Unit 4

- The planning problem (Air Cargo, The Spare tire)
- PDDL
- STRIPS
- PDDL desc for air cargo ltyre
- Planning as state space search
- forward & backward state space search.
- Heuristics for state space search

Planning in AI

- Planning in AI is about the decision making tasks performed by robots or computer programs to achieve the specific goal.

  Execution of planning is about choosing a sequence of actions with high likelihood to complete specific task.

  Planning is sequence of actions

Eg: Take the book and keep on table is task.

Actions performed Prelude.

Go to book. (move)
Grab the book (grab)
Move near table (move)
Put down on table (putdown)

move (pos)
grab (book)
move (pos, book)
putdown (book)

Actions: Set of action schemas that implicitly define the actions and result (s,a) functions needed to do a problem solving search.

Action schema means precond effect

STRIPS

- Stands for Stanford research Institute problem solver
- It is an automated planning technique that works by executing a domain and problem to find a goal.
- It was planner used by Shakey, one of 1st robots build using AI.
- You first need to describe the world, then it consist of start state, goal, and action
- Action then consist of precondition and effect.
- Language used to write the logic and strip contents is the planning domain definition language
- STRIPS, after describing world can search all possible states from start to goal state executing various actions

PDDL

- Stands for ~~program~~ planning domain definition language
- It is used to implement the logic and to represent all actions into one action schema.
- PDDL describe 4 things.
  - Initial state: It is the representation of ~~each~~ state that the agent starts in.
  - Actions: It is defined by set of action schemas it is basically what an agent does in the set. It consist of precondition and effect.
    - precondition: Precondition defines the state in which the action can be executed.
    - effect: It defines the result of executing the action.

Goal: It is just like precondition: a conjunction of literals (value is either positive or negative)

Air cargo

Init (At (C1, SFO) ∧ At (C2, JFK) ∧ At (P1, SFO) ∧ At (P2, JFK)
∧ Cargo (C1) ∧ Cargo (C2) ∧ 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, fr ∧~~
PRECOND : At (P, from) ∧ Plane (P) ∧ Airport (from) ∧ Airport (to)

EFFECT : At (P, from) ∧ At (P, to))

Actions are load, unload, fly.

(c, p) means cargo c is inside plane p

(c, a) means cargo c is in airport

(p, a) means plane p is in airport

Spare tire problem

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

Goal (At (Spare, Axle))

Action (Remove (obj, loc),

PRECOND: At (obj, loc)

EFFECT: ¬ At (obj, loc) ∧ At (obj, Ground))

Action (Put on (t, Axle),

PRECOND: Tire (t) ∧ At (t, ground) ∧ ¬ At (Flat, Axle)

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

Action (Leave Overnight;

PRECOND:

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

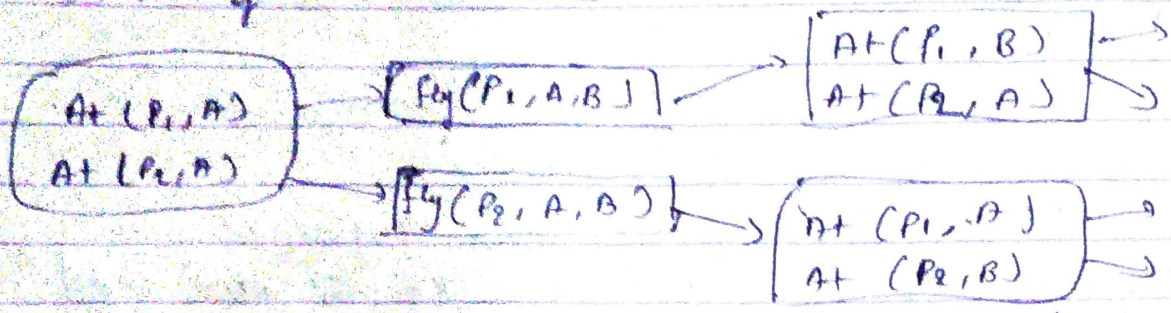
Search Algorithm: The objective of search procedure is to discover a path through a problem space from an initial configuration to a goal state.

2 ways

- Forward state search
- Backward state search

Forward State Space Search (Progression).

- It is also known as data driven or data driven.
- It starts from an initial state and uses actions that allows us to move forward until goal is reached

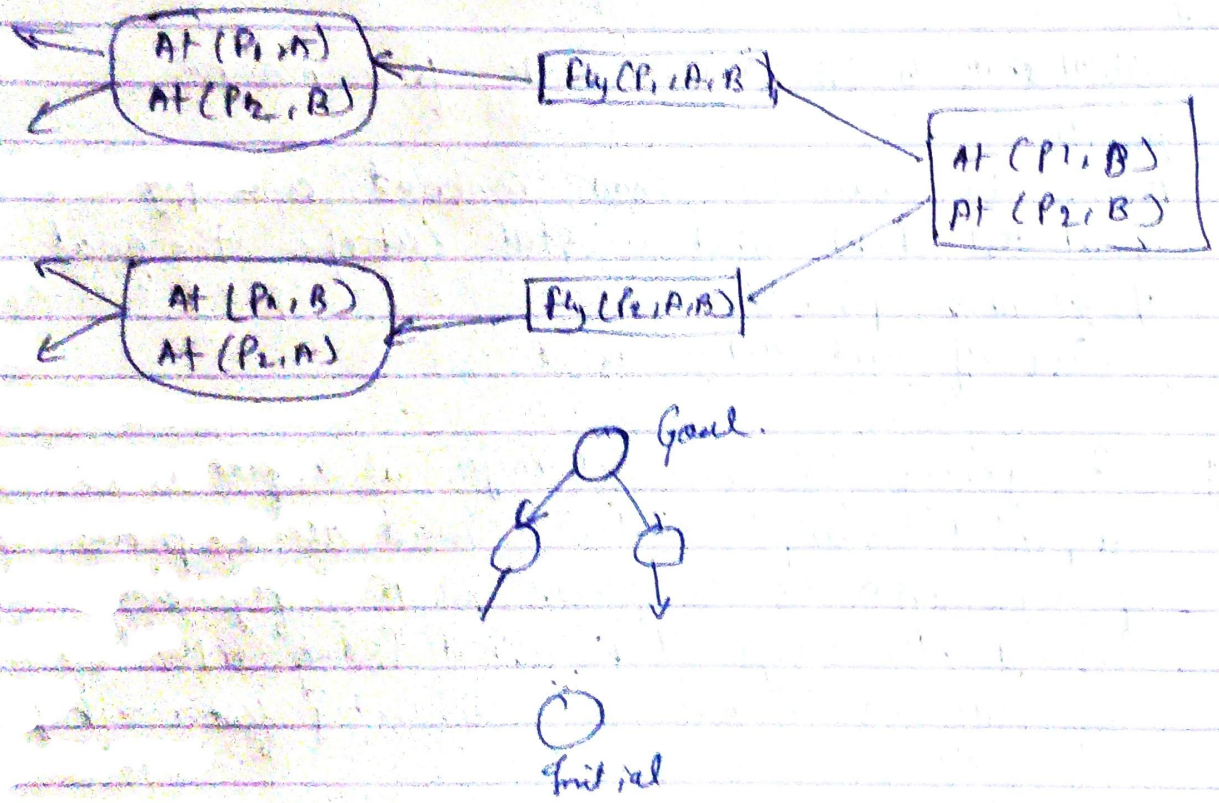


- This search is most useful when initial data is provided and it is not clear what the goal is
- Initial state is the initial state from planning problem
- Actions here can be load, unload and fly.
- Goal test checks whether state satisfies the goal of planning problem
- Cost of each action is 1.

Eg: The goal is to move all cargo at airport A to airport B. Simple solution: load all 20 pieces of cargo into one of planes at A, fly and unload the cargo at B. But finding sol^n is difficult because avg branching factor is huge.

Backward State Space Search

- It is also known as goal directed or goal driven.
- In this, search can start at the goal and work backwards the start state by seeing what moves could lead us to ~~that~~ goal state.
- This search is useful in situation where the goal is clearly specified.
- It can be difficult when Preplanning.
- Manually it is not possible to find exact predecessor state so STRIPES representation is used



Heuristic for State Space Search