Intro to AI

Anusha Sheelavant

Project 1:

I have programmed Bot 1, 2 and 3. Bot 4 is not programmed.

Bot 1 works on A\* approach by using Manhattan distance as its heuristic.

Bot 2 also uses A\*, but it treats cells that are currently on fire as blocked and will not consider them as part of the path to the button. This helps avoid cells that are on fire. Note that I run A\* after every move the bot makes because I assume the bot does not have access to information about cells that are on fire unless they are neighboring cells. Therefore, A\* is run every time the bot moves to determine the next best cell to move to.

Bot 3 works similarly to Bot 2 but it goes one step further by simulating the spread of fire to adjacent cells and avoiding those in the shortest path calculation. If no safe path is available, it defaults to using Bot 2's logic.

2) For each bot, repeatedly generate test environments and evaluate the performance of the bot, for many values

of q between 0 and 1. Graph the probability (or average frequency, rather) of the bot successfully putting out

the fire. Be sure to repeat the experiment enough times to get accurate results for each bot, and each tested

value of q.

For this question, I had two datasets. One where I generate success rate in this manner:

for q in float\_range(0.1, 1.1, 0.1):

for d in range(2, 60):

for i in range(0, 3):

subprocess.run(["python", "main.py", str(d), str(q)])

This approach generates random ship layouts with random coordinates for the fire, button, and bot. I thought averaging these results would provide reliable data, but it seems to be skewed when the dimensions are small or when the bot or button is placed close to the fire

|  |  |  |  |
| --- | --- | --- | --- |
| **q** | **bot1** | **bot2** | **bot3** |
| 0.1 | 88.42365 | 91.37931 | 91.133 |
| 0.2 | 79.31034 | 80.04926 | 81.7734 |
| 0.3 | 74.38424 | 71.92118 | 73.89163 |
| 0.4 | 61.08374 | 61.82266 | 63.5468 |
| 0.5 | 51.97044 | 51.47783 | 52.46305 |
| 0.6 | 41.87192 | 42.11823 | 42.61084 |
| 0.7 | 36.2069 | 35.22167 | 35.71429 |
| 0.8 | 33.74384 | 33.25123 | 34.48276 |
| 0.9 | 28.57143 | 27.83251 | 28.32512 |
| 1 | 22.41379 | 22.41379 | 22.41379 |

A graph with colored lines

Description automatically generated

For better comparison of the bots, **I fixed the ship layout and the coordinates to verify performance of the bots**.

ship\_map = [

    [1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1],

    [1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],

    [1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1],

    [0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1],

    [1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1],

    [1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1],

    [1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1],

    [1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1],

    [1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1],

    [1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1],

    [1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

    [0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0],

    [1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1],

    [1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1],

    [0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1],

    [1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1],

    [1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1],

    [1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1],

    [1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1],

    [1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1],

    [0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0]

]

fire\_coord = (0, 11)

bot\_coord = (9, 0 )

button\_coord = (17, 17)

Grid size = 21x21

|  |  |  |  |
| --- | --- | --- | --- |
| **q** | **bot1** | **bot2** | **bot3** |
| 0.1 | 1 | 1 | 1 |
| 0.2 | 1 | 1 | 1 |
| 0.3 | 1 | 1 | 1 |
| 0.4 | 1 | 1 | 1 |
| 0.5 | 1 | 1 | 1 |
| 0.6 | 0.986667 | 0.966667 | 1 |
| 0.7 | 0.886667 | 0.873333 | 0.86 |
| 0.8 | 0.42 | 0.5 | 0.526667 |
| 0.9 | 0.086667 | 0.073333 | 0.126667 |
| 1 | 0 | 0 | 0 |

A graph showing the growth of success rate

Description automatically generated

3) When bots fail or get trapped by the fire, why do they fail? Was there a better decision they could have made

that would have saved them? Why or why not? Support your conclusions.

When bots fail or get trapped by the fire, for random layouts and grid sizes:

1. Fewer or no alternate paths
2. Alternate paths are on fire
3. Button close to fire
4. Bot close to fire

1, 3 and 4 were avoided by using fixed coordinates.

1. q, i.e., flammability of the ship is so high that fire spreads quickly and the bot is at a point where it cannot take alternate paths to the button. Or, the bot is on fire, or the button is on fire

4) Speculate on how you might construct the ideal bot. What information would it use, what information would

it compute, and how?

The way I would construct an ideal bot for a given ship layout is:

* **During Initialization (Before the Fire Starts)**:
  + The bot can store in memory all the possible paths from one cell to every other cell. This will help it memorize the ship’s layout and identify blocked cells.
* **During Operation (When the Fire is Present)**:
  + The bot should have a sense of where the fire is located so it can take alternative paths earlier rather than later. This will help avoid situations where it is trapped in a cell with only one path to the button, and that path could be compromised by the fire.