Griffith School of Information Technology
Griffith University

3806ICT – Robotics, Agents and Reasoning
Trimester 1, 2021

Grid World Maze Runner

Ryan Williamson , s5135470
Connor Forbes , ss5068337

We, by signing this page, declare that the work presented in this report is all work done by us, unless appropriate reference has been made to the work of others. I acknowledge that should this not be the case the report will receive zero marks and due action may be taken.

Submitted on: ____________________

Mark Received: ____________________

TABLE OF CONTENTS

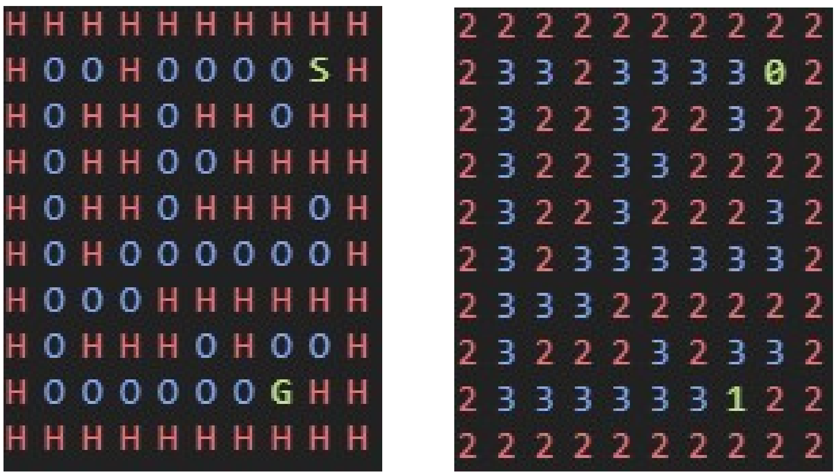
1 Model the Grid World in PAT

1.1 World Representation

Before the Grid World can be modelled with PAT each point needs to be parameterised as an integer as global constants can only represent boolean or integer values.This conversion is only required for PAT's internal computation and for readability convention will still be maintained. Each point within the grid world as been converted as per the following:

- **'O' for open space -** Modelled as a the integer 3
- **'H' for a wall -** Modelled as the integer 2
- **'G' for the goal location -** Modelled as the integer 1
- **'S' for the starting location -** Modelled as the integer 0

An example 10x10 grid world has been generated and can be seen below along with the PAT's internal representation of the world.



(a) Readable Grid World Representation

(b) Internal PAT Representation

Figure 1: 5x5 Grid World Representation

1.2 Game Choices

Four guarded choices are available at each game state that can be taken to move the current position around the maze, the guarded process serves to prevent any illegitimate states from being reached where the position is outside the bounds of the maze.

- **Move Up -** Move the current location up one row
- **Move Down -** Move the current location down one row
- **Move Left -** Move the current location left one column
- **Move Right -** Move the current location right one column

1.3 Assertion

With the Game choices and actions modelled, an assertion can be made that the goal state is reachable for the current board. The goal state can be defined as

$$Goal = Board[Row][Col] == G \tag{1}$$

Where Row and Col are the current position, meaning that the state has traversed through the maze from the starting point to the end location.

1.4 Model Verification Testing

Using PAT's inbuilt model verification tool and the Game Choice, World Representation and Assertion defined the model checker can run to determine if the current maze is indeed solvable and the resultant optimal path. The below optimal path has been given for the grid world defined within Figure 1, the time taken to calculate the path was 0.0004949s

$$go\_left- > go\_left- > go\_left- > go\_left- > go\_down- > go\_down- > go\_down- > go\_down- > go\_left- > go\_down- > go\_left- > go\_left- > go\_down- > go\_down- > go\_right- > go\_right- > go\_right- > go\_right- > go\_right- > go\_right$$

2 Model the Grid World Maze for RL

A reinforcement learning model for a Grid World Maze Runner has been developed within python, seperated into two seperate stages the

3 Parameterise the Grid World

4 Translate the Grid World to a PAT Model and RL Model

Slight modifcations are required to convert the generated maze world from the generated output to eligible PAT and RL models. Extending on the end of the maze generation program two text files are created within the Mazes subdirectory, first the PAT model is written in the following format:

- Define the starting X position
- Define the starting Y position
- Define the width of the grid world
- Define the height of the grid world
- Define the generated maze as a 1D array

These items are exported to a .csp file that can be then imported to the PAT model checker for verification, these parameters must also be removed from the orginal model to prevent overwriting the variables. Following from this the RL model is then written to a .txt file, the RL model only requires the generated maze world as a one dimensional array with no other defining variables. No modification is requried as the initial model as RL was designed to read the flattened maze array from a text file.

5 PAT vs RL for Maze Runner

An experiment has been conducted to compare PAT model verification vs Reinforcement Learning in the domain of the maze runner. The experiment will be conducted fairly by testing the models on randomly generated mazes of varying size, starting from 5x5 up to 200x200. Identical grids will be used for comparison with the times for each model to run. The following parameters will be used for comparison:

- **Time Taken -** How long for the model to determine the optimal path
- **Length of Path -** How long the optimal path is

In order to form a fair comparison the training time required for reinforcement learning will also be measured and factored in as it is a requirement prior to the model running. As reinforcement learning's performance can also be heavily modifed by the identified tuning parameters these will also remain constant throughout experimentation, the same values identified within Section 2 will be used. The experiment will be completed across seven different generated mazes, beginning with smaller mazes and then spiradically increasing to get a broad overview how each model works as the complexity significantly increases. The table below shows the collected results from the experimentation:

Maze Size	Time Taken (s)	Optimal Path Length	Training Time (s)
5x5	0.0000	6	0.21
10x10	0.0010	14	0.52
20x20	0.0010	33	2.19
40x40	0.0010	76	8.51
60x60	0.0190	Did not complete	23.06
100x100	0.0190	Did not complete	22.94
200x200	0.0230	Did not complete	23.24

Table 1: Reinforcement Learning Results

Maze Size	Time Taken (s)	Optimal Path Length
5x5	0.0098863	6
10x10	0.0007578	14
20x20	0.0049444	33
40x40	0.0500344	76
60x60	0.1773686	177
100x100	0.7564199	228
200x200	8.874657	476

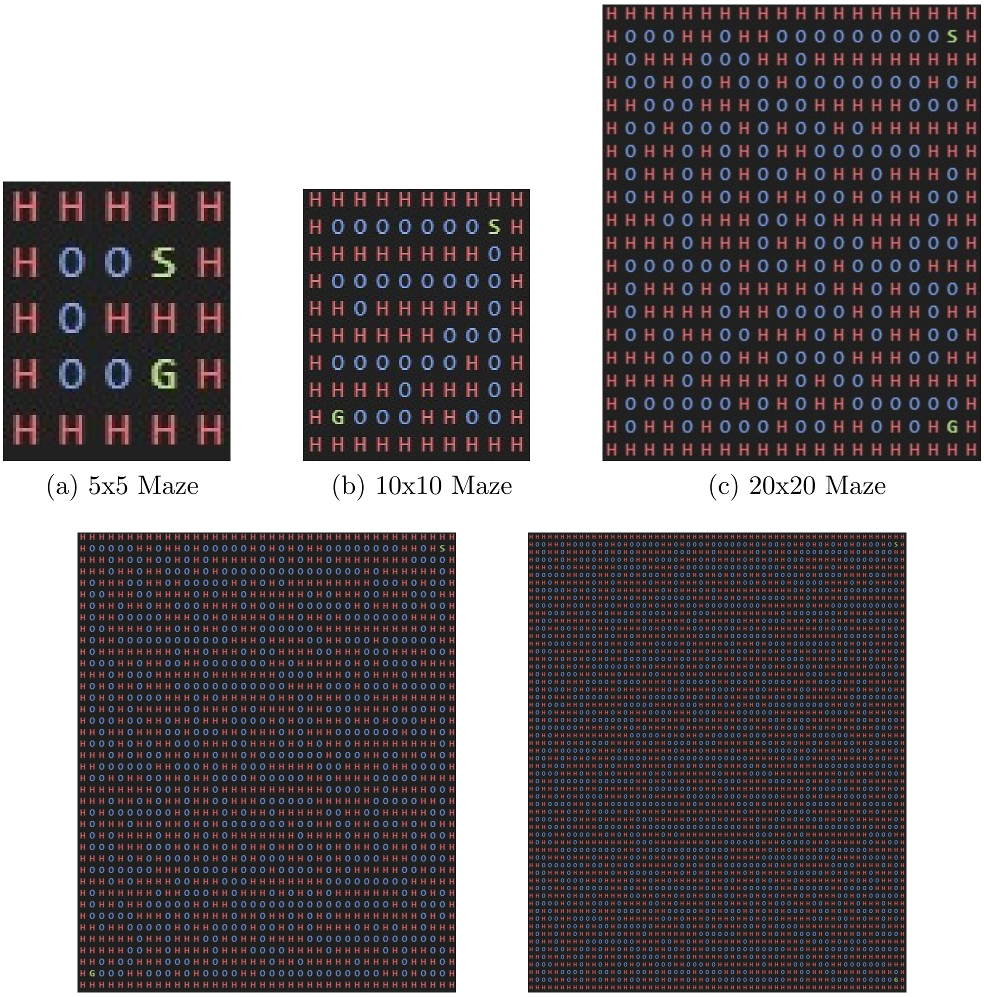
Table 2: PAT Verfication Results

As can be seen, the PAT model verification performs significantly better in all aspects of testing. Without modfiication to the tuning parameters for larger grid sizes reinforcement learning begins to struggle immensely with exploring enough of the state space to reach the goal location and properly populate enough Q values. With the drastic increases in training time as the grid sizes increases reinforcement learning becomes extremely undesirable for model verification.

6 REFERENCES

[1] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm," *IEEE Transactions on Biomedical Engineering*, vol. BME-32, pp. 230–236, March 1985.

Appendix A: Tested Maze Visualisations



(a) 5x5 Maze (b) 10x10 Maze (c) 20x20 Maze

(d) 40x40 Maze (e) 60x60 Maze

Figure 2: Tested Maze's Visualised