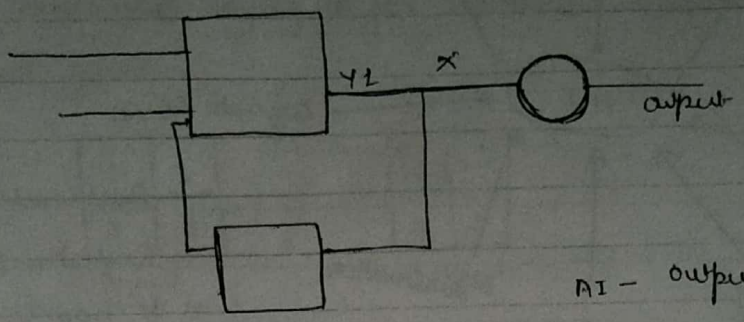
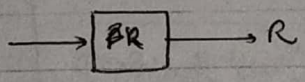


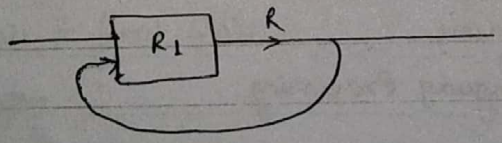
AI – output fixed
- input fixed.

Production Rules →

(1) It produces the output and the output should be in forwarded mode.



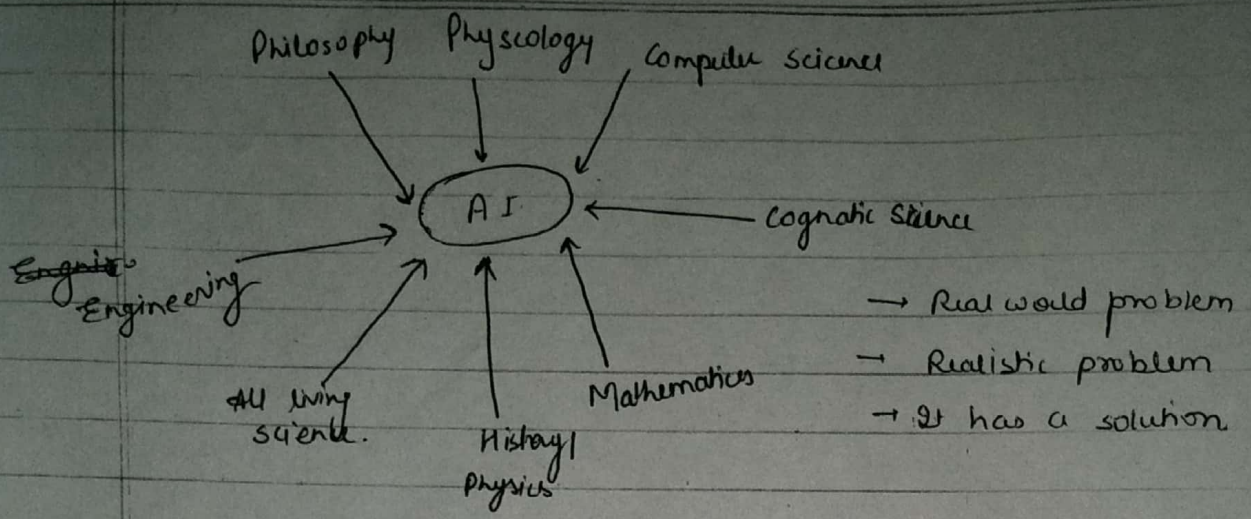
(2) It may be possible that we apply a rule and it is back to previous stage but firing of the rule at that stage is not ~~neces~~ possible.



(3) We can fire a rule n number of times in any sequence. Only necessary condition is moving forward.

6/01/2018

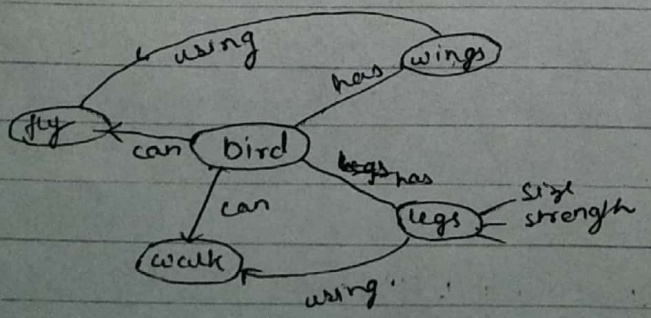
→ Natural language Processing (NLP) - Game playing system.
→ Recognition of ~~device~~ Images / Objects.
→ Proving of mathematical theorem.



AI is an interdisciplinary system

1- Knowledge representation
2⇒ Searching Technique.
3→ Learning System.
4→ Uncertainty Management – This is key factor of

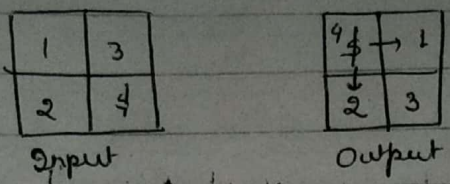
- → Game playing
- → Image procesing
- → Robot system
- → Expert system
- → Theorem proving
- → Natural Languag Processing.



Semantic Network representation of knowledge.
Space State Representation of a problem.

8/01/2018

→ We are not saying o for optimistic solution but a well defined solution.



→ In programming, there should be a temporary space so that final state could be found out.

Rule Set.
- B R
- B L
- B U
- B D

→ At a given time, only one rule has to be fired.

→ We represent the initial state as root. ~~and~~ and states are represented on a tree.

Algorithm:

1. State := Initial state.
2. While state ≠ final state

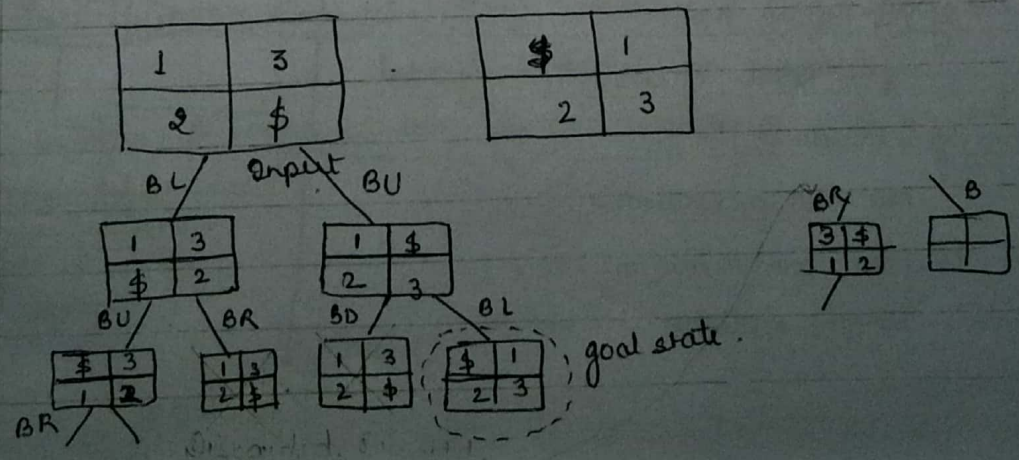
   do ~~begin~~

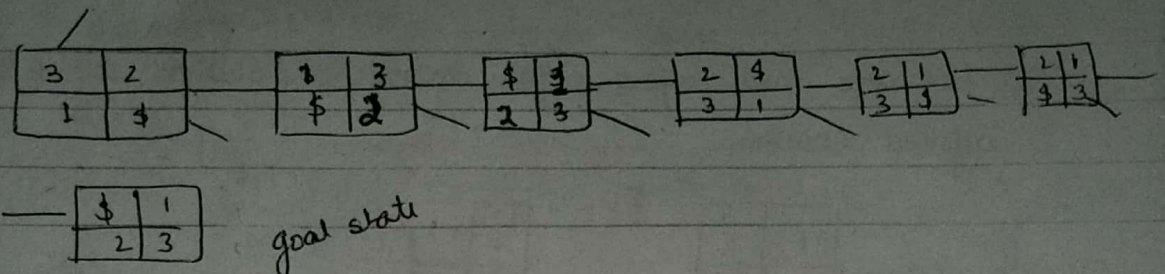
   begin

   (a) Apply operations from the set {BR, BL, BU, BD} so that it generates a state.

   (b) If state generated = any of the existing state, discard it.

   End while.





→ Write a C program to implement this. Through keyboard we have to give input and output.

→ Processing generates delay in the system. 17/01/21

→ If we want to generate a delay in the system we take the longer path and if we don't want to generate delay then we take the shorter path.

→ If rules are ~~not~~ well defined, then we get solution always.

→ If a rule is fired, we must get a new state.

Commu

~~Data Set~~ ~~Map~~ Components required for rule set - for designing expert system

(1)- Data Set

(2)- Knowledge Set

(3)- What rule has to be fired.

→ Rule set for no class ~~top~~ today.

Input parameter - (Cloudy Temperature, Wind).

R-1 If (it is hot) AND
(the sky is cloudy) THEN
it will rain.

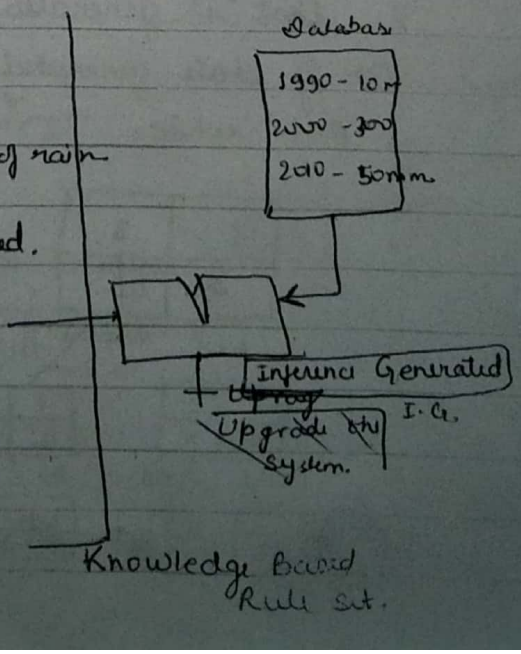
R-2 If (it Rains) AND
(the Road of GKP) THEN Flooded.

R-3 If (Flooded) THEN

R-4 No transportation.

R-5 IF (No transportation) THEN

R-6 ~~th~~ Not reach to class room.



Decisi

→ It is a fuzzy logic based concept design.
→ Because From the DB, we took level of water but not the temperature.

Q- Design a system with real life problem to design a weather forecasting system supported by database, searching based, logic and Knowledge concept.

17/01/2017:

Production Rules-

Production system are classified into two types:-

- Commutative system - $R = R_1 \cup R_2 \cup \dots R_n$
- Decompositable system

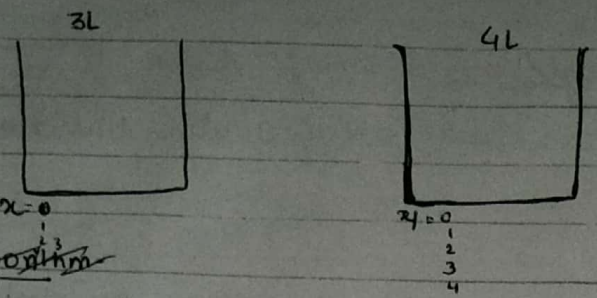
Commutative System- For each given ruleset are under working memory, wh. the following condition should be:-

(a) - Freedom in orderliness of rule.
(b) - Selected output can be prerequisites for the next stage.
(c) Pre condition attaining of the goal. If a pre-condition of a goal is satisfied by working memory, Wn before firing the rule then it should remain satisfactory after firing the rule

Example- Subjects required to be cleared to pass on to next year. semester

Decomposible System- Example -- Two unmarked jugs and we have to find maximum capacity.

- System through which water can be poured in and out is provided.

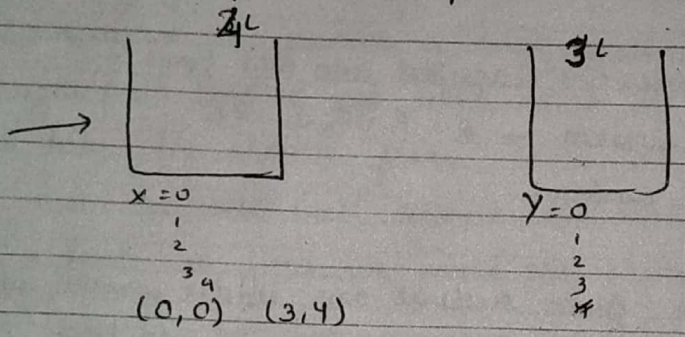


Algorithm

- Make the jar empty
- Fill the jug completely.
- Throw water

Algorithm

- Here we represent a problem through space state diagram.



19/01/2

- The state space of the problem can be described as a set of ordered pair integer (x, y)

$x = 0, 1, 2, 3, 4$

$y = 0, 1, 2, 3, 4$

and we want an output as $(2, 4)$

1- $(x, y \mid x < y) \rightarrow (4, y)$ Fill the 4L jug.

2- $(x, y \mid y < 3) \rightarrow (x, 3)$. Fill the 3L jug.

3→ $(x, y \mid x > 0) \rightarrow ((x-1), y)$. Pour some water out 4L.

4→ $(x, y \mid y > 0) \rightarrow (x, (y-1))$ Pour some water out 3L.

5→ $(x, y \mid x > 0) \rightarrow (0, y)$. Pour some water out 3L.

6→ $(x, y) \; y > 0) \rightarrow (x, 0)$

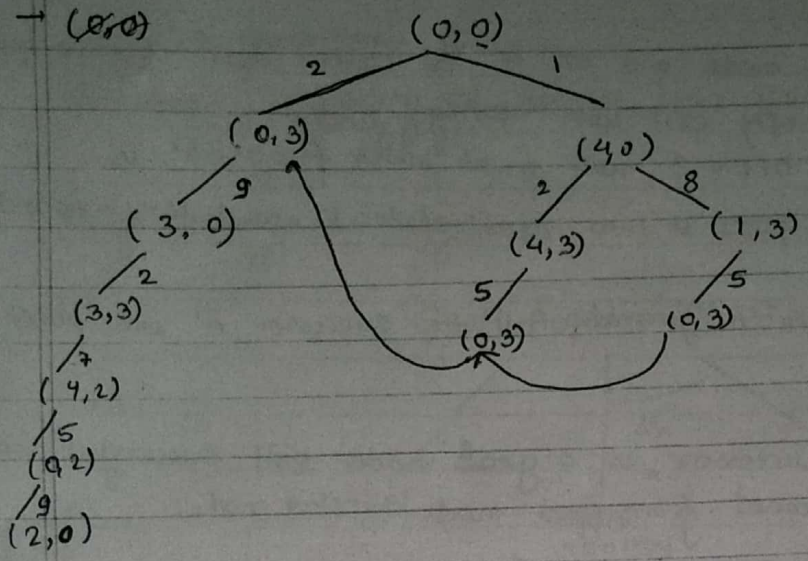
7→ $(x, y \mid x + y \geq 4 \wedge y > 0) \rightarrow (4, y - (4 - x))$

8→ $(x, y \mid x + y \geq 3 \wedge x > 0) \rightarrow (x - (3 - y), 3)$

9→ $(x, y \mid x + y \leq 4 \wedge y > 0) \rightarrow (x + y, 0)$

10→ $(x, y \mid x + y \leq 3 \wedge x > 0) \rightarrow (0, x + y)$

→ (0,0)



→ 4L-3L. Run.

19/01/2018

→ A* Algorithm →

→ ~~f = g+h~~ $f = g+h$.

Functions are known as value parameters.

$h(x) = 2$ when $0 < x < 4$ AND $0 < Y < 3$

$h(x) = 4$ when ~~0 <~~ $0 < x < 4$ OR $0 < Y < 3$.

$h(x) = 10$ (1) When $X = 0$ AND $Y = 0$

(2) When $X = 4$ AND $X = 3$

$h(x) = 8$ (1) When $X = 0$ AND $Y = 3$

(2) When $X = 4$ AND $Y = 0$.

Algorithm →

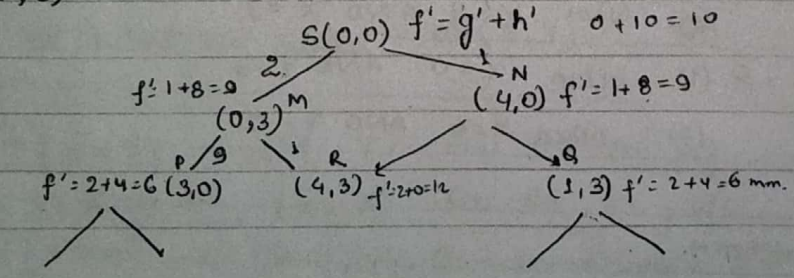
→ It is convinient two list of nodes. Node type defined as open and closed.

→ Node on the open list are the node that have been generated but not yet expanded but node in the closed list are the node that have been expanded and their children are available for seaching the values

(1)- Put the start node s on a list called open of unexpanded
(2)- If open is empty exit with repeating function.
(3)- Remove from OPEN a node n at which $f' = g' + h'$ is minimum. and place it into CLOSED to be used for expanded node.
(4)- Expand node n. generate all the successor n' with point back to n.
(5) If any of successor (n's) is a goal node exit successful and solution obtained from goal node to start node.
(6)- For every successor n' of n,
(a) Calculate $f'(n') = g' + h'$
(b) If n' either OPEN or CLOSED then Add it to ~~open~~ OPEN. Assign the newly computed value i.e. $f'(n')$ of node n'.
(c)- If n' already exist on open or CLOSED. Compare it with newly computed value $f'(n')$ will ~~h(n)=8~~ with the previously assigned to n'.

→ ~~S(0,0)~~

Expand it.



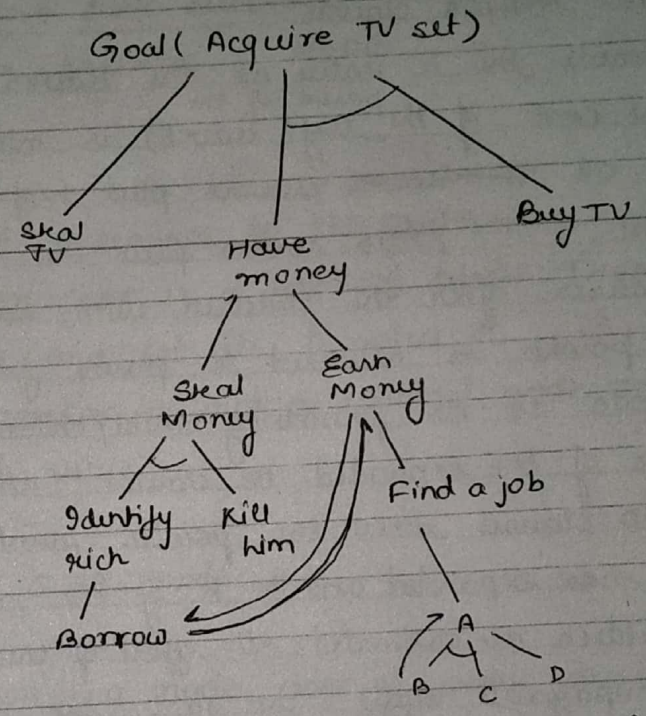
→ Execute this using heuristic function.
→ Expand the 4×4 problem 4×4, diagonal rules can be added to see the solution exist.

31/01/2018

AND-OR Algorithm-

Reverse forwarding or reverse bias algorithm.

Goal (Acquires TV set-)



~~Find a job~~

If (B OR C OR D) THEN A
IF((B AND C) OR D) THEN A

GOAL (Male, Female)

Male	Female	(1) Both are unmarried
Age	Age	(2)- He or She can be divorced.
Education	Education	
Religion	Religion	
Salary	—	
Family Back Ground.	FBG	
Height	Height	
Weight	Weight	
Knowledge.	Knowledge.	

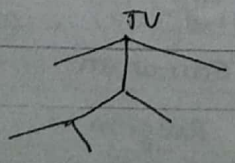
- Design a system, a male is married to female
(1) Anyone doesn't have a pair.

AO* Algorithm Concept →

(1)- Given the Goal node, node, here after & called on starting state and forward all possible off springs of the starting state such that Goal can be derived from AND/OR clauses.

(2)- Estimate the h value at the leaves with min h. The cost of the parent of the lay (leaves) is min of the cost of the OR clauses plus one or the cost of the AND classes plus the number of AND clauses. After the children with min h are estimated a pointer is attached to parent from the parent node to its promising child/children.

(3)- One of the expanded OR clauses / the set of unexpanded AND clauses where the pointer point from its parent is now expanded and the h' of the newly generated children are estimated. The effect of this h' has to be propogated upto the root by RE-CALCULATING the f' of the parent or parent of parent of newly created child/children clauses through a least cost. Thus the pointer (h) may be modified depending on the Revised cost of the exisking clauses.



Algorithm →

Procedure AO* →

Begin

1→ Given the goal node INT in the graph G, evaluates h' at INT.

2→ Repeat

(a)- Trace the marked arcs from INT of any such exist. and select one of the unexpanded node name NODE.

(b)- If NODE cannot be expanded then assign FOTLITY on the h values of Node no solution exist.

Else FOR each such successor called SUCCESSOR which is not an ancestor of NODE

do

begin

i) Append successor to the graph G.

ii) If successor is a terminal node. THEN label it SOLVED and set its h' value 0.

iii) If successor is not a terminal node THEN estimate its h' value.

END;

(c) Initialize S to node.

(d) Repeat.

(1) Select from S a node, none of whose descendent belongs to S called as current and move it from S.

(2) Estimate the cost of each of the arc merging from the current, the cost of each arc is equal to the sum of h value of each of the nodes at the end of the arc plus the cost of the arc itself. The new h value of the current is the minimum of the cost just computed from the arc emerging from it.

(3)- Label to the best part out of current by marking the arc that has the least cost computed in the last step. (just above the computed value).

(4)- If all the nodes connected to current through the new marked arc have been solved as a solution, we say that the solution exist.

(5)- If current is marked solved or the cost of the current was changed then propagate it new status

set of ?

back to it across the tree and add to its ancestor

6) Repeat 4) Until current of S is empty.

CLOSE