CSM6120 (Year 2022-23)
Department of Computer Science, Aberystwyth University
Practical1: A* search for the maze problem (21 Oct 2022 @MP3.03)
Instructor: Prof. Tossapon Boongoen

---

0 General information

This practical allows students to match concepts of **a search problem** and the **A*** technique to a simple software **implementation in Python**. Of course, individual hand-made solutions can well be checked against that provided by that set of codes. After that, it may be modified to present a new variant(s) of the problem: size or property. And hopefully, students can adapt it further to accommodate a new problem that they might later encounter. **Note that a practical work is not an assignment, so that no submission is needed.**

1 About the problem: *maze*

A matrix of 5 rows and 6 columns represents a basic structure of the maze, in which each cell can be identified using the coordinate, e.g., (0, 0) for that in the top left corner. Then, to make it more interesting, those cells with an orange background is a no-go area, i.e., it is blocked. **A goal-based robot agent** tries to move from its **initial state** ("being at (0, 0)") to its **goal state** ("being at (4, 5)"). Given that these states are *discrete* whilst operations (4 possible moves: *left, right, up*, or *down*) are *deterministic*, can you find a path for him using the A* technique?

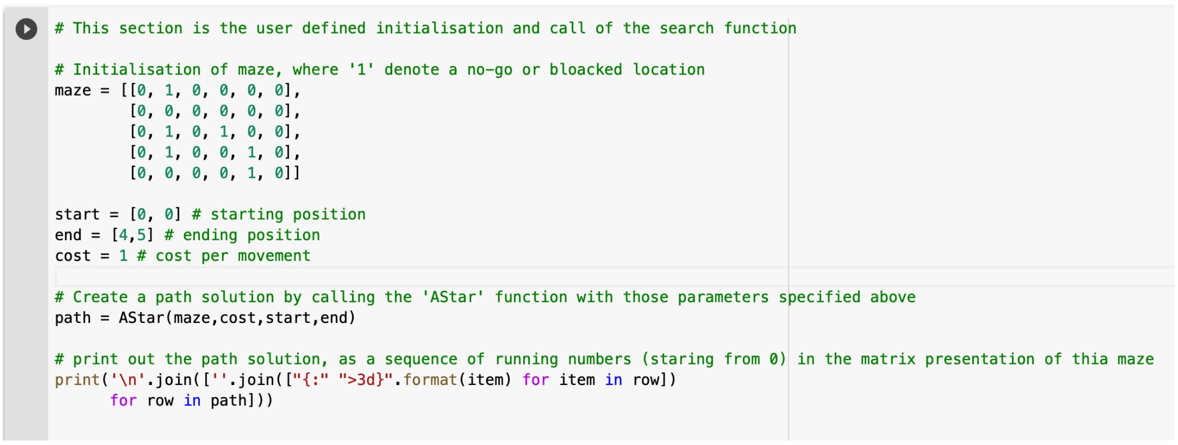
(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)

2 About the software implementation: *Python*

An implementation of this problem and the A* technique in Python can be accessed through the following link. A separate Jupyter notebook file is also provided, "Student-A-Star.ipynb".

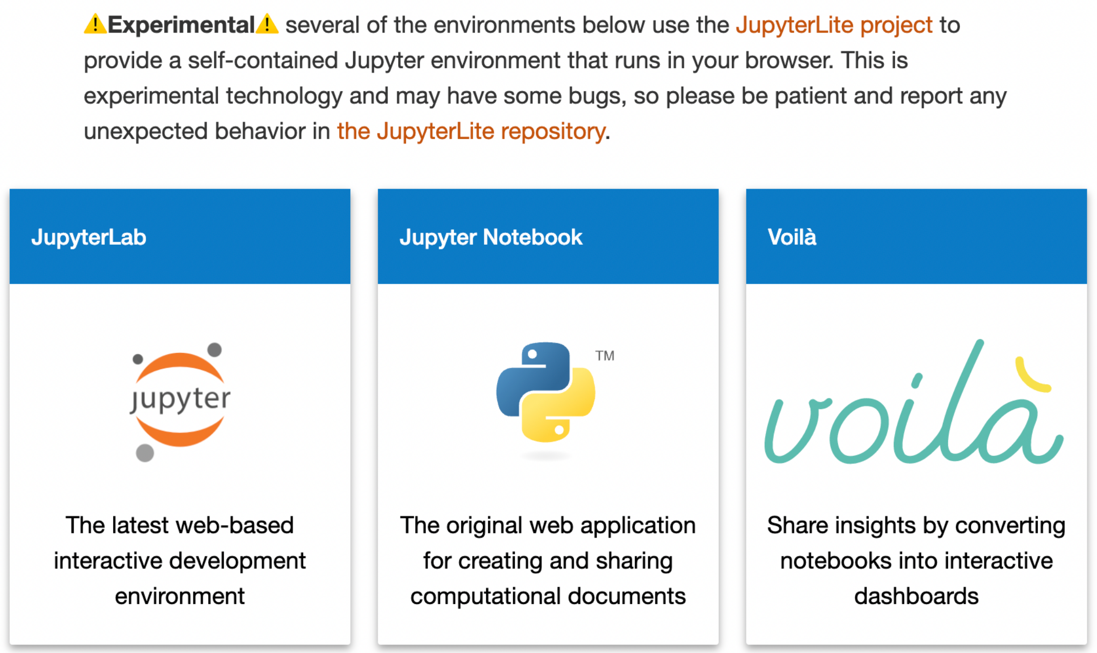
"https://colab.research.google.com/drive/1Y_LiAaN29r3cdBy2L6CMxutylntj3fpX?usp=sharing"

Please make use of comments that are included with codes to understand how the entire notebook works. The following image shows the code section, in which you can modify the maze (size and property), start and end positions for a search, as well as the operation cost.



```
# This section is the user defined initialisation and call of the search function

# Initialisation of maze, where '1' denote a no-go or bloacked location
maze = [[0, 1, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0],
        [0, 1, 0, 1, 0, 0],
        [0, 1, 0, 0, 1, 0],
        [0, 0, 0, 0, 1, 0]]

start = [0, 0] # starting position
end = [4,5] # ending position
cost = 1 # cost per movement

# Create a path solution by calling the 'AStar' function with those parameters specified above
path = AStar(maze,cost,start,end)

# print out the path solution, as a sequence of running numbers (staring from 0) in the matrix presentation of thia maze
print('\n'.join([''.join(["{:" ">3d}".format(item) for item in row])
      for row in path]))
```

If you have a gmail account, you can easily open the Google Colab console, in which you can upload the notebook file (see the previous point) and start working with it right away. If you do not have that account, you might prefer to make use of a free cloud-based one, e.g., "https://jupyter.org/try". On its landing page (see below), click on the "**JupyterLab**" dialogue on the left.



⚠**Experimental**⚠ several of the environments below use the JupyterLite project to provide a self-contained Jupyter environment that runs in your browser. This is experimental technology and may have some bugs, so please be patient and report any unexpected behavior in the JupyterLite repository.

JupyterLab

The latest web-based interactive development environment

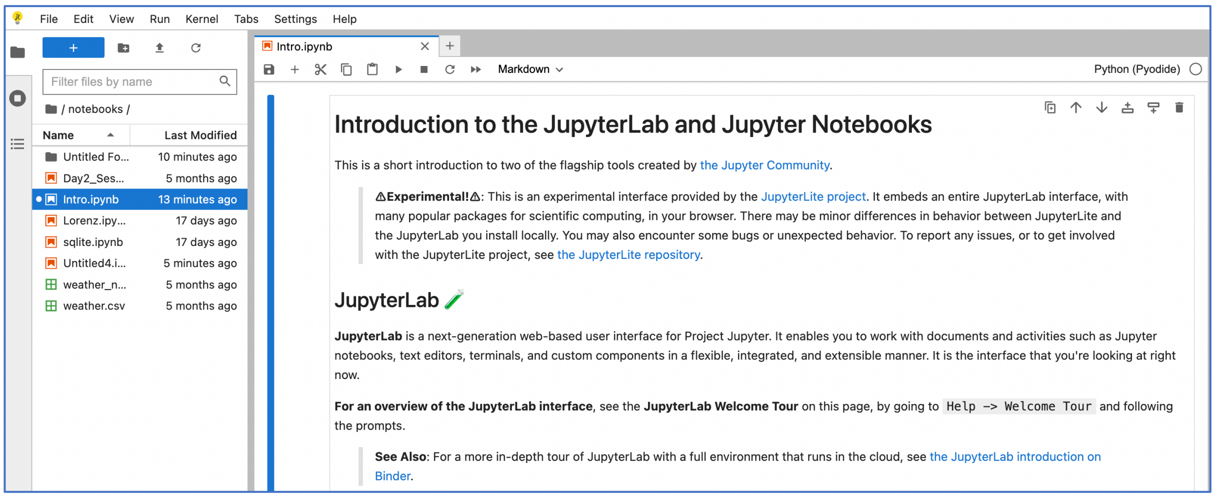
Jupyter Notebook

The original web application for creating and sharing computational documents

Voilà

Share insights by converting notebooks into interactive dashboards

This will bring up the next page, in which you can create a new notebook. Then, you can copy codes from the local file to the online one and run them in sequence (from top to bottom, by code section) to see the result.



Name	Last Modified
Untitled Fo...	10 minutes ago
Day2_Ses...	5 months ago
Intro.ipynb	13 minutes ago
Lorenz.ipy...	17 days ago
sqlite.ipynb	17 days ago
Untitled4.i...	5 minutes ago
weather_n...	5 months ago
weather.csv	5 months ago

Introduction to the JupyterLab and Jupyter Notebooks

This is a short introduction to two of the flagship tools created by the Jupyter Community.

> ⚠**Experimental!**⚠: This is an experimental interface provided by the JupyterLite project. It embeds an entire JupyterLab interface, with many popular packages for scientific computing, in your browser. There may be minor differences in behavior between JupyterLite and the JupyterLab you install locally. You may also encounter some bugs or unexpected behavior. To report any issues, or to get involved with the JupyterLite project, see the JupyterLite repository.

JupyterLab 🧪

JupyterLab is a next-generation web-based user interface for Project Jupyter. It enables you to work with documents and activities such as Jupyter notebooks, text editors, terminals, and custom components in a flexible, integrated, and extensible manner. It is the interface that you're looking at right now.

For an overview of the JupyterLab interface, see the **JupyterLab Welcome Tour** on this page, by going to `Help -> Welcome Tour` and following the prompts.

> **See Also**: For a more in-depth tour of JupyterLab with a full environment that runs in the cloud, see the JupyterLab introduction on Binder.

3 Exercises

Try the following questions and note down your findings. Does it match your expectation? If not, think a bit further, why?

Question 3.1 With the original problem setting, what is the actual cost of a path found by the A* search technique? Is that optimal for the problem?

Question 3.2 Change start and end locations to (1, 5) and (4, 2). Then, what is the actual cost of a path found by the A* search technique? Is that optimal for the problem?

Question 3.3 Change locations of no-go cells to those shown below. With original start and end locations, (0, 0) and (4, 5), can A* find the optimal path? What is it?

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)

Question 3.4 Change maze size and locations of no-go cells to those shown below. With original start and end locations, (0, 0) and (4, 5), can A* find the optimal path? What is it?

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)

Question 3.5 Any idea how to turn this implementation to perform like a uniform cost search instead, please explain?

4 Practical session and contact

We will have this practical on Friday, 21st October 2022, 4-5 pm. @MP3.03. If you can not participate, please email me (tob45@aber.ac.uk) if you need help.