Introduction to AI

Lecture 8
Planning

Dr. Tamal Ghosh
Department of CSE
Adamas University

What is Planning?

- Defines how an agent can take advantage of the structure of a problem to construct complex plans of action.
- The task of coming up with a sequence of actions that will achieve a goal is called planning.

Introduction to Planning

- Planning is an important topic in Artificial Intelligence.
- Planning is required for every task (Job). For example, reaching a particular destination requires planning.
- To reach a particular destination, first we should find the best route and then identify a set of actions to be done at a particular time, is called as planning.
- Planning is nothing but deciding a set of actions to be performed, in agent environment based on the perception to achieve the decided goal.

Components of Planning

- For any planning system, we need the **domain description, action specification**, and **goal description**.
- A plan is assumed to be a sequence of actions.
- Each action has its own set of preconditions
- All the preconditions to be satisfied before performing the action
- Some effects can be either positive or negative.

Types of Planning

- Two types of planning
- **1. Classical planning** environments
  - fully observable, deterministic, finite, static and discrete.
- **2. Nonclassical planning** is for partially observable or stochastic environments
  - The nonclassical planning involves a different set of algorithms and agent designs

Planning Problem

- Agent Environment States are represented as valuations of state variables
- an action can be represented as a *procedure* or a *program*
- The procedures are used to compute values of state variables
- After the execution of procedure (i.e. after the action), the environment state will be changed, towards the goal.

Planning Algorithm

- Representation of planning problems—states, actions, and goals—should make it possible for planning algorithms.
- Algorithms are nothing but logical structure of the problem.
- To define an efficient algorithm, language is very important.
- STRIPS Language – the language of classical planner.

Representation of States

- Planners decompose the agent world into logical conditions, and represent a state as a conjunction of positive literals.
- In first-order state descriptions, Literals must be **ground** and **function-free.**
  - At (x, y) or At (Father(Red), Sydney) are not allowed.
- The **closed-world assumption,** that is any conditions that are not mentioned in a state are assumed **false**

Representation of Goals

- A goal is a partially specified state, represented as a conjunction of positive ground literals, such as
  - 1. Rich ^ Famous 2. At (P2,Delhi).
- A propositional state s, **satisfies** a goal g, if s contains all the atoms in g
- The propositional state Rich ^ Famous ^ Happy satisfies the goal state Rich ^ Famous.

Representation of Actions

- To perform an Action, we need Precondition (how environment should be to perform this action) and Effect (how the environment will be after performing this action).
- An action for flying a plane from one location to another is:
  - Action : Fly(p, from, to)
  - PRECOND : At (p, from) ^ Plane(p) ^ Airport(from) ^ Airport(to)
  - EFFECT : At (p, from) ^ At (p, to)
- The variables p, from, and to are different constants

Action Schema

- It represents a number of different actions can be derived
- Action schema consists of three parts
- The **action name and parameter list**—
  - for example, Fly(p, from, to)— fly is action, and p, from, to are parameters.
- The **precondition** is a conjunction of function-free positive literals, and it must be true, before the action can be executed.
  - Variables in the precondition must also appear in the action's parameter list.
- The **effect** is a conjunction of function-free literals, describing how the state changes, when the action is executed.
  - A positive literal P are declared to be true in the state, and all the negative literal P is declared to be false.
  - Variables in the effect must also appear in the action's parameter list.

Add List and Delete List

- Planning systems divide the effect into two categories
- 1. Add List, 2. Delete List.
- the **add list** for positive literals and
- the **delete list** for negative literals

Applicable Actions

- An action is **applicable** in any state, iff that satisfies the precondition, otherwise, the action has no effect.
- a first-order action schema, first all the variables in precondition will be substituted
  - At (P1, JFK) ^ At (P2, SFO) ^ Plane(P1) ^ Plane(P2) ^ Airport(JFK) ^ Airport(SFO)
    This state satisfies the precondition
  - At (p, from) ^ Plane(p) ^ Airport(from) ^ Airport(to)
- with the substitution {p=P1, from=JFK, to=SFO}
- Thus, the concrete action Fly(P1, JFK, SFO) is applicable

Solution for Planning Problem

- An action sequence, that started from the initial state, and results in a state that satisfies the goal.
- Solutions to be partially ordered sets of actions
- Every action sequence, that respects the partial order, is a solution (i.e. every action has its own solution.)

STRIPS solver for Planning Problem

- The STandford Research Institute Problem Solver (**STRIPS**), is an automated planning technique used to find a goal, by executing a domain from the initial state of domain.
- With STRIPS, we can first describe the world, (Initial state and goal state) by providing **objects, actions, preconditions, and effects**
- To describe the world, we used two categories of terms
  - States – initial state and goal state
  - Action schema – objects, actions, preconditions, and effects

Flow of STRIPS

- Once the world is described, then provide a problem set.
- A problem consists of an initial state and a goal condition.
- STRIPS can then search all possible states, starting from the initial one, executing various actions, until it reaches the goal.

Planning Domain Definition Language (PDDL)

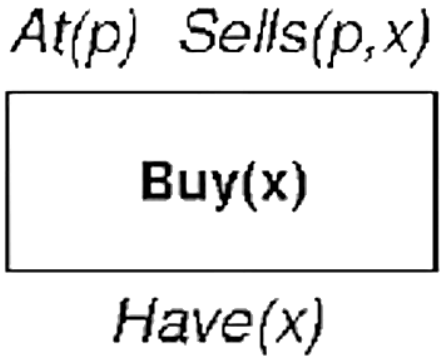
- A common language for writing STRIPS domain and problem sets, is the Planning Domain Definition Language (**PDDL**).
- In PDDL most of the codes are English words, so that it can be clearly read and well understood.
- It's a relatively easy approach to writing simple AI planning problems.

STRIPS – States, Goals, Actions

- States: conjunctions of ground, function-free, and positive literals, such as At(Home) ^ Have(Banana)
- To describe states, Closed-world assumption is used (the world model contains everything, the agent needs to know: there can be no surprise)
- Goals: conjunctions of literals, may contain variables (existential), goal may represent more than one state
  - E.g. At(Home) ^ ¬Have(Bananas)
  - E.g. At(x) ^ Sells(x, Bananas)
- Actions: preconditions that must hold before execution and the effects after execution

STRIPS Action Schema

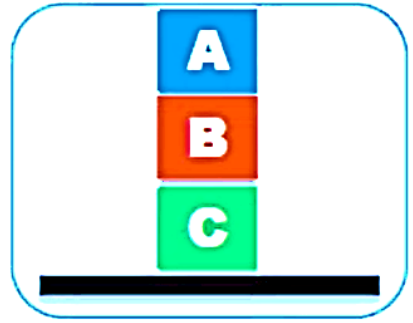
- An action schema includes:
  - Action name & parameter list (variables)
  - Precondition: a conjunction of function-free positive literals. The action variables must also appear in precondition.
  - Effect: a conjunction of function-free literals (positive or negative)
- Add-list: positive literals
- Delete-list: negative literals
- Example:
  - Action: Buy (x)
  - Precondition: At (p), Sells (p, x)
  - Effect: Have(x)



STRIPS Planning

On(A,Table)
On(B,Table)
On(C,Table)
Clear(A)
Clear(B)
Clear(C)

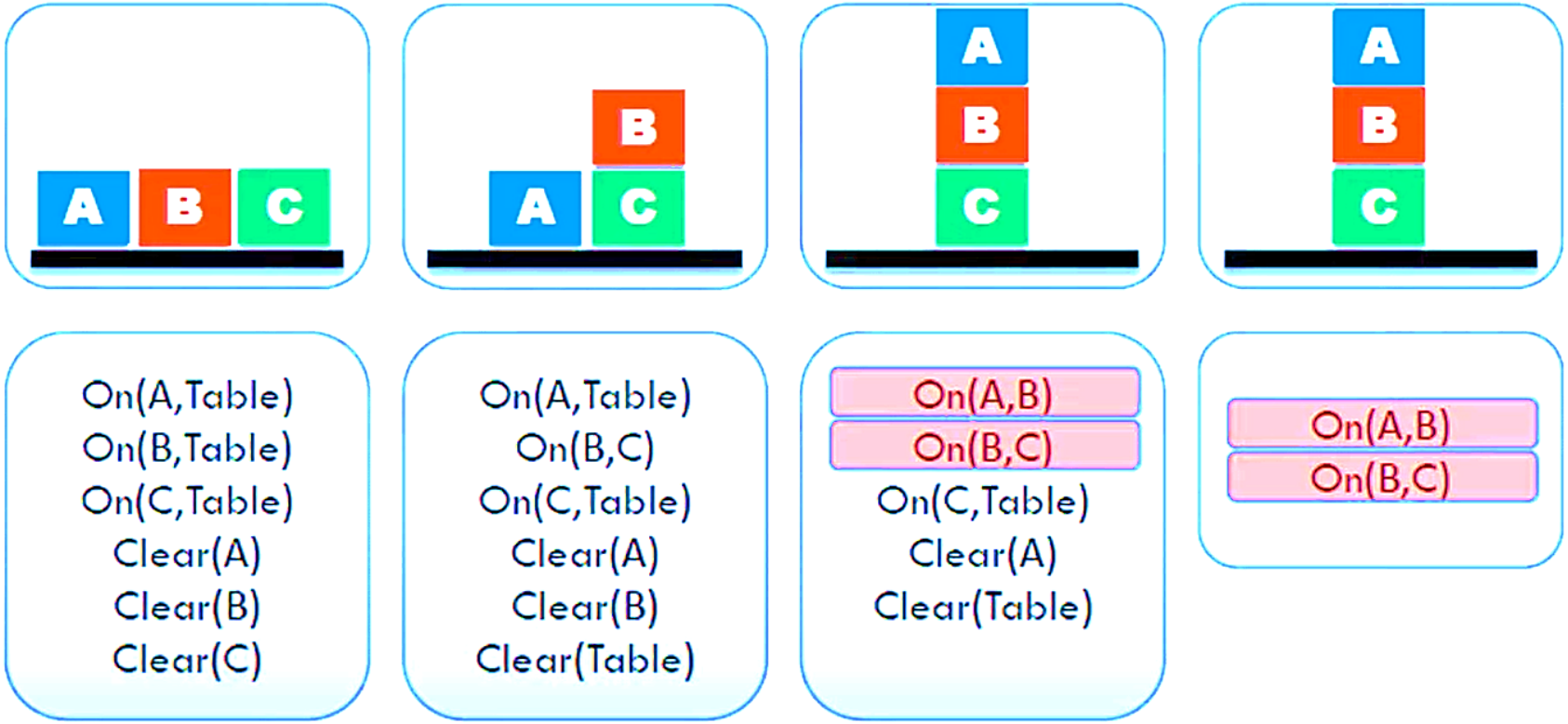
s_0



On(A,B)
On(B,C)

g

STRIPS Planning



Air Cargo Transport

- An air cargo transport problem involving loading cargo onto planes, and unloading cargo off of planes, and flying the planes from one place to another place.
- The problem can be defined with three actions:
  - Load, Unload, and Fly.
- The actions affect two predicates:
  - In(c, p) means that cargo c is inside plane p, and At (x, a) means that object x (either plane or cargo) is at airport a.
- The following plan is a solution to the problem:

  [Load(C1, P1, SFO), Fly(P1, SFO, JFK)

  Load(C2, P2, JFK), Fly(P2, JFK, SFO)]

A STRIPS problem involving transportation of air cargo between airports.

$$
\begin{aligned}
&Init(At(C_1, SFO) \land At(C_2, JFK) \land At(P_1, SFO) \land At(P_2, JFK) \\
&\quad \land Cargo(C_1) \land Cargo(C_2) \land Plane(P_1) \land Plane(P_2) \\
&\quad \land Airport(JFK) \land Airport(SFO)) \\
&Goal(At(C_1, JFK) \land At(C_2, SFO)) \\
&Action(Load(c, p, a), \\
&\quad \text{PRECOND}: At(c, a) \land At(p, a) \land Cargo(c) \land Plane(p) \land Airport(a) \\
&\quad \text{EFFECT}: \neg At(c, a) \land In(c, p)) \\
&Action(Unload(c, p, a), \\
&\quad \text{PRECOND}: In(c, p) \land At(p, a) \land Cargo(c) \land Plane(p) \land Airport(a) \\
&\quad \text{EFFECT}: At(c, a) \land \neg In(c, p)) \\
&Action(Fly(p, from, to), \\
&\quad \text{PRECOND}: At(p, from) \land Plane(p) \land Airport(from) \land Airport(to) \\
&\quad \text{EFFECT}: \neg At(p, from) \land At(p, to))
\end{aligned}
$$