Project 2: Airing Out

01:198:440

It is another day on the deep space vessel *Archaeopteryx*, and you are a lonely bot. You're responsible for the safety and security of the ship while the crew is in deep hibernation. As an example, in the case that the ship springs a leak, you need to find that leak and plug it.

1 The Ship

The ship layout is as in Project 1. For the purpose of this project, set the dimension to 50×50.

2 The Bots

The bot occupies an open cell somewhere in the ship (to be determined shortly). The bot can move to one adjacent cell every time step (up/down/left/right). Each bot will have access to a different kind of sensor data to help determine where the atmosphere leak in the ship is - the purpose of the bot is to utilize the available sensor information to locate and reach the atmosphere leak as quickly as possible. The bot will start at a randomly selected open square in the ship. The bot knows / can sense when it enters the cell where the leak is located.

> At each timestep - the bot must choose whether to **move** or to **sense**.

3 The Leak and the Task

The leak will be located at a randomly selected open square in the ship. The bot will not know the location of the leak, but can determine it via data collection, inference, and moving through the ship. If the bot enters the cell the leak is located in, the leak is plugged and the task is complete.

4 Part 1: Deterministic Leak Detectors

At every timestep, the bot can detect whether there is a leak in the vicinity of the bot (a boolean yes/no detection). For a given $k \geq 1$, the bot sits at the center of a $(2k+1) \times (2k+1)$ square, and can detect whether or not a leak is present in that square. The bot can detect if it enters the square with the leak.

> While the bot is initially placed randomly, the leak is initially placed in a random cell outside the detection square.

- **Bot 1:**

  All cells outside the initial detection square start with the possibility of containing the leak (essentially, the bot starts having taken a sense action, and detected nothing). When the bot enters a cell (or starts in a cell), however, if it is not the leak cell, it is marked as not containing the leak. If the bot detects no leak in proximity - all cells in the detection square are marked as not containing the leak. If the bot detects a leak in proximity - all cells in the detection square not already marked as not containing the

leak are marked as possibly containing the leak, and all cells outside the detection square are marked as not containing the leak. Note that if a single square remains that is marked as containing the leak and all others do not contain the leak - the leak must be in that one marked cell. Bot 1 acts in the following way:

  - At any time that has not detected a leak, it will proceed to the nearest cell that *might* contain the leak (breaking ties at random), enter it, and take the sense action, updating what it knows based on the results.
  - At any time that a leak has been detected, it will proceed to the nearest cell that *might* contain the leak, enter it, and in doing so either find the leak or rule that cell out.

  This proceeds until the leak is discovered.

- **Bot 2:** A bot of your own design that uses the data from the detection square.

5 Part 2: Probabilistic Leak Detectors

In this case, the detection is probabilistic - the bot has a sensor that, when activated, has a probability of giving a beep if the leak is nearby. The nearer the bot is to the leak, the more likely it is to receive a beep. Note that if the bot stays in place, it may receive a beep at some timesteps, and not at others (with the correct probability). If the bot is d-steps away from the leak (shortest path through the ship), the probability of receiving a beep is $e^{-\alpha(d-1)}$, for $\alpha > 0$. Note that if the bot is immediately next to the leak, the probability of receiving a beep is 1.

- **Bot 3:** All cells (other than the bot's initial cell) start with equal probability of containing the leak. Bot 3 proceeds in the following well:
  - At any time, the bot is going to move to the cell that has the highest probability of containing the leak (breaking ties first by distance from the bot, then at random).
  - After entering any cell, if the cell does not contain a leak, the bot will take the **sense** action. Based on the results, it updates the probability of containing the leak for each cell*.
  - After the beliefs are updated, this repeats, the bot proceeding to the cell that as the highest probability of containing the leak (breaking ties first by distance from the bot, then at random).
- **Bot 4:** A bot of your own design that uses the data from the observed beeps (or lack thereof). Note that Bot 4 may choose to say in place for a given timestep, if that is useful to do.

* *Note, there is a **correct** way to update the probabilities. You should work out what the correct updates are.*

6 Part 3: Multiple Leaks

In the above sections, we considered a single leak. But suppose there are two leaks simultaneously that need plugging. Assume that once one leak is located (and the square entered) it is plugged, leaving one remaining leak to locate and plug.

- Deterministic Leak Detection
  - **Bot 5:** Bot 5 is exactly Bot 1, but removes the first leak once its cell is entered, and continues until the second leak is also identified and plugged.

  - **Bot 6:** Modify your Bot 2 to better handle this two leak situation. Is there anything significant that needs changing?

- Probabilistic Leak Detection - Note that the probability of receiving a beep is now the probability of receiving a beep from one or the other leak.
  - **Bot 7:** Bot 7 is exactly Bot 3, but removes the first leak once its cell is entered, and then continues searching and updating until the second leak is identified and plugged.
  - **Bot 8:** Bot 8 is exactly Bot 3, *except that the probability updates must be corrected to account for the fact that there are two leaks - how?*
  - **Bot 9:** A bot of your own design, that uses the corrected probability updates for there being two leaks.

7 Data and Analysis

In your writeup, consider and address the following:

1) Explain the design and algorithm for the bots you designed, being as specific as possible as to what your bots are actually doing. How do your bots factor in the available information (deterministic or probabilistic) to make more informed decisions about what to do next?

2) How should the probabilities be updated for Bot 3/4? How should the probabilities be updated for Bot 8/9?

3) Generate test environments to evaluate and compare the performances of your bots. Your measure of performance here is the *average number of **actions** (moves + sensing) needed to plug the leak.*
   - Bot 1 vs Bot 2, as a function of k.
   - Bot 3 vs Bot 4, as a function of α.
   - Bot 5 vs Bot 6, as a function of k.
   - Bot 7 vs Bot 8 vs Bot 9, as a function of α.

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

Bonus: *Assume that instead of a stationary leak, there is an alien intruder that at every timestep moves in a random direction. Design (/modify), implement, and test updates to Bot 2, Bot 4, Bot 6, and Bot 9, with this in mind.*