1. My Bot 4’s algorithm is somewhat of the previous bot’s strategies combined but instead of breath-first search, I’m using A\*, which seems to make the algorithms extremely more efficient.

To test for efficiency, I pre-generated and saved 100 different 50x50 ship layouts created by the required algorithm and for each bot, I ran 10 different simulations with randomly generated bot, fire and button location on each map. After averaging the result, we get:

0.0003s Average delta-time

Notice: delta-time is the time used to run one frame in the gameloop, while the time does include the calculation of the fire spreading, but this is quite small and can probably be ignored. Flammablility 0

Ideal bot

One thing that we could possible do to increase the success rate even more is to predict the direction or the precise location of the next cell that is going to catch on fire, since from the equation the more burning neighbors that a cell has, the more likely its going to catch on fire. Bot 3 and 4 only tries to avoid the fire by not going anywhere near the cells that are burning which could make the path longer. For instance, if the fire somehow spreads the same or similar direction as the bot moves, the bot might end up wrapping around the fire just it doesn’t know what the fire might do next, but if at an earlier timestep the bot has taken a path that’s opposite to the fire spread, it certainly would be shorter.

Optimizations for the A\*/BFS searching algorithm for this specific task,

The heuristic that I’m using is the Manhattan distance, maybe there is something else that is better such as the Euclidean distance