STUDENT ID: 29644364

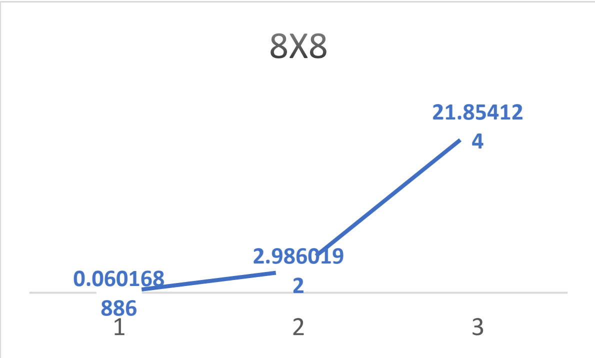
STUDENT NAME: HA TRI NHAN LUONG

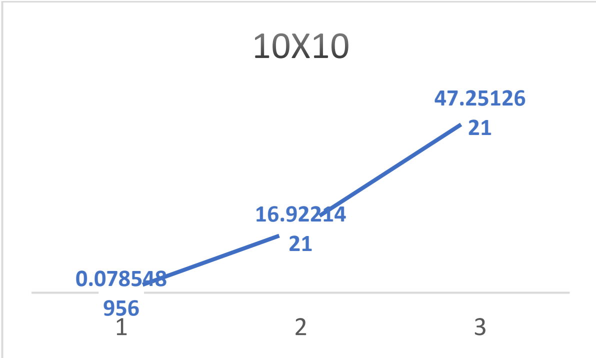
FIT3080 – Assignment 1 Answer Sheet

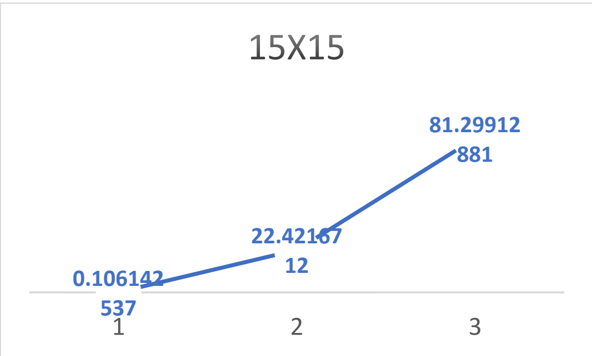
Question 2:

Performance graph: Data taken from 25 random seeds for each case size and number of agents.

	5x5	6x6	8x8	10x10	15x15
1	0.031171	0.043893	0.060169	0.078549	0.106143
2	0.63984	1.03819	2.986019	16.92214	22.42167
3	16.42552	19.21256	21.85412	47.25126	81.29913







We can see that the performance somewhat grows exponentially with number of agents.

The problem seems to be increasing linearly with board size for 1 agent, but exponentially for more agents (see ***Extra plots***).

There are cases where the runtime has already been extremely slow (or running out of memory before time limit) for 2 trains thus this implementation should not be used in the real-word at all.

These extremely slow cases are usually when a train needed to take a turn very early on but kept on going straight due to the precedence of move actions, thus it takes a very long time to back track to when it needed to take a turn.

Question 3:

Paper: **Fast and Memory-Efficient Multi-Agent Pathfinding**
Author: *Ko-Hsin Cindy Wang and Adi Botea*
PDF: https://www.aaai.org/Papers/ICAPS/2008/ICAPS08-047.pdf

In this paper, the author proposed an improved version of A* pathfinding. Unlike our implementation, this version runs individually for each agent (thus avoids exponential growth at each expansion), with some extensions. The first extension is that it creates an alternating one-way direction for each row and column, see the following figure:

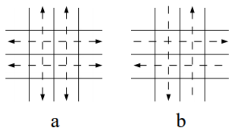


Figure 2: A classical map at the left and an annotated map at the right. For clarity, diagonal moves are not shown.

This avoids direct head-to-head collisions. Furthermore, it adds simple local fixing rules to non-head-to-head collisions instead of expansive replanning (such as: for side-on collisions, add rules to give priority to vertical first, horizontal after). When a deadlock is occurred, it simply forces one agent to escape it's intended path to break the lock, then continue the journey.

From the result given in the paper:

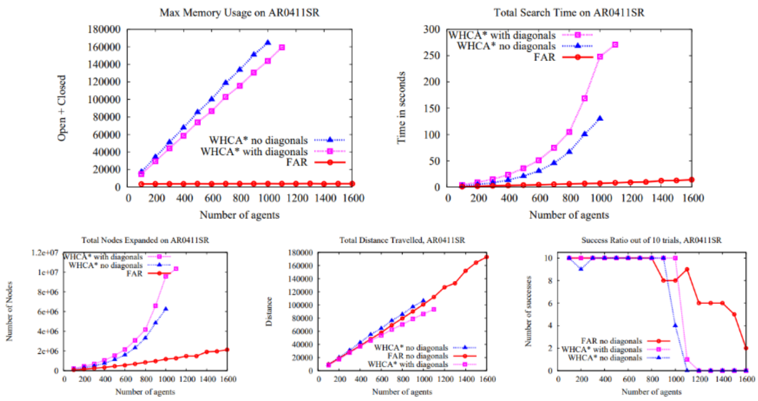


Figure 10: Results from experiments on BALDUR'S GATE map AR0411SR. Note that, for readability, the legend within each graph corresponds to the ordering of the graph lines.

This implementation scales well with the number of agents (works well with even 1000+ agents) compared to our implementation due to the non-expensive collision solving.

Extra plots:

(I was assuming the plots does not count towards to page length limit)