Ira A. Fulton Schools of Engineering
Arizona State University
Engineering | School of Computing, Informatics, and Decision Systems Engineering

CSE 571: Artificial Intelligence

Tools for Sequential Decision-Making: Extended Project

Purpose:

Planning Domain Definition Language (PDDL) strives to standardize representations of sequential decision-making problems. Using a standardized language supports successful collaboration across project teams to solve problems efficiently. The purpose of this project is to challenge students to analyze a domain file and determine how to modify it to get the AI agent to achieve the desired goal.

Objectives:

Students will be able to:

- Analyze domain files.
- Determine errors in faulty domain files.
- Determine solutions to achieve desired goals.
- Design a domain file.
- Design a problem file.
- Utilize digital tools and planners to solve sequential decision-making problems.
- Identify extensions required to model more general classes of sequential decision-making problems.
- Represent an environment to solve using Planning Domain Definition Language (PDDL).
- Execute steps in a sequential decision-making process to achieve specific goals.
- Modify code to achieve optimal outcomes.

Technology Requirements:

- Personal computer with reliable WiFi
- It is recommended to use the Fast-Downward Planner to verify your solution, but it is not required.

Project Support:

Running Your Code

- If needed to verify your solution, you are strongly encouraged to install the Fast-Downward Planner, which contains all necessary information for installing and using the planner.
    - Interested parties may learn more about the planner on the Fast-Downward Home Page.
- Your domain/problem file, if correctly designed will yield a solution quickly (in under 5 seconds).
    - Time- outs could mean that something is wrong with your domain or problem file.
- Your code results inform your selection of correct answers.

Project Strategy Questions:

This project is based on Pac-Man, a very famous arcade game popularized in the 1980s and often noted as one of the most famous video games of all time.

Maze 1

X_1Y_1, Start	X_1Y_2	X_1Y_3	X_1Y_4, Goal
X_2Y_1	X_2Y_2	X_2Y_3	X_2Y_4
X_3Y_1	X_3Y_2	X_3Y_3	X_3Y_4
X_4Y_1	X_4Y_2	X_4Y_3	X_4Y_4

Question 1

Review Maze 1, a 4X4 maze with locations defined as XiYi. It shows Pac-Man's current location as **Start** and his goal location as **Goal**. There are no walls or food pellets in Maze 1. Those will be added to other mazes as you continue progressing through the project.

Two files are required to represent any environment that you want to solve using PDDL:

1. **Problem file** - which would contain information about your objects in the environment, your initial state and a goal state.
2. **Domain file** - which would contain list of predicates in use, types for objects (similar to data type in any programming language) and actions.

A partially filled problem file (q1_problem.pddl) and a complete domain file (q1_domain.pddl) representing Maze 1 is provided for your testing.

Task: Write a goal condition so that solution plans will move Pac-Man to the goal location shown in Maze1. (To verify your solution, you can add the goal condition inside the partially filled

problem file and run it using the planner.) *Refer to the project area in the course to view and select answers.*

Maze 2

X_1Y_1, Goal	X_1Y_2	X_1Y_3	X_1Y_4
X_2Y_1	X_2Y_2	X_2Y_3	X_2Y_4
X_3Y_1	X_3Y_2	X_3Y_3	X_3Y_4
X_4Y_1	X_4Y_2	X_4Y_3	X_4Y_4,Pacman

Question 2

Review Maze 2. Maze 2 shows Pac-Man's start location as **Pacman** and goal location as **Goal**. Now, we will block the location X1Y3 such that the Pac-Man cannot move into this location.

Which propositions do you need to remove from the Initial State (:init) of the problem file (q2_problem.pddl) to incorporate this information (Pac-Man should not move into the blocked location)? (To verify your solution, you must use this updated q2_problem.pddl as problem file, q2_domain.pddl as domain file and run it using the planner). *Refer to the project area in the course to view and select answers.*

Question 3

After all the hard work we have put Pac-Man through, he is hungry, so food pellets will be added to Maze 2. Suppose there are food pellets at the locations X1Y4, X2Y1, X3Y3 and X4Y4. You

will need to add a predicate **dot_at (dot-at loc_x98y98)** indicating whether or not a dot is at a location in the initial state (you can edit the initial state inside the problem file q3_problem.pddl and to verify your solution, you would first have to solve Question 4).

Which option incorporates the scenario in this question? *Refer to the project area in the course to view and select answers.*

Question 4

Similarly to Question 3, your task will be to help Pac-Man eat all the food pellets present in locations X1Y4, X2Y1, X3Y3 and X4Y4. You will need to add a predicate **eaten (eaten loc_x98y98)** in the goal condition (you can add the goal condition inside the problem file q3_problem.pddl) indicating that the food at a particular location is eaten by Pac-Man. (To verify your solution for this question and the previous question, you must use the updated q3_problem.pddl as problem file, q3_domain.pddl as domain file and run it using the planner).

Which option, when added as a goal condition, will make Pac-Man eat *all* the food pellets? *Refer to the project area in the course to view and select answers.*

Question 5

We made Pac-Man go through a lot of trouble and because of that he is feeling very sleepy. Now, if Pac-Man tries to make a move, he will succeed with only 70% probability and will stay at the same location with 30% probability. Given this condition, can you figure out how the "effect" for action "move-West" will change? *Refer to the project area in the course to view and select answers.*

Note: You do not need to run code to answer this question.

Submission Directions for Project Deliverables

You will not be submitting your code for this project.

Refer to the project space in the course to view, select, and submit your answers.

Evaluation

Your answers to each question are individually evaluated by the auto-grader.