**An Introduction to PDDL**

What is PDDL?

PDDL = Planning Domain Definition Language (standard encoding language for “classical” planning tasks.

Components of a PDDL planning task:

* **Objects**: things in the world that interest us.
* **Predicates**: properties of objects that we are interested in; can be true or false.
* **Initial state**: the state of the world that we start in.
* **Goal specification**: things that we want to be true.
* **Actions/Operators**: ways of changing the state of the world.

How to put the pieces together

Planning tasks specified in PDDL are separated into two files:

* A domain file, for predicates and actions.
* A problem file, for objects, initial state ang goal specification.

*Domain Files*

A domain file looks like this:

(define (domain <domain name>)

<PDDL code for predicates>

<PDDL code for fist action>

[…]

<PDDL code for last action>

)

<domain name> is a string that identifies the planning domain, e.g., gripper.

*Problem Files*

A problem file looks like this:

(define (problem <problem name>)

(:domain <domain name>)

<PDDL code for objects>

<PDDL code for initial state>

<PDDL code for goal specification>

)

<problem name> is a string that identifies the planning task, e-g-, gripper-four-balls.

<domain name> must match the domain name in the corresponding domain file.

Running Example

Gripper task with four balls: there is a robot that can move between two rooms and pick up or drop off balls with either of his two arms. Initially, all balls and the robot are in the first room. We want the balls to be in the second room.

* Objects: the two rooms, four balls and two robot arms.
* Predicates: is x a room? Is x a ball? Is the ball x inside room y? is robot arm x empty? […]
* Initial state: all balls and the robot are in the first room. All robot arms are empty. […]
* Goal specification: all balls must be in the second room.
* Actions/Operators: the robot can move between rooms, pick up a ball or drop off a ball.

*Gripper task: Objects*

Objects:

* Rooms: rooma, roomb
* Balls: ball1, ball2, ball3, ball4
* Robot arms: left, right

In PDDL:

( :objects rooma roomb

ball1 ball2 ball3 ball4

left right)

*Gripper task: Predicates*

Predicates:

* ROOM(x) -> true if x is a room
* BALL(x) -> true if x is a ball
* GRIPPER(x) -> true if x is a gripper (robot arm)
* at-robby(x) -> true if x is a room and the robot is in x
* at-ball(x, y) -> true if x is a ball, y is a room and x is in y
* free(x) -> true if x is a gripper and x does not hold a ball
* carry(x, y) -> true if x is a gripper, y is a ball and x holds y

In PDDL:

( :predicates (ROOM ?x) (BALL ?x) (GRIPPER ?x)

(at-robby ?x) (at-ball ?x ?y)

(free ?x) (carry ?x ?y))

*Gripper task: Initial State*

Initial state:

* ROOM(rooma) and ROOM(roomb) are true.
* BALL(ball1), …, BALL(ball4) are true.
* GRIPPER(left), GRIPPER(right), free(left) and free(right) are true.
* at-robby(rooma), at-ball(ball1, rooma), …, at-ball(ball4, rooma) are true.

In PDDL:

( :init (ROOM rooma) (ROOM roomb)

(BALL ball1) (BALL ball2) (BALL ball3) (BALL ball4)

(GRIPPER left) (GRIPPER right) (free left) (free right)

(at-robby romma)

(at-ball ball1 rooma) (at-ball ball2 rooma)

(at-ball ball3 rooma) (at-ball ball4 rooma) )

*Gripper task: Goal Specification*

Goal specification

at-ball(ball1, roomb), …, at-ball(ball4, roomb) must be true.

Everything else we don’t care about.

In PDDL:

( :goal (and (at-ball ball1 roomb)

(at-ball ball2 roomb)

(at-ball ball3 roomb)

(at-ball ball4 roomb) ) )

*Gripper task: Movement Operator*

Action/Operator:

* Description: The robot can move from x to y.
* Precondition: ROOM(x), ROOM(y) and at-robby(x) are true.
* Effect: at-robby(y) becomes true. at-robby(x) becomes false.

Everything else doesn’t change.

In PDDL:

( :action move :parameters (?x ?y)

:precondition (and (ROOM ?x) (ROOM ?y)

(at-robby ?x)

:effect (and (at-robby ?y)

(not (at-robby ?x) ) ) )

*Gripper task: Pick-up Operator*

Action/Operator

* Description: the robot can pick up x in y with z.
* Precondition: BALL(x), ROOM(y), GRIPPER(z), at-ball(x, y).

at-robby(y) and free(z) are true.

* Effect: carry(z, x) becomes true. at-ball(x, y) and free(z) becomes false.

Everything else doesn’t change.

In PDDL:

( :action pick-up :parameters (?X ?y ?z)

:precondition (and (BALL ?x) (ROOM ?y) (GRIPPER ?z)

(at-ball ?x ?y) (at-robby ?y) (free ?z))

:effect (and (carry ?x ?y)

(not (at-ball ?x ?y) ) (not (free ?z) ) ) )

*Gripper task: Drop Operator*

Action/Operator:

* Description: the robot can drop x in y from z.

(Preconditions and effects are similar to the pick-up operator).

In PDDL:

( :action drop :parameters (?X ?y ?z)

:precondition (and (BALL ?x) (ROOM ?y) (GRIPPER ?z)

(at-ball ?z ?x) (at-robby ?y) )

:effect (and (at-ball ?x ?y) (free ?z)

(not (carry ?z ?x) ) ) )