State Space Search

* What is State Space Search?

=> State Space Search is a problem solving technique used in Artical Intelligence.

A State Space Search is a representation of all possible states of a system to solve the problem.

Process of create solution for given data to solve the problem we need to build a system.

Four things required to build a system to solve the problem.

(i) Define : In this step, Problem specify the inital condition.

Output will be contribute the expectable solution of problem.

(ii) Analysis : In this step, we have to find the appropiate way to

Find the possible method to solve the problem.

(iii) Isolate & Represent: In this step, we have to collect the information which is required to solve the problem.

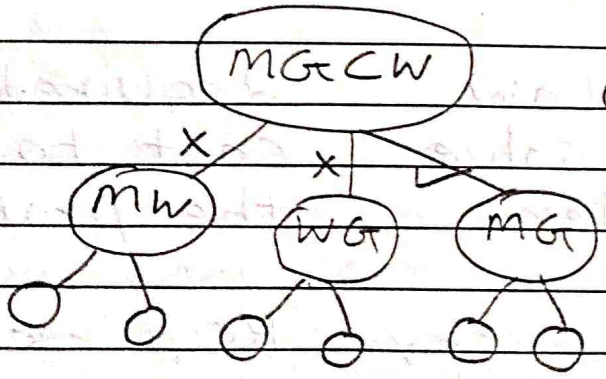
(iv) Technique: In this step, we have to select the best method to solve the problem.

Ex. Problem :

Male, Goat Cabbage Wolf

River } Two can be go together at a one time.

State Space Search



Only Men and Goat Go together From this Diagram.

* Write Difference Between Informed and Uninformed Search.

Or.

Explain Informed and Uninformed Search.

=>

	Informed Search	Uninformed Search
1	Informed Search is also known as Heuristic Search Algorithm.	Uninformed Search is also known as Blind Search Algorithm.
2	Every Problem have additional information to solve the Problem.	Problem does not have any informatic about the solve the problem.
3	This Problem use Heuristic that give the estimate of how close a state is to the goal.	This Problems explor the search space systematically to solve the problen
4	Required Low cost to solve the problem.	Required high cost to solve the problem.
5	Find the every problem solution more quickly.	Required more time to solve the problem.

	Informed Search	Uninformed Search
6	Every Problem starting and goal states are already define.	There is no starting and goal states information.
7	We will have Domain and additional information about problem.	We have to find the additional information about problem.
8	Example: ~~DFS~~, A* Search Hill Climbing, AO Search	Example: BFS, DFS

* Explain Water Jug Problem with its example.

=> Problem:

You are Given 2 Jugs, One Jug have the 3 gallons of water and other Jug has 4 gallons of water. There is no other measuring equipment available and Jug does not have any kind of marking.

x = 4 Gallon Jug
y = 3 Gallon Jug

So, How can You got 2 gallon water in the 4 gallon water Jug? Using only 2 Jug.

Solution:

For solve this Problem, We have to use Production Rules.

There are 12 Production Rules.

	Rule	Description
1	$(x, y) \rightarrow (4, y)$ if $(x < 4)$	Fill the Jug with 4 G water.
2	$(x, y) \rightarrow (x, 3)$ if $(y < 3)$	Fill the Jug with 3 G water.
3	$(x, y) \rightarrow (x-d, y)$ if $x > 0$	Pour some d water from 4 G water Jug.
4	$(x, y) \rightarrow (x, y-d)$ if $y > 0$	Pour some d water from 3 G Jug.
5	$(x, y) \rightarrow (0, y)$ if $x > 0$	Empty the 4 G Jug in the Ground
6	$(x, y) \rightarrow (x, 0)$ if $y > 0$	Empty the 3G Jug in the Ground

7	(x, y) → (4, y-(4-x)) if x+y ≥ 4, y > 0	Pour water From 3G Jug into 4G Jug Until the Jug is full.
8	(x, y) → (x-(3-y), y) if x+y ≥ 3, x > 0	Pour water from 4G Jug into 3G Jug until the Jug is full.
9	(x, y) → (x+y, 0) if x+y ≤ 4, y ≠ 0	Pour all water From 3G Jug to 4G Jug
10	(x, y) → (0, x+y) if x+y ≤ 3, x > 0	Pour all water from 4G Jug to 3G Jug.
11	(0, 2) → (2, 0)	Pour 2G From 3G into 4G Jug
12	(2, y) → (0, y)	Empty the 2G water in the 4G Jug in Ground

4G x Jug	3G Y Jug	Rule Number
0	0	-
0	3	2
3	0	9
3	3	2
4	2	7
0	2	5
2	0	11

At the end of the solution table, we are get 2G water in the Y Jug or 3G Jug.

* Explain Production ~~Syem~~ System with its Advantages and Disadvantages.

=> Production System is used to structure the AI Problem.

Production System helps in structure AI Program in which that help to describe, Facilate and Perform the its search process.

Production System is a compute program which consistes the set of rules, about the behaviour of Problem etc.

It is also include the mechanism to follow the rule as system response.

Production System separate the control system and knowledge base.

SMVS
Page No.
Date : / /

=> Steps to solve the Problem using Production System.

1 Reduce the Problem so, that it can be shown in

2 Program can be solved by searching a part through space.

3 Solving Process can be model through Production system.

=> Characteristics of Production System.

1 Monotonic : In Production System, One rule apply one time, after that another rule is apply.

2 Non-Monotonic : Production System is usefull for solving ingorable problem. This system can be prevent, without do backtrack two previous state ~~with~~ when it is discover that incorrect path is follow.

3 Partial Commutative : In which, we have to apply sequence of a rule for transform x -> y, then any permutation those rule is allowed to transfer Y to X.

4 Commutative : Combination of Monotonic + Non-Monotonic + Partial Commutative.

=> Advantages of Production System

1 Structure the AI Problem.

2 Production System is highly moduler, In which we can add, remove or modify the rule independently.

3 Rules are define in Natural Form.

4 Separate the Control system and Knowledge base.

5 Beneficial in Real-Time World.

=> Disadvantages of Production System.

1 Describe the operation that can be perform in a search.

2 Production System does not store result of Problem for Future also.

3 When new rule is add in the database, It should be ~~e~~ insure that rule conflict is not occures.

* Explain Control Strategies in AI.

=> Control Strategies use to decided the rule use for the Find the solution.

It is help to selecte to rule while searching for the solution of any problem.

-> First Requirement : Motion.

Motion means we have to select always first rule to solve the Problem.

In Motion, every time we have to select top of the list rule, then we can not solve the Problem.

Control Strategies that does not cause the motion will never give the solution.

-> Second Requirement: Systematic

Systematic means, for each step to solve the problem we can select the random applicable rule from the list.

We can use same rule in many time for the solve problem.

This approach is always give the solution of Problem.

Ex. Water Jug:

-> Motion: If we select the top of the list rule to solve the problem then we can not solve the water Jug Problem.

So, Using Motion we can not solve water Jug Problem.

-> Systematic: In water Jug, we can select the random applicable rule from the list.

We can use same rule in many time for solve water Jug Problem.

So, Using Systematic approach we can solve the water Jug Problem.

* Explain Traveling Salesman Problem.

=> Problem : Given a list of cities and the distance between each pair of cities, the task is to find the shortest possible tour that visits each city exactly once.

-> Solution :

Using the Brute Force and Heuristic approach we can find the all the possibility of the searching path.

Also we can use Uninformed Search and Nearest Neighbor Method to solve the problem.

For Finding the solution we have find optimal permutation of cities that minimizes the total distance traveled.

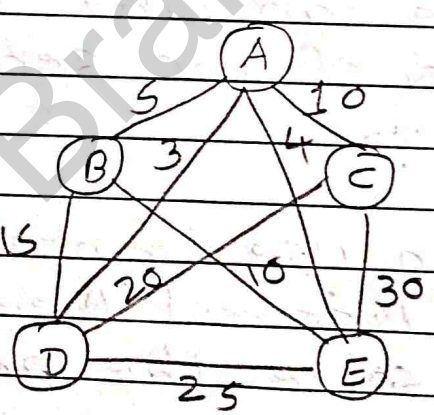
Using Permutation, we can find the number of possibility to travel path.

If there are n cities,

then Number of
Possibility = (n - 1)!

After that using the uninformed Search Method we can find the shortest path for travel.

Ex. There are 5 cities with given distance.



So, Number of
Possbility = (n - 1)!
= (5 - 1)!
= 4!

* Explain Breadth First Search with its algorithm and example.

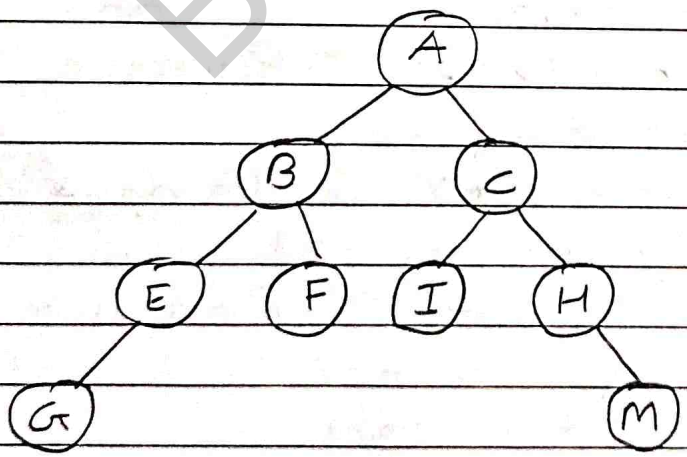
=> Breadth First Search is a Uninform Search Method, which is know as Blind search.

Using Brute Force Searching method, we can perform the BFS.

BFS starts searching From the root node and expand all subtree node at the same level, without moving the next level.

BFS Search is perform Level by Level of the tree.

Ex.



=> First A is enter in the Queue which is Follow First In First Out manner

	A	
2	B C	→ A Remove and B, C enter
3	C E F	→ B Remove and E, F add.
4	E F I H	→ C Remove and I, H add.
5	F I H G	→ E Remove. H add
6	I H G	→ F Remove
7	H G	→ I Remove
8	G m	→ H Remove
9	m	→ G Remove
10		→ M Remove

Stop.

Path : A → B → C → E → F → I → H → G → M

=> Algorithm:

1 Create Variable list which is called Node state and set the Intial state.

2 Until Goal state is Found and Node List is empty:

a) Remove First element From Node list and called as E.

IF Node List was Empty, then Quit.

b) For each way that each rule can match the state discribe in E
do:

1) Apply to rule to generate new state.
2) IF New state is Goal state, then Quit
3) otherwise, add the new state to end of node list.

=> Complexity : $O(b^d)$, b = Branching Factor
d = Deapth.

If we get the solution, then we get complete and optimal solution.

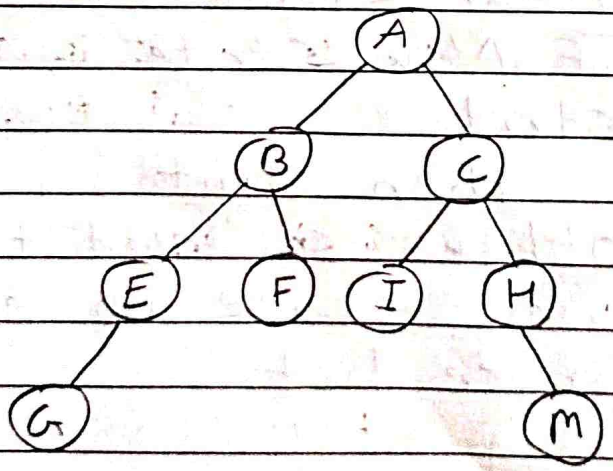
* Explain Depth First Search with its algorithm and example.

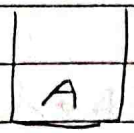
=> Depth First Search is a uninform Search which is use to trace the tree using the backtracking.

DFS Searching is starts From the Root Node and then travel to the deeper until the ~~root~~ leaf node is found.

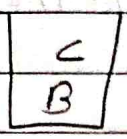
DFS is use Stack For the trace the tree which is follow LIFO Manner.

Ex.

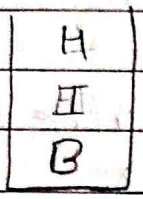


1 

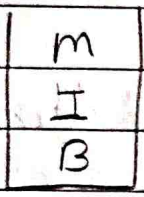
→ A Push From the stack

2 

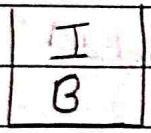
→ A Pop and B, C Push

3 

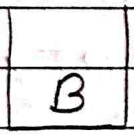
→ C Pop and I, H Push

4 

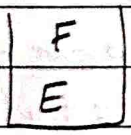
→ H Pop and M Push

5 

→ M Pop

6 

→ I Pop

7 

→ B Pop and F, E Push

8 

→ F Pop

9 

→ E Pop and G Push

10 
→ G Pop

Path : A -> C -> H -> M -> I -> B -> E -> F -> G

IF We get the solution, then we get incomplete and non-optimal solution.

=> Algorithm :

1 Create a variable list called as Node List and Intialize Root Node or State.

2 Until Goal State is Found and Node list is empty :

A) Remove the First Node From the stack.

IF Node List was empty then Quit.

B) For each way that each rule can match the state, do :

1) IF Node = Goal State Quit

2) Else enter all the node in stack.

3) Repeat Step (A).

* Explain Heuristic Search in AI.

=> Heuristic Search is a method used in artifical intelligence to solve problems more efficiently.

Heuristic Search is a method is used to solve the problem Faster than the classic problem solving method.

This method is find the solution using the past experiance with the similar problem.

This method uses practical method and short cut for the produce the solution.

Sometime, this method solution is optimal or not or sufficient For a limited time frame.

For performing the Heuristic Search we have to use Heuristic Function to solve the problem.

Heuristic Function is used to Find the most promising path to find solution.

This Function takes current states as a Input and produce the estimation of how close agent is from the goal.

Heuristic Function is a method that estimates the cost from the current state to goal state.

Heuristic Function is denoted by the h(n).

Heuristic Function, estimation cost is always positive.

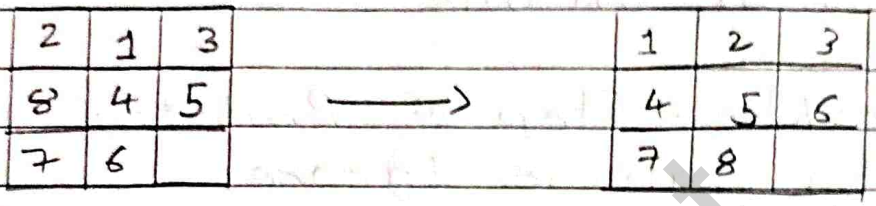
The effectiveness of Heuristic search is depends on the qualit of the Heuristic Function.

Heuristic Search is more efficient for problem solving and reduced Search Space.

Heuristic Search is always use Informed Search Algorithm or Guided Search Algorithms.

Heuristic Search is used in pathFinding, puzzle solving and game playing.

Ex 8-Puzzle Game.



Input Goal state

Total Cost = 3^{20}

* Explain Characteristics of Problem.

=> This are the Characteristics of Problem.

(1) Is the Problem is Decomposable?

Divided the Problem into a small part, which create multiple small problem.

Whem Problem is large then we have to divided the problem in small part.

Ex. $\int (x^2 + 3x) \cdot dx$ or Block World Problem.

$= \frac{x^3}{3} + \frac{3x^2}{2}$

(2) Can solution step can be Ingore or Undone?

- Ignorable : Step of Problem solving can be Ignore.
Ex. Theorem Proving.

- Recoverable : Step of Problem solving can be undone.
Ex. 8 Puzzle.

- Irrecoverable : Step of Problem solving can not be undone.
Ex. Chess.

(3) Is the Universe predictable?

We can not the predicte the Universe.

Ex. Bridge Games : We can not exactly no about the all the distribution and what other playing will do on that turn.

There are Two Types of Outcome.

- Certain Outcome : Planning can be used to generate sequence of operator that grantee for the solution.

Ex. 8 Puzzle.

- Uncertain Outcome : A sequence of generated operator can only have good probability for the solution.

Ex. Cards Game.

(4) Is solution is Good, Absolute or Relative ?

- Absolute : Facts either of any path that lead to the solution.

IF we not follow the path, there is no solution.

IF some other path might be lead to be solution.

Ex. Facts.

- Relative : Solution of Problem is minimum or best.
Ex. TSP

(5) Is a solution State or Path ?

- Path : Perform the action by anyone.
- State : Ambiguity of any word.

Ex. The Bank President ate a dish of pasta with a frock.

Path - Bank President
State - Bank → River be a state

(6) What is the role of Knowledge?

How much knowledge whould you required by perfect program ?

- Very little knowledge required to search and speed up the execution of program.

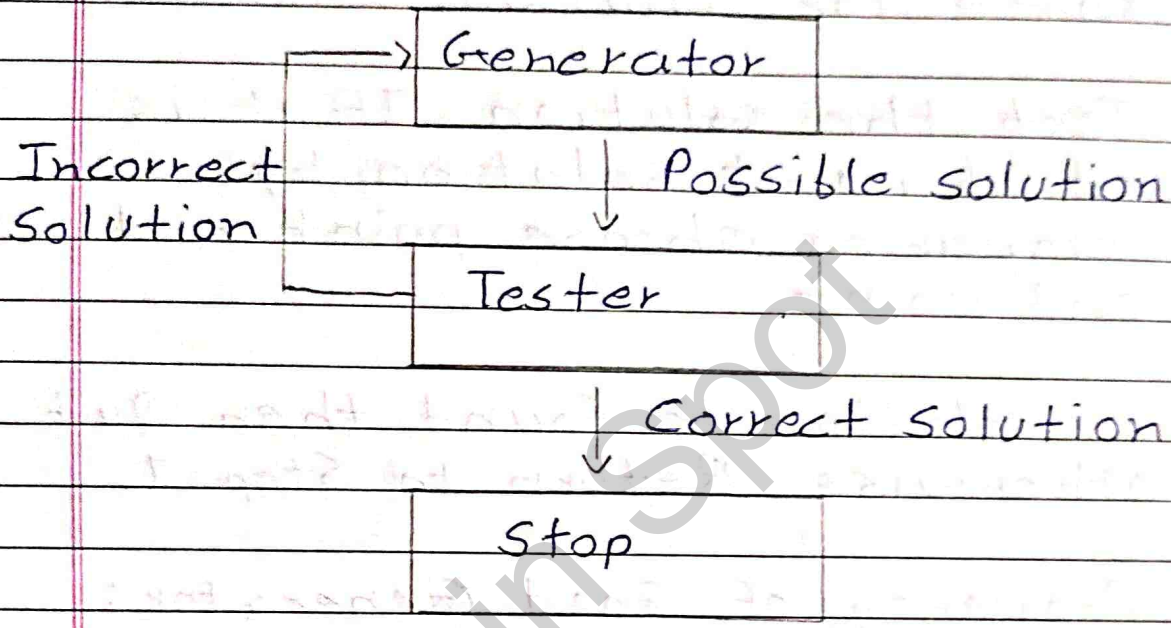
(7) Does the task required to interaction with person ?

- Yes, the level of interaction between human and machine user is a problem in solution out.

Ex. Maths Theorem.

* Explain Generate and Test Algorithm.

=> Generate and Test is one type of Heuristic Search Method.



This algorithm is also know as British Museam Algorithm or Systematic Generate and Test or Executive Generate and Test.

- Generator : Generate the every Possible solution of a problem.

- Tester : Test the every possible solution which is generate by a Generator.

=> Algorithm:

1 Generate the possible solution, this means generating a particular point in problem space for solution.

2 Test the solution, IF it is actual point solution by comparing choose point and end point.

3 IF solution is found then Quit otherwise, Return to Step-1.

=> Property of Good Generator:

1 Complete: Gives a all the possible solution of every problem.

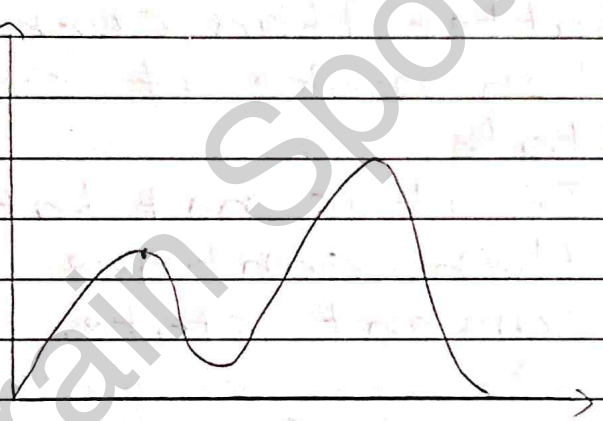
2 Inform

3 Non Redundant:

* Explain Simple Hill Climbing with its algorithm.

=> Simple Hill Climbing is use Greedy Method to solve the problem.

This method is use for Local Area Search which is not allowing the backtracking to solve the problem



=> Algorithm:

1 Evaluate the inital state.
IF it is Goal state -> Quit
otherwise, continue with inital state as a current state.

2 loop until the solution is found or No new operator left to be applied on current state.

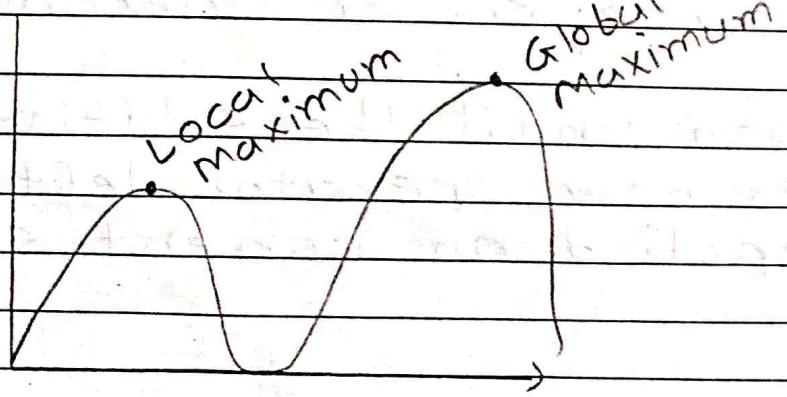
a) Select an Operator that has not yet applied ~~tha~~ to the current state and produces a new state.

b) Evaluate New State,

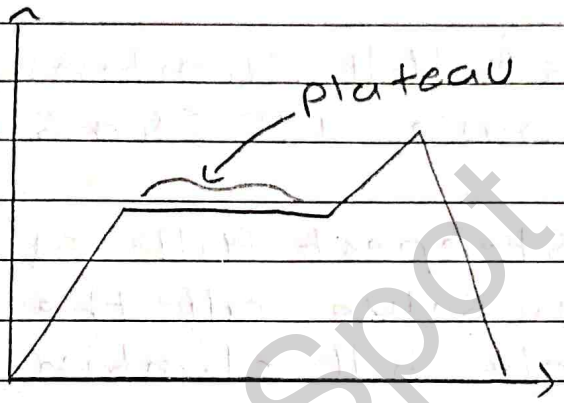
(i) If is a Goal state, Return and Quit.
(ii) If it is not Goal state but batter than current state then make it current state.
(iii) If it is not batter state, then continue with current state.

=> Problem of Simple Hill Climbing.

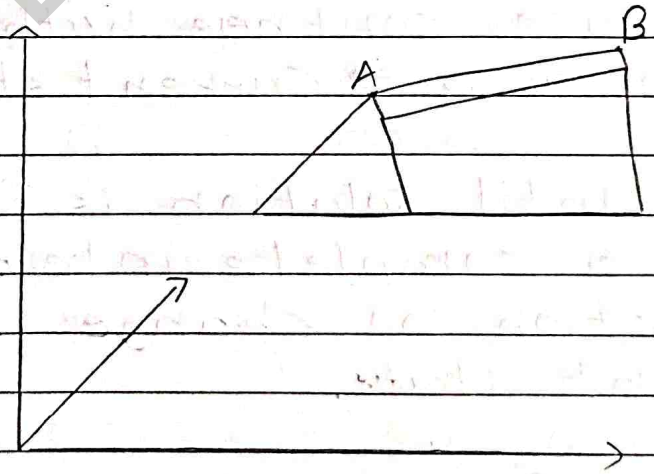
a Local Maximum: It is a state which is batter than previous state but, there is other state which is higher than it.



b Plateau: It is a flat area of search space in which all the neighbour state and current state contain same value. It does not find any direction to move against climbing.



c Ridge: It is a special form of local maximum. It has an area which is higher than its surrounding area but it can not reach in a single move.



* Explain Steepest Hill Climbing.

=> Steepest Hill Climbing is extension of a Simple Hill Climbing problem.

Using Steepest Hill Climbing, We can solve all the problem of simple hill climbing.

=> Algorithm:

1 Evaluate the inital state, IF it is goal state then Return and Quit.
otherwise, continoe with inital state as a ⊖ Current state.

2 Loop until solution is found or until a complete interaction production no change in current state.

a) Left Successor be a state search that any possible successor of the current state will be batter than successor.

b) For each operator that applied to the current state do:

(i) Applied operator and generate New state.
(ii) Evaluate New state if is goal state, return and Quit.
(iii) If it is not, compare to successor it is batter than successor to the state, if is not batter than successor than leave the successor alnoe.

=> Solution of Simple Hill Climbing Problem:

a Local Maximum: Backtrack the some early node and try to go in different direction.

Create list of possible path. So, algorithm can backtrack the search space and exprole the path.

b Plateau: Make a big jump in other direction to get new section of search space.

c Ridge : Use bidirectional search by moving in different direction, so we can improve the problem.

* Explain Block Problem with example.

=> For start or Local : For each block that has wrong structure is -1 to every block in support structure.

For Goal or Global : For each block that has correct support structure is +1 to every block.

Ex.

A	-1
D	-1
C	-1
B	-1

Start = -4

D	1
C	1
B	1
A	1

Goal = 4

1

D	-1
C	-1
B	-1

-3

A	+1

1

2

C				
B		A		D
		1		1

3

B		A		D		C
1		1		1		1

4

B			
A		D	C
2		1	1

5

C	1		
B	1		
A	1		D
3			1

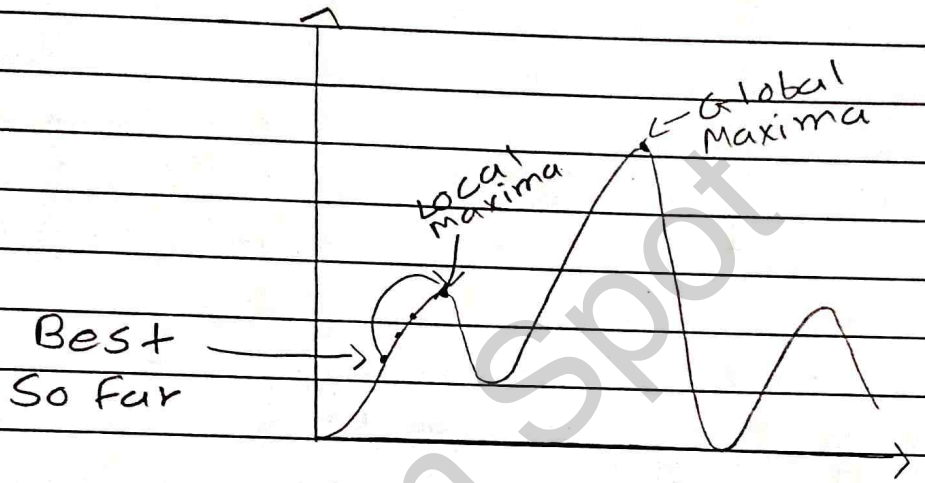
6

D	1
C	1
B	1
A	1
4	

SMVS
Page No.
Date : / /

* Explain Simulated Annealing Method with advantages and disadvantages.

=> Simulated Annealing Method is use Inform Search method to solve problem.



Simulated Annealing is use to find the Global and Local Maxima.

It allows to do backtrack and downward step to find the best point.

Annealing is a process where metal are slow cool down than it reach low state when they are very strong.

Simulated Anneling begins by creating a trial point, after that algorithm select the dest point then a current point through a probability distribution.

When Algorithm get best point than current point, then we have to change the current point by tacking long jump.

Algorithm try to find the best point using the backtrack method.

At the end algorithm, gives a Local Maxima and Global maxima.

=> Advantages:

Easy to code the complex Problem, which gives good or optimal solution of problem.

=> Disadvantages:

Process of finding the solution is slow, we can not tell the whether solution is optimal or not.

* Explain Best First Search Algorithm with example.

=) Best First Search Algorithm is follow informed search method with greedy approach.

In Best First Search, we have to use two type of list : Open and Close list.

Open list : Nodes that have heuristic Function and generated to them but not have examin in queue.

Close list : Nodes that which is already examin but we need to memory, if we want to search graph.

Those Node has low Cost, this Node is goes first in queue.

According to its Straight line distance we have to add all the Node in queue.

For Perform BFS we have to use Priority queue.

=> Algorithm For add the Node into the Queue.

1 Open Priority queue which contain the initial state.

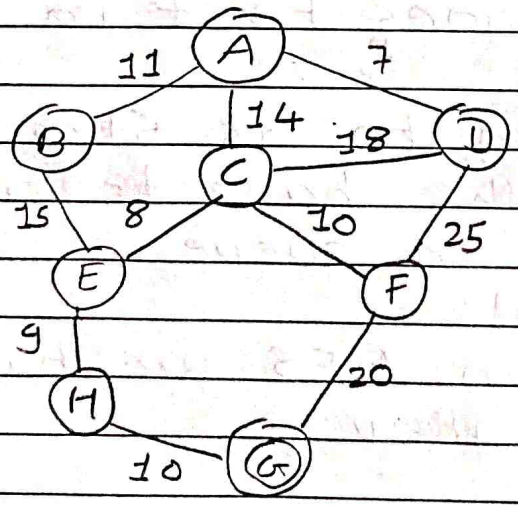
2 loop,
IF Open is empty -> Return
then Node <- Remove First Open

IF Node is Goal then -> Return to the path From initial to Node.

else, Generate all the Successor of Node and put new generated Node into Open according to there F value.

3 End.

Ex.



- F Values or Heuristic value:

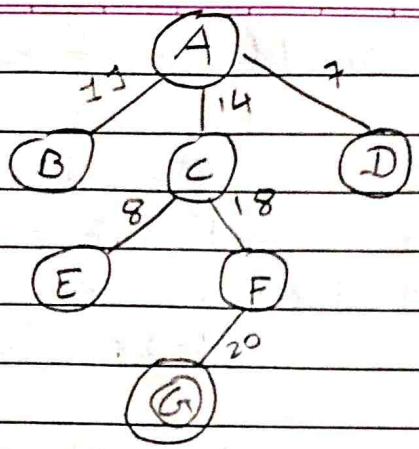
A -> G = 40	E -> G = 19
B -> G = 32	F -> G = 17
C -> G = 25	G -> G = 0
D -> G = 35	H -> G = 10

=> Here, From Node A to B, C and D Node are available, but C Node goes First in queue because the F value is low.

According to F value C, D and B Node is goes into Queue.

Open list	Close list
[A]	[ ]
[C, B, D]	[A]
[B, D]	[A, C]
[F, E, B, D]	[A, C]
[E, B, D]	[A, C, F]
[G, E, B, D]	[A, C, F]
[E, B, D]	[A, C, F, G]

Here, We reach From Node to Goal in the Graph.



=) Algorithm For BFS:

1 Start with Open list which contain initial state

2 Until the Goal is found or there are No Node left open do:

a) Pick the Best Node on Open list

b) Generate the Successor

c) For each Successor do:

1) If it is not generated before then evaluate and add to open list and record its parent

2) If it is generated before, change the parent.
If it is new path previous one in that case update the cost to getting to this

node and to any successor that this node may already have.

* Explain A* algorithm with example.

=> A* Algorithm is use Informed search method to find the shortest path.

A* Algorithm is always give a Optimal or conform solution.

A* Algorithm is the combination of Dijkstra's algorithm and Best First Search.

A* Algorithm finds the shortest path through a search space to goal state using heuristic function.

This method is finds minimal cost solution to is directed to a goal state.

In A* Algorithm, we have to Find every nodes' actual cost.

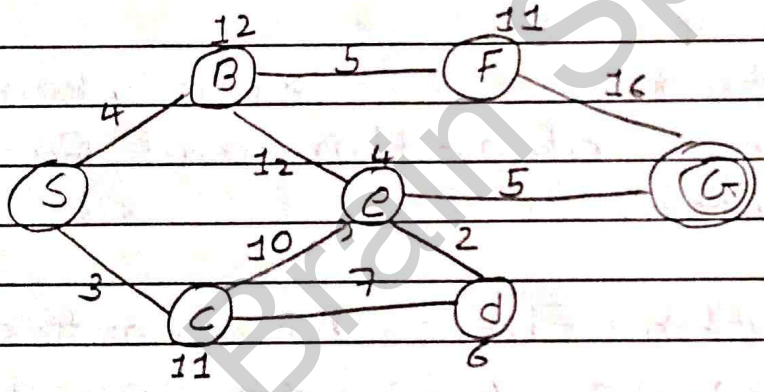
$$F(n) = g(n) + h(n)$$

Where, $F(n)$ = Actual cost From start node to goal.

$g(n)$ = Actual cost From start node to current state.

$h(n)$ = Actual cost From current state to goal state.

Ex.



We have to start From S Node.

For, $S \rightarrow B \Rightarrow F(n) = 4 + 12 = 16$

$S \rightarrow C \Rightarrow F(n) = 3 + 11 = 14$

Here, We have to select $S \rightarrow C$ path because $F(n)$ is low cost.

For,

S -> C -> e => F(n) = 3+10+4 = 17

S -> C -> d => F(n) = 5+7+6 = 16

Again, We have to select low cost F(n) path.

For, S -> C -> d -> e => F(n) = 3+7+2+4
= 16

For, S -> C -> d -> e -> G => F(n) = 3+7+2+5+0
= 17

From Start to Goal State
Path : S -> C -> d -> e -> G

=> Algorithm :-

1 Place the starting node into OPEN and find its F(n) value.

2 Remove the node from OPEN, having smallest F(n) value.

If it is a goal node
then stop and return

else, Remove the node from OPEN,
Find all its successors.

Page No.
Date : / /

3 Find the F(n) value of all success place them into OPEN and place the removed node into CLOSE.

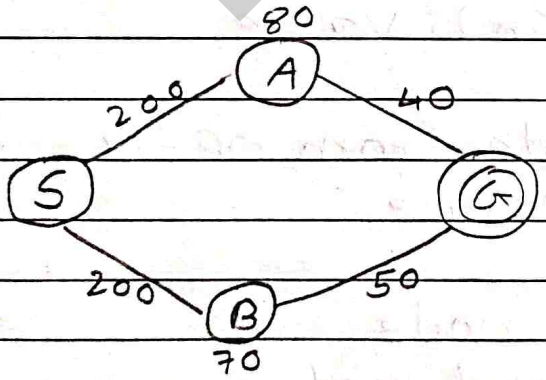
4 Go to Step-2

5 Exit.

=> How to Find a* or Greecfull DK of A* Algorithm.

If h' over estimated h by more than Θ than a* algorithm will find the solution, whose cost is more than Θ or greter than the cost of optimal solution.

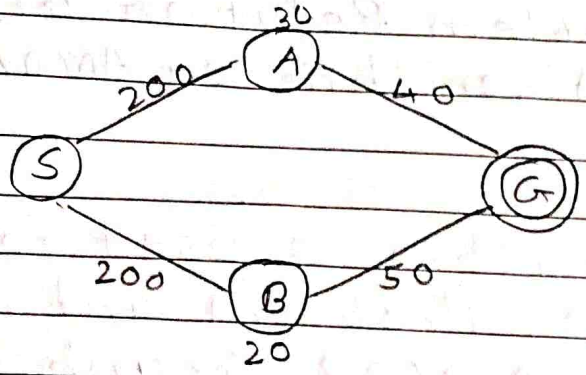
Case 1: Overestimated.



F(A) = 200 + 80 = 280
F(B) = 200 + 70 = 270

while, F(G) = 200 + 40 = 240
F(G) = 200 + 50 = 250

Case 2 : Underestimated



$F(A) = 200 + 30 = 230$
$F(B) = 200 + 20 = 220$

While, $F(G) = 200 + 40 = 240$
$F(G) = 200 + 50 = 250$

=> Advantages:

A* Algorithm is gives complete and optimal solution and used to solve very complex problem.

=> Disadvantages:

The Speed execution of A* Algorithm is highly dependant on accurancy of heuristic algorithm or Function.

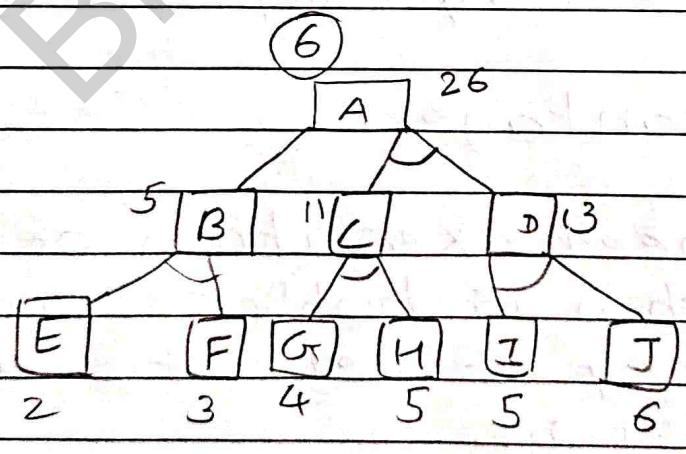
* Explain Problem Redution Graph or CAOJ* Algorithm or AND/OR Graph.

=> CAOJ* Algorithm is used for Reduction of Problem and also known as AND/OR Graph.

It is usefull for representing a solution of problem that can be salve by decompozing them into smallar problem.

OR Node represent a choice between possible decompoziable and AND Node represent given decompoziable.

Ex.



Solution : A -> B -> E -> F = 6

=> Algorithm :

1 Initalize the start node of a Graph.

2 Transverse the graph following to the current path accumulating nodes that have not yet been expanded.

3 Pick any of these nodes and expand it. If it has no successor call this value FUTILITY otherwise calculate only f for each of the successor.

4 If f is 0 then mark the node as solved.

5 Change the value of f for the newly created node to refelect its successor by back propagation.

6 Whenever possible use the most promising routes and if a node is marked as solved then mark the parent node as solved.

SMVS
Page No.
Date : / /

7 IF starting node is solved or value is greater than FUTILITY
stop
else, Repeat From 2.