AI

Artificial Intelligence
↓ ↓
"man made" + "thinking process"

"It is branch of CS by which we can create intelligent machine which can behave like a human, think like a humans & able to make decisions".

With AI, you do not need to preprogram machine to do some work. You've to create a machine with programmed algorithm which can work with own intelligence.

- Why AI?

1. With the help of AI, we can create such s/w or device which can solve real world problems very easily and with accuracy such as health issues, marketing, traffic issues.

2. With the help of AI, you can create your personal virtual assistant such as Google assistant, Siri, etc.

3. With the help of AI, you can build such robots which can work in a environment where survival of human can be at risk.

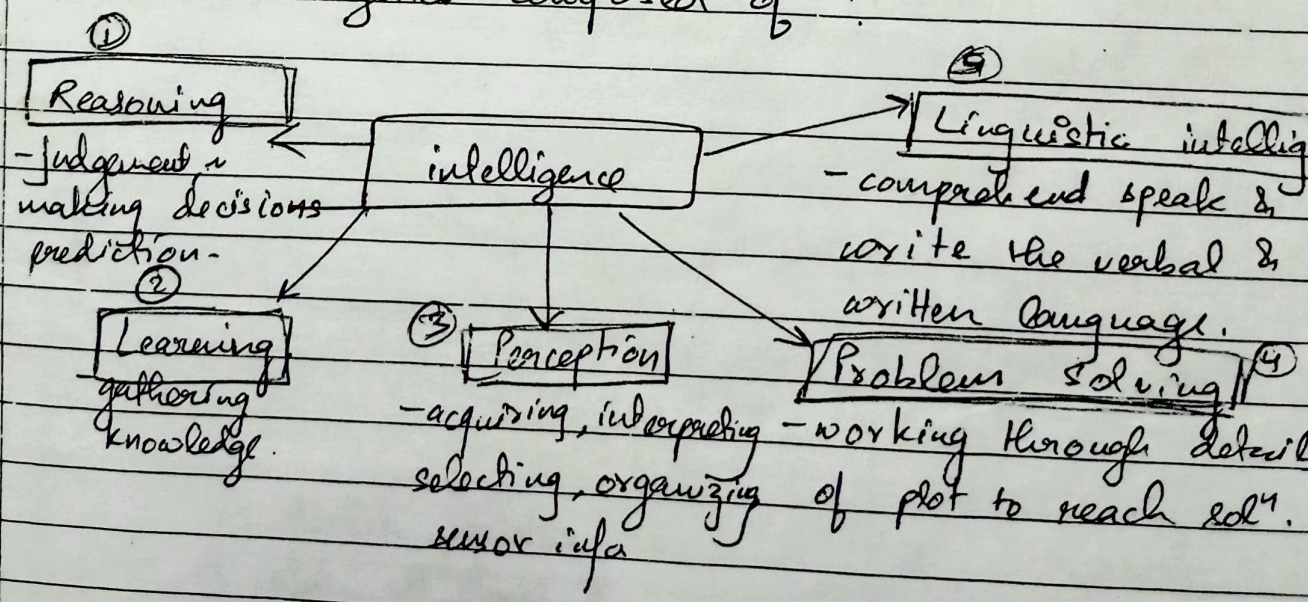
4. AI opens a path for new technologies, new devices and new opportunities.

- Goals of AI

1. Replicate human intelligence.
2. Solve knowledge intensive tasks.
3. An intelligent connection of perception and action.
4. Building a m/c which can perform tasks that require human intelligence such as
   - → providing a theorem.
   - → playing chess
   - → plan some surgical operation
   - → driving a car in traffic.

(5) Creating some system which can exhibit intelligent behaviour, learn new things by itself, demonstrate, explain and can advise to it's user.

- What is intelligence composed of ?



Camlin Page
Date / /

- Agents in AI

AI : Study of rational agent & it's environment.

"The agents sense the environment through sensors & act on their environment through actuators (is a device).
- AI agent can have mental properties like knowledge, belief intention etc.

"An Agent can be anything that perceive environment through sensors & act upon that environment through actuators."

Types of AI agents :

1. Human agent
   - eyes, ears & other organs which work for sensors, hand, legs works for actuators.

2. Robotic agent
   - cameras, infrared range finder.

3. S/w agent
   - key strokes.

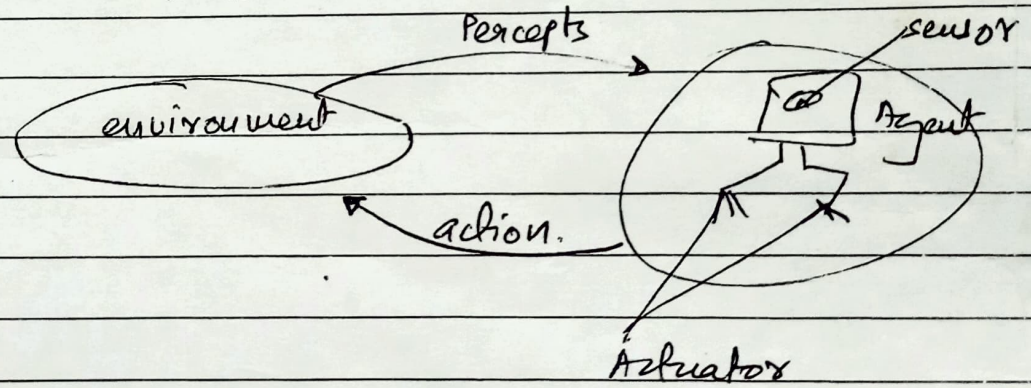
- Imp. terms :

1. **Sensors** : It's a device which detects the change in the environment & sends the info. to other electronic device.
   - An agent observes the environment through sensors.

2. Actuators
- It's a component of m/c that converts energy into motion.
- It's responsible only for moving and controlling the system.

3. Effectors : Which effect the environment. It can be legs, wheels, arms, fingers. & display screen.

Ex: Self driving car.
Here car is an agent
enviornment is road + other things.
wheel is actuator.



- Types of AI-agents.

- Agents can be grouped into five classes based on their degree of perceived intelligence & capacity.

1. Simple Reflex Agent
2. Model based reflex agent.
3. Goal based agents.
4. Utility based agent
5. Learning agent.

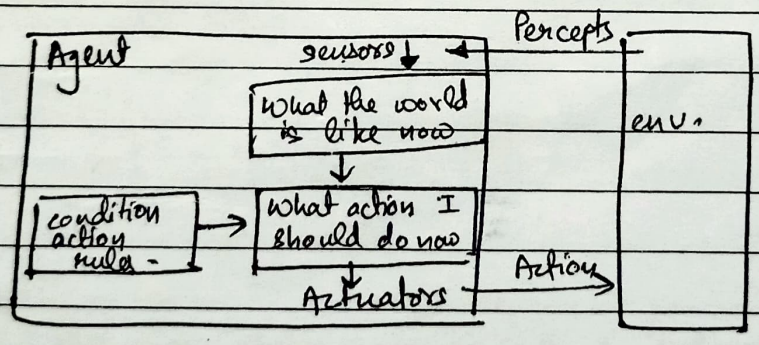
Camlin Page
Date / /

① Simple reflex agent

- This agent works only on the basis of current perception and it doesn't bother about the previous state in which the system was.
- This type of agent is based upon the "condition action rule".
  i.e. if condition is true, action is taken else not.

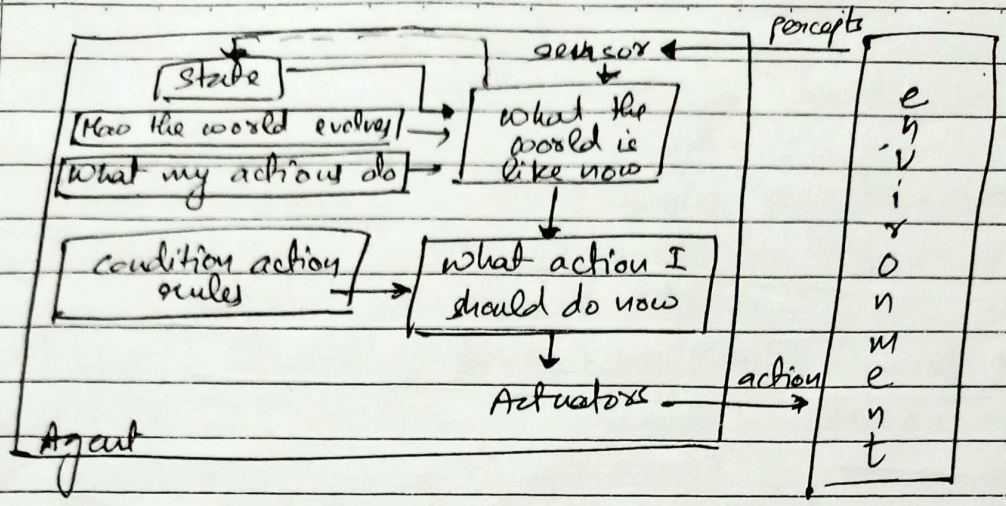
② Problems faced

1. Very limited intelligence.
2. No knowledge about the non-perceptual parts of the state.
3. Operating in a partially observable environment, infinite loops are unavoidable.



③ Model based reflex agents.

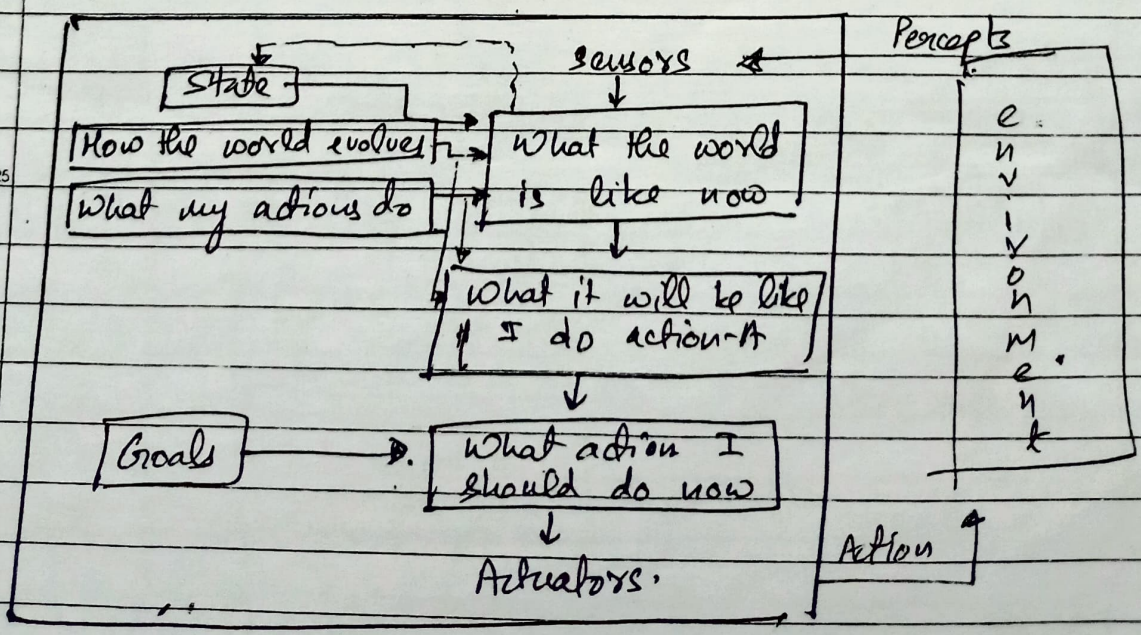
- It works by finding a rule whose condition matches the current situation.
- It can handle partially observable environment
- updating the state requires information about
  ⇒ how the world evolves independently from the agent &
  ⇒ how the agent actions affect the world.



(3) Goal based Agent

- The goal based agent focuses only on reaching the goal set and hence the decision took by the agent is based on how far it is currently from their goal or desired state.

- Their every action is intended to minimize their distance from the goal.
- This agent is more flexible and the agent develops the decision making skill by choosing the right from the various options available.



Camlin Page
Date / /

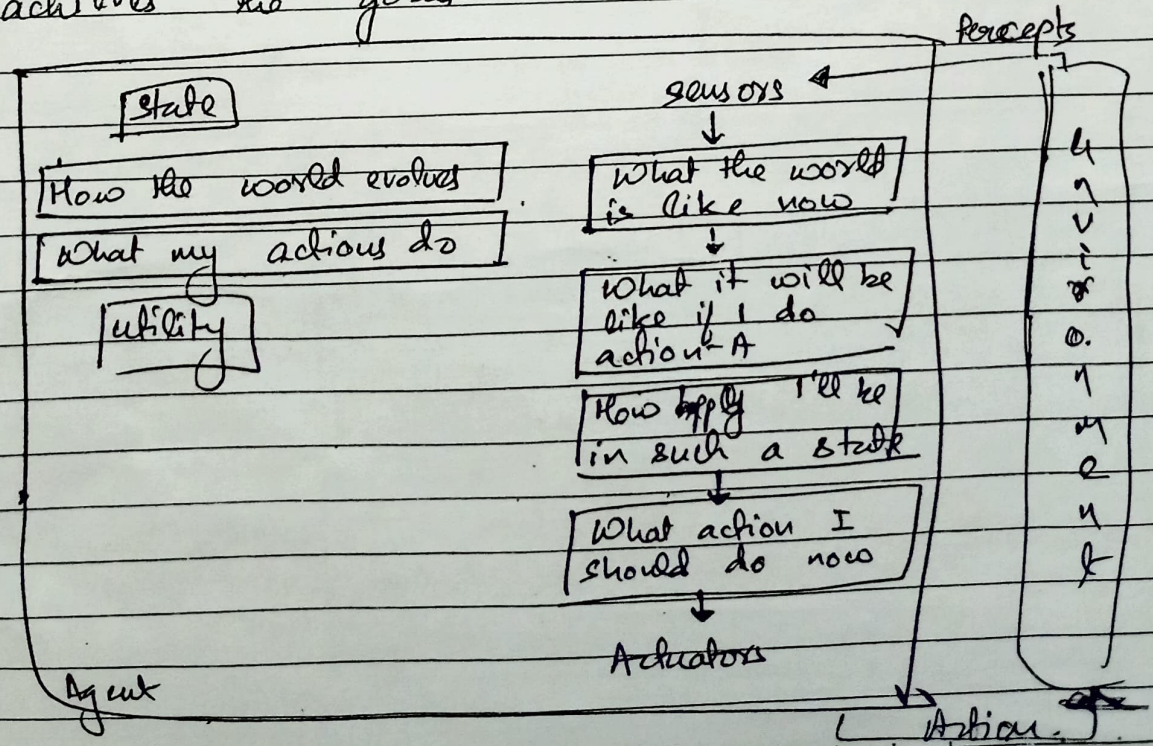
(4) Utility · based agents

- These agents are similar to the goal-based agent but provide an extra component of utility measurement which makes them different by providing a measure of success at a given state.

- Utility based agent act based not only on goals but also the best way to achieve the goal.

- The utility based agent is useful when there are multiple possible alternatives and an agent has to choose in order to perform the best action.

- The utility function maps each state to a real number to check how efficiently each action achieves the goals.



Camlin Page
Date / /

(5) Learning agent

- It can learn from it's past experience or it has a learning capabilities.
- It starts to act with basic knowledge and then able to act & adapt automatically through learning.
- A learning agent has mainly from conceptual components, which are:-

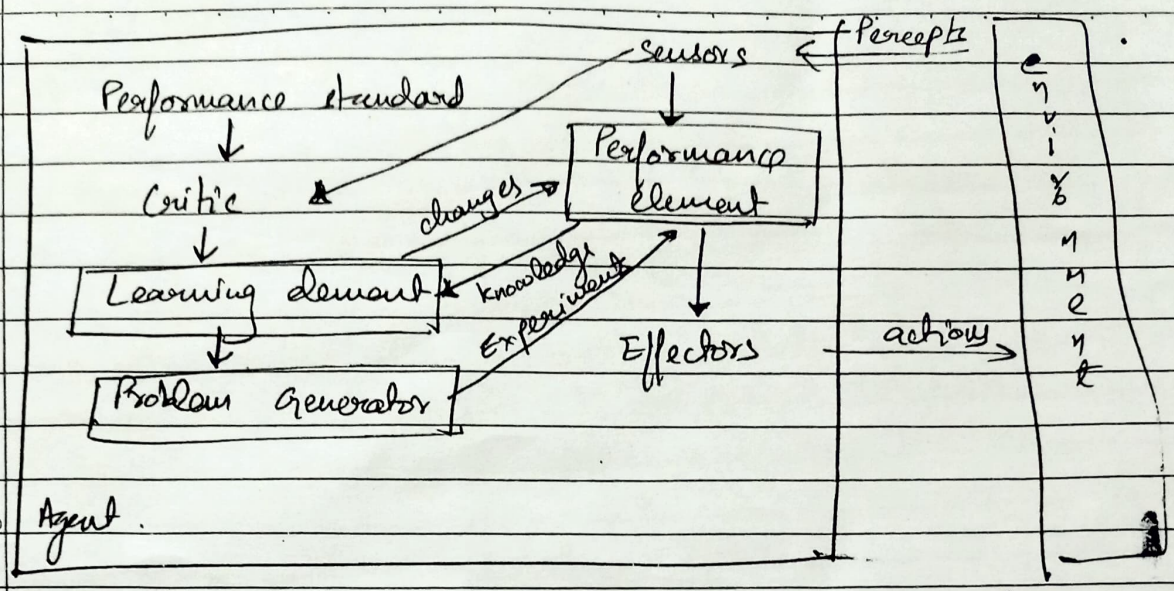
(a) Learning element : It's responsible for making improvements by learning from environment.

(b) Critic : Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.

(c) Performance element : It's responsible for selecting external action.

(d) Problem generator : The component is responsible for suggesting actions that will lead to new and informative experiences.

Hence, learning agents are able to learn, analyze performance and look for new ways to improve the performance.



Agents environment in AI

An environment is everything in the world which surrounds the agent, but it's not a part of an agent itself.
=> An environment can be ~~classit~~ described as a situation in which an agent is present.

Features of environment

- An environment can have various features from the point of view of an agent :-

1. Fully observable vs partially observable.
2. Static vs dynamic
3. Discrete vs continuous
4. Deterministic vs stochastic
5. Single agent vs multi-agent.
6. Episodic vs sequential

Camlin Page
Date / /

(7) known vs unknown
(8) Accessible vs inaccessible.

(1) Fully observable vs Partially observable.
- If an agent sensor can sense the complete state of an environment at each point of time then it's a fully observable environment else it is partially observable.

(2) Deterministic vs stochastic
- If an agent currents state & selected action can completely determine the next state of environment then such env. is called deterministic.
- A stochastic env. random in nature and can't be determined completely by an agent.

(3) Episodic vs sequential
- In episodic env., there is a series of one shot actions & only the current percept is required for the action.
- In sequential an agent requires memory of past actions to determine the next best actions.

(4) Single vs multi agent
- If only one agent is involved in an env, & operating by itself then such an env. is single agent
- If multiple agents are operating in an env., then such an env. is called multi agent.

Camlin Page
Date / /

(5) Static vs dynamic
- If env. can change itself while an agent is deliberating then such env. is called dynamic else it is called static.
- Taxi driving is an ex. of dynamic and crossword puzzles are ex. of static

(6) Problem solving
- Problem solving in games such as "Sudoku" eg. It can be done by building an AI system to solve that particular problem.

- To do, this one needs to define the problem statement first and then generating the solution by keeping the condition in mind.

- Some of the most popularly used problem solving with the help of AI are: chess, travelling salesman problem, Tower of Hanoi problem, water-jug problem, N-Queen problem.

(7) Problem searching
- In general, searching refers to as finding information one needs.
- Searching is the most commonly used technique of problem solving in A.I.
- The process of solving a problem consists of 5-steps.

Define the problem
↓
Analyzing the problem
↓
Identification of solution
↓
Choosing the solution
↓
Implementation

Good behaviour in AI

- An agent should act as a rational agent. A rational agent is one that does the right thing that is the right actions will cause the agent to be most successful in the environment.

" The environment is the Task environment (problem for which the rational agent is the solution. Any task environment is characterised on the basis of PEAS (Performance, Environment, Actuator Sensors)! ".

Structure of agents in AI

An intelligent agent is a combination of agent program and architecture.

intelligent agent = Agent program + architecture.

- Agent program is a function that implements the agent mapping from percepts to actions. There exists a variety of basic agent program designs, reflecting kind of information made explicit and used in the decision process. The designs vary in efficiency, compact and flexibility. The appropriate design of the agent program depends on AP.

Camlin Page
Date / /

- Properties of search algorithms

- The essential properties of search algorithms to complete the efficiency of these algo.

① **Completeness:** A search algo. is said to be complete if guarantees to return a solution if at least any sol. exist for any random i/p.

② **Optimality:** If a sol^n find for an algo. is guaranteed to be the best sol (lowest-path cost) among all other soln, then such sol for is said to be optimal soln.

③ **Time complexity:** Time complexity is a measure of time for an algo. to complete it's task.

④ **Space complexity:** It's the max. storage required at any point during the search, as the complexity of the problem.

- Types of search algorithm

Based on the search problems we can classify the search algorithm into
1. uninformed search (Blind search) &
2. informed search (Heuristic search)

Search Algorithm

Uninformed / Blind	Informed Search
1. BFS	1. Best first search
2. Uniform cost search	2. A* search.
3. Depth first search	3. AO* algorithm
4. Depth limited search	4. Problem Reduction
5. Iterative deeping search first search	5. Hill climbing.
6. Bidirectional search	

- Uninformed Search
  - The uninformed search doesn't contain any domain knowledge such as closeness, the location of the goal.
  - It operates in a brute force way, as it only includes information about how to traverse the tree & how to identify leaf & goal nodes.
  - Uninformed search applies a way in which search tree is searched without any information about the search space like initial state operators & test for the goal, so it's called blind search. It examines each node ~~at~~ until it achieves the goal node.

- Informed Search
  - It uses domain knowledge the problem info. is available which can guide the search.
  - Informed search strategies can find a solution more efficient than an uninformed. Informed search is also called as Heuristic Search.

- A heuristic is a way which might not always be guaranteed for the best soln but guaranteed to find a good soln in reasonable time.
- It can solve much complex problems which could not be solved in ~~and~~ another way.

(x) Uninformed search algorithms

(1) Breadth first search (BFS)

- It's the most common search strategy for traversing a tree or graph.
- This algo. searches breadth wise in a tree or graph so it's called breadth-first search.
- BFS algo. starts searching from the root node of the tree and expands all successor node at the current level before moving to node of next level.
- BFS algo. is an example of general-graph search algo.
- BFS implemented using FIFO queue data structure.

Advantage

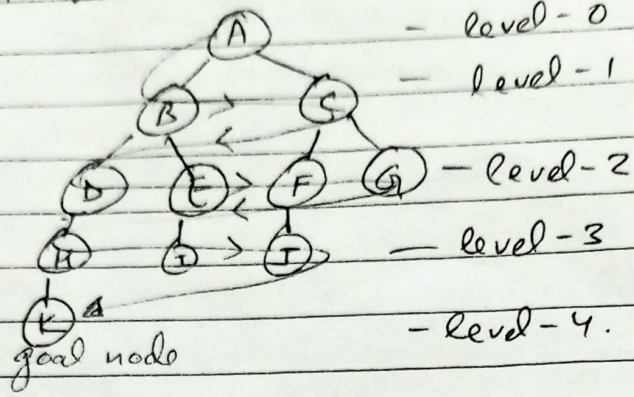
- BFS will provide a solution if any sol. exists.
- If there are more than one soln. for a given problem, then BFS will provide minimal soln. which requires the least no. steps.

Disadvantage

- It requires lots of memory since each level of the tree must be saved into memory to expand the

- BFS needs lots of time if the soln is far away from the root node.

ex:

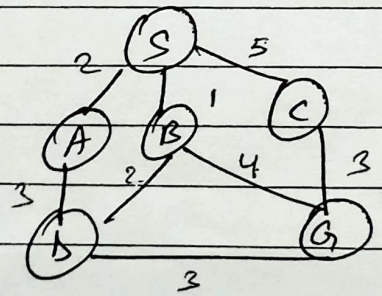


- Time complexity $O(b^d)$
  where d = depth of shallowest node
  b = node at every state.
  $T(b) = 1 + b^2 + b^3 + \dots + b^d$
  $= O(b^d)$

- Space complexity $O(b^d)$

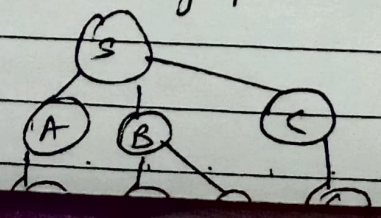
- Completeness : BFS is complete.
- Optimality : Yes optimal.

ex:



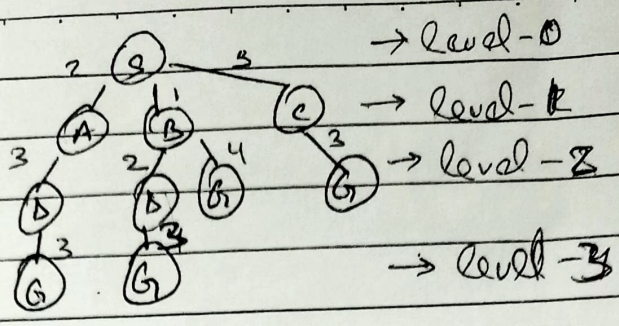
Find the route from S to A using BFS.

step 1: Visit the root node i.e. S
step 2: Visit all the successors of root nodes. i.e. A, B, C.
step 3: Expand the graph in the form of tree.
i.e.



Date / /

step-4 :



Ans : SBG or SCG.

(2) Depth First Search

- It is a recursive algo. for traversing a tree or graph data structure.
- It's called DFS because it start from the root & follows each path to it's greatest depth node before moving to the next path.
- DFS uses a stack as for it's implementation
- The process is similar to BFS algo.

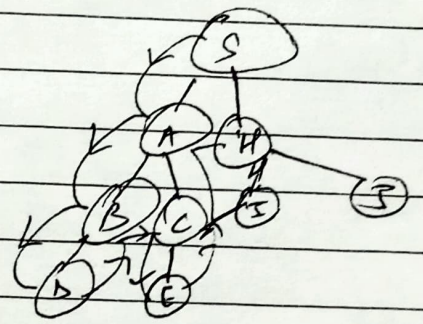
Advantage

- It requires very less memory as it only needs to store a stack of the nodes on the path from root node to the current node.
- It takes less time to reach to the goal node than BFS algo.

Disadvantage

- There is the possibility that many states keep re-occuring & there is no guarantee of finding the solution.
- DFS algo. goes for deep down searching and sometime it may go to the infinite loop.

- Completeness :- It's complete within finite state space.
- Time complexity: $T(n) = 1 + n^2 + n^3 + \dots + n^m = O(n^m)$
  m = max. depth of any node.
- Space complexity $O(bm)$ — b is level, m is max. depth of any node.
- Optimality : It's not-optimal.
  cuz sometime it may goes to infinite loop



(3) Depth Limited Search Algo.
- It's similar to DFS with a predetermined limit. Depth limited search can solve the drawback of the infinite path in DFS. In this algo., the node at the depth limit will treat as it has no successor nodes further.

- Depth limited search can be terminated with two conditions of failure:-

(i) Standard failure value : It indicates that problem doesn't not have any solution.

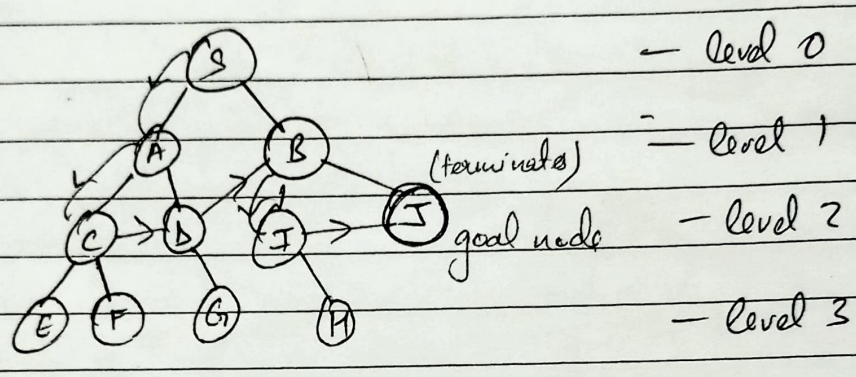
(ii) Cut-off failure value : It defines no soln for the problem within a given depth limit.

Advantage: It's memory efficient.

Disadvantage:

- incompleteness.
- It may not be optimal if the problem has more than one solution.

ex:



limit level = 2.

Time complexity : $O(b^l)$

Space complexity $O(b \times l)$

- It's not optimal.

(4) Uniform - cost search Algorithm

- It's used for traversing a weighted tree or graph.
- It comes into play when a different cost is available for each edge.
- The goal of ucs is to find a path to the goal node which has the lowest cumulative cost.
- It expands nodes according to their path costs from the root.
- It can be used to solve any graph / tree where the optimal cost is in demand.

- A UCS algo. is implemented by the priority queue.
- It gives max. priority to the lowest cumulative cost.
- It's equivalent to BFS if the path cost of all edges is the same.

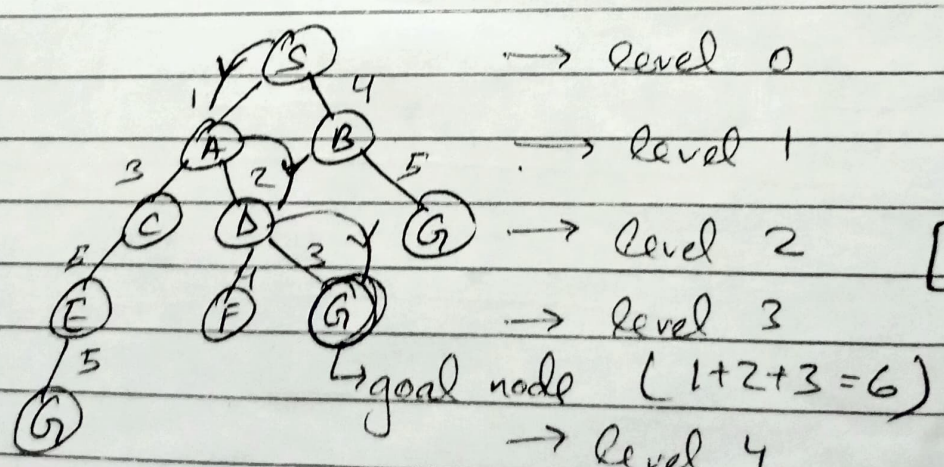
Advantages:

- Uniform cost search is optimal because at every state the path with the least cost is chosen.

Disadvantages

- It doesn't care about the no. of steps involve in searching and only concerned about path cost. Due to which this algo. may stuck in an infinite loop.

ex:



(5) Bidirectional search algo.

- It runs two simultaneous searches on from initialisation start node called as forwarard search & other from goal node called as backward search, to find goal node.

- It replaces one single search graph with two small ~~g~~ subgroup in which one starts the search from initial vertex & other starts from goal vertex.

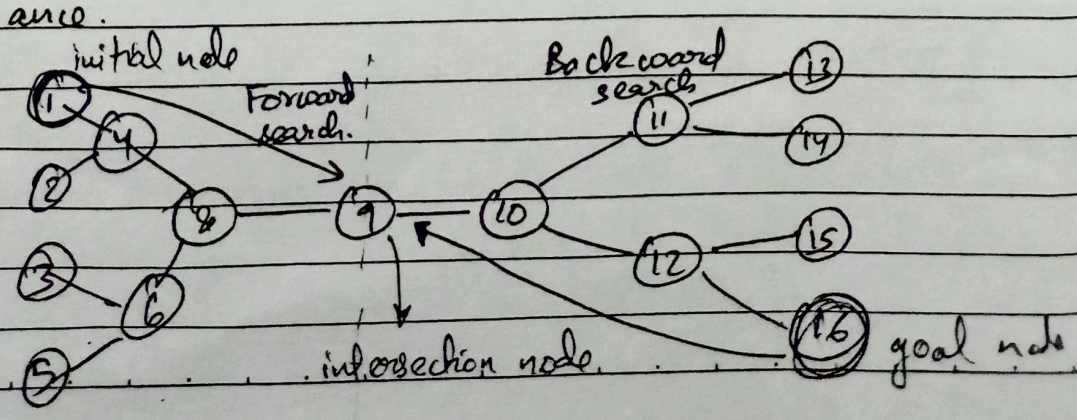
- The search stops when these two graphs intersect each other.
- Bidirectional search can use search techniques such as BFS, DFS, DLS, etc.

Advantage
- Bidirectional search is fast.
- Biderectional search required less memory.

Disadvantage

- Implementation of the bidirectional search tree is difficult.
- In this, one should know the goal state in advance.

ex:



- It's complete

Informed Search Algo.

① Best-first search algo. (Greedy search).
- It always selects the path which appears best at that moment.
- It's combination of DFS & BFS
- It uses the heuristic function $\boxed{h(n) <= h^*(n)}$ and search

where $h(n)$ = heuristic cost.
$h^*(n)$ = estimated cost.

- The greedy best first algo. is implemented by "priority queue".

Best first search Algo. :-

Step-1 Place the starting node into the open list.

Step-2 If the open list is empty, stop & return failure.

Step-3 Remove the node n, from the open list which has lowest value of h(n) & places it in the closed list.

Step-4 Expand the node n & generate the successors of node n.

Step-5 Check each successor of node n and find whether any node is a goal node or node. If any successor node a goal node then return success or terminate the search, and proceed to step-6.

Step-6 For each successor node, algo checks for evaluation function F(n) & then check if the node has been in either open or closed list. If the node has not been in both list then add it to open list.

Step-7. & Return to step-2.

Camlin Page
Date / /

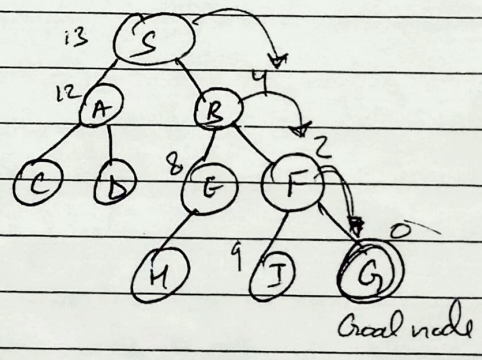
Advantages

- Best first search can switch between BFS & DFS by gaining the advantages of both the algo.
- This algo. is more efficient than BFS & DFS algo.

Disadvantage

- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS.
- This algo. is not optimal.

ex:



node	h(n)
S	13 ⇒ initially
A	12
B	4
C	7
D	3
E	8
F	2
H	4
I	9
G	0

open [A, B], close [S]

iteration-1 : open [A], close [S, B]

iteration-2 : open [E, F, A], close [S, B]

↳ open [E, A] close [S, B, F]

iteration-3 : open [E, A, I, G], close [S, B, F]

↳ open [E, A, I] close [S, B, F, G]

Final sol^n path : $\boxed{S \to B \to F \to G}$

- Time complexity is $O(b^m)$
- Space complexity is $O(b^m)$
- Incomplete
- Not optimal.

(2) A* search algorithm

- A* search algo. finds the shortest path through the search space using the heuristic function.

- It uses h(n) & cost to reach the node n from the start stage g(n).
- This algo. expands less search tree and provides optimal result faster.
- It's similar to UCS except that it uses g(n)+h(n) instead of g(n).

- A* use search heuristic as well as the cost to reach the node. Hence we combine both costs as, $F(n) = g(n) + h(n)$ {fitness number}

where $F(n)$ = estimated cost of the cheapest solution.
$g(n)$ = cost to reach node n from start state.
$h(n)$ = cost to reach from node to goal node.

Algorithm of A* search

step-1: Place the starting node in open list.
step-2: Check if the open list is empty or not, if the list is empty then return failure & stops.

Camlin Page
Date / /

step-3: Select the node from the open list which has the small value of evaluation function (g+h) if node n is goal node then return success & stop otherwise.

step-4: Expand node n & generate all of it's successors & put n in the closed list.
- For each successor 'n', check whether 'n' is already in the open or closed list.
- If not then compute evaluation function for 'n' and place into open list.

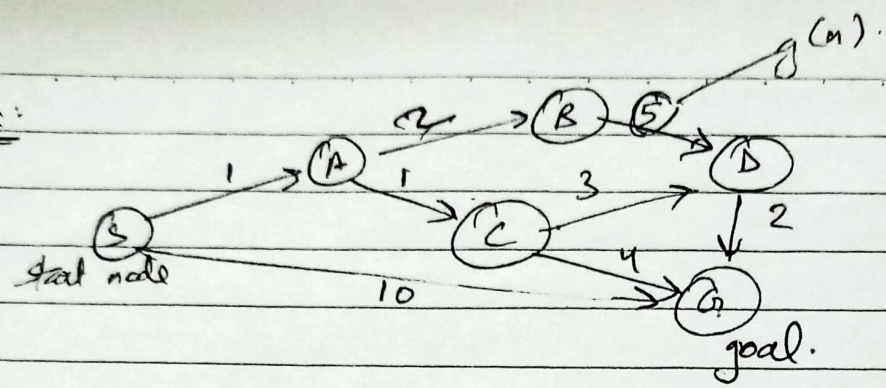
step-5: Else if node 'n' is already in open & closed then it should be attached to the back pointer which reflects the lowest g(n) value.

<u>step-6</u>: Return to step-2.

<u>Advantages</u>
- It's the best algo. than other search algos.
- It's optimal & complete.
- It can solve very complex problems.

<u>Disadvantages</u>
- It doesn't always produce shortest path
- It's not practical for various large scale problems.

ex:



state	h(n)
S	5
A	3
B	4
C	2
D	6
G	0

step-1 : $S \to A$ $\quad F(n) = g(n) + h(n)$
$= 1 + 3 = 4$

$S \to G$; $F(n) = 10 + 0 = 10$
(hold)

step-2: $S \to A \to B$, $F(n) = \underbrace{(1+2)}_{g(n)} + \underbrace{4}_{h(n)} = 7$ (hold)

$S \to A \to C$; $F(n) = \underbrace{(1+1)}_{g(n)} + \underbrace{2}_{h(c)} = 4.$

step-3: $S \to A \to C \to D$; $F(n) = (1+1+3) + 6$
$= 11$ (hold)

$S \to A \to C \to G$; $F(n) = (1+1+4) + 0 = 6$

$S \to A \to C \to G$ (cost is 6).

(3) AO* search algorithm (AND-OR graph)

Camlin Page
Date / /

- Local search algo. & optimization problem

The informed and uninformed search expands the nodes systematically in two ways:-
1. Keeping different paths in the memory and
2. Selecting the best suitable path,

which leads to a solution state required to reach the goal node. But beyond these "classical search algos." we've some "local search algo." where the path cost does not matters and only focus on solution state needed to reach the goal node,

A local search algo. completes it's task by traversing on a single current node rather than multiple paths and following the neighbours of that node generally.

Although local search algos are not systematic, still they have the following two advantages:-

- Local search algo. use a very little or constant amount of memory as they operate only on a single path

- Most often, they find a reasonable solution in large or infinite state spaces where the classical or systematic algos. don't work.

Does the local search algo. work for a pure optimized problem?

Yes, it works for pure optimized problem.

A pure optimized problem is one where all the nodes can give a solution. But the target is to find the best state out of all according to the objective function. Unfortunately the pure OP fails to find high quality sols. to reach the goal state from the current state.

Some different types of local searches:-

1. Hill-climbing search.
2. Simulated Annealing.
3. Local Beam search.