Patryk Dziedzic and Mathew Umano

Dr. Cowan

CS440 - Introduction to Artificial Intelligence

13 October 2023

Project 1 Writeup - Data and Analysis

1. Explain the design and algorithm for your Bot 4, being as specific as possible as to what your bot is actually doing. How does your bot factor in the available information to make more informed decisions about what to do next?

- At every time step, the bot re-plans the *optimal* path to the button, similarly to bots 2 and 3. This *optimal* path is calculated with the A* algorithm. The algorithm uses a priority queue of PQCells, which contain the cell itself with all its properties as well as a field for the heuristic value. The heuristic is calculated for each neighbor of the current cell the bot is at by running simulations on the neighbors and returning the number of wins. The simulation recycles the bot 1 logic and performs the simulation on a copied ship with the "bot" being the neighbor cell. The simulation runs until the temporary bot either makes it to the copied button or fails to do so. There are 4 simulations run for each neighbor. Therefore, at each time step the bot will choose which neighbor to go to based on the number of wins each neighbor returned. This is modeled after Monte Carlo Tree Search.
- The bot takes into account the current state of the ship with all the cells on fire at each time step. It takes advantage of this knowledge by running simulations on its neighbors to see what neighbors are more likely to lead the bot to finding the button and winning.

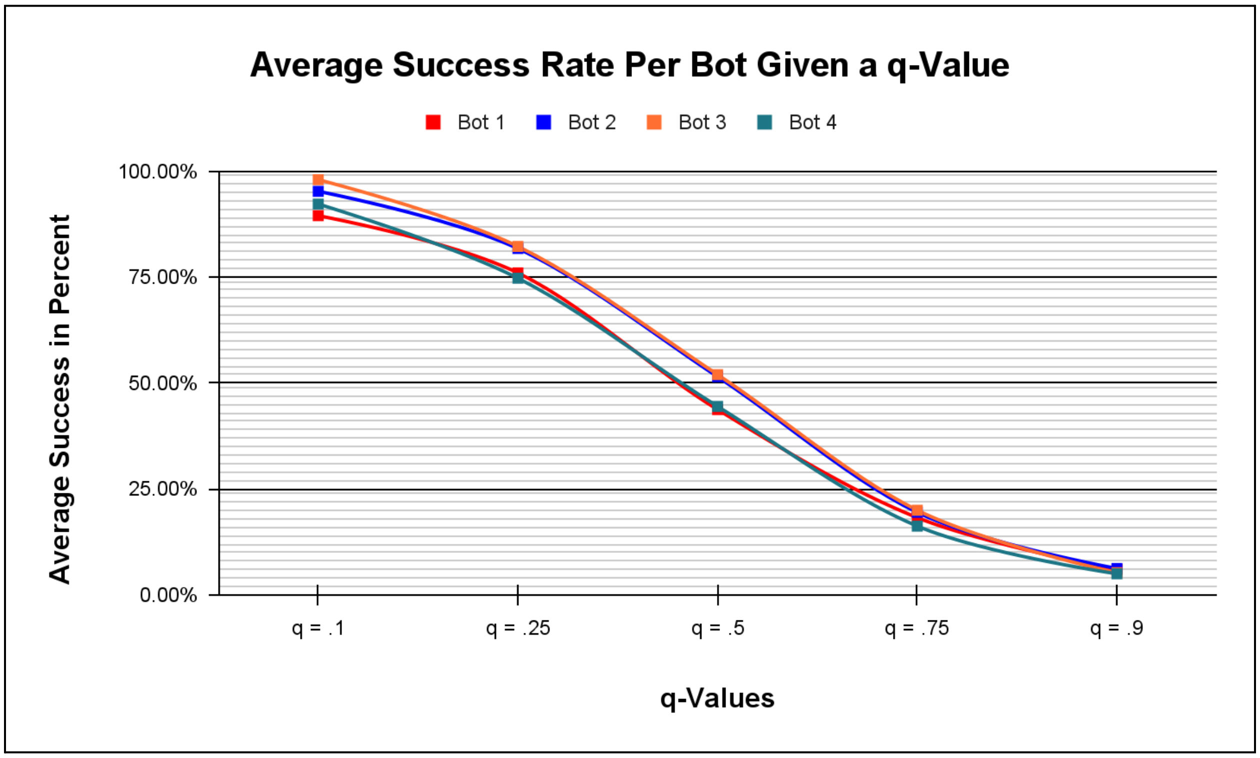
2. For each bot, repeatedly generate test environments and evaluate the performance of the bot, for many values of *q* between 0 and 1. Graph the probability (or average frequency, rather) of the bot successfully putting out the fire. Be sure to repeat the experiment enough times to get accurate results for each bot, and each tested value of *q*.

- **Note:** For the purposes of analyzing the efficiency of each bot most accurately between bots 1-4, we decided to omit the following cases from the test results at t = 0:
  - The bot spawns on the button.

- The fire spawns on either the bot or the button.
- The bot is closer to the button than the fire. (In this case, even if the spread of the fire is q = 1, the bot will outrun the fire by taking the shortest/optimal path.)

- Bots 1-3 were run on a 100x100 ship with 400 test cases for each q.
- Bot 4 was run on a 50x50 ship instead (still with 400 tests for each q) due to taking a very long time for each test for a larger size ship. Bot 4 in total took ≈30 hours to complete.

	Bot 1	Bot 2	Bot 3	Bot 4
q = 0.1	89.50	95.25	98.00	92.25
q = 0.25	76.00	81.75	82.25	74.75
q = 0.5	43.75	51.50	52.00	44.50
q = 0.75	18.25	19.50	20.00	16.25
q = 0.9	6.00	6.25	5.25	5.00



- Bots 1-3
  - The results that were produced by our code for bots 1-3 were consistent with what was expected via different implementations. The success rate for each q value

gradually increased for each bot with the exception of Bot 3's q = 0.9. This may be attributed to the fact that Bot 3 does not take as many risks in that it may prefer a more cautious path away from the "ideal, shortest path." This could put Bot 3 in an unfortunate situation where it ends up cornering itself or never proceeding down the only viable path to the button that isn't caught on fire yet but may. If that's the case, the bot would not reach the button, whereas, Bot 1 and 2 might have proceeded down a path to the button that was safe enough to pass through, yet a riskier route as it could have placed them closer to the fire.

- Bot 4
    - Generally, Bot 4 behaves similarly to Bot 1. This is understandable because the logic to run Bot 1 was used for the simulations in Bot 4. With only 4 simulations being run for each neighbor, the bot was not able to get as accurate results as it could have. This is why it performs worse than Bot 1 at times. Ideally, if more time was given to compute, Bot 4 could have a larger number of simulations per neighbor. This would make the utility values for the neighbors more accurate, which would improve its success rates as Bot 4's decisions would be even more informed. We could also have used Bot 3's logic in place of Bot 1's logic for each simulation given more time, increasing the success rates even further.

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, it can happen when the bot is unsure about which path it should advance with, therefore resulting in an "indecisive" bot that moves back and forth between two cells with the same utility value, or rather, a bot that does not make a "smart" decision that puts them closer to the button. As the bot steps between the two cells, the fire spreads and moves closer to both the bot and the button. The bot would not move until the fire either got too close to force a move or the fire reached the button. The bot needs to make a decision prior to the fire reaching the bot. Furthermore, the bot should be progressively getting closer to the button and not step back and forth between two cells. It is possible that two adjacent cells return the same amount of wins every time the bot runs its simulation, therefore, the bot does not have enough information to make a decision as to how to proceed. To combat this, a priority queue was implemented, allowing the bot the ability to make decisions when faced with cells with equal chances of success.

- In some scenarios, the bot fails because it is purely unlucky. If the fire happens to spawn in a hallway/corridor that is the only route to the button, the bot loses because of chance and there is nothing the bot could have done differently to alter the result of the run. The fire could also spawn on the bot or button at the zeroth time step which is also unlucky and impossible to avoid.
- An additional circumstance is if the fire spreads to a cell that the bot had planned on taking. The bot reroutes and the fire spreads to a cell that the new route contains. This rerouting and spreading of fire could happen many times until the fire spreads to all cells that now block all routes for the bot to take, thus, resulting in the bot or button eventually catching on fire. One scenario that was avoided was the bot entering a corridor or hallway that is a deadend. The bot could potentially trap itself by entering a hallway with a deadend and the fire happens to spread to the cell at the entrance. If that happens, the bot will get trapped and die after a few steps.

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

- Assuming we had infinite time to execute the code, it is possible to create an ideal bot that works similar to our Bot 4, but performs more simulated searches at each neighbor. Bot 4 currently only does 4 simulations on each neighbor at each step that the bot takes, for a maximum of 16 simulations per step, allowing the bot to make the most informed and calculated decisions while still running in a feasible amount of time. An ideal bot would run a very high number of simulations on each neighbor before each step so that it could make the most informed decision based on the most accurate data. This would take an unreasonable amount of time to execute, but since we're in a timeless environment let's say each step takes a time unit of 1 to calculate and move; the time it would take to execute the code from start to finish would never take a time of more than DxD time where D is the size of the columns of the ship. After every step, the bot would then run simulations on the neighbors of the new cell to then predict the success rates of moving to another tile closer to the button while avoiding the spreading fire, which then brings us to another thought.

- This ideal bot could also work similarly to Bot 4 but implement Bot 3's logic of BFS in every simulation that is run at each neighbor instead. Currently Bot 4 implements Bot 1's logic for every simulation at each neighbor as a way of reducing the runtime of each simulation. Without the restriction of time, implementing Bot 3 logic at every simulation could prove to be effective for maximizing the success rate for the ideal bot.
- Furthermore, an ideal bot could be "more careful" and put a wider buffer on the fire cells to avoid (i.e. avoiding the neighbors of the neighbors of the fire cells). However, the bot would have to account for the possibility of not being able to find a route that meets this criteria. In that case, the bot can choose a way to proceed based off of the same logic bot 4 or bot 3 uses for its searches and simulations.
- An ideal bot would also be able to accurately predict where the fire could spread, even from a few steps out, which could be calculated or assumed during the simulation stages before each step. This way, the bot can keep track of which cells may catch on fire in the next few steps, thus the bot can avoid these cells as it travels to the button. Perhaps the use of a data structure that keeps track of this would be beneficial for implementing this logic. With the inkling of where the fire could spread over the next few steps, the bot can avoid that entire general area, given that the button is not located in that area of the ship. Though it would take a lot of memory on a computer and time to fully execute all of this, assuming that memory and time is not an issue, the outline above could be an exemplary model of an ideal bot for the ship.
- It should also be noted that even an ideal bot may result in loss situations as the fire could just happen to spawn closer to the button, reducing the chance of the bot making it to the button immensely. That is, even a perfect or ideal bot might not have 100% success rates for low values of q, or comparatively higher success rates for higher values of q, though it can be expected that this bot would perform significantly better than bots 1-4.

***Note: Both Patryk and Mathew worked very closely together on each aspect of the project so it is hard to say who contributed what specifically.**