3 Oct 2024
Thursday

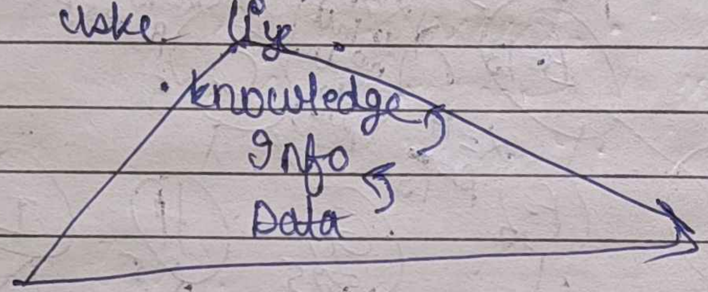
[lec-8] :-

WJMK :)

* Beam Search Algo (H/W) (Not ~~in~~ for Exam)

⇒ Knowledge Engineering :- We'll discuss knowledge representation

→ knowledge can be enhanced, but koi training chahiye uske liye.



→ How to represent knowledge?

* Data can be represented in form of text.

organized, unorganized, Data mining etc...

Textual data, Image data, ~~sign~~ speech data etc.. different-2 types of data.

→ in form of predicates or in form of objects.

WJMK 30 MM.

DATE / /
PAGE NO.

object → entity having certain attributes or methods.

predicate → a property (statement calculus mein hain)
↳ $P(x)$

statement → sentence with truth value.
↳ affirmative sentence hote hain bs, no ?, ! etc.

True or false, not both, not none.

Input to robot: Can you clean a robot? → sentence
Output: Yes or No!

⇒ jo wo environment se perceive krta hai i.e. knowledge.
for eg: if obstacles in room, then they will be recorded
↳ in form of predicates or objects.
↓
propositions hote hain!

p: A is student
q: B " "
r: C " "
↓ common property → predicate.

say ⇒ P : is a student
$P(x)$: x is a student.
$x \in \{A, B, C\}$
i.e. $P(A)$: A is a student

WJMK
ॐ

Date / /
Page No.

Operator	connector			
Conjuction (and) →	$\wedge$			
Disjunction (or) →	$\vee$			
Not →	$\neg$			
NAND ($\uparrow$) →			$\neg\wedge$	($\uparrow$)
NOR ($\downarrow$) →			$\neg\vee$	($\downarrow$)
XOR →			∇	

$\oplus$ — exclusive or

inclusive or → 'or' hi hota.

Inference?

x is statement,
$P(x)$ is property / predicate.

$\rightarrow$ (if then)
$\leftrightarrow$ (iff)

p	q	p → q	p ↔ q
F	F	T	T
F	T	T	F
T	F	F	F
T	T	T	T

$\forall$ → universal quantifier
$\exists$ → existential quantifier

WJMK
ॐ

→ Everything in real world can not be represented.

* Toy world problem v/s Real world problem

Toy world problem → Games (board)

Real world problem → Agents solving a problem

→ move krwaana agent ko → big problem!
↳ like stairs chdna.

eg!
Clean → move, clean

* Ontological diagram:

events
time
physical objects
belief

→ inn saari cheezon ko as diagram represent krenge.

imp **.

Upper ontological:-

(hm yehi follow krenge)

(ek upar cheez hogi, jisse aage expand hongi cheezein).

⇒ Anything can not be represented but somethings can be represented.

Anything
- Abstract_obj
  - Sets
  - Numbers
  - Representational_objects
    - Sentences
    - Measurements
- Generalized_Events

AI

WJMK : 3v MM.

10/10/24 Thursday

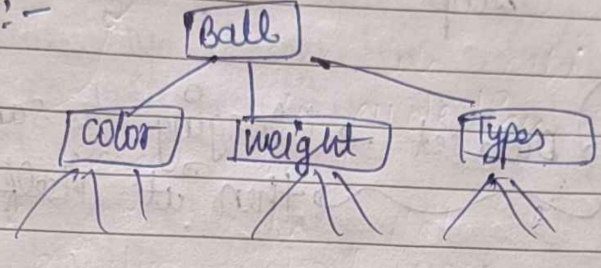
[lec 9]

Beam Search → to decide what will be next move.

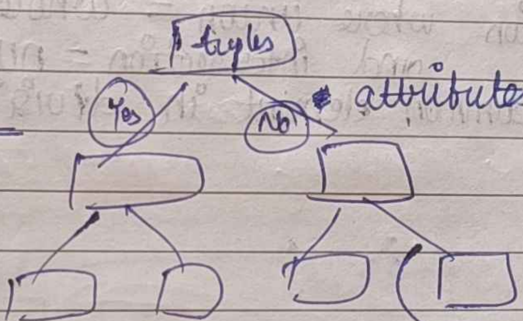
* Decision Tree (DT) :- will be used for decision making.

→ object will be represented by certain subcategories.

eg :-



→ different categories/ ~~cat~~ classification / taxonomies :



If always 2 categories mein tutega, then binary tree.

attributes (object into certain parts on basis of attributes).

these lines are for making decisions.

→ Decision Tree is generated based on SVM.

DT (Algo) :-

C → classes → (say based on color) → red, blue, green etc.

i/p D(Training set), C

o/p T(decision tree)

Step 1 → 1. if all ~~tuples~~ tuples ∈ C_j

Add a leaf labelled C_j → (some class C_j)

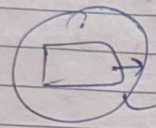
return

(as no further classification)

Base Case

WJMK

2. select A_i (attribute)
3. Partition $D = \{D_1, \ldots, D_n\}$ based on A_i.
4. $\forall D_K \in D$
   create a node and add an edge,
   $D_K \to D$, label : A_i
5. $\forall D_K \in D$
   $DT(D_K) \longrightarrow$ recursion
   stop

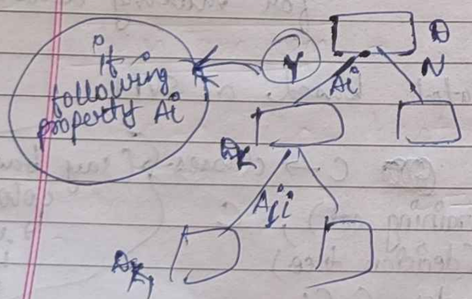


Data set → belonging to same class
→ then it itself is DT.

→ Partition v/s Division
→ division where union = whole set, $(\cup S_i = S)$
and intersection = null. $(S_i \cap S_j = \phi)$
→ no common element in division

Pairwise intersection is null

means ek element 2 classes mein nahi jaa skta.



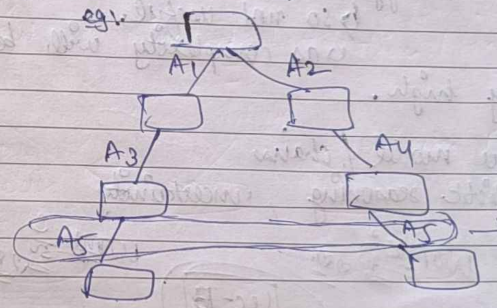
finally yaa toh ek element bchega

WJMK

Yaa fir elements belonging to same class, then recursion khatam ho jaayegi
→ How to partition $D \to \{D_1, \ldots, D_n\}$??
~~Example~~
↳ Simple if, else-if, else statements toh bar naa kr paayoge!!

→ agar 2 different nodes have same attribute anywhere in future, then merge nahi kreinge, as partition kiya huya hai already based on certain property formely, toh alag hi chleinge dono fir!
↳ merge nahi honge.

eg:-



→ same attribute, but not merging!
↳ eiko pehle 'colour' different, but baad mein 'weight' same (or in same range), toh merge nahi kreinge!

Complex Decision Making → or satisfying

Constraints Satisfaction problems
↳ conditions

given $f(x,y) =$ ______

st. $x \geq ?$, $? < x < ?$ --- (some condⁿ)
and $y \geq ?$, $? < y < ?$ --- (" ")

↳ As two variables, so can use 'graphical method'.
↓
make random line and shift it,
jab tak wo conditions satisfy na kare!
↳ but this is also kind of 'blind search' (trigger method).
↳ so not useful.
↳ as complexity will be very high.

→ Markov model / chain
→ Probabilistic reasoning, uncertainty

19/10/2024
Thursday

Lec-10 ,-

(Imp.) (will be asked in Exam.)

→ Decision making using Markov Model.

→ Planning.

① Decision tree → data set classified on basis of attributes.

Agent → 'present state' known to us, want to move to 'next state'.

↳ How to decide to go to 'next state'.

1) is based on Binary ~~entity~~ → Yes/No decisions

we'll extend it to Markov model.

2) based on attribute / present situation.

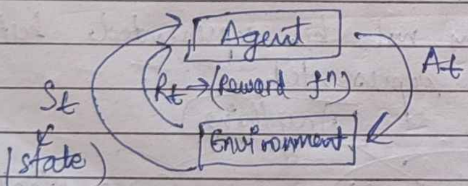
→ Agent does not know 'next state', has no idea ki certain moves k baad wo khaan hoga, ya 'final state' is not known to agent itself.

$S_0 - S_1 - \dots - S_n$

initial state

Goal state → agent has to search it, o/w agent has no idea of goal state.

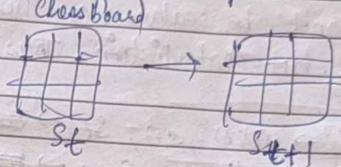
→ Decision is made by binary decisions.



⇒ S_0 se S_1 gye, Reward is associated with this movement to new state ⇒ If Reward '+', then award, if '-ve', then punishment.

Markov Model :-

→ Statistical model, which will decide something as next state, but 'present state' must be defined.



kyi vaar S_{t+1} pta hoga, and usse S_t estimate kreinge

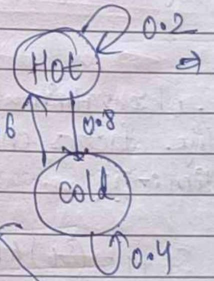
kyi vaar next state pta hogi, like attendance leli toh bche baahar jaayenge, but attendance hogi?

↳ uncertain / undetermined

So S_t → " , S_{t+1} → known.

have to guess, so probability chahiye but kis cheez ki?

eg :-



How got these values?

⇒ representing two states :-
S_t = Hot (say)
S_{t+1} = cold
assumption.

S_{t+2} = ?
↳ have to decide.

we'll have probability matrix / transition matrix → to represent prob. values
↳ such that row sum = 1.

here,

$$\text{transition matrix} = \begin{array}{c} \\ H \\ C \end{array}\begin{array}{c} \begin{array}{cc} H & C \end{array} \\ \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix} \end{array}$$

If 'n' states, matrix size = $n \times n$

WOMK 3 ~~ ~ ~

thus, state is row vectors → having no. of elements = total no of states

→ This transition matrix can be used to decide 'next state'.

$S_{t+1} = \cancel{(P_{ij}) \cdot S_t}\ (S_t)(P_{ij})$

↳ so state is matrix Now!!

obs → S_t can't be scalar as then also S_{t+1} will be matrix, but contradiction. Thus S_t must be matrix.

{ as $P_{ij} \to n \times n$, so, $S_t \to$ ~~n×1~~ $1 \times n$
thus, $S_{t+1} \to$ ~~n×1~~ $1 \times n$

usually this

OR

$S_t \to n \times n$, thus $S_{t+1} \to n \times n$.

~~→ S_{t+1}~~

eg: $S_0 = [1 \quad 0 \quad 0]$ (say)
job, no job, startup

$S_0 = [1 \quad 0 \quad 0 \quad 0]$ (initial state → say)
sunny, rainy, clearday, cloudy

→ and yeh probability niklegi 'observations' se!!
↳ experiment se!
↳ days par

experiment nahi, observation ho skta hai!

→ observation ke liye factors chahiye!

factors for observation :- ??

example continued :- $S_{t+1} = (S_t)(P_{ij})$

$S_{t+1} = \begin{array}{cc} H & C \end{array} [1 \quad 0] \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix}$ low, high

$= [0.2 \quad 0.8]$ H, C

So HOT, HOT hoga → 0.2
HOT, cold bn jaayega → 0.8

$S_{t+2} = [0.2 \quad 0.8] \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix} = [0.52 \quad 0.48]$ high, low

WOMK 3 5/3/25

$S_{t+3} = [0.52 \quad 0.48] \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix} = [0.392 \quad 0.608]$ low, high

so low - high flip ho rahe hain har vaar.

$S_{t+4} = [0.392 \quad 0.608] \begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \end{bmatrix}$

$= [0.4432 \quad 0.5568] \simeq [0.4 \quad 0.6]$ low, high

as, $S_1 = S_0 P \to S_2 = S_1 P = (S_0 P) P = S_0 P^2$

$S_{t+1} = (S_0) P^{(t+1)}$

high high saturation aa gyi, saturated result
↓
ab next state mein almost same prob. aayegi
↓
$S_{t+1} = S_t$

so can't reach to multiple states with P

[next state now similar to previous state
Now can't get guess further.

↳ jahan se start kiya → fog aayi
~~Dune tak~~ bhi 'fog' aayegi!! bcoz of 'P'.

so 'P' should not be reused!

→ so decision making k liye 'factors' pe depend hoyo, don't use 'P' always! → so after certain states / situations, 'P' must be changed!!
↳ part of 'Markov decision process'
↳ probabilistic planning
↓

when we do not have deterministic solution, we have to guess and its called probabilistic planning.

* MDP → Markov decision ~~tree~~ planning (??) process

set of states → $S = [S_0 \quad S_1 \quad \ldots \quad S_t]$

set of possible actions → $A = [A_0 \quad A_1 \quad \ldots \quad]$ } → will be ass. with next state

set of reward values → $R = [R_0, R_1, \ldots]$

description of each action → $T = [T_0, T_1, \ldots] = P_{ij}$ (??)
↓
Transition matrix

$\underline{Q.\ Imp}$ Markov's property → next state will be dependent on present state, not on past state. (directly) (inherently toh krega).
↳ 1st order markov property.

⇒ agar previous 'n' states pe depend krta → then 'nth order markov property'.

→ action will be deterministic or ~~scho~~ stochastic.
↳ hypothetical

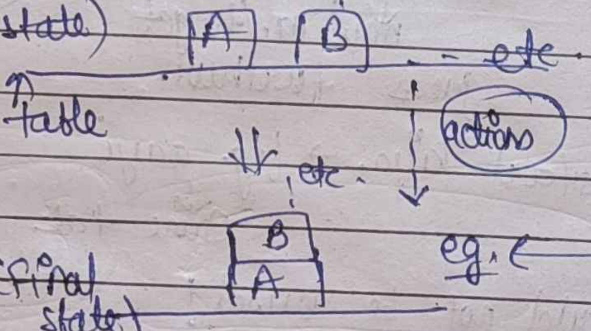
* $S \times A \to S$ (deterministic) ← state
$S \times A \to P(S)$ (stochastic)
↳ action

Policy (π)
↳
$\pi = S \to A$, actions and decisions will also lead to policy.

$\boxed{\text{Probabilistic Planning}}$:-
↳ we will apply predicates (knowledge representation) here and actions.

⇒ Blocks (Toy) world :- blocks → move them s.t. they are placed in certain conditions.

hypothetical robotic arm hogi → jo actions legi

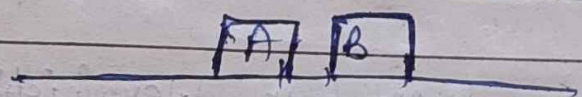


(Initial state) ... etc. Table ... etc. actions eg. (Final state)

→ as 'Agent' ~~does~~ is not aware of 'next state', so uske liye yeh kaam 'very hard'.

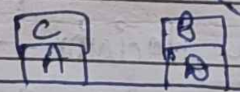
Example :- [ON(A, B)
block A is placed ON block B.

Initial situation :- ONTABLE(A) ^ ONTABLE(B) → situation representation using predicates.



* CLEAR(A) (*) HOLDING(A) (tower / stack of only 2 blocks is allowed)

→ ON(C, A)
→ ON(B, D) →not needed
→ ON(C, A) ^ ON(B, D) ^ ONTABLE(A) ^ ONTABLE(D)
↳ sirf this is enough



goal stack.

Q. exam mein diagram diya jayega → usko predicate form mein likhna hoga.

24 oct 2024
Thursday

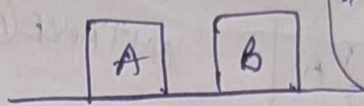
AI
lec-10

WJMK
ॐ

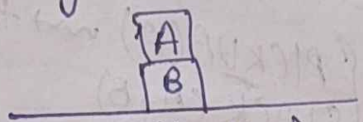
PLANNING GRAPH → 4/10
→ to check whether konsa plan accha hai !!

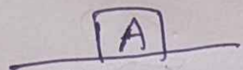
(*) Planning (Classical) :-

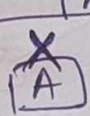
Blocks World :-



(Length of table wgera not given)

(i) ON(A, B) ⇒ 
(Predicate) (situation)

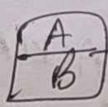
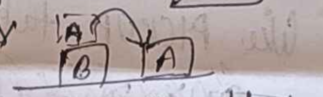
(ii) ONTABLE(A) ⇒ 

(iii) CLEAR (A) ⇒  (means A ke upar no block)

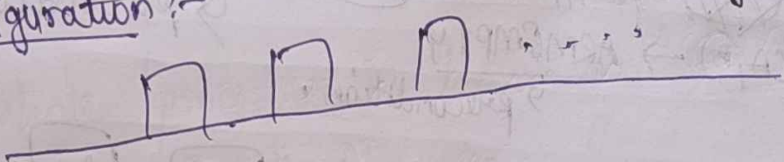
(iv) HOLDING (A) ⇒ Arm holds A

(v) ARMEMPTY ⇒ Arm is empty.

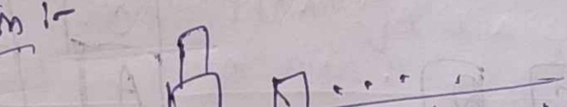
Predicates for representation purpose

Actions :-
STACK (A, B) → stack with A over B

UNSTACK(A, B) → A, B ko alag-2 kro

PICKUP(A)
PUTDOWN (A) → put ~~may be~~ on table ~~or any other block~~.

initial configuration :-



final configuration :-



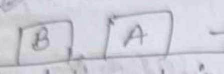
isse yeh bnaana hai actions use krke following predicates.
↳ representing their positions

→ sequence of actions to achieve some goal
↳ 'Plan' of actions.

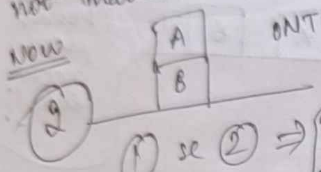
[HOLDING (A) ON(A, B) ARMEMPTY] → etc.
↳ is a plan.

(*) (1)  initial state
↳ ONTABLE(A) ∧ ONTABLE (B) ∧
CLEAR(A) ∧ CLEAR(B) ∧ ARMEMPTY

WOMK

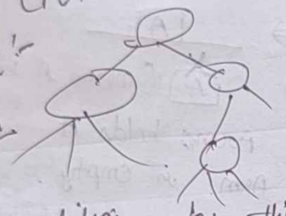
Now 

→ Identical to previous one as difference is in locations only and location not given in this predicate, thus does not matter.

Now 

ONTABLE(B) ∧ ON(A,B) ∧ CLEAR(A) ∧ ARMEMPTY

① se ② ⇒ PICKUP(A) → then HOLDING(A) is true.
STACK(A,B)
PUTDOWN(A)

State Space Tree !- 

Here can also generate this!

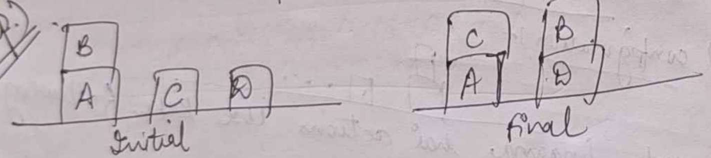
STACK(A,B) → preconditions for this action?

Like PICKUP table/a state agar ARMEMPTY hai!

~~ON(A,B)~~ PUTDOWN(A) → CLEAR(B), HOLDING(A)
action, put A on B. → precondition

STACK(A,B) → ARMEMPTY
→ precondition.

Q. (Imp.)



Initial situation:- ONTABLE(A) ∧ ONTABLE(C) ∧ ONTABLE(D) & ∧ ON(B,A) ∧ CLEAR(B) ∧ CLEAR(C) ∧ CLEAR(D) ∧ ARMEMPTY

Final situation:- ONTABLE(A) ∧ ONTABLE(D) ∧ ON(C,A) ∧ ON(B,D) ∧ CLEAR(C) ∧ CLEAR(B) ∧ ARMEMPTY

Actions !- possibility → ② ~~B pehle table par~~ ... konsa krenge? ~~kreinge D, bhi~~

① B ~~sidha~~, D ~~par~~ wala pehle bnaao   C, A wala pehle bnaao

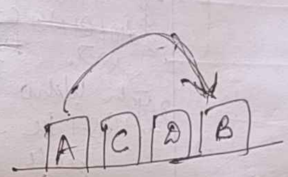
WOMK

robot ko thodi pta hai !!

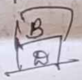
STACK(C,A) → for this requires some preconditions that must be true!

ONTABLE(A), CLEAR(A)
(?) → True   → Not true
for this UNSTACK(B,A)
ARMEMPTY ?
(preconditions)
Yes! dekho

So Action-! → PICKUP(B).
→ ARMEMPTY = false
→ HOLDING(B)

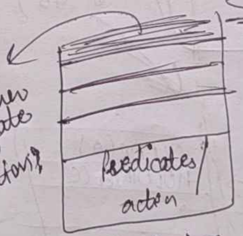


PUTDOWN(B)
ARMEMPTY = true

→ and 

bnaado!

This whole procedure is known as "Goal Stack" Planning.



Here goal stack :-

GS-1

ON(C,A)
ON(B,D)
ON(C,A) ∧ ON(B,D) ∧ CLEAR(C) ∧ ONTABLE(D) ∧ CLEAR(B) ∧ ONTABLE(A) ∧ ARMEMPTY

laomke 3 ♡♡.

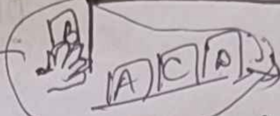
OR 2nd possibility:- Goal stack:-

This whole plan changes agar GS-2 choose kete

GS-2:-

ON(B, D)
ON(C, A)

we'll try to solve GS-1:-



Precondition:- ONTABLE(A) ∧ HOLDING(B)

UNSTACK(B, A)

pop ← HOLDING(B) false ← then PUTDOWN(B)

PUTDOWN(B)

Now Replace HOLDING(C) with PICKUP(C), Now ~~~~ so STACK(C, A), then ON(C, A).

so GS-1 stack

Matlab likhna HOLDING(C) ⇒ PICKUP(C).

true so pop ← CLEAR(B)

ON(B, A)

ON(B, A) ∧ CLEAR(B) ∧ ARMEMPTY

toh yeh sb pop hogya! & Now check ~~HOLDING(C)~~ ~~true~~ or not,

then yeh push

UNSTACK(B, A)

yeh pop hoga

yeh likho GS-1 ke pehle wale mein, ON(C, A) ko replace karke

ON(C, A):- $\boxed{\text{STACK(C, A)}}$.

↳ CLEAR(A) ∧ ~~ON(C, A)~~ HOLDING(C)

Not part of stack

CLEAR(A)

HOLDING(C)

Now try to solve ON(C, A)

ON(B, D)

ON(C, A) ∧ ON(B, D) ∧ ONTABLE(A) ∧ ONTABLE(D) ∧ ARMEMPTY

same

Now this is GS-1

Just to understand this

as

ON(C, A)
ON(B, D)
...

⇒

STACK(C, A)

⇒

CLEAR(A)
HOLDING(C)

Now UNSTACK(B, A)

precondition:- ON(B, A) CLEAR(B) ARMEMPTY

Now, STACK(B, D)

ONTABLE(B) ∧ CLEAR(B) ∧ HOLDING(B)

so on! → when stack is empty, we reached to goal state ☺

WJMK & Th OM.

Page No.:
Date: / /

7 Nov 2024
Thursday

$\boxed{\text{Lec-12}}$ (AI)

⇒ Goal stack (Imp). (will be asked in Exam)

(Heuristic based Planning)

Planning Graph :- poly. size approx. of state space tree

as can have (∞) states, and very difficult to find goal state, so we come to 'planning graph'.

eg :- To have a cake and eat it too. [this is problem].

→ Can have cake (step-1)

→ only then can eat it (step-2)

\# States :- $S_0, S_1, \ldots$

\# actions :- $A_0, A_1, \ldots$

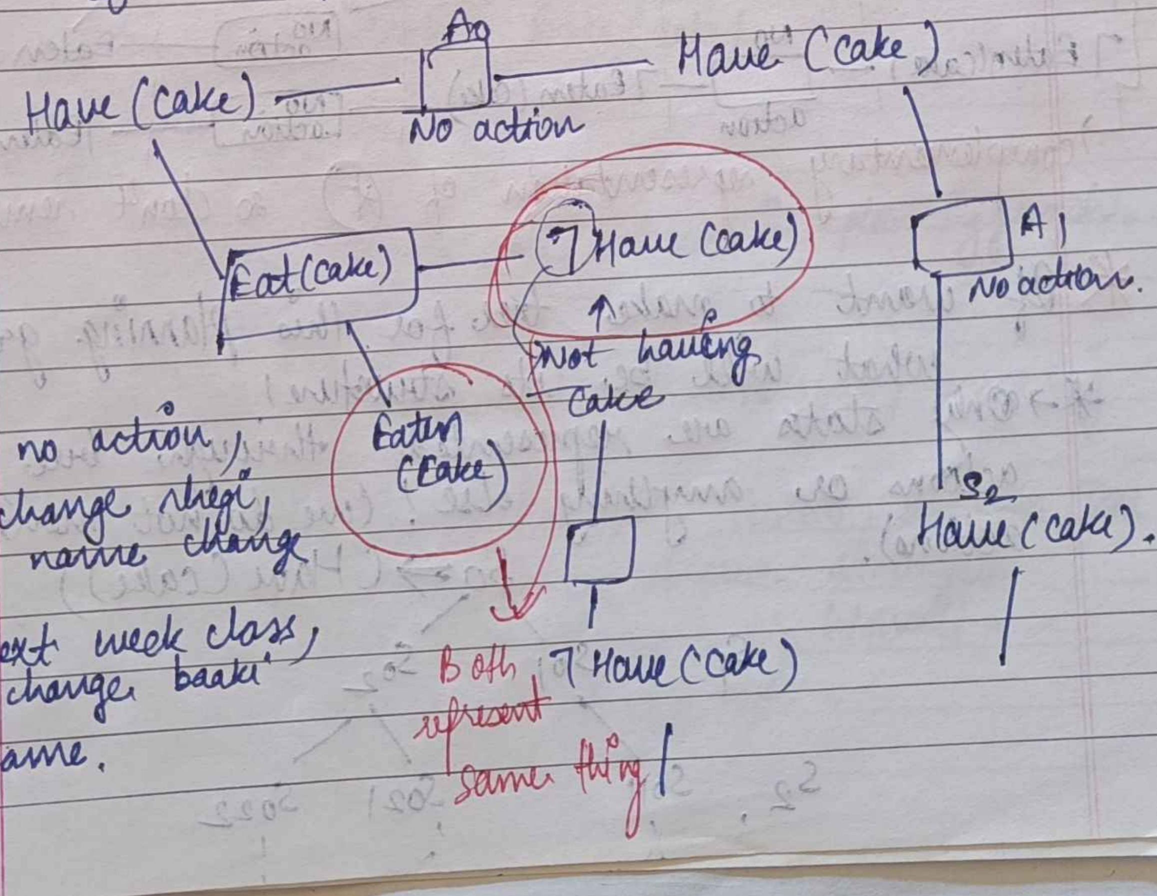
→ Initial state → Have (cake) (S_0)

→ Goal state → Eat (cake) (S_g)

→ in b/w some actions honge!

$S_0 \xrightarrow{A_0} S_1 \xrightarrow{A_1} S_2 \cdots \xrightarrow{A_g} S_g$

(Imp.) eg :- (Ask in class)



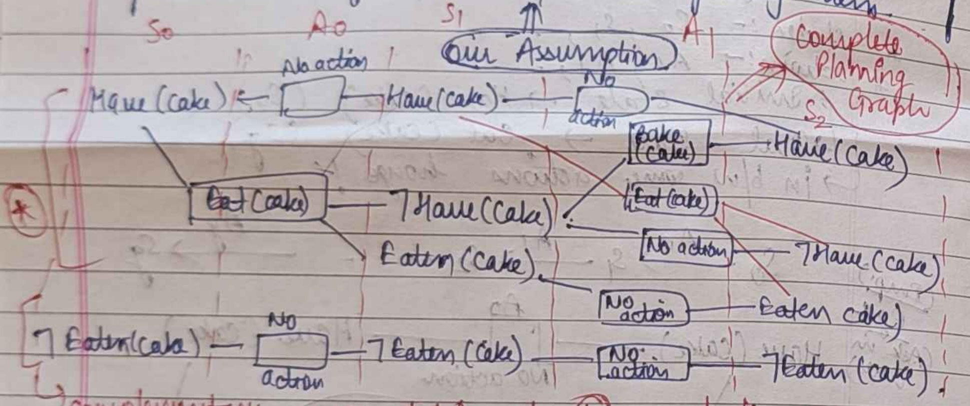
(So if no action, state change nhi, bss state name change hoga)

→ like next week class, date change baaki sab same.

¬ Eaten (cake) —— No action —— ¬ Eaten (cake) ——
not Eaten

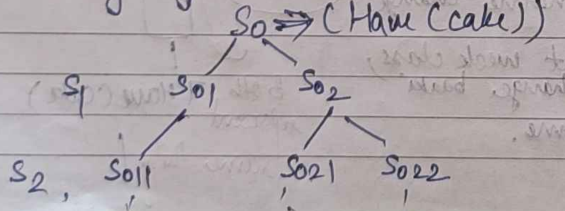
→ Planning graph does not tell whether goal state is reached or not but it can tell no. of steps in which goal is reached (if goal exists).

* If we donot have, then bake it!
↳ but need instruments etc. (preconditions) for that, we are ignoring them. ← Our Assumption



complementary representation of * so don't remove it

* If want to make tree for this planning graph. What will be its structure?
*→ Only states are represented through tree, not actions or anything else! (we do not mention actions).



So in planning graph, only (3) states (S_0, S_1, S_2) are possible.

\# Sequence of actions to reach from initial to goal state is known as "Action Plan".

\# Same problem representation by Predicates :-

initial state (Have (cake)) ← S_0
Goal state (Have (cake) ∧ Eaten (cake)) ← S_g
↳ simultaneously nahi ho
1st u have state, then eat it!!

Predicates

Action :- Eat (cake)
Precondition :- Have (cake)
Effect :- ¬Have (Cake) ∧ Eaten (cake)
↳ (after performing Action)

Action :- ~~¬Have (Cake)~~ Bake (Cake)
Precondition :- ¬Have (cake)
Effect :- Have (cake)
⇓
all these things are drawn in planning graph.

* ~~Uncertainity~~
* H/W → supervised, unsupervised, reinforcement, learning from observation.

⇒ Ensemble Learning :- Random Forest, decision Tree.

* Agent will try to copy or observe strategies followed by other agents, may be himself.
Eg: auto driving car.

WIMK
5O
MM.

Page No.:
Date: / /

→ How such observation based methods can be implemented?
→ Haan! learn krega from observation, but how robot will observe?

[Steps]
→ Attention (agent pay attention to set of steps)
→ retain in memory
→ Reproduction those actions

Agent following actions done by another agent, thus agent learning from observation.

↳ method use :- Bagging, voting.
→ observe what others do and choose best one!