Planning Domain Definition Language (PDDL)

Lab Assignment – Action Specification and Planning

This lab assignment focuses on the formal representation of **actions**, **preconditions**, and **effects** in Planning Domain Definition Language (PDDL). You will model two classical AI planning problems using predicates, objects, and operators to describe dynamic world transitions.

Question 1: Gripper Task Planner

A robot operates between two rooms and manipulates balls using two gripper arms. Initially, the robot and all balls are in the first room. The objective is to have all the balls moved into the second room. This problem demonstrates multi-object manipulation in a constrained environment.

Objects

Rooms: rooma, roomb
Balls: ball1, ball2, ball3, ball4
Robot arms (grippers): left, right

Predicates

(ROOM ?x) – ?x is a room.
(BALL ?x) – ?x is a ball.
(GRIPPER ?x) – ?x is a gripper arm.
(at-robby ?x) – the robot is in room ?x.
(at-ball ?x ?y) – ball ?x is in room ?y.
(free ?x) – gripper ?x is empty.
(carry ?x ?y) – gripper ?x holds ball ?y.

Initial State

• All rooms, balls, and grippers exist.
• The robot and all balls are in rooma.
• Both grippers are free.
• at-robby(rooma) and at-ball(ball1–4, rooma) are true.

Goal Specification

All balls should be in roomb. Formally:
(at-ball(ball1, roomb)), (at-ball(ball2, roomb)), (at-ball(ball3, roomb)), (at-ball(ball4, roomb)).

Actions / Operators

Action: Move(x, y)

Preconditions: ROOM(x), ROOM(y), and at-robby(x).

Effects: Robot moves from x to y. (at-robby(y)) becomes true; (at-robby(x)) becomes false.

Action: Pick-Up(x, y, z)

Preconditions: BALL(x), ROOM(y), GRIPPER(z), at-ball(x, y), at-robby(y), and free(z).

Effects: (carry(z, x)) becomes true; (at-ball(x, y)) and (free(z)) become false.

Action: Drop(x, y, z)

Preconditions: BALL(x), ROOM(y), GRIPPER(z), carry(z, x), and at-robby(y).

Effects: (at-ball(x, y)) and (free(z)) become true; (carry(z, x)) becomes false.

Question 2: Hospital Emergency Planner

Ambulances must pick up patients from different locations and deliver them to hospitals. Each ambulance can carry one patient at a time. This problem models a city emergency response system where the ambulances, hospitals, and patients are represented through appropriate predicates and actions.

Objects

Ambulances: a1, a2
Patients: p1, p2, p3
Locations: loc1, loc2, loc3
Hospitals: h1, h2

Predicates

(at ?a ?l) – ambulance ?a is at location ?l.
(patient-at ?p ?l) – patient ?p is at location ?l.
(in-ambulance ?p ?a) – patient ?p is inside ambulance ?a.
(connected ?l1 ?l2) – locations ?l1 and ?l2 are connected.
(blocked ?l1 ?l2) – route between ?l1 and ?l2 is blocked.
(empty ?a) – ambulance ?a is empty.

Initial State

• Two ambulances and three patients are placed across locations.
• Some routes are connected, some blocked.
• Both ambulances start empty.
• Example: (at a1 h1), (at a2 h2), (patient-at p1 loc1), (patient-at p2 loc2), (patient-at p3 loc3).

Goal Specification

All patients must be delivered to hospitals.
Example: (patient-at p1 h1), (patient-at p2 h2), (patient-at p3 h1).

Actions / Operators

Action: Move(a, from, to)

Preconditions: At(a, from), Connected(from, to), and not Blocked(from, to).

Effects: (at(a, to)) becomes true; (at(a, from)) becomes false.

Action: Pickup-Patient(p, a, l)

Preconditions: At(a, l), PatientAt(p, l), and Empty(a).

Effects: (in-ambulance(p, a)) becomes true; (patient-at(p, l)) and (empty(a)) become false.

Action: Dropoff-Patient(p, a, h)

Preconditions: At(a, h), InAmbulance(p, a).

Effects: (patient-at(p, h)) and (empty(a)) become true; (in-ambulance(p, a)) becomes false.