Unit-4 :-

Imp questions :-

a) Difference & similarities blw problem solving & planning
b) Explain Heuristics for state space search
c) Explain Forward (progression) & backward (regression).
d) PDDL, with example

a) Problem solving & planning :-

→ Problem solving is the process of finding a solution to a specific problem. It may be well defined or open ended. main goal is to find a solution that satisfies the problem's constraints & objectives.

→ problem planning is the process of creating a sequence of actions that will achieve a goal, It may be simple/complex. main goal is to create a plan that is feasible & efficient.

Feature	problem solving	planning
Focus	Find a soln to specific problem	Creating a sequence of actions
Approach	Ad hoc [neccessary or needed]	systematic
Scope	Narrow	Broad
Example	solving a math problem, Debugging a program	planning a trip, schedule a meet

Similarities :-

→ Both involve finding a way to achieve a desired goal.

→ Both can be used to solve a wide variety of problems

→ Both can be implemented in various techniques i.e searching algorithm.

3) STRIPS [Standford research institute problem solver]

a) It is an automated planning technique that works by executing a domain & problem to find a goal.
b) It was used by Shakey ; one of the 1st robots build by using AI.
c) You first need to describe the world, then it consist of start state, goal & action)
d) Action then consist of pre condition & effect.
e) Language used to write the logic & strip is PDDL
f) STRIP Now can search all possible states from start to end goal state.

4) PDDL [Planning domain definition language]

→ It is used to implement the logic & to represent all actions into action schema.

It describes 4 things :
a) Intial state : Representation of the state where agent starts
b) Actions : defined by set of action schemas, it's basically what agent does & consist of pre condition & effect.

a) Precondition: Defines the state in which action can be excuted.

b) effect: It defines the results of exeting the action.

c) Goal:- It is final part / destination to reach.

ex:- FLAT Tire problem!

Init (At(Flat, Axle) ∧ At(Spare, Trunk))

Goal (At(spare, Axle))

Action (Remove (spare, Trunk)

PRECOND: At(spare, Trunk)

EFFECT: ¬At(Spare, Trunk) ∧ At (spare, Ground))

Action (Remove (Flat, Axle),

PRECOND: At(Flat, Axle)

EFFECT: ¬At(Flat, Axle) ∧ At(Flat, Ground))

Action (put on (spare, Axle),

PRECOND: At (spare, Ground) ∧ ~ At(flat, Axle)

EFFECT: ¬At (spare, Ground ∧ At (spare, Axle))

Action (leave overnight)

PRECOND:

EFFECT: ¬At (spare, Ground) ∧ ¬At (spare, Axle) ∧

∧ ¬At (spare, TRUNK)

¬At (Flat, Ground) ∧ ¬At (Flat, Axle))

Q) Ans:- Air cargo problem:-

Init (At (C1, SFO) ∧ At (C2, JFK) ∧ At (P1, SFO) ∧ At (P2, JFK)
∧ (cargo(C1) ∧ (cargo((2) ∧ plane(P1) ∧ plane(P2)
∧ Airport(JFK) ∧ Airport(SFO))
Goal (At(C1, JFK) ∧ At(C2, SFO))

Action (Load(C, P, a)
PRECOND: At(C, a) ∧ At(P, a) ∧ (cargo(C) ∧ plane(P)
∧ Airport(a)
EFFECT ¬At(C, a) ∧ In(C, P))

Action (Unload(C, P, a),
PRECOND: In(C, P) ∧ At(P, a) ∧ (cargo(C) ∧ plane(P)
∧ Airport(a)
EFFECT: At(C, a) ∧ ¬In(C, P))

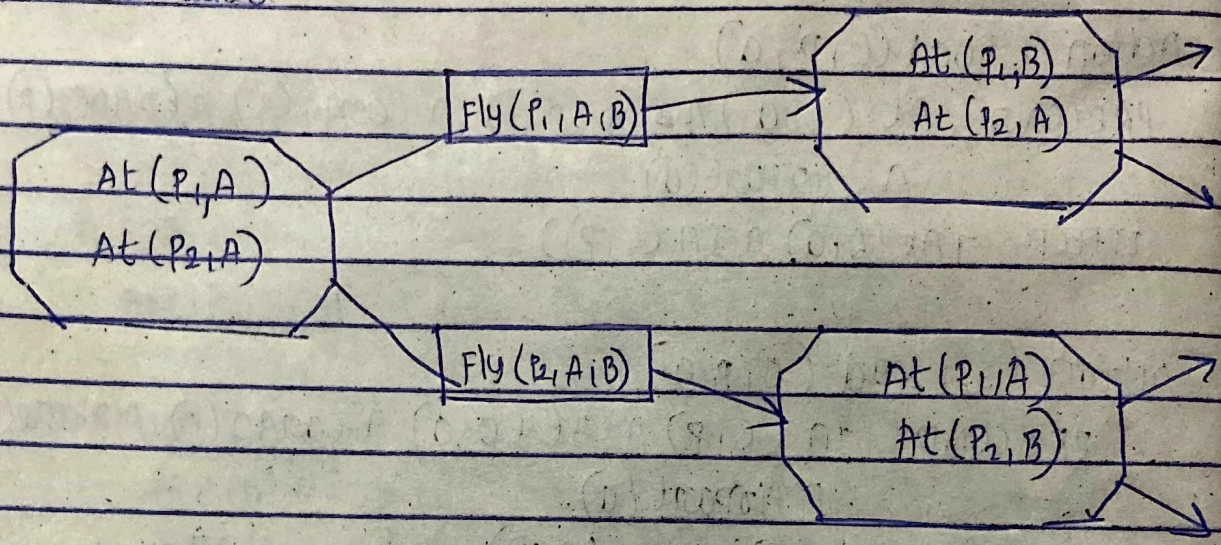
Action (Fly(P, from, to)
PRECOND: At(P, from) ∧ plane(P) ∧ Airport(from) ∧
Airport(to)
Effect: ¬At(P, from) ∧ At(P, to))

Q)

a) Forward state space search (progression)

→ It is also known as data driven

→ It starts from as intial state & uses actions that allows us to move forward until goal is reached.



→ Intial state is form planning

→ Actions here can be load, unload & fly

→ Goal test checks whether state satiesfies the goal of planning problem.

→ cost of each action is 1.

Cargo

eg:- The goal is to move all ~~layer~~ ~~at accept~~ at airport A to airport B.

simple soln: load all cargo into one of planets at A fly & unload the cargo at B. But finding soln is difficult because avg branching factor is huge.

b) Backward state space ~~tree~~ search (Regression)

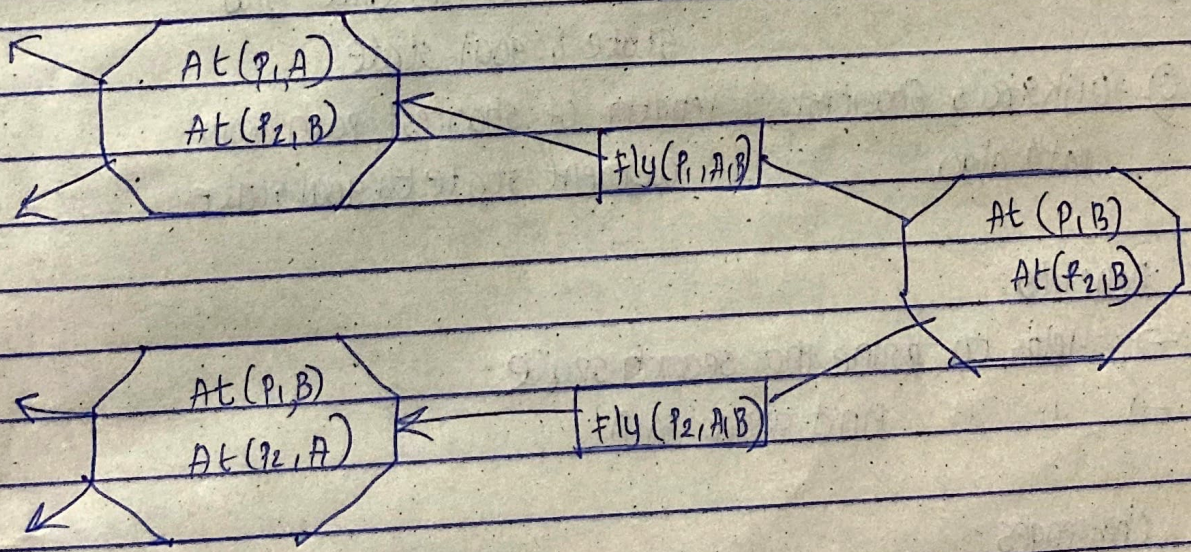
→ It is also known as goal driven

→ In this search, ~~we~~ we start at the goal & work backwards to the start state by seeing what moves could lead us to goal state.

→ this search is useful when the goal is clearly specified

→ It sometimes becomes becomes difficult to propogate manually as it is not possible to find exact predeccasor state | previous state.

→ so we use STIRPS.



eg:- If the goal is to have 20 pieces of cargo at airport B, we would start by regressing the goal state through the operators that achieve the conjunct.

9) Heuristics for state space search

→ Intro:-

Heuristics are used in AI to guide the search for a solution to a problem. Heuristics are rules that are used to estimate the cost of reaching a goal state.

Heuristic Function:-

→ It is a function that estimates how close a state is to goal state.

Types of heuristics:-

a) Manhattan Distance: number of steps to move from state to goal state

b) Euclidean distance: straight line distance blw state & goal state

c) Dijkstra's Shortest path algo: length of shortest path from current state to goal state.

Benefits:-

→ Helps to prune the search space

→ " " Find good soln

Challenges:-

→ Functions can be inaccurate

→ " " difficult to design

→ " may not be consistent in long run.

Conclusions:-

→ Powerful tool that can be used to improve efficiency However we need to be aware of the limitations.