# Project 1: Invasion of the Bot-Grabbers

16:198:520

It is another day on the deep space vessel *Archaeopteryx*, and you are a tiny bot. You're responsible for the safety and security of the ship and crew. Unfortunately, the ship has been invaded by hostile aliens intent on kidnapping you and your crew. Your crew have taken refuge beneath the floorpanels in the ship - your job is to find them and teleport them away to safety, before you are grabbed by the aliens.

## 1 The Ship

The layout of the ship (walls, hallways, etc) is on a square grid, generated in the following way:

- Start with a square grid, $D \times D$, of 'blocked' cells. Define the neighbors of cell as the adjacent cells in the up/down/left/right direction. Diagonal cells are not considered neighbors.
- Choose a square in the interior to 'open' at random.
- Iteratively do the following:
  - Identify all currently blocked cells that have exactly one open neighbor.
  - Of these currently blocked cells with exactly one open neighbor, pick one at random.
  - Open the selected cell.
  - Repeat until you can no longer do so.
- Identify all cells that are 'dead ends' - open cells with one open neighbor.
- For approximately half these cells, pick one of their closed neighbors at random and open it.

*Note: How big an environment $D$ you can manage is going to depend on your hardware and implementation, but you should aim to generate data on as large an environment as is feasible.*

## 2 The Bot

The bot occupies an open cell somewhere in the ship (to be determined shortly). The bot can move to one adjacent open cell every time step (up/down/left/right). If the bot enters a cell occupied by an alien, or an alien enters the bot's cell, the bot is grabbed, and fails its mission.

## 3 The Aliens

$K$ many aliens are initially randomly placed at open cells throughout the ship. At every time step, the aliens each can choose to move, up/down/left/right, or stay in place. The aliens can be advanced according to the following steps:

- Order the aliens at random.
- In the selected order, for each alien, choose an action at random that doesn't move that alien into a blocked or alien-occupied cell.

- Move the selected alien according to that action.
- Process each alien in this way, in the selected order.

Note, the order should be re-selected at random every time the aliens are advanced.

# 4 The Crew

A crew member will be located at a random open cell. It does not matter if the cell is ever occupied by aliens. The bot's goal is to make it to that cell while it does not have an alien in it. If the bot is successful, the crew member is successfully teleported away.

# 5 The Task

At time $t = 0$, place the bot and aliens at random, distinct open cells. Additionally, place the first crew member at a non-bot-occupied open cell. At each time step $t = 1, 2, 3, 4, \ldots$, the following happens in sequence:

- The bot decides which open neighbor to move to.
- The bot moves to that neighbor.
- If the bot enters the crew member cell, the crew member is teleported away. At this point, a new crew member is detected (hidden in a non-bot-occupied cell).
- Then the aliens advance.
- If at any point the bot and an alien occupy the same cell, the task is failed.

Ideally, the bot navigates to the crew member, avoids aliens, and teleports the crew member away. This continues at most $t = 1000$ time steps, or until the bot is captured. The goal of this assignment is to build and evaluate different strategies for governing how the bot decides where to go.

Note: we can evaluate the quality of the bot in two different ways. The first way, we can judge a bot based on how many crew it successfully saves before the time limit or it does. The second way, we can judge a bot based on how likely it is to survive to the time limit. We will use both of these to evaluate bot designs.

# 6 The Strategies

At any time, the bot can choose from the following actions: move to a neighboring open cell, or stay in place. The thing that differentiates the strategies is how the bot decides what to do.

For this assignment, you will implement and compare the performance of the following strategies.

- **Bot 1** - This bot plans the shortest path to the current crew member, avoiding the current aliens, and then executes that plan. The movement of the aliens is ignored by the bot between crew member rescues.
- **Bot 2** - At every time step, the bot re-plans the shortest path to the current crew member, avoiding the current alien positions, and then executes the next step in that plan. Because it repeatedly replans, it constantly adapts to the movement of the aliens.

- **Bot 3** - At every time step, the bot re-plans the shortest path to the current crew member, avoiding the current alien positions *and any cells adjacent to current alien positions, if possible*, then executes the next step in that plan. If there is no such path, it plans the shortest path based only on current alien positions, then executes the next step in that plan.
- **Bot 4** - A bot of your own design.

*Note: The only restriction I am putting on Bot 4 is that it cannot be a version of Bot 3 that just puts a wider buffer on the aliens. Do something more interesting. Be creative.*

Each bot will be evaluated on a number of ships, at different numbers of aliens.

# 7 Data and Analysis

In your writeup, consider and address the following:

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?
2) For each bot, repeatedly generate test environments and evaluate the performance of the bot, for many values of $K$. Graph the two measures of success by averaging results over multiple experiments, graphing them as a function of $K$. Be sure to repeat the experiments enough time to get accurate results for each bot, and each tested value of $K$.
3) When the bots fail to perform well, with respect to either measure - why? Why do they fail? Was there a better decision that could have been made that would've saved the bot longer, or saved more crew members? Support your conclusions.
4) Speculate on how you might construct the ideal bot. What information would it use, what information would it compute, and how?

***Bonus:*** *In the above, we considered randomly generated ship layouts. If you were to design the ship layout deliberately, maybe the bots would be more effective. The only restriction is that all open cells must be reachable from each other. Write a program to design/discover the best ship layout you can. Explain your logic and algorithm design choices, as well as giving the final ship layout. Justify, if you can, why it is as effective as it is.*