UNIT - 4

Artificial Intelligence Planning

The intelligent way to do things !

INTRODUCTION

* planning is required to convert objectives into action
* It's decision making task performed by the Robot or comp. sys. to achieve a specific goal.
* planning is the task of coming up with a sequence of actions that will achieve the goal.

example : Goal - Get distinction or clear all subjects
planning - How many days left for exam, No. of chapters to study, Notes availability etc.

What is AI planning ?

Planning is a long-standing sub-area of AI. It's the task of finding a procedural course of action for a declaratively described sys. to reach its goals while optimizing overall performance measures.

Automated planners find the transformations to apply in each given state out of the possible transformations for that state.

Why is it Imp : planning App'(s) in Industry

* Automation is an emerging trend that requires efficient automated planning
* Many app'(s) of planning in Industry (e.g. Robots &

autonomous systems), cognitive sys., cyber security, service composition).

Advices of AI planning Technique

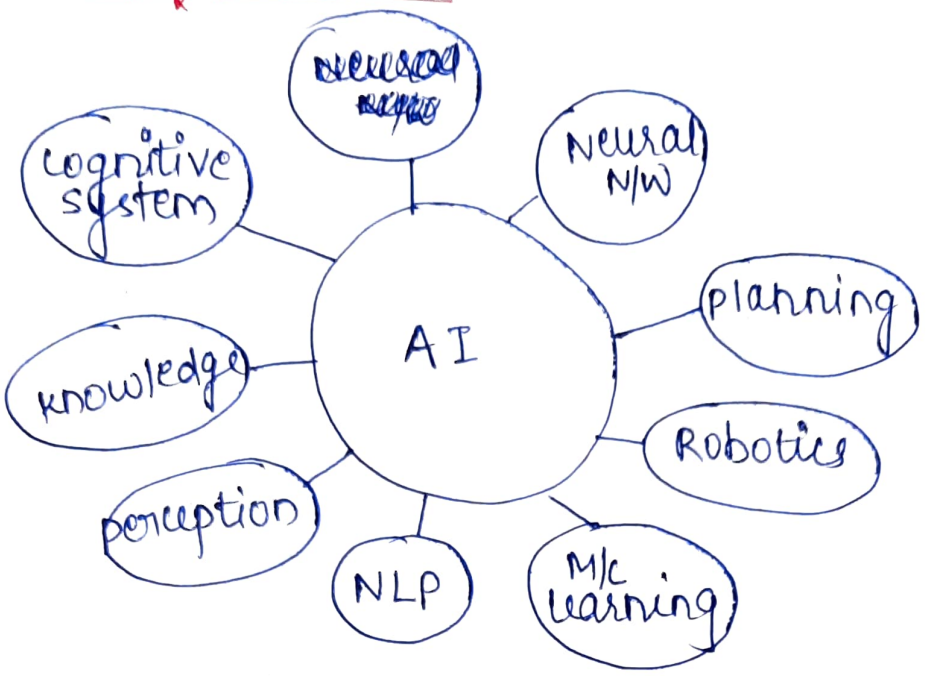
* when explainability is desired
  → when you want to be able to explain why a particular course of action is chosen
  → assignment of responsibility / blame is essential for automation of processes (eg. autonomous driving, medical expert systems)
* Rapid prototyping : short time to solution
* variety of off-the-shelf planners available both IBM proprietary & open source.
* your prob. is frequently changing, even small changes. → no need to change the solution, only tweak the model.

When planning meets Deep Learning (DL)

Solving optimizat° probes. with learning is hard, but integrating planning techniques with heuristic guidance learned by DL will result success.

1) Go player AlphaGO uses planning (Monte-carlo tree search) with DL (Heuristic guidance) to select the next move.
2) Cognitive Assistant Viv (Samsung) uses knowledge graph, planning & DL to answer complicated queries.

PLANNING PROBLEM



Types of planning :

1) Hierarchical planning - a prob. is divided into smaller sub-problems, & each sub prob. is solved one at a time. It's often used in Robotics appⁿ where a Robot needs to figure out how to complete a task by breaking it down into smaller steps.

2) Non-Hierarchical planning - a prob. is not divided into smaller sub problems. Instead, the AI sys. tries to find a single plan that will solve the entire prob. It's often used in Games where a comp. player needs to figure out the best way to win the game.

3) Reactive planning - is a type of planning that is used in situations where the environ. is changing & the AI sys needs to be able to respond quickly.

It's often used in Robotics appⁿ where a Robot needs to be able to avoid obstacle or respond t changes in the environ.

what is planning problem ?

PP in AI are problems that arise when an AI sys. is trying to plan its next move or course of action. These problems can be difficult to solve bcoz the AI sys. must take into a/c all of the possible outcomes of its actions & choose the on that will lead to the best results.

The Planning Problem is the question that how to go to the next move or goal state from the current state. The PP is defined with :

1. Domain Modul - it define the actions along with the objects.
2. Initial state - it's state where any action is yet to take place.
3. Goal state - state which the plan is intended to achieve.

components of Planning System

1) choose the best rule based upon Heuristics (selection of simple subject it's based on past experience or i/p from the seniors or lecture attended)

2) Apply this rule to create a new state (studying of the subject, so as to achieve next stage)

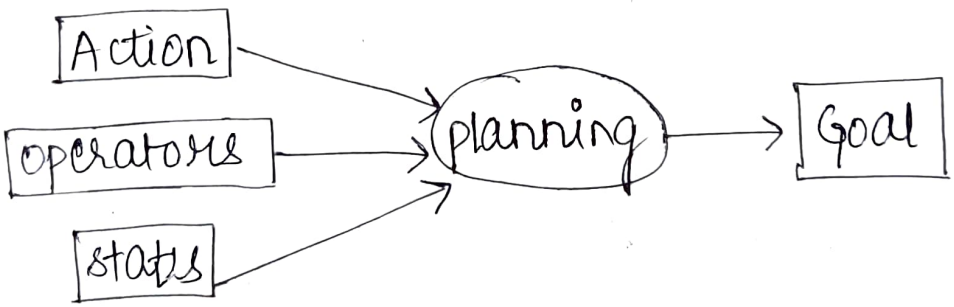
3) Detect when a solution is found (a position is reached where he can score more than cut-off marks)

4) Detect dead ends so that they can be avoided. (find the difficult topic & avoid them)

5) Detect when a nearly solved state occurs & use special methods to make it a solved state. (In position to get more than cut-off marks in all subjects except Maths & then apply strategies to reach the goal)

Planning Representation

It consist of 3 components : state, Action & Goal



1. State → is conjunction of +ve literals which can't contain variables & invoke f'cs). This is represented of facts.
2. Action → these are precond'cs) that must hold before execut'n & the effects after execution
3. Goal → It is the most specificied state; a state is satisfy the given goal if it consists of all objcs reqd. for the goal.

Agent tries to find a solution i.e a sequence of actions that solves a problem.

Planning Agent = problem solving agent + Knowledge Based Agent

(Generate sequences of actions to perform tasks and achieve objectives)

Problem solving agent → to consider the consequences of sequences of actions before acting

Knowledge Based Agent → can select actions based on explicit logical representat° of the current state & the effects of actions.

PLANNING LANGUAGES

PL should be expressive & every planning lang. makes use of the representat° schema so that the algo. can be operated on it.

Types:- 1. standard Research Institute problem solver (STRIPS)

2. Action Discription Lang. (ADL)

3. planning Domain Discription Lang. (PDDL)

(1) STRIPS :

* developed in 1970s at Stanford for the first intelligent robot
* it can describe the world (initial state, goal state) by providing objs, Actions, precond & effects.

* A STRIPS instance is composed of : 1) State 2) Goal 3) Action

State - It's conjunction of +ve literals which can't contain variables & invoke f^cs.

ex. At(Home) ^ Have(Banana)

Goal - these are conjunction of literals may contain variables. ex At(Home) ^ ¬ Have(Banana)

At(X) ^ Sells(X, Banana)

Action - these are precond(s) that must hold before execution & the effects after execution.

ex. Action (fly(P, from, to)

PRECOND : At(P, from) ∧ plane(P) ∧ Airport(from) ∧ Airport(to)

EFFECT : ¬At(P, from) ∧ At(P, to))

(2) ADL (Action Descript^n Lang.) :

* used to overcome the limitations of STRIPS
* more expensive than STRIPS
* propes) : 1. It allows −ve literals
2. It use quantified variables with the disjunction & the conjunction.

ex. ¬ poor ∧ (famous ∨ smart)

3. cond^al post cond^s are allowed
4. variables with diff. types at the same time are allowed & also equality prop. is available.

ex. consider car driving case with STRIPS:

Action (drive (c, from, to))

pre_cond^n : at (c, from) ∧ car(c)

post cond^n : at (c, from) ∧ at (c, to)

In strips, post cond^n means remove the 1st 'at' cond^n & add the 2nd 'at' cond^n.

With ADL, same action is represented as follows:

Action (drive (c, from, to))

pre_cond^n : at (c, from) ∧ car(c) ∧ (from ≠ to)

post_cond^n : at (c, from) ∧ at (c, to)

(3) PDDL : its std encoding Lang. for 'classical' planning task.

* superset of STRIPS & ADL
* components of PDDL planning task :

1) objects → things in the world that interest us
2) predicates → propers. of objc(s) that we are interested in ; can be true or false.
3) Initial state → state of the world that we start in.
4) Goal specification → things that we want to be true
5) Actions / operators → ways of changing the state of the world.

planning tasks specified in PDDL are separated into two files:

1. Domain file for predicates & actions
2. problem file for objects; initial state & goal specification.

Domain file looks like:

```
(define (domain <domain_name>)
<PDDL code for predicate>
<PDDL " " first action>
<PDDL " " last " >
)
```

Problem files look like:

```
( define (Problem <Problem_name>)
 (: domain <domain_name>
 <PDDL code for objects >
 <PDDL " " initial state>
 <PDDL " " Goal>
```

BLOCKS WORLD

* also known as Sussman Anomaly
* In this prob., 3 blocks labeled as 'A', 'B', 'C' are allowed to rest on the flat surface. The given cond^n is that only one block can be moved at a time to achieve the goal.

It's consist of following :

1) table 2) Identical blocks with unique letters on them
3) Blocks are put one to another in stack form
4) stack is built with a robot arm. The arm can perform op's of lifting a single block at a time & placing it.

example predicate used to describe states in block world are :

1. On (A, table) : Block A is on the table
2. On (B, table) : " B " "
3. On (B, A) : Block B is on Block A
4. Clear (A) : Block A has nothing on it.
5. Clear (B) : " B " "
6. Holding (B) : Robot arm is holding B
7. Empty-Arm : arm is not holding anything

State Representation -

Initial state : On (A, table) ∧ On (B, table) ∧ clear (A) ∧ clear (B) ∧ Empty-Arm

Goal state : On (A, table) ∧ On (B, A) ∧ Clear (B) ∧ Empty-Arm

4 action (op) used to describe states in Block world are :

1. UNSTACK (B, A) : to lift block B from A
2. Stack (B, A) : To place block B on A
3. Lift (B) : to lift the block B from the table

4. Place(B): to put the block B on the table.

The Blocks world is chosen because:

* It's sufficiently simple & well-behaved
* easily understood
* provides a good sample environ. to study planning: problems can be broken into nearly distinct subproblems.

GOAL STACK PLANNING

* one of the earliest methods in AI in which we work backwards from the goal state to the initial state.
* this approach uses a Stack for plan generation. The stack can contain sub-goal & actions described using predicates.
* the sub-goals can be solved one by one in any order.
* we start at the Goal state & we try fulfiling the pre-cond'(s) reqd. to achieve the initial state.
* we keep solving these 'goals' & 'sub-goals' until finally arrive at the initial state.

The Robot Arm can perform 4 op'(s):

* STACK (X, Y): stacking Block X on Block Y
* UNSTACK (X, Y): picking up Block X which is on top of Block Y

* PICKUP(X): picking up Block X which is on top of the table.

* PUTDOWN(X): put Block X on the table

Operation	Pre-condition	Action
1. PICK UP(X)	ARM-EMPTY ^ ON(X, table) ^ Clear(X)	Holding(X)
2. PUTDOWN(X)	Holding(X)	Arm-Empty ^ On(X, table) ^ Clear(X)
3. STACK(X,Y)	Holding(X) ^ Clear(Y)	On(X,Y) ^ Clear(X) ^ Arm_Empty
4. UNSTACK(X,Y)	On(X,Y) ^ Clear(X) ^ Arm-Empty	Holding(X) ^ Clear(Y)

Ex. Initial state:

Goal state:

Solution

Step 1: our goal is ON(A, B)

explanation → goal is achieved by opr no. 3 so,

Step 2: *Stack(A, B)

Step 3: Holding(A) ^ Clear(B) - precondn

Explanation → In step 3, Holding(A) is not true, when we see initial state. It's expand by opr no. 1.

Step 4 : *PICKUP(A)

Step 5 : Arm Empty ^ ON(A, table) ^ Clear(A) - precond^ TRUE

Note: Hence, here step 5 is True, so, it's also prove the step 3.

So, when step 3 & 5 true then step 2 *Stack(A,B) TRUE & we can achieve goal ON(A,B)

precond's are underlined with Red & actions are marked as star!

MEAN END ANALYSIS

it solves probss. by defining the goal & establishing the right action plan. This technique is used in AI progss. to limit search.

It combines forward & Backward strategies to solve complex problems.

In this technique, sys. evaluates the diff. b/w current state / position and target / goal state. It then decides the best action to be undertaken to reach the end goal.

How MEA works

1) First, sys. evaluates the current state to establish whether there is a prob.. If prob. is identified then it means that an action should be taken to correct it.

2) Second step involves defining the target or desired goal that needs to be achieved.

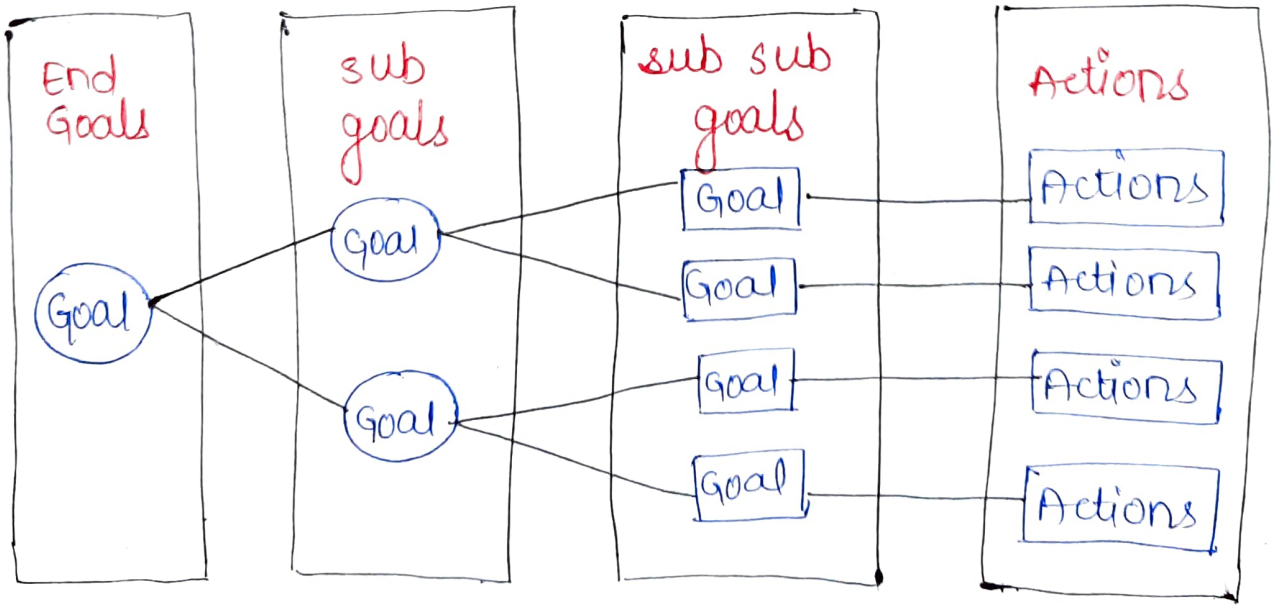
3) Target goal is split into sub-goals, that are further split into other smaller goals.

4) it involves establishing the actions / ops that will be carried out to achieve the end state.

5) In this, all the sub-goals are linked with corresponding executable actions (ops)

6) After that is done, intermediate steps are undertaken to solve the probes. in the current state The chosen operators will be applied to reduce the diff. b/w the current state & end state.

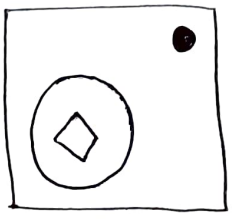
7) it involves tracking all the changes made to the actual state. changes are made until the target state is achieved.



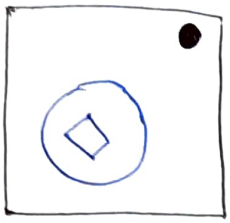
Algo. steps for MEA:

1. Conduct a study to assess the status of the current state. This can be done at a macro or micro level.
2. Capture the problem in the current state & define the target state. This can also be done at a macro or micro level.
3. Make a comparison b/w the current state & end state that you defined. If these states are the same, then perform no further action. This is an indication that the prob. has been tackled. If the two states are not the same, then move to step 4.
4. Record the differences b/w two states at the two aforementioned levels (macro & micro)
5. Transform these differences into adjustments to the current state.
6. Determine the right action for implementing the adjustments in step 5.
7. Execute the changes & compare the results with the target goal.
8. If there are still some diff. b/w current state & target state, perform course correction until the end goal is achieved.

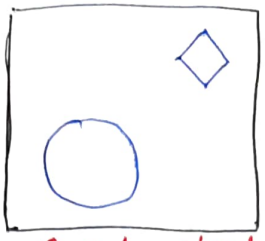
ex. Initial state:



Apply the concept of MEA to establish whether there are any adjustments needed. The first step is to evaluate the initial state & compare it with the end goal to establish whether there are any diff. b/w the two states.



Initial State



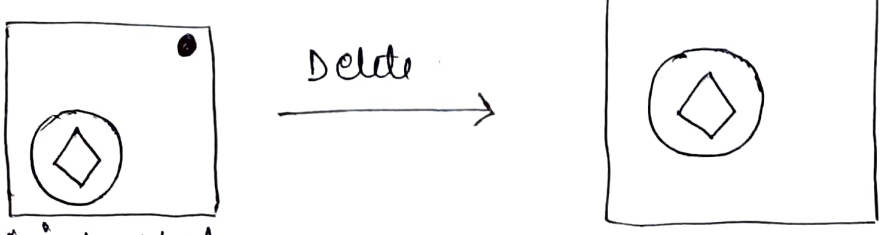
Goal state

Above images shows that there is a diff. b/w current & the target state, this indicates that there is a need to make adjustments to the current state to reach the end goal.

The goal can be divided into sub-goals that are linked with executable actions or op^rcs).

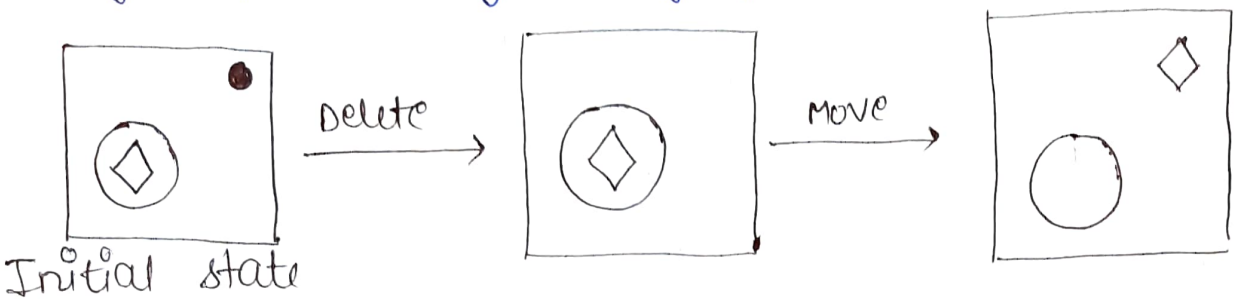
The follo. are the 3 operators that can be used to solve the problem.

1. Delete operator - dot symbol at the top right corner in the initial state doesn't exist in goal state. The dot symbol can be removed by applying the delete operator.

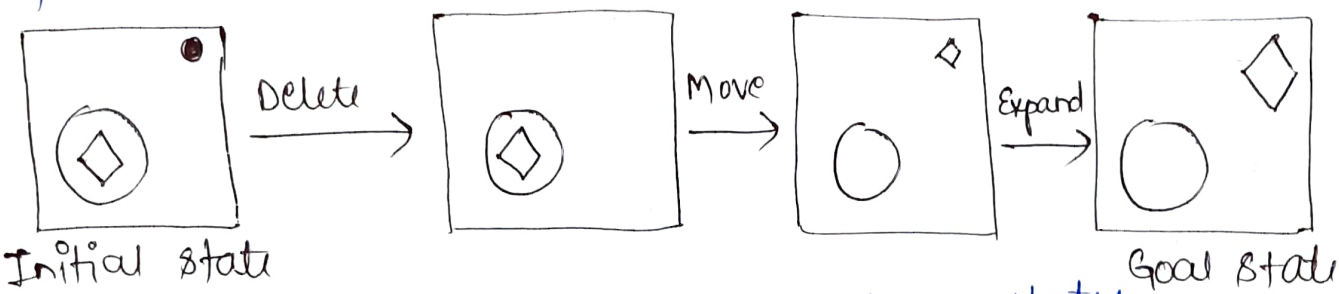


Initial state

2. Move operator - Now, compare the new state with 9 the end state. The diamond in the new state is inside the circle while in the end state, it's at the top right corner. move this diamond symbol to the right position by applying the move operator.



3. Expand operator - after evaluating the new state generated in step 2, find that the diamond symbol is smaller than the one in the end state, we can ↑ the size of the symbol by applying expand operator.



There are no diff. b/w these two states; which means that prob. has been solved.

Applications of MEA

* organizational planning → MEA used in org'ns to facilitate general mgmt. It helps org'al managers to conduct planning to achieve the objectives of the org'. The mgmt reaches the desired goal by dividing

the main goals into sub-goals that are linked with actionable task.

* Business Transformation → this technique is used to implement transformat^{n} projects. If there are any dsired changes in the current state of a busi. project, MEA is applied to establish the new prousses to be implemented. The processes are split into sub-processes to enhance effective implementation.

* Gap Analysis → is the comparison b/w the current performance & the reqd. performance. MEA is applied in this field to compare the existing technology & the dsired tech. in org'cs. various op^{r}cs are applied to fill the existing gap in technology.

MACHINE LEARNING

is a branch of AI which enables m/cs to learn from past data or experiences without being explicitly programmed.

ML enables a comp. sys. to make predictions or take some decisions using historical data without being explicitly programmed.

ML process: ML sys. learns from historical data, builds the prediction moduls & whenever it receives