TÉCNICO LISBOA

IASD 2023/24 Project Assignment #3: Autonomous shared vehicle transportation system

Rodrigo Ventura and Luis Custódio
Instituto Superior Técnico, University of Lisbon

(Version 1.0, October 12, 2023)

1 Introduction

The project for this assignment remains the same. The system receives as input a set of transportation requests from users, and schedules a fleet of autonomous vehicles to drive users from their pickup points to the desired destinations. Each vehicle has a pre-defined amount of seats. The goal is to maximize efficiency in passenger transportation.

2 Second Assignment and Deliverable

This assignment is a followup of both Assignment #1, where the problem was properly formalized, and of Assignment #2, aiming at obtaining optimal solutions to the problem

using uninformed search methods. In this assignment, the goal is to use **informed search methods**, using algorithms from `search.py` from the IASD course book repository[1]. As in the previous assignment, you only need to import search.py, which in turn imports utils.py.

The Python program to be delivered should be called `solution.py` and include (at least) a class with name `FleetProblem` containing (at least) the methods required in Assignment #2 (see Annex A for the class template), as well as your heuristic function as the following class method:

`h(state)` returns the heuristic of the given state.

In order to solve a problem one needs to:

- read the problem from a file object like you did for Assignment #1,
- you may change the state representation from Assignment #2 if you find a better one; in that case, methods `result`, `actions`, `goal_test`, and `path_cost` need to be adjusted accordingly;
- design and implement a good heuristic function `h`,
- choose one **informed** search algorithm that the group considers best fits the problem at hand, and use it in method `solve`. See `search.py` for the set of informed search algorithms available.

3 Evaluation

The deliverable for this assignment is shown through DEEC Moodle, with the submission of a single python file, called `solution.py`, implementing the modules mentioned above. Instructions for this platform are available on the course webpage. Finally, the grade is computed in the following way:

- 50% from the public tests;
- 50% from the private tests; and
- -10% from the code structure, quality and readability.

Deadline: **27-October-2023**.
Projects submitted after the deadline will not be considered for evaluation.

[1]The link for the repository is `https://github.com/aimacode/aima-python`.

Closing Remarks on Ethics:

- Any kind of sharing code outside your group is considered plagiarism;
- Developing your code in any open software development tool is considered sharing code;
- You can use GitHub. Make sure you have private projects and remove them afterward;
- If you get caught in any plagiarism, either by copying the code/ideas or sharing them with others, you will not be graded; and
- The scripts and other supporting materials produced by the instructors cannot be made public!

A Class Template

```
import search

class FleetProblem(search.Problem):

    def result(self, state, action):
        """Return the state that results from executing
        the given action in the given state."""
        pass

    def actions(self, state):
        """Return the actions that can be executed in
        the given state."""
        pass

    def goal_test(self, state):
        """Return True if the state is a goal."""
        pass

    def path_cost(self, c, state1, action, state2):
        """Return the cost of a solution path that arrives at state2 from
        state1 via action, assuming cost c to get up to state1."""

    def load(self, fh):
        """Loads a problem from the file object fh.
        You may initialize self.initial here."""
        pass
```

```
def h(self, state):
    """Return the heuristic value for the given state."""
    pass

def solve(self):
    """Calls the informed search algorithm
    chosen. Returns a solution using the specified format."""
    pass
```