

Chapter 1

INTRODUCTION TO AI

Intelligence

The ability to reason

The ability to understand

The ability to create

The ability to Learn from experience

The ability to plan and execute complex tasks

The intelligent behavior may include

- **Everyday Tasks:** recognize a friend, recognize who is calling, translate from one language to another, interpret a photograph, talk, and cook a dinner
- **Formal Tasks:** prove a logic theorem, geometry, calculus, play chess, checkers, or Go
- **Expert Tasks:** engineering design, medical designers, financial analysis

Artificial Intelligence

- › AI is the branch of computer science concerned with making computers behave like humans.
- › AI is the science and engineering of making intelligent machines, especially intelligent computer programs.
- › The process may include
 - ✗ Learning (Gaining of information and rules for using the information)
 - ✗ Reasoning (Using the rules to reach approximate or definite conclusions)
 - ✗ Self-Correction

According to Barr and Feigenbaum:

“Artificial Intelligence is the part of computer science concerned with designing intelligence computer systems, that is, systems that exhibit the characteristics we associate with intelligence in human behavior.”

According to Elaine Rich:

“AI is the study of how to make computers do things at which, at the moment, people are better”

An AI system should have

- Capability to provide reason about something
- Capability of natural language processing
- Capability of learning past experience
- Capability of self-correction

Views of AI fall into four categories

Thinking humanly

Acting humanly

Thinking rationally

Acting rationally

	Humanly	Rationally
Thinking	Thinking humanly – cognitive modeling. Systems should solve problems the same way humans do.	Thinking rationally – the use of logic. Need to worry about modeling uncertainty and dealing with complexity.
Acting	Acting humanly – the Turing Test approach.	Acting rationally – the study of rational agents: agents that maximize the expected value of their performance measure given what they currently know.

Thinking humanly: *The cognitive modeling approach*

- If we are going to say that a given program thinks like a human, we must have some way of determining how humans think.
- We need to get inside the actual workings of human minds.
- Two ways of doing this is:
 - Predicting and testing human behavior (cognitive science)
 - Identification from neurological data (Cognitive neuro science)
- If the program's input/output and timing behavior matches human behavior, that is evidence that some of the program's mechanisms may also be operating in humans.

Acting humanly: *The Turing Test approach*

- The Turing Test is a method for determining whether or not a computer is capable of thinking like a human. The test is named after Alan Turing, an English mathematician who pioneered artificial intelligence during the 1940s and 1950s, and who is credited with devising the original version of the test. According to this kind of test, a computer is deemed to have artificial intelligence if it can mimic human responses under specific conditions.

Thinking rationally: *The “laws of thought approach”*

- Undeniable reasoning processes
- The Greek philosopher Aristotle was one of the first to attempt to codify ‘‘right thinking,’’
- His famous syllogisms provided patterns for argument structures that always gave correct conclusions given correct premises.
- For example, ‘‘Socrates is a man; all men are mortal; therefore Socrates is mortal.’’ These laws of thought were supposed to govern the operation of the mind, and initiated the field of logic.

Acting rationally : *The rational agent approach*

- Acting rationally means acting so as to achieve one's goals, given one's beliefs.
- An agent is just something that perceives and acts.
- In this approach, AI is viewed as the study and construction of rational agents.

Artificial Intelligence Problem Characteristics :-

1. Decomposable to smaller or easier problems
2. Solution steps can be ignored or undone
3. Predictable problem universe
4. Good solutions are obvious
5. Uses internally consistent knowledge base
6. Requires lots of knowledge or uses knowledge to constrain solutions
7. Requires periodic interaction between human and computer

The Turing Test approach

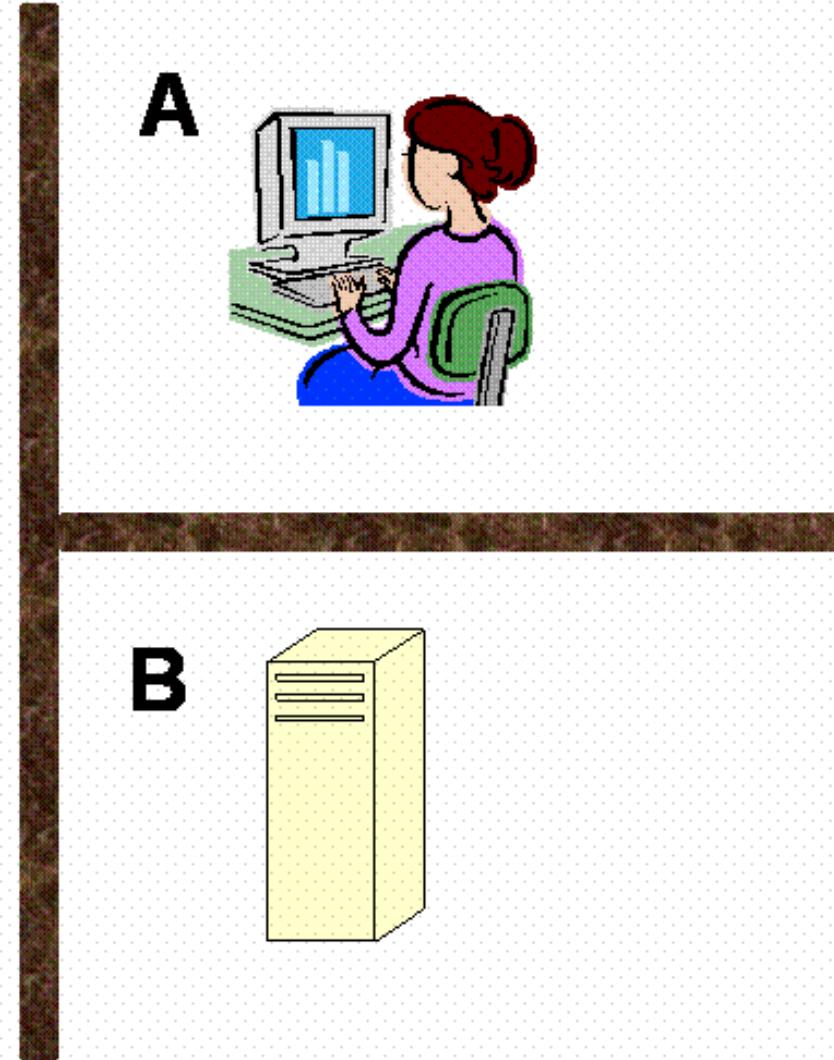
The Turing Test is a method for determining whether or not a computer is capable of thinking like a human.

The test is named after Alan Turing, an English mathematician who pioneered artificial intelligence during the 1940s and 1950s, and who is credited with devising the original version of the test.

According to this kind of test, a computer is deemed to have artificial intelligence if it can mimic human responses under specific conditions.



Interrogator



Consider the following setting.

There are two rooms, A and B. One of the rooms contains a computer. The other contains a human. The interrogator is outside and does not know which one is a computer. He can ask questions through a teletype and receives answers from both A and B. The interrogator needs to identify whether A or B are humans. To pass the Turing test, the machine has to fool the interrogator into believing that it is human.

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

AI and related fields/ Foundations of AI

Philosophy:

Logic, reasoning, mind as a physical system, foundations of learning, language and rationality.

Mathematics:

Formal representation and proof algorithms, computation, undesirability, intractability, probability.

Psychology:

Adaptation, phenomena of perception and motor control.

Economics:

Formal theory of rational decisions, game theory.

Linguistics:

Knowledge representation, grammar

Neuro science:

Physical substrate for mental activities

Control theory:

Homeostatic systems, stability, optimal agent design

Brief History of AI

The term “Artificial Intelligence” was used for the first time in 1956 by an American scientist John McCarthy who is referred to as the Father of AI. McCarthy also came up with a programming language called LISP (i.e. List-Processing), which is still used to program computer in AI that allow the computer to learn.

Further, the major achievements can be listed as below:

1943	First electronic computer “Colossus” was developed.
1949	First commercial stored program computer was developed.
1950	<ul style="list-style-type: none">- Alan Turing proposes the Turing test as a measure of machine intelligence.- Claude Shannon published a detail analysis of chess playing as search.- Isaac Asimov published his three laws of Robotics.
1951	The first working AI programs were written to run on the Ferranti Mark machine of the University of Manchester; a checkers-playing program written by Christopher Stavechey and a chess-playing program is written by Dietrich Prinz
1955	The first Dartmouth college summer AI conference is organized by John McCarthy, Marvin Minsky, Nathan Rochester of IBM and Claude Shannon.
1956	<ul style="list-style-type: none">- The name artificial intelligence is used for the 1st time as the topic of the second Dartmouth Conference, organized by John McCarthy.

	<ul style="list-style-type: none"> - The first demonstration of the Logic Theorist (LT) written by Allen Newell, J.C. Shaw and Merbart Simon pus is called the first AI program
1957	The general problem Solver (GPS) demonstrated by Newell, Shaw and Simon
1958	John McCarthy at MIT invented the Lisp Programming Language.
1959	<ul style="list-style-type: none"> - John McCarthy and Marvin Minsky founded the MIT AI Lab. - First industrial robot company, animation was established.
1972	Prolog programming language was developed by Alain Colmerauer
1980	First National Conference of the American Association for Artificial Intelligence (AAAI) was held at Stratford.
Mid 1980's	Neural networks become widely used with the Back propagation algorithm.
1994	AI system exist in real environments with real sensory inputs (i.e. Intelligent Agents)
1997	First time AI system controlled a spacecraft named “Deep Space II”
2007	Checkers is solved by a team of researchers of the University of Alberta.
Present	Programmers are still trying to develop a computer which can successfully pass the “Turing Test”.

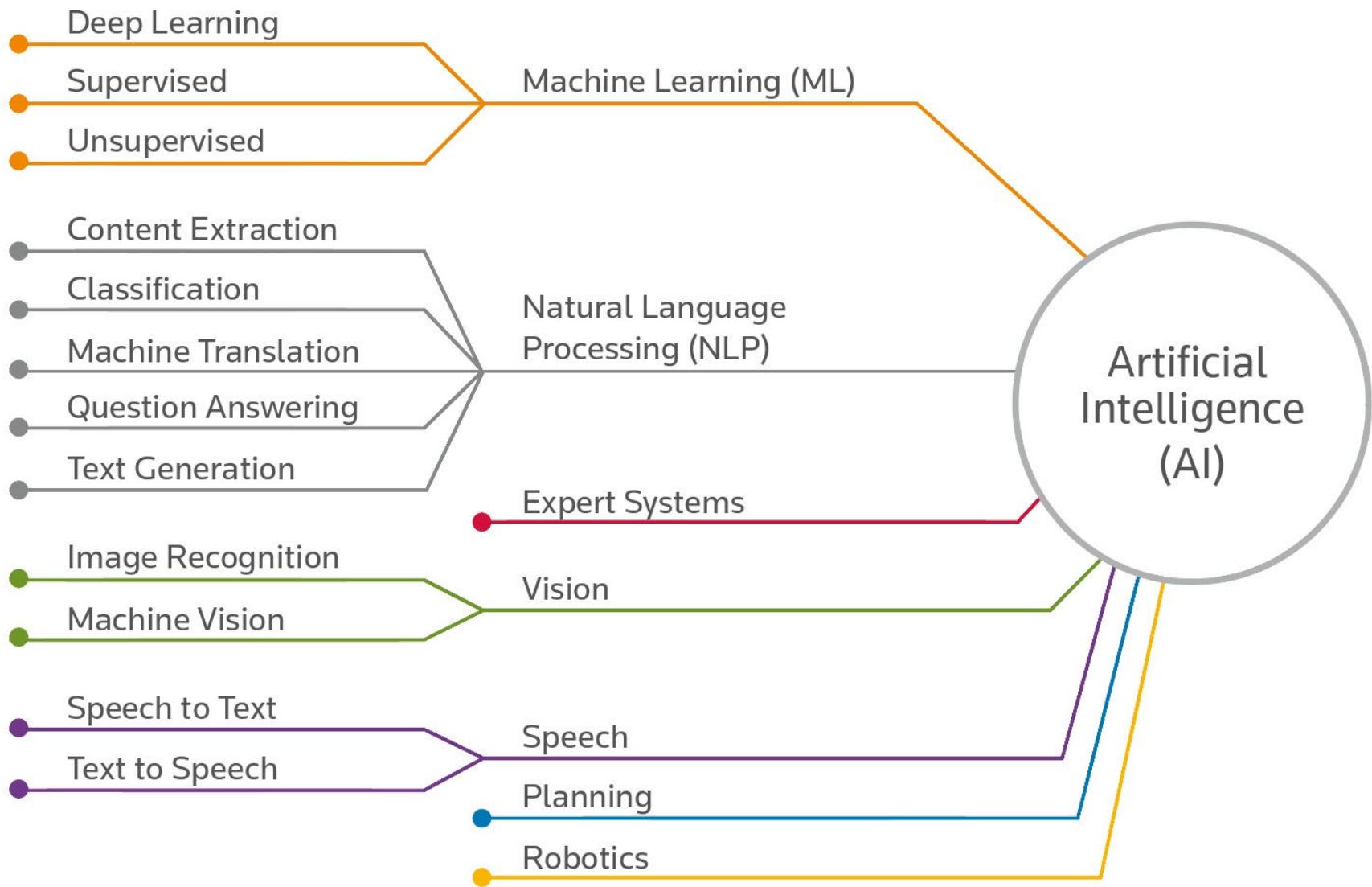
Difference between AI and NI

Application of AI

Artificial intelligence has been used in a wide range of fields including medical diagnosis, stock trading, robot control, law, remote sensing, scientific discovery and toys.

Many thousands of AI applications are deeply embedded in the infrastructure of every industry.

In the late 90s and early 21st century, AI technology became widely used as elements of larger systems, but the field is rarely credited for these successes.



Game Playing

Machines can play master level chess. There is some AI in them, but they well against people mainly through brute force method, looking at hundreds of thousands of positions.

Speech Recognition

It is possible to instruct some computers using speech. In 1990s, computer speech recognition reached a practical level for limited purposes.

Understanding Natural Language

To perform many natural language processing tasks such as machine translation, summarization, information extraction, word sense disambiguation need the AI in machine.

Computer Vision

Computer vision is concerned with the theory behind artificial system that extract information from images. The image data can take many forms such as videos sequences views from multiple cameras and data from a medical scanner.

Expert System

Expert system needs the AI to perform its task. One of the first expert system was MYCIN in 1974 which diagnosis bacterial infections of the blood and suggests treatments.

Finance

Financial institutions have long used artificial neural network systems to detect charges or claims outside of the norm, flagging these for human investigation.

Use of AI in banking can be traced back to 1987 when Security Pacific National Bank in USA set-up a Fraud Prevention Task force to counter the unauthorized use of debit cards.

Hospitals and medicine

Artificial neural networks are used as clinical decision support systems for medical diagnosis, such as in Concept Processing technology.

Other tasks in medicine that can potentially be performed by artificial intelligence include:

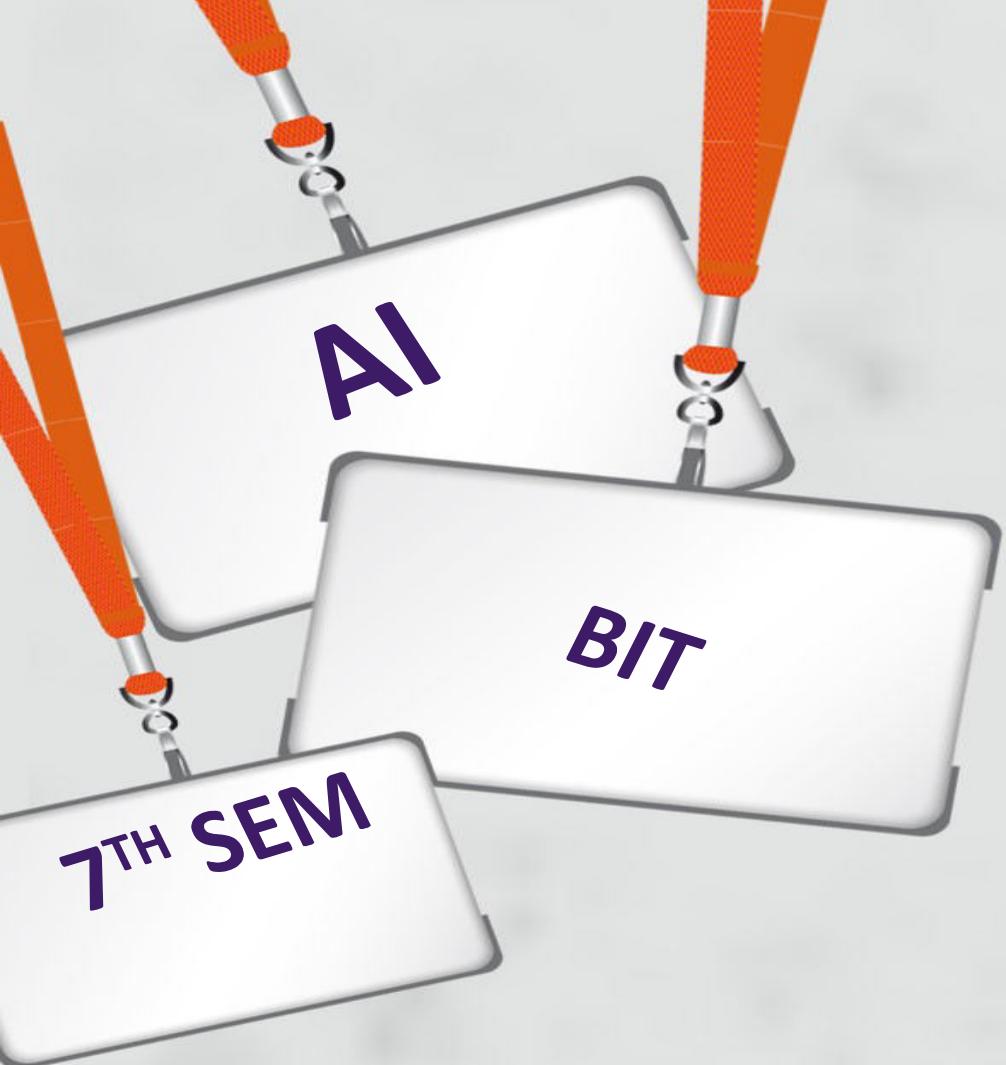
- ✗ Computer-aided interpretation of medical images. Such systems help scan digital images, *e.g.* from computed tomography, for typical appearances and to highlight conspicuous sections, such as possible diseases. A typical application is the detection of a tumor.
- ✗ Heart sound analysis
- ✗ Companion robots for the care of the elderly

Music

The evolution of music has always been affected by technology. With AI, scientists are trying to make the computer emulate the activities of the skillful musician. Composition, performance, music theory, sound processing are some of the major areas on which research in Music and Artificial Intelligence are focusing.

Aviation

The Air Operations Division (AOD) uses AI for the rule based expert systems. The AOD has use for artificial intelligence for replacement operators for fighting and training simulators, mission management aids, support systems for tactical decision making, and post processing of the simulator data into symbolic summaries.

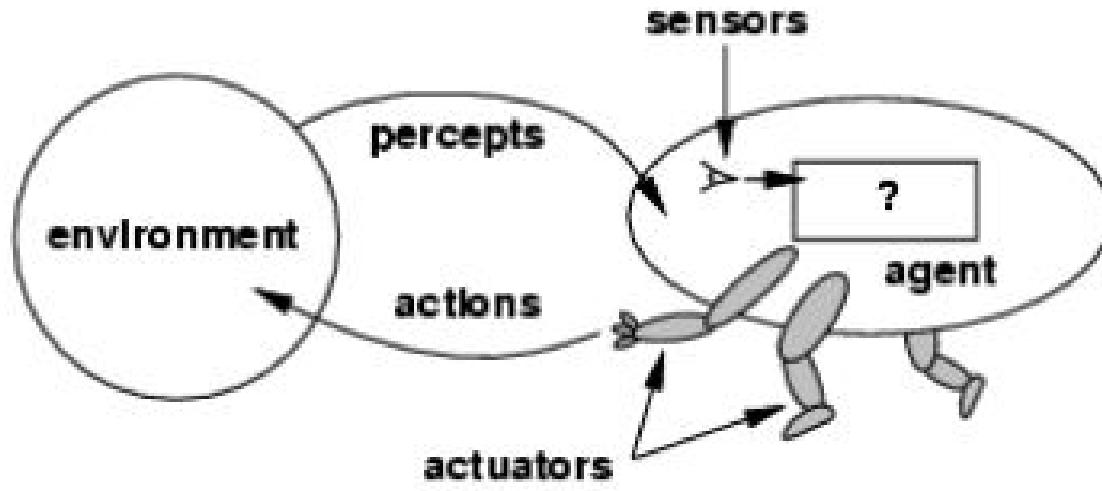


Chapter 2

AGENTS

Agent

- An AI system is composed of an agent and its environment. The agents act in their environment. The environment may contain other agents
- An agent is anything that can be viewed as:
 - perceiving its environment through sensors and
 - acting upon that environment through actuators



A **human agent** has sensory organs such as eyes, ears, nose, tongue and skin parallel to the sensors, and other organs such as hands, legs, mouth, for effectors.

A **robotic agent** replaces cameras and infrared range finders for the sensors, and various motors and actuators for effectors.

A **software agent** has encoded bit strings as its programs and actions.

Agent Terminologies

Performance Measure: determines how successful the agent is.

Behavior: activities the agent performs to achieve the goal

Percepts: formation of concepts based on the inputs.

Percept Sequence: set of all the perceptions till date

Agent Function: mapping of perception

Agent's structure

Agent = Architecture + Agent Program

- ✓ Architecture = the machinery that an agent executes on.
- ✓ Agent Program = an implementation of an agent function

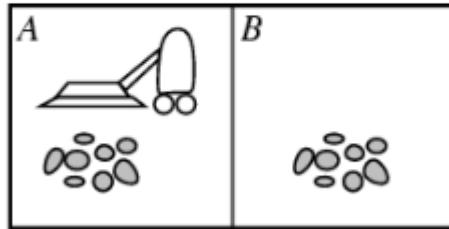
Agent Function

The agent function maps from percept sequences to actions:

$$[f: \mathcal{P}^* \rightarrow \mathcal{A}]$$

The agent program runs on the physical architecture to produce f

Vacuum-cleaner world



- **Percepts:**
location and state of the environment, e.g., [A,Dirty], [B,Clean]
- **Actions:**
Left, Right, Suck, No Op

Agent classification

- Collaborative Agents
- Interface Agents
- Mobile Agents
- Reactive agents

Assignment 1:

Explain Collaborative Agents , Interface Agents, Mobile Agents and Reactive agents

Properties of Agent

- Autonomous
- Interacts with other agents plus the environment
- Reactive to the environment
- Pro-active (goal- directed)

Environment types

- **Fully observable :**
An agent's sensors give it access to the complete state of the environment at each point in time
- **Deterministic :**
The next state of the environment is completely determined by the current state and the action executed by the agent.
- **Episodic :**
The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action)
- **Static :**
The environment is unchanged while an agent is deliberating.

Intelligent agents

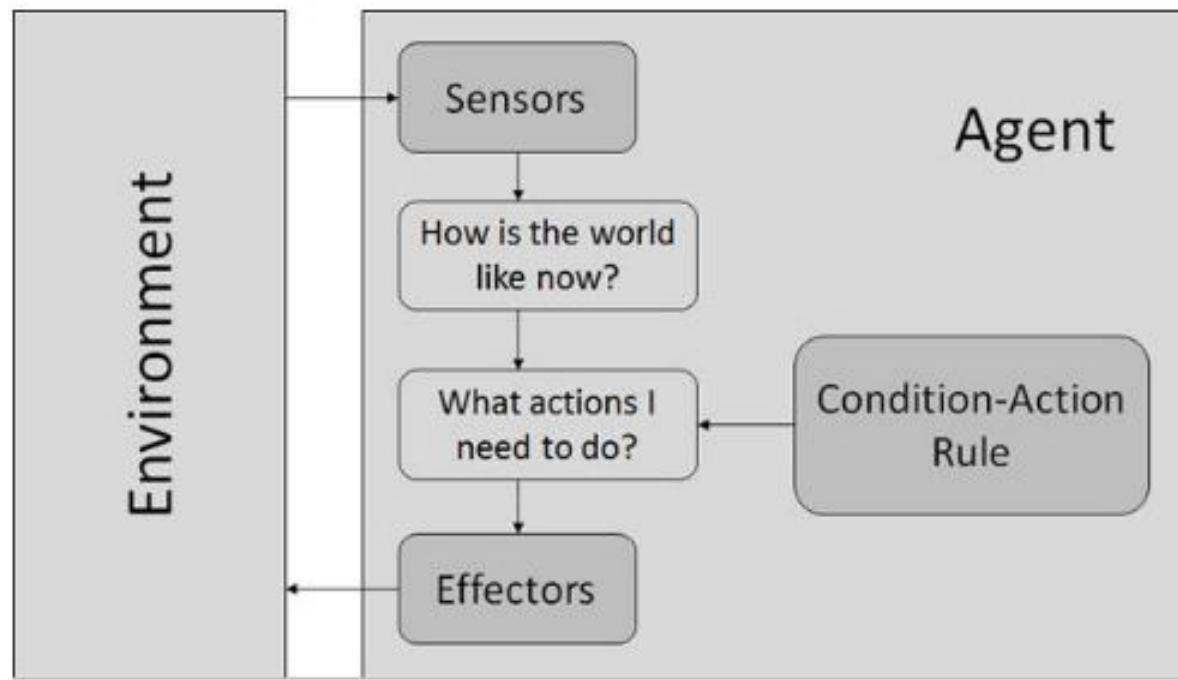
An Intelligent Agent is an **autonomous entity** which observes through sensors and acts upon an environment using actuators and directs its activities towards achieving the goal.

Hence, an agent gets percepts one at a time, and maps this percept sequence to actions.

Types of Agent

- Simple Reflex Agent

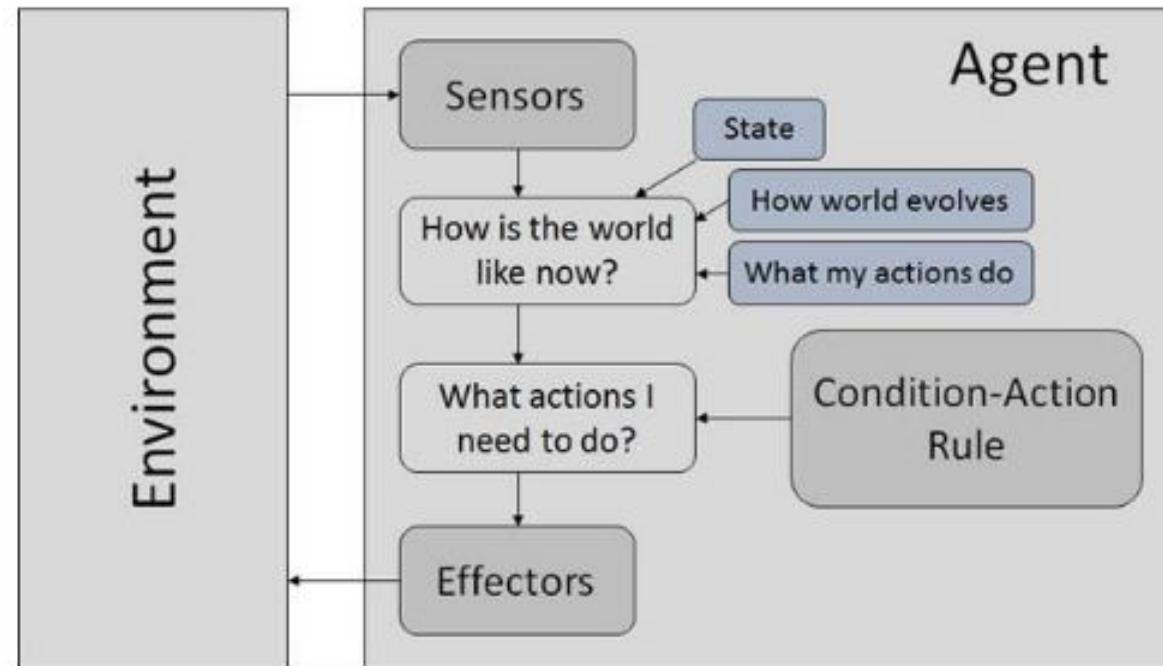
- They choose actions only based on the current percept or situation.
- This is useful when a quick automated system is required. Human have very similar reaction to fire.



(Condition- Action Rule/IF-THEN Rule IF hands on fire, THEN pull away)

- **Model Based Agents (Reflex Agent with Internal State)**

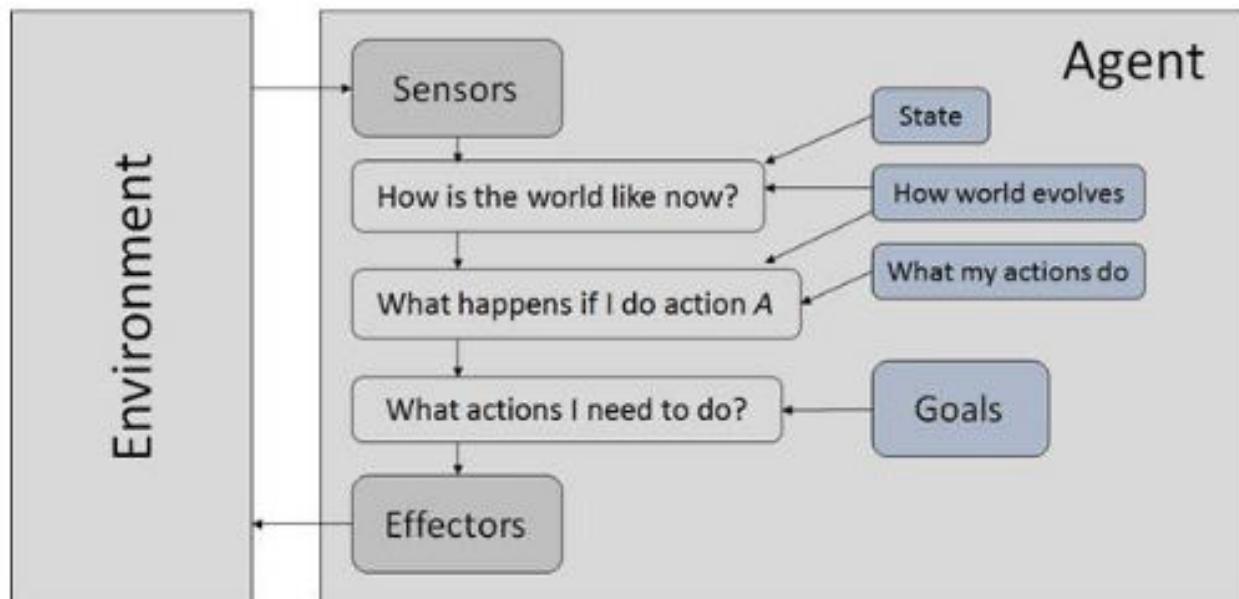
- They choose action only based on their model and maintain their internal state where model means the information about perception function and Internal States are



the representation of unobserved aspects of current state depending on percept history.

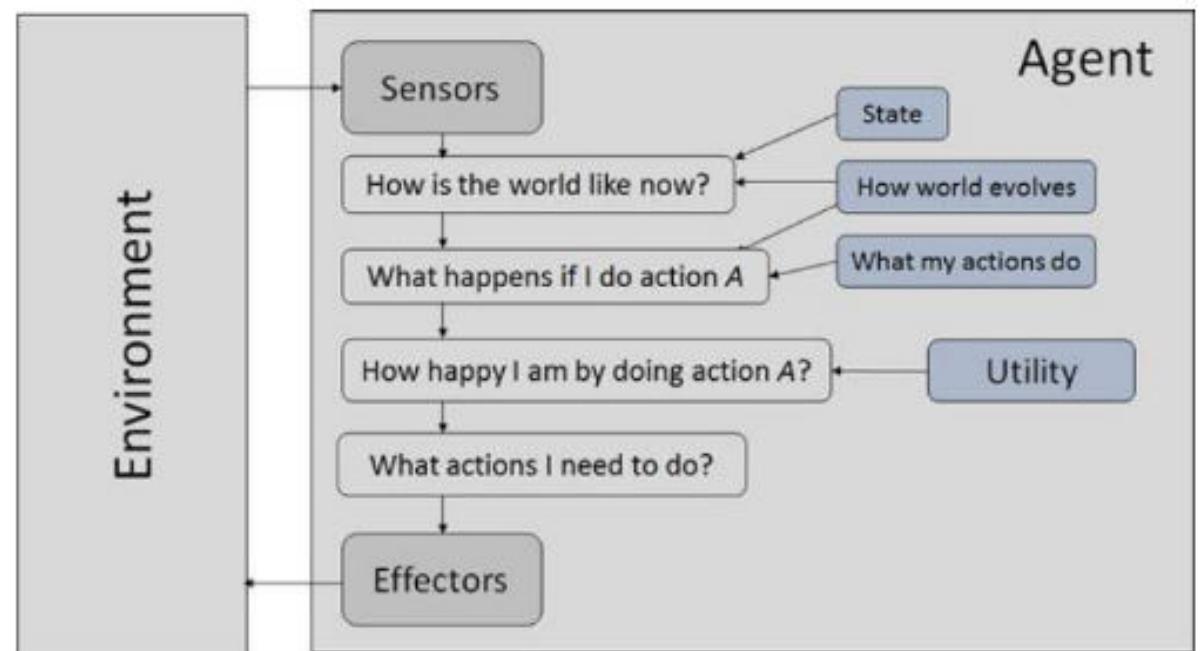
- Goal Based Agents

- They choose their actions in order to achieve goals. Goal-based approach is more flexible than reflex agent since the knowledge/information is modeled in such a way that they are easy for modifications.



- **Utility Based Agents**

- They choose actions based on a utility for each state.
- Utility function maps each state after each action to achieve the goal. This is useful when we either have



many actions for same goal or we have many goals for same actions.

PEAS

- To design a rational agent we must specify its task environment.
- Standing for performance, environment, actuators and sensors, PEAS define task environments about formulating the performance of intelligent agents.

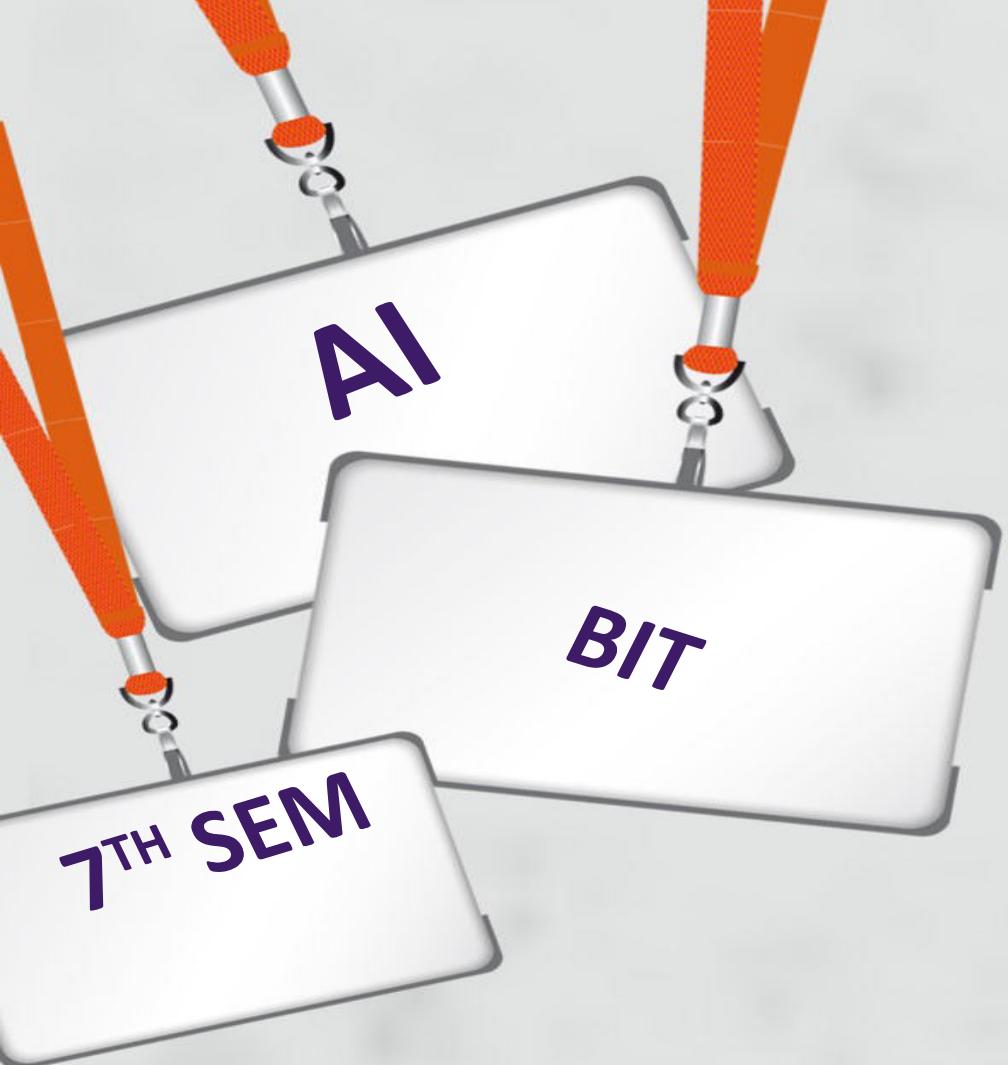
Q. Point out the task of designing an automated taxi driver according to PEAS description.

- Performance measure: Safe, fast, legal, comfortable trip, maximize profits
- Environment: Roads, other traffic, pedestrians, customers
- Actuators: Steering wheel, accelerator, brake, signal, horn
- Sensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors and keyboard



Q. Point out the task of designing a Medical diagnosis system according to PEAS description.

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



Chapter 3

PROBLEM SOLVING

Planning

- Planning is the process of deciding in detail how to do something before you actually start to do it.
- It is the task of coming up with a sequence of actions that will achieve a goal.
- **Planner** is viewed as the producer or generator of the solution.
- Eg: STRIPS (STanford Research Institute Problem Solver)

- **Classical Planning**

Planning for which environments that are fully observable, deterministic, finite, static (change happens only when the agent acts), and discrete (in time, action, objects, and effects)

- **Non Classical Planning**

Planning for which environments for partially observable and involves a different set of algorithms and agent designs

Problems

- A problem is a situation (difficult/easy) experienced by an agent.
- Problem is solved by a sequence of actions that reduce the difference between the initial situation and the goal.

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.

Four general steps in problem solving:

- Goal formulation
 - What are the successful world states
- Problem formulation
 - What actions and states to consider given the goal
- Search
 - Determine the possible sequence of actions that lead to the states of known values and then choosing the best sequence.
- Execute
 - Give the solution perform the actions.

Problem formulation:

A problem is defined by:

- An initial state: State from which agent starts
- 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.

A solution is a sequence of actions from initial to goal state. Optimal solution has the lowest path cost.

Defining problem as a state space search

A set of initial state, actions and the goal state for a given problem is known as state space for that problem. A state space represents a problem in terms of states and operators that changes the states. A state space essentially consists of a set of nodes representing each state of the problem, arcs between nodes representing the legal moves from one state to another, an initial state and a goal state

The major components of state space representation are:

- Initial state
- Goal state, and
- Operator or legal moves.

Q. Explain problem as a state space with an example.

To explain problem as a state space representation in more detailed manner, let us consider a problem of 8-puzzle game. The puzzle consists of an 8-square frame and an empty slot. The tiles are numbered from 1-8. It is possible to move the tiles in the square field by moving tile into the empty slot. The objective is to get square in a numeric order.

- Initial State

1		2
4	5	3
7	8	6

- Operators

- UP: If the empty slot is not touching the up-frame, move it up.
- DOWN: If the empty slot is not touching down-frame, move it down.
- LEFT: If the empty slot is not touching left-frame, move it left.
- RIGHT: If the empty slot is not touching right-frame, move it right.

- Final State

1	2	3
4	5	6
7	8	

Planning	Problem Solving
The task of coming up with a sequence of actions that will achieve a goal is called planning	Problem Solving is the systematic search through a range of possible actions in order to reach some predefined goal or solution
Planning is a generic term of problem solving	Problem solving comprises standard search process
Planning is involved in plan generation	Problem solving is typically more concerned with plan execution
Eg: STRIPS	Eg: A* Search

Q. Vacuum World Problem

The world has only two *locations*

Each location may or may not contain *dirt*

The agent may be in one location or the other

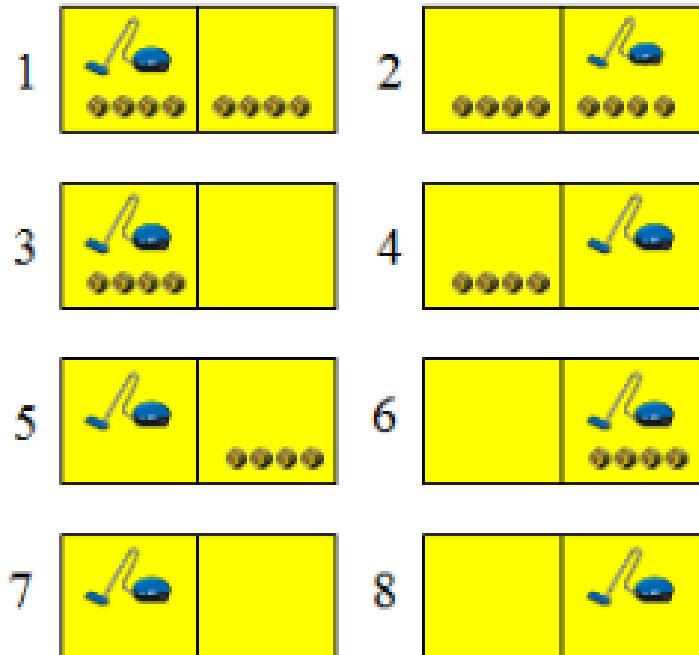
8 possible *world states*

Three possible actions: *Left, Right, Suck*

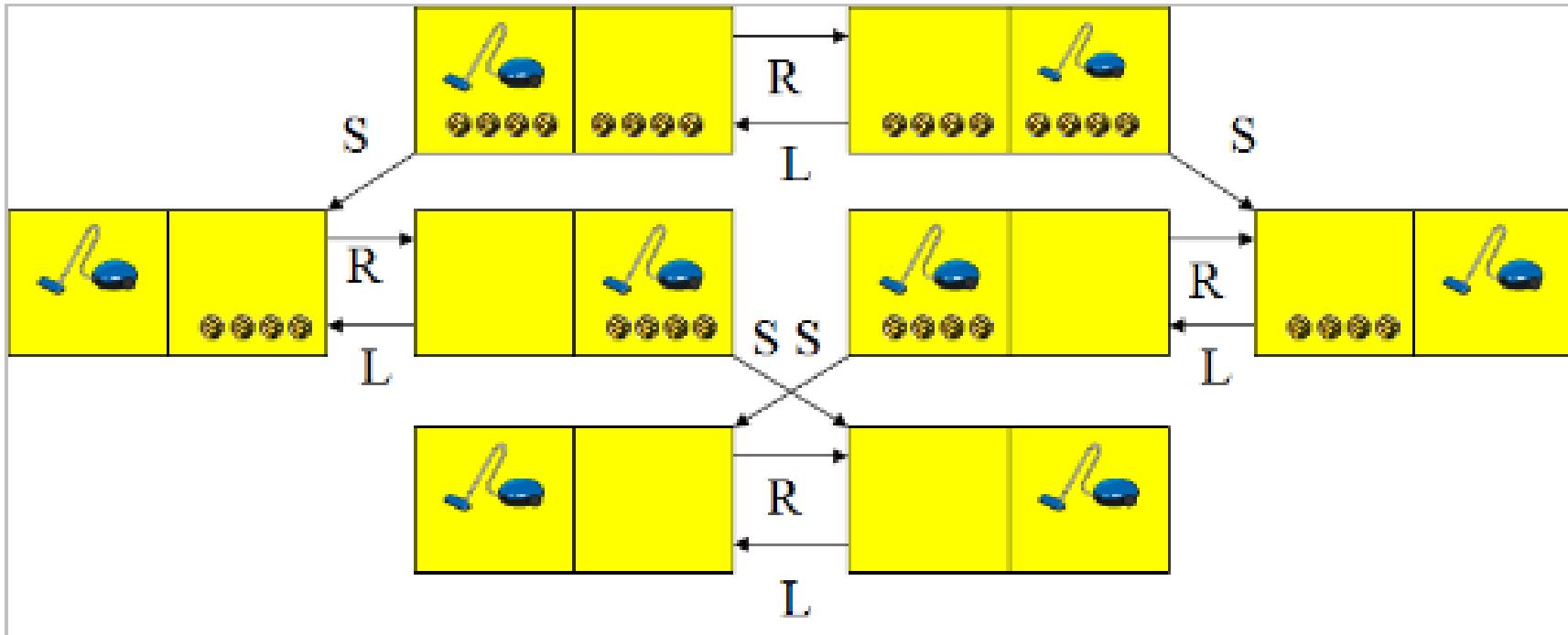
Goal: clean up all the dirt

Here,

- **Initial States:**
One of the 8 states
- **Operators:**
Move left, Move right, Suck
- **Final State:**
No dirt left in any square



- **Solution**



Q. You are given two jars of a six gallons and eight gallons capacity. There is no marking on the jars. There is a water tap, which can be used to fill jar. Your goal is to have exactly four gallons of water in eight gallons jar without taking any other jar or measuring device.

Q. River Crossing Problem

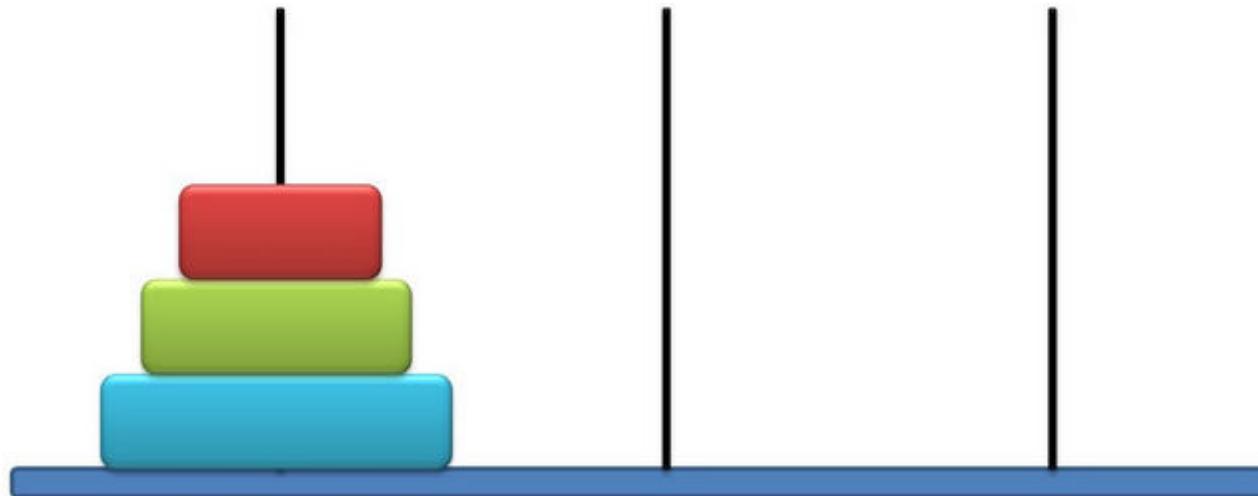
In a distinct land, bigamy is common. There are six people who want to cross a river in this land. This group consists of two men, each two wives. No man can tolerate any of his wives being in the company of another man unless yet least he or his next wife is present in the boat or in next land. There is a boat that holds two people to be used for crossing the river. How is the trip possible?

- **Initial State**
 $W \{ H_1, W_1, W_1' \text{ and } H_2, W_2, W_2' \}, E \{ \phi \}$
- **Operation**
 - No man can tolerate any of his wives being in the company of another man.
 - Crossing the river
- **Final State**
 $W \{ \phi \}, E \{ H_1, W_1, W_1' \text{ and } H_2, W_2, W_2' \}$

- **Solution**

S. N	West (W)	River	East (E)
0.	H1,W1,W1', H2,W2,W2'		(\emptyset)
1.	H1,H2,W2,W2'	W1, W1'	W1,W1'
2.	H1,W1,H2,W2,W2'	W1,	W1'
3.	H1,W1,H2,	W2, W2'	W1',W2,W2'
4.	H1,W1,H2,W2	W2	W1',W2'
5.	H1, H2,	W1, W2	W1,W1',W2,W2'
6.	H1, H2, W2	W2	W1,W1', W2'
7.	H1,	H2, W2	W1,W1', H2,W2, W2'
8.	H1,H2,	H2,	W2
9.	(\emptyset)	H1, H2,	H1, W1,W1', H2,W2, W2'

Q. Pegs and Disks problem

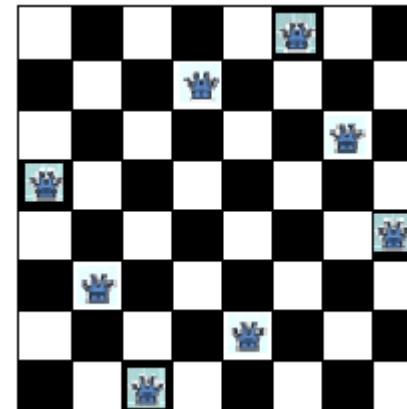
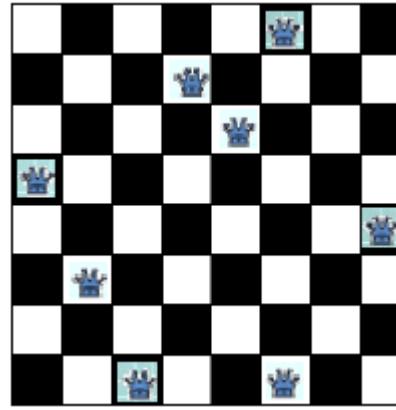


Q.8 queen's problem

The problem is to place 8 queens on a chessboard so that no two queens are in the same row, column or diagonal

Here,

- **Initial State**
- **Operation**
 - Add a queen in any square
- **Final State**



SEARCHING

- Searching is the process of finding the required states or nodes.
- Searching is to be performed through the state space.
- Search process is carried out by constructing a search tree.
- Search is a universal problem-solving technique.

SEARCH TERMINOLOGY

- **Problem Space:** Environment in which the search takes place.
- **Problem Instance:** It is Initial state + Goal state
- **Problem Space Graph:** It represents problem state. States are shown by nodes and operators are shown by edges.
- **Depth of a problem:** Length of a shortest path or shortest sequence of operators from Initial State to goal state.

- **Space Complexity:** The maximum number of nodes that are stored in memory.
- **Time Complexity:** The maximum number of nodes that are created.
- **Admissibility:** A property of an algorithm to always find an optimal solution.
- **Branching Factor:** The average number of child nodes in the problem space graph.

CLASSIFICATION

Uninformed Search (Blind Search/Brute force search)

The search algorithms that do not use any extra information regarding the problem

- ✓ Depth First Search
- ✓ Breath First Search
- ✓ Depth Limit Search
- ✓ Iterative deepening
- ✓ Uniform Cost
- ✓ Bidirectional Search

Informed search or Heuristic search

Informed search have problem specific knowledge apart from problem definition

- ✓ Hill climbing Search
- ✓ Best first Search
 - Greedy Best First Search
 - A* Search
- ✓ Simulated Annealing

DEPTH FIRST SEARCH (DFS)

- Proceeds down a single branch of the tree at a time
- Expands the root node, then the leftmost child of the root node
- Always expands a node at the deepest level of the tree
- Only when the search hits a dead end (a partial solution which can't be extended), the search backtrack and expand nodes at higher levels.

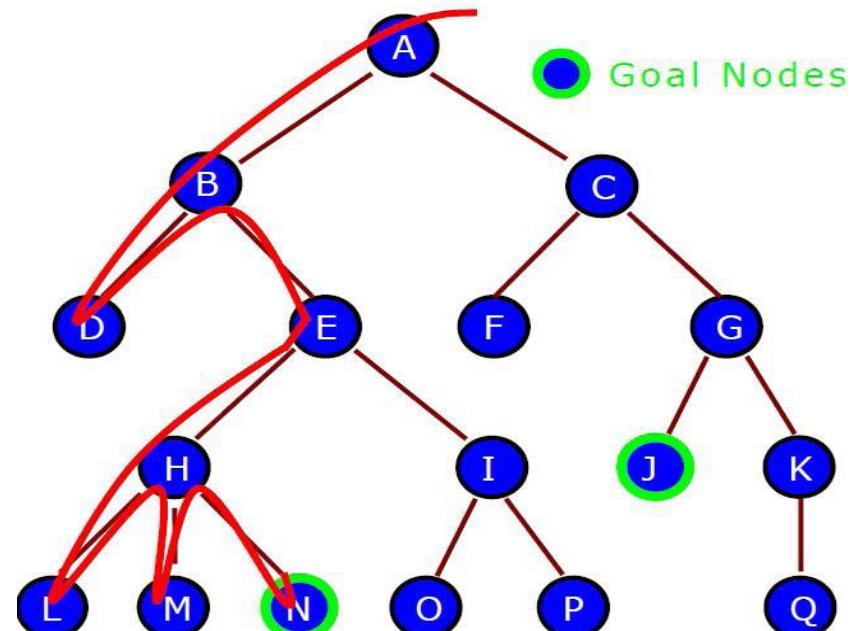


Fig. Depth-first search (DFS)

DEPTH FIRST SEARCH (DFS)

- ***Completeness:*** Incomplete as it may get stuck going down an infinite branch that doesn't leads to solution.
- ***Optimality:*** The first solution found by the DFS may not be shortest.
- ***Space complexity:*** b as branching factor and d as tree depth level, Space complexity = $O(b^{d+1})$
- ***Time Complexity:***
 $b + b^2 + b^3 + \dots + b^d + b^{d+1} = O(b^{d+1})$

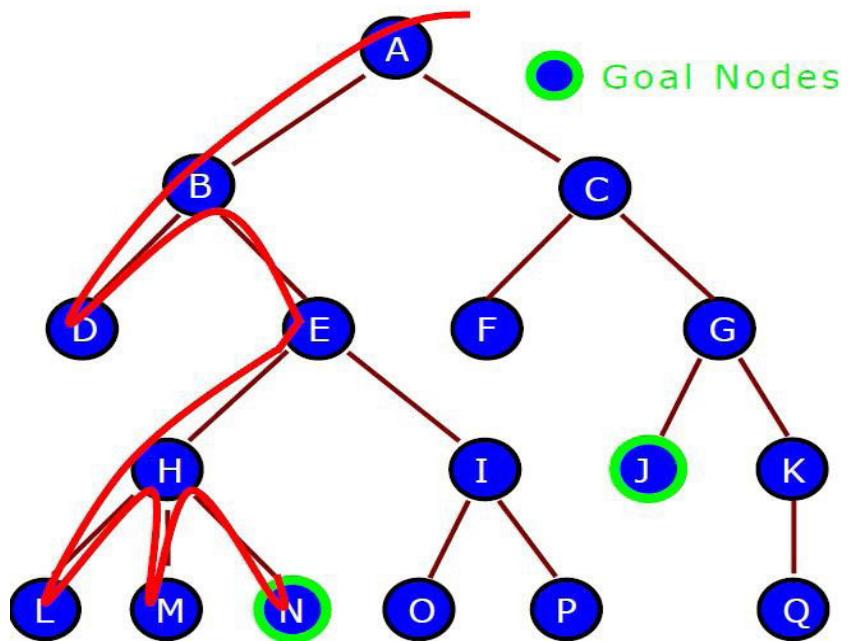
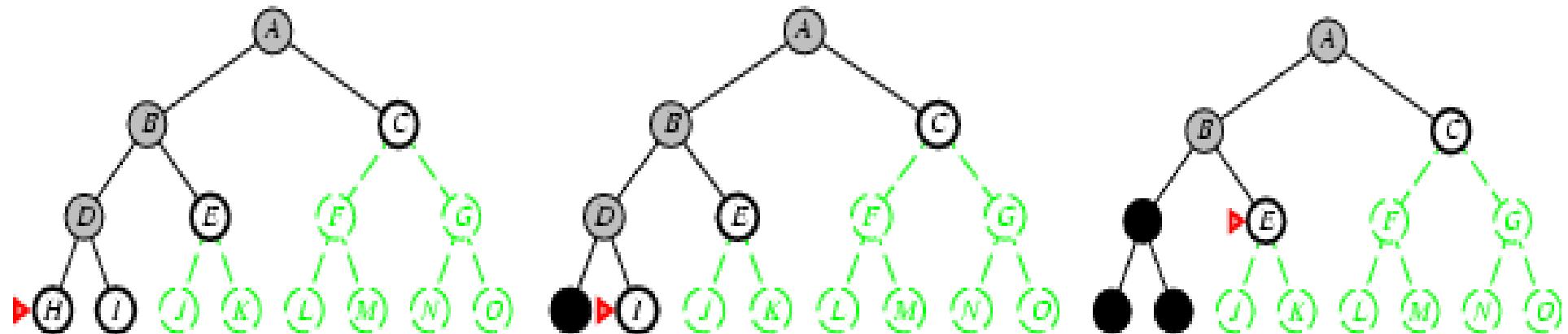
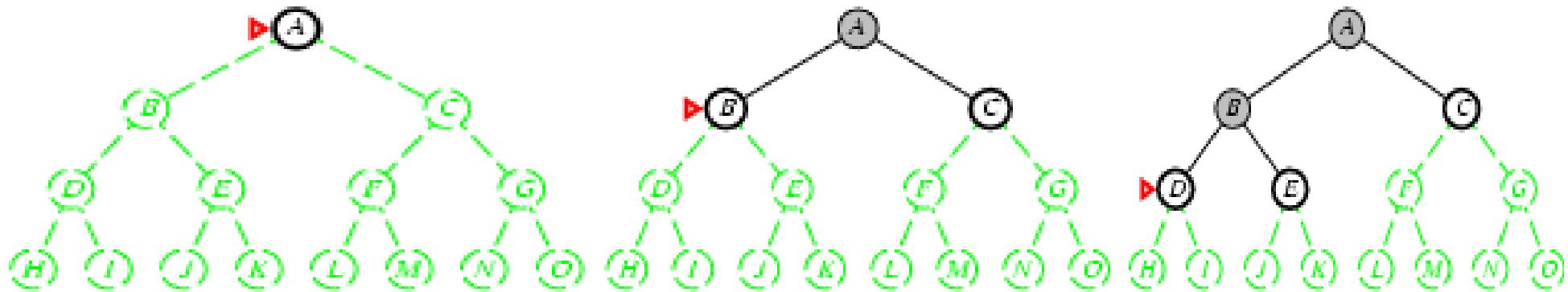


Fig. Depth-first search (DFS)

DFS example (find path from A to E)



Breadth-First Search (BFS)

- Proceeds level by level down the search tree
- Starting from the root node (initial state) explores all children of the root node, left to right
- If no solution is found, expands the first (leftmost) child of the root node, then expands the second node and so on

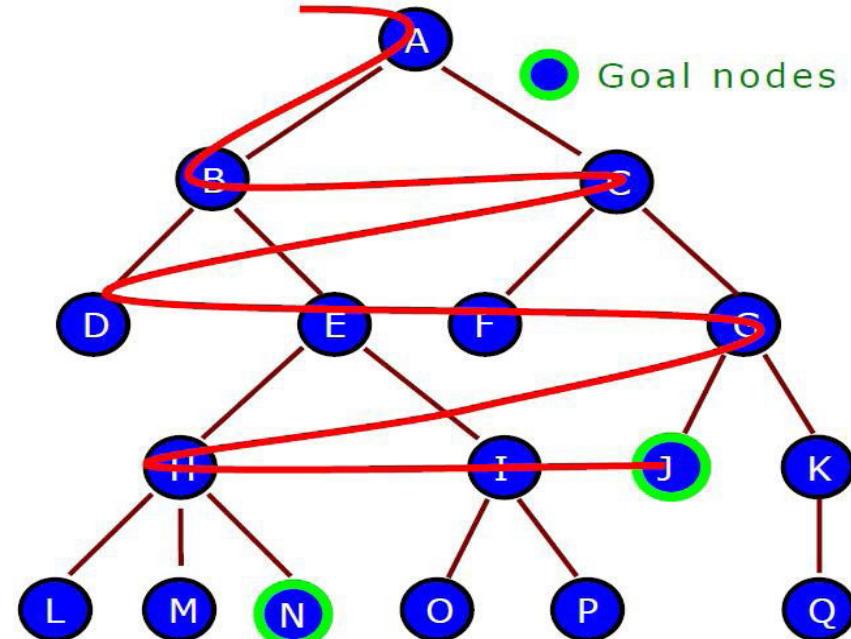


Fig. Breadth-first search (BFS)

Breadth-First Search (BFS)

❑ Completeness: Complete if the goal node is at finite depth

❑ Optimality: It is guaranteed to find the shortest path

❑ Time Complexity: $b + b^2 + b^3 + \dots + b^m = O(b^m)$

❑ Space Complexity: $(1 + b + b^2 + b^3 + \dots + b^m) \text{ times} = O(b^m)$

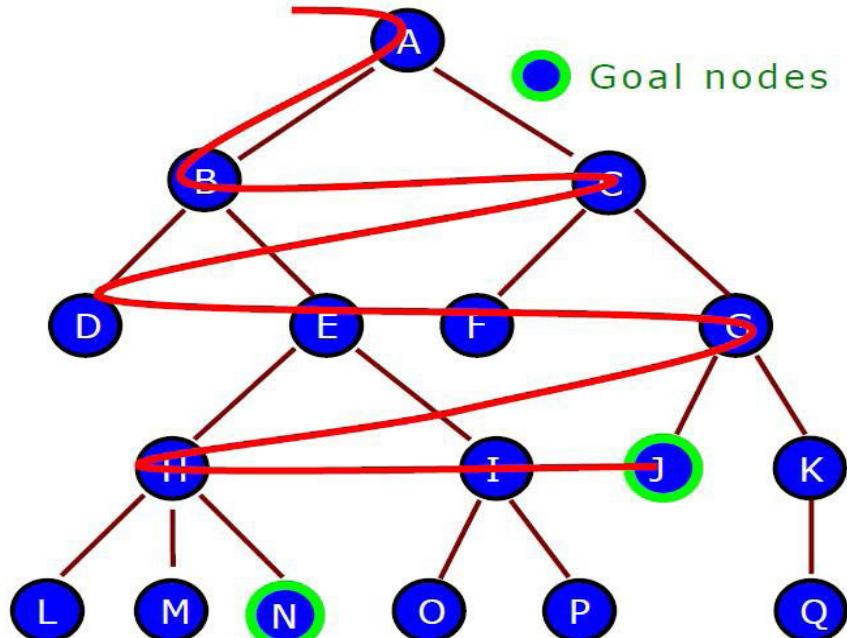
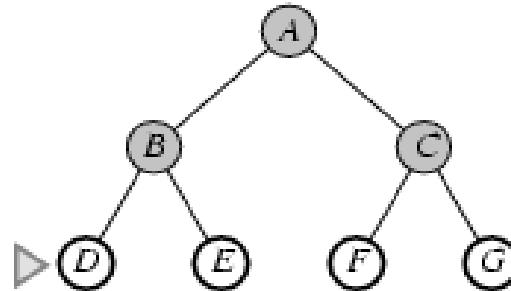
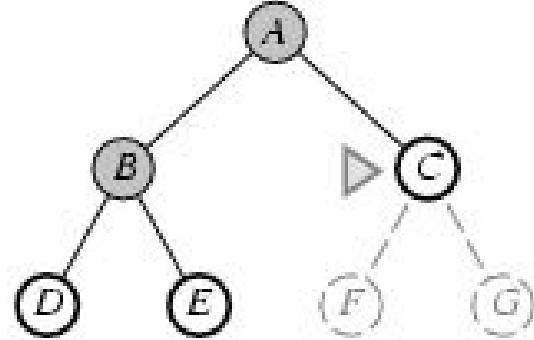
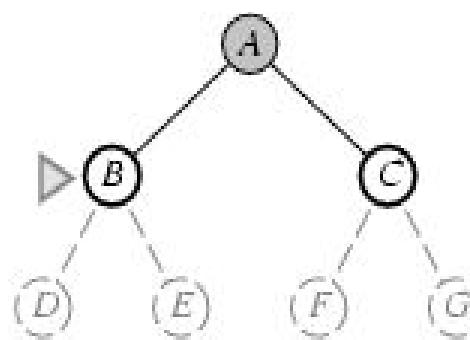
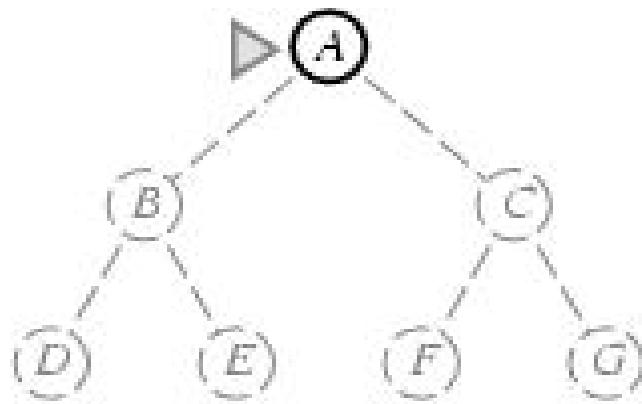


Fig. Breadth-first search (BFS)

BFS example (Find path from A to D)

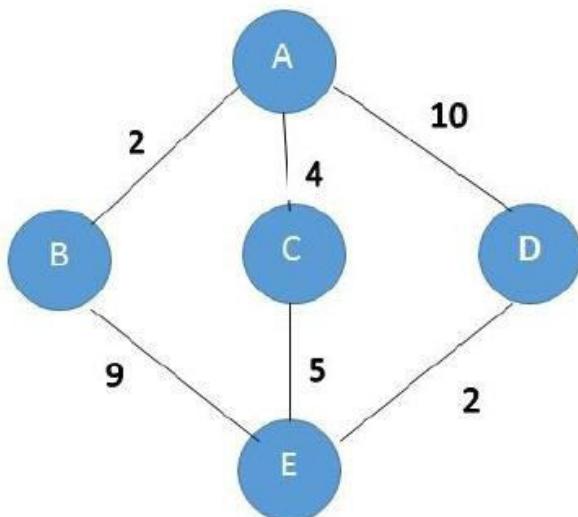


Uniform-Cost Search

Uniform-cost is guided by path cost rather than path length like in BFS, the algorithms starts by expanding the root, then expanding the node with the lowest cost from the root, the search continues in this manner for all nodes.

- **Completeness:**
Complete if the cost of each step exceeds some small positive integer, this to prevent infinite loops.
- **Optimality:**
Optimal in the sense that the node that it always expands is the node with the least path cost.
- **Time Complexity:** $O(b^{C/e})$.
- **Space Complexity:** $O(b^{C/e})$

UCS example (Find path from A to E)



- Expand A to B, C, and D.
 - The path to B is the cheapest one with path cost 2.
 - Expand B to E
 - Total path cost = $2+9 = 11$
 - This might not be the optimal solution since the path AC as path cost 4 (less than 11)
-
- Expand C to E
 - Total path cost = $4+5 = 9$
 - Path cost from A to D is 10 (greater than path cost, 9)
 - Hence optimal path is ACE.

Depth Limit Search

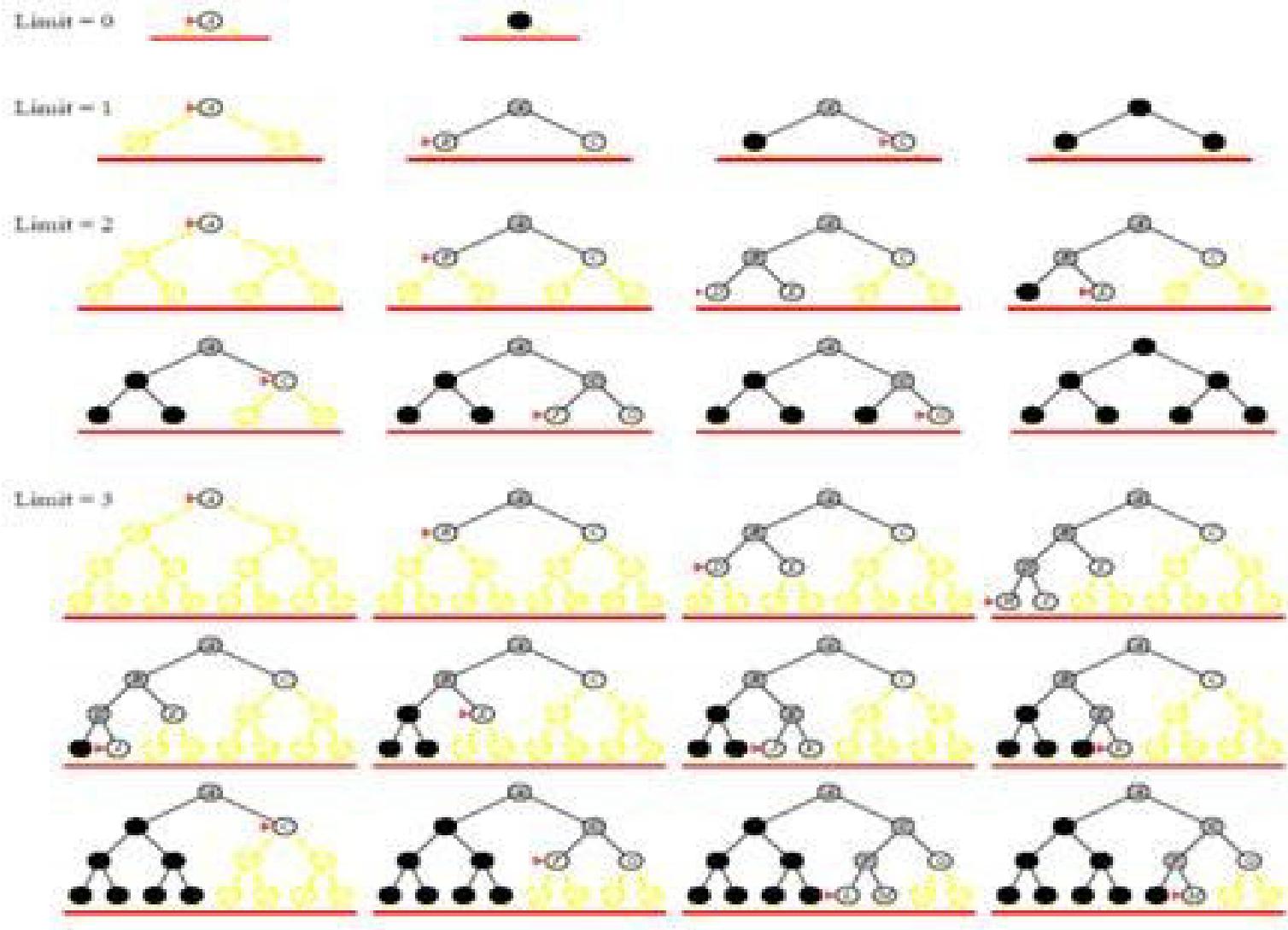
- Depth-first search will not find a goal if it searches down a path that has infinite length. So, in general, depth-first search is not guaranteed to find a solution, so it is not complete.
- This problem is eliminated by limiting the depth of the search to some value L . However, this introduces another way of preventing depth-first search from finding the goal: if the goal is deeper than L it will not be found.
- Perform depth first search but only to a pre-specified depth limit L .
- No node on a path that is more than L steps from the initial state

Depth Limit Search

- ***Completeness:*** Incomplete as solution may be beyond specified depth level.
- ***Optimality:*** not optimal
- ***Space complexity:*** b as branching factor and L as tree depth level, $O(b \cdot L)$
- ***Time Complexity:*** $O(bL)$

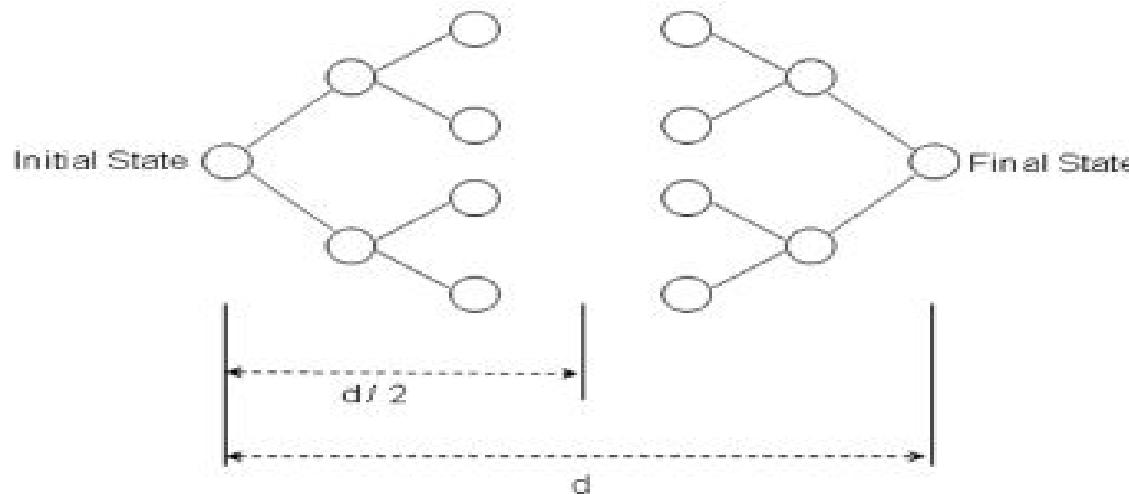
Iterative Deepening Search (IDS)

- Iterative deepening search is a strategy that sidesteps the issue of **choosing the best depth limit by trying all possible depth limits**: first depth 0, then depth 1, then depth 2, and so on.
- In effect, iterative deepening **combines the benefits of depth-first and breadth-first search**.
- It is optimal and complete, like breadth-first search, but has only the modest memory requirements of depth-first search.



Bidirectional Search

- As the name suggests, bidirectional search suggests to run 2 simultaneous searches
- One from the initial state and the other from the Final state
- Those 2 searches stop when they meet each other at some point in the middle of the graph.



Bidirectional Search

- ❑ Completeness:

- Bidirectional search is complete when we use BFS in both searches

- ❑ Optimality:

- Like the completeness, bidirectional search is optimal when BFS is used.

- ❑ Time/Space Complexity : $O(b^{d/2})$

Informed Search (Heuristic Search)

- Informed search have problem specific knowledge apart from problem definition.
- They use experimental algorithm which improves efficiency of search process.
- The idea is to develop a domain specific heuristic function $h(n)$ where $h(n)$ guesses the cost of getting to the goal from node n .

Heuristic Function

The heuristic function is a way to inform the search about the direction to a goal. It provides an informed way to guess which neighbor of a node will lead to a goal.

Best-First Search (BFS)

- Best-First search is a graph-based heuristic search algorithm
- The name “best-first” refers to the method of exploring the node with the best “score” first.
- An evaluation function is used to assign a score to each candidate node. The evaluation function must represent some estimate of the cost of the path from state to the closest goal state

Algorithm

1. Put the initial node on a list START
2. If $\text{START} = \text{GOAL}$ or $\text{START} = \text{EMPTY}$, then terminate search
3. Assign the next node as START and call this node-A
4. If $A = \text{GOAL}$, terminate the search with success
5. Else-if, node has successor and generate all of them. Find out how far they are from the GOAL node.
6. Sort all the children generated so far by remaining distance from the goal. Name the list as START-1. Replace $\text{START} = \text{START-1}$
7. Go to step-2

Types of BFS

- Greedy Best First Search
- A* Search

Greedy Best First Search

- It tries to get as close as it can to the goal.
- It expands the node that appears to be closest to the goal
- It evaluates the node by using heuristic function only.
- Evaluation function $f(n) = h(n)$
 - heuristic= (estimate of cost from n to goal)
 - $h(n) = 0$ for goal state

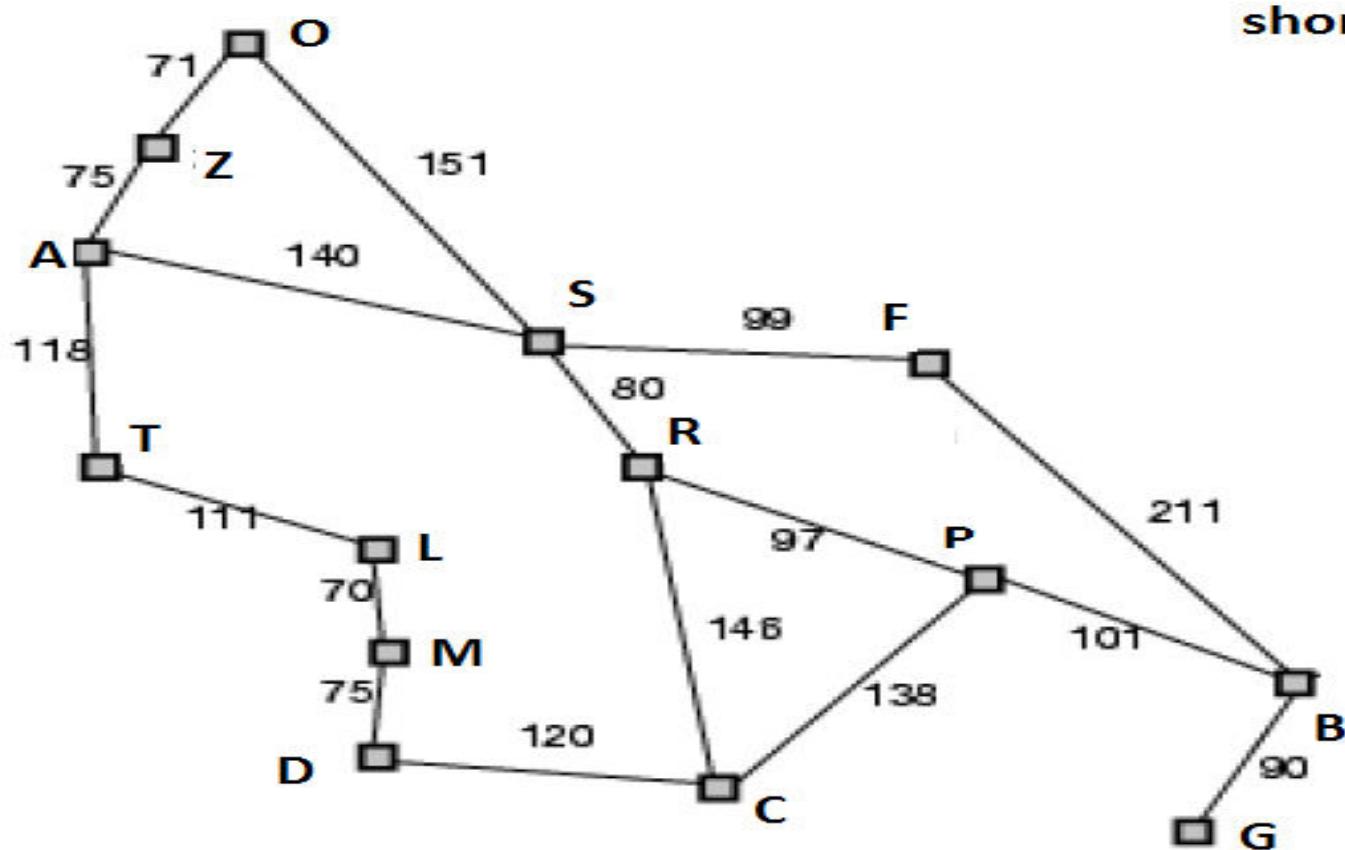
Properties

- ❑ **Completeness:** No – can get stuck in loops
- ❑ **Time Complexity:** $O(b^m)$, but a good heuristic can give dramatic improvement
- ❑ **Space Complexity:** $O(b^m)$, keeps all nodes in memory
- ❑ **Optimality:** No

Applications:

This algorithm is used in Huffman encoding, minimum spanning tree, Dijkstra's algorithm etc.

Given following graph of cities, starting at City “A”, problem is to reach to the “B”

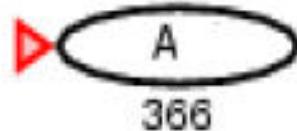


shortest line distance to B

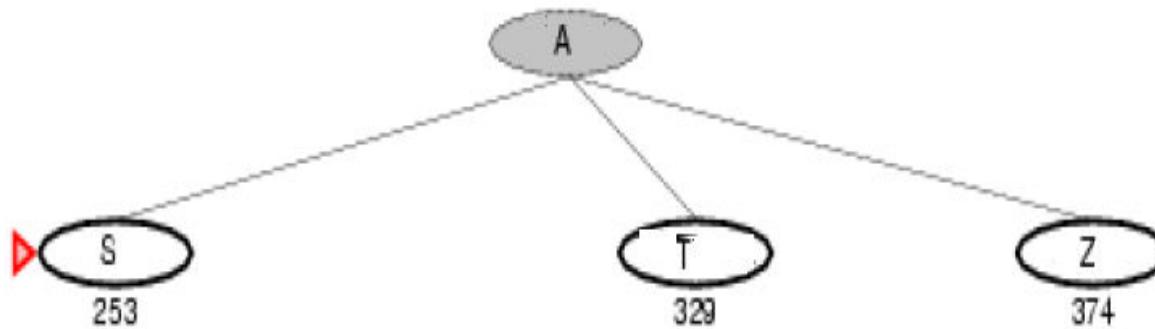
A	366
B	0
C	160
D	242
F	176
G	77
L	244
M	241
O	380
P	101
R	193
S	253
T	329
Z	374

Solution using greedy best first can be as below:

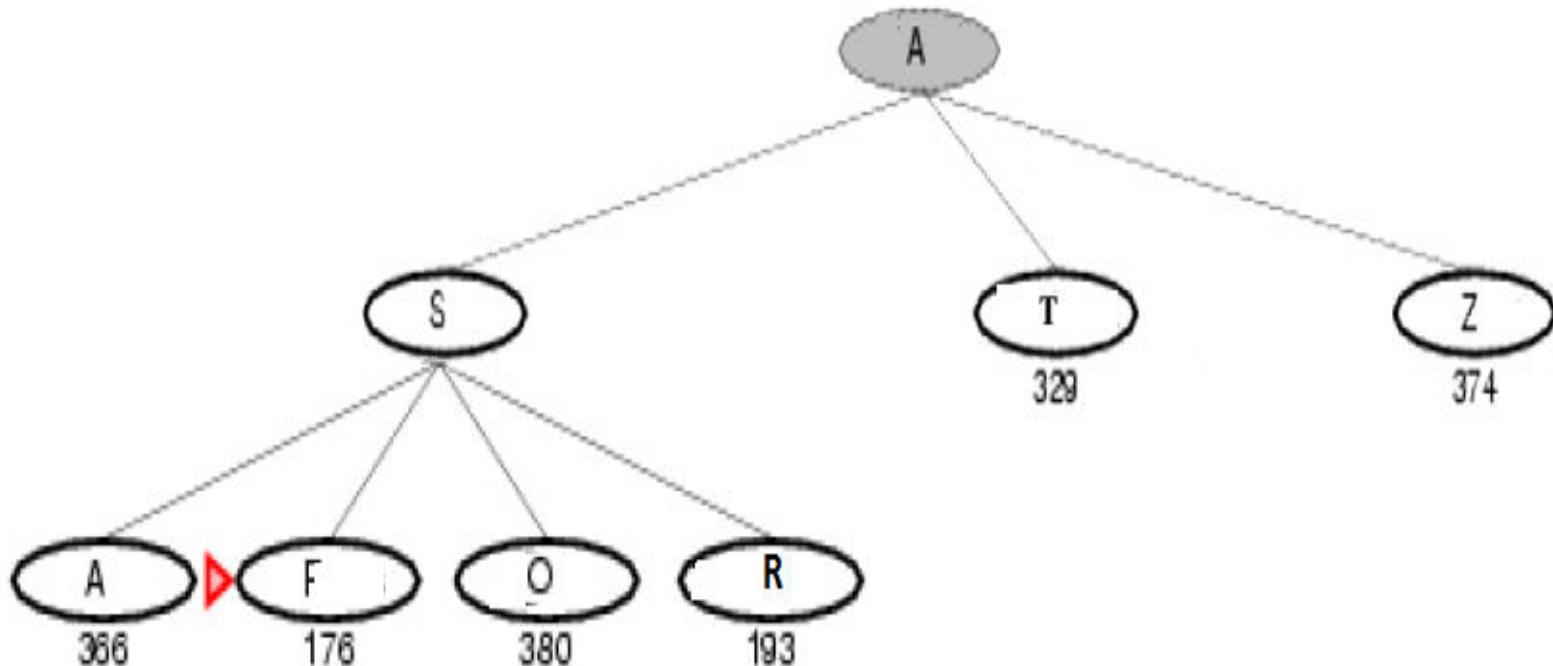
Step 1: Initial State



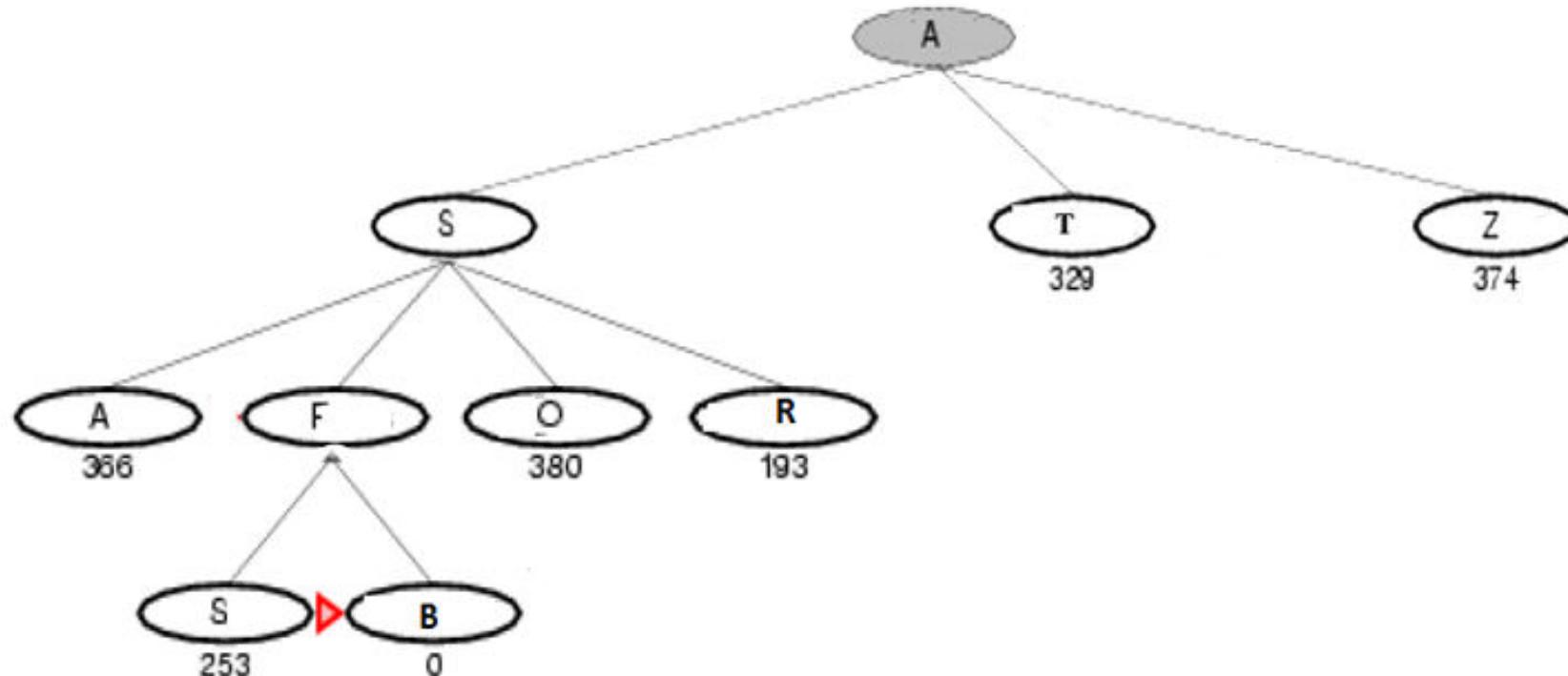
Step 2: After expanding A



Step 3: After expanding S



Step 3: After expanding F



A* Search

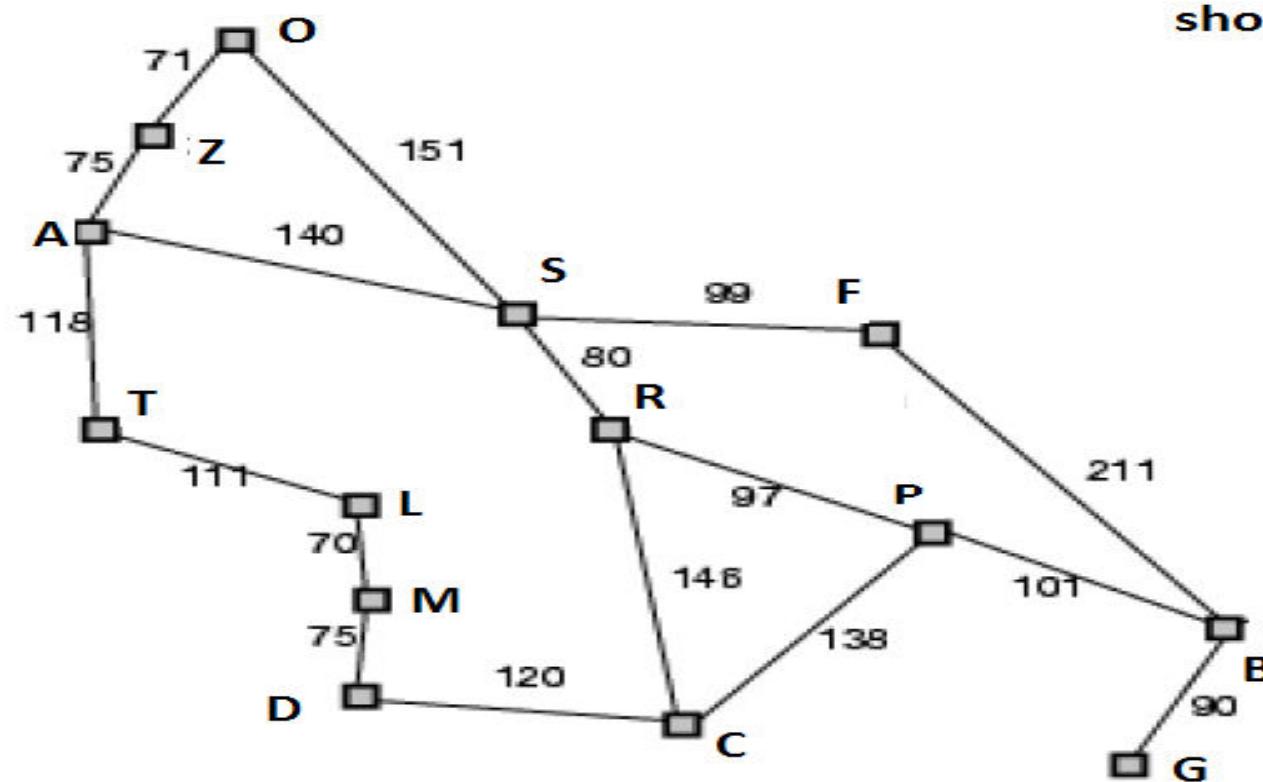
- It finds a minimal cost-path joining the start node and a goal node for node n.
- Evaluation function: $f(n) = g(n) + h(n)$

Where,

 - $g(n)$ = cost so far to reach n from root
 - $h(n)$ = estimated cost to goal from n

Thus, $f(n)$ estimates always the lowest total cost of any solution path going through node n.
- Avoid expanding paths that are already expensive
- The main drawback of A* algorithm and indeed of any best-first search is its memory requirement.

A* search example (Find path from A to B)



shortest line distance to B

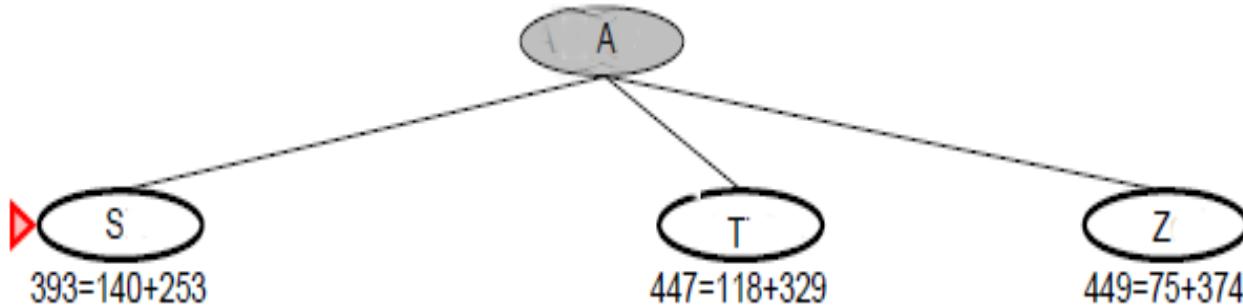
A	366
B	0
C	160
D	242
F	176
G	77
L	244
M	241
O	380
P	101
R	193
S	253
T	329
Z	374

Here, evaluate nodes connected to source. Evaluate $f(n) = g(n) + h(n)$ for each node. Select node with lowest $f(n)$ value.

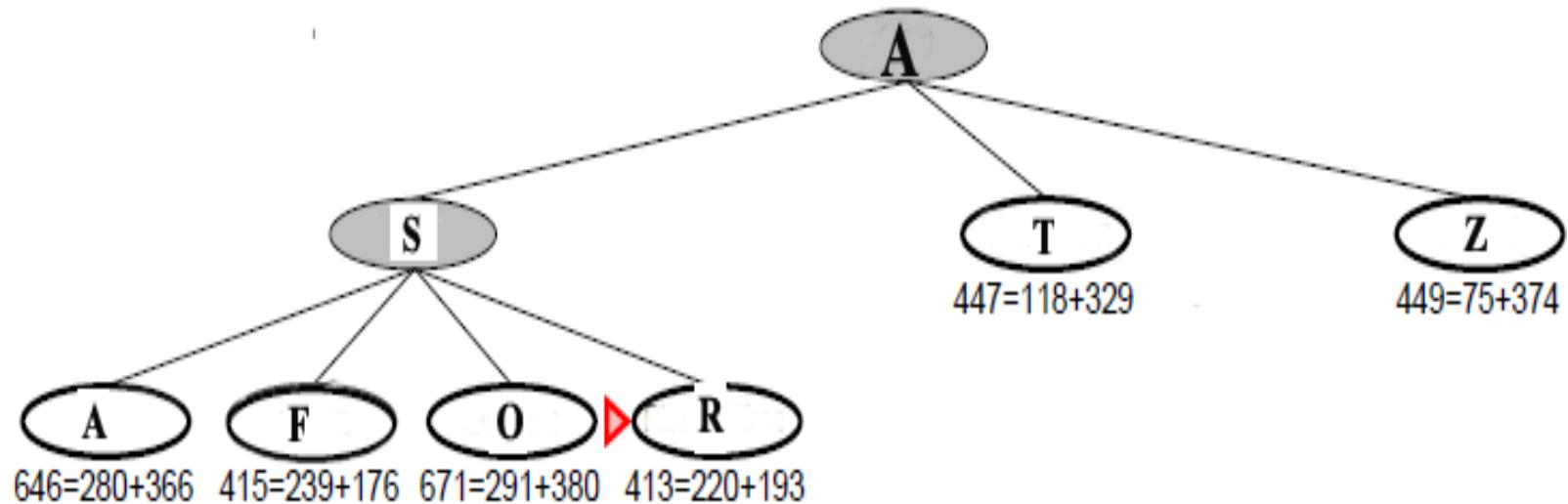
Step 1:



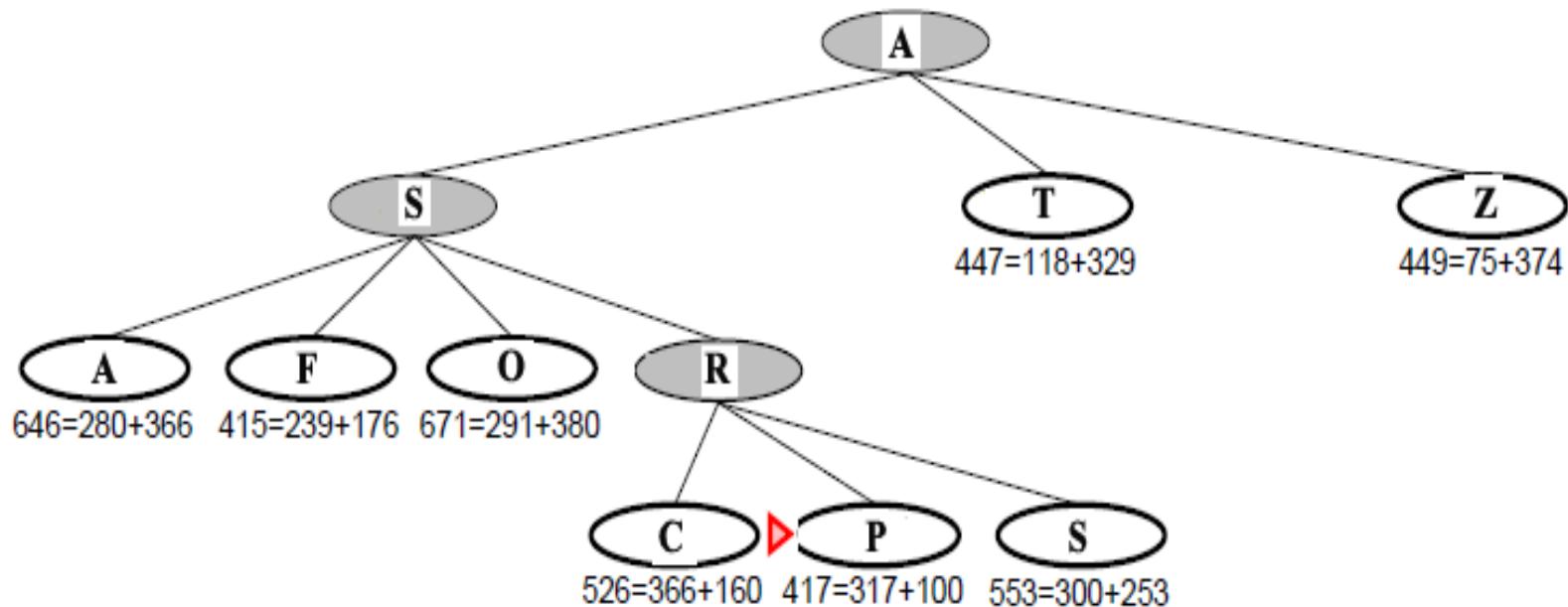
Step 2



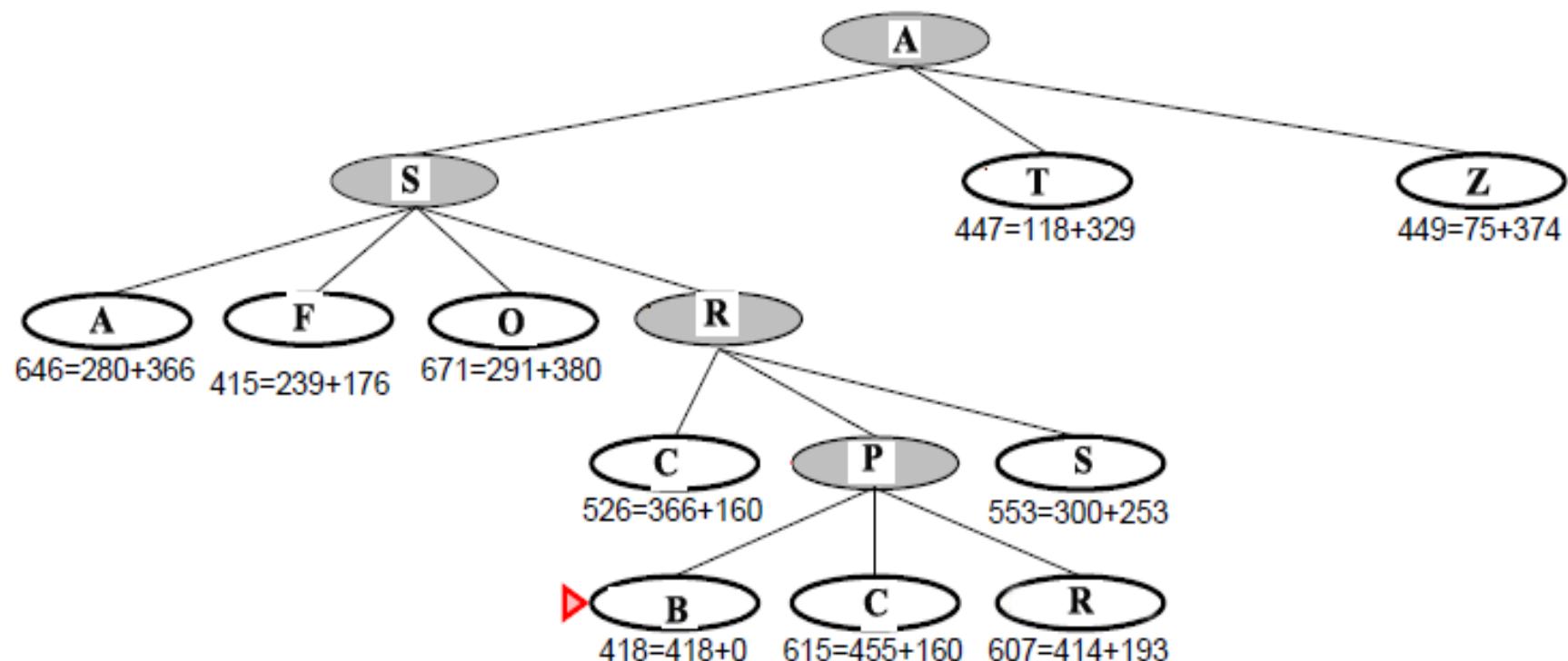
Step 3:



Step 4:

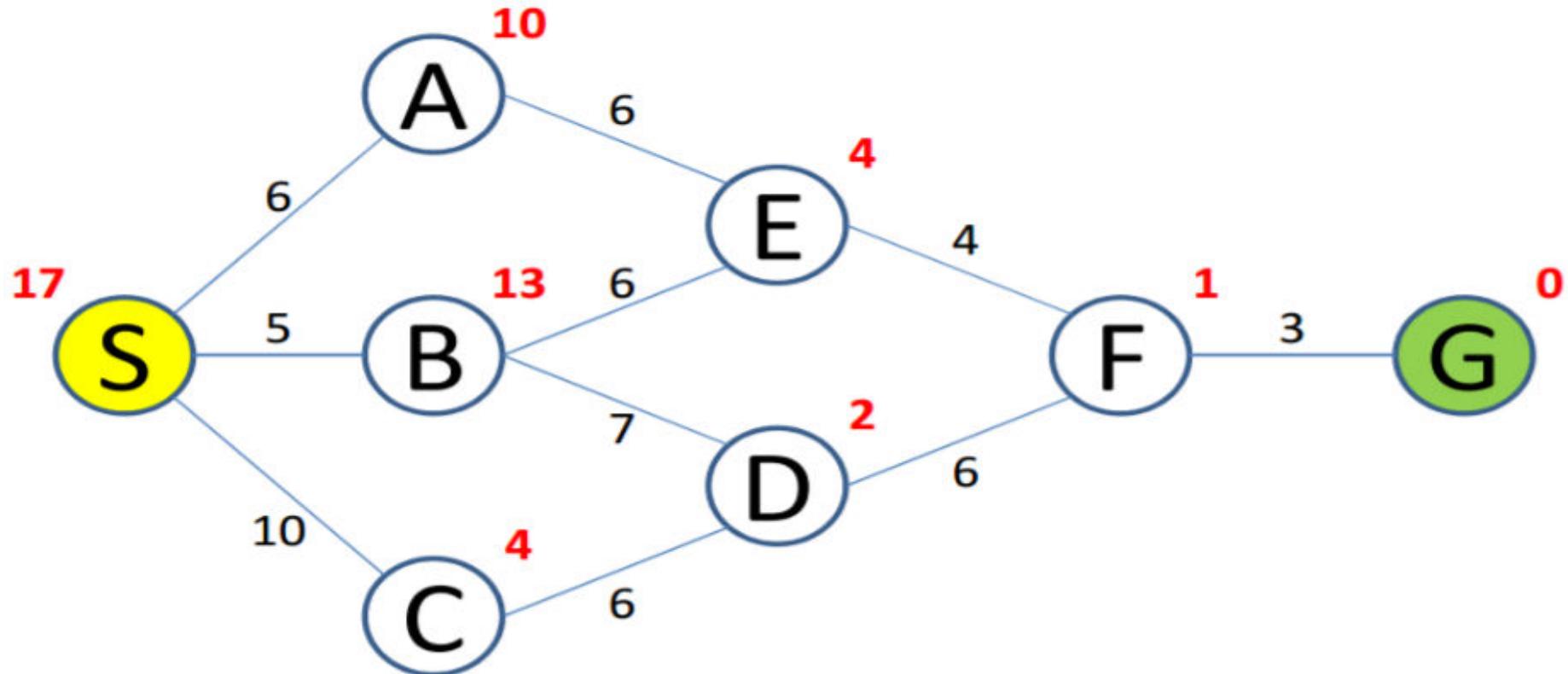


Step 5:



Assignment

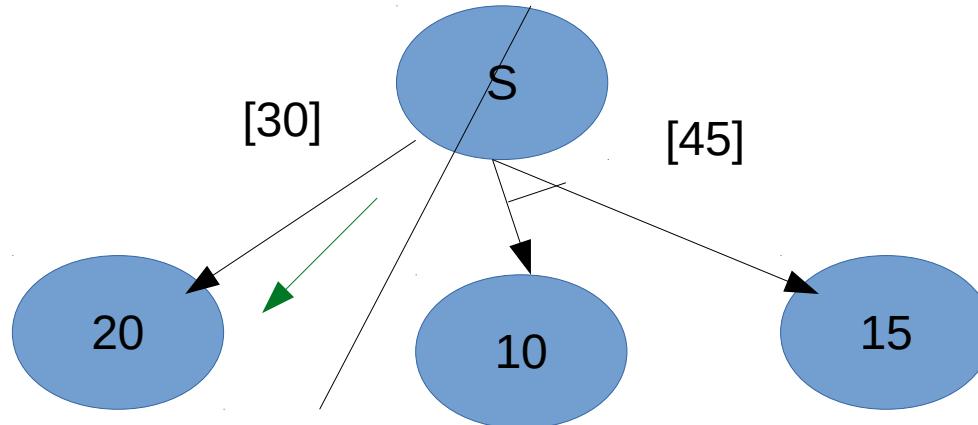
Perform A* Search



AO* Searching

- AO* Search is a type of heuristic search algorithm .
- AO* Search is used when problems can be divided into sub parts and which can be combined
- AO* in artificial intelligence is represented using AND OR graph or AND OR tree
- AO* have one or more and arc in it

Edge = 10



Hill Climbing Search

- Hill climbing is an extension of depth-first search which uses some knowledge such as estimates of the distance of each node from the goal to improve the search
- It is simply a loop that continually moves in the direction of increasing value—that is, uphill. It terminates when it reaches a “peak” where no neighbor has a higher value.
- Hill climbing is sometimes called greedy local search because it grabs a good neighbor state without thinking ahead about where to go next

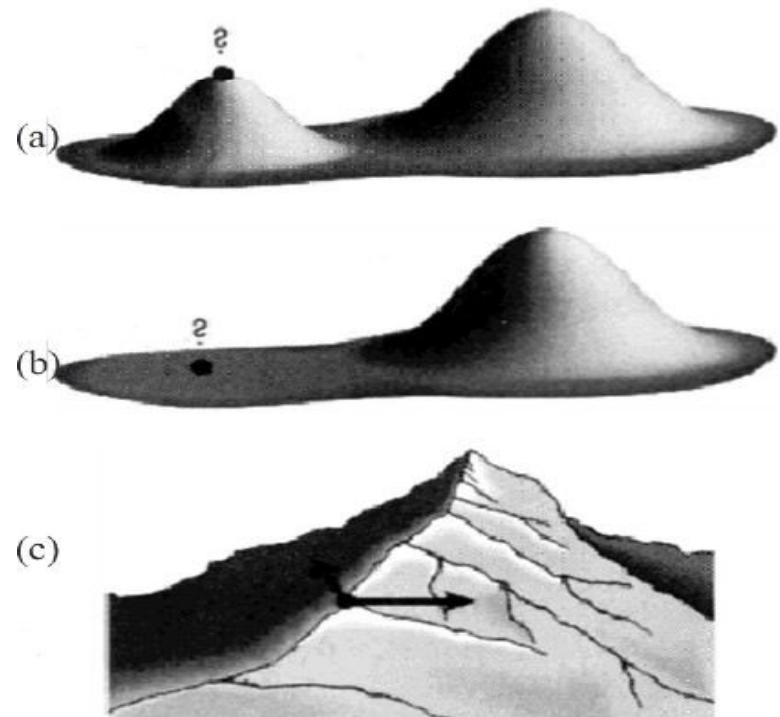


Figure 5.9 Local maxima, Plateaus and ridge situation for Hill Climbing

Hill Climbing Search

➤ Drawbacks

❑ Local Maxima

A local maximum is a peak that is higher than each of its neighboring states but lower than the global maximum.

❑ Plateaus: A plateau is an area of the search space where evaluation function is flat, thus requiring random walk

❑ Ridges: Where there are steep slopes and the search direction is not towards the top but towards the side

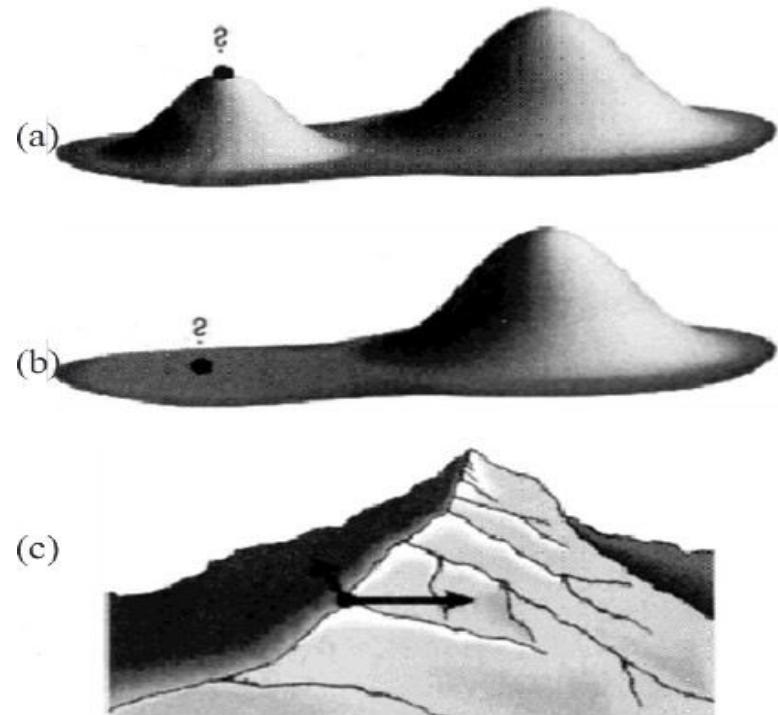


Figure 5.9 Local maxima, Plateaus and ridge situation for Hill Climbing

Hill Climbing Search

➤ Remedies

Back tracking for local maximum:

The back tracking help in undoing what is been done so far and permit to try totally different part to attain the global peak.

Big Jump:

A big jump is the solution to escape from plateaus because all neighbors' points have same value using the greedy approach

Random restart:

Keep restarting the search from random locations until a goal is found

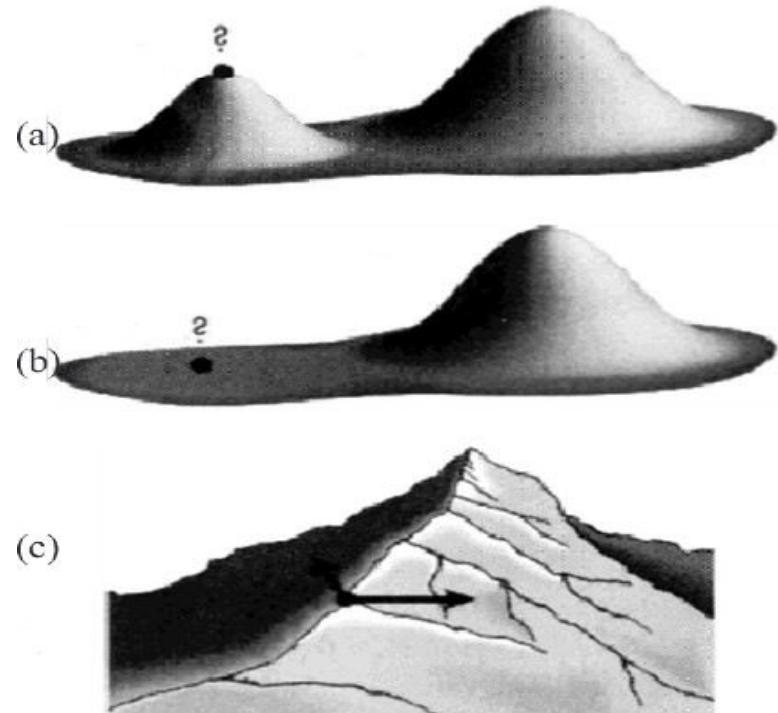
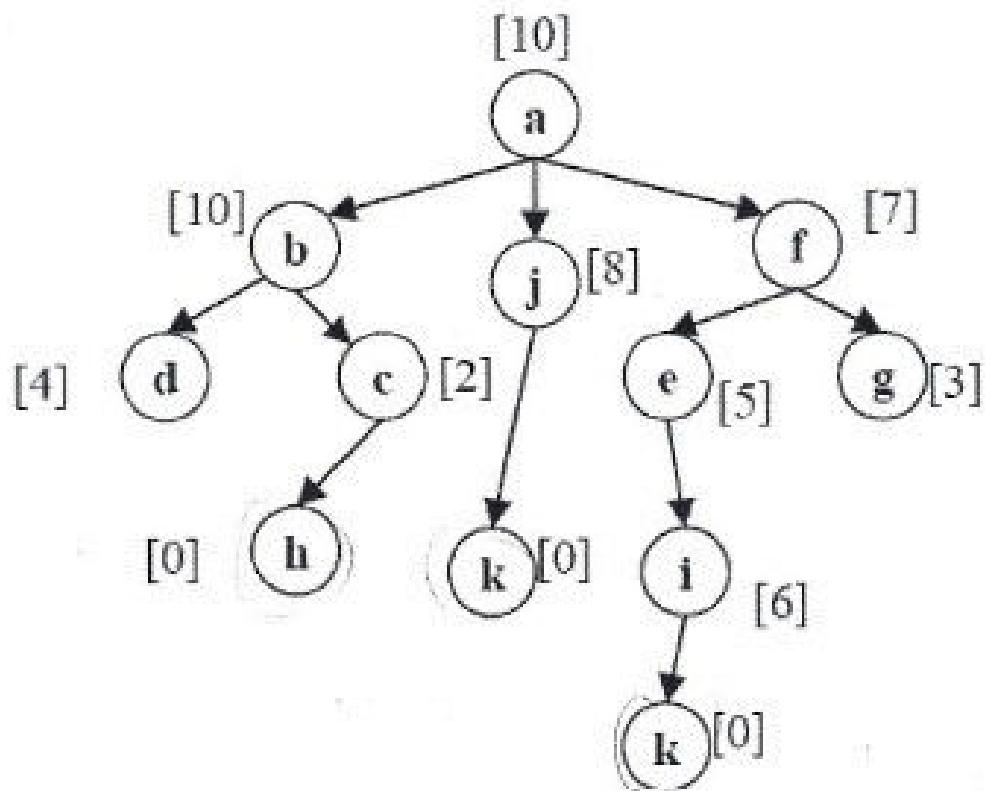


Figure 5.9 Local maxima, Plateaus and ridge situation for Hill Climbing

Why Hill Climbing is not Complete?

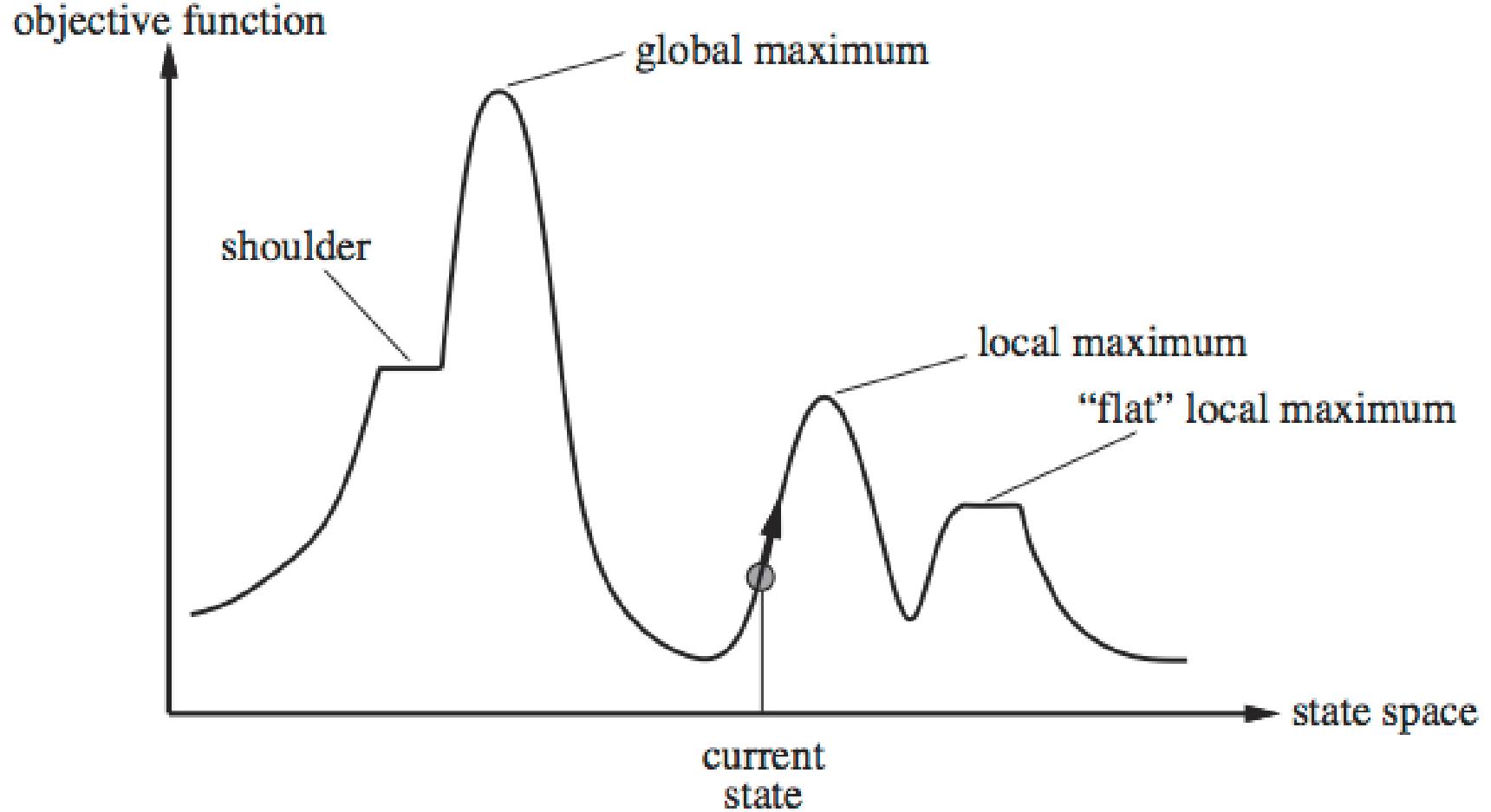
- › Hill-climbing always attempts to make changes that improve the current state.
- › The main problem that hill climbing can encounter is that of local maxima. This occurs when the algorithm stops making progress towards an optimal solution; mainly due to the lack of immediate improvement in adjacent states.

Problems in Hill Climbing



Here, "a" is initial and h and k are final states

- › We start a-> f-> g and then what ??finish(without result)
- › A common way to avoid getting stuck in local maxima with Hill Climbing is to use random restarts. In your example if G is a local maxima, the algorithm would stop there and then pick another random node to restart from. So if J or C were picked (or possibly A, B, or D), this would find the global maxima in H or K



Simulated Annealing

- Simulated Annealing escapes local maxima by allowing some "bad" moves but gradually decrease their frequency.
- Instead of restarting from a random point, we allow the search to take some downhill steps to try to escape local maxima

Means Ends Analysis

- Means ends analysis is the problem solving techniques used commonly in Artificial intelligence for limiting search in AI program.
- The MEA technique as a problem-solving strategy was first introduced in 1961 by Allen Newell and Herbert A. Simon in their computer problem-solving program General Problem Solver (GPS).In that implementation, the correspondence between differences and actions, also called operators, is provided a priori as knowledge in the system.

- How MEA works?

The MEA technique is a strategy to control search in problem-solving. Given a current state and a goal state, an action is chosen which will reduce the difference between the two. The action is performed on the current state to produce a new state, and the process is recursively applied to this new state and the goal state. Note that, in order for MEA to be effective, the goal-seeking system must have a means of associating to any kind of detectable difference those actions that are relevant to reducing that difference. It must also have means for detecting the progress it is making (the changes in the differences between the actual and the desired state), as some attempted sequences of actions may fail and, hence, some alternate sequences may be tried. When knowledge is available concerning the importance of differences, the most important difference is selected first to further improve the average performance of MEA over other brute-force search strategies. However, even without the ordering of differences according to importance, MEA improves over other search heuristics (again in the average case) by focusing the problem solving on the actual differences between the current state and that of the goal

Forward chaining

- Forward chaining is one of the two main methods of reasoning when using inference rules
- An inference engine, using forward chaining, searches the inference rules until it finds one where the antecedent (If clause) is known to be true. When such a rule is found, the engine can conclude, or infer, the consequent (Then clause), resulting in the addition of new information to its data.
- Forward chaining is a popular implementation strategy for expert systems, business and production rule systems.
- Forward chaining starts with the available data and uses inference rules to extract more data until a goal is reached.

Idea: fire any rule whose premises are satisfied in the *KB*,
– add its conclusion to the *KB*, until query is found

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

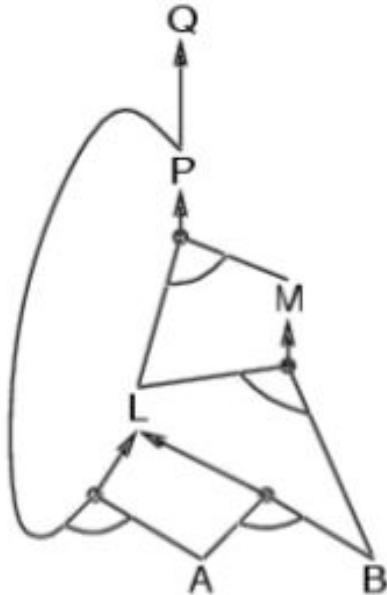
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

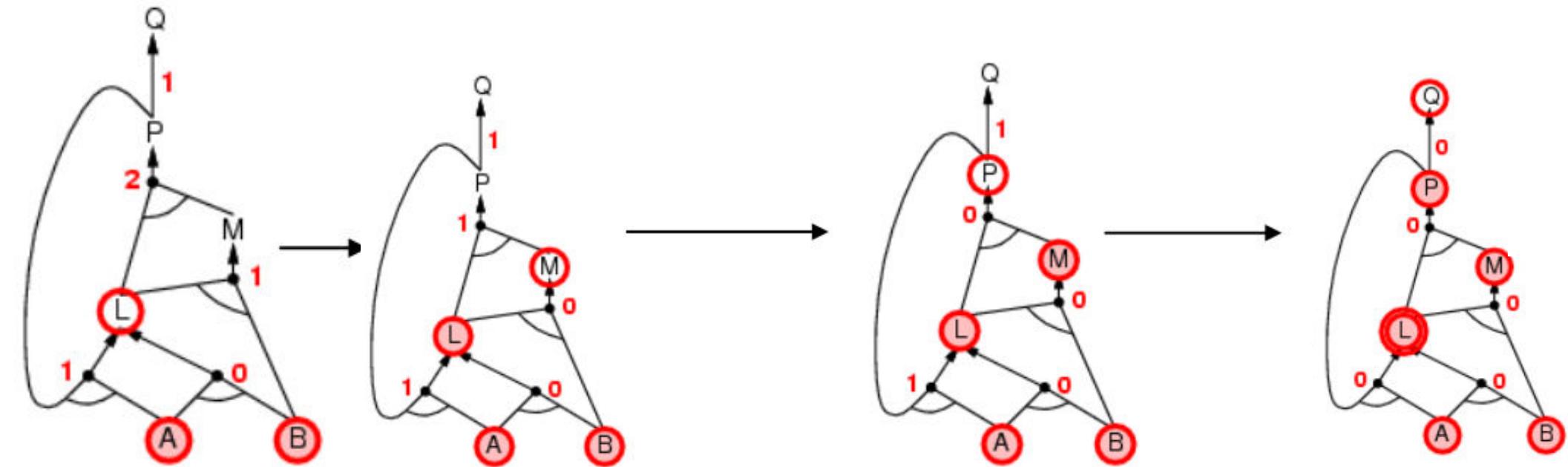
$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$



Prove that *Q* can be inferred from above *KB*



Backward chaining

- Backward chaining (or backward reasoning) is an inference method that can be described as working backward from the goal(s).
- In game theory, its application to sub games in order to find a solution to the game is called backward induction.

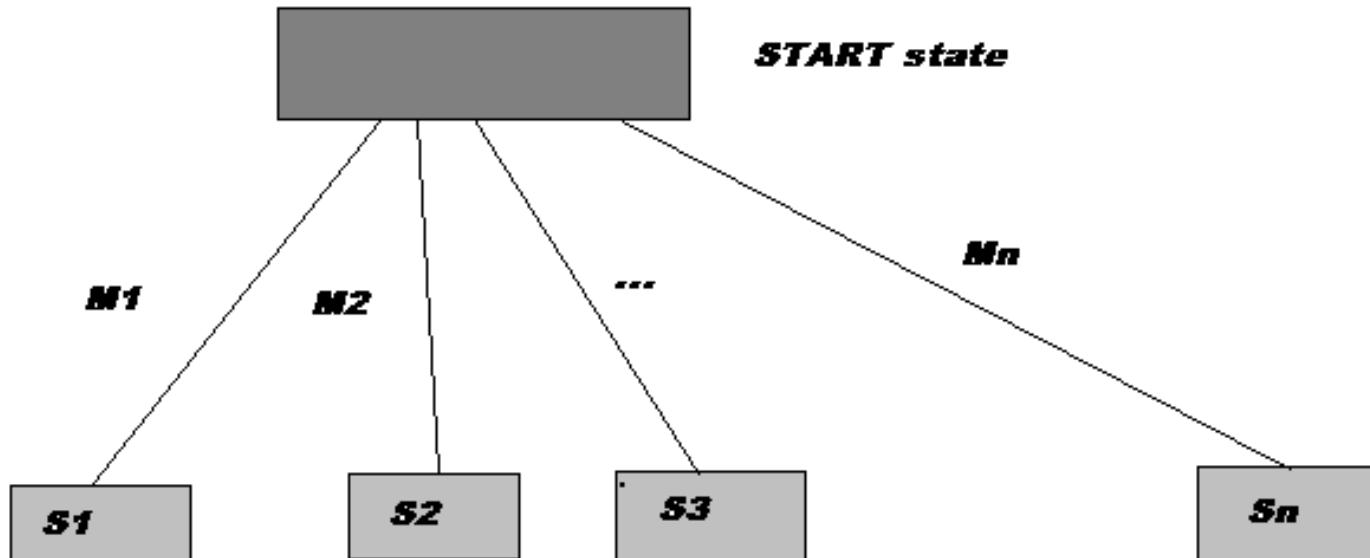
Game Playing

A game can be formally defined as a kind of search problem as below:

- › **Initial state:** It includes the board position and identifies the player's to move.
- › **Successor function:** It gives a list of (move, state) pairs each indicating a legal move and resulting state.
- › **Terminal test:** This determines when the game is over. States where the game is ended are called terminal states.
- › **Utility function:** It gives numerical value of terminal states. E.g. win (+1), loose (-1) and draw (0).

Game Trees

- › We can represent all possible games(of a given type) by a directed graph often called a game tree.
- › The nodes of the graph represent the states of the game. The arcs of the graph represent possible moves by the players (+ and -)



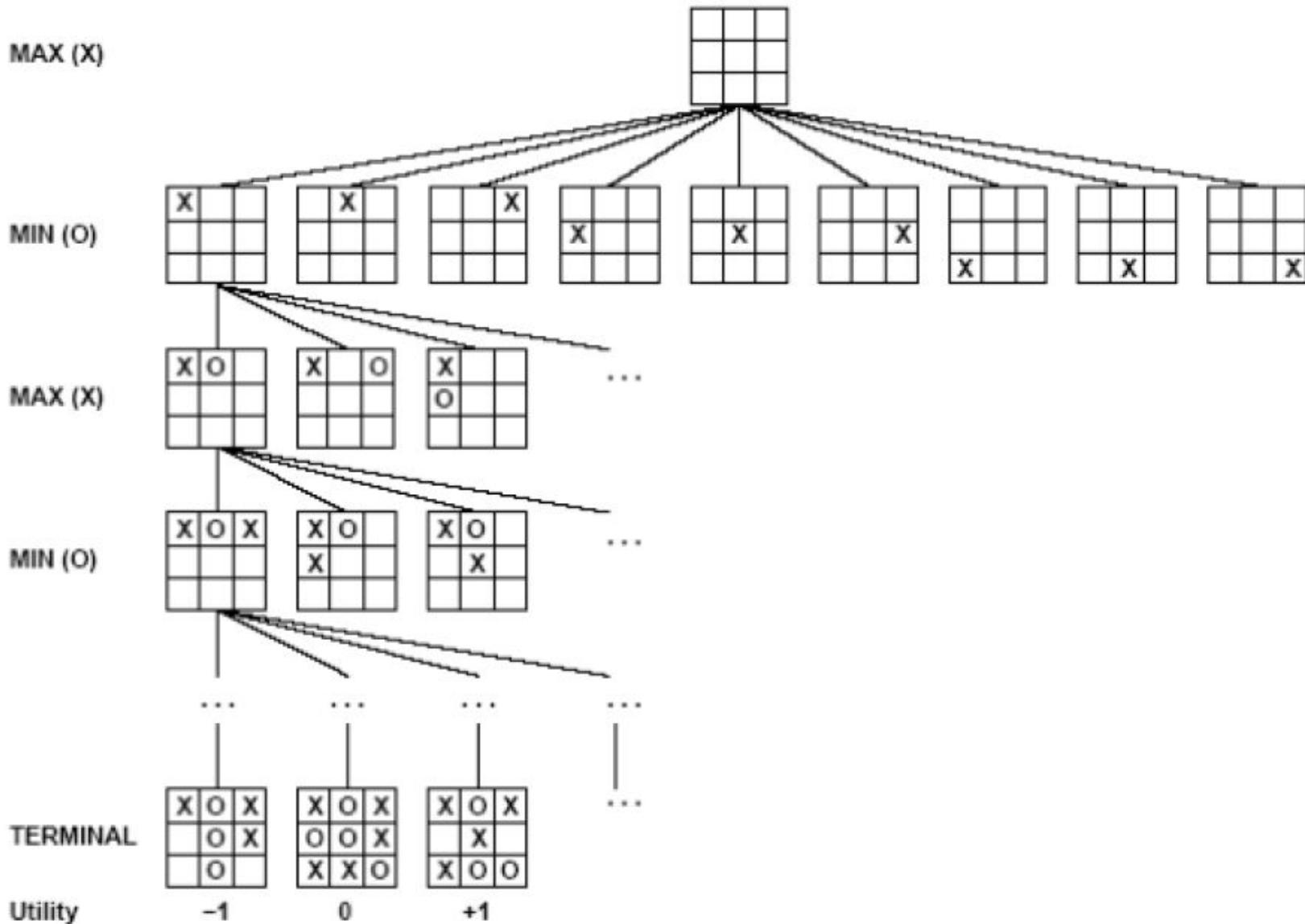
Example: Tic-tac-toe

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 the one who succeeds in writing three letters in line.

The game begins with an empty board. It ends in a win for one player and a loss for the other, or possibly in a draw.

A complete tree is a representation of all the possible plays of the game. The root node is the initial state, in which it is the first player's turn to move (the player X).

The successors of the initial state are the states the player can reach in one move, their successors are the states resulting from the other player's possible replies, and so on.



Assignment

Write about:

- Mini Max algorithm
- Alpha Beta pruning

Constraint Satisfaction Problems

- The process of finding a solution to set of variables V_i (V_1, V_2, \dots, V_n) and a set of constraint C_i (C_1, C_2, \dots, C_m)
- There is no any specified rule to define the procedure to solve the CSP.
- Eg: **8-queens problem**
- The CSP is a two-step process
 - ✗ Constraints are discovered and propagated as far as possible
 - ✗ There is still not found solution, then search begins

Crypto Arithmetic Problems

- › Crypt-Arithmetic Problems are substitution problems where digits representing a mathematical operation are replaced by unique digits.
- › Eg:

	A	B	C
+	D	E	F
	G	H	I

→

	0	2	4
+	5	8	9
	6	1	3

Solution:

$$1. \ A \neq B \neq C \neq D \neq E \neq F \neq G \neq H \neq I$$

$$2. \ C + F = I$$

$$C + F = 10 + I \text{ (I as carry)}$$

$$3. \ B + E = H$$

$$B + E = 10 + H$$

$$B + E + 1 = H$$

$$B + E + 1 = 10 + H \text{ (H as carry)}$$

$$4. \ A + D = G$$

$$A + D + 1 = G$$

Step 1:

Domain of C = {1, 2, 3, 4, 5, 6, 7, 8, 9}

Domain of F = {1, 2, 3, 4, 5, 6, 7, 8, 9}

So,

Domain of I = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

Select C = 4 & F = 9 Then I = 3 (Carry = 1)

So,

	A	B	4
+	D	E	9
	G	H	3

Step 2:

Domain of B = {1, 2, 5, 6, 7, 8}

Domain of E = {1, 2, 5, 6, 7, 8}

So,

Domain of H = {0, 1, 2, 5, 6, 7, 8}

Select B = 2 & E = 8

Then H = 10+1 {previous carry = 1 + (Carry = 1)}

So,

	A	2	4
+	D	8	9
	G	1	3

Step 3:

Domain of A = {0, 5, 6, 7}

Domain of D = {0, 5, 6, 7}

So, Domain of G = {5, 6}

Select A = 0 & D = 5 Then G = 6 (with addition of Carry)

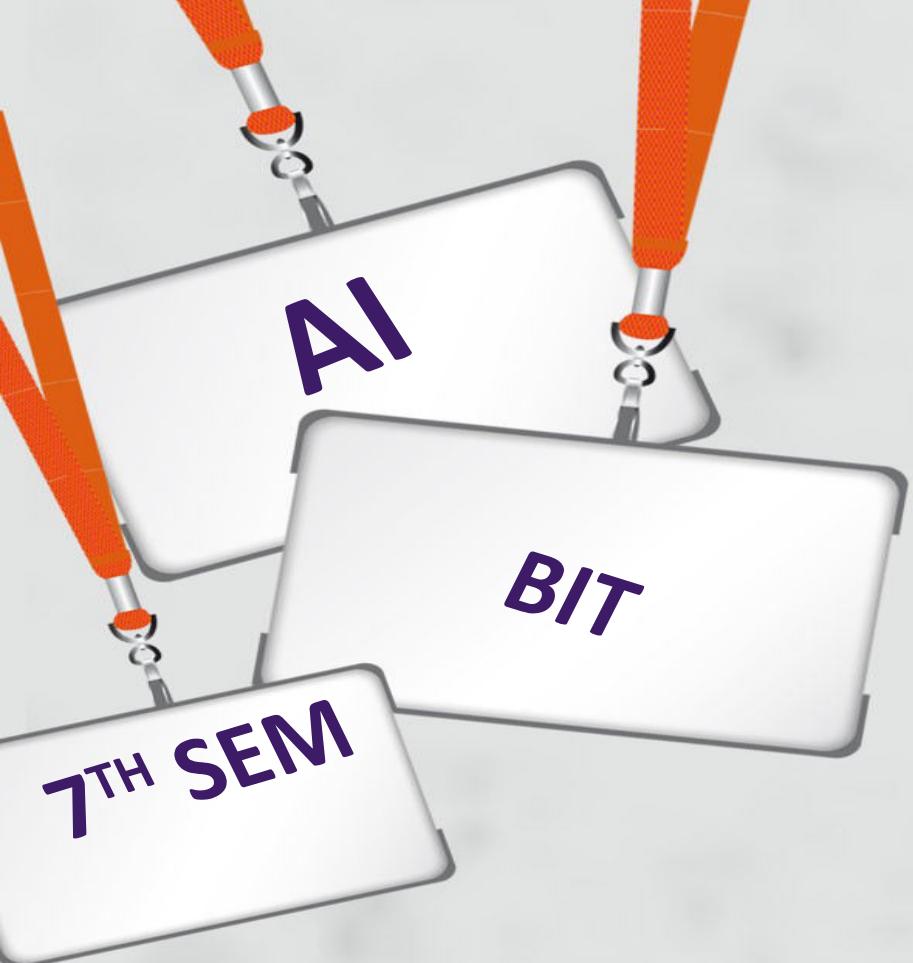
So,

Hence, the required solutions are:

A = 0, B = 2, C = 4, D = 5, E = 8, F = 9, G = 6, H = 1, I = 3.

Assignment

Solve any 2 cryptoarithmetic problems.



Chapter 4

KNOWLEDGE REPRESENTATION

Knowledge

Knowledge is the information about a domain that can be used to solve problems in that domain.

There are many other definitions such as:

- Knowledge is "information combined with experience, context, interpretation, and reflection. It is a high-value form of information that is ready to apply to decisions and actions." (T. Davenport et al., 1998)
- Knowledge is "human expertise stored in a person's mind, gained through experience, and interaction with the person's environment." (Sunasee and Sewery, 2002)
- Knowledge is "information evaluated and organized by the human mind so that it can be used purposefully, e.g., conclusions or explanations." (Rousa, 2002)

Research literature classifies knowledge as follows:

- | | |
|--------------------------------|---|
| Classification-based Knowledge | » Ability to classify information |
| Decision-oriented Knowledge | » Choosing the best option |
| Descriptive knowledge | » State of some world (heuristic) |
| Procedural knowledge | » How to do something |
| Reasoning knowledge | » What conclusion is valid in what situation? |
| Assimilative knowledge | » What its impact is? |

Types of knowledge

Meta Knowledge

It is knowledge about knowledge and how to gain them.

Procedural knowledge

Procedural knowledge is related to the performance of some task.

For example, sequence of steps to solve a problem is procedural knowledge.

Declarative knowledge

Declarative knowledge is passive knowledge in the form of statements of facts about the world. For example, mark statement of a student is declarative knowledge.

Heuristic knowledge

Heuristic knowledge is used to make judgments and also to simplify solution of problems. It is acquired through experience.

An expert uses his knowledge that he has gathered due to his experience and learning.

Structural Knowledge

Describes what relationship exists between objects.

Knowledge Representation

Knowledge representation (KR) is the study of how knowledge about the world can be represented and what kinds of reasoning can be done with that knowledge.

Knowledge Representation is the method used to encode knowledge in Intelligent Systems.

Some issues that arise in knowledge representation from an AI perspective are:

- How do people represent knowledge?
- What is the nature of knowledge and how do we represent it?
- Should a representation scheme deal with a particular domain or should it be general purpose?
- How expressive is a representation scheme or formal language?
- Should the scheme be declarative or procedural?

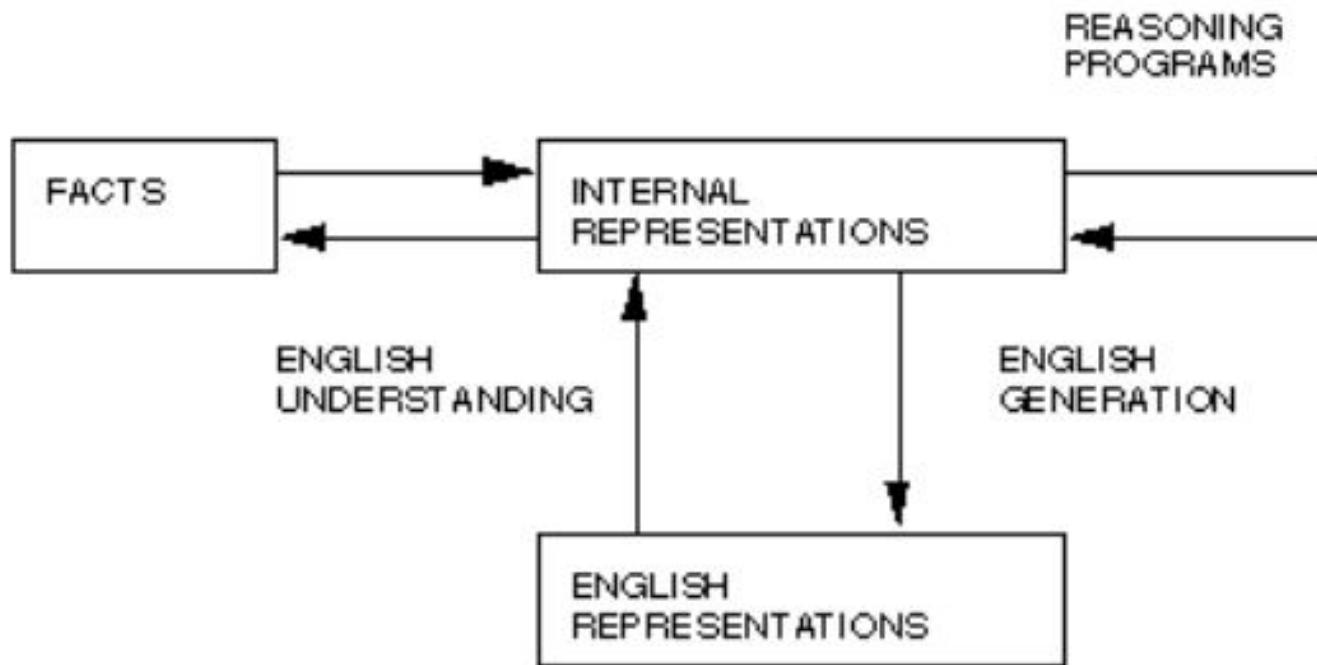


Fig: Two entities in Knowledge Representation

The following properties/Characters should be possessed by a knowledge representation system.

➤ **Representational Adequacy**

The ability to represent the required knowledge;

➤ **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 mechanisms into the most productive directions by storing appropriate guides;

➤ **Acquisitional Efficiency**

The ability to acquire new knowledge using automatic methods wherever possible rather than reliance on human intervention.

Approaches to Knowledge Representation

- **Rule-based**
 - IF <condition> THEN <conclusion>
- **Object-based**
 - Frames
 - Scripts
 - Semantic Networks

Rule based approach:

Rule-based systems are used as a way to store and manipulate knowledge to interpret information in a useful way. In this approach, idea is to use production rules, sometimes called IF-THEN rules. The syntax structure is

IF <premise> THEN <action>

<premise> - is Boolean. The AND, and to a lesser degree OR and NOT, logical connectives are possible.

<action> - a series of statements

Example:

“If the patient over 39 has stiff neck, high fever and a headache, check for Brain Meningitis”. Then it can be represented in rule based approach as:

IF

<fever, over, 39> and <neck, stiff, yes> and <head, pain, yes>

THEN

Add(<PATIENT,DIAGNOSE, MENINGITIS>)

Semantics net

- Study of Meaning
- Semantic networks can
 - Show natural relationships between objects/concepts
 - Be used to represent declarative/descriptive knowledge
- Semantic Network is the graphical representation of the knowledge.
- Semantic networks are constructed using nodes linked by directional lines called arcs

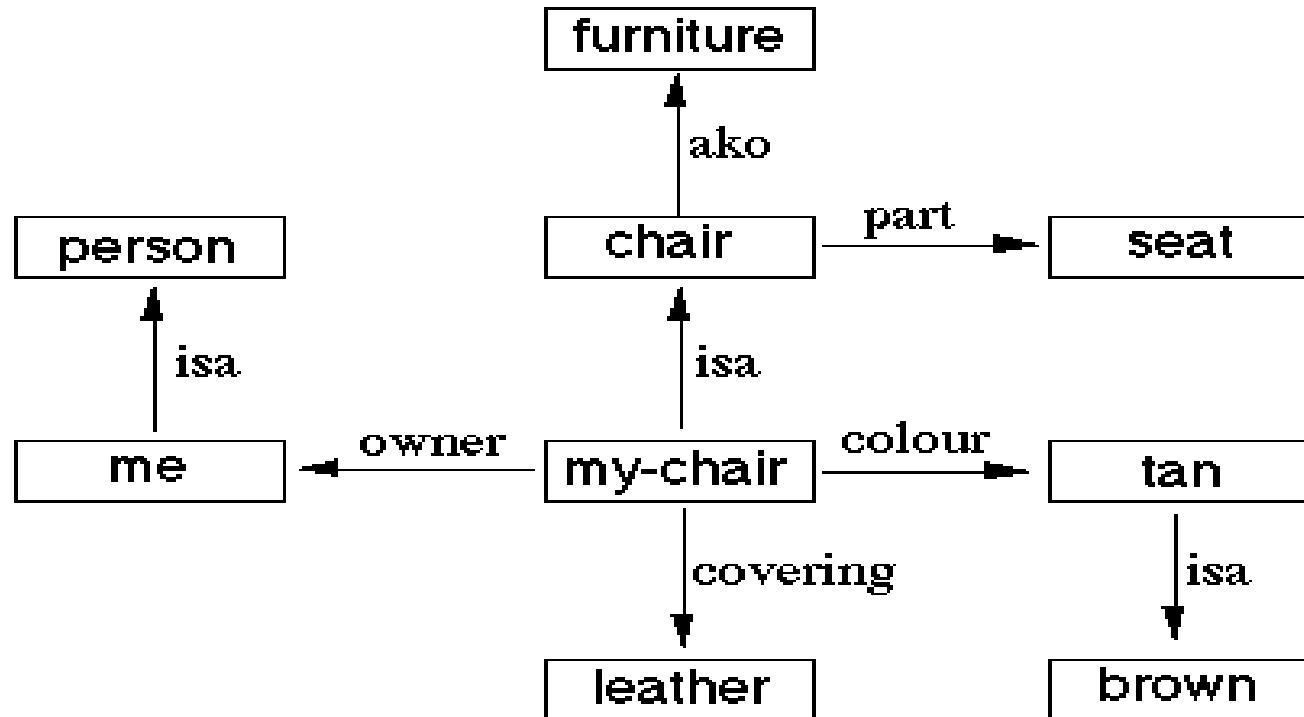
Advantages of a semantic network

- Easy to understand.
- Quick inference possible.
- Supports default reasoning.

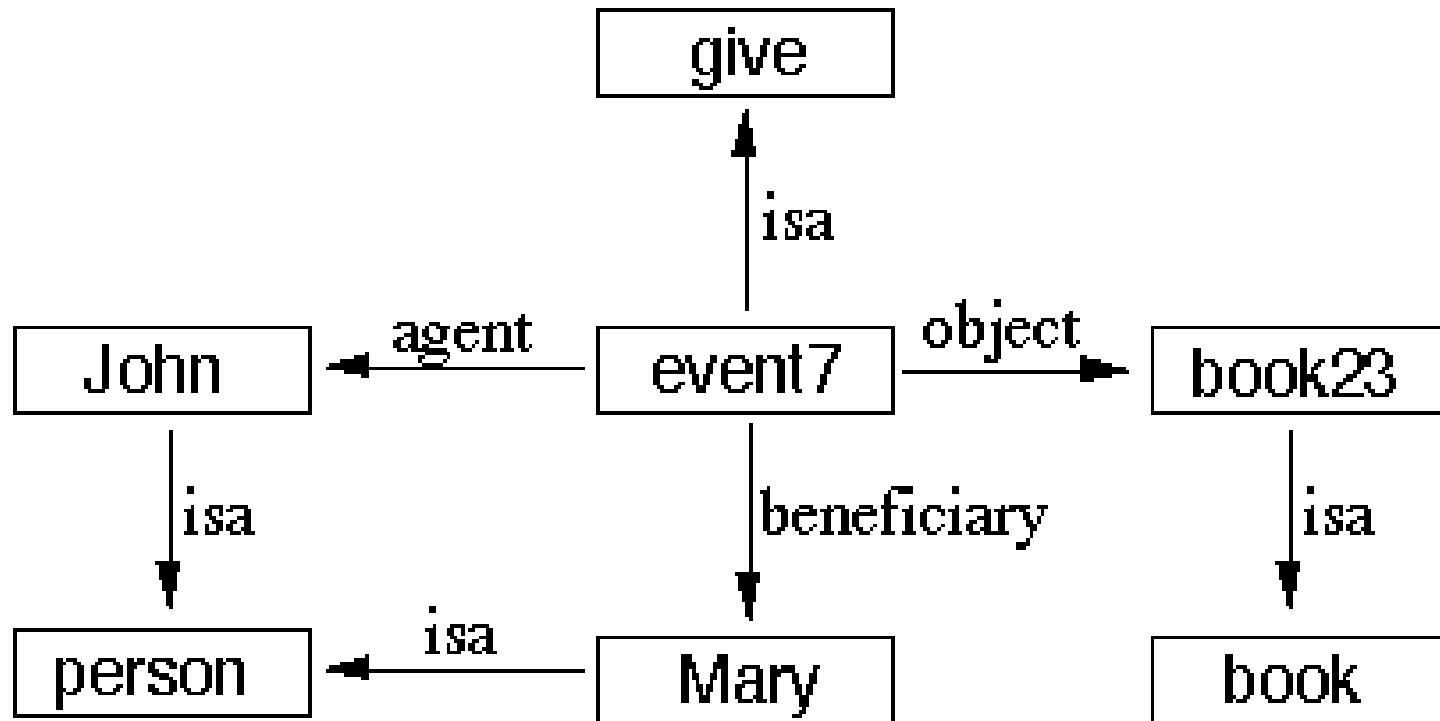
Disadvantages of a semantic network

- incomplete (no expressed operational/procedural knowledge)
- no interpretation standard
- lack of standards, ambiguity in node/link descriptions

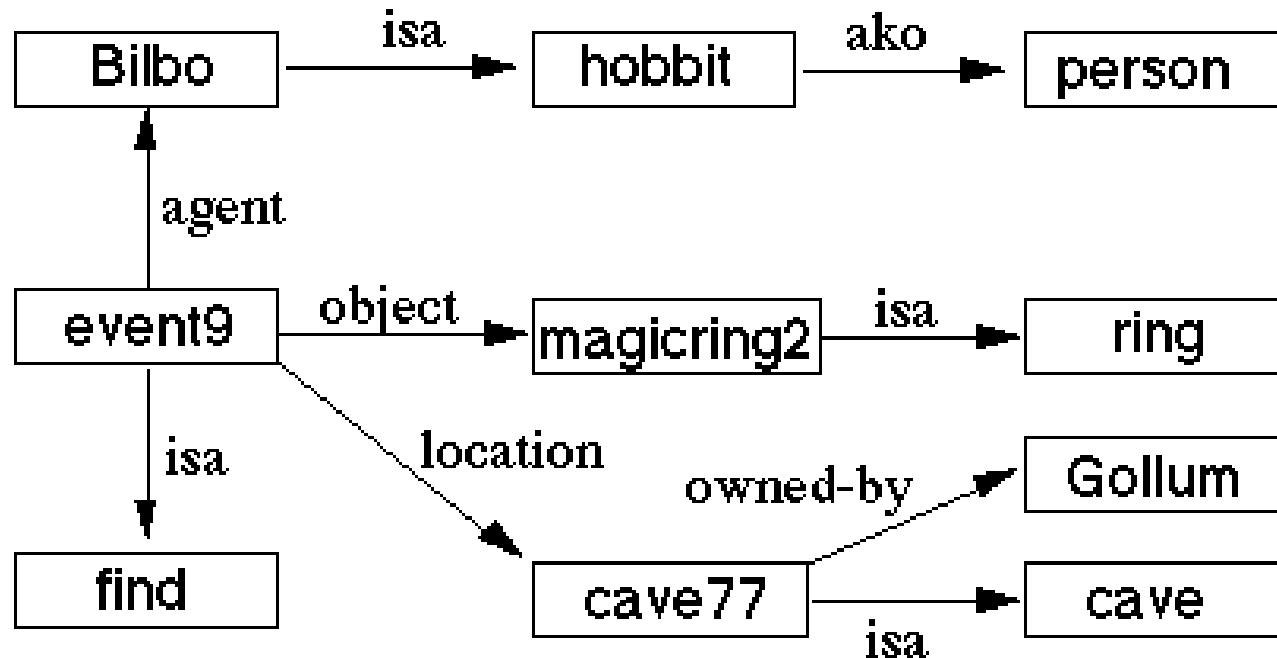
I own a tan leather chair



John gives the book to Mary



Bilbo finds the magic ring in Gollum's cave



Frame

- A *frame* is a data structure containing typical knowledge about a concept or object (Marvin Minsky (mid 1970s)).
- A frame represents knowledge about real world things (or entities).
- Each frame has a name and slots. Slots are the properties of the entity that has the name, and they have values or pointer to other frames (a table like data structure).
- A particular value may be:
 - a default value
 - an inherited value from a higher frame
 - a procedure, called a daemon, to find a value
 - a specific value, which might represent an exception.

Wilfried Honekamp	
Class	Human
Location	Home
Occupation	Officer
	
Human	
Class	Creature
Location	Earth
Walks on	Two legs

Fig: Two frames describing a human being

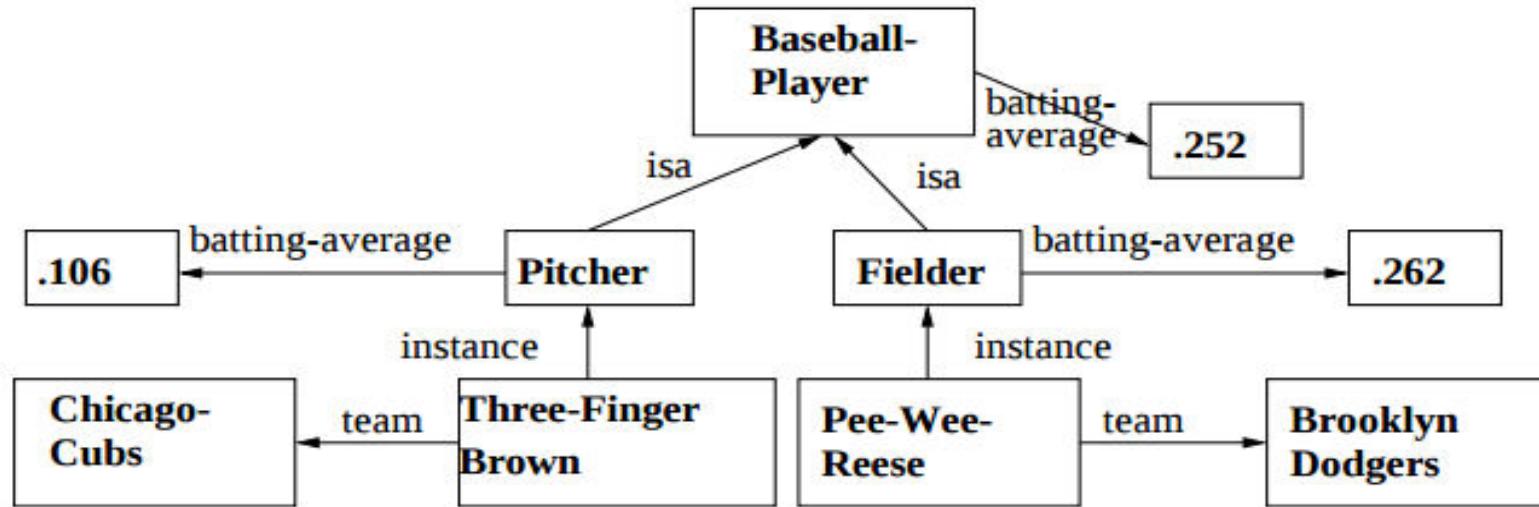
Disadvantages

- complex
- reasoning (inferencing) is difficult
- explanation is difficult, expressive limitation

Advantages

- knowledge domain can be naturally structured [a similar motivation as for the O-O approach].
- easy to include the idea of default values, detect missing values, include specialised procedures and to add further slots to the frames

Converting Semantic Net into Frame



Sample frame for above semantic network

Baseball Player
<i>is-a:</i> Adult Male
<i>batting average:</i> .252
<i>bats:</i> equal to handed
<i>team:</i>
:

Fielder
<i>is-a:</i> Baseball player
<i>batting average:</i> .262
Pee-Wee-Reese
<i>instance:</i> Baseball player
<i>team:</i> Brooklyn Dodgers

Axiom

An **axiom** is a sentence or proposition that is not proved or demonstrated and is considered as self-evident or as an initial necessary consensus for a theory building or acceptance. According as requirements, the new sentences are added to the knowledge base and then new sentences are also derived from old axiom & theorems, called **inference**.

Logic is method of reasoning process in which conclusions are drawn from premises using rules of inference. The logic is knowledge representation technique that involves:

- **Syntax:** defines well-formed sentences or legal expression in the language
- **Semantics:** defines the "meaning" of sentences
- **Inference rules:** for manipulating sentences in the language

Basically, the logic can be classified as:

- Proposition (or statements or calculus) logic
- Predicate [or First Order Predicate Logic (FOPL)] logic

Propositional Logic

A proposition is a declarative sentence to which only one of the “Truth value” (i.e. TRUE or FALSE) can be assigned (but not both). Hence, the propositional logic is also called Boolean logic. When a proposition is true, we say that its truth value is T, otherwise its truth value is F.

For example:

- The square of 4 is 16 →T
- The square of 5 is 27 →F

Atomic Sentences(Simple)

- The atomic sentences consist of a single proposition symbol.
- Each such symbol stands for a proposition that can be true or false.
- We use symbols that start with an uppercase letter and may contain other letters or subscripts, for example: p, q, r, s etc.
- For example:
p = Sun rises in West. (False sentence)

Complex Sentences (molecular or combined or compound)

The two or more statements connected together with some logical connectives such as AND (\wedge), OR (\vee), Implication (\rightarrow), etc.

Name	Representation	Meaning
Negation	$\neg p$	not p
Conjunction (true when both statement are true, otherwise false)	$p \wedge q$	p and q
Disjunction (false when both statement are false, otherwise true)	$p \vee q$	p or q (or both)
Exclusive Or (false when both statement are same)	$p \oplus q$	either p or q, but not both
Implication (false when p is true and q is false)	$p \rightarrow q$	if p then q
Bi-conditional or Bi-implication (true when both statement have same	$p \leftrightarrow q$	p if and only if q

Converse:	If $p \rightarrow q$ is an implication, then its converse is $q \rightarrow p$
Inverse:	If $p \rightarrow q$ is an implication, then its inverse is $\neg p \rightarrow \neg q$
Contrapositive:	If $p \rightarrow q$ is an implication, then its contrapositive is $\neg q \rightarrow \neg p$

Q. Verify that $p \leftrightarrow q$ is equivalent to $(p \rightarrow q) \wedge (q \rightarrow p)$

p	q	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \wedge (q \rightarrow p)$	$p \leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

Q. Construct the truth table of $\neg(p \wedge q) \vee (r \wedge \neg p)$

p	q	r			$\neg p$		
T	T	T	T	F	F	F	F
T	T	F	T	F	F	F	F
T	F	T	F	T	F	F	T
T	F	F	F	T	F	F	T
F	T	T	F	T	T	T	T
F	T	F	F	T	T	F	T
F	F	T	F	T	T	T	T
F	F	F	F	T	T	F	T

Logical equivalence

Two proposition p and q are logically equivalent and written as if both p and q have identical truth values

$$\neg(p \wedge q) \equiv (\neg p \vee \neg q)$$

The following logical equivalences apply to any statements; the p's, q's and r's can stand for atomic statements or compound statements.

i. Double Negative Law

$$\neg(\neg p) \equiv p$$

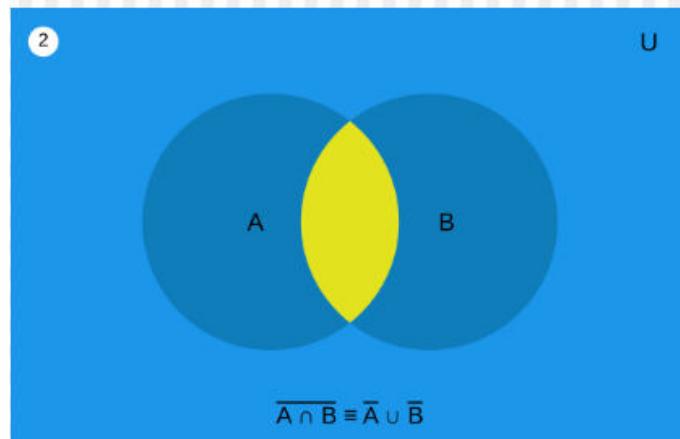
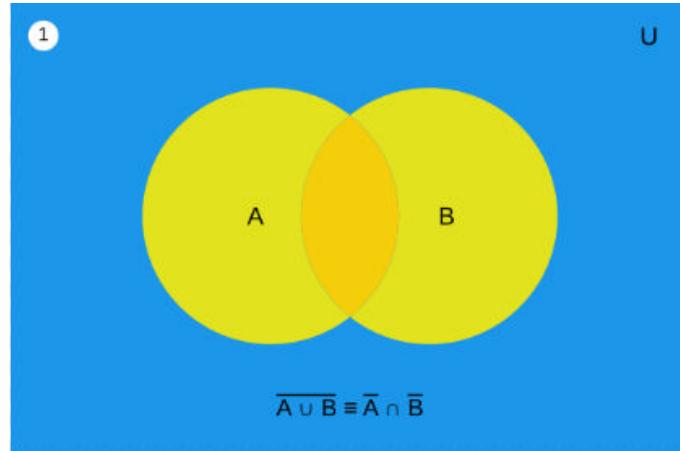
ii. De Morgan's Laws

$$\neg(p \vee q) \equiv (\neg p) \wedge (\neg q)$$

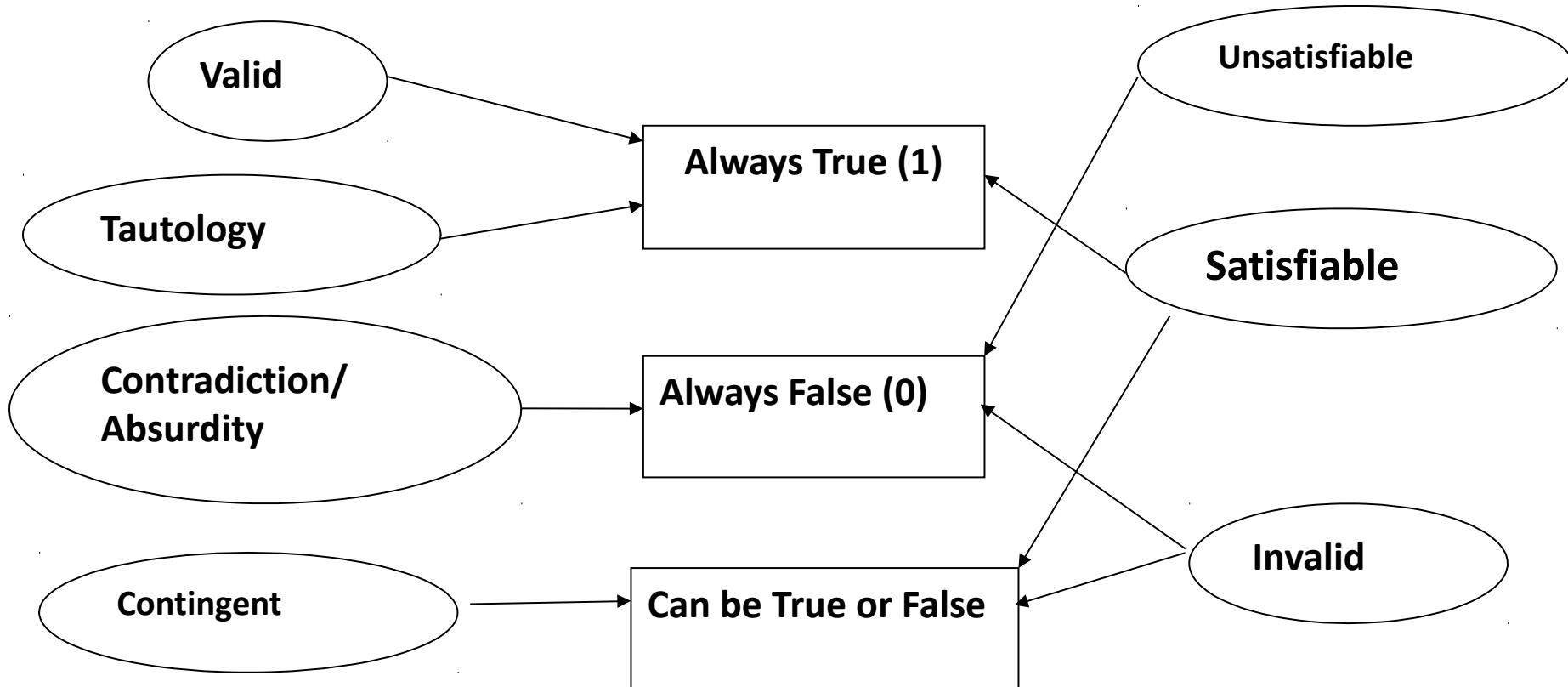
$$\neg(p \wedge q) \equiv (\neg p) \vee (\neg q)$$

iii. Distributive Laws

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$$



Tautology	If a proposition have a truth value for every interpretation
	<p>E.g.:</p> <p style="text-align: center;">→ → → →</p> <p>i) $p \vee \neg p$</p> <p>ii) $[(p \quad q) \wedge (q \quad r)] \quad (p \quad r)$</p>
Contradiction	If a proposition have a false value for every



Q. Write implication, converse, inverse, negation, contra positive of the following integration.

“The program is well structured only if it is readable”

p = “the program is readable”

q = “the program is well structured”

➤ Implication: $(p \rightarrow q)$

If the program is readable, then it is well structured.

➤ Converse: $(q \rightarrow p)$

If the program is well structured, then it is readable.

➤ Inverse: $(\neg p \rightarrow \neg q)$

If the program is not readable, then it is not well structured.

➤ Contrapositive: $(\neg q \rightarrow \neg p)$

If the program is not well structured, then it is not readable

➤ Negation of p : $(\neg p)$

The program is not readable.

Q• There are two restaurants next to each other.

One has a sign board as: “Good food is not cheap”.

The other has a sign board as “Cheap food is not good”.

Are both the sign board saying the same thing?

G = “Food is Good”

C = “Food is Cheap”

Sentence 1:- “Good food is not cheap” can be symbolically written as $G \rightarrow \neg C$

Sentence 2:- “Cheap food is not good” can be symbolically written as $C \rightarrow \neg G$

Now, The Truth Table is:

G	C	$\neg G$	$\neg C$	$G \rightarrow \neg C$	$C \rightarrow \neg G$
T	T	F	F	F	F
T	F	F	T	T	T
F	T	T	F	T	T
F	F	T	T	T	T

Since, $G \rightarrow \neg C$ and $C \rightarrow \neg G$ are logically equivalent. So both are saying same thing.

Predicate

- Predicate is a part of declarative sentences describing the properties of an object or relation among objects. For example: “is a student” is a predicate as ‘A is a student’ and ‘B is a student’.
- A predicate logic is a formal system that uses objects/variables and quantifiers (\forall, \exists) to formulate propositions.

Assignment

Differentiate between Propositional Logic and Predicate Logic

Quantification

- A quantifier is a symbol that permits one to declare the range or scope of variables in a logical expression.
- The process of binding propositional variable over a given domain is called quantification.
- Two common quantifier are the existential quantifier (“there exists or for some or at least one”) and universal quantifier (“for all or for each or for any or for every and or for arbitrary”).

Universal Quantifier (\forall : For All)

- It is denoted by \forall and used for universal quantification.
- The universal quantification of $p(x)$ denoted by $\forall x p(x)$ is proposition that is true for all values in universal set
- The universal quantifier is read as:
 - For all x , $p(x)$ holds
 - For each x , $p(x)$ holds
 - For every x , $p(x)$ holds

Existential Quantifier(\exists : For Some)

- It is denoted by \exists and used for existential quantification.
- The existential quantification of $p(x)$ denoted by $\exists x p(x)$ is proposition that is true for some values in universal set.
- The existential quantifier is read as:
 - There is an x , such that $p(x)$
 - There is at least one x such that $p(x)$
 - For some x , $p(x)$

Q. Assume that

P (x) denotes “x is an accountant.”

Q (x) denotes “x owns a maruti.”

Now, represent the following statement symbols.

- a) All accountants own maruti.

Meaning: For all x, if x is an accountant, then x owns maruti

$$\forall x \ p(x) \rightarrow q(x)$$

$$\Rightarrow \forall x (p(x) \rightarrow q(x))$$

- b) Some accountants own maruti

Meaning: For some x, x is an accountant and x owns maruti

$$\exists x \ p(x) \wedge q(x)$$

$$\Rightarrow \exists x (p(x) \wedge q(x))$$

- c) All owners of maruti are accountants

Meaning: For all \underline{x} , if x is a owner of maruti, then x is an accountant.

$$\Rightarrow \forall \underline{x} (q(x) \rightarrow f(x))$$

- d) Someone who owns a maruti, is an accountant

Meaning: For some x , who owns a maruti and x is an accountant

$$\Rightarrow \exists \underline{x} (q(x) \wedge f(x))$$

Q. Convert into FOPL

- a. All men are people.
- b. Marcus was Pompeian.
- c. All Pompeian were Roman.
- d. Ram tries to assassinate Hari.
- e. All Romans were either loyal to caser or hated him.
- f. Socrates is a man. All men are mortal; therefore Socrates is mortal.
- g. Some student in this class has studied mathematics.

a. All men are people.

$$\Rightarrow \forall x \text{MAN}(x) \rightarrow \text{PEOPLE}(x)$$

b. Marcus was Pompeian.

$$\Rightarrow \text{POMPEIAN}(\text{Marcus})$$

c. All Pompeian were Roman.

$$\Rightarrow \forall x \text{POMPEIAN}(x) \rightarrow \text{ROMAN}(x)$$

d. Ram tries to assassinate Hari.

$$\Rightarrow \text{ASSASSINATE}(\text{Ram}, \text{Hari})$$

e. All Romans were either loyal to caser or hated him.

$$\Rightarrow \forall x \text{ ROMAN}(x) \rightarrow \text{LOYAL}(x, \text{caser}) \vee \text{HATES}(x, \text{caser})$$

f. Socrates is a man. All men are mortal; therefore Socrates is mortal.

$$\Rightarrow \text{MAN}(\text{Socrates}), \forall x \text{ MAN}(x) \rightarrow \text{MORTAL}(x), \text{MORTAL}(\text{Socrates})$$

g. Some student in this class has studied mathematics.

\Rightarrow Let

- $S(x)$ = “ x is a student in this class”
- $M(x)$ = “ x has studied mathematics”

Hence, required expression is: $\exists x [S(x) \wedge M(x)]$

Clausal Forms

- A clause is an expression formed from a finite collection of literals (variables or their negations)
- A clause that contains only \vee is called a **disjunctive clause** and only \wedge is called a **conjunctive clause**.
 - Negation is allowed, but only directly on variables.
 - $p \vee \neg q \vee r$: a disjunctive clause
 - $\neg p \wedge q \wedge \neg r$: a conjunctive clause
 - $\neg p \wedge \neg q \vee r$: neither

CNF and DNF

If we put a bunch of disjunctive clauses together with \wedge , it is called conjunctive normal form.

- For example: $(p \vee r) \wedge (\neg q \vee \neg r) \wedge q$ is in conjunctive normal form.

Similarly, putting conjunctive clauses together with \vee , it is called disjunctive normal form.

- For example: $(p \wedge \neg q \wedge r) \vee (\neg q \wedge \neg r)$ is in disjunctive normal form.

Conversion Procedure for proposition into Normal Forms

- › We can use the definitions to get rid of \rightarrow , \leftrightarrow , and \oplus

$$B \leftrightarrow (P \vee Q) \text{ as } (B \rightarrow (P \vee Q)) \wedge ((P \vee Q) \rightarrow B)$$

$$(B \rightarrow (P \vee Q)) \text{ as } (\neg B \vee P \vee Q)$$

$$(P \oplus Q) \text{ as } (P \wedge \neg Q) \vee (\neg P \wedge Q)$$

- › Use DeMorgan's laws.

$$\neg(\alpha \wedge \beta) \equiv (\neg \alpha \vee \neg \beta) \text{ (De Morgan)}$$

$$\neg(\alpha \vee \beta) \equiv (\neg \alpha \wedge \neg \beta) \text{ (De Morgan)}$$

- › Use double negation to get rid of any $\neg\neg$ that showed up.
- › Use the distributive rules to move things in/out of parens as we need to.

$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$$

Converting to conjunctive normal form

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

$$\equiv \neg((\neg\neg p \vee \neg q) \wedge \neg r) \quad [\text{definition}]$$

$$\equiv \neg(p \vee \neg q) \wedge \neg r \quad [\text{double negation}]$$

$$\equiv \neg(p \vee \neg q) \vee \neg\neg r \quad [\text{DeMorgan's}]$$

$$\equiv \neg(p \vee \neg q) \vee r \quad [\text{double negation}]$$

$$\equiv (\neg p \wedge \neg\neg q) \vee r \quad [\text{DeMorgan's}]$$

$$\equiv (\neg p \wedge q) \vee r \quad [\text{double negation}]$$

$$\equiv (\neg p \vee r) \wedge (q \vee r) \quad [\text{distributive}]$$

b. $(p \rightarrow q) \rightarrow (\neg r \wedge q)$ into DNF

$$\equiv \neg(p \rightarrow q) \vee (\neg r \wedge q) \quad [\text{definition}]$$

$$\equiv \neg(\neg p \vee q) \vee (\neg r \wedge q) \quad [\text{definition}]$$

$$\equiv (\neg\neg p \wedge \neg q) \vee (\neg r \wedge q) \quad [\text{DeMorgan's}]$$

$$\equiv (p \wedge \neg q) \vee (\neg r \wedge q) \quad [\text{double negation}]$$

$$\equiv (p \wedge \neg q) \vee (\neg r \wedge q)$$

$$\equiv (p \vee (\neg r \wedge q)) \wedge (\neg q \vee (\neg r \wedge q))$$

[distributive]

$$\equiv (p \vee (\neg r \wedge q)) \wedge (\neg q \vee \neg r) \wedge (\neg q \vee q)$$

[distributive]

$$\equiv (p \vee (\neg r \wedge q)) \wedge (\neg q \vee \neg r) \wedge \mathbf{T}$$

[negation]

$$\equiv (p \vee (\neg r \wedge q)) \wedge (\neg q \vee \neg r)$$

[identity]

$$\equiv (p \vee \neg r) \wedge (p \vee q) \wedge (\neg q \vee \neg r)$$

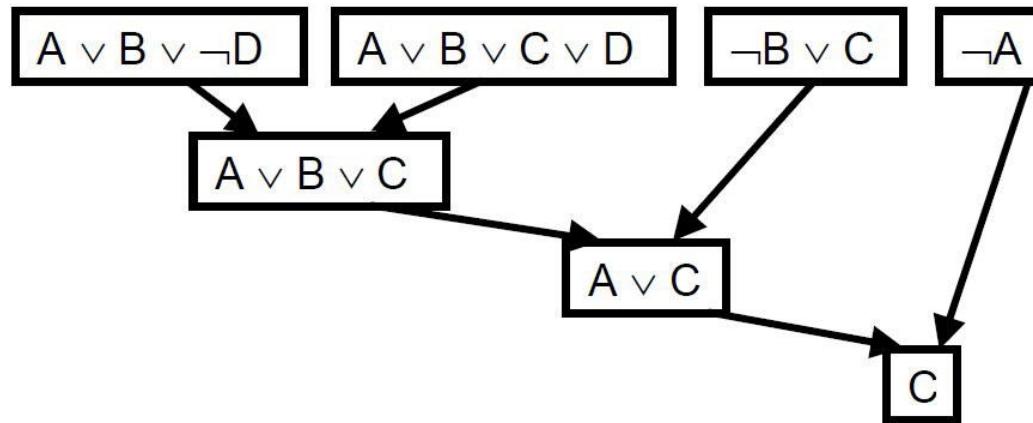
[distributive]

Resolution in Propositional Logic

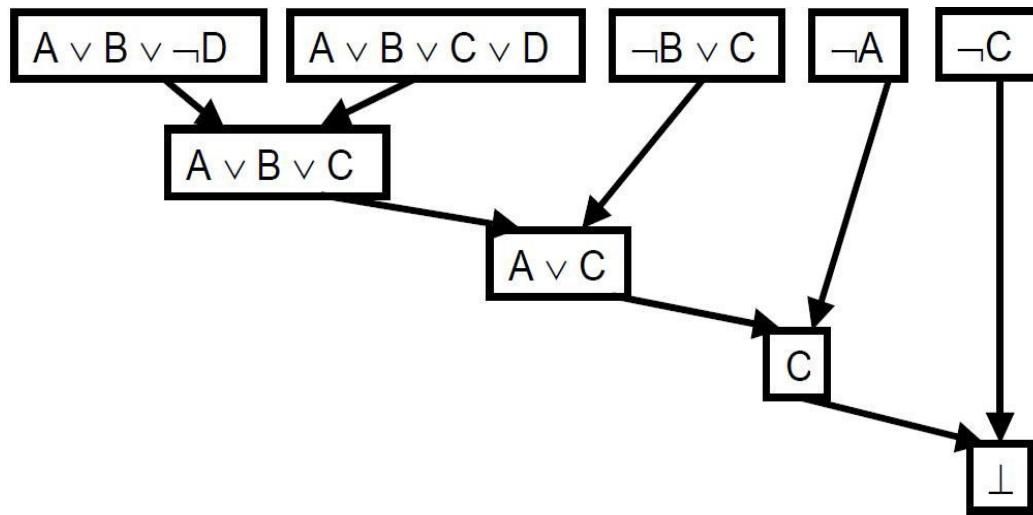
- The resolution rule in propositional logic is a single valid inference rule that produces a new clause implied by two clauses
- Resolution principle was introduced by John Alan Robinson in 1965.
- The resolution technique can be applied for sentences in propositional logic and first-order logic.
- Resolution technique can be used only for disjunctions of literals to derive new conclusion.
- The resolution rule for the propositional calculus can be stated as following:
 $(P \vee Q) \text{ and } (\neg Q \vee R), \text{ gives } (P \vee R).$

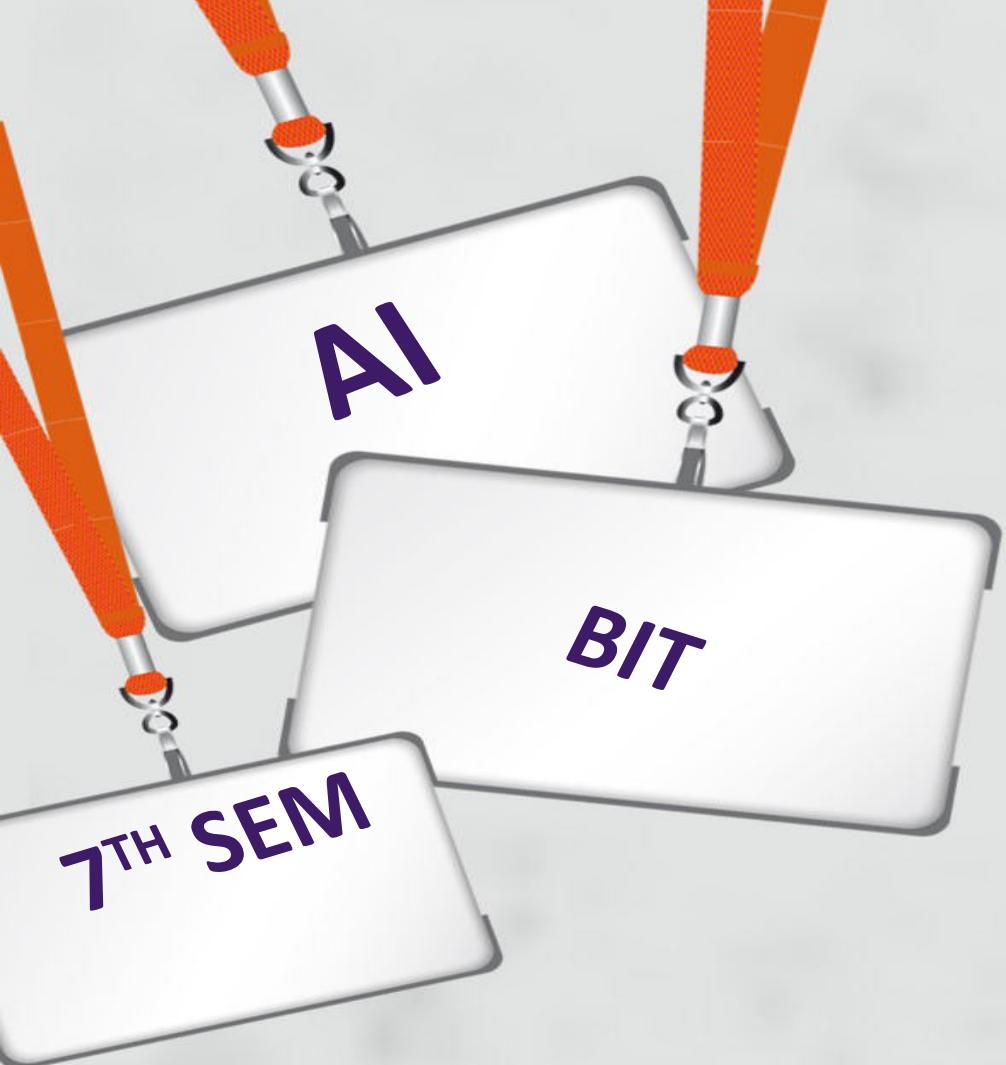
Q. Let $P_1 = A \vee B \vee \neg D$, $P_2 = A \vee B \vee C \vee D$, $P_3 = \neg B \vee C$, $P_4 = \neg A$, $P_5 = C$
then Show that $\{P_1, P_2, P_3, P_4\} \Rightarrow P_5$

Solution:



Q. Let $P_1 = A \vee B \vee \neg D$, $P_2 = A \vee B \vee C \vee D$, $P_3 = \neg B \vee C$, $P_4 = \neg A$, $P_5 = C$ then
 Show that $\{P_1, P_2, P_3, P_4, \neg P_5\} = \perp$





Chapter 5

REASONING

Inference Theorem on Propositional Statements

Let, P, Q, and R be any propositional statements, then

Addition rule

$$\frac{p}{\therefore p \vee q}$$

Example:

- Ram is a student of BE.
-
- Ram is a student of BE or BCA.

Simplication rule

$p \wedge q$

$\therefore p$

Example:

-Subodh and Shyam are the students of BE.

\therefore Subodh is the student of BE

or

Shyam is the student of BE.

Modus Ponens Rule

$$p \rightarrow q$$

$$\frac{p}{\therefore q}$$

Example:

- If Ram is hard working, then he is intelligent.
- Ram is hard working.

\therefore Ram is intelligent.

Modus Tollens Rule

$$p \rightarrow q$$

$$\neg q$$

$$\therefore \neg p$$

Example:

- We will go swimming only if it is sunny.
- It is not sunny.

\therefore We will not go swimming.

Hypothetical Syllogism Rule

$$p \rightarrow q$$

$$q \rightarrow r$$

$$\therefore p \rightarrow r$$

Example:

- If Subodh is a BE student, then he loves programming.

- If Subodh loves programming, then he is expert in java.

\therefore If Subodh is a BE student, then he is expert in java.

Disjunctive Syllogism Rule

$p \vee q$

$\neg p$

$\therefore q$

Example:

- Today is Wednesday or Thursday.
- Today is not Wednesday.

\therefore Today is Thursday.

Conjunction rule

p

q

$\therefore p \wedge q$

Example:

- Shyam is the student of BE.
- Hari is the student of BE.

\therefore Shyam and Hari are the students of BE.

Resolution Rule

$$\frac{\begin{array}{c} p \vee q \\ \hline \neg q \vee r \end{array}}{\therefore p \vee r}$$

Can we conclude that the conclusion is true if the premises are true?

- If Socrates is human, then Socrates is mortal.
 - Socrates is human.
- ∴ Socrates is mortal.

Can we conclude that the conclusion is true if the premises are true?

- If George does not have eight legs, then he is not a spider.
 - George is a spider.
- ∴ George has eight legs.

If I work all night on this homework, then I can answer all the exercises. If I answer all the exercises, I will understand the material. Therefore, if I work all night on this homework, then I will understand the material.

Q. “If you send me an e-mail message then I will finish writing the program”, “If you do not send me an e-mail message then I will go to sleep early”, and “If I go to sleep early then I will wake up feeling refreshed”. Lead to the conclusion “If I do not finish writing the program then I will wake up feeling refreshed”.

Solution:

Let

 $p = \text{"You send me an e-mail message"}$ $q = \text{"I will finish writing the program"}$ $r = \text{"I will go to sleep early"}$ $s = \text{"I will wake up feeling refreshed"}$ **Hypothesis:**

- $p \rightarrow q$
- $\neg p \rightarrow r$
- $r \rightarrow s$

Conclusion: $\neg q \rightarrow s$

Steps	Operations	Reasons
1	$p \rightarrow q$	Given hypothesis
2	$\neg q \rightarrow \neg p$	Using contra positive on 1
3	$\neg p \rightarrow r$	Given hypothesis
4	$\neg q \rightarrow r$	Using hypothetical syllogism on 2 and 3
5	$r \rightarrow s$	Given hypothesis
6	$\neg q \rightarrow s$	Using hypothetical syllogism on 4 and 5

Hence the given hypotheses lead to the conclusion $\neg q \rightarrow s$

Q.“ Hari is playing in garden”, “If he is playing in garden then he is not doing homework”, “If he is not doing homework, then he is not learning” leads to the conclusion “He is not learning”.

Use rules of inference to show that the hypotheses “Randy works hard,” “If Randy works hard, then he is a dull boy,” and “If Randy is a dull boy, then he will not get the job” imply the conclusion “Randy will not get the job.”

Rules of Inference for Quantified Statements

- a. Universal Instantiation

$$\forall x p(x)$$

$$\frac{}{\therefore p(d)}$$

Where d is in the domain of discourse D.

For example: If all balls in a box are red then any randomly drawn ball is also red

- b. Universal Generalization

$$p(d)$$

$$\frac{}{\therefore \forall x p(x)}$$

Where, d is the domain of discourse D.

For example: If all the ball in a box are taken one by one in randomly manner and if all are red then we conclude that all balls in the box are red.

c. Existential Instantiations

$$\exists x p(x)$$

$$\therefore p(d)$$

For some d in the domain of discourse.

For example: If some balls in a box are red then resulting ball after experiment will also be red.

d. Existential Generalization

$$p(d)$$

$$\therefore \exists x p(x)$$

For some d in the domain of discourse.

For example: If we take only one random experiment for the ball drawn and apply the result of ball to the domain.

Q. Given Expression: All men are mortal. Einstein is a man. Prove that “Einstein is mortal” using FOPL.

Solution: Let

$$M(x) = \text{“}x \text{ is a man}\text{”}$$

$$N(x) = \text{“}x \text{ is mortal}\text{”}$$

Hypothesis: $\forall x [M(x) \rightarrow N(x)]$, $M(\text{Einstein})$

Conclusion: $N(\text{Einstein})$

Steps	Operations	Reasons
1.	$\forall x [M(x) \rightarrow N(x)]$	Given Hypothesis
2.	$M(\text{Einstein}) \rightarrow N(\text{Einstein})$	Using universal instantiation on 1
3.	$M(\text{Einstein})$	Given Hypothesis
4.	$N(\text{Einstein})$	Using modus pollens on 2 and 3

Hence the given hypotheses lead to the conclusion “Einstein is mortal”.

Q. Given Expression: “Lions are dangerous animals”, and “There are lions”. Prove that “There are dangerous animals” using FOPL.

Solution: Let

$$D(x) = \text{“}x \text{ is a dangerous animal”}$$

$$L(x) = \text{“}x \text{ is a lion”}$$

Hypothesis: $\forall x [L(x) \rightarrow D(x)]$, $\exists x L(x)$

Conclusion: $\exists x D(x)$

Steps	Operations	Reasons
1.	$\forall x [L(x) \rightarrow D(x)]$	Given Hypothesis
2.	$L(a) \rightarrow D(a)$	Using universal instantiation on 1
3.	$\exists x L(x)$	Given Hypothesis
4.	$L(a)$	Using existential instantiation on 3
5.	$D(a)$	Using modus pollens on 2 and 4
6.	$\exists x D(x)$	Using existential generalization on 5

Hence the given hypotheses lead to the conclusion “There are dangerous animals”

Q. Show that the premises “Everyone in this discrete mathematics class has taken a course in computer science” and “Marla is a student in this class” imply the conclusion “Marla has taken a course in computer science.”

Show that the premises “A student in this class has not read the book,” and “Everyone in this class passed the first exam” imply the conclusion “Someone who passed the first exam has not read the book.”

Solution: Let $C(x)$ be “ x is in this class,” $B(x)$ be “ x has read the book,” and $P(x)$ be “ x passed the first exam.” The premises are $\exists x(C(x) \wedge \neg B(x))$ and $\forall x(C(x) \rightarrow P(x))$. The conclusion is $\exists x(P(x) \wedge \neg B(x))$. These steps can be used to establish the conclusion from the premises.

Step	Reason
1. $\exists x(C(x) \wedge \neg B(x))$	Premise
2. $C(a) \wedge \neg B(a)$	Existential instantiation from (1)
3. $C(a)$	Simplification from (2)
4. $\forall x(C(x) \rightarrow P(x))$	Premise
5. $C(a) \rightarrow P(a)$	Universal instantiation from (4)
6. $P(a)$	Modus ponens from (3) and (5)
7. $\neg B(a)$	Simplification from (2)
8. $P(a) \wedge \neg B(a)$	Conjunction from (6) and (7)
9. $\exists x(P(x) \wedge \neg B(x))$	Existential generalization from (8)

Show that the premises:

- A student in Section A of the course has not read the book.
- Everyone in Section A of the course passed the first exam.

imply the conclusion

- Someone who passed the first exam has not read the book.

“All movies produced by John are wonderful”, and “John produced a movie about refugee”. If so, show the conclusion “There is a wonderful movie about refugee”.

$S(x)$ = “ x is a movie produced by John”

$C(x)$ = “ x is a movie about refugee”

$W(x)$ = “ x is a wonderful movie”

Hypothesis: $\forall x [S(x) \rightarrow W(x)], \exists x [S(x) \wedge C(x)]$

Conclusion: $\exists x [C(x) \wedge W(x)]$

Let “ m ” represent a particular movie

Steps	Operations	Reasons
1.	$\exists x [S(x) \wedge C(x)]$	Given hypothesis
2.	$S(m) \wedge C(m)$	Using existential instantiation on 1
3.	$S(m)$	Simplification on 2
4.	$\forall x [S(x) \rightarrow W(x)]$	Given hypothesis
5.	$S(m) \rightarrow W(m)$	Using universal instantiation on 4
6.	$W(m)$	Using modus penance on 3 and 5
7.	$C(m)$	Simplification on 2
8.	$W(m) \wedge C(m)$	Conjunction on 6 and 7
9.	$\exists x [C(x) \wedge W(x)]$	Using existential generalization on 8

Hence the given hypotheses lead to the conclusion “There is a wonderful movie about refugee”.

Monotonic Reasoning

- › A reasoning system is monotonic if the truthfulness of a conclusion does not change when new information is added to the system
- › Adding knowledge base does not reduce the set of propositions that can be derived.
- › Eg: We are A//B and B//C. We can conclude A//C.

Now adding a fact that A//D, does not change the result A//C

Non-Monotonic Reasoning

- A logic is non-monotonic if some conclusions can be invalidated by adding more knowledge in the knowledge base.
- Non monotonic system are harder to deal with than monotonic systems
- Non – monotonic systems require more storage space as well as more processing time than monotonic systems.
- Eg: Given “Birds Fly”. Simply means all birds fly.

But a fact “Ostrich is bird” is added will result the conclusion is false.

Bayes rule

Let A and B are two dependent events then the probability of the event A when the event B has already happened is called the conditional probability. It is denoted by $P(A|B)$ and is given by:

$$P(A|B) = P(A \cap B)/P(B), \text{ where } P(B) \neq 0, \Rightarrow P(A \cap B) = P(A|B) \cdot P(B) \quad \text{--- (i)}$$

Similarly,

$$P(B|A) = P(A \cap B)/P(A), \text{ where } P(A) \neq 0, \Rightarrow P(A \cap B) = P(B|A) \cdot P(A) \quad \text{--- (ii)}$$

From equation (i) and (ii), we have

$$P(B|A) \cdot P(A) = P(A|B) \cdot P(B)$$

Bayes rule is useful for those cases where $P(A|B)$ can be estimated but $P(B|A)$ is hard to find experimentally

- In a task such as medical diagnosis, we often have conditional probabilities on causal relationships and want to derive a diagnosis.
- A doctor knows the probability of symptoms condition to disease $P(S|D)$, and the patient knows his own feeling or symptoms $P(S)$. Also the doctor knows about probability of disease $P(D)$, then the probability of disease condition to symptoms can be defined as:

For example:

Let,

- S = Symptoms on patients such as stiff neck whose probability $P(S)$ is $=1/20$
- D = Disease known by doctor whose probability $P(D)$ is $= 1/50000$
- Given $P(S|D) = 0.5$ or 50%

Now, the probability of disease condition to symptoms,

$$P(D|S)$$

$$= [P(S|D) \cdot P(D)] / P(S)$$

$$= 0.0002$$

Application of Bayes Theorem

- Selecting best products among two manufacturers,
- In bio-chemistry deciding the diseases based on various blood sample tests. In fact those results are based on probability, **so it never be 100% true.**
- For project managers : All project managers want to know whether the projects they're working on will finish on time. So, as our example, we'll assume that a project manager asks the question: what's the probability that my project will finish on time? There are only two possibilities here: either the project finishes on (or before) time or it doesn't on the basis of various factors like tools, number of workers, efficiency of workers..

- **Bayesian Network** is a probabilistic graphical model that represents a set of variables and their conditional dependencies via a directed acyclic graph (DAG).
- For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

Case-Base Reasoning:

- AI programs solve problems by reasoning from first principles. They can explain their reasoning by reporting the string of deductions that led from the input data to the conclusion, with the Human Experts.
- An expert encountering a new problem is usually reminded of similar cases seen in the past, remembering the result of those cases and perhaps the reasoning behind those results.
Example: Medical expertise follow this pattern.
- Computer systems that solve new problems by analogy with old ones are often called Case Base Reasoning (CBR).

A successful CBR systems must answer the following questions.

1. How are cases organized in memory ?
2. How are relevant cases retrieved from the memory ?
3. How can previous cases be adopted to new problems ?
4. How are cases originally acquired ?

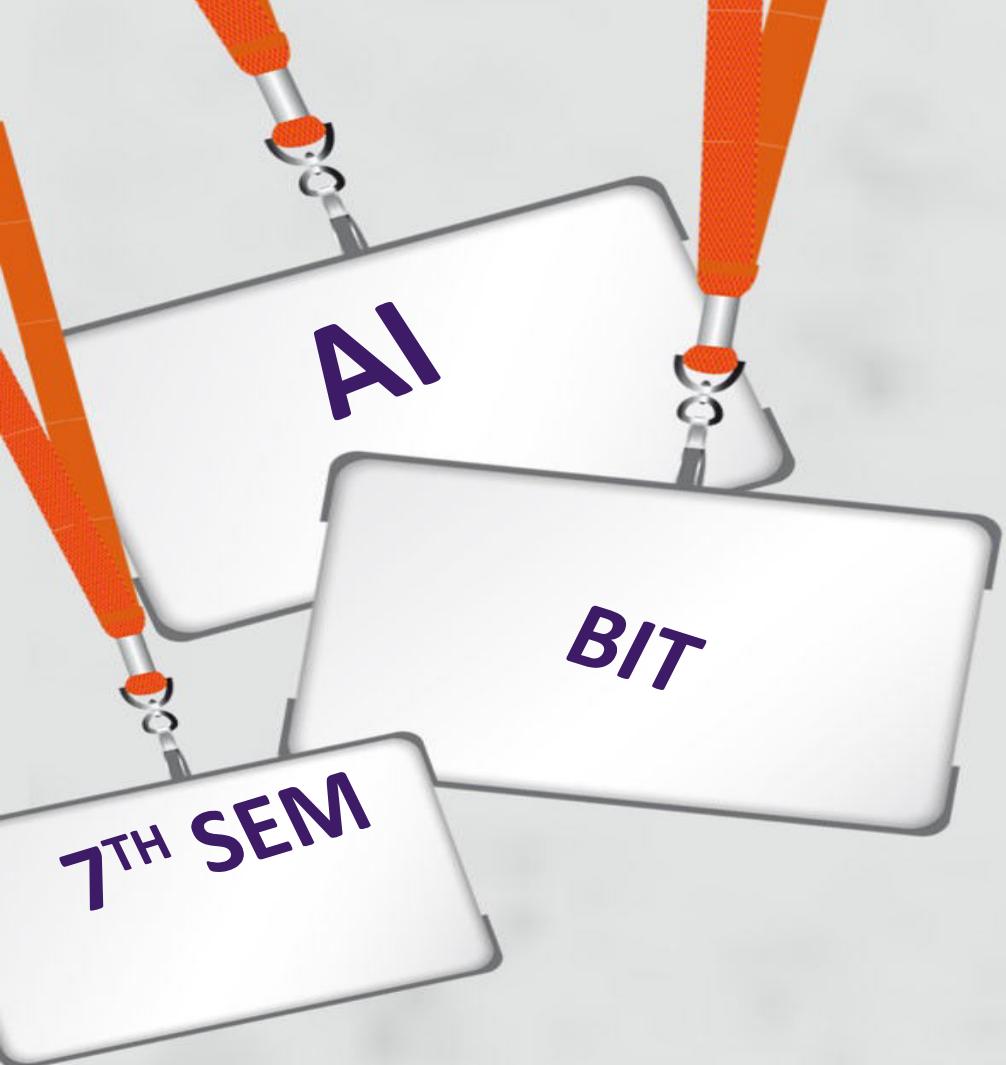
Reasoning with Uncertainty

- Though there are various types of uncertainty in various aspects of a reasoning system, the "reasoning with uncertainty" research in AI has been focused on the uncertainty of truth value, that is, to allow and process truth values other than "true" and "false".
- To develop a system that reasons with uncertainty means to provide the following:
 - ✗ a semantic explanation about the origin and nature of the uncertainty
 - ✗ a way to represent uncertainty in a formal language
 - ✗ a set of inference rules that derive uncertain (though well-justified) conclusions
 - ✗ an efficient memory-control mechanism for uncertainty management

➤

Assignment

Write short notes on Fuzzy Logic



Chapter 6

LEARNING

Learning

- Learning is the process of acquiring new or modifying existing knowledge, behaviors, skills, values, or preferences.
- Learning is the improvement of performance with experience over time.
- Learning element is the portion of a learning AI system that decides how to modify the performance element and implements those modifications.
- We all learn new knowledge through different methods, depending on the type of material to be learned, the amount of relevant knowledge we already possess, and the environment in which the learning takes place.

➤ There are five methods of learning . They are:

➤

- - Memorization (rote learning)
 - Direct instruction (by being told)
 - Analogy
 - Induction
 - Deduction

Memorization (rote learning)

- Learning by memorizations is the simplest form of learning.
- It requires the least amount of inference and is accomplished by simply copying the knowledge in the same form that it will be used directly into the knowledge base.
- Example:- Memorizing multiplication tables, formulate , etc.

Direct Instruction

- Direct instruction is a complex form of learning.
- This type of learning requires more inference than rote learning since the knowledge must be transformed into an operational form before learning when a teacher presents a number of facts directly to us in a well organized manner.

Analogical learning

- Analogical learning is the process of learning a new concept or solution through the use of similar known concepts or solutions.
- We use this type of learning when solving problems on an exam where previously learned examples serve as a guide or when make frequent use of analogical learning.
- This form of learning requires still more inferring than either of the previous forms. Since difficult transformations must be made between the known and unknown situations.

Learning by Induction

- Learning by induction is also one that is used frequently by humans .
- It is a **powerful form of learning like analogical learning which also requires more inferring** than the first two methods. This learning requires the use of inductive inference, a form of invalid but useful inference.
- We use inductive learning of instances of examples of the concept.
- For example we learn the concepts of color or sweet taste after experiencing the sensations associated with several examples of colored objects or sweet foods.

Deductive learning

- Deductive learning is accomplished through a sequence of deductive inference steps using known facts.
- From the known facts, new facts or relationships are logically derived.
- Deductive learning usually requires more inference than the other methods.

Explanation-Based Learning

- Humans appear to learn quite a lot from one example.
- An Explanation-based Learning (EBL) system accepts an example (i.e. a training example) and explains what it learns from the example.
- The EBL system takes only the relevant aspects of the training. This explanation is translated into particular form that a problem solving program can understand. The explanation is generalized so that it can be used to solve other problems.

An EBL accepts 4 kinds of input:

- **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.

- **A operational criterion**

A description of which concepts are usable.

- **A domain theory**

A set of rules that describe relationships between objects and actions in a domain.

Learning may be classified into three categories:

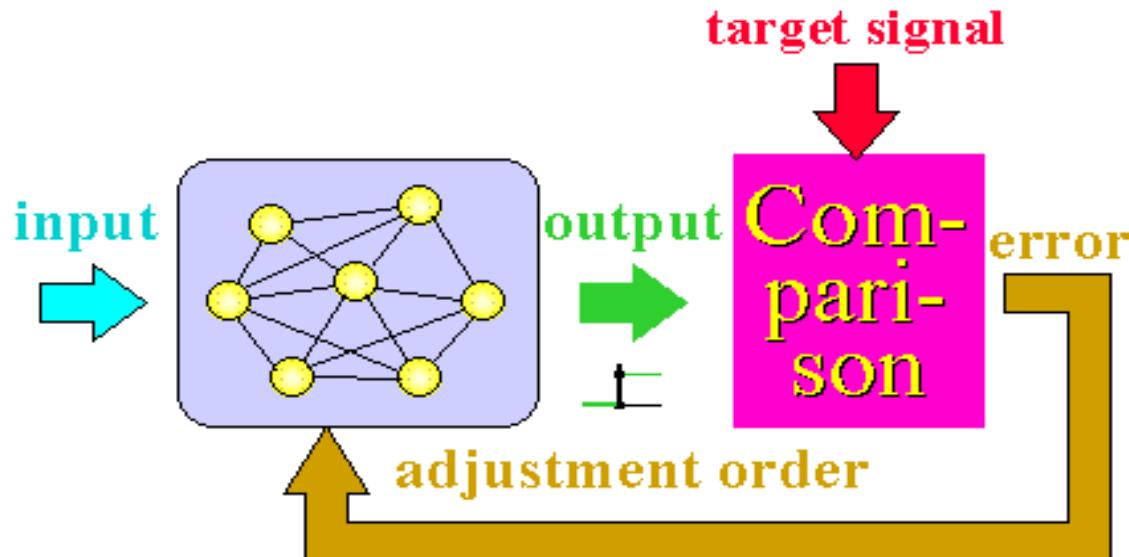
- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

- Supervised Learning:

- The network is presented with inputs together with the target (teacher signal) outputs.
- 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”.

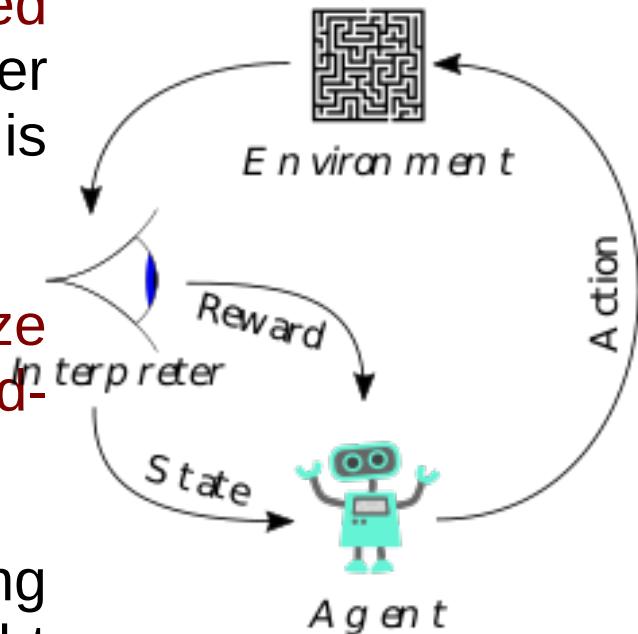
Error correction method

- › Neural networks are trained with this method in order to reduce the error (difference between the network's output and the desired output) to zero.



- Reinforcement Learning

- Reinforcement learning is similar to supervised learning in that some feedback is given, however instead of providing a target output a reward is given based on how well the system performed.
- The aim of reinforcement learning is to maximize the reward the system receives through trial-and-error.
- This paradigm relates strongly with how learning works in nature, for example an animal might remember the actions it's previously taken which helped it to find reward.



- Unsupervised Learning

- In unsupervised learning, there is no teacher (target signal) from outside and the network adjusts its weights in response to only the input patterns.
- A typical example of unsupervised learning is Hebbian learning.

Genetic Algorithm

- A genetic algorithm is a search heuristic that is inspired by Charles Darwin's theory of natural evolution.
- This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

The Algorithms

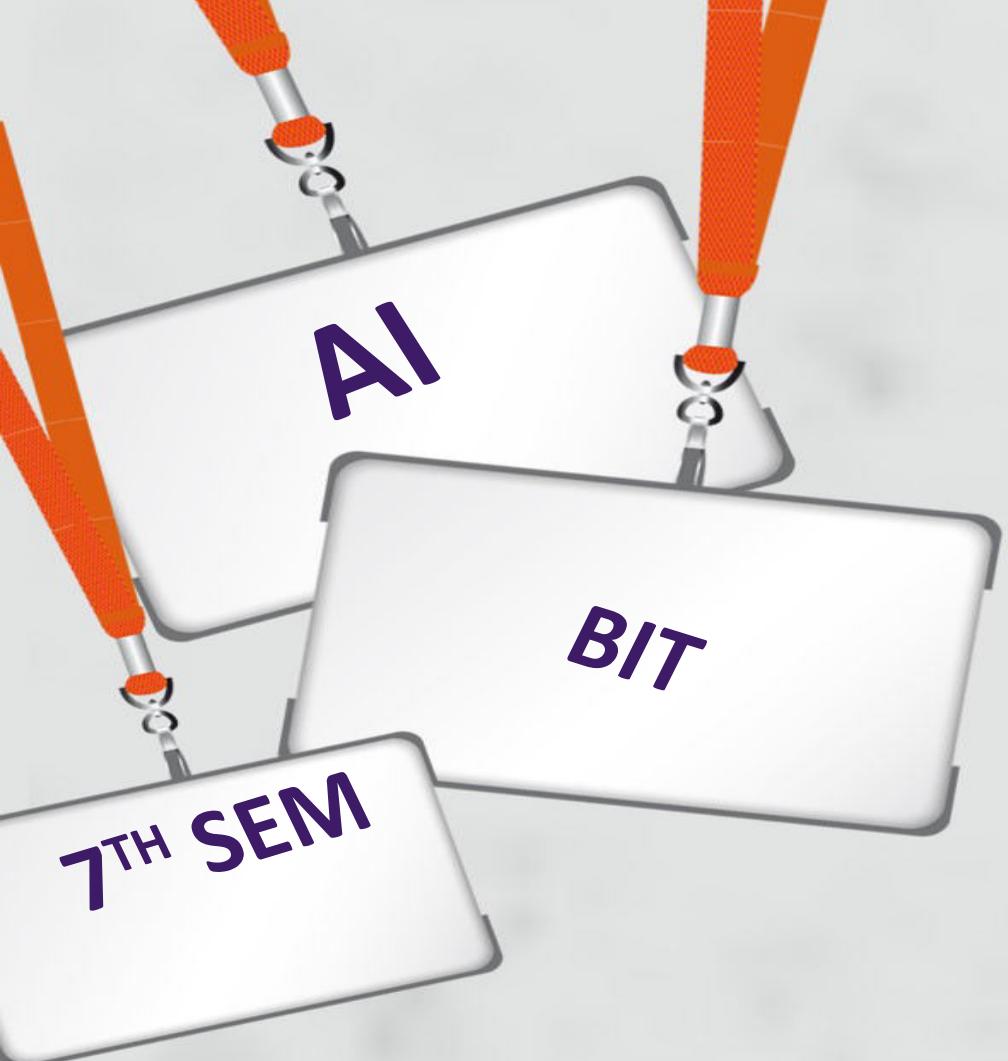
1. Randomly initialize population(t)
2. determine fitness of population(t)
3. Repeat
 - a. Select parents from population(t)
 - b. Perform crossover on parents creating population($t+1$)
 - c. Perform mutation of population($t+1$)
4. Determine fitness of population($t+1$) until best individual is good enough

Advantages of GAs

- Does not require any derivative information (which may not be available for many real-world problems).
- Is faster and more efficient as compared to the traditional methods.
- Has very good parallel capabilities.
- Optimizes both continuous and discrete functions and also multi-objective problems.
- Provides a list of “good” solutions and not just a single solution.
- Always gets an answer to the problem, which gets better over the time.
- Useful when the search space is very large and there are a large number of parameters involved.

Assignment

Define initial population, selection, fitness, mutation, cross over on the basis of genetic algorithm



Chapter 7

NEURAL NETWORK

A neuron is a cell in brain whose principle function is the collection, Processing, and dissemination of electrical signals. Brains Information processing capacity comes from networks of such neurons.

Due to this reason some earliest AI work aimed to create such artificial networks/ connectionism / Parallel distributed processing/ Neural computing

Neural Network

- An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information.
- It is composed of a large number of highly interconnected processing elements (neurones) working to solve specific problems.
- ANNs, like people, learn by example.
- An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

- A neural network is a system of hardware and/or software patterned after the operation of neurons in the human brain.
- A neural network usually involves a large number of processors operating in parallel and arranged in tiers. The first tier receives the raw input information -- analogous to optic nerves in human visual processing. Each successive tier receives the output from the tier preceding it, rather than from the raw input -- in the same way neurons further from the optic nerve receive signals from those closer to it. The last tier produces the output of the system.

Misconceptions

- Neural Networks are model of human brain
- Bigger size neural Networks works better
- Many Training data exists in neural networks

Why Neural Networks

- › **Pattern Extraction** : can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques.
- › **Expert Systems**: A trained neural network can be thought of as an expert in the category of information it has been given to analyze.
- › **Adaptive learning**: An ability to learn how to do tasks based on the data given for training or initial experience.
- › **Fault Tolerance via Redundant Information Coding**: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage

How Do Neural Networks Differ From Conventional Computing?

- ANNs are not sequential or necessarily deterministic.
- There are no complex central processors, rather there are many simple ones which generally do nothing more than take the weighted sum of their inputs from other processors.
- ANNs do not execute programmed instructions; they respond in parallel (either simulated or actual) to the pattern of inputs presented to it.
- There are also no separate memory addresses for storing data. Instead, information is contained in the overall activation 'state' of the network.

Units of Neural Network

- **Nodes (units)**

Nodes represent a cell of neural network.

- **Links**

Links are directed arrows that show propagation of information from one node to another node.

- **Activation**

Activations are inputs to or outputs from a unit.

- **Weight**

Each link has weight associated with it which determines strength and sign of the connection.

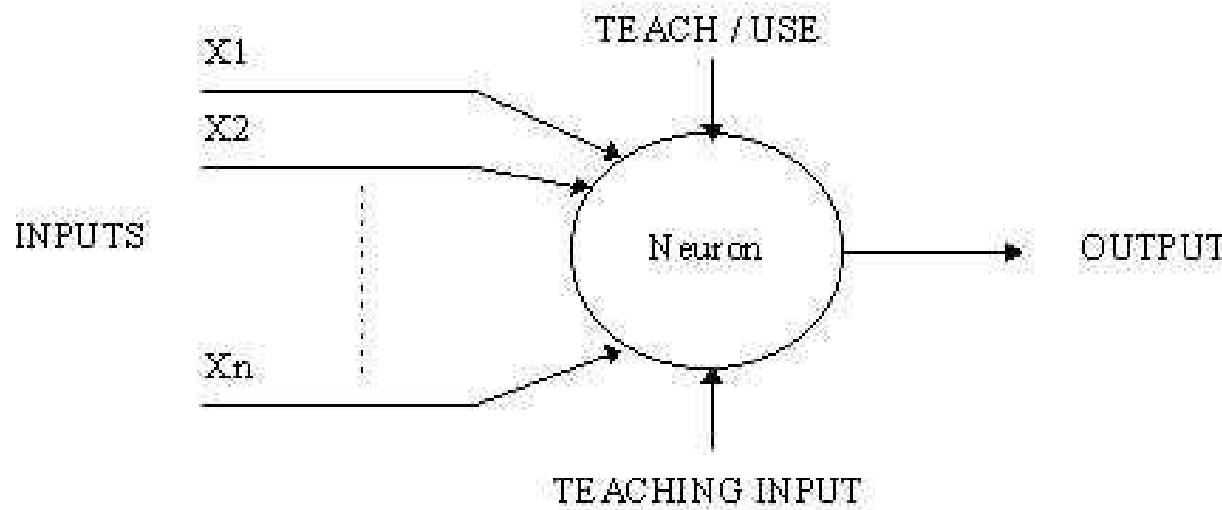
- **Activation function:**

A function which is used to derive output activation from the input activations to a given node is called activation function.

- **Bias:**

Bias is a feature of a statistical technique or of its results whereby the expected value of the results differs from the true underlying quantitative parameter being estimated.

A simple neuron



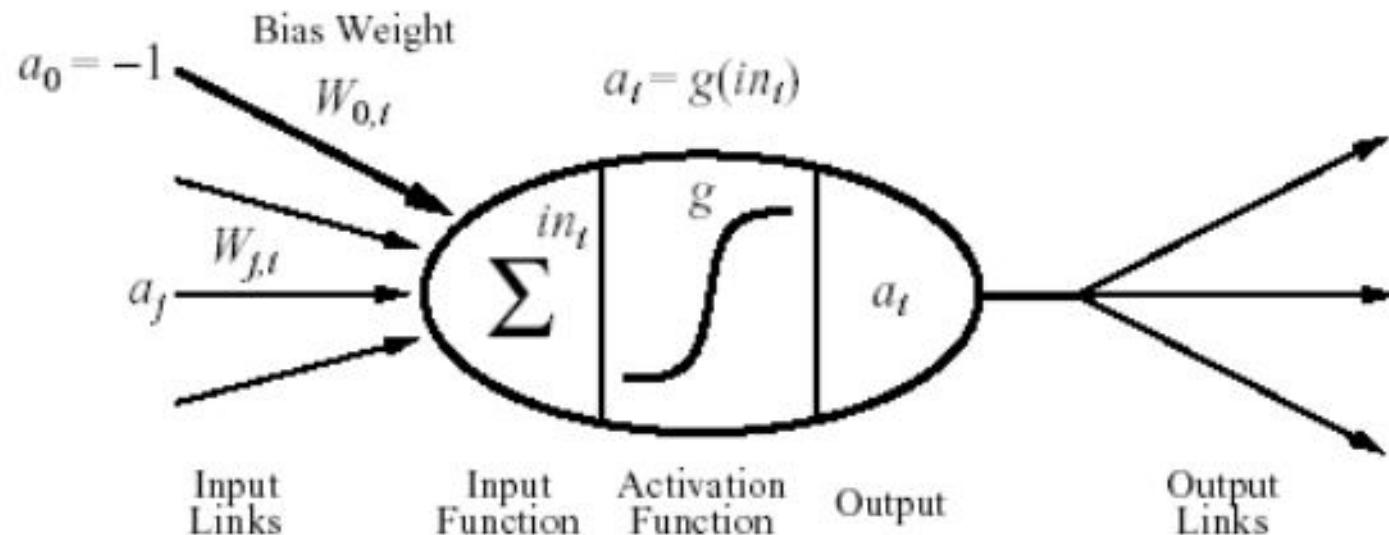
An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output.

If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

Simple Model of Neural Network

A simple mathematical model of neuron is devised by McCulloch and Pitt is given in the figure given below:

$$a_i \leftarrow g(in_i) = g \left(\sum w_{ij} a_j \right)$$



Transfer Function

The behavior of an ANN (Artificial Neural Network) depends on both the weights and the input-output function (transfer function) that is specified for the units.

This function typically falls into one of three categories:

- **Linear (or ramp)**

The output activity is proportional to the total weighted output

- **Threshold**

The output is set at one of two levels, depending on whether the total input is greater than or less than some threshold value

- **Sigmoid**

The output varies continuously but not linearly as the input changes. Sigmoid units bear a greater resemblance to real neurones than do linear or threshold units

The Learning Process

The memorization of patterns and the subsequent response of the network can be categorized into two general paradigms:

- **Associative mapping**

Associative mapping in which the network learns to produce a particular pattern on the set of input units whenever another particular pattern is applied on the set of input units.

- **Regularity Detection**

Regularity detection in which units learn to respond to particular properties of the input patterns

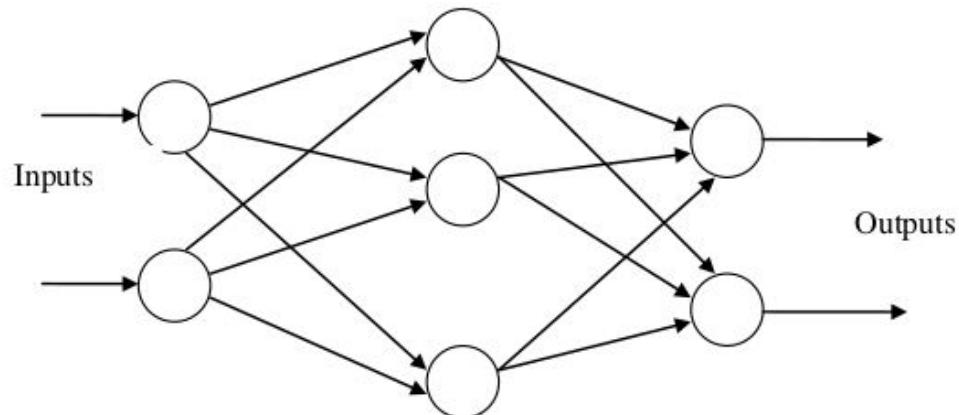
Perceptron

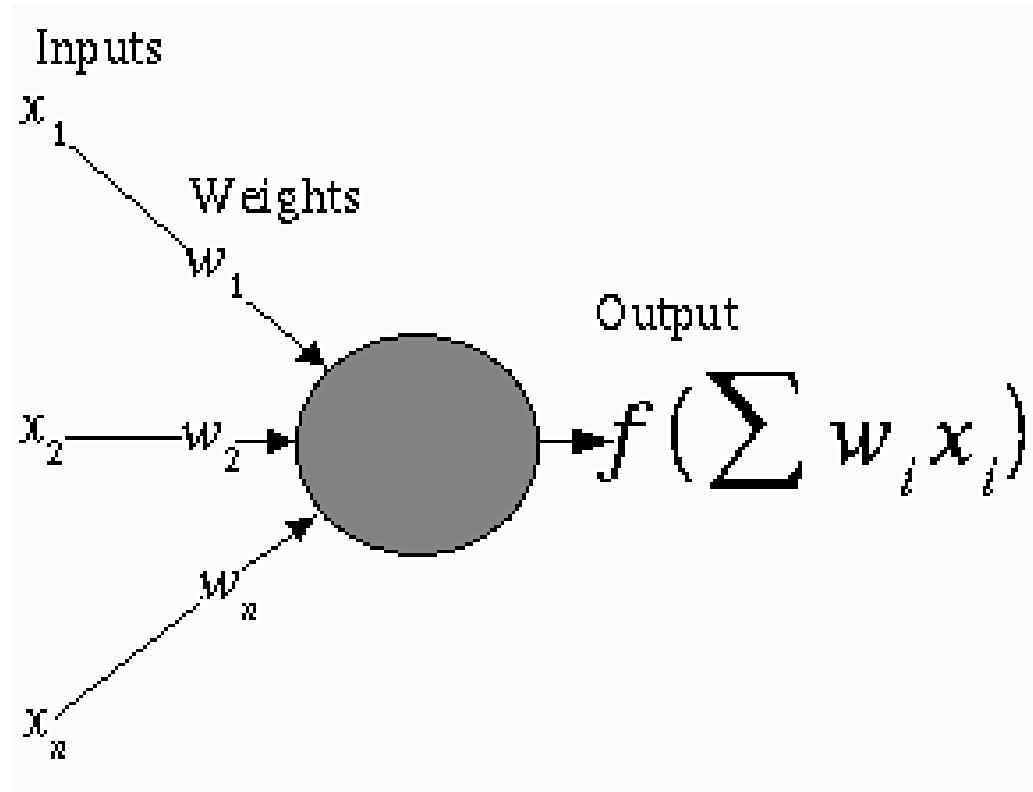
- The perceptron is an algorithm for supervised learning
- Functions that can decide whether an input, represented by a vector of numbers, belongs to some specific class or not

Architecture of Neural Networks

Feed-forward networks

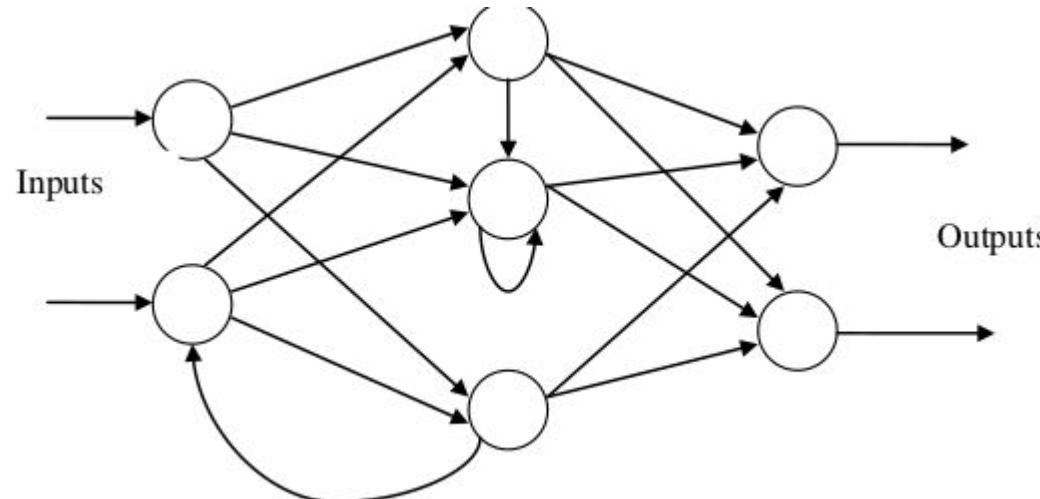
- Feed-forward ANNs allow signals to travel one way only; from input to output.
- There is no feedback (loops) i.e. the output of any layer does not affect that same layer.
- Feed forward ANNs tend to be straight forward networks that associate inputs with outputs.
- They are extensively used in pattern recognition.
- This type of organization is also referred to as bottom-up or top-down.





Feedback networks

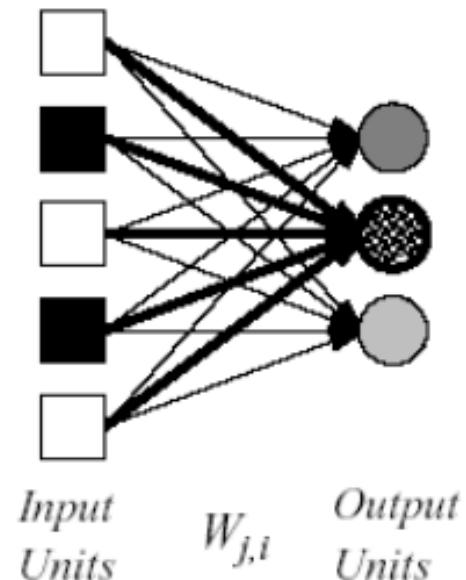
- › Feedback networks can have signals traveling in both directions by introducing loops in the network.
- › Feedback networks are very powerful and can get extremely complicated.
- › Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found.



Types of Feed Forward Neural Network

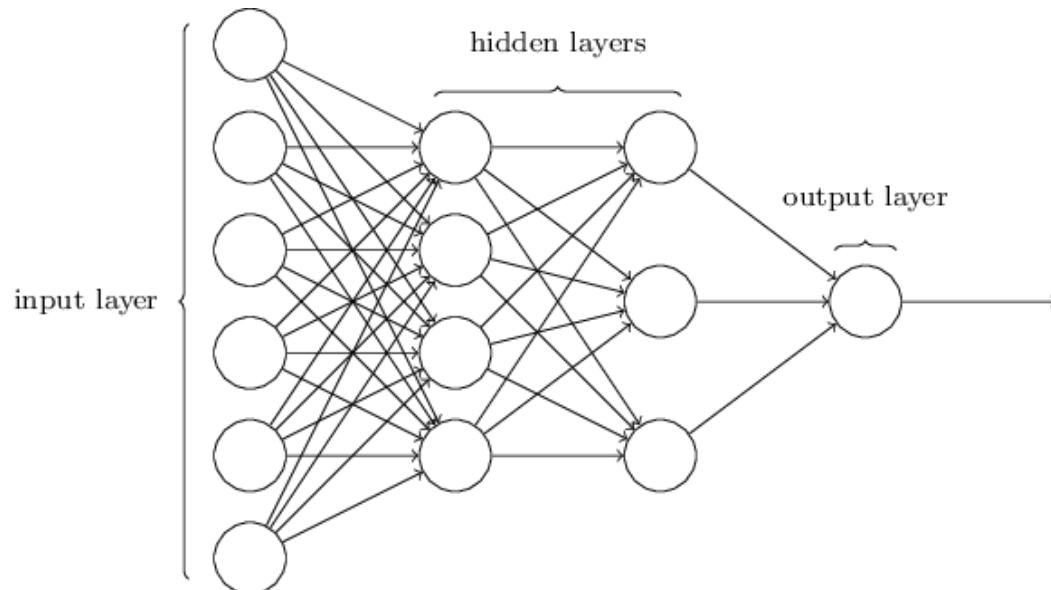
Single-layer neural networks (perceptrons)

- A neural network in which all the inputs connected directly to the outputs is called a single- layer neural network, or a perceptron network.
- Since each output unit is independent of the others each weight affects only one of the outputs.



Multilayer neural networks (perceptrons)

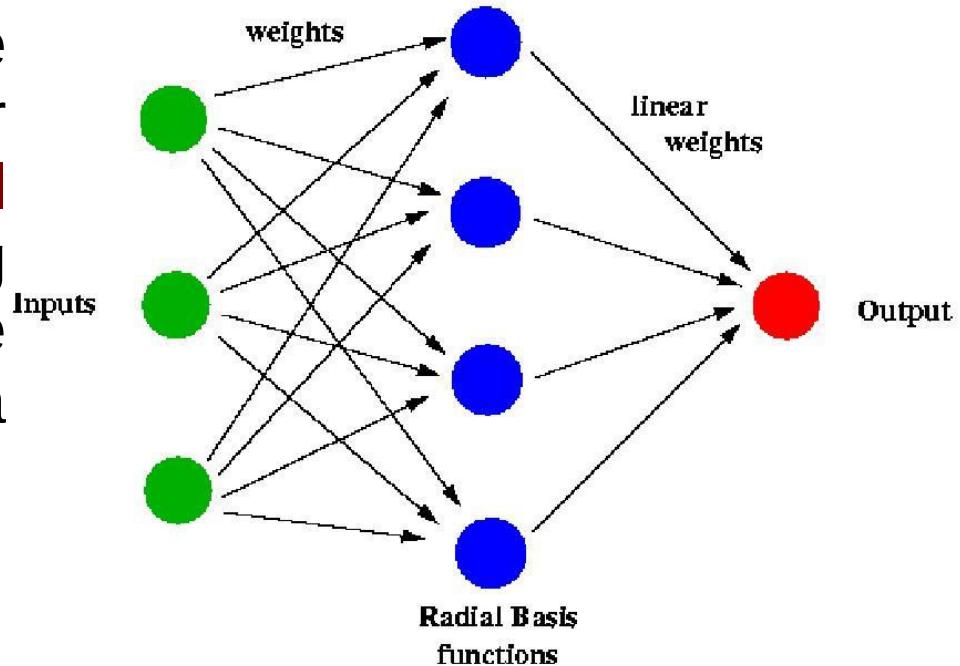
- The neural network which contains input layers, output layers and some hidden layers also is called multilayer neural network.
- The advantage of adding hidden layers is that it enlarges the space of hypothesis. Layers of the network are normally fully connected.



Other Special Types of Neural Networks

a. Radial Basis Function (RBF) Neural Network

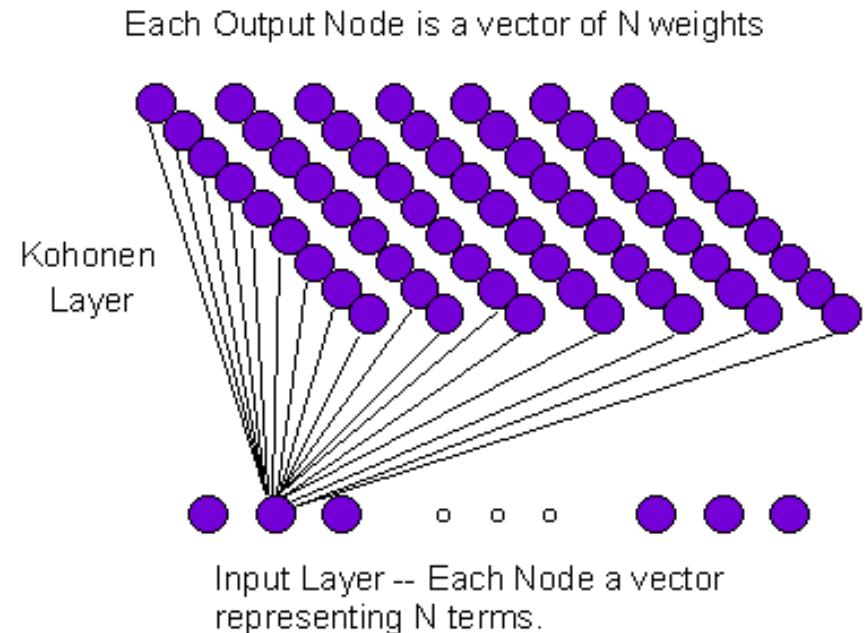
Radial basis functions are powerful techniques for **interpolation in multidimensional space.** (method of constructing new data points within the range of a discrete set of known data points)



Other Special Types of Neural Networks

b. Kohonen Self-organizing Neural Network

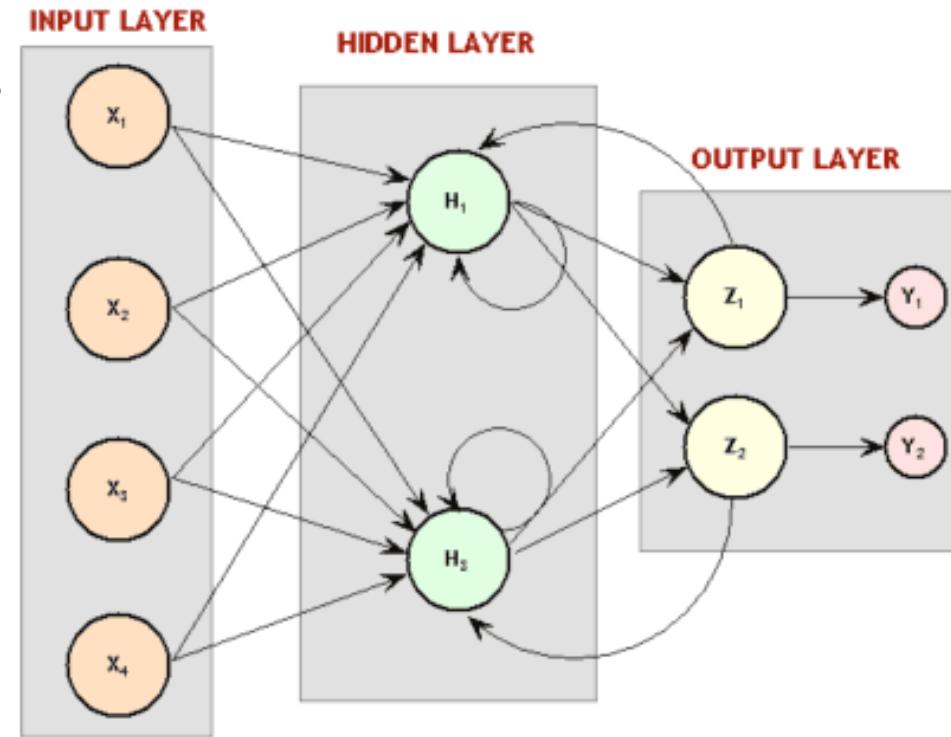
- › The self-organizing map (SOM) performs a form of unsupervised learning.
- › A set of artificial neurons learn to map points in an input space to coordinates in an output space



Other Special Types of Neural Networks

c. Recurrent Neural Networks

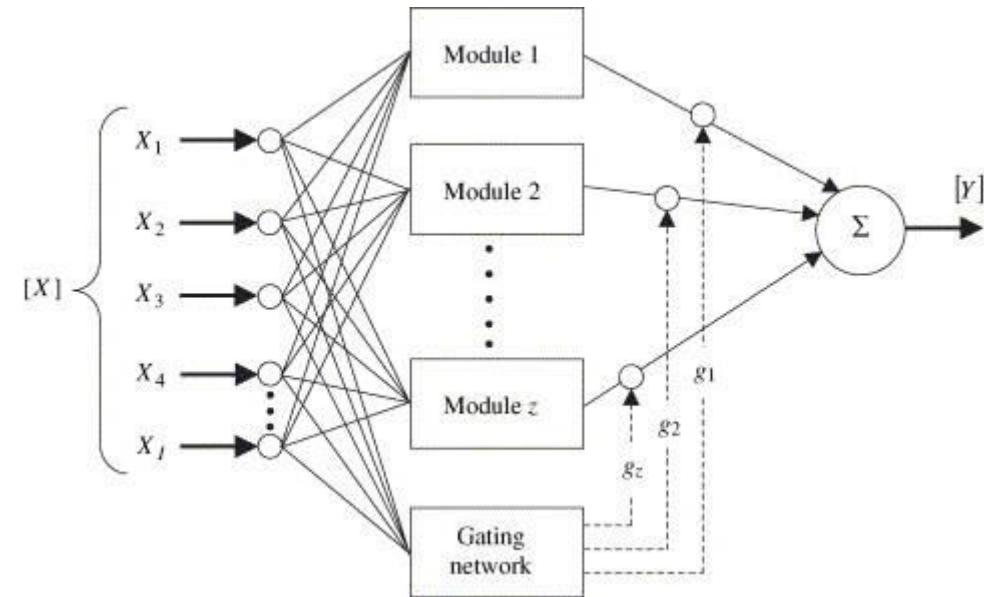
- Recurrent neural networks (RNNs) are models with bi-directional data flow.
- Recurrent neural networks can be used as **general sequence processors**



Other Special Types of Neural Networks

d. Modular Neural Network

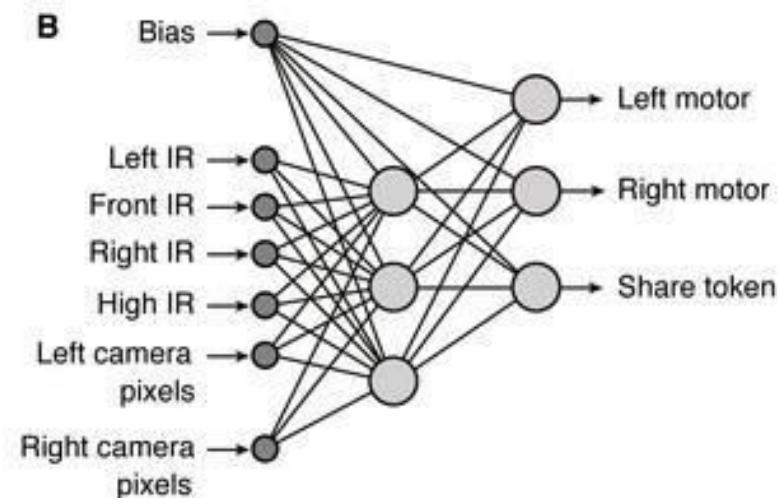
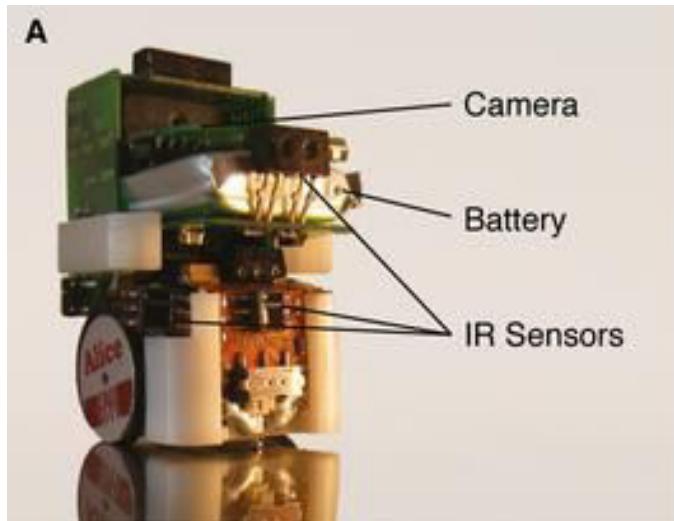
- › Biological studies have shown that the **human brain functions** not as a single massive network, but as a collection of small networks.
- › This realization gave birth to the concept of **modular neural networks**, in which several small networks cooperate or compete to solve problems.



Other Special Types of Neural Networks

e. Physical Neural Network

A physical neural network includes electrically adjustable resistance material to simulate artificial synapses



Neural Networks in Practice

Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

- sales forecasting
- industrial process control
- customer research
- data validation
- risk management
- target marketing

Hebbian Learning:

- Hebb's postulate of learning is :-

“When an axon of cell A is near enough to excite a cell B and repeatedly taking 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 cells firing B is increased”

- **The weight between two neurons increases if the two neurons activate simultaneously—and reduces if they activate separately.**
- Nodes that tend to be either both positive or both negative at the same time have strong positive weights, while those that tend to be opposite have strong negative weights.

Hebb's Algorithm:

Step 0: initialize all weights to 0

Step 1: Given a training input, s , with its target output, t , set the activations of the input units: $x_i = s_i$

Step 2: Set the activation of the output unit to the target value: $y = t$

Step 3: Adjust the weights: $w_i(\text{new}) = w_i(\text{old}) + x_i y$

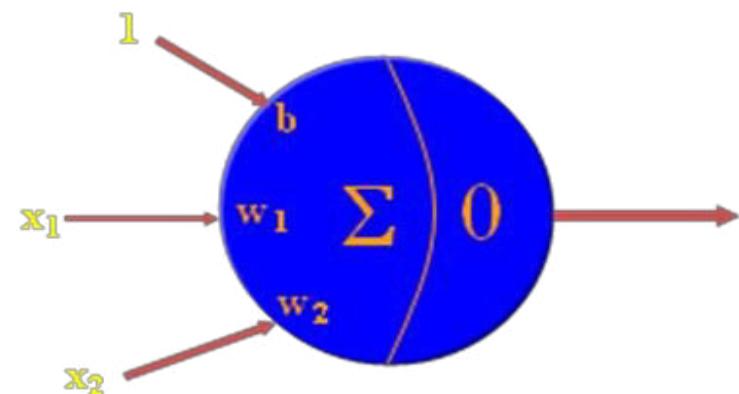
Step 4: Adjust the bias (just like the weights): $b(\text{new}) = b(\text{old}) + y$

(bias: the difference between the expectation of sample estimator and true value)

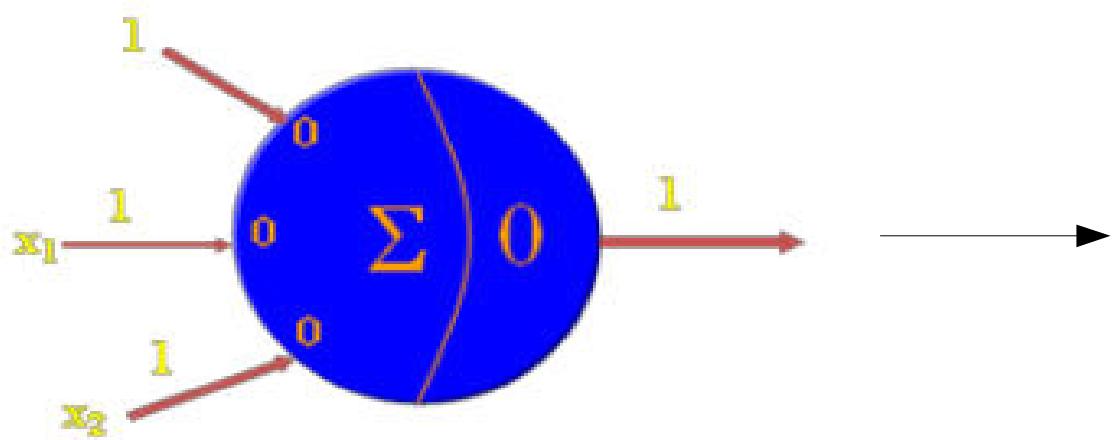
Construct a Hebb Net which performs like an AND function, that is, only when both features are “active” will the data be in the target class.

TRAINING SET (with the bias input always at 1):

x1	x2	bias	Target
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1



- › Initialize the weights to 0
- › Present the first input (1 1 1) with target of 1



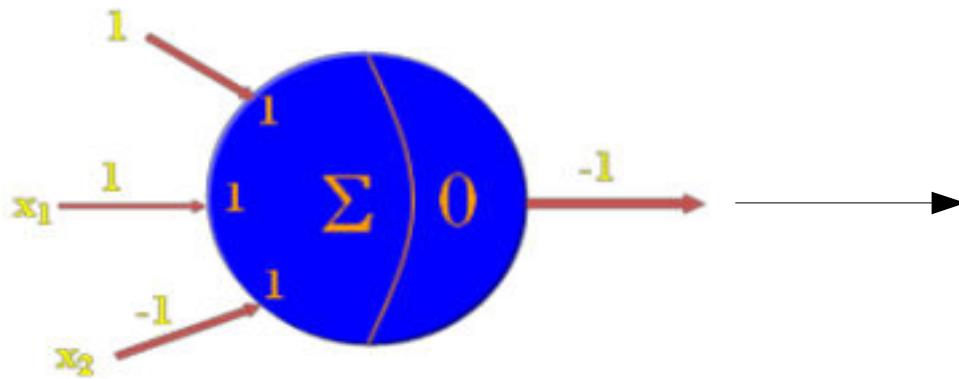
Update the weights:

$$w_1(\text{new}) = w_1(\text{old}) + x_1 t \\ = 0 + 1 = 1$$

$$w_2(\text{new}) = w_2(\text{old}) + x_2 t \\ = 0 + 1 = 1$$

$$b(\text{new}) = b(\text{old}) + t \\ = 0 + 1 = 1$$

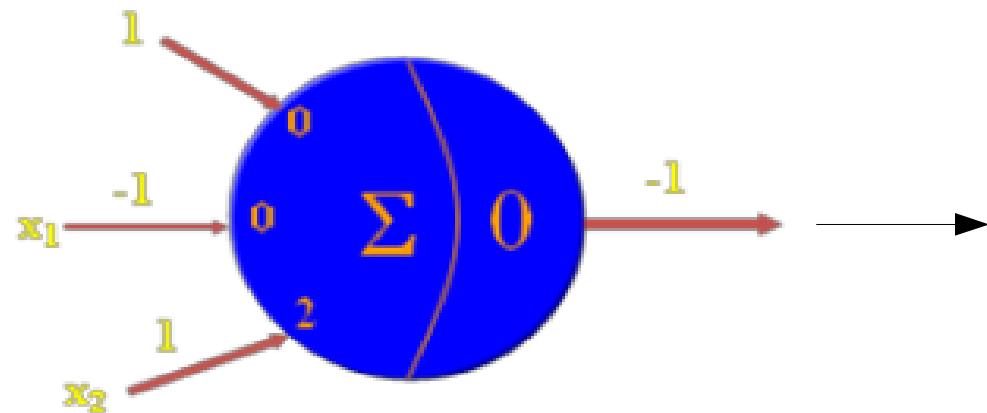
- Present the second input $(1 \ -1 \ 1)$ with target of -1



Update the weights:

$$\begin{aligned}
 w_1(\text{new}) &= w_1(\text{old}) + x_1 t \\
 &= 1 + 1(-1) = 0 \\
 w_2(\text{new}) &= w_2(\text{old}) + x_2 t \\
 &= 1 + (-1)(-1) = 2 \\
 b(\text{new}) &= b(\text{old}) + t \\
 &= 1 + (-1) = 0
 \end{aligned}$$

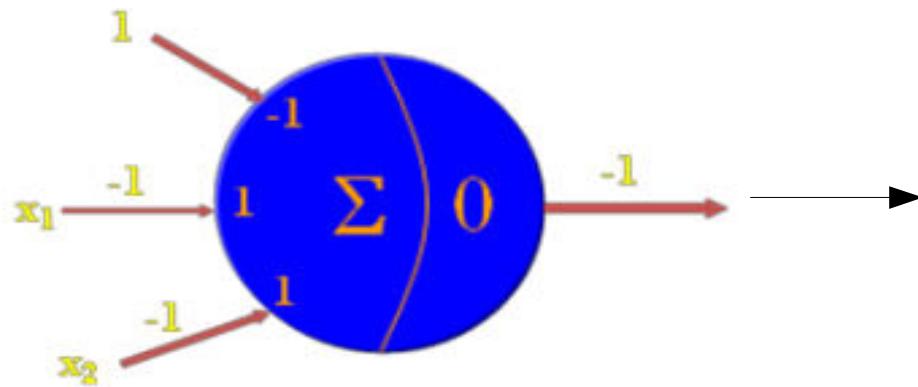
- Present the third input (-1 1 1) with target of -1



Update the weights:

$$\begin{aligned}
 w_1(\text{new}) &= w_1(\text{old}) + x_1 t \\
 &= 0 + (-1)(-1) = 1 \\
 w_2(\text{new}) &= w_2(\text{old}) + x_2 t \\
 &= 2 + 1(-1) = 1 \\
 b(\text{new}) &= b(\text{old}) + t \\
 &= 0 + (-1) = -1
 \end{aligned}$$

- Present the Fourth input $(-1 \ -1 \ 1)$ with target of -1



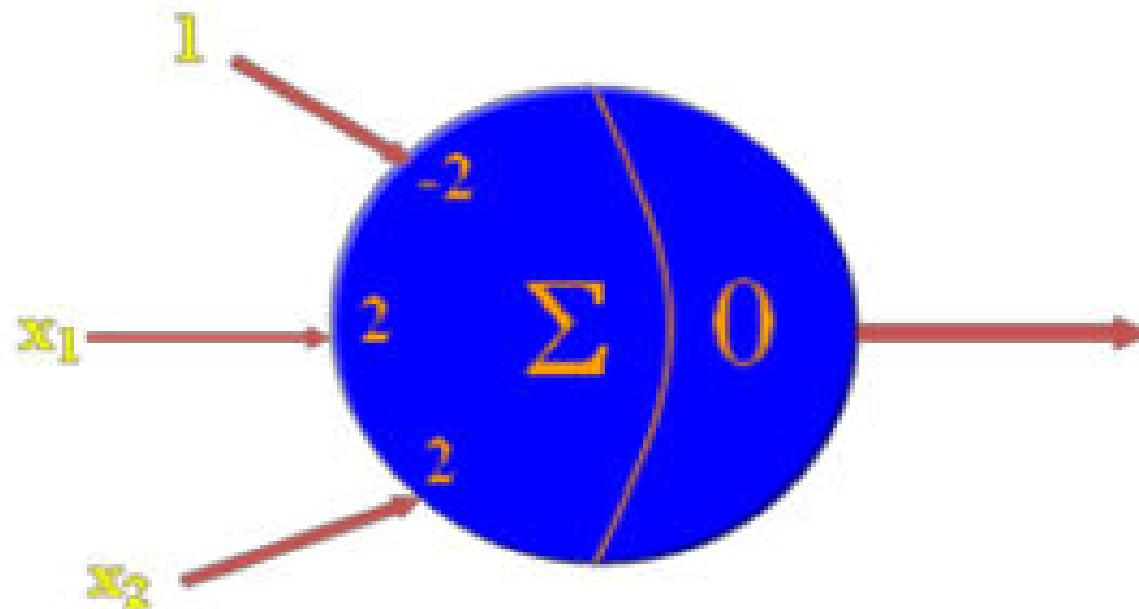
Update the weights:

$$\begin{aligned} w_1(\text{new}) &= w_1(\text{old}) + x_1 t \\ &= 1 + (-1)(-1) = 2 \end{aligned}$$

$$\begin{aligned} w_2(\text{new}) &= w_2(\text{old}) + x_2 t \\ &= 1 + (-1)(-1) = 2 \end{aligned}$$

$$\begin{aligned} b(\text{new}) &= b(\text{old}) + t \\ &= -1 + (-1) = -2 \end{aligned}$$

The Final Neuron



Perceptron Learning Theory

- Perceptron (an element of artificial Neural Network consisting of one or more layer of artificial neuron)
- The term "Perceptrons" was coined by Frank RosenBlatt in 1962 and is used to describe the connection of simple neurons into networks.
- These networks are simplified versions of the real nervous system where some properties are exaggerated and others are ignored.

- › We need to train the neural network so they can do things that are useful.
- › To do this we must allow the neuron to learn from its mistakes.
- › There is in fact a learning paradigm that achieves this, it is known as supervised learning and works in the following manner.
 1. Set the weight and thresholds of the neuron to random values.
 2. Present an input.
 3. Calculate the output of the neuron.
 4. Alter the weights to reinforce correct decisions and discourage wrong decisions, hence reducing the error. So for the network to learn we shall increase the weights on the active inputs when we want the output to be active, and to decrease them when we want the output to be inactive.
 5. Now present the next input and repeat steps 3.

Algorithm:

- i. Initialize weights and threshold.

Set $w_i(t)$, ($0 \leq i \leq n$), to be the weight i at time t , and ϕ to be the threshold value in the output node. Set w_0 to be $-\phi$, the bias, and x_0 to be always 1.

Set $w_i(0)$ to small random values, thus initializing the weights and threshold.

- ii. Present input and desired output

Present input $x_0, x_1, x_2, \dots, x_n$ and desired output $d(t)$

- iii. Calculate the actual output

$$y(t) = g [w_0(t)x_0(t) + w_1(t)x_1(t) + \dots + w_n(t)x_n(t)]$$

- iv. Adapts weights

$w_i(t+1) = w_i(t) + \alpha[d(t) - y(t)]x_i(t)$, where $0 \leq \alpha \leq 1$ (learning rate) is a positive gain function that controls the adaption rate.

Steps iii. and iv. are repeated until the iteration error is less than a user-specified error threshold or a predetermined number of iterations have been completed.

Delta Rule/ Widrow – Hoff Rule

- Called as the Least Mean Square (LMS) method Delta rule is one of the most commonly used learning rules.
- It is a special case of the more general back propagation algorithm.
- For a given input vector, the output vector is compared to the correct answer. If the difference is zero, no learning takes place; otherwise, the weights are adjusted to reduce this difference.

For a neuron j with activation function $g(x)$, the delta rule for j 's i^{th} weight w_{ji} is given by

$$\Delta w_{ji} = \alpha(t_j - y_j)g'(h_j)x_i$$

where

α is a small constant called *learning rate*

$g(x)$ is the neuron's activation function

t_j is the target output

h_j is the weighted sum of the neuron's inputs

y_j is the actual output

x_i is the i^{th} input.

It holds that $h_j = \sum x_i w_{ji}$ and $y_j = g(h_j)$.

Back propagation

- It is a supervised learning method, and is an implementation of the Delta rule.
- It requires a teacher that knows, or can calculate, the desired output for any given input. It is most useful for feed-forward networks.
- The term is an abbreviation for "backwards propagation of errors".
- Back propagation networks are necessarily multilayer perceptrons (usually with one input, one hidden, and one output layer).

As the algorithm's name implies, the errors (and therefore the learning) propagate backwards from the output nodes to the inner nodes. So technically speaking, back propagation is used to calculate the gradient of the error of the network with respect to the network's modifiable weights.

Back propagation technique

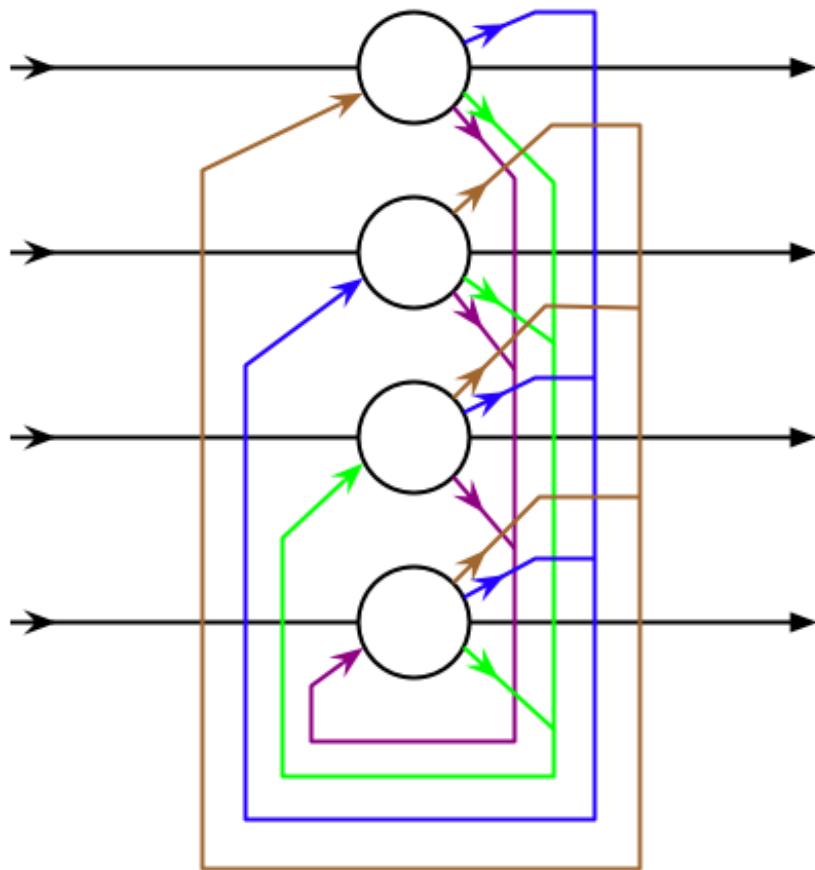
- Randomly choose the initial weights
- While error is too large
 - For each training pattern (presented in random order)
 - ✗ Apply the inputs to the network
 - ✗ Calculate the output for every neuron from the input layer, through the hidden layer(s), to the output layer
 - ✗ Calculate the error at the outputs
 - ✗ Use the output error to compute error signals for pre-output layers
 - ✗ Use the error signals to compute weight adjustments
 - ✗ Apply the weight adjustments
 - Periodically evaluate the network performance

- The delta rule is an update rule for single layer perceptrons. It makes use of gradient descent.
- Back propagation is an efficient implementation of gradient descent for multilayer perceptrons.

Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function

Hopfield Network

- A Hopfield network is a form of **recurrent artificial neural network** popularized by John Hopfield in 1982, but described earlier by Little in 1974.
- The **units in Hopfield nets are binary threshold units**, i.e. the units only take on two different values for their states and the value is determined by whether or not the units' input exceeds their threshold. Hopfield nets normally have units that take on values of 1 or -1



Updating one unit (node in the graph simulating the artificial neuron) in the Hopfield network is performed using the following rule:

$$s_i \leftarrow \begin{cases} +1 & \text{if } \sum_j w_{ij} s_j \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases}$$

Where

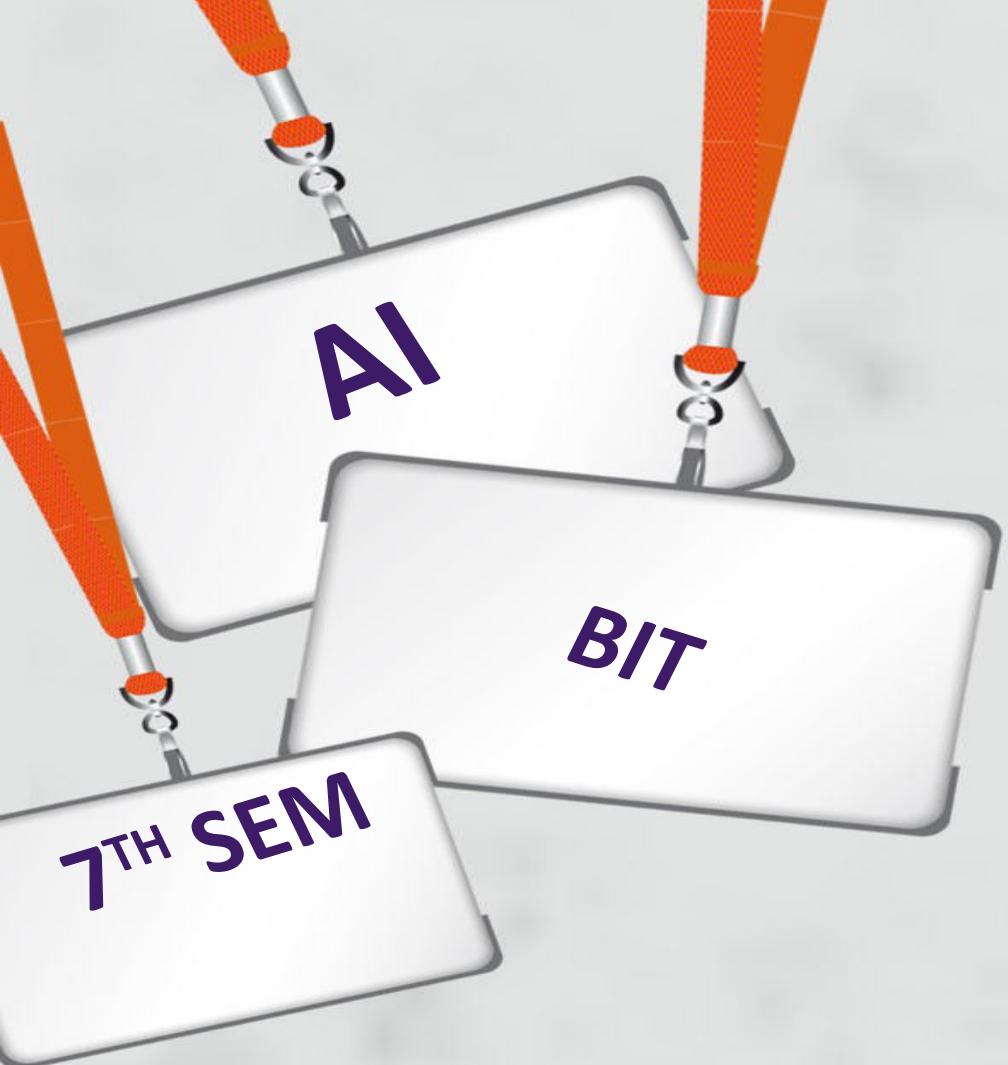
s_i = is the state of unit j

w_{ij} = is the strength of the connection weight from unit j to unit i

θ_i = is the threshold of unit i .

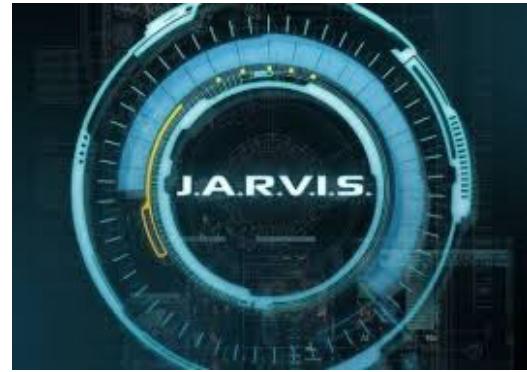
Assignment

Write about Boltzmann Machines



Chapter 8

EXPERT SYSTEM



It recognize voice commands to do something, even for locking door and turn off the lamp.

MYCIN would attempt to diagnose patients based on reported symptoms and medical test results.

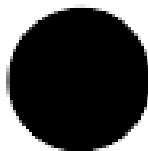
The program could request further information concerning the patient, as well as suggest additional laboratory tests, to arrive at a probable diagnosis, after which it would recommend a course of treatment.

If requested, MYCIN would explain the reasoning that led to its diagnosis and recommendation.

CALEX is a blackboard based integrated system for agricultural management, developed at University of California.

CALEX can be used by growers, pest control advisors, consultants and other managers.

DENDRAL's primary aim was to help organic chemists in identifying unknown organic molecules, by analyzing their mass spectra and using knowledge of chemistry.



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

Characteristics of Expert Systems

- High performance
- Reliable
- Highly responsive

Capabilities of Expert Systems

Capabilities	Incapabilities
Advising	Possessing human capabilities
Instructing and assisting human in decision making	Substituting human decision makers
Deriving a solution	Producing accurate output for inadequate knowledge base
Predicting results	Refining their own knowledge

Human Expert Vs Expert System

Factor	Human Expert	Expert System
Time (can be obtained)	Working days only	Anytime
Geography	Local	Anywhere
Safety	Cannot be replaced	Can be replaced
Damages	Yes	No
Speed and Efficiency	Changes	Consistent
Cost	High	Intermediate

Components of Expert Systems

The components of ES include –

- Knowledge Base
- Inference Engine
- User Interface

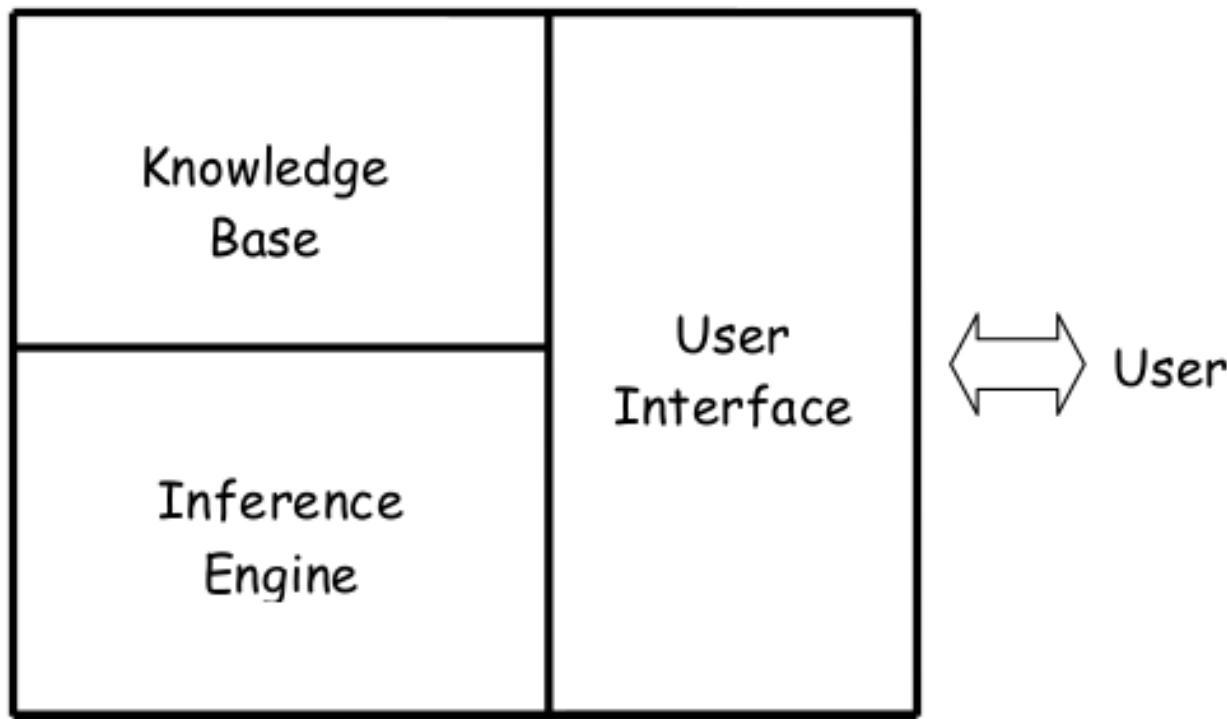


Fig: Block Diagram of expert system

Knowledge Base

- Knowledge is required to exhibit intelligence.
- The success of any ES depends upon the collection of highly accurate and precise knowledge.
- **Components of Knowledge Base**
 - The knowledge base of an ES is a store of both, factual and heuristic knowledge.
 - ➔ Factual Knowledge – It is the information widely accepted by the Knowledge Engineers and scholars in the task domain.
 - ➔ Heuristic Knowledge – It is about practice, accurate judgment, one's ability of evaluation, and guessing.

Inference Engine

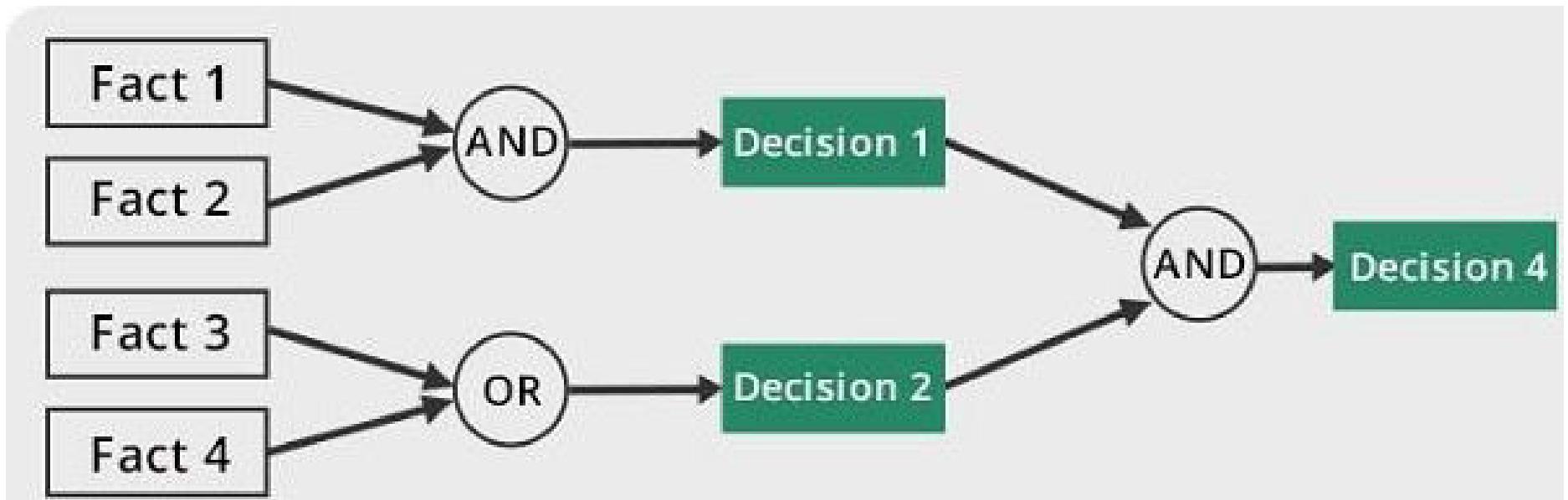
In case of knowledge-based ES, the Inference Engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution.

In case of rule based ES, it –

- Applies rules repeatedly to the facts, which are obtained from earlier rule application.
- Adds new knowledge into the knowledge base if required.
- Resolves rules conflict when multiple rules are applicable to a particular case.
- To recommend a solution, the Inference Engine uses the following strategies –
 - Forward Chaining
 - Backward Chaining

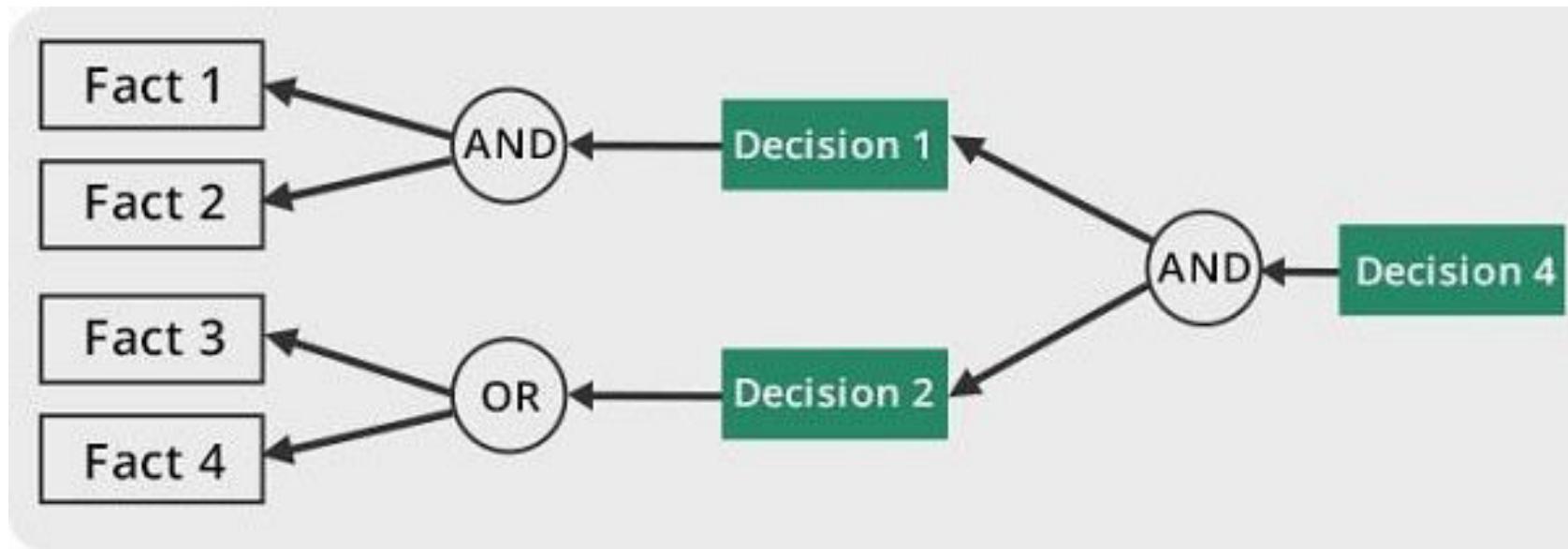
Forward Chaining

It is a strategy of an expert system to answer the question, “What can happen next?”



Backward Chaining

With this strategy, an expert system finds out the answer to the question, “Why this happened?”



User Interface

- User interface provides interaction between user of the ES and the ES itself.
- It is generally Natural Language Processing so as to be used by the user who is well-verses in the task domain.
- The user of the ES need not be necessarily an expert in Artificial Intelligence.

Expert Systems Limitations

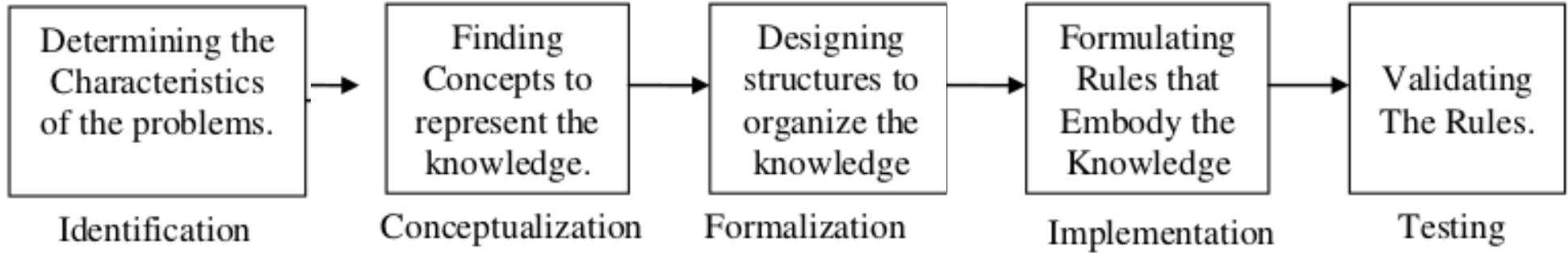
- Limitations of the technology
- Difficult knowledge acquisition
- ES are difficult to maintain
- High development costs

Applications of Expert System

Application	Description
Design Domain	Camera lens design, automobile design.
Medical Domain	Diagnosis Systems to deduce cause of disease from observed data, conduction medical operations on humans.
Monitoring Systems	Comparing data continuously with observed system or with prescribed behavior such as leakage monitoring in long petroleum pipeline.
Process Control Systems	Controlling a physical process based on monitoring.
Knowledge Domain	Finding out faults in vehicles, computers. Detection of possible fraud, suspicious transactions, stock market trading, Airline scheduling, cargo scheduling.
Finance/Commerce	

Stages of Expert System Development:

- An expert system typically is developed and refined over a period of several years.
- We can divide the process of expert system development into five distinct stages. In practice, it may not be possible to break down the expert system development cycle precisely.
- However, an examination of these five stages may serve to provide us with some insight into the ways in which expert systems are developed.



Identification

- The problem must be suitable for an expert system to solve it.
- Find the experts in task domain for the ES project.
- Establish cost-effectiveness of the system.

Conceptualization

- Identify the ES Technology
- Know and establish the degree of integration with the other systems and databases.
- Realize how the concepts can represent the domain knowledge best.

Formalization

- The various techniques of knowledge representation and heuristic search used in expert systems.
- The expert system tools that can greatly assist the development process.
- Other expert systems that may solve similar problems and thus may be adequate to the problem at hand.

Implementation

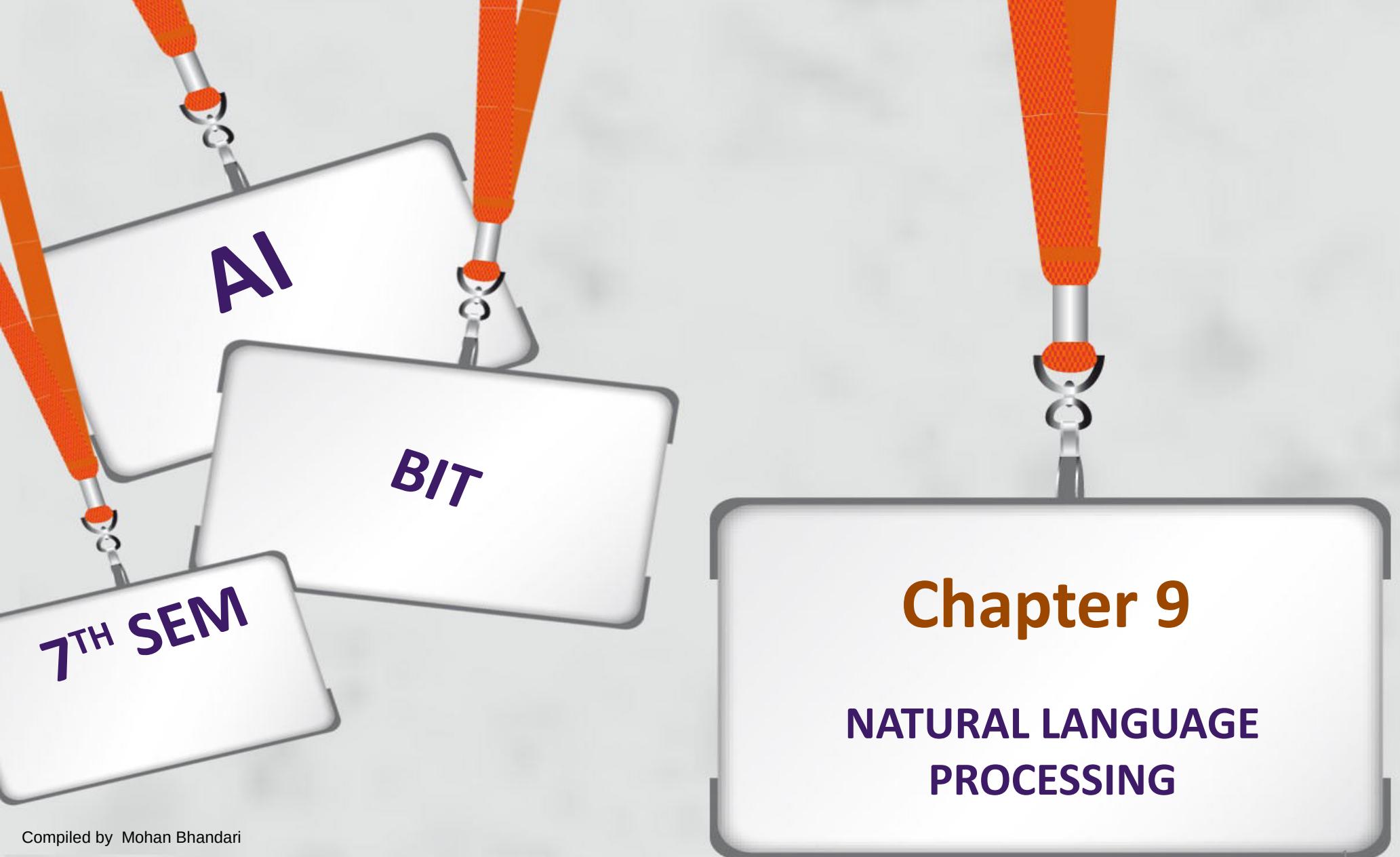
During the implementation stage, the formalized concepts are programmed onto the computer that has been chosen for system development, using the predetermined techniques and tools to implement a first pass prototype of the expert system.

Testing

- Testing provides opportunities to identify the weakness in the structure and implementation of the system and to make the appropriate corrections.
- Depending on the types of problems encountered, the testing procedure may indicate that the system was

Benefits of Expert Systems

- **Availability** – They are easily available due to mass production of software.
- **Speed** – They offer great speed. They reduce the amount of work an individual puts in.
- **Less Error Rate** – Error rate is low as compared to human errors.
- **Reducing Risk** – They can work in the environment dangerous to humans.
- **Steady response** – They work steadily without getting motional, tensed or fatigued.



Natural Language

- In philosophy of language, a natural language or ordinary language is any language that has evolved naturally in humans through use and repetition without conscious planning or premeditation.
- Natural languages can take different forms, such as speech or signing.
- They are distinguished from constructed and formal languages such as those used to program computers or to study logic.
- In computing, natural language refers to a human language such as English, Russian, German, or Japanese as distinct from the typically artificial command or programming language with which one usually talks to a computer.

- › Humans perceive and communicate through their five basic senses of sight, hearing, touch, smell and taste, and their ability to generate meaningful utterances.
- › Developing programs that understand a natural language is a difficult problem.

- Developing programs to understand natural language is important in AI because a natural form of communication with systems is essential.
- AI programs must be able to communicate with their human counterparts in a natural way, and natural language is one of the most important mediums for that purpose.
- So, **Natural Language Processing (NLP)** is the field that deals with the computer processing of natural languages, mainly evolved by people working in the field of Artificial Intelligence.

What makes NLP challenging?

Natural Language contain **infinity** of different sentences. No matter how many sentences a person has heard or seen, **new ones can always be produced**. Also, there is much **ambiguity** in a natural language. Many words have several meanings and sentences can have different meanings in different contexts. This makes the creation of programs that understand a natural language, one of the most challenging tasks in AI.

Some of the better-known applications of NLP include:

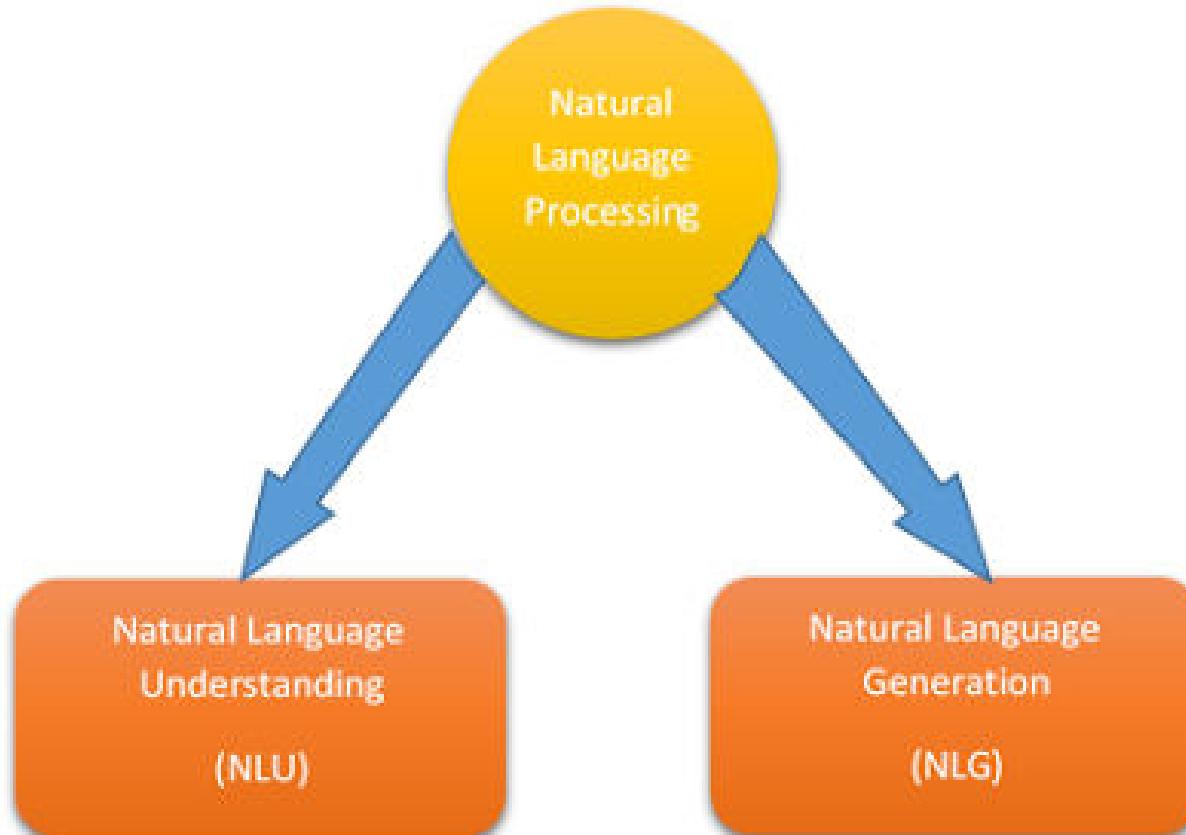
Voice recognition software: Translates speech into input for word processors or other applications;

Text-to-speech synthesizers: Read text aloud for users such as the hearing-impaired;

Grammar and style checkers: Analyze text in an attempt to highlight errors of grammar or usage;

Machine translation: Systems which automatically render a document such as a web page in another language.

Components in NLP



Natural Language Generation

- NLG also referred to as text generation, is a subfield of natural language processing.
- Natural Language Generation (NLG) is the natural language processing task of generating natural language from a machine representation system such as a knowledge base or a logical form.
- NLG system is like a translator that converts a computer based representation into a natural language representation.

NLG includes

Text planning : - Retrieving the relevant content from database. Here database can be includes vocabulary, sentences, knowledge, sample data and many more.

Sentence planning : - We get our content using text planning now next step to do is choosing required words and forming meaningful sentence setting the words in right grammatical way.

Text realization :- We have all the thing need to create actual text in humans language.

Natural Language Understanding

- It deals with machine reading comprehension : ability to read text, to do process on it, and understand its meaning.
- It is involved with mapping the given inputs in useful representations and analyzing the different aspects of the language.
- Many words have several meaning such as can, bear, fly, bank etc, and sentences have different meanings in different contexts.
- Understanding the language is not only the transmission of words. It also requires inference about the speakers' goal, knowledge as well as the context of the interaction

Difficulties in NLU

- NL has an extremely rich form and structure.
- It is very ambiguous.
- Different levels of ambiguity -
 - ✗ **Lexical ambiguity** – It is at very primitive level such as word-level.
 - ✗ **Syntax Level ambiguity** – A sentence can be parsed in different ways.
 - ✗ **Referential ambiguity** – Referring to something using pronouns.
 - ✗ One input can mean different meanings.
 - ✗ Many inputs can mean the same thing.

Levels of knowledge used in Language Understanding

- › A language understanding knowledge must have considerable knowledge about the structures of the language including what the words are and how they combine into phrases and sentences.
- › It must also know the meanings of the words and how they contribute to the meanings of the sentence and to the context within which they are being used.

The component forms of knowledge needed for an understanding of natural languages are sometimes classified according to the following levels.

- **Phonological:**

Relates sound to the words we recognize. A phoneme is the smallest unit of the sound.

- **Morphological:**

This is lexical (Linguistic unit) knowledge which relates to the word construction from basic units called morphemes. A morpheme is the smallest unit of meaning.

Eg:- friend + ly = friendly.

➤ **Syntactic:**

This knowledge **relates to how words are put together or structured** together to form grammatically correct sentences in the language.

➤ **Semantic :**

This knowledge is **concerned with the meanings of words and phrases** and how they combine to form sentence meaning

➤ **Pragmatic:**

This is high – level knowledge which **relates to the use of sentences in different contexts** and how the context affects the meaning of the Sentence.

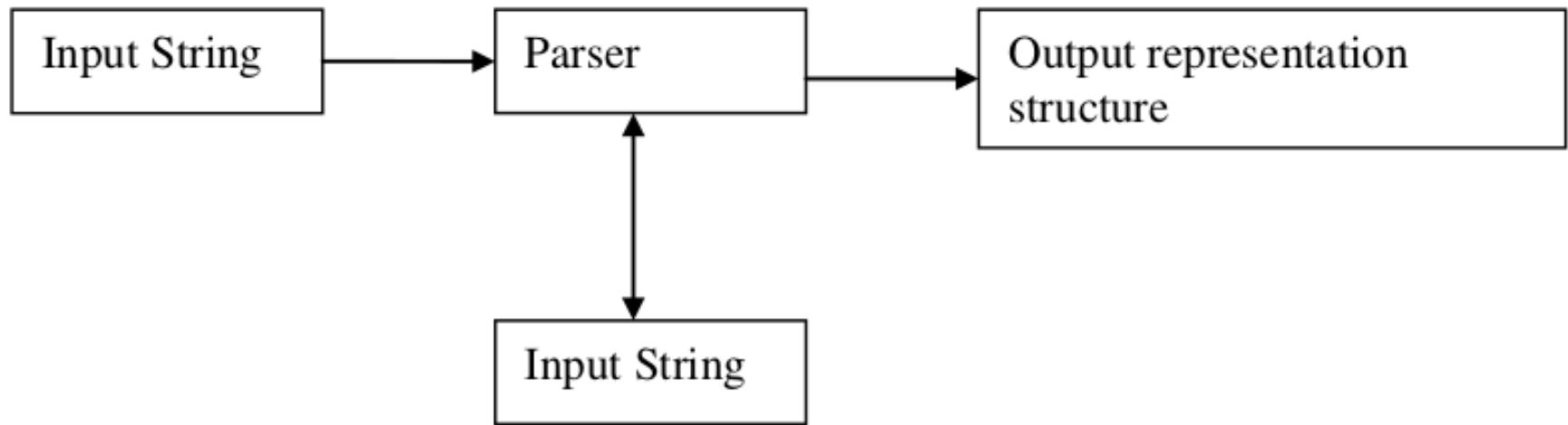
➤ **World :**

Includes the knowledge of the physical world, the world of human social interaction, and the roles of goals and intentions in communication.

Basic Parsing Techniques

- › Before the meaning of a sentence can be determined, the meanings of its constituent parts must be established.
- › This requires knowledge of the structure of the sentence, the meaning of the individual words and how the words modify each other.
- › The process of determining the syntactical structure of a sentence is known as parsing

- Parsing is the process of analyzing a sentence by taking it apart word – by – word and determining its structure from its constituent parts and sub parts.
- The structure of a sentence can be represented with a syntactic tree. When given an input string, the lexical parts or terms (root words), must first be identified by type and then the role they play in a sentence must be determined.
- These parts can be combined successively into larger units until a complete tree has been computed.



A. Top-Down parsing:

- ✗ Using Top-Down technique, parser searches for a parse tree by trying to build from the **root node S down to the leaves**.
- ✗ The algorithm starts by assuming the input can be derived by the designated start symbol S.
- ✗ The next step is to find the tops of all the trees which can start with S, by looking on the grammar rules with S on left hand side, all the possible trees are generated.
- ✗ Top-Down parsing is goal directed search technique.

B. Bottom – Up

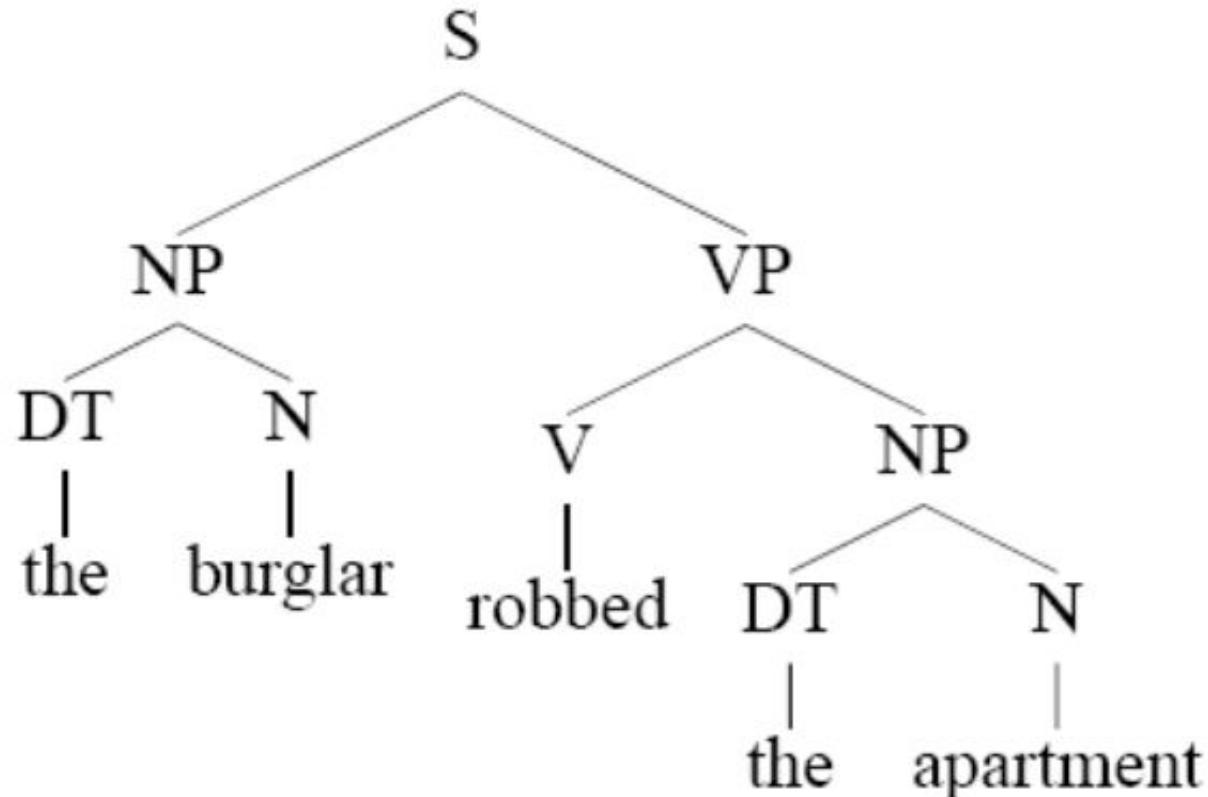
- › Bottom – Up parsing starts with the words of input and tries to build trees from the words up, again by applying rules from the grammar one at a time.
- › The parse is successful if the parse succeeds in building a tree rooted in the start symbol S that covers all of the input.
- › Bottom up parsing is a data directed search.

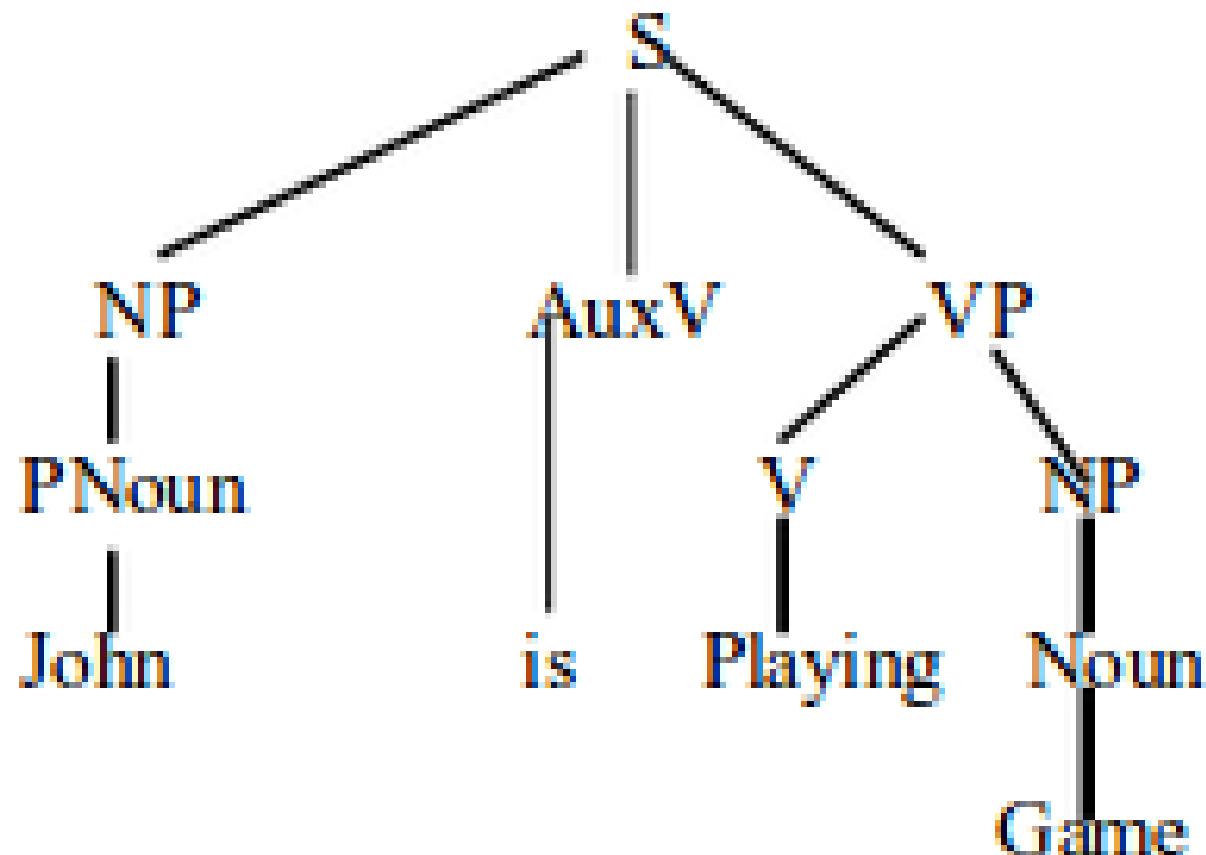
N = NOUN

V = VERB

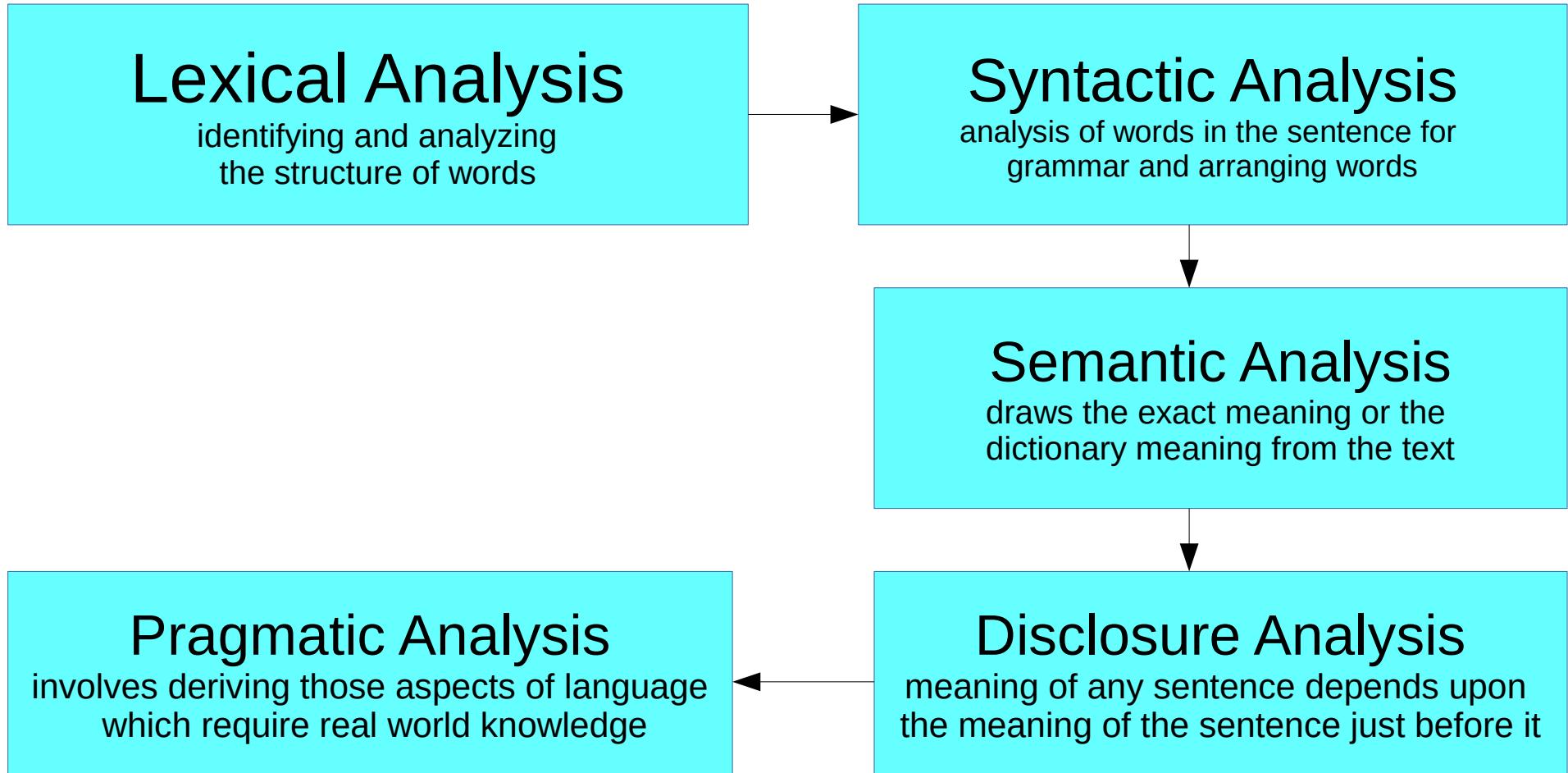
DT =

DETERMINER





Steps used in NLP



Advantages of NLP

- Relieves burden of Learning syntax
- No training needed for learning Languages

Disadvantages of NLP

- Requires clarification of dialogue
- May not show context
- Unpredictable

Parameters in Natural Language Processing:

- To be exact, the parameters always (regardless of the research field) depend on the task.
- Different parameters may include
 - ✗ Auditory Inputs
 - ✗ Segmentation
 - ✗ Syntax Structure
 - ✗ Semantic Structure
 - ✗ Pragmatic Analysis

a. Auditory Inputs

- Three of our five senses – **sight, hearing and touch** – are used as **major inputs**.
- These are usually referred to as the visual, auditory and tactile inputs respectively.
- They are **sometimes called input channels**.
- In some cases, an audio output device can be used as an input device, in order to capture produced sound.
 - ✗ Microphone
 - ✗ MIDI keyboard or other digital musical instrument

b. Text segmentation

- Segmentation: Determining the positions at which topics change in a stream of text or Speech
- Text segmentation is the process of dividing written text into meaningful units, such as words, sentences, or topics.
- The term applies both to mental processes used by humans when reading text, and to artificial processes implemented in computers, which are the subject of natural language processing.

Problems in Text Segmentation

- **Word segmentation** is the problem of dividing a string of written language into its component words.

For example, ice box = ice-box = icebox

- **Intent segmentation** is the problem of dividing written words into key phrases

[All things are made of atoms]

- **Sentence segmentation** is the problem of dividing a string of written language into its component sentences

- **Topic Segmentation** consists of two main tasks:

- ✗ topic identification

- ✗ text segmentation

c. Syntax Structure

- Syntactic analysis takes an input sentence and produces a representation of its grammatical structure.
- A grammar describes the valid parts of speech of a language and how to combine them into phrases.
- The English Grammar is nearly context free.
- A computer Grammar specifies which sentences are in a language and their parse trees. A parse tree is a hierarchical structure that shows how the grammar applies to the input. Each level of the tree corresponds to the application of one grammar rule.

Consider the sentence

“John saw Mary with a telescope”

Two different readings based on the groupings.

- John saw (Mary with a telescope).
- John (saw Mary with a telescope).

d. Semantic Structure

- Semantic analysis is a process of converting the syntactic representations into meaning representation.
- This involves the following tasks:
 - Word sense determination
 - Sentence level analysis
 - Knowledge representation

- **Word sense**

Words have different meanings in different contexts.

Example :

- Mary had a bat in her office.

 - bat = `a baseball thing'

 - bat = `a flying mammal'

- **Sentence level analysis**

Once the words are understood, the sentence must be assigned some meaning

- I saw an astronomer with a telescope.

- **Knowledge Representation**

Understanding language requires lots of knowledge.

e. Pragmatic Analysis

- Pragmatics is a subfield of linguistics that studies the ways in which context contributes to meaning
- It is the study of the ways in which language is used and its effect on the listener
- Pragmatic analysis refers to a set of linguistic and logical tools with which analysts develop systematic accounts of logical interactions.

Example

“When is the next flight to Sydney?”
“Does *it* have any seat left?”

Here, “it”, refers to a particular flight to Sydney, not Sydney itself.

Pragmatics – Ambiguity in Language

“I saw an astronomer with a telescope.”

Above sentence **has a prepositional phrase** “with a telescope” which may be attached with either with verb to make phrase “saw something with telescope” or to object noun phrase to make phrase “a astronomer with a telescope”. If we do first, then it can be interpreted as “I saw an astronomer who is having a telescope”, and if we do second, it can be interpreted as “Using a telescope I saw an astronomer”

Now, to remove such ambiguity, one possible idea is that we have to consider the context. If the knowledge base (KB) can prove that whether the telescope is with astronomer or not, then the problem is solved.

If A says the same sentence “I saw an astronomer with a telescope.” To B, then in practical, it is more probable that, B (listener) realizes that “A has seen astronomer who is having a telescope”. It is because, normally, the word “telescope” belongs to “astronomer”, so it is obvious that B realizes so.

If A has says that “I saw a lady with a telescope.” In this case, B realizes that “A has seen the lady using a telescope”, because the word “telescope” has not any practical relationship with “lady” like “astronomer”.