Artificial Intelligence

Books - Rich & Knight
Luger

Concept of A.I. - AI is a branch of computer science that deals with the study and creation of computer systems that exhibits some form of Intelligence.

By "Intelligence" we mean

1) → system that learns new concepts and tasks.
2) → system that can reason and draw useful conclusion about the world around us.
3) → System that can understand a natural language
4) → System that can perform other types of tasks (skills) that require human type of intelligence

AI Tasks Domain

- Mundane tasks
  - perception (vision & speech)
  - Natural language understanding
  - Common reasoning
  - Robotics
- Formal tasks
  - Game playing (Chess, 8-queen problem, tower of Hanoi problem)
  - Mathematics
    - Geometry
    - Calculus
- Expert System

Engineering field | Scientific Analysis | Medical & financial Analysis

ALAN TURING's Test :- * In 1950, Alan Turing published an article in the "Mind Magazine" which proposed a question "Can Machine think"?

* Turing Test :- "A computer is programmed well enough to have a conversation with an interrogator & passed the test if the interrogator cannot discern if there is a computer or human at the other end".

There are 3 actors :-
a) A Male / human
b) A Machine
c) An interrogator

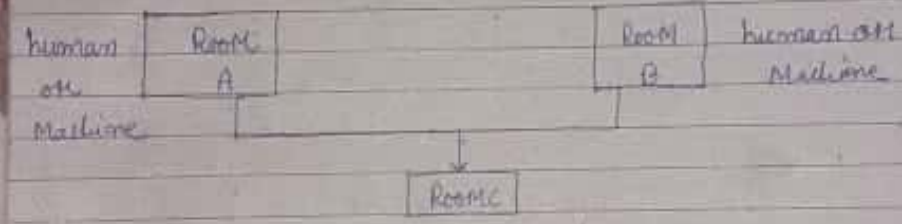


Fig: Turing Test

AI Approaches

• Problem Solving

1) Problem Specification :- To formulate any problem, four components are defined in a formal way.

a) The initial state (starting state)
b) State space i.e. description of all possible states reachable from the initial state.
c) Goal test that determines whether a given state is a goal state
d) Path cost is a function that assign a numerical cost of each path.

2) Problem Solution :- Any problem can be solved a few steps

Step 1 → Define a state space
Step 2 → Identify initial state
Step 3 → Specify goal states
Step 4 → Specify set of rules

3 Search space problem :- 1) Eight tile puzzle problem
2) Travelling salesman problem
3) Water jug problem

1) Eight tile puzzle problem :-

3	8	1
6	2	5
	4	7

Start State (Initial)

→

1	2	3
8		4
7	6	5

Goal State (Final)

Problem Formulation:- The Eight tile puzzle consist of 3/3 square frame board which have moveable eight tile numbered as 1 to 8 one square is empty, allowing the adjacent tile to be slided. The object is to find a sequence of tile movements that leads from standing configuration to goal configuration as shown in fig.

Standard formulations:-

1) States:- It specifies the location of each of the 8 tiles and the blank is one of the nine squares

2) Initial State:- Any state can be designed as initial state

3) Goal:- Many goal configurations are possible

4) Legal moves:- a) blank moves left
b) " " right
c) " " up
d) " " down

5) Pathcost:- Each step cost 1, so the path cost is the numbers of steps in the path.

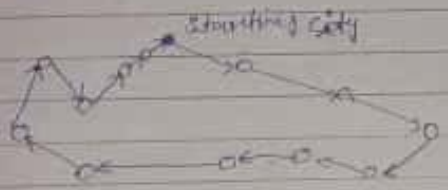
ii) Travelling salesman problem:-

Problem formulation:- It involves n-cities with paths connecting the cities. A tour is any path with begins with some starting city, visits each of the other city exactly

once & returns to the starting city.

Objectives:- Find a minimal distance tour.

* Brut The explore all tours requires an exponential amount of time.
* If there are N cities, then number of different paths among them will be N factorial.



all cities are connected each other

Fig:- TSP

Solution Case 1:- If combinational explosion is used, then paths to be examined are [5! = 120]

Case 2:- If nearest neighbour heuristic is used, then number of paths to be examined are [5! = 25] only

Standard formulation

1) States:- location specified by city
2) Actions:- Driving along the roads between cities
3) Goal:- city B.
4) Path cost:- Total distance are expected travel time.

Water Jug problem :-

Problem domain :- 1) 2 jugs given, a 4 gallon jug & a 3-gallon jug.
2) No measuring markers on jugs
3) Pump is used to fill the jugs with water.

State space :- describe as set of ordered pairs of integers

(x,y) such that $x = 0, 1, 2, 3, \text{or } 4$
$y = 0, 1, 2, \text{or } 3$

x → No. of gallons of water in 4-gallon jug.
y → " " " " " " 3-gallon "

Start state : (0,0)
Goal state : (2,n) where, n is any value.
* This problem has 1 initial state & many goal states

Start :

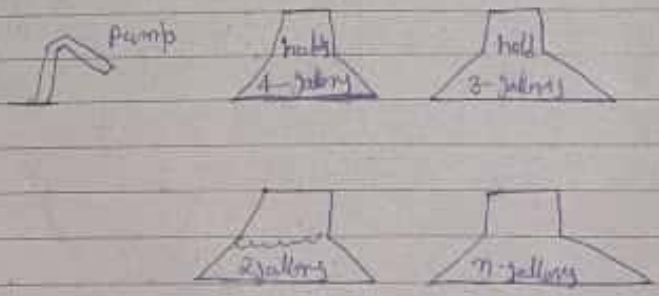


Fig: A water jug problem

Production Rules for Water jug problem:

1) $(x,y) \rightarrow (4,y)$ if $x<4$ Fill the 4-gallon jug.
2) $(x,y) \rightarrow (x,3)$ if $y<3$ " " 3-gallon ".
3) $(x,y) \rightarrow (x-d,y)$ if $x>0$ pour some water out of 4-gallon jug.
4) $(x,y) \rightarrow (x, y-d)$ if $y>0$ pour some water out of 3-gallon jug.
5) $(x,y) \rightarrow (0,y)$ if $x>0$ Empty 4-gallon jug onto the ground.
6) $(x,y) \rightarrow (x,0)$ if $y>0$ " 3- " " " " "
7) $(x,y) \rightarrow (4, y-(4-x))$ if $x+y \geq 4$ and $y>0$
Pour water from the 3-gallon jug into 4-gallon until 4-gallon jug is Full.
8) $(x,y) \rightarrow (x-(3-y), 3)$ if $x+y \geq 3$ and $x>0$
" " " " 4-gallon jug " 3-gallon
" 3-gallon " " ".

9) $(x,y) \rightarrow (x+y, 0)$ if $x+y \leq 4$ and $y>0$
Pour all the water from 3-gallon jug into 4-gallon jug
10) $(x,y) \rightarrow (0, x+y)$ if $x+y \leq 3$ and $x>0$
Pour all the water from 4-gallon jug into 3-gallon jug

11) $(0,2) \rightarrow (2,0)$ Pour 2 gallon from 3-gallon jug into 4-gallon jug.

12) $(2,y) \rightarrow (0,y)$ Empty 2 gallon from 4-gallon jug on the ground.

One solution for watering problem :-

Water in 4 gallon jug (x)	Water in 3-gallon jug (y)	Rule Applied
0	0	—
0	3	2 Rule
3	0	7 Rule
3	3	2 Rule
4	2	7 Rule
0	2	5 Rule
2	0	9 Rule

Second Solution for watering jug problem

Water in 4-gallon jug (x)	Water in 3-gallon jug (y)	Rule applied
0	0	—
4	0	1 Rule
1	3	8 Rule
1	0	6 Rule
0	1	10 Rule
4	1	1 Rule
2	3	8 Rule

Production System:-

* These systems were proposed by Emil Post in 1943.
* They are also know as
  - Inferential system
  - Rule base systems
  - Productions

* For describing & performing search operations in AI programm it is useful to structure system

* Rules of production system

Rule 1:- Production system (PS) representation both knowledge as well action.

Rule 2:- It provide a language in which the representation of expert knowledge is very natural.

Rule 3:- PS provide a heuristic model for human behaviour.

Rule 4:- PS model strong data driven nature of intelligent Action. whenever new inputs enters the database, the behaviour of system changes.

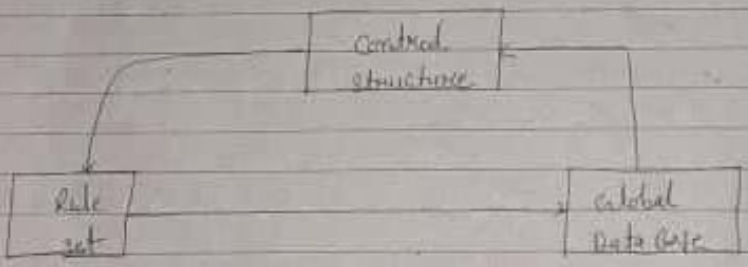
Rule 5:- Production system allow us to add new rules easily without disturbing the rest of the system.

Architecture of a production system:-

It Comprises of 3 main Components
1) Rule base
2) Global DB
3) Control structure

* The process of transforming a problem statement into these components of the production system is called as problem representation

The effort required to solve a problem does depend on the problem representation scheme



Rulebase:- Product systems are forbidden KR- structure.

→ They are Conditional IF - Then branches only.
→ These rules apply on global DB.
→ Each rule have a Pre condition.
If pre condition of the rule is satisfied then only the corresponding is applied to state

Solution → Action

Such problem situation action rules are often called as production rules or if-then rules and a system which uses this form of knowledge is called as a production system.

Each of the if-condition is called as clause (s). a set of clauses joined by logical AND is called as HORN order clause.

Each production rule consists a chunk of knowledge. The action part of one production rule become the condition part by of another production rule. this is known as Network ing of rules are production rule chain.

For example:- University grading system might be as follows

IF the marks obtained are less than 75 THEN declare the student as in First class

Associated with these production Rules are certainity factor/ Probability factors

IF car doesn't work THEN battery is down or Battery connection are poor

* We can use an certainity factor & Key that this is a strong evidence (0.9) that battery is down

* Range of this factor is taken from 0 to 1
  $0 \leq$ Probability ≤ 1

This type of reasoning is called as Inexact Reasoning

The syntax of this rule is

IF <antecedent -1>
<antecedent -2>
.
.
<antecedent -n>
THEN
<consequence 1> / (with certainty c1)
<consequence 2> / ------ (c2)
<consequence n> / (with certainty cn)

* Antecedent part is c/a condition part / LHS
* Consequence part is c/a Action part / RHS

→ When LHS matches then the designating condition is (production rule (action) will be executed also note here that the application of the rule changes the database.

Searching techniques:-

2 types → Uninformed search / blind search / Unguided
Informed or heuristic search / Guided.

Global Database:→

It is a central data structure used by the production system.
→ Database may be simple as a small matrix of numbers or as complex a large relational index file structure.
→ Rule interpreter apply the rules to the database against

again until the description of goal state is produce. This process of identifying the rule which are fired until some sequence of them is found which produce a database satisfying the termination condition (goal) is called as searching processes.

Uniform search:- In this search there is no order in which solution path are considered that is it uses no domain specific information to search a solution in the search space. It is also called as brute force search because it it examine every node in the tree until a solution is found.

These also makes some assumptions

1) A procedure must be there to find all successors of a given node
2) The state space graph is a tree
3) Whenever a node is expanded, creating a node for each of it's successor, the successor node contains pointer back to the parent node.

Breadth - First Search (BFS)

* Most simple search techniques
* Root node is expanded first, then all of its successors are
* expanded & then their successors and so on.
* BFS uses a Data structure that works on FIFO principle.
* This queue will hold all generated but still Unexpanded nodes

Algorithm:-

① Put the start (initial) node on a list, called OPEN of unexpanded nodes.
If the start node is goal node, a solution has been found.

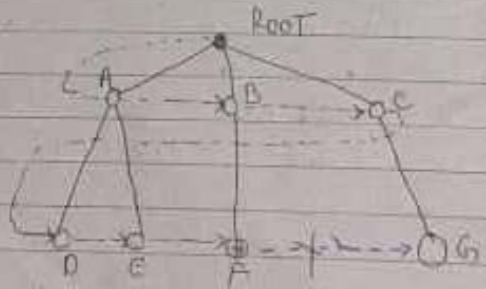
② If (OPEN is empty) OR (OPEN = GOAL), terminate search.

③ Remove the first node, a, from OPEN & place it in a list called CLOSED, of expanded nodes.

④ Expanded node, a. If it has no successors then goto step-2.

⑤ Place all successors of node, a, at the end of OPEN.

⑥ If any of the successors of node, a, is a goal state then solution is found.



(GOAL NODE)

Fig:- Search tree for BFS

Advantages of BFS:-

→ BFS will never get trap exploring a blind alley.
→ It is guranted to find a solution if one exist.

Disadvantage:- all nodes are to generated in BFS. so even unwanted nodes are to be remembered (stored in cache) out which is of no practical use of the search.

Depth first search:- DFS begains by expanding the initial node that is using an operator, generate all successors of the initial node and test them.

→ It is characterized by the extension of the most recently generated or deepest node first.

→ It needs to store the path from the root to leave nodes as well as unexpanded nodes.

algo:-
1) Put the initial node on a list, START-LIST
2) If (START-LIST is empty) or (Start = Goal) then terminate (end) search.
3) Remove the first node from start list. call this node a.
4) If (a = Goal) then terminate search with success.
5) Else if node a has successors, generate all of these & add them at the begging of ~~the first~~ START-LIST.

6) Can do state.

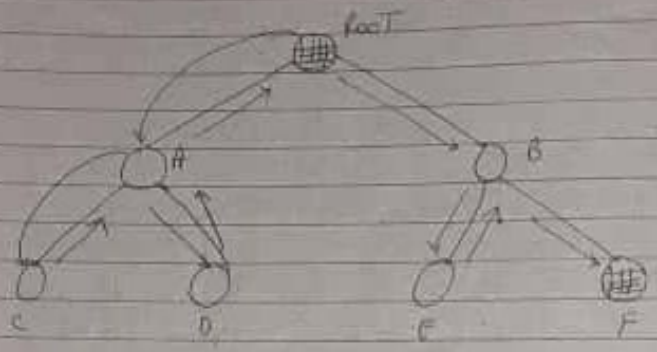


Fig:- Search tree for DFS.

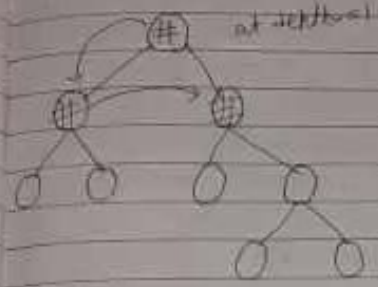
Depth First Iterative Deepening Search:→

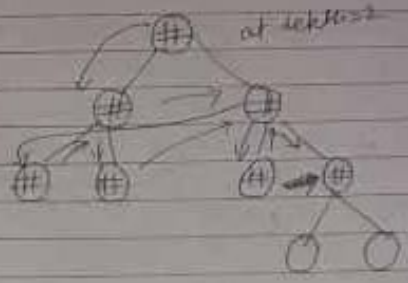
* It is a general strategy used by combination with DFS that finds the best depth limit.
* It begins by performing DFS to depth of one, then depth of two & so on.
* It combines benefits of BFS & DFS-DFS.

Algo:- ① Start
② Initiate $d \leftarrow 0$
③ do

$d \leftarrow d+1$;
DFID (root, d);
Until, success; SUCCESS;

1) End.





- This algo starts with Smallest d = 1
- It then increments d on a loop on encountering Failure until a success is found.
- It finds a shortest path as it expands at all nodes at a given depth before expanding nodes at a greater depth.

Informed or heuristic search techniques:-

→ The term heuristic is used for algorithms which find solution among all possible ones.

→ heuristic is a rule of thumb or judgement techniques that leads to a solution but provide no garuntee of success.

→ They help to reduce the no. of alternatives from an exponential number to a polynomial number.

→ We get a solution a reasonable amount of time.

① Generate-And-Test Algorithm:-
This also works in two modules:-

1) Generation Module:- It create the possible solutions

2) Tester module:- It test each of the proposed solution either accepting or rejecting that solution.

Generate & test algo.

1)

Hill climbing algo :- it like DFS where the most promising node is selected for expansion. When the children have been generated alternative are generated using heuristic function. The path that appears most promising is then chosen.

* Please note that a new state has to be better, meaning that if we considered the cost function then it means a lowers value and if we considered the objective function it means a higher value.

Hill climbing is some time known as Greedy local search because it grabs a good neibour state without think ahead about where to go next.

i) Local Maximum :- It is a state which is better than all it's neibour but is not better than some other state which are far away.

ii) Plateau :- It is a state area of search space in which a whole set of neibouring state have the same value. It is not possible here to determine the best direction in which to move by making local comparisions. It also called flate local maximum or Shoulder.

iii) Ridge :- It is special kind of local maximum. It is the area of search space which is higher than surrounding area and it's self as a slope.

Algorithm for Hill climbing:-

Step 1:- Put a first initial node and a list start_list

2- if list (start list is empty) start_list = Goal) then terminate search

3- Remove the first node from the start list. Call this node, n.

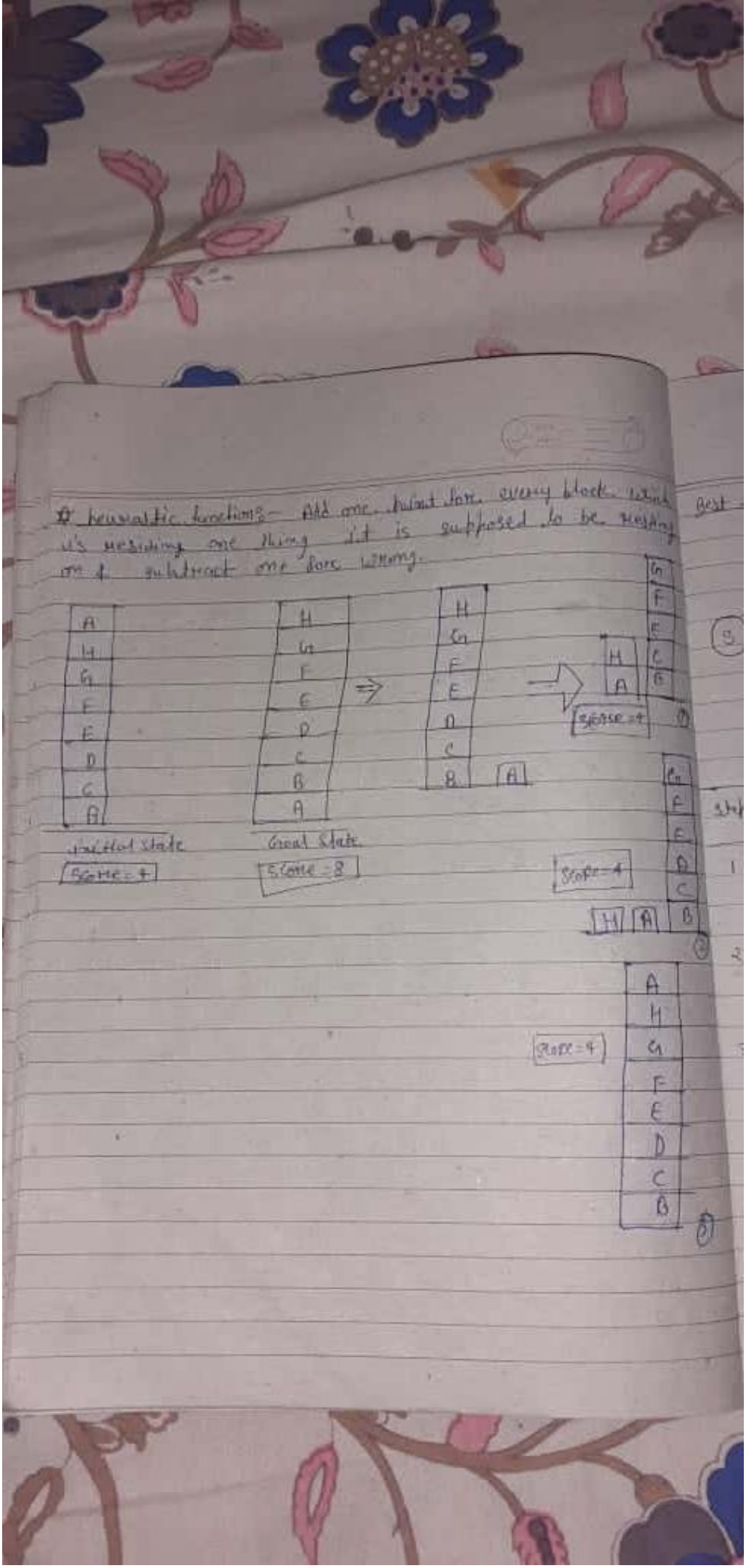
4- If (small n = GOAL) then terminate search with success.

5- as if node, n, has successors then generate all of them.
Find how each they are from the Goal node.
Sort them by the remaining distance by the Goal and add them to the beggining of the start list

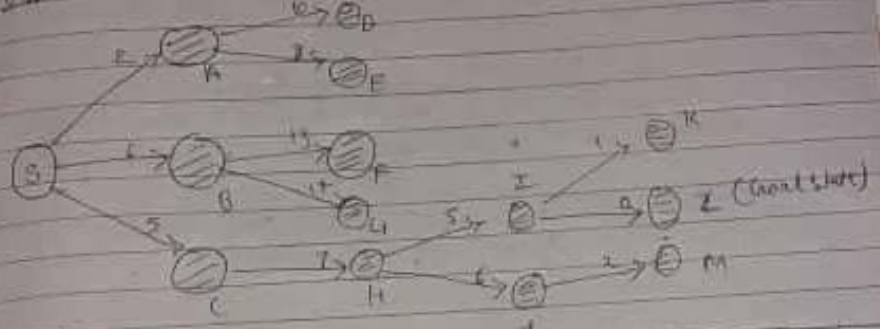
6- go to step-2.

Block problem:-

- It has 8 Blocks
- A set which are similar & equal in size
- Two operators can be used
  - a) pick up one block & put it on table.
  - b) Pick up one block & put it on another



Best first search / Greedy search



Steps	Node being extended	children or successors	Available nodes to search	Next chosen
1	S	(A:2), (B:4), (C:5)	(A:2), (B:4) (C:5)	(A:2)
2	A	(D:10), (E:8)	(B:4), (C:5) (D:10), (E:8)	(C:5)
3	C	(H:7)	(B:4), (E:8), (H:7) (D:10)	(B:4)
4	B	(F:13), (G:14)	(D:10), (E:8), (H:7), (F:13) (G:14)	(H:7)
5	H	(I:5), (J:6)	(D:10), (E:8), (F:13), (G:14) (I:5), (J:6)	(I:5)
6	I	(K:1) (L:0) (M:2)	(D:10), (E:8),	

		(Arad), (F:13), (O:14) (T:6), (R:1), (L:0), (M:2)	Goal state is found

The Approach is to select a single path at a time which is more promising one than the current path.
•The most promising node is chosen for expention that is all of it's successors are generated

* Again the most promising node is chosen and this algorithm uses a heuristic function known as evaluation function which is an indication of how close a node is from the Goal node

Algorithm for best first search :-

1 — place the starting node S (initial state) on the Queue

2 — If the Queue is Empty, return failure and stop

3 — If the first element on the queue is the Goal Node g, return success and stop otherwise

4 — Remove the first element from the queue, expand it and compute estimated Goal distance for each child place the children on queue (at either end) and arrange all queue element in ascending order corresponding to Goal distance

from the front of the Queue

5. Return to step 2

Constraint satisfaction:-

Problem

```
    S E N D
+   M O R E
-------------
M O N E Y
```

Constraints

(1) Values are to be assigned to letters from 0 to 9 only.
(2) No two letters should have the same value
(3) If the same letter occurs more than once, it must be assigned the same digit each time
(4) The sum of the digits must be as shown in the problem

Solution:

The solution process proceed in cycles. At each cycle two important things are done

(1) Constraints are propagated by using rules that correspond to the properties of arithmetic. A value is guess for some letter whose value is not yet determined, as this affects the effort involved in the search.

(2) Most constraint are added due to propagating constraints