ, that responds to light falling on it).
    - retina cell transmit signal to the brain.
      - * What the neurons do? In what sequence?

- Algorithms
- Complex 3D interconnection

- Machine Vision "Hardware"
  - PC's, workstations, etc.
  - Signal: 2D-image
  - Low level processing
  - Simple interconnections.

Artificial Neural Network:-

- An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by biological nervous system.
- It is composed of a large number of highly interconnected processing elements called neurons.
- Conventional computers use an algorithmic approach but neural networks works similar to human brain and learn by example.



wires - dentrites

weight - synapses

Threshold function: activity in Soma or cell body.

Biological Neuron Model:-

Four parts of a typical nerve cell.

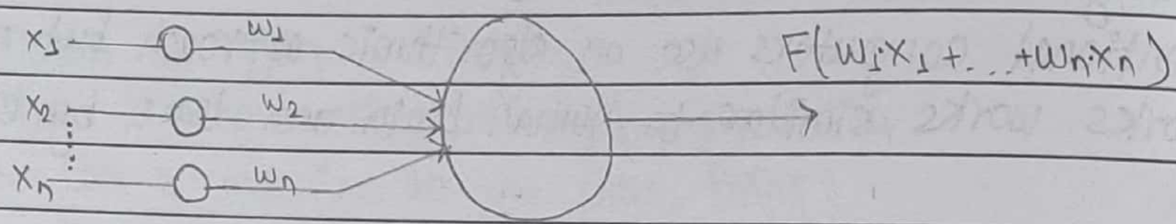
1) Dentrites: Accept the input
2) Soma (cell body): process the input
3) Axon: Turn the processed input into output.
4) Synapses: The electrochemical contact between the neurons.

Artificial Neuron Model:

*. input to the networks are represented by the mathematical symbol x_n.
*. Each of these inputs are ~~the~~ multiplied by weight w_n.

$$Sum = w_1x_1 + \dots + w_nx_n.$$

*. These products are simply summed, fed through the transfer function F( ), to generate a result and then output.



Learning:-

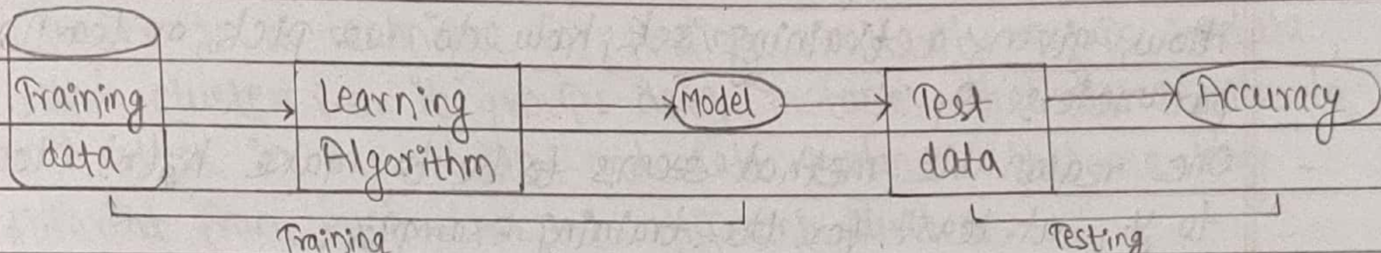
- In ANNs learning refers to the method of modifying the weights of connections between the nodes of a specified network.
- The learning ability of neural network is determined by its architecture and by the algorithmic method choosen for training.

1) Supervised Learning :-

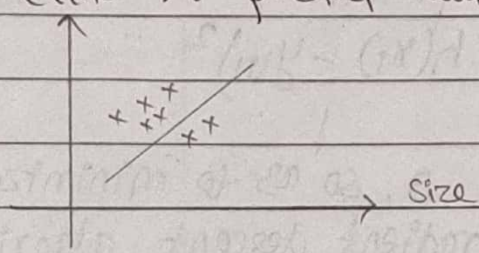
- Supervised learning is the machine learning task of inferring a function from labelled training data. The training data consists of a set of training examples.
- Supervised Learning process:-

  Two steps :

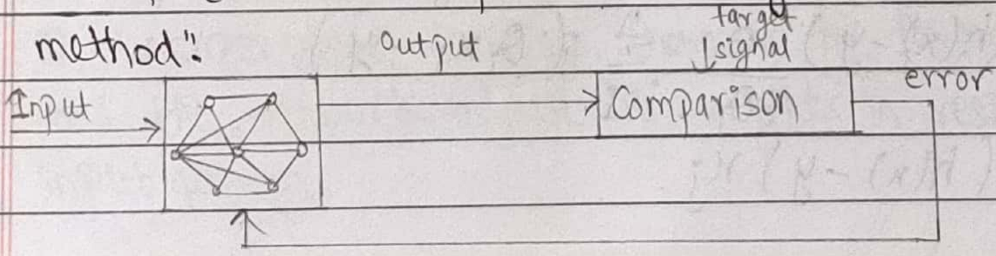
  1) Learning (training) :- Learn a model using the training data.
  2) Testing :- Test the model.



Eg: Regression (linear) : predict values from observation



- In supervised learning, the network is presented with inputs together with the target output.
- The neural network tries to produce an output as close as possible to the target signal by adjusting the values of internal weights.
- The most common supervised learning method is the "error-correction method":



Delta Rule :-

- The delta rule is a gradient descent learning rule for updating the weights of the artificial neurons in a single layer perceptions.
- To perform supervised learning, we must decide how we are going to represent functions / hypothesis 'h' in a computer.

$$h_\theta(x) = \sum_{i=1}^{n} \theta_i x_i$$

Now, given a training set, how do we pick, or learn the parameters θ_i.

- One reasonable method seems to be to make $h_\theta(x)$ close to y, at least for the training examples.

we have,

- We define the cost function,

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h(x_i) - y_{(i)})^2$$

- We want to choose θ, so as to minimize $J(\theta)$.
- Let's consider the gradient descent algorithm.

$$\boxed{\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)}$$

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_{(x)} - y)^2$$

$$= 2 \cdot \frac{1}{2} (h(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h(x) - y)$$

$$= (h(x) - y) \frac{\partial}{\partial \theta_j} \sum_{i=1}^{n} (\theta_i x_i - y)$$

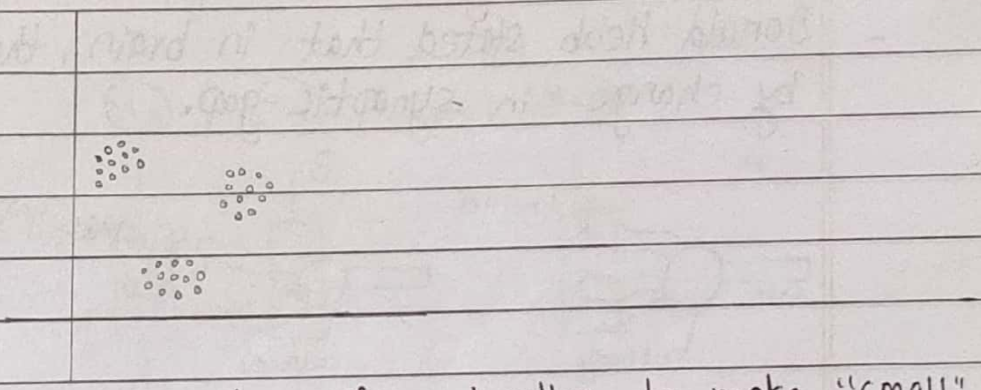
$$= (h(x) - y) x_j .$$

- For a single training example, this gives the update rule.

$$\boxed{\theta_j := \theta_j + \alpha\left(y - h_\theta(x)\, x_j\right)}$$

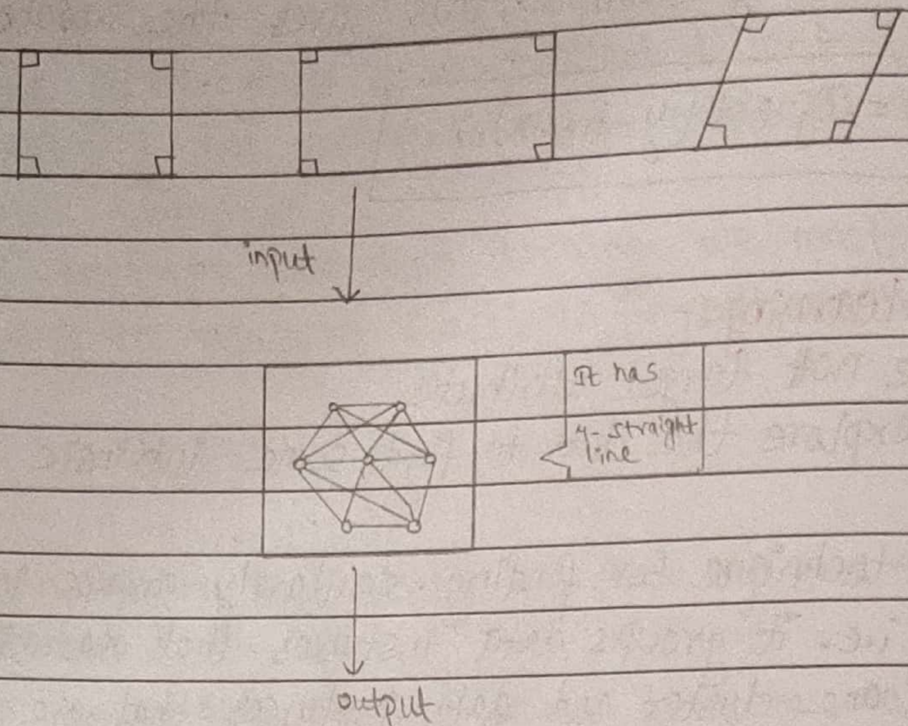
Unsupervised Learning :-

- The data have not target attributes
- We want to explore the data to find some intrinsic structure in them.
- Cluster is a technique for finding similarity groups in data, called clusters i.e. it groups data instances that are similar to each other in one cluster and data instances that are very different from each other into different clusters.
- The data set has three natural groups of data points, i.e. 3 natural groups.



Examples: Group people of similar size together to make "small", "medium" and "large". T-shirts.

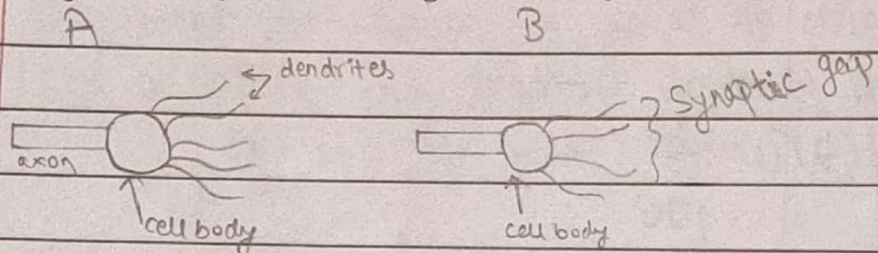
- Tailor-made for each person : to expensive.
- One-size-fits-all : doesnot fit all
- In unsupervised learning, there is no target signal from outside and the network adjust its weights in response to only the input pattern.



It is a rectangle

Hebbian Learning :-

- Donald Hebb stated that in brain, the learning is performed by change in synaptic gap.



- "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cell B firing B is increased."

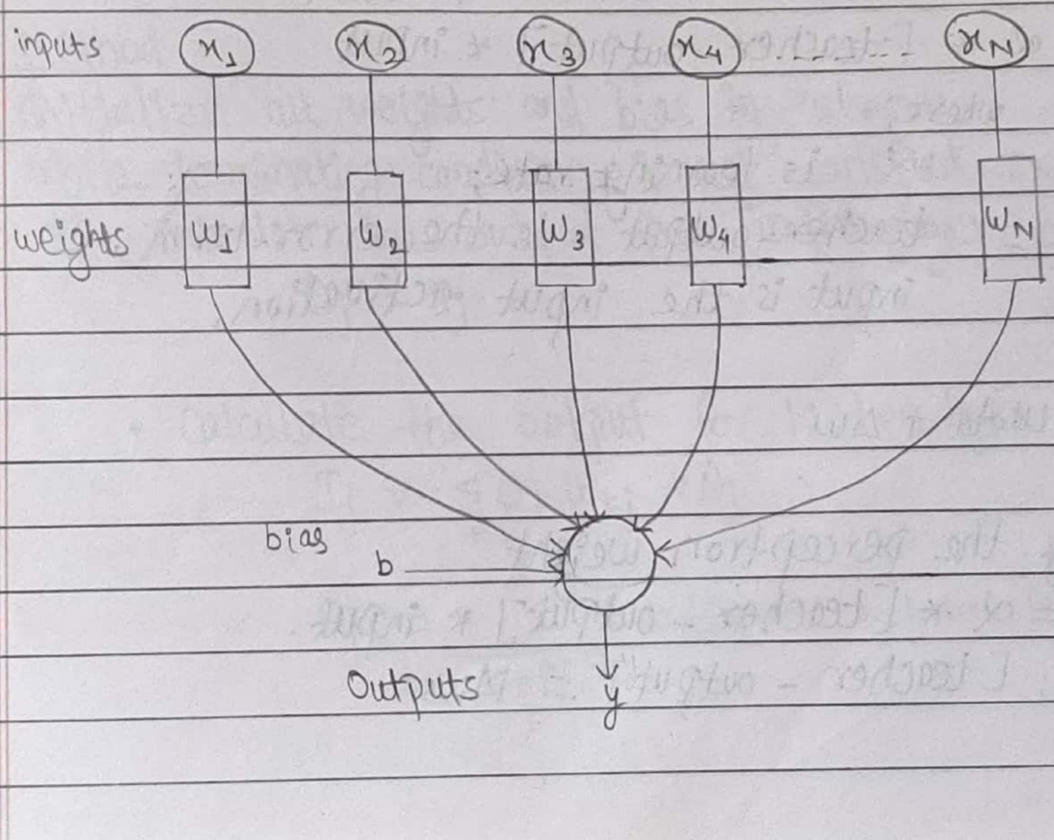
- According to the Hebb's rule, the weight vector is found to increase proportionally to the product of input and the learning signal.

$$W_{new} = W_{old} + x_i y$$

- Here, the learning signal is equal to the network output.

Perceptron Learning:-

- The term "perceptrons" was coined by Frank RosenBlatt in 1962 and is used to describe the connection of single neuron into network.
- A model for information storage and organization in the brain.
- Consists of single neuron with adjustable synaptic weights and bias.
- Weights are adapted using error-correction rule.



- Unit y can be calulated by a weighted sum of inputs $\{x_1, x_2, \ldots, x_N\}$.
- Linear Unit:

$$y = \sum_{i=1}^{N} w_i x_i + b$$

- Threshold Unit:

$$y = \begin{cases} 1 & \text{if } \sum_{i=1}^{N} w_i x_i + b > 0 \\ 0 & \text{if } \sum_{i=1}^{N} w_i x_i + b \leq 0. \end{cases}$$

Perceptron Learning-Rule:

- Teacher specifies the desired output for a given input.
- Network changes its weight is proportional to the error between the desired and calculated results.
- $\Delta w = \alpha * [\text{teacher} - \text{output}] * \text{input}$

where,
α is learning rate,
teacher_output is the error term, &
input is the input activation.

- $W_{new} = W_{old} + \Delta w$.

- Adjusting the perceptron weight

$\Delta w = \alpha * [\text{teacher} - \text{output}] * \text{input}$.

Miss is; [teacher - output] = Miss

	Miss < 0	Miss = 0	Miss > 0
input < 0	α	0	$-\alpha$
input = 0	0	0	0
input > 0	$-\alpha$	0	α

- Adjust each weight w based on input & miss.
- The above table shows adaptation.

Backpropagation Algorithm :-

Input:- D, training data set associated with their class level.

- l, learning rate.
- Multilayered feed forward neutral network

Output: A trained neural network

Method

1) Initialize all weights and bias in network.
2) While terminating condition is not satisfied,

I). Calculate the output for input layer,

$$O_j = I_j$$

- Calculate the output for hidden layer and output layer

$$I_j = \sum_k O_k W_{kj} + \theta_j$$

$$O_j = \frac{1}{1+e^{-I_j}}$$ (Sigmoid activation function)

Scanned with CamScanner

II) • Calculate the error for output layer

$$err_j = O_j(1-O_j)(T_j - O_j)$$

$\searrow$ Target

• Calculate the error for hidden layer

$$err_j = O_j(1-O_j)\sum err_k W_{jk}$$

III) Update the weight

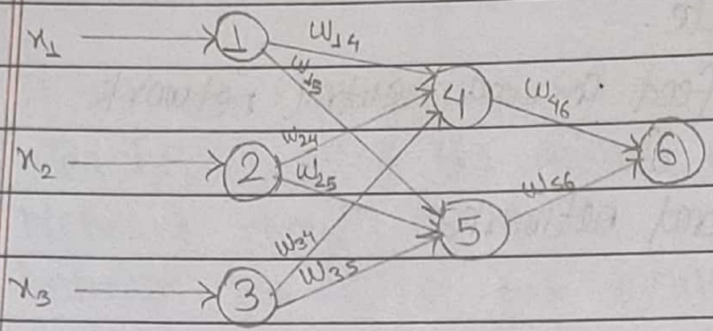
$$W_{ij}(new) = W_{ij}(old) + l(O_i)\, err_j$$

IV) Update bias.

$$\theta_j = \theta_j + (l)\, err_j$$

Example:- simple calculation for learning by backpropagation.

Let training tuple $X = (1,0,1)$ with class level 1.



Initial input, weight and bias values.

x_1	x_2	x_3	W_{14}	W_{15}	W_{24}	W_{25}	W_{34}	W_{35}	W_{46}	W_{56}	θ_4	θ_5	θ_6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0

Calculate output.

• For input layer

$O_1 = I_1 = 1, \quad O_2 = I_2 = 0, \quad O_3 = I_3 = 1$

• For hidden layer & output layer

$$I_4 = O_1 * W_{14} + O_2 * W_{24} + O_3 * W_{34} + \theta_4$$

$$= 1 * 0.2 + 0 * 0.4 + 1 * (-0.5) + (-0.4)$$

$$= -0.7$$

$$O_4 = \frac{1}{1+e^{-I_J}} = \frac{1}{1+e^{-(-0.7)}}$$
$$= 0.332$$

$$I_5 = O_1 * W_{15} + O_2 * W_{25} + O_3 * W_{35} + \theta_5$$
$$= 1*(-0.3) + 0*(0.1) + 1*(0.2) + 0.2$$
$$= 0.1$$

$$O_5 = \frac{1}{1+e^{-0.1}} = 0.525$$

$$I_6 = O_4 * W_{46} + O_5 * W_{56} + \theta_6$$
$$= 0.392 * (-0.3) + 0.525 * (-0.2) + 0.1$$
$$= 0.1$$

$$O_6 = \frac{1}{1+e^{-I_J}} = 0.474$$

Calculation of error

$$err_6 = O_6(1-O_6)(I_6 - O_6)$$
$$= 0.474(1-0.474)(1-0.474)$$
$$= 0.1311$$

$$err_5 = O_5(1-O_5)\, err_6 * W_{56}$$
$$= 0.525(1-0.525)(0.1311)(-0.2)$$
$$= -0.0065$$

$$err_4 = O_4(1-O_4)\,err_6 * W_{46}$$
$$= 0.332(1-0.332)(0.1311 * (-0.3))$$
$$= -0.0087$$

Bias

$$\theta_6 = \theta_6 + (l)\,err_6$$
$$= 0.1 + 0.9 * (0.1311)$$
$$= 0.218$$

UNIT-5 ⇒ STRUCTURED KNOWLEDGE REPRESENTATION

Structured knowledge Representation :

Knowledge:

*. facts, information, and skills acquired through experience or education; the theoretical or practical understanding of a subject.

*. The sum of what is known: the body of truth, information and principles acquired by human kind.

*. The range of one's information or understanding.

Source of knowledge:

Source → Documented: flow diagrams, Books

Source → Undocumented: Resides in people's mind.

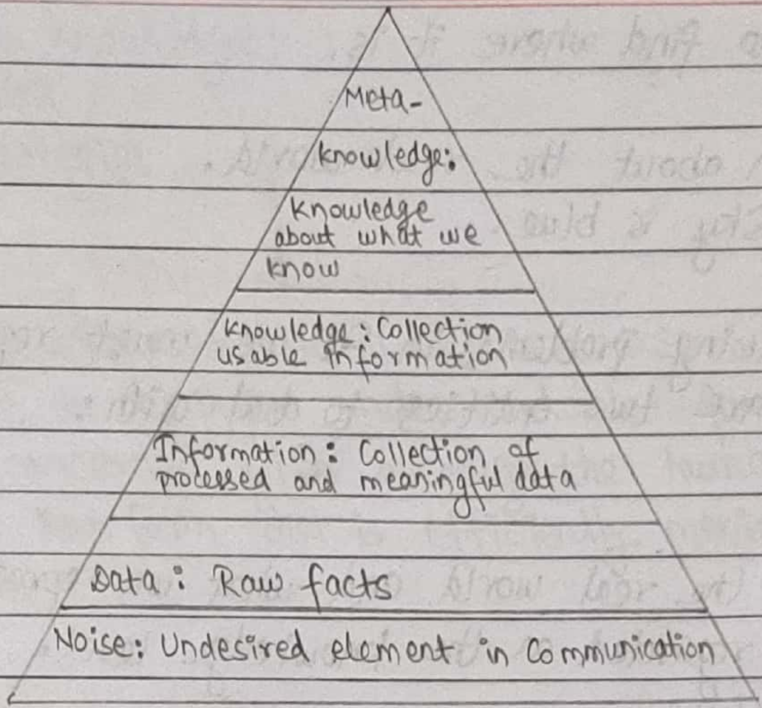


Fig: Knowledge pyramid

What to represent

- Let us first consider what kind of knowledge might need to be represented in AI system.

1) Objects: Facts about objects in our world domain.
   Eg: Guitars have strings.

2) Events: Actions that occur in our world.
   Eg: Ram played Guitar.

3) Performance: A behaviour like dirt cleaned by vacuum cleaner.

4) Meta-knowledge: knowledge about what we know.
   Eg: Renuka knows that she can read street signs along

the way to find where it is.

5) Facts : Truth about the real world.
Eg: Sky is blue.

Thus, in solving problems in AI we must represent knowledge and there are two entities to deal with :

- Facts :
  - Truth about the real world and what we represent.
  - This can be regarded as the knowledge level.
  Eg: Sky is blue.

- Representation of the facts :
  - which are manipulated.
  - This can be represented as symbol level.
  Eg : Sky (blue) V Sky (cloudy).

Types of knowledge :-

- Declarative knowledge :
  - Concepts
  - facts
  - Objects

- Procedural knowledge:
  - rules
  - strategies

Knowledge Representation:

- Knowledge Representation (KR) is a science that translate real knowledge to computer understandable form.
- KR is concerned with encoding the human knowledge into the ~~know~~ form that is efficiently manipulated by the computer.

Representation Mapping:

1) Forward Representation
   (fact to representation)
   eg: $\underbrace{\text{Ram is a man}}_{\text{fact}} \Rightarrow \underbrace{\text{Ram (Man)}}_{\text{Representation}}$,

2) Backward Representation
   Eg:
   $\underbrace{\text{Ram (Man)}}_{\text{representation}} \Rightarrow \underbrace{\text{Ram is a Man}}_{\text{fact.}}$,

Properties of knowledge representation system:

The following properties posed by a knowledge representation system:

- Representational Adequacy:
  - the ability to represent all kinds of knowledge that are needed in domain.

- Inferential Adequacy:
  - The ability to manipulate the knowledge represented to produce new knowledge corresponding to that inferred from the original.

- Inferential Efficiency:
  - The ability to direct the inferential mechanism into the most productive direction by storing appropriate guide.

- Acquisitional Efficiency:
  - The ability to acquire new information easily.

Approaches to Knowledge Representation (KR):-

-> Simple relational knowledge:

- It is a simplest way of sorting facts using relational method, when each & every fact about a set of object in set out sequentially.

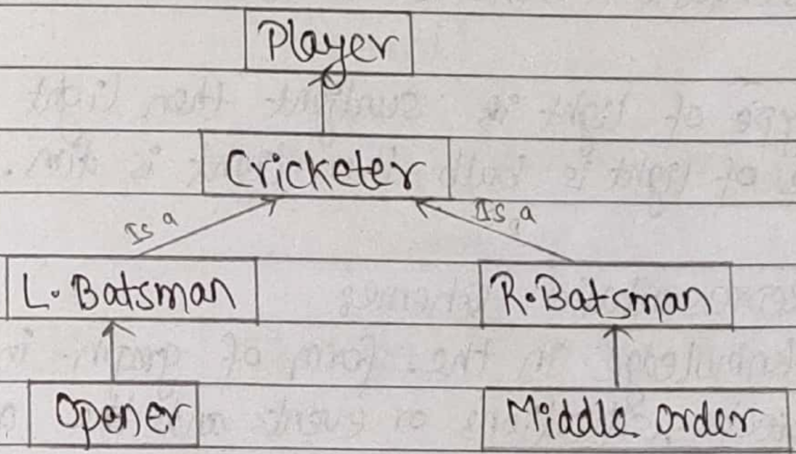
Eg:-

Player	weight	age
A	60	30
B	50	25

Playerinfo ('A', '60', '30')

2) Inheritable knowledge:
- All data should be stored into a hierarchy of classes.
- Classes must be arranged in a generalization.

Eg:-



3) Inferential knowledge:
- Represents knowledge as formal logic.

Eg: All dogs have tail.

$\forall x$: dog$(x) \rightarrow$ has a tail(x).

KR Schemes:-

1) Logical Representation Scheme:
- It uses expressions in the form of formal notation to represent the knowledge base.
- Propositional logic is the example of the logical representation scheme.

2) Procedural Representation Scheme:
- These representation scheme constitute rule based expert systems.

Eg: If type of light is sunlight then light is bright.
If type of light is bulb then light is dim.

3) Network Representation Scheme:
- Represents knowledge in the form of graph in which node represents objects, situations or events and the arc represents the relationship between them. Semantic Network, Conceptual dependencies, and conceptual graph are examples of network representation.

4) Structured Representation Scheme:
- This is the extension of network representation scheme where where nodes are complex data structure consisting of named slot with attached values, scripts, frames and objects are examples of structured representation scheme.

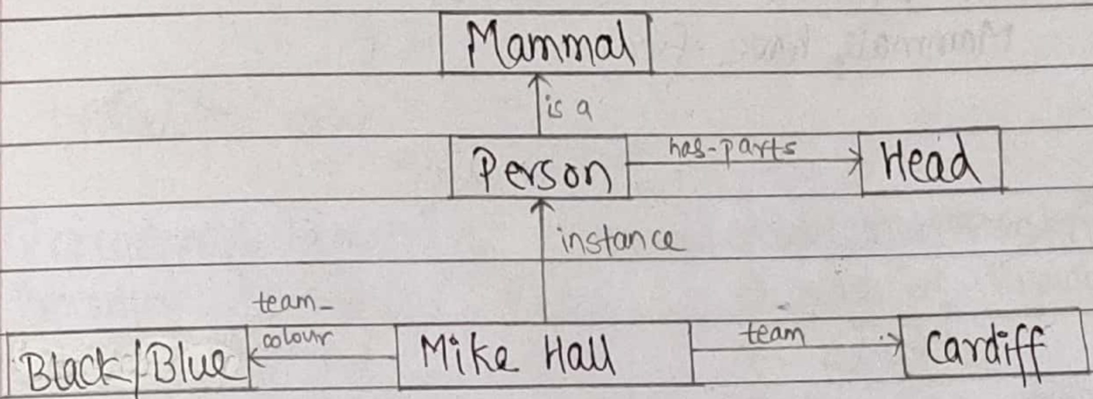
2074 How knowledge is represented using semantic networks?

Semantic Networks:-

- A semantic network is often used as a form of knowledge representation.
- It is a directed graph consisting of vertices which represent concepts and edges which represents semantic relation between the concepts.

Example:-

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

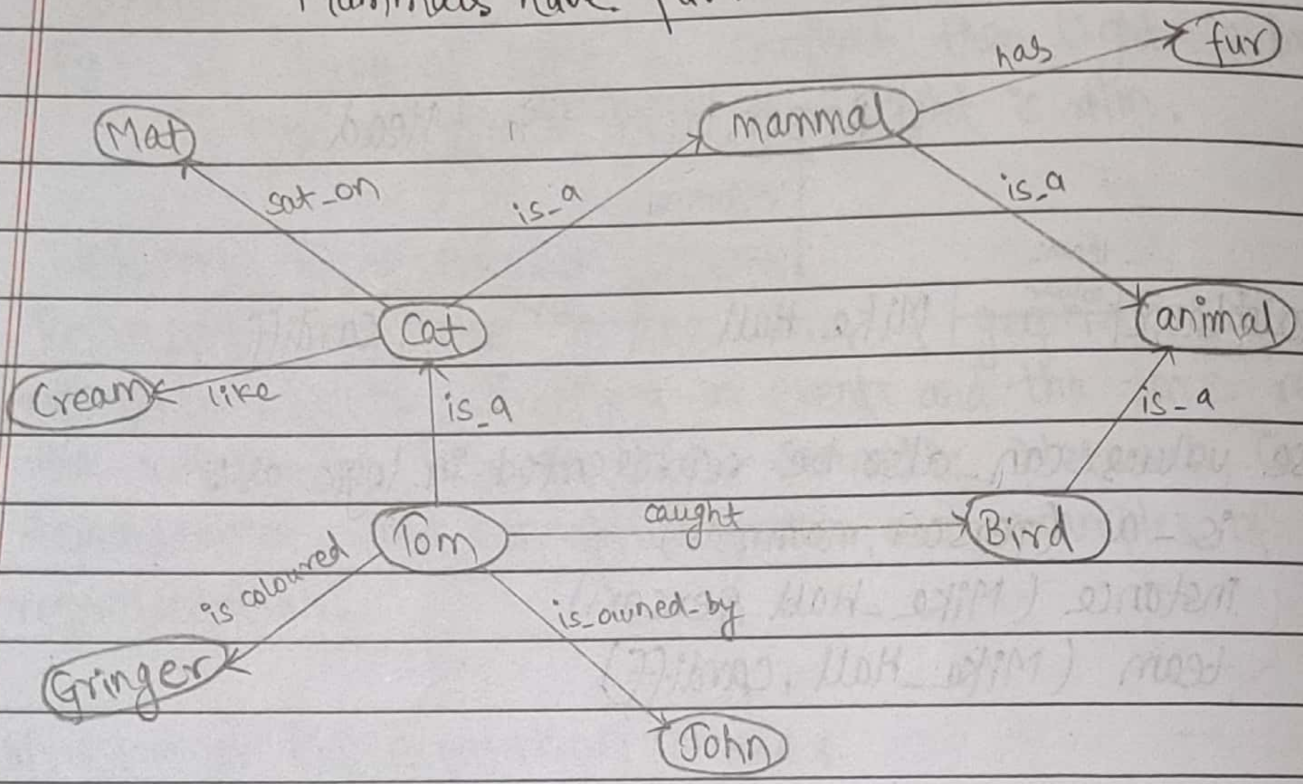


- These values can also be represented in logic as:
  - is_a (person, mammal)
  - instance (Mike_Hall, person)
  - team (Mike_Hall, cardiff)

- Thus, semantic is the study of meaning that is used to understand human expression through language.

Another example:

Tom is a cat.
Tom caught a bird
Tom is owned by John
Tom is gringer in colour.
Cats like cream
The cat sat on the mat.
A cat is a mammal.
A bird is an animal
All mammals are animals.
Mammals have fur.



Advantages:
- simple
- Understandable
- Translate to prolog without any difficulty.

Disadvantages:
- There is no standard definition of link name.
- Semantic nets are not intelligent, depend on creators.
- Links on objects represents only binary relations.

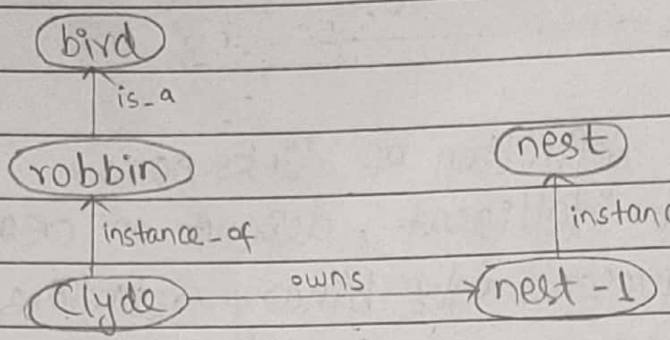
Types of knowledge:

Procedural knowledge	Declarative knowledge
- Procedural knowledge: rules, strategies, agendas	- Declarative knowledge: concepts, objects, facts.
- Is knowing how to do something.	- Is knowing about something.
- Can be directly applied to a task.	- Cannot be directly applied to a task.
- depends upon the task on which it can be applied.	- Is expressed in declarative sentences.
- also known as imperative knowledge.	- also known as descriptive knowledge.
- less general	- more general
- Eg: Type a text in a computer using keyboard.	- Eg: knowledge about the placement of keys in a keyboard.
- Process of planting herbs.	- knowing something about herbs.
- Draw line graph from a data set.	- Familiarity with the data sets & line graph.

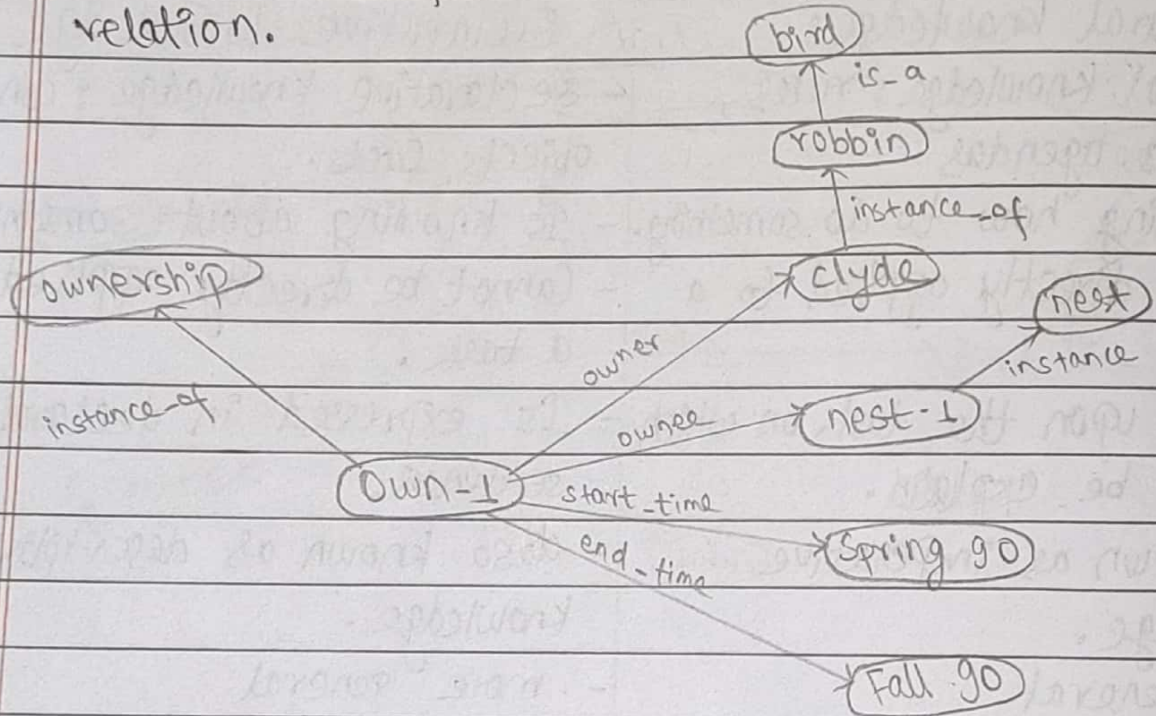
Scanned with CamScanner

Representing non-binary predicates:

- Suppose we wish to represent the fact that ~~cycle~~ clyde which is a name, owns a nest. This may be encoded as follows:



- How can one encode the addition information that clyde owned nest-1 from spring 90 to fall 90?
- We need to represent ownership as a node rather than a relation.

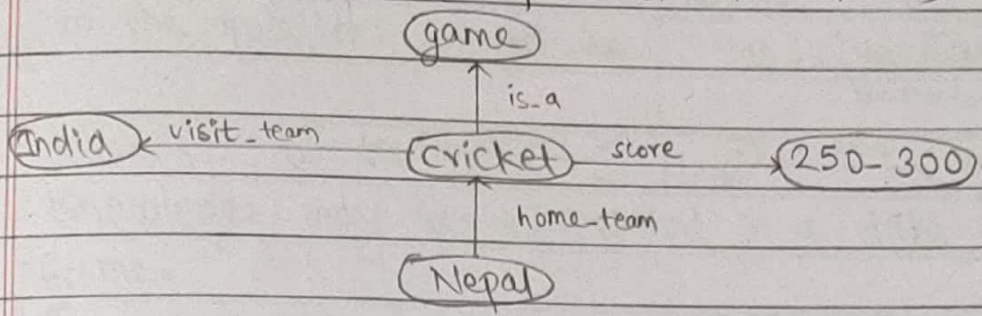


Ownership(owner, ownee, start_time, end_time)

- A semantic network representing "Ownership" as a node.

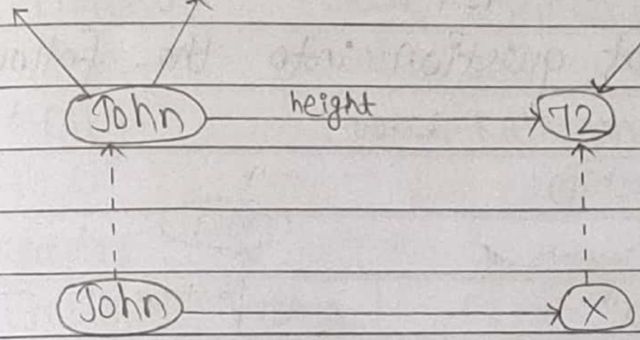
Example:

Avg_score (Nepal, India, 250, 300)



Network Matching:

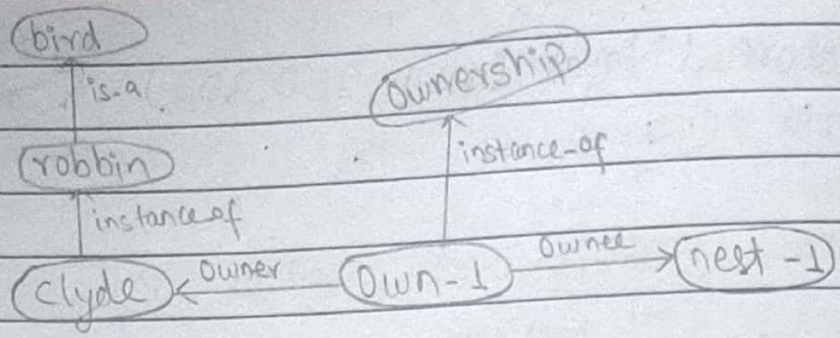
- A network fragment is constructed, representing a sought-for object or a query, and then matched against the network database to see if such an object exists.

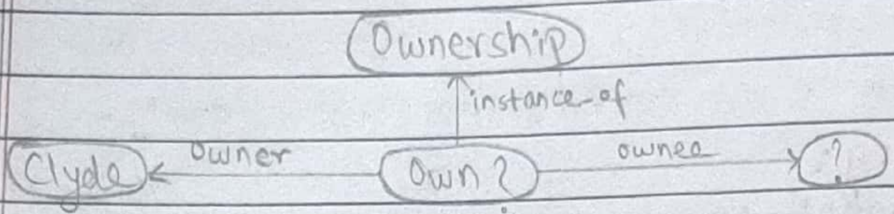


Q. What is the height of John!

Ans: The height of John is 72.

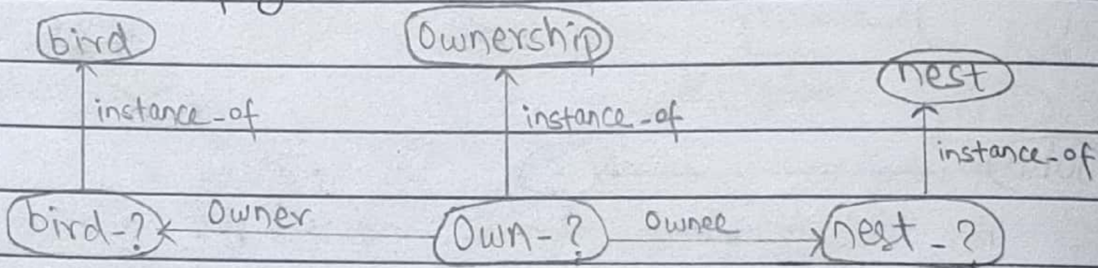
- As another example, let us suppose that we use the network given below as a database & suppose we wish ~~for~~ to answer the question "What does clyde own?"





For example; Suppose we wish to answer the question "Is there a bird who owns a nest ?"

We could translate that question into the following network fragment.



- Here, bird-?, nest-?, and own-? nodes represents as: bird-owning-nest relation.

- Query network fragment does not match the knowledge base exactly. The deductive procedure would have to construct an "instance-of" link from clyde to bird to make match possible.

- The matcher would bind bird -? to the node clyde, own -? to own - 1 and nest -? to nest -? and owner to the question would be "Yes, clyde".

Frames:-

- Knowledge may be represented in a data structure called frame.
- Frames are record like structure that consists of collection of slots and or attributes and the corresponding slot value.
- Slots have name and value called as facts.

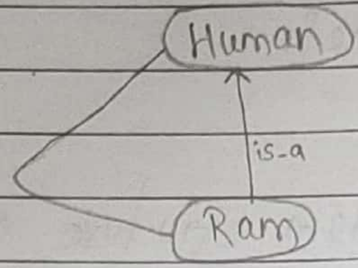
XYZ
Professor (value Engineer)
Age (value 25)
City (value Kathmandu)

Example:

Employee Frame	
Name	
Address	
SSN	
Job Type	

Job Type Frame	
Title	
Payrate	
Hours	
Duties	

Weak slot & filler structure / Semantic network



Conceptual Dependency (CD) :-

- A model for Natural language understanding in AI.
- Represents knowledge acquire from Natural Language input.
- CD theory was developed by Shank in 1973 to 1975 to represent the meaning of NL sentences.
    - *. It helps in drawing inferences
    - *. It is independent of the language.
- CD representation of sentences is not built using conceptual primitives which give the intended meaning of the words.

Primitive Acts as CD theory:

- ATRANS (transfer of an abstract relationship) (give)
- PTRANS (transfer the physical location of object) (go)
- PROPLE (application of physical force to an object) (push)
- MOVE (movement of the body parts) (kick)
- GRASP (grasping of an object by an actor) (throw)
- INGEST (eat, drinks)
- EXPEL (cry)
- MTRANS (transfer of mental information (tell)
- MBUILD (decide)

- SPEAK (say)
- ATTEND (listen)

Six Premitives Conceptual Categories:

PP → real world object
ACT → real world actions
PA → attributes of object
AA → attributes of actions

Eg: Man took a book.

CD Rep. : Man $\Leftrightarrow$ took ← book —
→ Man
← Someone

I gave a book to man.

CD Rep. : I $\overset{\text{Past tense}}{\Leftrightarrow}$ gave $\overset{\text{Object}}{\leftarrow}$ book —
$\overset{\text{Recipients}}{\rightarrow}$ Man
← I

Rules:

1) PP $\Leftrightarrow$ ACT

eg: John ran

CD Rep : John $\overset{P}{\Leftrightarrow}$ PTRANS

2) ACT $\Leftrightarrow$ PP

eg: John pushed the bike.

CD Rep. : John $\Leftrightarrow$ PROPLE $\overset{o}{\leftarrow}$ bike

3) $PP \longleftrightarrow PP$

eg: John is Doctor

CD Rep. : John $\leftrightarrow$ doctor

4) $PP \Leftrightarrow PA$

eg: John is fat.

CD Rep. : John $\Leftrightarrow$ weight (>80)

5)

$$ACT \leftarrow R \begin{cases} \rightarrow PP\ (\text{to}) \\ \leftarrow PP\ (\text{from}) \end{cases}$$

eg: John took the book from Mary.

CD Rep. : John $\Leftrightarrow$ ATRANS $\leftarrow R \begin{cases} \rightarrow \text{John} \\ \leftarrow \text{Mary} \end{cases}$

6)

$\Leftrightarrow \{x\}$

$\uparrow$

$\Leftrightarrow \{y\}$ i.e. if x then y.

eg: Bill shot Bob.

$\{x\}$: Bill shot Bob

$\uparrow$

$\{y\}$: Bob's health is poor.

Scripts :-

- A script is a structure that describes a set of circumstances which could be expected to follow on from one another.
- Scripts are used for representing knowledge about common sequences of events.
- It is similar to sequence or chain of situation.
- It can have number of slots and frames with more specialized rules.
- Scripts are beneficial because :
  * Event tend to occur in known runs or patterns.
  *. Casual relationship between events ~~occurs~~ exists.

Scripts components :

- Entry condition
  * These must be satisfied before events in scripts can occur.

- Results
  * Conditions that will be true after events in scripts occur.

- Props
  *. Slot representing objects involved in events.

- Roles
  * Persons involved in the events.

- Track:
  *. Variations on scripts. Different track may share components of the same scripts.

- Scenes:
  *. The sequence of events that occur. Events are represented in CD form.

The classic example in the restaurant scripts

- Scene: A restaurant with an entrance and table.
- Actors: The dinners, servers, etc.
- The table setting, ordering a meal, serving a meal, eating the meal, requesting the check, paying, leaving, etc.

Why represent knowledge in this way? fixed ideas
- Because real world events do follow stereotyped patterns. Human being use previous experiences to understand verbal accounts. Computer can use scripts instead.

- Because people when relating events, do leave large amount of assumed detail out of their accounts.

Advantages:
- Ability to predict events.
- A single coherent interpretation may be built up from a collection of observation.

Disadvantages:

- Less general than frames
- May not be suitable to represent all kinds of knowledge.

Introduction to Uncertainty:

- The information available to human is often imperfect. An expert can cope with defects.
- Classical logic permits only exact reasoning.
- If A is true
  THEN
  A is ¬False
  &
- If B is false
  THEN
  B is ¬true.

- Most real world problems do not provide exact information. It can be inexact, incomplete or even immeasurable.

Definition:

- Uncertainty is defined as the lack of exact knowledge that would enable us to reach a perfectly reliable conclusion.

Let Action A_t = leave for airport t minutes before flight.
Will A_t get me there on time?

Problems:-

1. Partial observability: (road state, other driver plan's etc).
2. Noisy sensors (radio traffic reports)
3. Uncertainty in action outcome (flat tyres).
4. Complexity of modeling & predicting traffic.

Hence, a purely logical approach either.
1. Risk falsehood: "A_{25} will get me there on time" or
2. Leads to conclusion, that are too weak for decision making."
"A_{25} will get one there on time if there is no accident on the bridge and it doesn't rain, etc.

A_{1440} might reasonably be said to get me there on time but I have to stay overnight in the airport.

Baye's Rule:-
Theorem:

$$P(b/a) = \frac{P(a/b) * P(b)}{P(a)}$$

Proof:
We know that:

$$P(a/b) = P(a \wedge b) / P(b)$$
$$P(a \wedge b) = P(a/b) \cdot P(b) \quad \text{—— (I)}$$

Similarly,

$$P(b/a) = P(a \wedge b) / P(a)$$
$$P'(a \wedge b) = P(b/a) \, P(a) \quad \text{—— (II)}$$

Equating (I) and (II).

$$P(a/b) \cdot P(b) = P(b/a) \cdot P(a)$$

i.e. $$P(b/a) = \frac{P(a/b) \cdot P(b)}{P(a)}$$

A
2R
3B

B
3R
4B

A ko thaxa, R ko thaxaina.

$\Rightarrow$ $P(R/A) \Rightarrow$ bag A is choosen

$\Rightarrow$ Red ball is drawn from bag A

$$P(A \cap R) = P(A) \cdot P(R/A) \quad P(A/R)$$

$\Rightarrow$ $P(R) = P(A \cap R) + P(B \cap R)$

(Reverse probability)

$\Rightarrow$ Given that red ball is drawn. What is the probability that the ball is from bag A?

$$P(A/R) = \frac{P(A \cap R)}{P(A \cap R) + P(B \cap R)}$$

$$= \frac{P(A) \cdot P(R/A)}{P(R)}$$

Example:
A doctor knows the disease Meningitis causes the patient to have a stiff neck 50% of the time. The doctor also know that the probability that a patient has Meningitis is 1/50,000, and the probability that any patient has a stiff neck is 1/20. Find the probability that a patient with a stiff neck has meningitis.

- Here,
We are given,
$P(S|m) = 0.5$
$P(m) = 1/50,000$
$P(S) = 1/20$

Now,
Using Baye's Rule,

$$P(m|S) = \frac{P(S|m)\ P(m)}{P(S)}$$

$$= \frac{(0.5) * (1/50,000)}{(1/20)}$$

$$= 0.0002.$$

Bayesian Network:-
- A data structure to represent the dependencies among variables and to give a specification of any full joint probability distribution.
Also called belief network or probabilistic network or causal network or knowledge map.

The basic idea is:

- Knowledge in the world is modular - most events are conditionally independent of most other events.
- Adopt a model that can use a more local representation to allow interactions between events that only effects each other.
- Some events may be unidirectional other may be bidirectional. Make a distinction between these in model.
- Events may be causal and then get chained together in a network.

A Bayesian network is a Directed Acyclic Graph (DAG) which consists of:

- A set of random variables which makes up the nodes of the network.
- A set of directed links connecting pairs of nodes. If there is an arrow from node X to node Y, X is said to be parent of Y.
- Each node X_i has a conditional probability distribution.

$$P(Y_i / Parent(X_i))$$

Example:

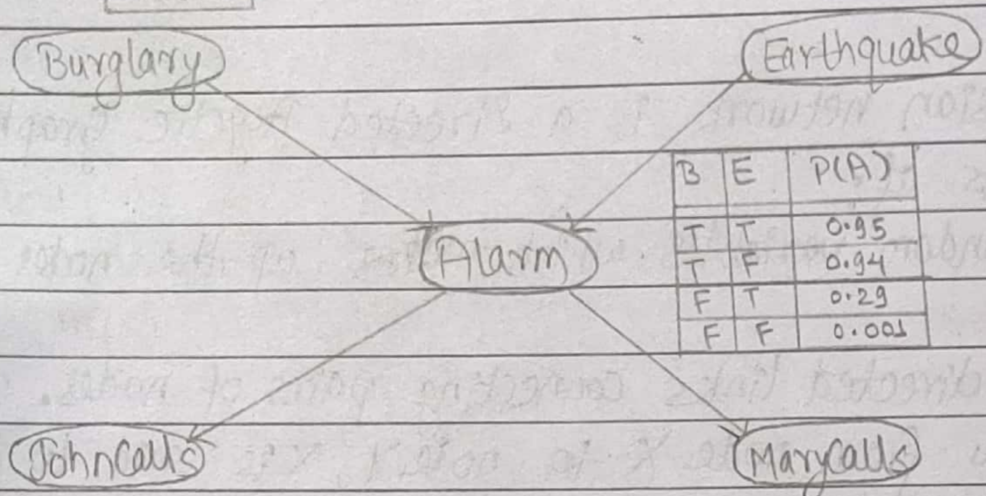
You have a burgler alarm installed in your home. It is fairly reliable at detecting a burglary, but also responds on occasion to minor earthquake. You also have two neighbours John & Mary, who have promised to call you at work when they hear the alarm. John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm & calls then too. Mary, on the other hand, likes rather loud music and sometimes misses the alarm altogether.

We would like have to estimate the probability of a burglary with given evidence who has or has not call.

Variables : Burglary, Earthquake, Alarm, JohnCall, MaryCalls

P(B)
0.001

P(E)
0.002



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

A	P(J)
T	0.90
F	0.05

A	P(M)
T	0.70
F	0.01

Scanned with CamScanner

UNIT - 6

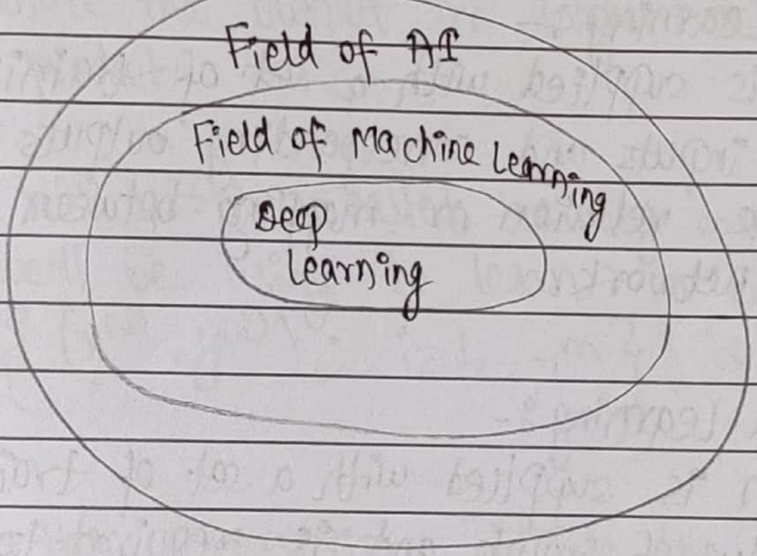
MACHINE LEARNING

Machine Learning :-

- "Learning is any process by which a system improves performance from experience." - Herbert Alexander Simon.

- Learning is making useful changes in our mind.
- Machine Learning is an application of AI that provides a systems the ability to automatically learn and improve from experience without being explicitly programmed.

- Machine Learning refers to the changes in systems that perform tasks associated with artificial intelligence (AI). Such task involves recognition, diagnosis, planning, robot control, prediction, etc. The changes might be either enhancements to already performing systems or synthesis of new system.



Types of learning :-

1) Rote Learning
2) Supervised learning
3) Unsupervised Learning

1) Rote Learning :-

- Rote learning can be defined as a memorization technique based on repeating the material again and again till you get through with it and begin to memorize.
- Rote learning is generally based on only repeating of the topics and not clear understanding of the topic.
- Rote learning methods are generally used when quick memorization is required, such as one's memorizing a telephone number, memorization of past events.

2) Supervised Learning :-

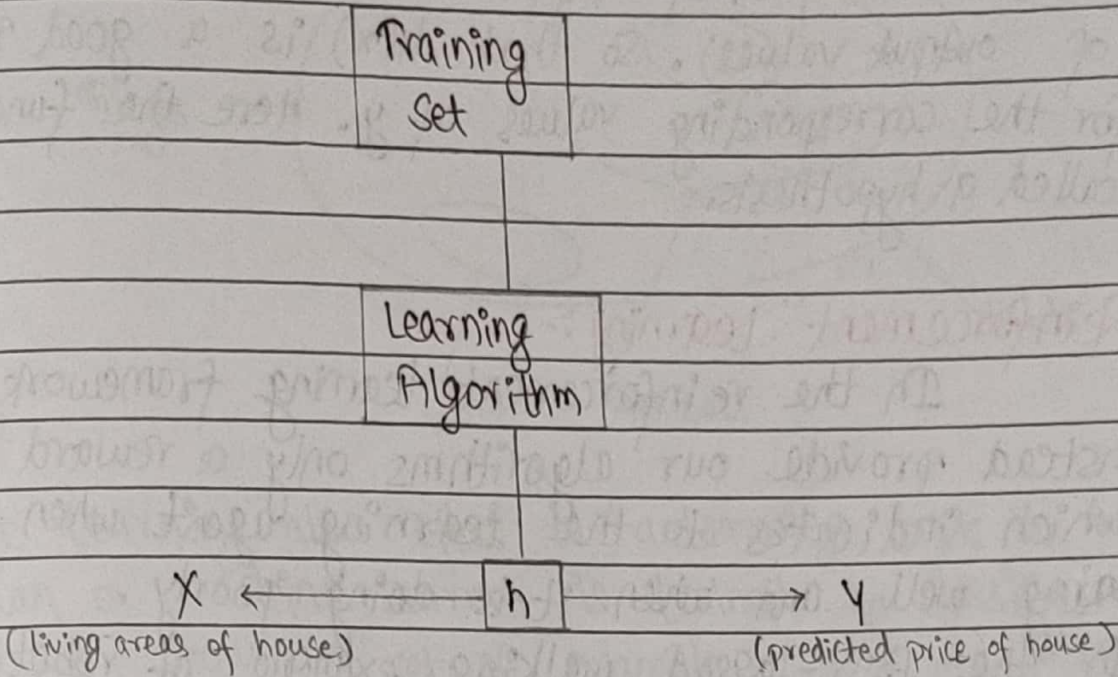
- The system is supplied with a set of training examples consisting of inputs and corresponding outputs, and is required to discover the relation or mapping between them.

  Eg :- neural network.

3) Unsupervised Learning :-

The system is supplied with a set of training examples consisting only of inputs and is required to discover for itself what appropriate outputs should be.

Eg :- Self organizing map.



- To establish a notation for future use, we'll use $x^{(i)}$ to denote the input variables, also called input features and $y^{(i)}$ to denote the output or target variables that we are trying to predict.

- A pair $(x^{(i)}, y^{(i)})$ is called training examples and the data set that we'll be using to learn - a list of m training examples $\{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$ is called a training set.

- Note that the superscript "i" in the notation is simply an index into the training set, and has nothing to do so with exponentiation.

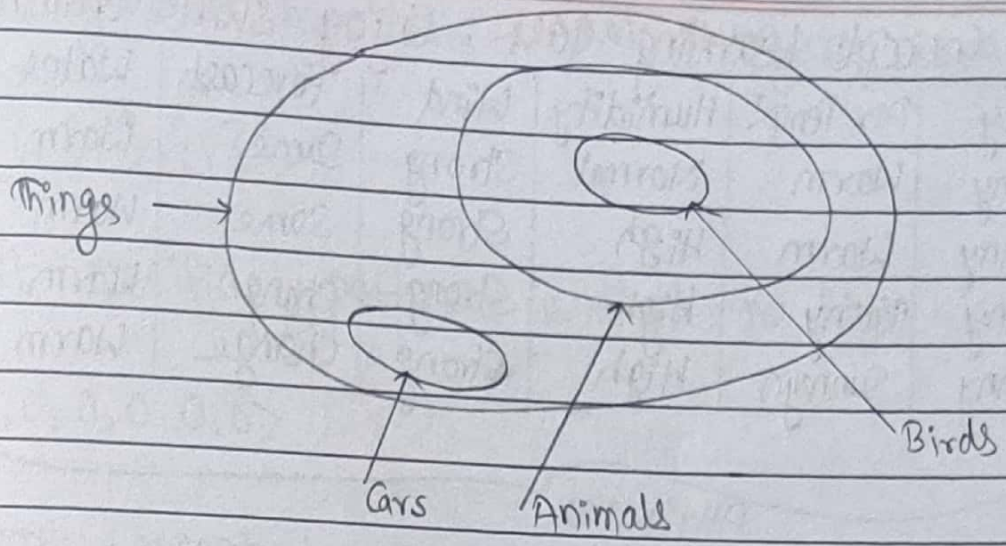
- To describe the supervised learning problem, our goal is given a training set, to learn a function $h: X \rightarrow Y$ (where X denote the space of input values, and the Y is the space of output values). So that h(x) is a good predictor for the corresponding values of y. Here the function h is called a hypothesis.

Reinforcement Learning:-

- In the reinforcement learning framework, we'll instead provide our algorithms only a reward function, which indicates to the learning agent when it is doing well, and when it is doing poorly.
- In the four-legged walking example of robot, the reward function might give the robot positive reward for moving forward and negative reward for moving backward or falling over.
- Reinforcement learning has been successfully used in many applications such as, robot legged locomotion, cell-phone network routing, marketing strategy selection, etc.

Learning through Examples (A Type of Concept Learning):

- A concept is a subset of objects or events defined over a larger set.
- For example, we refer to the set of everything (i.e. all objects) as the set of things.
- Animals are subset of things and birds are subsets of animals.



- Concept learning also refers to a learning task in which a human or machine learner is trained to classify objects by being shown a set of example objects along with their class levels.
- It is the form of learning which requires higher order mental processes like thinking, reasoning, intelligence, etc.
- This learning is very useful in recognizing, identifying things.

Level	Concept C		Data Object X
1, +1, T	X-belong	or	to a concept C
0, -1, F	not belong		to a concept C.

A Concept Learning Task: Enjoy Sport Training Examples

Example	Sky	Air Temp.	Humidity	Wind	Forecast	Water	Enjoy sport
					Same	Warm	Yes
1	Sunny	Warm	Normal	Strong	Same	Warm	Yes
2	Sunny	Warm	High	Strong	Change	Warm	No
3	Rainy	Cold	High	Strong	Change	Warm	Yes
4	Sunny	Warm	High	Strong			

Attributes

Concept

- A set of example days, and each is described by six attributes.
- The task is to learn to predict the value of Enjoy sport for arbitrary days, based on the values of its attribute values.

Enjoy Sport : Hypothesis Representation

- Each hypothesis will be a vector of six constraints specifying the values of the six attributes.
- (Sky, AirTemp, Humidity, Wind, Water, Forecast).
- Each attributes will be

? - indicating any value is acceptable for the attribute

Single value - specifying a single required value (eg: warm)

0 - indicating no value is acceptable for the attribute (no value).

- A hypothesis:

Sky	AirTemp	Humidity	Wind	Water	Forecast
<Sunny,	?,	?,	Strong,	?,	Same>

- The most general hypothesis - that everyday is a +ve example.
  < ?, ?, ?, ?, ?, ? >

- The most specific hypothesis is that no day - is a +ve example.
  <0, 0, 0, 0, 0, 0>

- Enjoy Sport Concept learning task requires learning the sets of days for which Enjoy sport = Yes.

Explanation-Based Learning (EBL) :-

- Explanation Based Learning (EBL) is a form of machine learning that exploits a very strong, or even perfect, domain theory to make generalizations or form concepts from training example.
- Learning general problem solving techniques by observing and analyzing solutions to specific problems.
- Explanation Based Learning system accepts an example (i.e. a training example) and explain what it learn from the example.

EBL accepts fair inputs :-

- A training example:
  What the learning sees in the world.

- A goal concept:
  a high level description of what the program is supposed to learn (the set of all possible conclusion).

- A operational criterion:
  a description of which concepts are usable. (Criteria for determining which features in domain are efficiently recognizable, e.g: which features are directly detectable using sensors)

- A domain theory:
  a set of rules that describes relationships between objects and actions in a domain.

From this EBL compute a generalization of the training example that is sufficient not only to describe the goal concept but also satisfies the operational criterion.

This has two steps:

- Explanation:
  the domain theory is used to prune away all unimportant aspects of the training example with respect to the goal concept.

- Generalization:
  the explanation is generalized as for possible

while still describing the goal concept.

Learning By Analogy:-

- Analogy is a cognitive process of transferring information or meaning from a particular subject - the analog or source to another - the target or a linguistic expression corresponding to such a process.
- It is a powerful inference tool.
- Analogical learning generally involves developing a set of mapping between feature of two instances.

Eg: The thief was caught red handed.

Two ways of solving this problem:

- Transformational Analogy:-

  * Look for the similar situation and copy it to the new situation making suitable substitution where appropriated.

- Derivational Analogy:-

  * Transformational analogy does not look at how problems was solved, it only look at the final solution.
  * In this, it replays in previous derivation and modify it if necessary.

Learning By Simulation:

- It is an experimental approach of learning.
- The prototype is developed to learn new knowledge from the environment.
- Calibration & Validation.

Route finding problem:

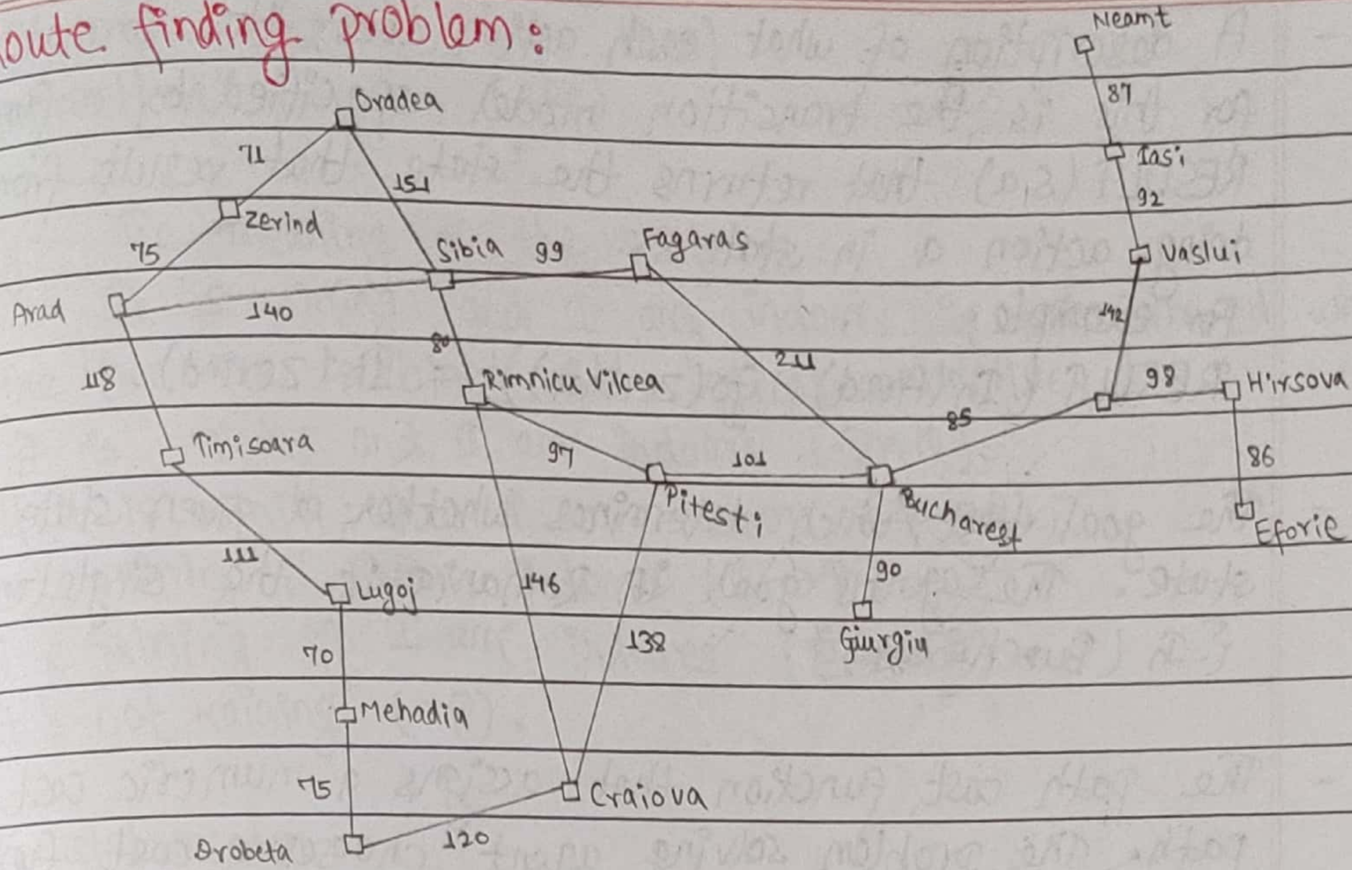


Fig :- ① A simplified road map of part of Romania

A problem can be defined formally by five components:

- The <u>initial state</u> that the agent starts in. For example; the initial state for our agent in Romania might be described as <u>In (Arad)</u>.

- A description of the possible actions available to the agent. Given a particular state S, ACTION (s) returns the set of actions that can be executed in S. We say that each of these actions is applicable in S. For example; from the state In(Arad), the applicable actions are { Go(Sibia), Go(Timisoara), Go (Zerind) }.

- A description of what each action does; the formal name for this is the transition model, specified by a function RESULT(s,a) that returns the state that results from doing action a in state s.
For example;
RESULT(In(Arad), Go(zerind)) = In(zerind).

- The goal test, which determines whether a given state is goal state. The agent's goal in Romania is the singleton set {In(Bucharest)}.

- The path cost function that assigns a numeric cost to each path. The problem solving agent chooses a cost function that reflects its own performance measure.

UNIT-4

Formal Logic-connectives:

- In logic, a logical connective (also called a logical operator) is a symbol or word used to connect two or more sentences (of either a formal or a natural language) in a grammatically valid way, such that the compound sentences produced has a truth value dependent on the respective truth value of the original sentences.

- Commonly used logical connectives includes:
1) Negation ($\neg$) (not)
2) Conjunction ($\wedge$) (and)
3) Disjunction ($\vee$) (or)

4) Implication ($\rightarrow$) (if.... then)
5) Bidirectional ($\leftrightarrow$) (iff)

Eg:- The meaning of the sentences.
It is raining and I am indoors is transformed when the two are combined with logical connectives.

- It is raining and I am indoors ($P \wedge Q$)
- If it is raining, then I am indoors. ($P \rightarrow Q$)
- It is raining if I am indoors ($Q \rightarrow P$)
- It is raining iff I am indoors ($P \leftrightarrow Q$)
- It is not raining ($\neg P$).

For statement:
P = It is raining
& Q = I am indoors.

Truth Table:

OR		
P	Q	PVQ
F	F	F
F	T	T
T	F	T
T	T	T

- The table shows the relationship between the values of P, Q & PVQ.

Logic:-

- Logic is the formal language for representing knowledge such that conclusion can be drawn.
- It has syntax, semantics and proof theory.
- Syntax defines the sentences in the language.
- Semantics defines meaning of the sentences. Semantics property includes truth and falsity.
- Proof theory: (Inference method)
  Set of rules for generating new sentences that are necessarily true given that the old sentences are true.

- We'll consider two kinds of logic:
  1) Propositinal logic
  2) First-order logic

1) Propositional logic:-

- A proposition (or statement) is a declarative sentence (i.e. a sentence that declares a fact) that is either true or false, but not both.
- Propositional logic is the simplest logic. We use the symbol like P_1, P_2 to represent sentences. Propositional logic is defined as:
  - If S is a sentence, $\neg S$ is a sentence (negation)
  - If S1 and S2 are sentences, $S1 \wedge S2$ is a sentence (conjun
  - If S1 and S2 are sentences, $S1 \vee S2$ is a sentence (Disjunctio
  - If S1 and S2 are sentences, $S1 \Rightarrow S2$ is a sentence (implication)
  - If S1 and S2 are sentences, $S1 \Leftrightarrow S2$ is a sentence (Bidirection

- Formal grammer for propositional logic can be given as below:

Sentence → AutomicSentence | Complex Sentence
AutomicSentence → True | False | Symbol
Symbol → P | Q | R ...
ComplexSentence → ¬Sentence
| (Sentence ∧ Sentence)
| (Sentence ∨ Sentence)
| (Sentence ⇒ Sentence)
| (Sentence ⇔ Sentence)

Note:
Automic Sentence:
"The Sky is blue."

Complex Sentence:
"The sky is blue & the plants are green."

Propositional logic : Semantics

- Each model specifies true/false for each proposition symbol.
- Rules for evaluating truth with respect to a model.

$\neg S$ is true, if S is false.
$S_1 \wedge S_2$ is true if, S_1 is true and S_2 is true.
$S_1 \vee S_2$ is true if, S_1 is true or S_2 is true.
$S_1 \Rightarrow S_2$ is true if, S_1 is false or S_2 is true.
$S_1 \Leftrightarrow S_2$ is true if, $S_1 \Rightarrow S_2$ is true and $S_2 \Rightarrow S_1$ is true.

Example:

- "I will get wet if it rains and I go out of the house."

Let propositions be:

p: "I will get wet."

q: "It rains."

r: "I go out of the house."

$(q \wedge r) \rightarrow p$.

- Representing simple facts

* It is raining

RAINING

* It is sunny

SUNNY

* It is windy

WINDY

If it is ~~not~~ raining, then it is not sunny.

$RAINING \rightarrow \neg SUNNY$

Inference Rule:

Rule	Premise	Conclusion
Modus Ponens	$A, A \rightarrow B$	B
And-Elimination	$A \wedge B$	A
Resolution	$A \vee B, \neg B \vee C$	$A \vee C$
Modus Tollens	$A \Rightarrow B, \neg B$	$\neg A$

Monotonicity:-

The set of entailed sentences can only increase as information is added to the knowledge base.
For any sentence α & β if $KB \models \alpha$ then

$$KB \wedge \beta \models \alpha$$

CNF (Conjunctive Normal Form):-

- Conjunction of disjunctions of literals (clauses).

eg: $(P \vee Q \vee R) \wedge (P \vee Q) \wedge (P \vee R)$

It is conjunction ($\wedge$) of disjunction ($\vee$).

where, disjunction are:

$$\left.\begin{array}{l} P \vee Q \vee R \\ P \vee Q \\ P \vee R \end{array}\right\} \text{clauses}$$

Conversion to CNF :-

- A sentence that is expressed as a conjunction of disjunction of literals is said to be in conjunctive Normal Form (CNF).

Algorithm:

Eliminate $\leftrightarrow$ rewriting $P \leftrightarrow Q$ as $(P \rightarrow Q) \wedge (Q \rightarrow P)$

Eliminate $\rightarrow$ rewriting $P \rightarrow Q$ as $\neg P \vee Q$

Use De-morgans law to push $\neg$ inwards.

- rewrite $\neg(P \wedge Q)$ as $\neg P \vee \neg Q$
- rewrite $\neg(P \vee Q)$ as $\neg P \wedge \neg Q$

Eliminate double negation : rewrite $\neg\neg P$ as P.

Use the distributive laws to get CNF :

- rewrite $(P \wedge Q) \vee R$ as $(P \vee R) \wedge (Q \vee R)$

Flatten nested clauses :

- $(P \wedge Q) \wedge R$ as $P \wedge Q \wedge R$
- $(P \vee Q) \vee R$. as $P \vee Q \vee R$

Example:

$$B \Leftrightarrow (A \vee C)$$

sol:

- Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$

*. $(B \Rightarrow (A \vee C)) \wedge ((A \vee C) \Rightarrow B)$

- Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

*. $(\neg B \vee A \vee C) \wedge (\neg(A \vee C) \vee B)$

- Move ¬ inwards using de-Morgans rule and double negation.
  - $(\neg B \vee A \vee C) \wedge ((\neg A \wedge \neg C) \vee B)$

- Apply distributive law ($\wedge$ or $\vee$) and flatten:
  - $(\neg B \vee A \vee C) \wedge (\neg A \vee B) \wedge (\neg C \vee B)$

Resolution Algorithm:-

Convert KB into CNF

Add negation of sentence to be entailed into KB (i.e. $KB \wedge \neg\alpha$)

Then apply resolution rule to resulting clauses:

The process continues until

- There are no new clauses that can be added
  - Hence KB does not entail α.
- Two clauses resolve to entail the empty clause.
  - Hence KB does entail α.

Resolution Example:

- To prove: $\neg P$
- Transform KnowledgeBase into CNF

	Regular	CNF
Sentence 1 :	$P \rightarrow Q$	$\neg P \vee Q$
Sentence 2 :	$Q \rightarrow R$	$\neg Q \vee R$
Sentence 3 :	$\neg R$	$\neg R$.

Proof:

1. $\neg P \vee Q$ sentence 1
2. $\neg Q \vee R$ sentence 2

3. $\neg R$ sentence 3
4. P Assume opposite
5. Q Resolve 4 & 1
6. R Resolve 5 & 2
7. Nil (Empty Clause) Resolve 6 & 3.
8. Therefore, original theorem ($\neg P$) is true.

Forward and Backward Chaining :-

- KB conjunction of Horn clauses.
- Horn clause (at most one literal is positive)
- For example : $(\neg P \vee \neg Q \vee V)$ is a Horn clause.
- So, $(\neg P \vee \neg W)$ is Horn clause, But $(\neg P \vee Q \vee V)$ is not.
- Definite clause : Exactly one literal is positive.
- Horn Clauses can be rewritten as implication
  $(\alpha \rightarrow \beta \,;\, \neg\alpha \vee \beta)$
  - *. Preposition symbol (facts) or
  - *. Conjunction of symbols
  - *. eg: $(\neg C \vee \neg B \vee A)$ becomes $((C \wedge B) \Rightarrow A)$

Forward chaining:

Idea : Fire any rule whose premises are satisfied in the KB.
- add its conclusion to the KB, until query is found.

$P \Rightarrow Q$

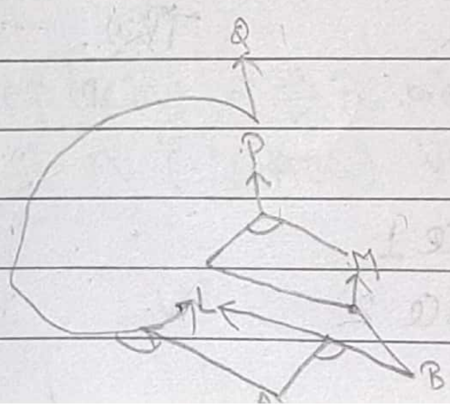
$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A



$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

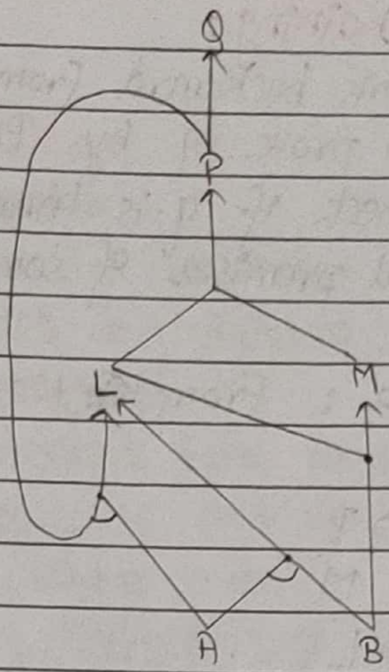
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

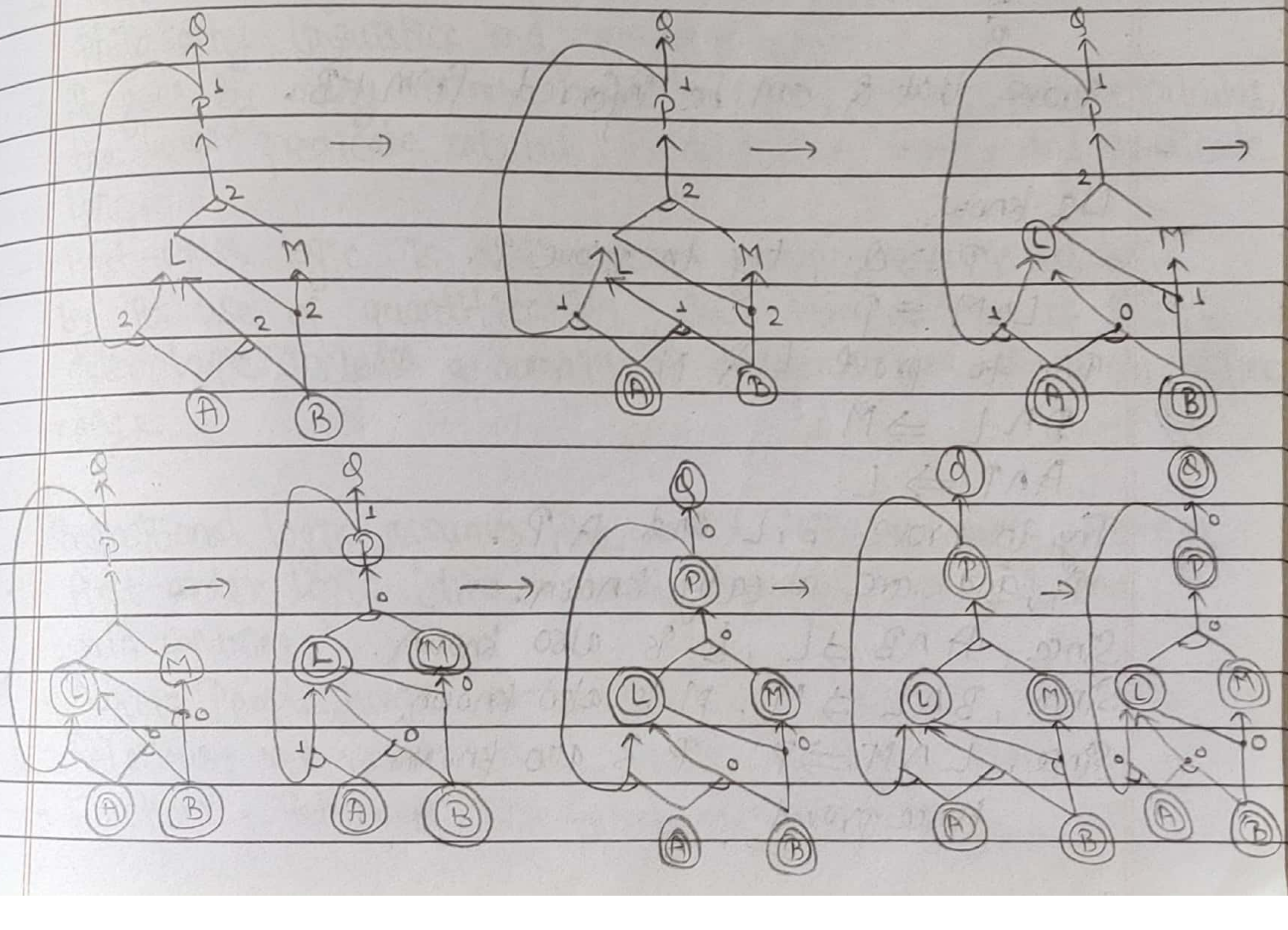
$A \wedge B \Rightarrow L$

A

B



Prove that Q can be inferred from KB.



Backward chaining:

Idea: Work backward from the query q;
to prove q by BC;
Check if q is known already or prove by BC all premises of some rules concluding q.

For example: From KB,

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Prove that Q can be inferred from KB.

We know,
$P \Rightarrow Q$, try to prove P.
$L \wedge M \Rightarrow P$
Try to prove L & M
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
Try to prove B, L and A, P.
A, B are already known.
Since, $A \wedge B \Rightarrow L$, L is also known.
Since, $B \wedge L \Rightarrow M$, M is also known.
Since, $L \wedge M \Rightarrow P$, P is also known.
Hence proved.

First-Order logic:

Pros and Cons of Propositional logic:

- Propositional logic is declarative.
- Propositional logic allows partial / disjunctive / negated information. (Unlike must data structures and database)
- Propositional logic is compositional
  - *. Meaning of $B \wedge P$ is derived from meaning of B and P.
- Meaning in propositional logic is context-independent. (Unlike natural language, where meaning depends on context).
- Propositional logic has very limited expressive power. (Unlike natural language).
- First-Order logic is a formal logical system used in mathematics, philosophy, linguistics and computer sience.
- It goes by many names, including: first-order predicate calculus, the lower predicate calculus, quantification theory and predicate logic.
- First-order logic is distinguished from propositional logic by its use of quantification; each interpretation of first-order logic includes a domain of discourse over which quantifiers range.

Propositional logic assumes the world contains facts, whereas first-order logic (like natural language) assumes the world contains:

- Object: People, house, number, colors....
- Relations: red, round, prime, brother of, bigger than, ...
- Function: Father of, Best friend, one more than, ...

Syntax of FOPL : Basic Element

- Constant : KingJohn, 2, ...
- Predicates : Brother, >, ...
- Functions : sqrt, LeftOfLeg, ...
- Variables : $x, y, a, b, \ldots$
- Connectives : $\neg, \Rightarrow, \vee, \wedge, \Leftrightarrow$
- Equality : =
- Quantifiers : $\forall, \exists$

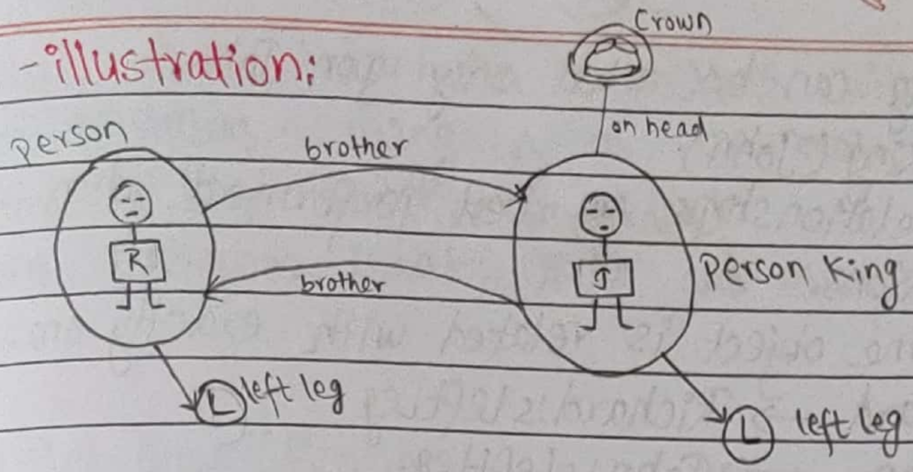
Example : King John and Richard the lionheart are brothers.

$$\underbrace{\underbrace{\text{Brother}}_{\text{predicate}}(\underbrace{\underbrace{\text{KingJohn}}_{\text{constant}}}_{\text{term}}, \underbrace{\underbrace{\text{Richard the Lionheart}}_{\text{constant}}}_{\text{term}})}_{\text{Atomic Sentence}}$$

- The length of LeftLeg of Richard is greater than the length of left Leg of King John.

$$\underbrace{\underbrace{>}_{\text{Predicate}}(\underbrace{\underbrace{\text{Length}}_{\text{function}}(\underbrace{\text{LeftLegOf}}_{\text{function}}(\underbrace{\text{Richard}}_{\text{constant}}))}_{\text{term}}, \underbrace{\underbrace{\text{Length}}_{\text{function}}(\underbrace{\text{LeftLegOf}}_{\text{function}}(\underbrace{\text{KingJohn}}_{\text{constant}}))}_{\text{term}})}_{\text{Atomic Sentence}}$$

FOPL - illustration:



- Five Objects
* Richard the Lionheart
* Evil King John
* Left leg of Richard
* Left leg of John
* The Crown

- Objects are related with relations.
- For example, King John and Richard are related with Brother relationship.
- This relationship can be denoted by (Richard, John), (John, Richard).
- Again, the crown and King John are related with on Head relationship.

    On Head (Crown, John)

- Brother and OnHead are binary relations as they relate couple of objects.
- Properties are relations that are unary.
- In this case, person can be such a property acting upon both Richard and John person.

    Person (Richard)
    Person (John)

- Again, king can be acted only upon John.

  king (John)
- Certain relationships are best performed when expressed as functions.
- Means one object is related with exactly one object.

  Richard $\rightarrow$ Richard's leftleg

  John $\rightarrow$ John's leftleg.

Quantifiers:

- Allows us to express properties of collections of objects instead of enumerating objects by name.
- Two quantifiers are:
  1) Universal : "for all" $\forall$
  2) Existential : "there exists" $\exists$

1) Universal quantification
- $\forall$ <variable> <sentence>
- Eg: Everyone at UAB is smart.

  $\forall x \; At(x, UAB) \Rightarrow Smart(x)$
- $\forall x \; P$ is true in a model m iff P is true for all x in the model.
- Roughly speaking, equivalent to the conjunction of instantiations of P.

  $At(KingJohn, UAB) \Rightarrow Smart(KingJohn) \wedge$

  $At(Richard, UAB) \Rightarrow Smart(Richard) \wedge$

  $At(UAB, UAB) \Rightarrow Smart(UAB) \wedge \ldots$

- Typically, $\Rightarrow$ is the main connectives with $\forall$.
- Common mistakes: using $\wedge$ as the main connective with $\forall$.

$\forall x\ At(x, UAB) \wedge Smart(x)$

means everyone is at UAB and everyone is smart.

2) Existential quantification

- $\exists$ <variable> <sentence>
- Someone at UAB is smart.

$\exists x\ At(x, UAB) \wedge Smart(x)$.

- $\exists x\ P$ is true in a model m iff P is true for at least one x in the model.
- Roughly speaking, equivalent to the disjunctions of conjunction of instantiation of P.

$At(KingJohn, UAB) \wedge Smart(KingJohn) \vee$
$At(Richard, UAB) \wedge Smart(Richard) \vee$
$At(UAB, UAB) \wedge Smart(UAB) \vee \ldots$

- Typically, $\wedge$ is the main connective with $\exists$.
- Common mistakes: using $\Rightarrow$ as the main connective with $\exists$.

$\exists x\ At(x, UAB) \Rightarrow Smart(x)$ is true even if there anyone who is not at UAB.

FOPL: Inference (Inference in First-Order Logic):

- First-order inference can be done by converting the knowledge base to PL and using propositional inference.
- How to convert universal quantifiers?
  *. Replace variable by ground terms.
- How to convert existential quantifiers?
  *. Skolemization

Universal instantiation (UI):

- Substitute ground term (term without variables) for the variables.
- For example, consider the following KB.

$\forall x \; King(x) \wedge Greedy(x) \Rightarrow Evil(x)$
$King(John)$
$Greedy(John)$
$Brother(Richard, John)$

It's UI is:

$King(John) \wedge Greedy(John) \Rightarrow Evil(John)$
$King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard).$
$King(John)$
$Greedy(John)$
$Brother(Richard, John).$

Existential instantiation (EI):

- For any sentence α and variable v in that, introduce a constant that is not in the KB (called Skolem constant) and substitute the constant for v.

Eg: Consider the sentence, $\exists x$ [Crown(x) $\wedge$ onHead(x, John)]

After EI,

Crown(C1) $\wedge$ onHead(C1, John)

where,

C1 is skolem constant.

Q: Anyone passing his history exams and winning the lottery is happy. But anyone who studies or is lucky can pass all his exams. John did not study but John is lucky. Anyone who is lucky wins the lottery.

1. $\forall x$ Pass(x, History) $\wedge$ win(x, Lottery) $\Rightarrow$ Happy(x)
2. $\forall x \forall y$ Study(x) $\vee$ Lucky(x) $\Rightarrow$ Pass(x, y)
3. $\neg$Study(John) $\wedge$ Lucky(John)
4. $\forall x$ Lucky(x) $\Rightarrow$ win(x, Lottery)

Unification :-

- Unification is all about making the expression look identical. So for the given expression to make them look identical we need to do substitution.
- A unifier of two atomic formulae is a substitution of terms for variables that makes them identical.
  - * Each variables has at most one associated term.
  - * Substitutions are applied simultaneously.
- Unifier of $P(x, f(a), z)$ and $P(z, z, u)$: $\{x/f(a), z/f(a), u/f(a)\}$
- We can get the inference immediately if we can find a substitution α such that King(x) and Greedy(x) match King(John) and Greedy(y).

  $\alpha = \{x/John, y/John\}$ works.

- Unify $(\alpha, \beta) = \theta$ if $\alpha\theta = \theta\beta$.

P	q	
Knows (John, x)	Knows (John, Jane)	{x / Jane}
Knows (John, x)	Knows (y, OJ)	{x / OJ, y / John}
Knows (John, x)	Knows (y, mother(y))	{y / John, x / mother(y)}
Knows (John, x)	Knows (x, OJ)	{fail}.

- Last unification is failed due to overlap of variables, x cannot take the values of John and OJ at the same time.
- We can avoid this problem by renaming to avoid the name clashes (standardizing apart).

Eg:- Unify {knows (John, x) knows (z, OJ)}
= {x | OJ, z | John}

- Let C_1 and C_2 be two clauses. If C_1 and C_2 have no variables in common, then they are said to be standardized apart.
Standardized apart eliminates overlap of variables to avoid clashes by renaming variables.

Another complication:
- To unify knows (John, x) & knows (y, z) gives
α = {y | John, x/z}

or,

α = {y | John, x | John, z | John}

Generalized Modus Ponens (GMP):

$$\frac{P_1', P_2', \ldots, P_n', (P_1 \wedge P_2 \wedge \ldots \wedge P_n \Rightarrow q)}{q\theta}$$

where,
$P_i'\theta = P_i\theta$ for all i.

P_1' is King (John)	P_1 is King (x).
P_2' is Greedy (y)	P_2 is Greedy (x)
	q is Evil (x).

θ is {x | John y | John}
$q\theta$ is Evil (John).

Forward Chaining :-

- When based on the available data a decision is taken, then the process is called as the forward chaining.
- It works from a initial state and reaches to goal state.

Example :-

Given facts :

- It is a crime for an American to sell weapons to the enemy of America.
- Country Nono is an enemy of America.
- Nono has some missiles.
- All the missiles were sold to Nono by colonel.
- Missile is a weapon.
- Colonel is American.

We have to prove that colonel is criminal.

Facts to FOPL :

$\Rightarrow$ American (x) ∧ Weapon (y) ∧ Sell (x, y, z) ∧ ~~enem~~ enemy (z, America) read as: z is enemy of America

$\Rightarrow$ Criminal (x)

$\Rightarrow$ Enemy (Nono, x) country Nono is an enemy of American

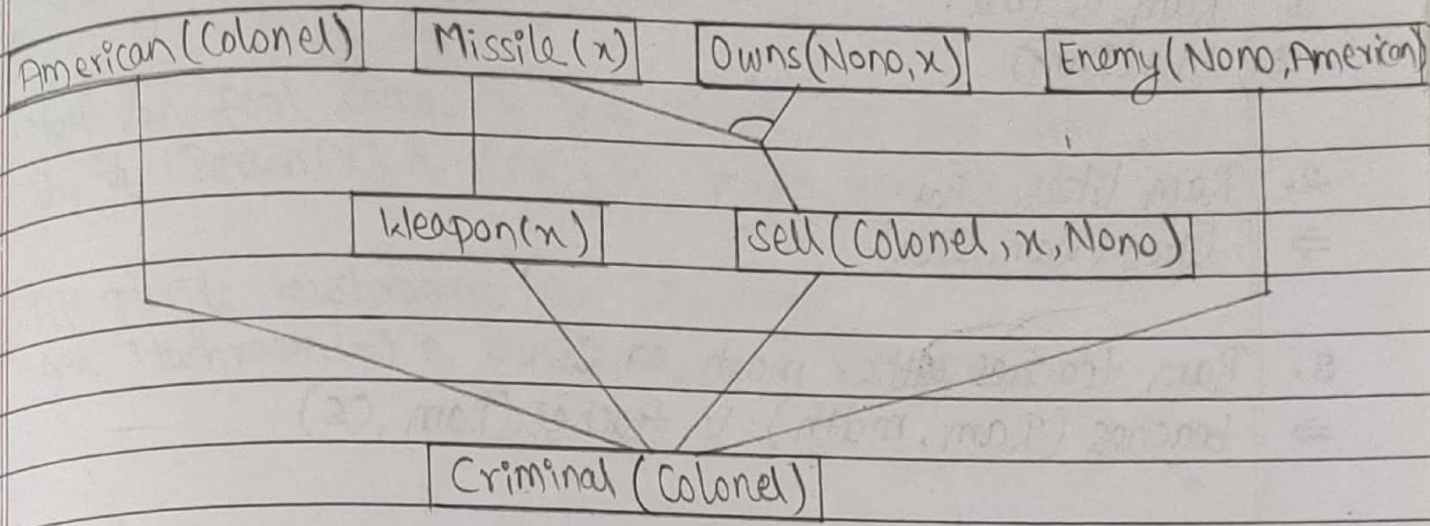
$\Rightarrow$ Owns (Nono, x)
Missile (x)

$\Rightarrow$ ∀x Missile (x) ∧ Owns (Nono, x)
$\Rightarrow$ Sell (Colonel, x, Nono) colonel sells missiles to Nono.

$\Rightarrow$ Missile (x) $\Rightarrow$ Weapon (x)

$\Rightarrow$ American (colonel)

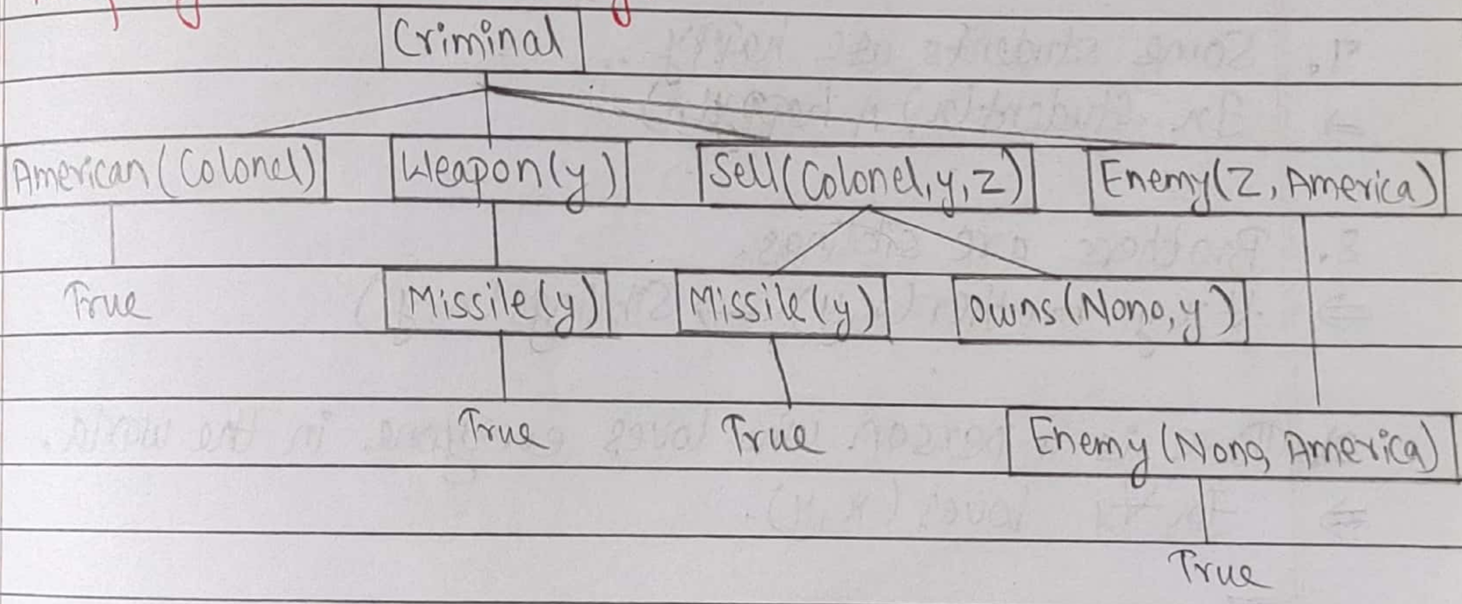
Proof by forward chaining :-



Backward Chaining:

- If based on the decision the initial state is fetched, then it is called as backward chaining.
- Backward chaining is also called as a decision-driven or goal driven inference technique.

Proof by Backward Chaining:



First Order Predicate Logic :-

1. Ram is tall.
⇒ tall(Ram)

2. Ram likes Sita.
⇒ likes(Ram, Sita)

3. Ram teaches either math or CS.
⇒ teaches(Ram, math) $\vee$ teaches(Ram, CS)

4. All students like football.
⇒ $\forall x$ Student(x) $\rightarrow$ like(x, football)

5. Some students like football.
⇒ $\exists x$ Student(x) $\wedge$ likes(x, football)

6. All students are happy.
⇒ $\forall x$ Students(x) $\rightarrow$ happy(x)

7. Some students are happy.
⇒ $\exists x$ Student(x) $\wedge$ happy(x)

8. Brothers are siblings.
⇒ $\forall x, y$ brother(x, y) $\rightarrow$ Siblings(x, y)

9. There is a person who loves everyone in the world.
⇒ $\exists x \forall y$ loves(x, y).

10. Every gardener likes the sun.

$\Rightarrow$ $\forall x$ gardener $\rightarrow$ likes (x, Sun)

11. You can fool some of the people all the time.

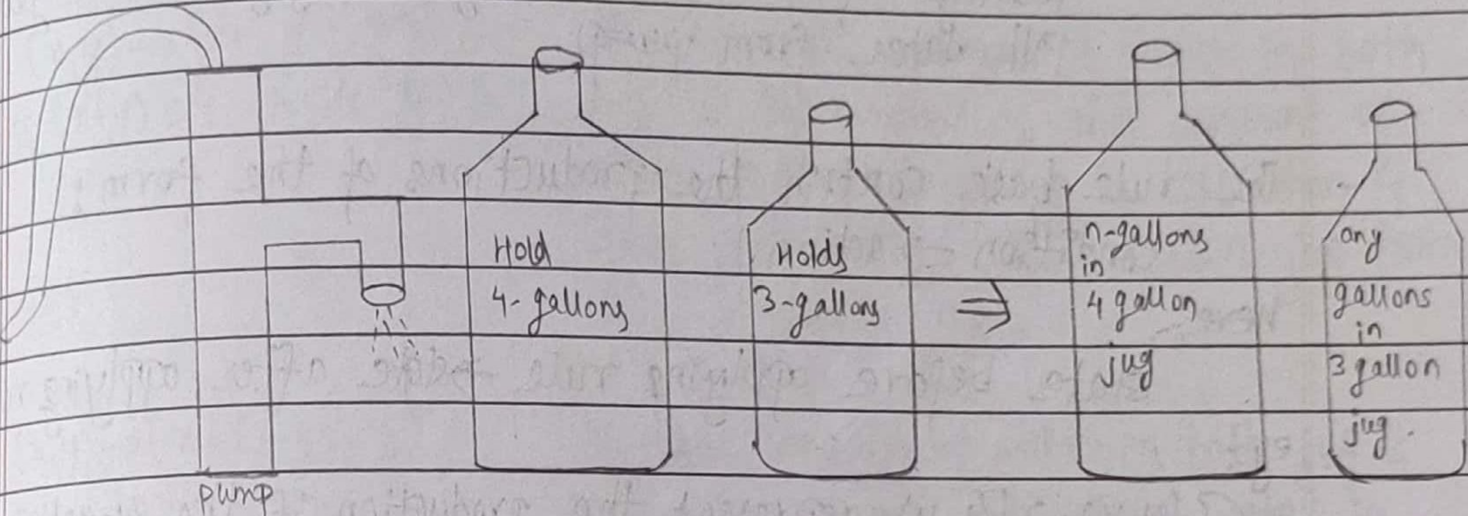
$\Rightarrow$ $\exists x \forall t$ $Person(x) \wedge time(t) \rightarrow can\_fool(x, t)$.

12. All purple mushrooms are poisonous.

$\Rightarrow$ $\forall x$ $Mushroom(x) \wedge Purple(x) \longrightarrow Poisonous(x)$

Water jug problem:

There are two jugs, a four-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 as shown in figure below. How can you get exactly n (0,1,2,3,4) gallons of water into one of the two jugs?



⇒ Goal: (4,2)

⇒ State representation: A state of the problem is represented as a tuple (x,y) where:

$x \in \{0,1,2,3,4\}$ represents the amount of water in 4-gallon jug.

$y \in \{0,1,2,3\}$ represents the amount of water in 3-gallon jug.

Therefore, state is the amount of water in each jugs.

⇒ Initial state: The initial state represents the initial content of the two jugs. For instance, it may be (2,3) meaning that the 4-gallon jug contain 2 gallon of water and the 3-gallon jug contains 3 gallon of water.

⇒ goal: specified amount of water in both jugs.

⇒ Action : Pouring of water from 4 gallon jug into 3-gallon
Pouring of water from 3-gallon jug into ground
Pouring of water from 4-gallon jug into ground.
Pouring of water from 3-gallon jug into 4-gallon
Fill water from pump.

- The rule base contain the productions of the form:
condition → action.

here,
state before applying rule → state after applying rule.

eg:
$(x,y) \rightarrow (4,y)$ if $(x<4)$ represent the production if the 4-gallon jug is not full then fill it from the pump.

The rule base contains the following productions:

	Production rules	Description
1.	$(x,y) \rightarrow (4,y)$ if $(x<4)$	If 4-gallon jug is not full then fill it from the pump.
2.	$(x,y) \rightarrow (x,3)$ if $(y<3)$	If 3-gallon jug is not full then fill it from the pump.

classmate Date Page

	Production rules	Description.
3.	$(x,y) \rightarrow (0,y)$ if $x>0$	If 4-gallon jug contains amount of water then empty it on ground.
4.	$(x,y) \rightarrow (x,0)$ if $y>0$	If 3-gallon jug contains amount of water then empty it on ground,
5.	$(x,y) \rightarrow (4, y-(4-x))$ if $(x+y) \geq 4$ $x<4$ $y>0$	If the total amount of water in both jug is more than 4 and amount of water in 4-gallon jug is less than 4 then completely fill 4-gallon jug from the 3-gallon jug.
6.	$(x,y) \rightarrow (x-(3-y), 3)$ if $(x+y) \geq 3$ $x>0$ $y<3$	If total amount of water in both jug is more than 3 and amount of water in 3-gallon jug is less than 3 then completely fill 3-gallon jug from the 4-gallon jug.
7.	$(x,y) \rightarrow (x+y, 0)$ if $(x+y) \leq 4$, $y>0$	If total amount of water in both jug is less than 4 and 3-gallon jug contains some water then empty the 3-gallon jug into the 4-gallon jug.
8.	$(x,y) \rightarrow (0, x+y)$ if $(x+y) \leq 3$, $x>0$	If the total amount of water in both jug is less than 3 and 4-gallon jug contains some water then empty the 4-gallon jug into the 3-gallon jug.

Bayesian network :-

- A data structure to represent the dependencies among variables and to give a concise specification of any full joint probability distribution.
- Also called belief network or probabilistic network or causal network or knowledge map.
- The basic idea is:
- Knowledge in the world is modular i.e. most events are conditionally independent of most other event.
- Adopt at model that can use a more local representation to allow interaction between events that only affect each other.
- Some events may only be unidirectional others may be bidirectional - make a distinction between these in model.
- Events may be casual and thus get chained together in a network.

- A Bayesian network is a directed acyclic graph which consist of:

i) Node :-
Each node correspond to a random variable, which may be discrete or continuous.

ii) Arrow :-
A set of directed lines or arrows connects pair of nodes. If there is an arrow from node 'X' to node 'Y', X is said to be a parent of Y. The graph has no directed cycles.

iii) Each node X_i has a conditional probability distribution $P(X_i / Parent(X_i))$ that quantifies the effect of parents on the nodes.

Intuitions:

- A Bayesian network models our incomplete understanding of the casual relationship from an application domain.
- A node represents some state of affairs or event.
- A link from X to Y means that X has a direct influence on Y.

Implementation:

- A Bayesian network is a directed acyclic graph; a graph where the directions are links which indicate dependencies that exist between nodes.
- Node represent propositional about events or events themselves.
- Conditional probabilities quantifies the strength of dependencies.

Example:-

- Consider the simple world, consisting of variable toothaches, cavity catch and weather. In this world weather is independent of the other variables.
- These relationship are represented by the Bayesian network structure as shown in figure below:

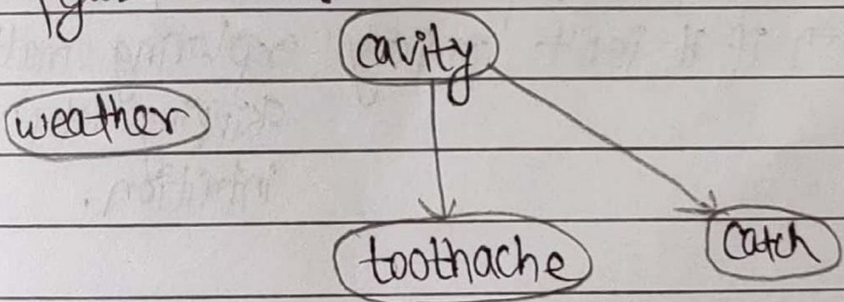


Fig:- Simple Bayesian network.

Differences between Inference and Reasoning:

Inference	Reasoning
Inference can be defined in general term, representing derivation of new knowledge from existing knowledge and axioms within a single step.	Reasoning can be defined in a context of a goal and is carried out via search process involving multiple inferences.
It is a set of rules for generating new sentences that are necessarily true given that the old sentences are true.	Reasoning arise question that 'what conclusion is valid in what situation?
For example: Modus tollens is a rule of inference which derives new knowledge	For example: In deciding whether a propositional formula is satisfiable or not, choices have to be made.
It is a single step of deductive chain.	A mental process of logic in deduction.
It can derive conclusion from premises even if it isn't properly deductive.	It can derive conclusion by exploring multiple pathways skipping steps and using our intuition.

classmate
Date
Page

Probabilistic Reasoning :-

Probabilistic Reasoning is basically logical reasoning based on the chance that some event will or will not occur. One could use a coin flip as an example of probabilistic reasoning.

Importance:

Applying them is societal ~~changes~~ trends, advertisors such as consumer report often use probability statistics to sell their products. Automobile to the public that their product is better than the other and often time these probabilities.

2068/2074 Q?

For each of the following agents, determine what type of agent architecture is most appropriate (i.e. table lookup, simple reflex, goal-based or utility based).

a) Medical diagnosis system

Agent : Medical diagnosis system
Action : Questions, tests, treatments
Goal : Healthy patient, minimize cost.
Environment : patient, Hospital, staffs.
Agent architecture : Table-look up Agent.
Percepts : System, finding patients answers.

b) Satellite image analysis system:-

Agent: Satellite image analysis system
Percepts: pixels of varying intensity color
Goal: Correct image categorization
Environment: Downlink from arbitrary stallice
Agent architecture: Simple-Reflex Agent.
Actions: Print a categorization of scene.

c) Part picking robot:-

Agent: Part-picking robot
Percepts: Duel of varying intensity
Actions: Pick up parts and sort into bins
Goals: place parts in correct bins.
Environment: convey or belts with parts bins
Architecture: Goal-based Agent.

d) Refinery controller:-

Agent: Refinery controller
Percepts: temperature, pressure reducing,
Actions: open, close, values, adjust temperature.
Goals: maximize purity, yield, safety.
Environment: Refinery, operators
Architecture: Simple-Reflex Agent