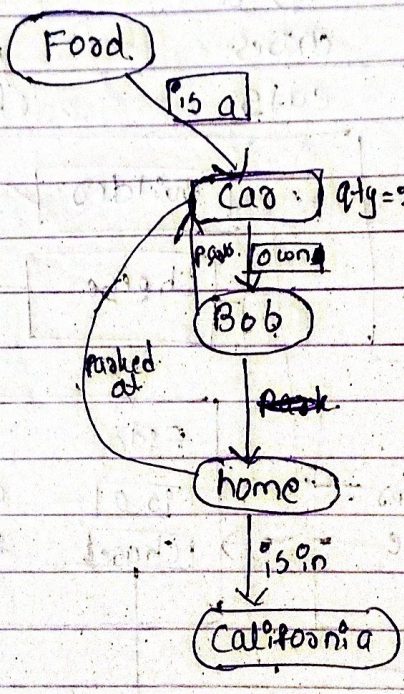
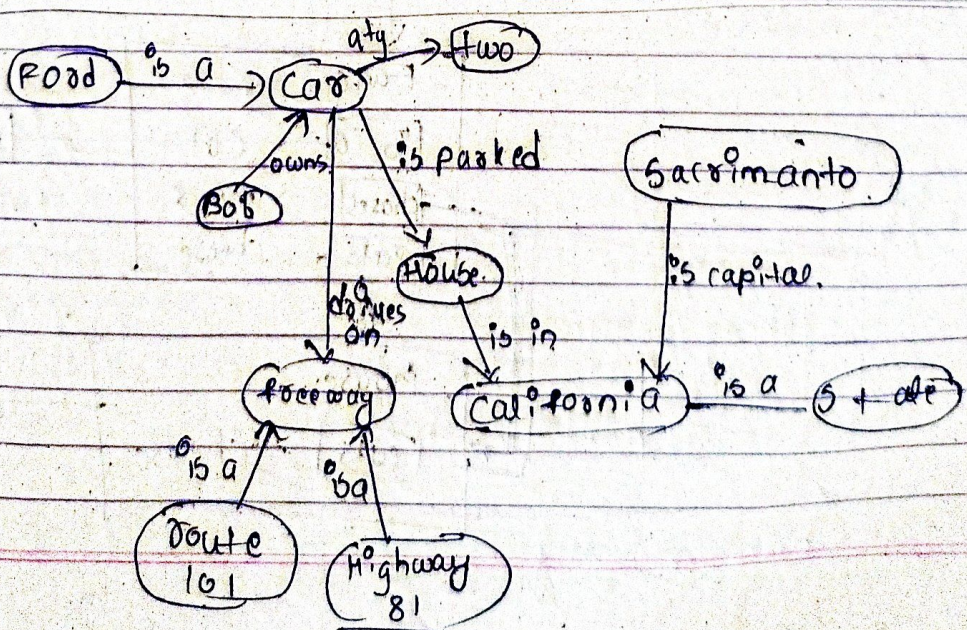
20/03/2024

AI and ML

Ford is a type of car
Bob owns two cars
Bob parks his car at home.
His house is in california which is a state.
Sacramento is a state capital of California.
Cars drive on the free way such as routelol & highway81.
Convert these statments into semantic net.





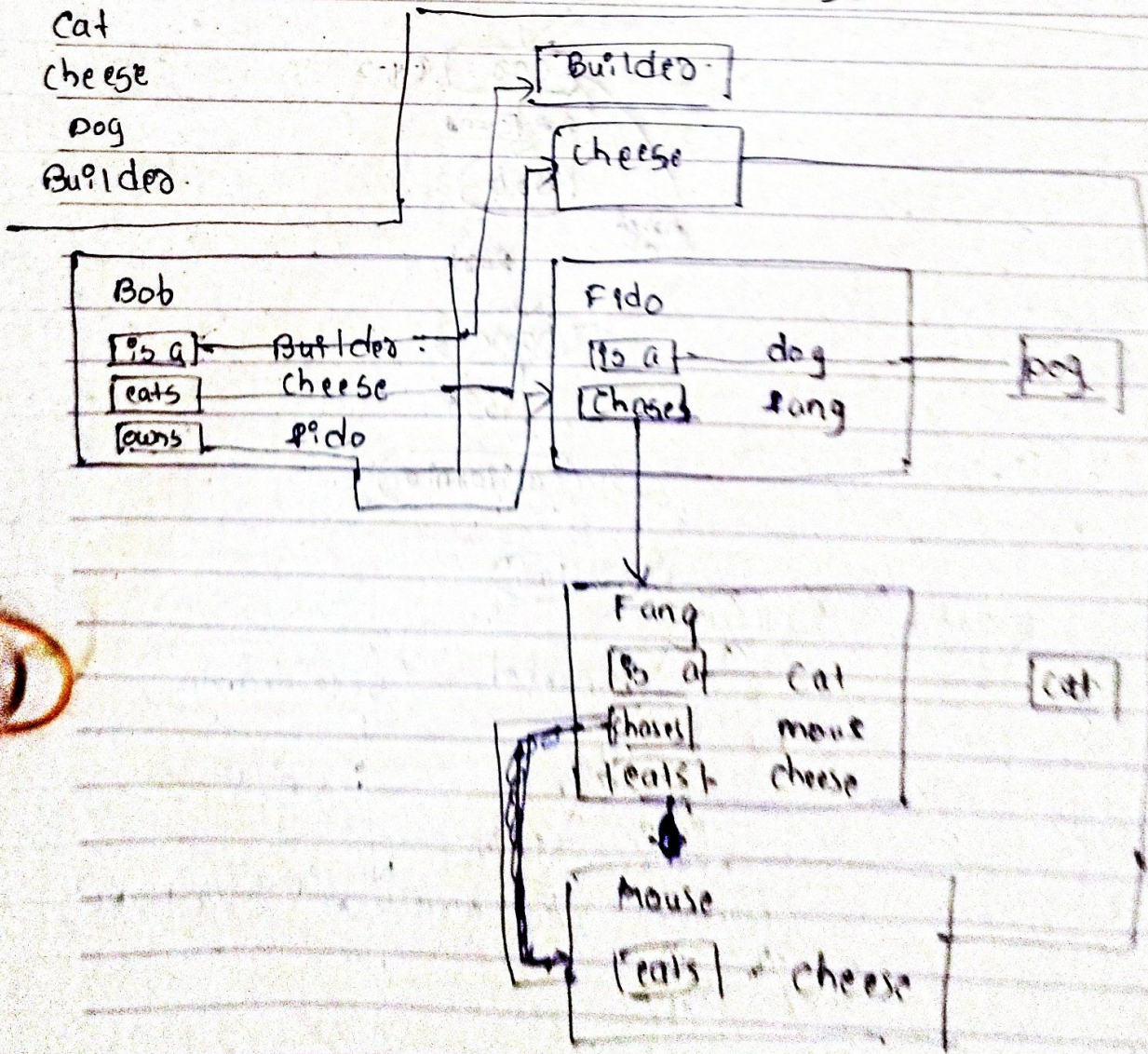
Turing test & chinese storytelleo } imp.

Frames

set of frames, slot & slot values,

Frame name	slot	slot value
Bob	is a	Builder
	eats	cheese
	Owns	fido
Fido	is a	dog
	Chases	fang
Fang	is a	Cat
	chases	mouse
mouse	eats	cheese

Cat
cheese
Dog
Builder



Types of Relationships

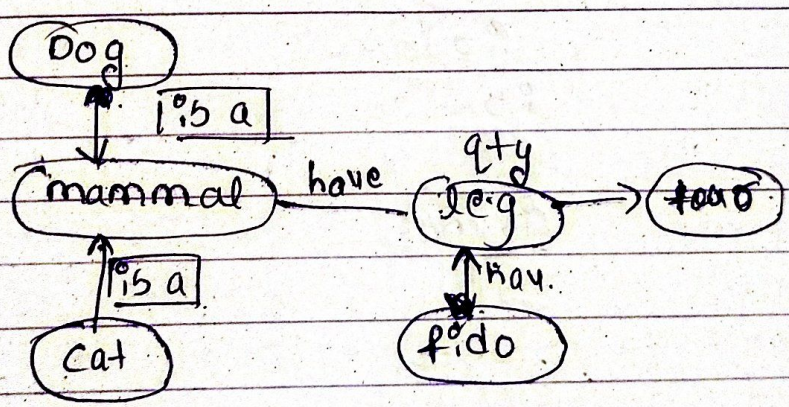
1) Generalization — is a, & subclass ership
2) Aggregation — has, have, or property of node
3) Association — & remaining

Ex:

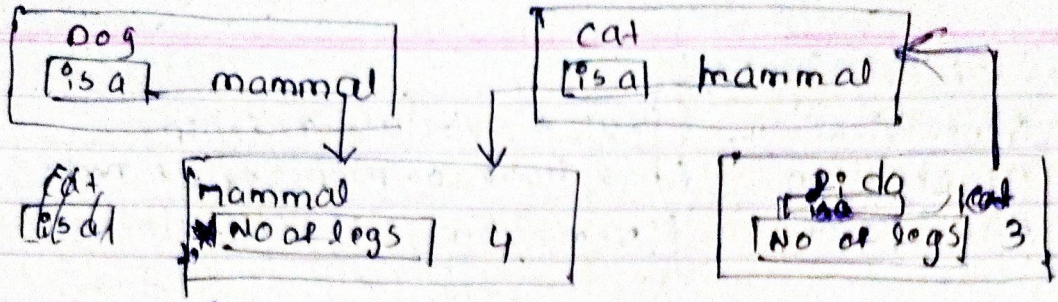
Dogs are mammals
Cats are mammals
mammals have 4 legs
fido has 3 legs

Frame Name	Slot	Slot value
Dog	are	mammals

Semantic Net



Frame Name	Slot	Slot value
Dog	is a	mammal
Cat	is a	mammal
mammal	NO of legs	four
fido	NO of legs	3



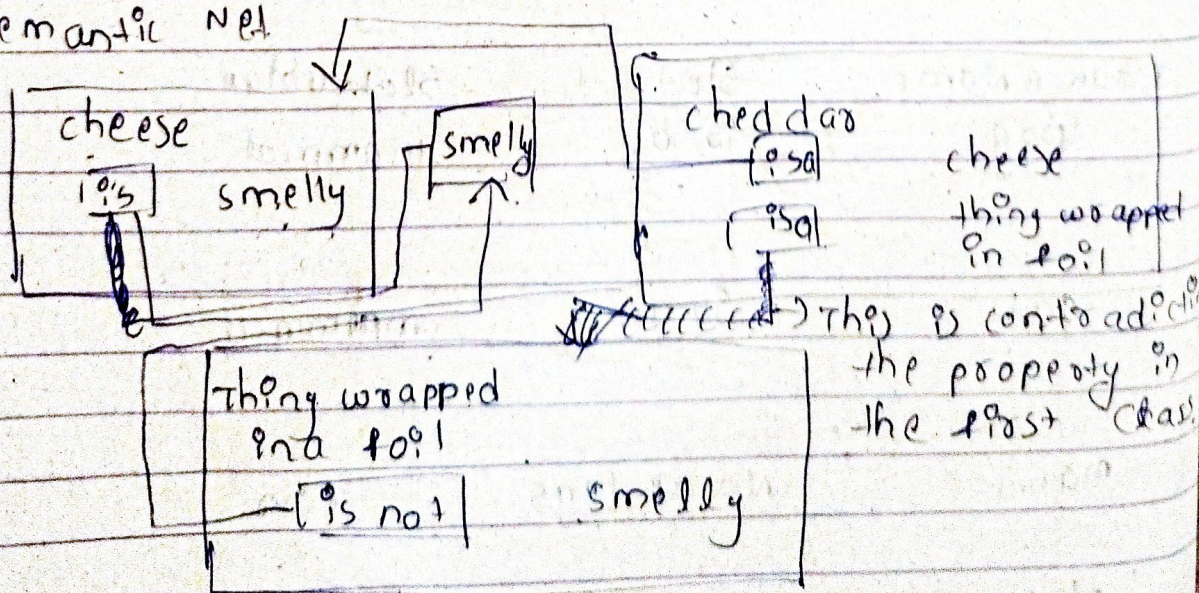
asterisk sign to override

22/03/24

Ex: cheese is smelly
thing wrapped in a foil is not smelly
cheddar is a cheese
cheddar is a thing wrapped in a foil
convert into frames

FrameName	Slot	Slot value
cheese	is	smelly
cheddar	is	cheese
	is	thing wrapped in a foil
Thing wrapped in a foil	is not	smelly

Semantic Net

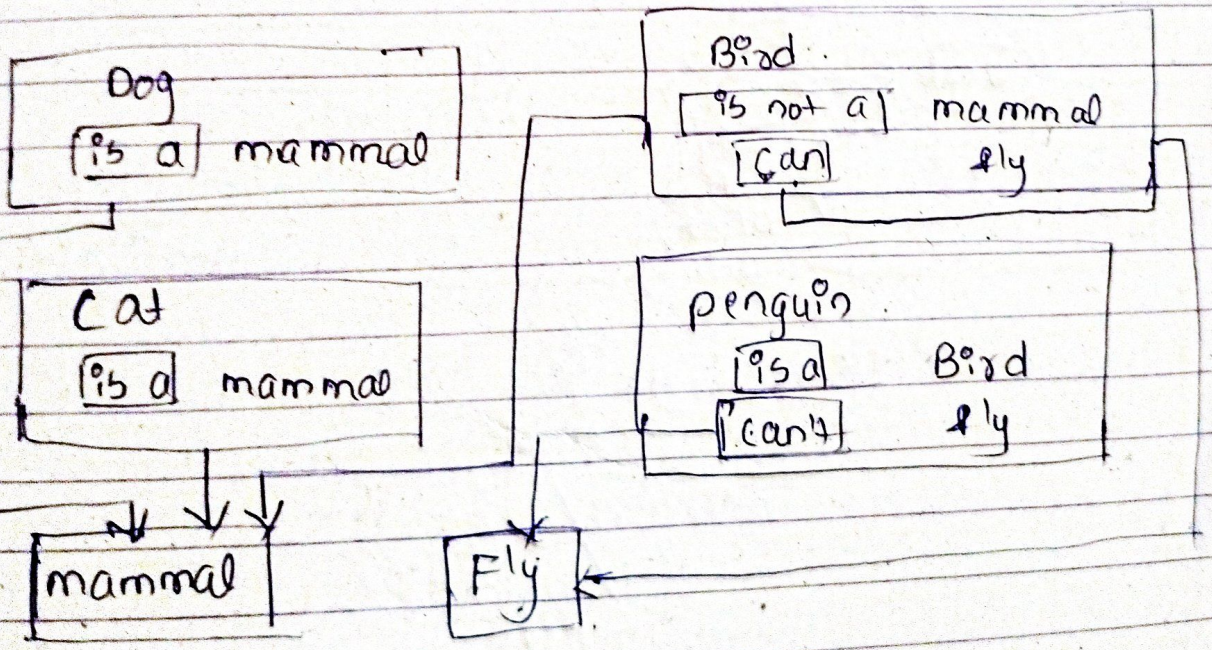


Conclusion: cheddar is smelly

Dogs are mammals
Cats are mammals
birds are not mammals
penguin is a bird
penguin cannot fly } > specific property so this all birds can fly. property cannot be contradicted.
Derive the conclusion about Penguin and also draw the frames.

→

Frames	slot	slot value
Dog	is a	mammals
bird	is not a	mammal
	can	fly
Cat	is a	mammal
penguin	is a	bird
~~penguin~~	Can not	fly
~~Bird~~	~~Can~~	~~fly~~



Conclusion:- penguin can not fly.
penguin is not a bird. } x

Penguin is not a mammal.

Q2)- All humans eat icecream
Humans feed dogs and cats
Dog likes Bones
Humans are mammals
mammals do not like icecream
mammals eat ice cream
conclude the conclusion

Sean Disadvantage of semantic net
be Cannot represent ns
Difficulty in identifying object & classes

Search Tree Space

Search Tree is defined as representation of all possible choices in a given problem where some choices are solutions.

Search Tree

Identify five real time examples of implementation of graphs & trees

State Transitions

23/03/24

State 1
Robot in room A
C in room A

State 2
R in room B
C in room A

State 3
R in room C
C in Room A

State 4
R in room A
C in room B

State 5
R in room B
C in room B

State 6
R in room C
C in room B

State 7
R in room A
C in room C

State 8
R in room B
C in room C

State 9
R in Room C
C in room C

Search state:- All the possible states for a given dgm

Semantic Net:- It is a graph

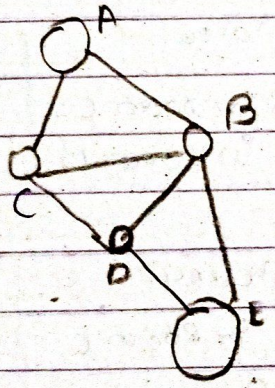
Semantic Tree :- Hierarchical structures

-> It is a kind of semantic net which has special properties as follows

1) Each node has exactly one predecessor and one or more successors
2) one node which does not have predecessor is called the root node. hence it represents starting point for the given property.
3) nodes which do not have any successors are called leaf nodes

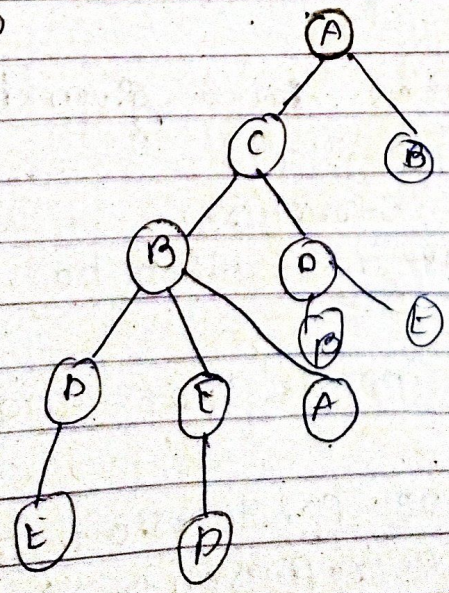
3) One or more leaf nodes are called goal nodes where the search has succeeded
4) An ancestor of a node is a node further up the tree on particular path.
5) descendent is a node which comes after a node on a path.

A path that leads from the root node to a goal node is called a complete path.
A path that leads from the root node to a leaf node that is not the ~~Complete~~ goal node is called the partial node.



Convert it into tree
A is start node & goal node
length of the node cannot be zero.

→



27/03/24

Missionaries and Cannibals problems

3 missionaries & 3 cannibles are one side of river with a canal they all want to get on the other side of the river the canal can only hold one or two people at a time at no time should there be more cannibles than missionaries on either side of the river as this would probably result in the missionaries being eaten

operator

1) move 1 cannibal to the other side
2) move 2 cannibales to the other side
3) move 1 missionary to the other side
5) Move 1 missionary & 1 canibal to the other side
4) Move 2 missionary to the other side

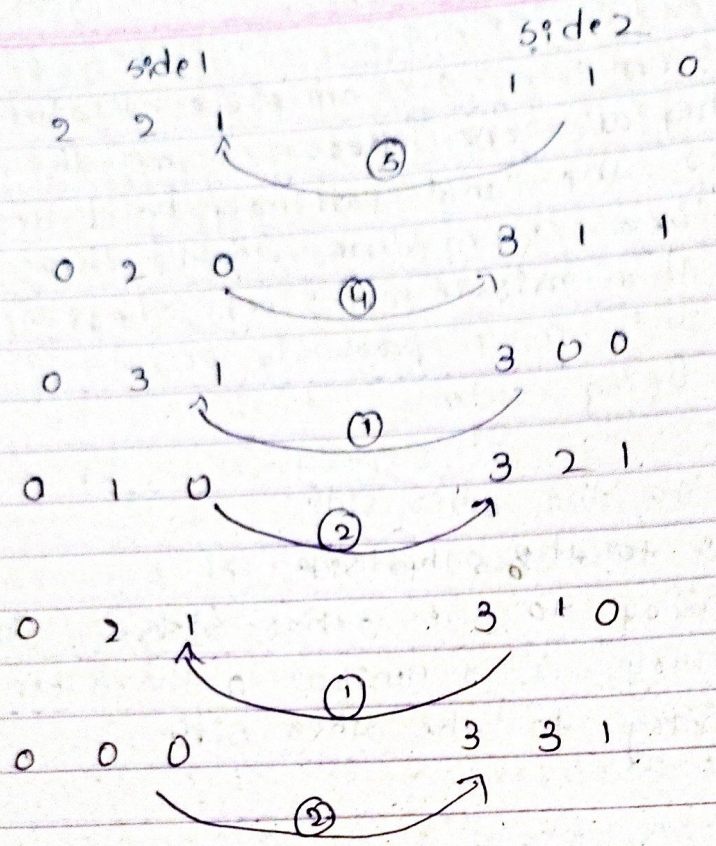
side 1	side 2	
M C C	0 0 0	→ Initial state
3 3 1	3 3 1	→ Goal state

Invalid states

2 3 1

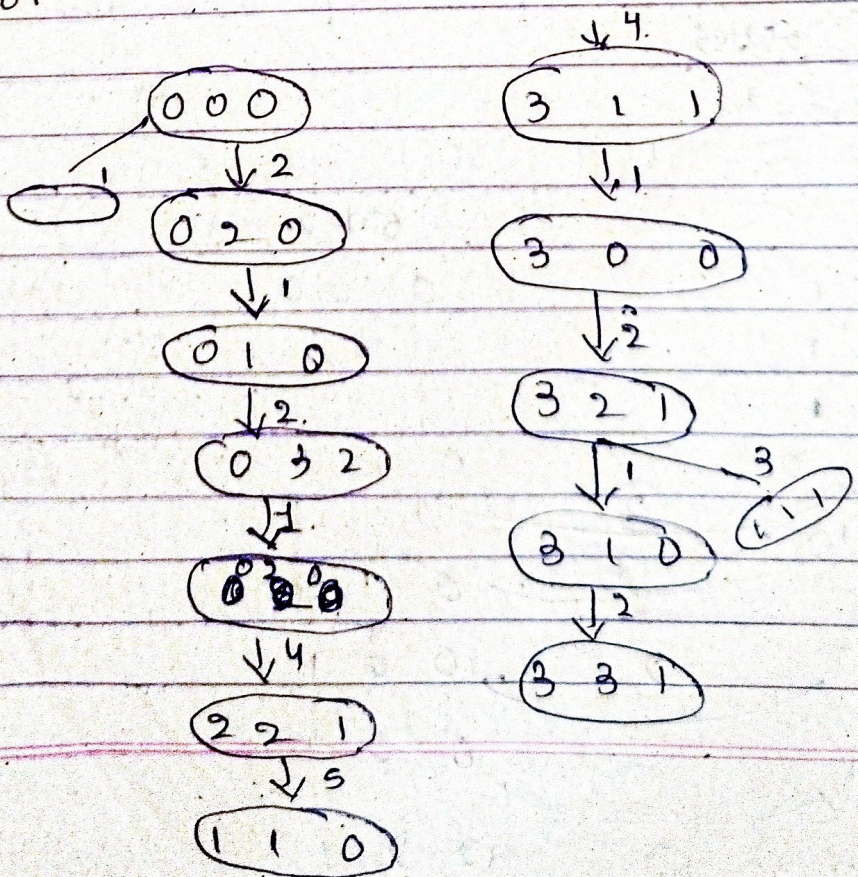
⇒

side 1	side 2	operator used
M C C	0 0 0	1
3 3 1		
3 1 0	0 2 1	②
3 2 1	0 1 0	①
3 0 0	0 3 1	②
3 1 1	0 2 0	①
1 1 0	2 2 1	④



Search Tree (only side 2 states)

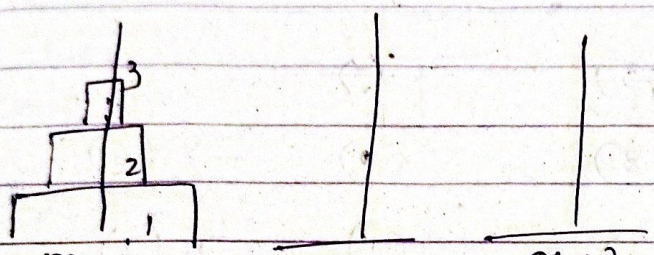
Complete search tree will have all the possible sol^n. partial search tree means it should have atleast one sol^n.



28/03/24

Towers of Hanoi Problem is defined as follows we have 3 pegs and no of disks of different size the aim is to move from the starting state where all the disks are on the first peg in size order (smallest at the top) to the goal state where all the pegs are on the third peg and also in size order, and we are allowed to move one disk at a time as long as there are no disks on the top of and as long as we do not move it on top of a peg that is smaller than it.

→



(1,2,3) () () → Initial state

() () (1,2,3) → Goal state.

Invalid Strings

() (3,2,1) ()

() (3,2) (1)

() (3,1) (2)

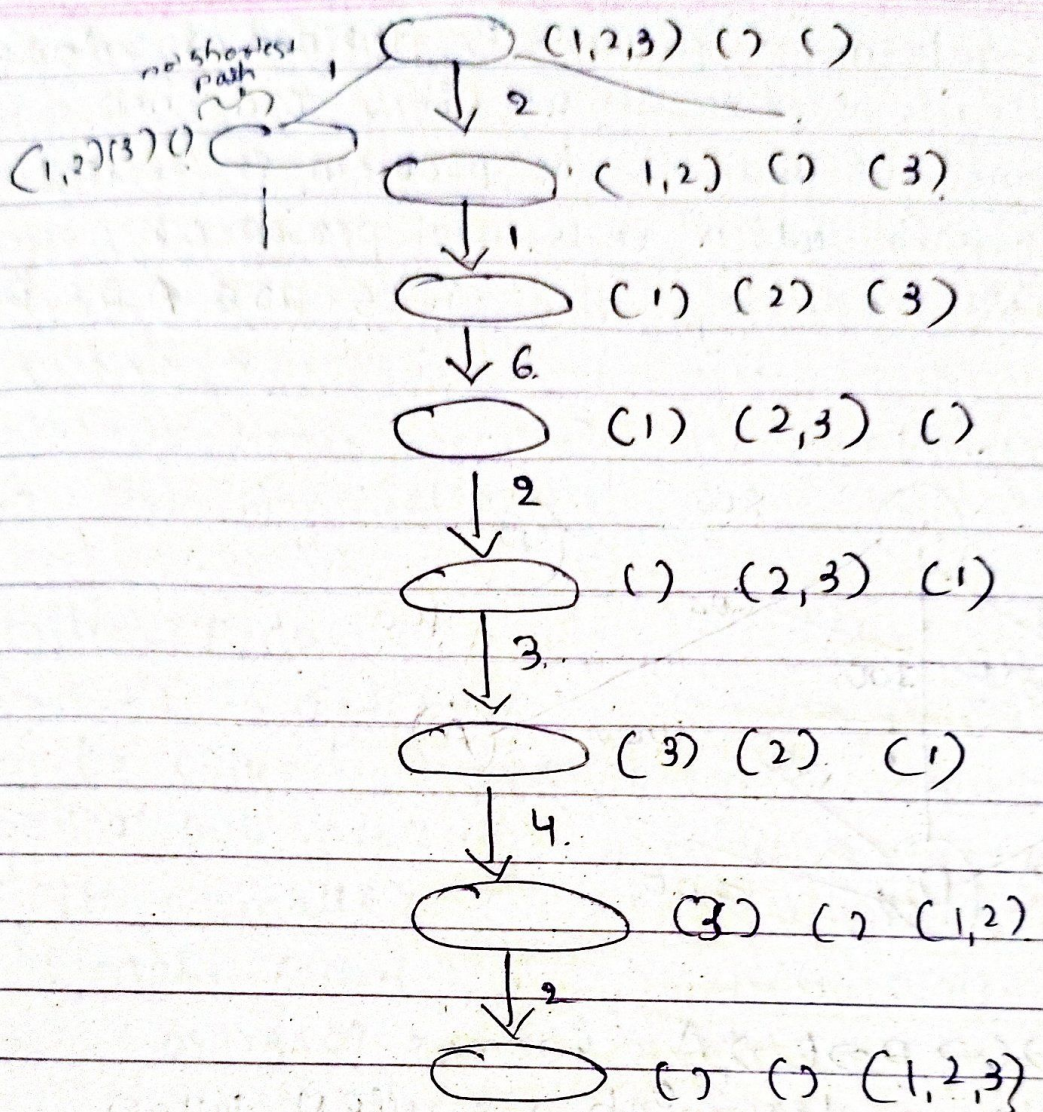
() () (3,2,1)

Operators:-

1) move disk from peg1 to peg2
2) move disk from peg1 to peg3
3) move disk from peg2 to peg1.
4) move disk from peg2 to peg3.
5) move disk from peg3 to peg1.
6) move disk from peg3 to peg2.

peg 1	peg 2	peg 3	operator used.
(1, 2, 3)	( )	( )	
(1, 2)	( )	(3)	→ (2)
(1)	(2)	(3)	→ (1)
~~(1)~~	~~(2, 3)~~	~~( )~~	
(1)	(2, 3)	(•)	→ (6)
( )	(2, 3)	(1)	→ (2)
(3)	(2)	(1)	→ (3)
(3)	( )	(1, 2)	→ (4)
( )	( )	(1, 2, 3)	→ (2)

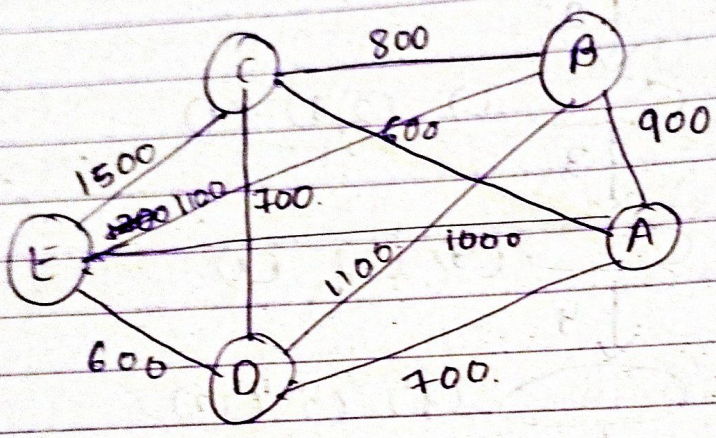
Search Tree



Try with 4 disks.

Travelling salesman problem

Travelling salesman problem is defined as salesman must visit each of set of cities and then return home the aim of the problem is to find the shortest path which lets the person or salesman visit each city. The cities are A, B, C, D, E



A → B → C → D → E → A = 4000

Visit the nodes with smallest values

A → C → D → E → B → 4500

Heuristic soln visit the city which is nearest to other and we can't visit the node twice until we visit all other nodes.

as we increase city the no of soln gets increased exponentially.

Imp

Combinatorial explosion

→ The problems which have the property that has the number of individual items increases the no of possible paths in the search tree increases exponentially which means that as the problms get longer it becoms more and more unreasonable to expect a computer program to be able to solve it. Such problems are known as combinatorial explosion.

Eg: Travelling Salesman.

Problem Reduction

→ It is defined as a complex problem which can be solved by breaking it down into several smaller sub problems. and when we solve all of these smaller sub problems then we have solved the main complex problem. This approach is also known as goal reduction.

Goal tree

AND → Solving all the subproblems A

OR →

15/4/24

Unit - II

Search Methodologies

1) Data driven search → forward chain
2) Goal driven search → backward chain

Goal Driven search is a search which starts at the goal and work backwards towards a start state by seeing what moves or actions have led to the goal state.

Data driven search is defined as a search that starts from an initial state & uses actions that are allowed to move forward until a goal is reached

Generate and Test (Blind search) } no info abt problem's properties

This search should have a suitable generator which has the following properties

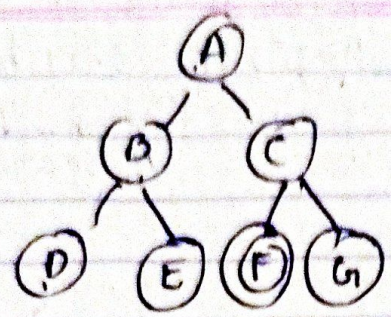
→ It must be complete
→ It must be non redundant } should not generate the
→ It must be well informed

Heuristic search → there will be additional info

Properties of search method

complexity
completeness
optimality
admissibility { acceptable
irrevokable { DFS is revokable
BFS is irrevokable }

diff sm with respect to search properties



A

A → B

A → B → D → E

A → C

ABCDEF

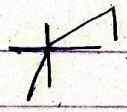
d = 3 b = 2

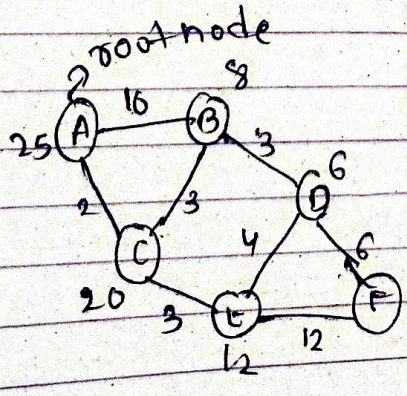
To find
Total no of nodes to be examined.

Hill climbing Algorithm (Heuristic search)
Informed search

```
Function Hill()
{
    queue = [];
    State = root node;
    while(true)
    { if is_goal (state)
        return successful;
      else{
          sout('successors(State);
          add to_front_of_queue (successors(state);
        }
      if queue = []
          then .report FAILURE;
      state= queue[0];
       remove_first_item_from(queue);
    }
}
```

Drawbacks
1) can get stuck in local maxima
2) plateau
3) Ridge





higher Heuristic value
= root node
No value = F = goal node

State	Open Frontier queue	Closed queue
A	[ ]	[ ]
A	B C sorted.	
B	C	A
B	DC ~~C~~ C sorted.	A
D	CC (Add from)	AB (Add back.)
D.	~~CC~~ ~~sorted~~ FECC sorted	AB
F (↳ If F has successors don't add)	ECC	ABD.
	ECC	ABDF = 19 maintain final path

H.W