Fire Extinguisher Lab Report

Jeeva Ramasamy
Rutgers University
jeeva.ramasamy@rutgers.edu
Course: 16:198:520
Instructor: Professor Cowan

Abstract

In this project, we explore the design and evaluation of various bot strategies for navigating the fire-prone environment of the deep space vessel *Archaeopteryx*. As the sole guardian of the ship during crew hibernation, the bot is tasked with locating and pressing a button to activate the fire suppression system while avoiding fire hazards. The ship's layout is represented as a 40 x 40 square grid, generated through a series of iterative processes that create open paths and dead ends, simulating the complex interior of the vessel. Four distinct bots are implemented, each utilizing different strategies for pathfinding in the presence of a spreading fire. Bot 1 plans a static shortest path to the button, ignoring the spread of fire; Bot 2 recalculates the path at every time step based on current fire locations; Bot 3 adapts its route to avoid both fire and adjacent cells. Bot 4 utilizes a different approach that integrates exploration and exploitation by taking into consideration both the shortest path to the button and the risk of fire, enabling more informed decision-making. The performance of each bot is evaluated across multiple simulations at varying flammability parameters, q, to assess their effectiveness. Results indicate that Bot 4 outperforms the other bots on average, achieving a higher success rate while navigating complex scenarios.

1 Introduction

The bot operates in a simulated 40 x 40 grid environment, designed to replicate the complex layout of the deep space vessel *Archaeopteryx*. This grid is dynamically generated with a mix of open paths and blocked cells, creating a complex navigation challenge. At the start of each simulation, a fire ignites in a random open cell, adding a critical time-sensitive component as the bot must quickly navigate through the ship to reach a designated button that triggers the fire suppression system. At the same time, the bot must avoid the spreading flames, which escalate the urgency and complexity of its task.

This project explores four distinct bot strategies, each employing unique algorithms and decision-making methodologies to solve the pathfinding problem in the presence of fire. The bots range from basic approaches that disregard the fire's dynamics to more advanced strategies that continuously adjust their routes based on the fire's spread and potential threats. Through systematic evaluation across a wide range of flammability parameters, we can determine which strategy proves most effective at navigating and surviving in this hazardous, fire-prone environment.

2 Methodology

This section outlines the processes for creating the simulated environment, defining bot behaviors, and evaluating their performance in navigating toward the button while avoiding the spreading fire.

2.1 Environment Simulation

The first step in setting up the environment involves generating a 40×40 grid to represent the ship's layout. The process for populating this grid with blocked and open cells is designed to create a navigable maze-like environment for the bots, while maintaining some randomness and flexibility in the ship's layout. The steps are as follows:

- **Grid Initialization**: The grid begins as a completely blocked space, where all cells are marked as closed. A random starting cell is selected and opened to initiate the process of creating pathways.
- **Opening Cells**: The grid is progressively populated with open cells. Each new open cell is chosen based on having exactly one neighboring cell that is already open. This constraint ensures that the paths formed are narrow and interconnected, resulting in a maze-like structure. The process continues until no additional cells can be opened using this rule, preventing the creation of isolated open cells and ensuring connectivity.
- **Opening Some Dead Ends**: Approximately half of the dead-end cells (cells with only one open neighbor) are randomly opened once the initial maze is generated. This step enhances navigability by reducing bottlenecks and providing more potential routes, preventing overly restrictive pathways that could trap bots.
- **Bot, Fire, and Button Placement**: After the grid layout is established, distinct open cells are randomly selected for the bot's starting position, the fire location, and the button that will activate the fire suppression system. The bot's starting location and the button are always chosen to ensure that a viable path exists between them, verified by Breadth First Search (BFS). If a reachable path to the button cannot be found, the entire grid is reset and recreated to avoid impossible scenarios.

2.2 Bot Design and Behavior

The bots are designed to navigate the grid and respond to the fire threat based on the following strategies:

- **Bot 1**: This bot uses a simple Breadth First Search (BFS) algorithm to compute the shortest possible path from its starting point to the button, ensuring it avoids the initial fire cell. However, it does not account for any future fire spread during its journey, so its strategy is static and vulnerable to dynamic changes in the environment.
- **Bot 2**: Like Bot 1, this bot initially calculates the shortest path to the button, but it goes further by recalculating the path at every time step, taking into account the current spread of the fire. This real-time dynamic replanning allows the bot to adapt and adjust its route as new fire cells emerge, making it more responsive to changes.
- **Bot 3**: This bot adopts a highly cautious strategy, avoiding not only the active fire cells but also any cells adjacent to the fire. It preemptively navigates around potentially dangerous areas to minimize risk. If it finds that no safe path exists, it defaults to planning the shortest route while factoring in only the current fire cells, ensuring a balance between safety and efficiency.
- **Bot 4**: Bot 4 is designed to optimize pathfinding in a fire-spreading environment by using the A* search algorithm. The key challenge in this environment is to avoid the fire and cells that are likely to catch fire soon. Bot 4 differentiates itself from other bots by incorporating dynamic risk assessment in its pathfinding strategy, making use of available information to maximize its chances of success.
    - **A* Search Overview**: A* is a pathfinding algorithm that uses cost-so-far (actual distance from the start node) and a heuristic (estimated distance to the goal) to find the shortest path. For Bot 4, the algorithm evaluates both the distance to the goal and the risk posed by the fire, adjusting its movement to avoid dangerous areas.
    - **Heuristic Function**: Bot 4 uses the **Manhattan distance** as its heuristic, which is a simple measure of the distance between two points on a grid. This gives a base estimate of how far the bot is from its destination. While this may not be as accurate as Breadth First Search (BFS), the Manhattan heuristic significantly speeds up the decision-making process.

      $$Heuristic(cell) = \left|x_{goal} - x_{cell}\right| + \left|y_{goal} - y_{cell}\right|$$

    - **Fire Risk Assessment**: In addition to the standard A* approach, Bot 4 also incorporates a **fire risk cost**, based on the distance from the nearest fire. This risk is used as a multiplier in the bot's decision-making. Cells closer to fire have a higher risk factor, and Bot 4 actively avoids these areas whenever possible. The fire risk is calculated as the inverse of the distance from the nearest fire cell:

      $$Fire\ Risk\ Cost(cell) = \frac{10}{distance\ from\ nearest\ fire\ cell}$$

The farther a cell is from fire, the lower the risk, making that cell more attractive for Bot 4 to traverse. This allows Bot 4 to weigh both the direct path to the goal and the proximity to the fire.

  - **Adaptive Strategy Based on *q* Value**: The parameter q represents the fire-spread rate. Bot 4 adapts its strategy depending on the value of q. For **low *q* values (below 0.6)**, the fire spreads slowly, and Bot 4 actively balances between reaching the goal quickly and avoiding high-risk cells near the fire. The fire spreads rapidly for **high *q* values (above 0.6)**, making it more dangerous to explore further into risky territory. At this point, Bot 4 reduces its exploratory behavior and shifts to a more aggressive pathfinding strategy, mirroring Bot 3 by prioritizing the fastest possible path while avoiding immediate fire and its neighbors.

2.3 Fire Spread Simulation

The fire dynamics are modeled to simulate spreading behavior based on the defined rules:

- **Fire Spread Algorithm**: At each time step, for every open cell that is not on fire, the algorithm calculates the number of adjacent burning cells (K). The probability of the cell catching fire is computed using the formula $P = 1 - (1 - q)^K$, where q is a parameter that defines the flammability of the ship. A random number is generated to determine if the cell catches fire based on this probability.
- **Updating Fire States**: The list of new fires is stored separately until all probabilities are calculated; this list is then used to update the list of fires on the grid. This ensures that fires do not iteratively spread within the same turn due to immediate updates.

2.4 Performance Evaluation

To assess the effectiveness of each bot, the following steps are undertaken:

- **Experimental Design and Repetition**: Each bot is tested in numerous environments with varying values of the flammability parameter q, ranging from 0 to 1 in increments of 0.05. For each value of q, 500 independent simulations were conducted to ensure robust statistical evaluation. The random generation of grid layouts and starting fire positions allowed the bots to be tested for adaptability to diverse conditions.
- **Success Metrics**: The primary measure of success is whether the bot could reach the button before being overtaken by the fire. The bot's ability to navigate the grid, avoid fire hazards, and reach the goal was monitored, and the number of successful outcomes was recorded for each bot in every simulation. In addition to the raw success count, an "impossibility" metric was tracked: instances where the fire spread in such a way that the button was rendered unreachable, regardless of the bot's strategy. This provided an additional layer of analysis to distinguish failures due to bot strategy from scenarios where success was inherently unattainable.

- **Visualization of Results**: The success rates for each bot across different values of *q* were plotted for a visual comparison. Two lines were plotted on each chart: one showing success rates for all simulations and the other focusing on simulations where reaching the button was feasible.

2.5 Identifying Possible Success

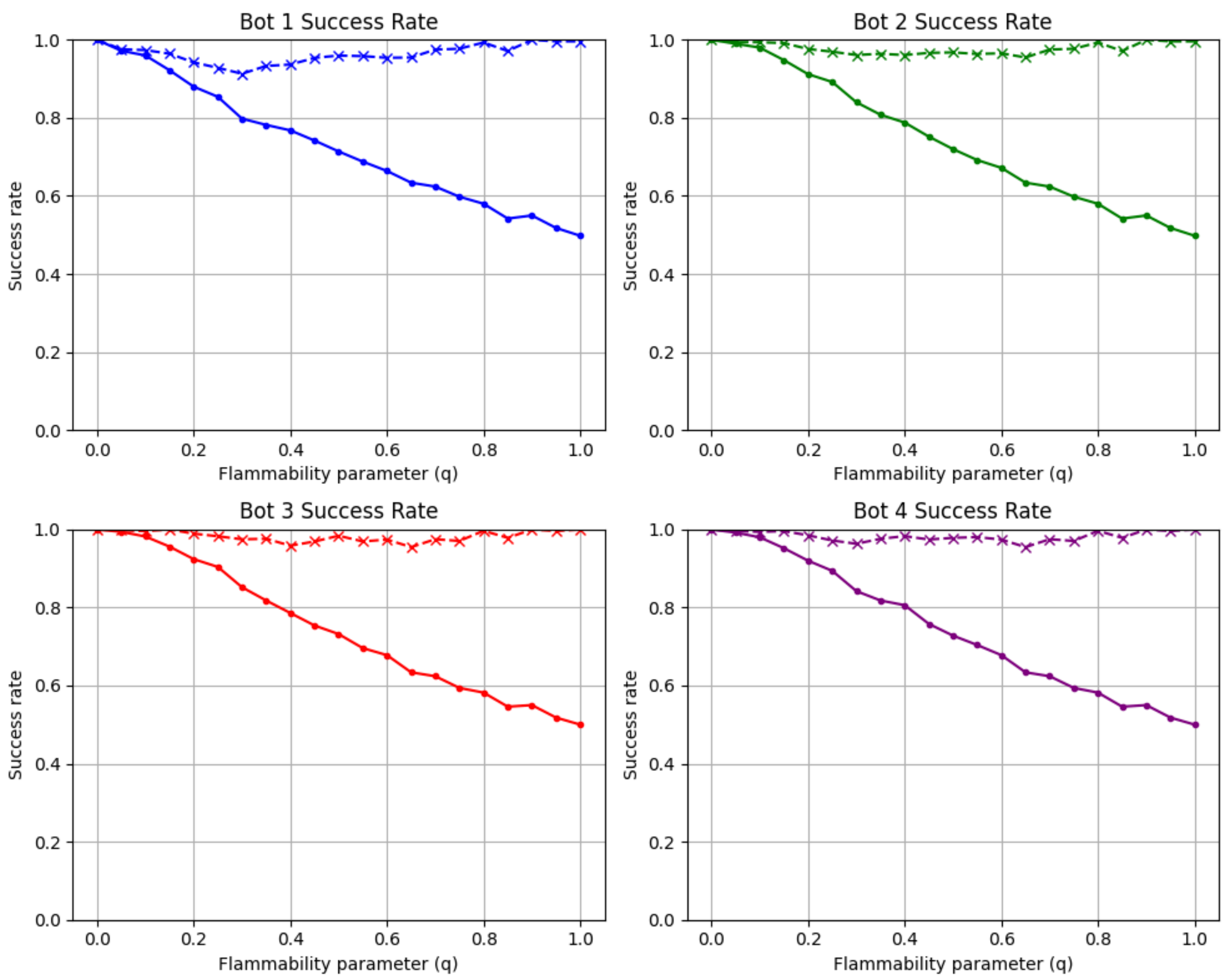
In order to assess the effectiveness of each bot, it is important to determine whether a successful outcome was ever possible in a given simulation. This is accomplished by using Breadth-First Search (BFS) to verify if there was any viable path to reach the button, regardless of the bots' in-game performance. The grid is stored in a list at each time step, capturing the current layout of obstacles, fire, and open spaces. These grids serve as historical snapshots, allowing us to trace the fire's progression and determine whether the bots could have succeeded given optimal movement. When a bot fails to reach the button, these snapshots are analyzed using BFS to simulate whether a path to the button could have existed. The BFS starts from the bot's initial position and explores its surroundings layer by layer. It traverses the grid while incrementing the time step, mimicking how fire spreads and blocks potential paths. This transforms the problem into a BFS search on a 3D grid, where each layer represents the grid at a specific time. For each new time step, the BFS considers the state of the environment, including fire spread and obstacles, and checks whether the button remains accessible. If BFS identifies a valid path, it confirms that the button was reachable, meaning the bots failed due to their strategy rather than an impossible scenario. On the other hand, if no path is found, it confirms that the button was never reachable due to the fire's rapid spread or environmental obstacles. This method not only evaluates each bot's performance in real time but also distinguishes between cases where the failure was due to poor strategy versus scenarios where success was unattainable.

3 Results

This section presents the findings from implementing and evaluating the four bots across various simulated environments on the ship.

3.1 Bot Performance Across Varying *q* Values

The bots were evaluated in numerous simulations while varying the flammability parameter *q* from 0 to 1 in increments of 0.05. The average success rates for each bot were recorded and graphed to visualize the trends. In order to ensure a fair comparison, all bots were tested on the same grid configurations, operating independently so that the spread of fire affected each bot equally. None of the bots could interact with one another. Furthermore, consistent grid generation across varying *q* values was achieved by setting the random number generator's seed to '520,' eliminating discrepancies in starting positions.



The figure above displays the success rates for each bot across the range of q values. The solid lines with dots indicate the actual success rates, while the dashed lines with x's illustrate the success rates excluding impossible cases.

For each q value, 500 simulations were conducted to evaluate the performance and effectiveness of the bots accurately. A clear linear decline in success rates is evident for all bots as q increases. However, Bots 3 and 4 exhibited enhanced performance at lower q values. When analyzing success rates after excluding cases where no viable path existed, a nearly flat trend can be seen, indicating that in nearly all scenarios where success was possible, the bots did not fail, particularly for Bots 3 and 4. For further details on the methodology used to identify possible success, refer to Section 2.5.

3.2 Comparative Analysis of Bot Strategies

A comparative analysis reveals distinct patterns in the bots' performances, especially regarding their responses to increasing flammability:

- **Bot 1** demonstrated high success rates at lower *q* values, relying on pre-planned paths, but struggled significantly as q increased, as it failed to adapt to changing conditions.
- **Bot 2** showed better adaptability by replanning each time step, resulting in a slightly higher success rate than Bot 1 at lower *q* values.
- **Bot 3** exhibited very high success overall. By avoiding cells adjacent to the fire, the bot often picked safer routes than Bot 1 and Bot 2.
- **Bot 4** outperformed, though not significantly, all bots at lower *q* values, demonstrating that its unique strategy effectively combined pathfinding with real-time adaptations to the fire's behavior. However, at higher *q* values, the bot performs equal to Bot 3 due to similar strategic approaches.

Once *q* exceeded 0.7, all bots succeeded in two-thirds or fewer of their trials. High values of *q* create a race to the button, which is almost always determined by which of the two—either the bot or the fire—is closer to the button. The results demonstrate that the bots' effectiveness is significantly influenced by the parameter *q*, with lower values favoring the bots' strategies, while higher values necessitate an advantageous starting position.

4 Discussion

The results of this comparative analysis highlight the complex interplay between bot strategies and the varying degrees of flammability represented by the parameter *q*.

4.1 Adaptability and Strategy

Bot 1, which relied on static pre-planned paths, showed high success rates at lower *q* values but faltered as conditions worsened. This highlights an important limitation: the inability to adapt in real time. In contrast, Bot 2, which altered its path at every timestep, demonstrated improved success rates at lower *q* values. Bot 3's success, which included avoiding adjacent fire cells, highlights the importance of risk assessment in pathfinding. By including a buffer zone around the fire, Bot 3 could select safer routes, demonstrating that predictive strategies can enhance overall performance. Finally, Bot 4's superior performance at most *q* values indicates the effectiveness of integrating both pathfinding and real-time adaptability.

4.2 Challenges with High Flammability

The observation that all bots struggled to succeed when *q* exceeded 0.6 shows a threshold beyond which the fire's spread becomes overwhelming. This emphasizes the importance of the initial positioning, as the outcome often relied on the bot being closer to the button than the fire. In such scenarios, it is not easy to compare the true performance of the bots since the environment plays the biggest role in success.

4.3 Why do Bots Fail?

Bots can fail or become trapped by fire due to a combination of decision-making, environmental constraints, and the unpredictable nature of fire spread.

- **Lack of Adaptability**: Bot 1 relies on pre-planned or static paths that do not adjust to changing conditions, such as the rapidly spreading fire. When the environment shifts, it may find itself in positions where escape routes are blocked.
- **Inadequate Risk Assessment**: Bots may not correctly assess the risk associated with their proximity to fire or obstacles. This can lead to choices that put them in harm's way, such as moving too close to a fire source or ignoring safer but longer routes.

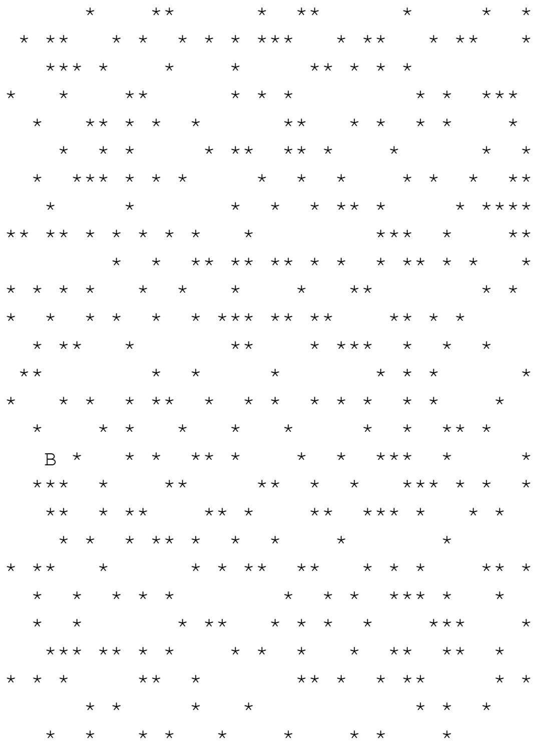
Some methods to improve decision making include:

- **Risk-averse Behavior**: Bots could adopt more conservative strategies by prioritizing exploration over exploitation. For instance, they could choose longer paths that avoid high-risk areas rather than rushing toward the goal, potentially reducing their chances of getting trapped. This approach is represented by Bots 3 and 4, which outperformed the greedy Bot 2 by demonstrating greater caution and adaptability in their decision-making processes.
- **Fire Spread Prediction**: Incorporating predictive modeling of fire spread could enable bots to foresee potential dangers. They can proactively avoid those areas if they can anticipate where the fire will spread. This approach will be explored in greater detail in Section 4.4.
- **Fallback Mechanisms**: Establishing fallback strategies for when a path is blocked can be beneficial. For instance, if a bot encounters an obstacle caused by the spreading fire, having predefined alternate routes will allow it to escape danger quickly. Additionally, picking initial routes that offer multiple fallback options can greatly improve the bot's chances of survival in dynamic environments.

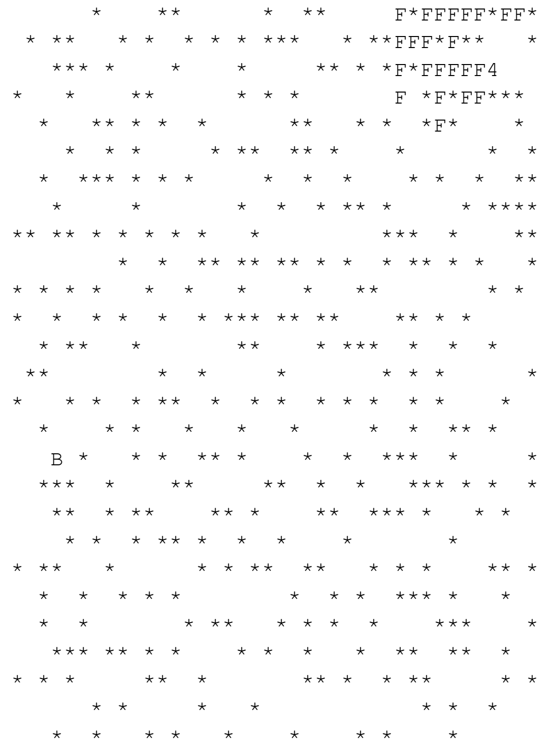
In addition to the line graph highlighting the success rates of Bots 3 and 4, the following scenario illustrates the critical need for improved decision-making:

```
   ****   ***    *    *              *
 *  **     *       *     * * * * *  *
   *   *   *  * *    * *    *  * * *  *
*    * * *   *    **  *  ***     *  * *
  * *  * * *   ** *  *  *    * **  * S
   **       * ***  *  *   *  * * *   **
*   * *   *       ** *** *     * * *   *
 *    ** * * * ** *   *     * *  *  *
    *          **  ** *   **  **  *  *
 *   ****  * *      * *  * **  **     *
   *     *    ** *                **
 *    **    *     * *  *  **  *  F* *
 * *     * *** *     *   * **          *
```

```
   ****   ***    *    *              *
 *  **     *       *     * * * * *  *
   *   *   *  * *    * *    *  * * *  *
*    * * *   *    **  *  ***     *  * *
  * *  * * *   ** *  *  *    * **  * S
   **       * ***  *  *   *  * * *   **
*   * *   *       ** *** *     * * *   *
 *    ** * * * ** *   *     * *  * F*
    *          **  ** *   **  **FF*F *
 *   ****  * *      * *  * **FF**FFF  *
   *     *    ** *          FFFFFF**
 *    **    *     * *  *  **FF*FFF*F*
 * *     * *** *     *   * **FFFFFFFFF *
```

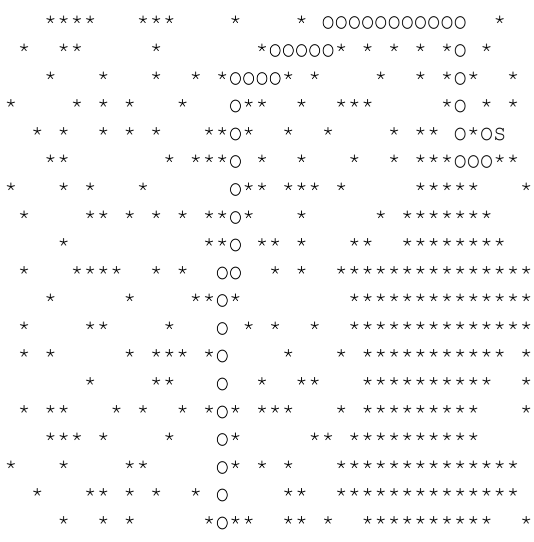


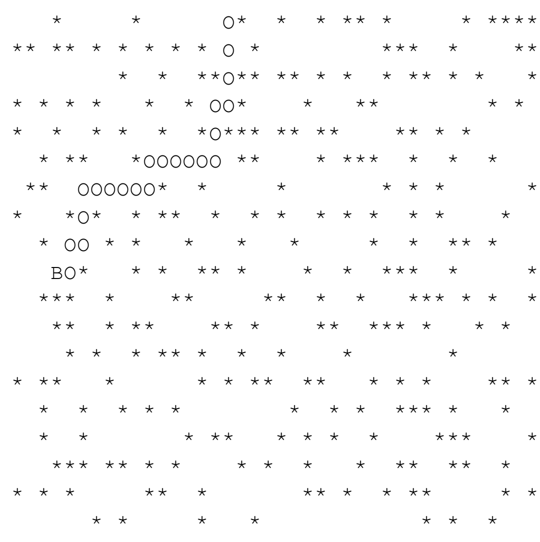
40 x 40 grid showing start of simulation where q = 0.2



All bots (represented by 4) are trapped due to picking paths that became blocked by fire.

The example above shows that the bots initially selected a path that appeared safe. However, their failure to account for potential obstacles and the absence of fallback options ultimately led to their entrapment in this simulation. The path that leads to the goal is outlined below:





The viable path to the goal is marked with O's, showing a route that could have led to success.

By employing better decision-making strategies, such as exploring multiple routes and incorporating fallback options, the bots might have avoided this entrapment and improved their chances of reaching the goal.

4.4 Construction of an Ideal Bot

The ideal bot would be able to stop the fire, given that it is possible based on the environment and the game's progression. This bot would possess several key features:

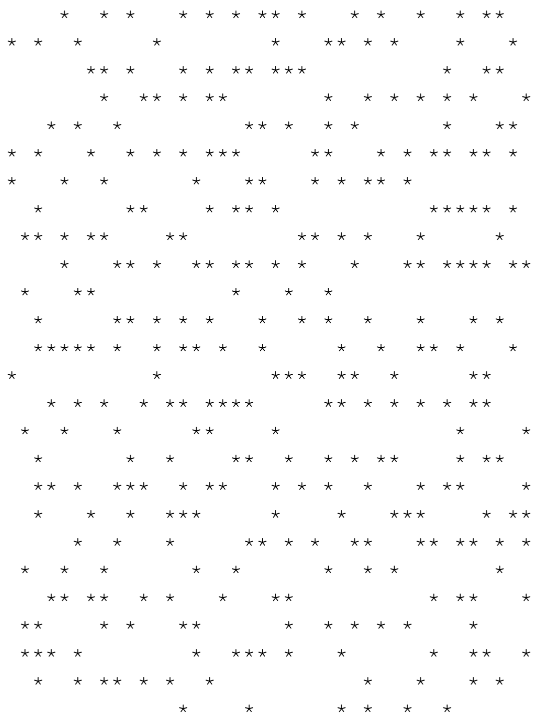
- **Predictive Modeling**: Utilizing predictive modeling techniques, the ideal bot would forecast fire spread based on the grid layout and the flammability parameter q. By anticipating potential danger zones, the bot could avoid high-risk areas and navigate to safer paths.
- **Fallback Strategies**: To ensure continuous operation even in the face of obstacles, the ideal bot would incorporate pre-calculated fallback strategies. These will allow the bot to instantly identify alternative routes if its current path is blocked by fire.
- **Balancing Exploration vs. Exploitation**: The ideal bot would adopt a balanced approach between exploration and exploitation. While it would strive to reach the goal efficiently, it would also prioritize thoroughly exploring the environment to identify safe paths and resources. This would involve assessing different routes' potential risks and rewards before committing to a path. Although Bot 4 attempts to achieve this capability, it makes several mistakes.

The implementation of Predictive Modeling and the Fallback Strategies would require a considerable amount of time and resources. Effective predictive modeling relies on extensive datasets, which in this case would include numerous instances of fire spread outcomes. As a result, if these calculations were conducted at every decision point, it could lead to slower decision-making. By incorporating these features, the bot would be better equipped to navigate the challenges posed by fire spread and achieve its objective of stopping the fire when feasible. However, they may impede the bot's timely judgment if used in real-time scenarios.

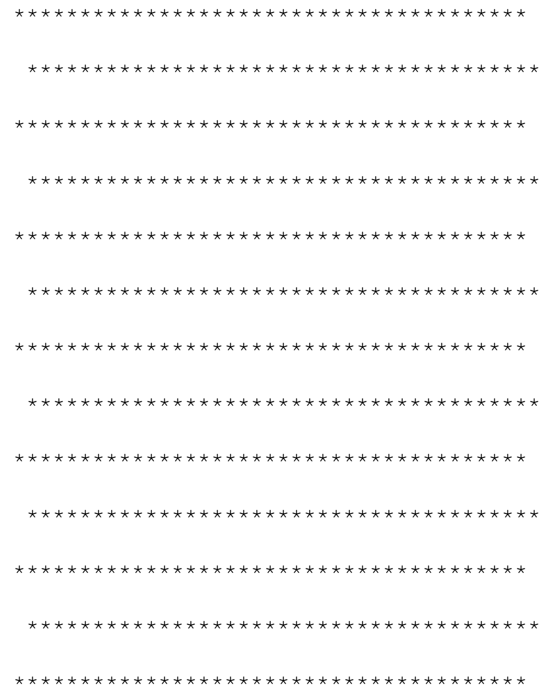
4.5 Optimizing Ship Layout

```
  *   ** **  *      *    * *      * * *
 *  *       *  * *           ** * * *
   * *** **   *   * * *** * *   * * * **
 * * *  ****          *   **** *
   *  *  **  *     **   *   *     * *
**    * * * **** * ***    **   * *   *
  **    *    *   *  * ***  * **    ****
 *   **   ** * *   ** *   *  *  * *  **
   ****  * *     *      *  * * *  **
 *          *  * * * *  * *  * **   * **
   ** * *** * **    *** *  * *  * * *
**               **  *  ** *   *    * *
   * * **  * ** **  *  *     *   ** *
 **   *   * *     *     * *   *       *
```

```
 ***************************************
***************************************
 ***************************************
***************************************
 ***************************************
***************************************
 ***************************************
```



Example of a randomly generated grid



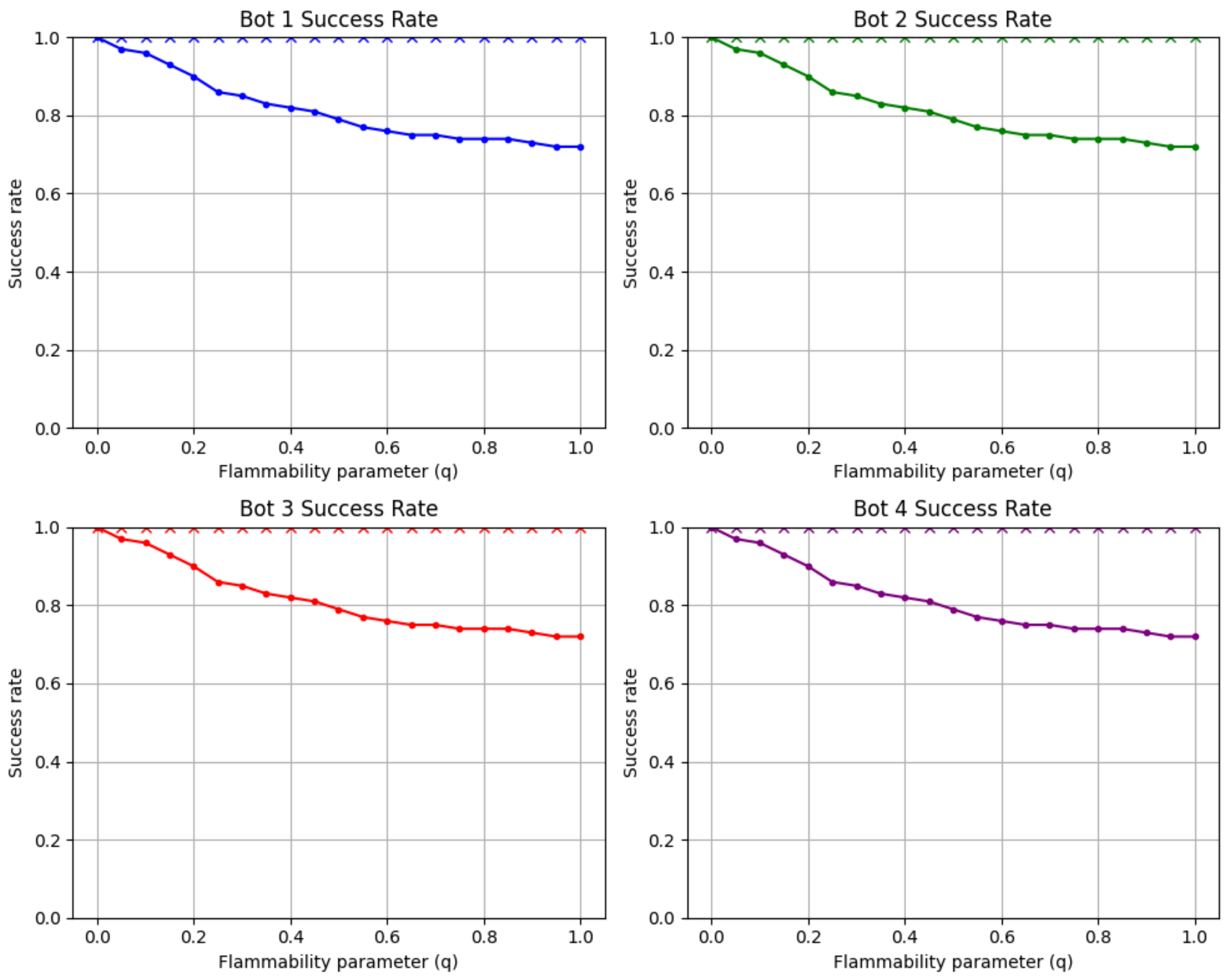
Example of an optimal grid for bot success

The optimal grid abides by the rules used to create the random grid, and a similar layout could theoretically be generated by the algorithm outlined in Section 2.1. However, dead ends are not opened here because they would decrease the bots' success rates. The resulting grid has 51.25% open cells, significantly lower than the average of a randomly generated grid. Other valid optimal grids include forming a spiral of open cells or any other form with a single long open path.

Intuition: In a typical randomly generated grid, the fire has a considerable advantage over the bots due to the ample space available for it to spread. This often makes it difficult for the bots to succeed. The bots are given a higher chance of winning by creating a single, long corridor where the fire can only spread in two directions. This is because the fire's movement is limited, and only one of the two possible directions can impact the bots' progress, given the constraints of the grid. Let us consider the following situations:

- **Fire between Bots and Button**: As there is only one long path for the bots, which is now blocked by the fire, this scenario is treated as impossible and skipped by the program.
- **Bots between Fire and Button**: The maximum distance the fire can travel at each instance is one cell, even with the highest flammability. Since the bots also move at the same speed, they will always reach the button and succeed.
- **Button between Bots and Fire**: This is the only case in which the bots can fail since it becomes a race between the bots and the fire. If the fire reaches the button first, the bots

fail. However, it is essential to note that if the bots fail, no alternative strategy would have led to success, as the path is singular.



The bots were tested using the same conditions described in Section 3.1 to ensure no inherent advantage to any bot or flammability parameter q. For each q value, 100 simulations were conducted to assess performance. Since the grid has only one path to the button, the performance of all bots is identical. Bot 1's simple BFS strategy leads to success just as effectively as Bot 4's more complex combination of exploration and exploitation strategies. As expected, the results show that the bots achieved a 100% success rate in all cases where success was possible.

5 Conclusion

The exploration of bot performance in fire-spreading environments aboard the deep-space vessel *Archaeopteryx* has provided valuable insights into the complexities of navigating dynamic hazards. The comparative analysis of the bots highlighted distinct performance patterns, revealing that success is significantly influenced by the parameter q. Bots demonstrated varied levels of adaptability and effectiveness, with lower values of q favoring their strategies. In comparison, higher values often resulted in diminished success rates due to the escalating intensity of fire. The analysis showed that the ideal bot should possess advanced pathfinding capabilities, real-time adaptability, and predictive fire behavior modeling. By leveraging these attributes, the ideal bot could enhance its chances of successfully extinguishing fires while minimizing operational risks.