Robot Autonomy – Homework 3

Ankit Aggarwal - ankitagg@

2. Propositional States and Actions

The three actions are given as:

Action 1: Move to Pantry

Description: The robot can move to a pantry if it is in the kitchen

```
### Move to Pantry
Precond=np.zeros([nrObjects, nrPredicates])
# TODO: Robot in the kitchen and Robot not in the pantry
Precond[0][1]=  1  # Robot in the kitchen
Precond[0][5]= -1 # Robot not in the pantry

Effect=np.zeros([nrObjects, nrPredicates])
# TODO: Move robot from the kitchen to the pantry (remove from kitchen and add to pantry)
Effect[0][1]= -2 # Robot not in the kitchen
Effect[0][5]=  2  # Robot in the pantry

ActionPre.append(Precond)
ActionEff.append(Effect)
ActionDesc.append("Move to Pantry from Kitchen")
```

Action 2: Move to Kitchen

Description: The robot can move to the kitchen if it is in the pantry

```
### Move from Pantry
Precond=np.zeros([nrObjects, nrPredicates])
# TODO: Robot not in the kitchen and Robot in the pantry
Precond[0][1]= -1 # Robot not in the kitchen
Precond[0][5]=  1  # Robot in the pantry

Effect=np.zeros([nrObjects, nrPredicates])
# TODO: Move robot from the pantry to the kitchen (remove from pantry and add to kitchen)
Effect[0][1]=  2  # Robot in the kitchen
Effect[0][5]= -2 # Robot not in the pantry

ActionPre.append(Precond)
ActionEff.append(Effect)
ActionDesc.append("Move to Kitchen from Pantry")
```

Action 3: Cut Fruit

Description: The robot can cut fruit if the robot, knife, and fruit are in the kitchen and the fruit has not been cut.

```
###Cut fruit in kitchen
for j in [1,2]:
    Precond=np.zeros([nrObjects, nrPredicates])
    # TODO: Robot in the kitchen, fruit in the kitchen, knife in the kitchen, fruit not chopped
    Precond[0][1]=  1
    Precond[j][1]=  1 # fruit j in kitchen
    Precond[4][1]=  1
    Precond[j][6]= -1 # fruit j not chopped

    Effect=np.zeros([nrObjects, nrPredicates])
    # TODO: Fruit is chopped
    Effect[j][6]=  2  # fruit j is chopped

    ActionPre.append(Precond)
    ActionEff.append(Effect)
    ActionDesc.append("Cut "+Objects[j]+" in the kitchen")
```

3. Discrete Search

Searching for a plan from the initial state to a goal state.

Djisktra Search

Number of vertices/states explored = 5894
Length of Plan = 16

Screenshot of the final plan

```
(mujuco) burger@burger-LOQ-15AHP9:~/Autonomy/16_662_HW3$ /usr/bin/python3 /home/burger/Autonomy/16_662_HW3/SymDiscreteSearch.py
Path Found:  True
States Explored:  5894
Plan Length:  16

Plan:
Move to InKitchen from InHallway
Move to Pantry from Kitchen
Pick up Lemon from InPantry
Move to Kitchen from Pantry
Move to InHallway from InKitchen
Move to InOffice from InHallway
Pick up Knife from InOffice
Move to InHallway from InOffice
Move to InGarden from InHallway
Pick up Strawberry from InGarden
Place Lemon at InGarden
Move to InHallway from InGarden
Move to InKitchen from InHallway
Place Strawberry at InKitchen
Place Knife at InKitchen
Cut Strawberry in the kitchen
```

A* Search

Number of vertices/states explored = 2005
Length of Plan = 16

Screenshot of the Final Plan

```
(mujuco) burger@burger-LOQ-15AHP9:~/Autonomy/16_662_HW3$ /usr/bin/python3 /home/burger/Autonomy/16_662_HW3/SymDiscreteSearch.py
Path Found:  True
States Explored:  2005
Plan Length:  16

Plan:
Move to InKitchen from InHallway
Move to Pantry from Kitchen
Pick up Lemon from InPantry
Move to Kitchen from Pantry
Move to InHallway from InKitchen
Move to InGarden from InHallway
Pick up Strawberry from InGarden
Place Lemon at InGarden
Move to InHallway from InGarden
Move to InOffice from InHallway
Pick up Knife from InOffice
Move to InHallway from InOffice
Move to InKitchen from InHallway
Place Strawberry at InKitchen
Place Knife at InKitchen
Cut Strawberry in the kitchen
```

Cost Function Snippet

```
# Djikstra: cost is the cost to come
cost = cost2come[-1]

# Comment the heuristic cost for Dijkstra
# A*: cost is cost to come + heuristic
heuristic_cost = Heuristic(nextState, GoalIndicesOneStep, GoalIndicesTwoStep)
cost += heuristic_cost
```