QUIZ 2

Hoang-Dung Bui,
George Mason University
Fairfax, USA
hbui20@gmu.edu

I. Problem Setting

In this quiz, I would like to set up a cooking robot. The robot is a manipulator with 5-finger effector configuration. It can make several foods and drink such as rice, pasta, coffee, tea, ... etc. During preparation, one robot-hand holds an equipment (such as cup, pot, pan...) and the other will get ingredient and water. As all the ingredient is ready, the robot will move the equipment to the cooking-spot, and the final action cook.

The domain of robot is as following:

- There are some type of objects:
  - *location supply - object*
  - *cooking-facility manipulator - supply*
  - *ingredient equipment cooking-facility*
  - *food drink - ingredient*
  - *rice pasta - food*
  - *tea instance-coffee water - drink*
  - *cooking-spot drink-spot rice-holder tea-holder coffee-holder faucet cooking-shelf rest-position table - location*
  - *pot kettle cup bowl - equipment*
- There some predicates:
  - *(at ?s - supply ?l - location)*
  - *(on ?s - cooking-facility ?l - location)*
  - *(hold ?r - manipulator ?cf - cooking-facility)*
  - *(inside ?ing - ingredient ?e - equipment)*
  - *(fd-ready ?ing - ingredient)*
  - *(hot ?w - water)*
  - *(empty-hand ?r - manipulator)*
  - *(occupied ?l - location ?e - equipment)*
- There are several action as following:
  - *move*
  - *get-equipment*
  - *release-equipment*
  - *put-ingredient-in*
  - *pour-water-in*
  - *cook-rice*
  - *stirring-coffee-done*
  - *pour-hotwater-into*
  - *boil-water*
- There is a cost variable *total-cost* used to estimate the cost to cook food. It can be used in the future extension.

II. Operators

The detail of the operators are described as following:

1) *get-ingredient-in*: get an ingredient from one location by a manipulator and holding it,
2) *move* move equipment between two locations by a manipulator,
3) *get-equipment*: get equipment at a location by a manipulator and hold it,
4) *release-equipment*: a manipulator releases equipment at a location,
5) *put-ingredient-in*: put an ingredient into an equipment,
6) *pour-water-in*: pour water from faucet in an equipment,
7) *cook-rice*: As water and ingredients are in the equipment at the cooking-spot, make food.
8) *boil-water*: prepare hot water to make tea or instance coffee.
9) *pour-hotwater-into*: pour hot water between two equipments (i.e. kettle and cup) by a manipulator.
10) *stirring-coffee-done*: stirring the mixing of coffee and hot water, and then the coffee is ready to use.

III. Operator's Implementation

In this section, we will describe several operators which are implemented in PDDL. We assume that at each location, there is a device such as a table, or frame, or shelf which equipments or ingredients can lie on it.

put-ingredient-in operator: The condition to perform this action is : (1) Does robot's free, (2) Is robot at the predefined location? and (3) is the ingredient on a table at the location and not attached to the robot? and (4) is the equipment to put the ingredient in, on a table at the location (such as a table)? The input's parameters are: robot, equipment, location, and ingredient. If the operator is performed, it changes the *ingredient-in-equipment* status and increases the cost of cooking. The list of ingredient can be get: rice, pasta, tea, instance-coffee.

```
(:action put-ingredient-in
  :parameters (?r - manipulator ?e - equipment ?ing - ingredient ?l - location)
  :precondition (and
        (empty-hand ?r)
        (at ?r ?l)
        (on ?ing ?l)
        (on ?e ?l)
        )
  :effect (and
        (inside ?ing ?e)
        (empty-hand ?r)
        (not (at ?ing ?l))
        (increase (total-cost) 1)
        )
)
```

Fig. 1

get-equipment operator: Similar to *get-ingredient* operator,

the condition to perform this action is : (1) Is robot's end-effector empty?, (2) Is robot at the predefined location? and (3) at the location, is there the required equipment? The input's parameters are: robot, equipment, location and empty state of hand. If the operator is performed, it changes the state of the end-effector from empty to hold the equipment. It then increases the cost of cooking. The list of equipment consists: cup, pot, pan.

```
(:action get-equipment
  :parameters (?r - manipulator ?e - equipment ?l - location)
  :precondition (and
                (empty-hand ?r)
                (at ?r ?l)
                (on ?e ?l))
  :effect ( and
    (hold ?r ?e)
    (not (empty-hand ?r))
    (not (on ?e ?l))
    (increase (total-cost) 1)))
```

Fig. 2

For the **cook-rice** operator, it checks whether the equipment lies on the cooking-area, is ingredient and water are filled out the equipment, and it is the right location. As all the condition is correct, it will set the *fd-ready* predicate to true (the food is done).

```
(:action cook-rice
  :parameters (?p - pot ?cp - cooking-spot  ?ing - ingredient ?w - water)
  :precondition (and
    (on ?p ?cp)
    (inside ?ing ?p)
    (inside ?w ?p)
    )
  :effect ( and
    (fd-ready ?ing)
    (increase (total-cost) 3)
  )
)
```

Fig. 3

IV. Problems

A. Problem 1 - Cooking Rice

In this problem, we would like to cook rice as an common Asian food. The setting, initial conditions, and goal are defined as the following.

1) objects
   a) rice1 - rice
   b) water1 - water
   c) cooking-spot1 - cooking-spot
   d) rice-holder1 - rice-holder
   e) faucet1 - faucet
   f) pot1 - pot
   g) robot - manipulator
   h) rest-loc1 - res-location
   i) cooking-shelf1 cooking-shelf
2) *initial condition*
   - (at robot rest-loc1)
   - (on rice1 rice-holder1)
   - (on pot1 cooking-shelf1)
   - (at water1 faucet1)
   - (empty robot)
   - (not (hot water1))
   - (= (total-cost) 0)
3) the goal: (:goal (fd-ready rice1))

The result for this problem is following (solved by my PDDL code):

- *step 1*: move robot res-loc1 cooking-shelf1 (3)
- *step 2*: get-equipment robot pot1 cooking-shelf1 (1)
- *step 3*: move robot cooking-shelf1 faucet1 (3)
- *step 4*: pour-water-in robot pot1 faucet1 water1 (1)
- *step 5*: move robot faucet1 rice-holder1 (3)
- *step 6*: release-equipment robot pot1 rice-holder1 (1)
- *step 7*: put-ingredient-in robot pot1 rice1 rice-holder1 (1)
- *step 8*: get-equipment robot pot1 rice-holder1 (1)
- *step 9*: move robot rice-holder1 cooking-spot1 (3)
- *step 10*: release-equipment robot pot1 cooking-spot1 (1)
- *step 11*: cook-rice pot1 cooking-spot1 rice1 water1 (3)

The Plan length is 11 step(s) with plan cost of 21, Expanded 12 state(s), Evaluated 38 state(s), and Generated 58 state(s).

B. Problem 2 - Make an instance-coffee

In this problem, we would like to make a drink (coffee). The setting, initial conditions, and goal are defined as following.

1) objects
   a) coffee1 - coffee
   b) water1 - water
   c) drink-spot1 - drink-spot
   d) cooking-spot1 - cooking-spot
   e) coffee-holder1 - coffee-holder
   f) faucet1 - faucet
   g) kettle1 - kettle
   h) cup1 - coffee-mud
   i) robot - manipulator
   j) res-loc1 - res-location
   k) cooking-shelf1 - cooking-shelf
2) *initial conditions are:*
   - (at robot res-loc1)
   - (on coffee1 coffee-holder1)
   - (on kettle1 cooking-shelf1)
   - (at water1 faucet1)
   - (on cup1 - drink-shelf1)
   - (empty-hand robot)
   - (not (hot water1))
   - (= (total-cost) 0)
3) the goal: (:goal (and (df-ready coffee1) (on cup1 drink-spot1))

This problem is even more challenge than problem 1, because it consists of transferring action hot water from the kettle into the cup to make the coffee. If not defining states carefully, the planner usually gets water into the kettle, boil it; then it pours cold water from the faucet into the cup, not the hot one from the kettle. To fix this, we set the number of time to get water to 1, so the planner must use the given water. The step to make a cup of instance coffee as the following:

The list of actions should be following:

1) *step 1*: move robot res-loc1 cooking-shelf1 (3)

2) *step 2*: get-equipment robot kettle1 cooking-shelf1 (1)
3) *step 3*: move robot cooking-shelf1 faucet1 (3)
4) *step 4*: pour-in robot kettle1 faucet1 water1 (1)
5) *step 5*: move robot faucet1 cooking-spot1 (3)
6) *step 6*: release-equipment robot kettle1 cooking-spot1 (1)
7) *step 7*: boil-water kettle1 cooking-spot1 water1 (2)
8) *step 8*: get-equipment robot kettle1 cooking-spot1 (1)
9) *step 9*: move robot cooking-spot1 drink-spot1 (3)
10) *step 10*: pour-hotwater-tocup robot kettle1 cup1 drink-spot1 water1 (2)
11) *step 11*: release-equipment robot kettle1 drink-spot1 (1)
12) *step 12*: get-equipment robot cup1 drink-spot1 (1)
13) *step 13*: move robot drink-spot1 coffee-holder1 (3)
14) *step 14*: release-equipment robot cup1 coffee-holder1 (1)
15) *step 15*: get-put-ingredient-in1 robot cup1 coffee1 coffee-holder1 (1)
16) *step 16*: get-equipment robot cup1 coffee-holder1 (1)
17) *step 17*: move robot coffee-holder1 drink-spot1 (3)
18) *step 18*: release-equipment robot cup1 drink-spot1 (1)
19) *step 19*: stirring-coffee-done cup1 coffee1 water1 (0)

The Plan length is 19 step(s) with plan cost of 32, Expanded 20 state(s), Evaluated 86 state(s), and Generated 125 state(s).