School of Computer Science
The University of Adelaide

Artificial Intelligence
Assignment 1

Semester 1, 2018
due 11.59, Thursday 5th April 2018

Introduction

With a suitable abstractions planning a path for a mobile robot can be converted into a search problem. Begin by abstracting the environment into 2D grid of square "cells". The robot state can be represented by a [x,y] pair, with [0,0] being the top-left cell of the grid. Movements of the robot are possible at any time either UP, DOWN, LEFT RIGHT (denoted `U`, `D`, `L`, `R`) unless the neighbouring cell is occupied. The UP action moves to the cell immediately above (if possible) by changing [x,y] to [x,y-1]. Likewise RIGHT changes the current state from [x,y] to [x+1,y], and so on. Given a start state, the goal of the robot is to reach a new (user specified) state [X*, Y*], where this must be an unoccupied cell in the grid. An action that would take the robot outside the grid or into and occupied cell results in no change to the current state.

Your task is to write a program that will read in an environment, a start state and a goal state, and conduct a search to find a path between start and goal. You may implement any of the search algorithms discussed in lectures. Your output should be in the form of a space-separated list of actions (e.g. `U R R D D L L`).

Test data are provided on the course pages along with this Assignment description. A week before the deadline more data will be made available and you must run your code on these new data and include these results in your report (see below).

You **must write the program yourself** in either C, C++, Java or Python. If you use a library package or language function call for doing the search, you will be limited to 50% of the available marks (noting that this assignment is a hurdle for the course with min mark to achieve of hurdle of 45%). If there is evidence you have simply copied code from the web, you will be awarded no marks and referred for plagiarism.

The program must accept 5 arguments from the command-line, a filename (which contains the environment) and 4 integers specifying start state and goal state, eg:

```
./robotplanner env.txt 0 2 4 1
```

would read the environment specification from the file `env.txt` and should plan a path from [0 2] to goal state [4 1]

Submission

You must submit, by the due date, a zip file containing:

1. your code
2. a document (briefly) describing your implementation and detailing your results

I will update as soon as possible whether there will be web-testing and auto-marking for your program outputs.

This assignment is **due 11.59pm on Thursday 5th April, 2018**. If your submission is late, the maximum mark you can obtain will be reduced by 25% per day (or part thereof) past the due date or any extension you are granted.

Optional component

The search algorithm you use is deliberately not specified, however extra marks will be available for a successful implementation and description of A* search. It is up to you how you define the heuristic.

File format

The environment will be stored as text file in the following format: the first line contains the width and height as two (space or tab separated) integers. Each subsequent line contains a set of space-or-tab-separated 0s and 1s, with 0 representing that a cell is free-space (and therefore navigable) and 1 indicating it is occupied, and therefore cannot be entered or passed through by the robot.

For example

```
5 3
0 0 1 0 0
1 0 1 0 0
0 0 0 0 1
```

If we suppose that the start state is [0,0] (i.e. at the top left) and the goal state is [4,0] (i.e. top right), then a valid solutions is

```
R D D R R U U R
```

Note that this solution is optimal but not unique.

Assessment

Assessment will be made of the basis of quality of code (40), quality of write-up (30) and accuracy of results (30) for a total of 100. Up to 25 bonus marks are available from successful completion of an A* implementation, but the maximum mark remains 100.

Prof. Ian Reid, 10 March 2018