PROJECT 1: Maze & Fire

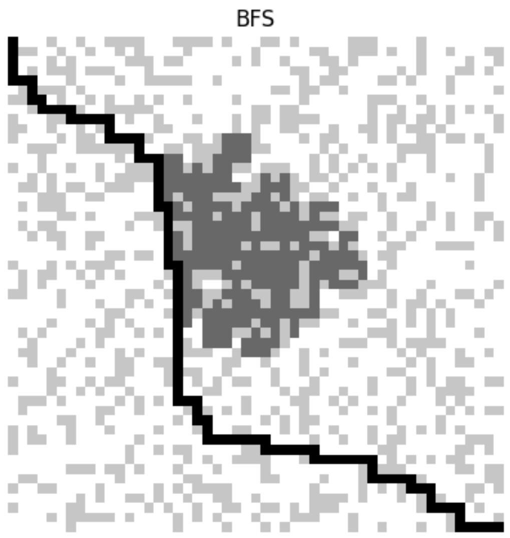
Intro: This project has been completed by analyzing the differences between multiple search algorithms. We liked to think of this project to compare and contrast the differences between how different types of algorithms like DFS, BFS, etc., make decisions based on the general approach of each different algorithm and also given specific constraints. In this project, we will see how these algorithms *(Not all)* make decisions based on the 51 x 51 square grid and go into a deeper understanding and dialogue about the paths these search algorithms take.

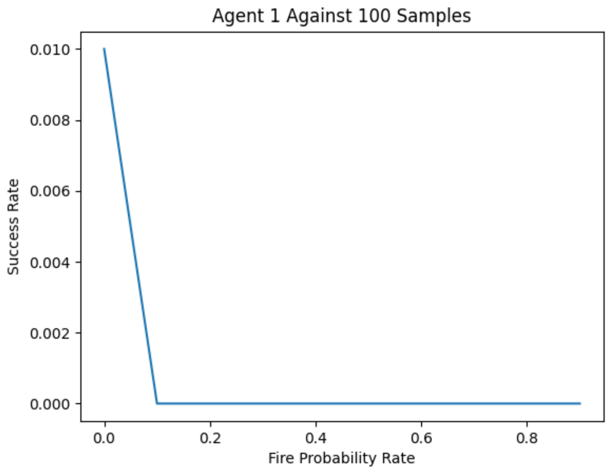
Tools Used: The tools that have been used for this project will be listed below in this section of the write-up. When running our code, please ensure these tools have been installed on your local machine and coding environment.

- Python 3.8
- NumPy *(Python Library)*
- Matplotlib *(Python Library)*

Explanation: The prompt that was given to us for this project was split into four different agents doing different things, given different constraints, using somewhat of a different algorithm for each. Below will describe how each different agent differs from each others and how they all work, with an explanation of our findings from each agent

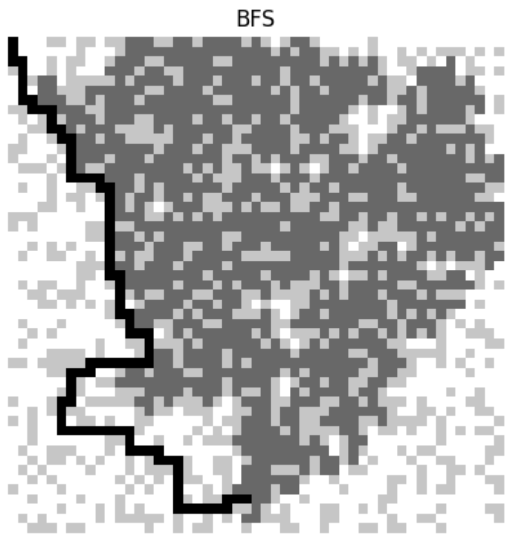
Agent 1: Agent one was designed to be the most superficial agent out of the four, we kind of thought about agent one as a "base case" in a sense compared to the other agents. Agent one was relatively simple. Plan a route from start to end having the agent strictly follow this route until he either makes it to the end of the path or dies during the process. Agent one does not follow any specific constraints or obstacles within its path, at least in our case, it strictly follows the Breadth-First Search algorithm until the agent either makes it or it doesn't.





In the above image, we see that in the specific example displayed, agent one is able to reach across the maze. Black represents the agent, dark grey represents the fire, and light grey represents walls of the maze. The fire spread probability in the above example was set to .10, once the fire spread probability reaches anything above .10, there is a steep drop in the Agents' success rate in completing the maze without hiting the fire in its path. In the analysis to the left, we can see that the success rate of agent one declines as the fire probability rate increases. There is a noticeable drop in the success rate when the fire probability rate reaches a rate of anything above .10, which makes sense since for agent one, the agent isn't adapting to the fires spread but only following its predetermined path from start to death *(or finish)*.

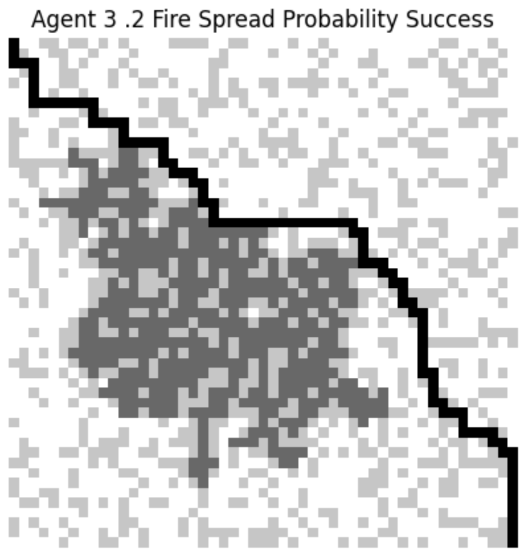
Agent 2: Agent two is a bit different from Agent one, but follows the same Breadth-First Search algorithm as previously implemented in agent one. Agent one doesn't quite adapt to the constraints around it, in this case, the fire, but agent two does. Each time agent two moves, we check to see if one of his points through the maze will be on fire. If there is a point that will be on fire, agent two will reroute, otherwise, it will continue its path. Throughout the entire algorithm, this is what agent two does. It continuously checks to see if any of its paths will be on fire after every single move it makes through the maze unlike agent one.



Although agent two was extremely similar to the algorithm in agent one, the results were interesting to analyze and compare and contrast. Lets recall that agent two adapts to the spread of the fire, rather than just running through the maze without worrying about the fire at all like agent one. Now, since agent two adapts to the fires spread and reroutes depending on where the fire has spread at the agents' each step, we noticed that although the results were extremely similar to agent one *(agent not being able to complete the path),* we noticed that the agent was lasting a lot longer. In the above image, agent two travels ninety-two steps before dying by the fire. What's also extremely important to note is that the the fire probability rate is at .30. In agent two, the agent is lasting longer through the maze at a higher fire probability rate, showing us that the algorithm is much more smarter compared to the algorithm used for agent one which is the main takeaway when analyzing agent two and comparing it to agent one.

When we ran our python script that tests analysis on the specified agent function call, we noticed that the plot for agent one was the same if not almost the same as agent two. It took some time to think about why this might be happening, but thinking about it on a baseline level, agent one and agent two are essentially using the same type of algorithm, Only that agent two is continuously trying to work around the fire while agent one was not. You can consider the algorithm that agent two is using as smarter than the algorithm that agent one was using since agent twos algorithm is adapting to its environment and trying to survive.

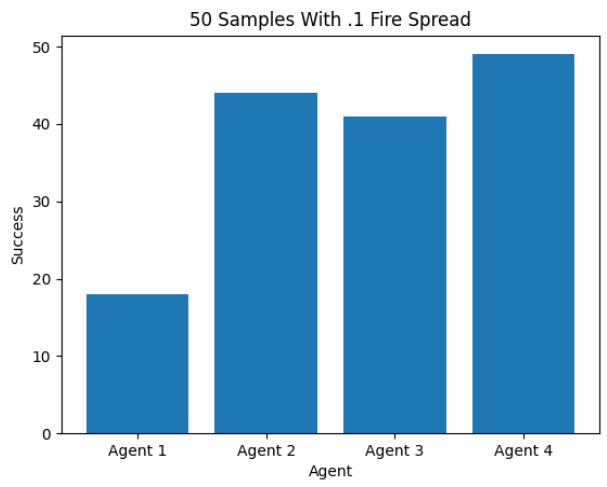
<u>Agent 3</u>: Although we can consider agent two being smarter than agent one, lets introduce a new agent, agent three. Agent three builds off of the working knowledge of agent two. Agent three is able to know where there might be a fire in its possible path well in advance and it is useful through the agents navigation through its generated maze. Agent three is constantly checking and "looking" into the future of its path to see if it must plan a new route around the fire in advanced.

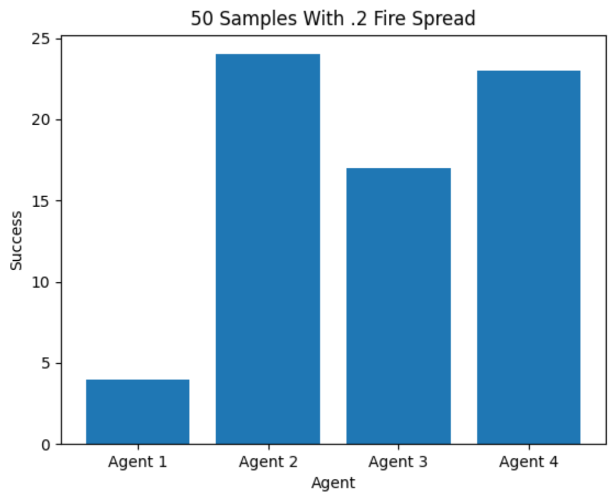


Since agent three is able to “look” into the future more deeply into its path compares to agent two and make better informed decisions about its path and survival, you would assume that the survival rate of agent three would be higher than agent two’s? That is not always the case and there will be data at the end of the report to prove that point. Although you can technically classify agent three as a smarter algorithm compared to agent two, they both tend to run somewhat at the same survival rate. Comparing Agents one and three are a night and day difference. As we are starting to see inside the project, Agent one is starting to have the worst success rate out of agents’ one, two, and three, which makes sense since we classified agent one as somewhat of a base-case for this assignment.

Agent 4: Agent four can be considered to be the best and smartest agent out of them all. When running our analysis, we noticed that agent four had the highest survival rate out of all of the agents, but more importantly it had the highest success rate with the most amount of steps out of all of the agents. Comparing agent four to the other agents, agent four shares characteristics of agents one and three. The only planning that takes place within agent four is done at the start of the path, just like in agent one. While it follows the same planning scheme as agent one, it takes agent threes ability to plan its path based on the state of the maze three moves in advance to the extreme value. This entire process is done by simulating the maze and the fire speed up until the path from the start to the goal “node”no longer exists. At this point, agent four takes the last possible path to the goal all while anticipating the fire directly.

Comparison of all Agents' Success


50 Samples With .1 Fire Spread
Success
Agent 1
Agent 2
Agent 3
Agent 4
Agent


50 Samples With .2 Fire Spread
Success
Agent 1
Agent 2
Agent 3
Agent 4
Agent