Snake AI – Pathfinding: Project 2

By Pierre Turle (pturl001)

Overview

I had an extremely hard time with this assignment. There were parts where I felt like resubmitting an improved version of my previous assessment, however, since this was a pathfinding assignment I tried right up until the end to NOT use turn left/ right statements, which I understood to mean that I cannot logically maximise the amount of free space on the grid. The obvious way to maximise space is to make the snake go between corners between finding food, and this is how I started.

Eventually, as I could not beat the goToFood method which you made, I ended up just using it alongside a time stalling function when trapped. Because of the ambiguity between the rules you set and the marking for the last assignment, I was unsure of whether you wanted creativity or efficiency. I tried various methods using pathfinding, and none were as effective as GoToFood. I believe this may be a trend that you have seen in previous years as well.

Below I will list various methods I tried out which didn't work as well as a basic GoToFood statement. You can see various snippets of the code in the "graveyard" at the bottom of my code, as I had before.

Failures

Home Variable and Alternation

I made 2 variables called "oldMe" and "home". Home was initially set to (2, 2) and oldMe was the snakes body length. oldMe would only update when the snake was at the home tile, meaning that I could differ between when the snake had eaten and when it was going home. Using this I tried to alternate between the snake eating food and going home. If the food spawned near home the snake would not be able to reach it and would just crash, so I added code where if home was not reachable it would re-assign itself to (3,3) and so on, but that still just added another point which it might not be able to reach due to blockages, which I realised was the point of the assignment. To try avoid this I adapted the concept to have multiple homes.

Multiple home variables

To reduce the likelihood of getting trapped, I initialised a home in each corner. If one was not reachable (irrelevant to its own body blockages) after getting food the snake would go towards another one. I later realised how little this actually did seeing as getting trapped in its own body away from a corner was very possible, and that the snake going to the corner it was trapped in would not add very much longevity (going diagonally towards a space ¼ of the grid away is often a few moves and not enough to allow the snake to untangle itself). I initially had code for the snake to go between homes after obtaining food too but instead of facilitating the snakes longevity this just changed the path which may be blocked, and though the snake did survive for longer, it was not long enough to beat the vanilla pathfinding you mentioned.

Safe Spaces if trapped

Similar to above, I added a home seeking behaviour to “food not reachable” parts of my code. The issue, as I found, is that when the snake is trapped it needs to react to the space it’s trapped in, and not to a default space. Even using points like Vector2Int(Snake.HeadPosition.x, Snake.GridSize.-2) would only work if the right wall was free. In my last submission the snake behaved always in response to what was in front of it and due to it’s one directional cyclic nature, I made it go the opposite way when reacting to its own body. This ensured longevity in response to its immediate surroundings which would constantly change. It’s at this point that I gave up and tried to port some of my old code for collision avoidance into this project.

Collision avoidance

At this stage I deleted everything and tried to set 3 different behaviours depending on different states for my snake: “about to hit wall”, “can reach food” and “is trapped”. I used my old code for collision avoidance (which was amazing) but without the food seeking part (seeing as you wanted us to use pathfinding). My old collision avoidance code would make the snake randomly spiral and get trapped. Even when I isolated the code snippet, my snake would spiral endlessly into a corner even though it could reach and had the option to go towards food. As I was separately testing these 3 sections of code to isolate the issue, I found it to work really well once I’d removed my collision avoidance and it worked and so, even though I have 3 statements for this submission, this is the only solution I can come up with where the snake can score high enough to not fail the specs you set us. I Suppose my submission is not this few lines of code but the long hours I’ve spent trying to make something more fun work.

```
return new Selector
(
    // food seeking
    new Filter
    (
        Snake.IsFoodReachable,                  // if the snake can reach food
        new Action(() => Snake.MoveTowardsFood())   //go to food
    ),

    // if CANNOT reach food
    new Filter
    (   // can turn left
        Snake.IsFreeLeft,
        new Action(() => Snake.TurnLeft())
    ),
    new Filter
    (   // can turn right
        Snake.IsFreeRight,
        new Action(() => Snake.TurnRight())
    )
);
```

Conclusion

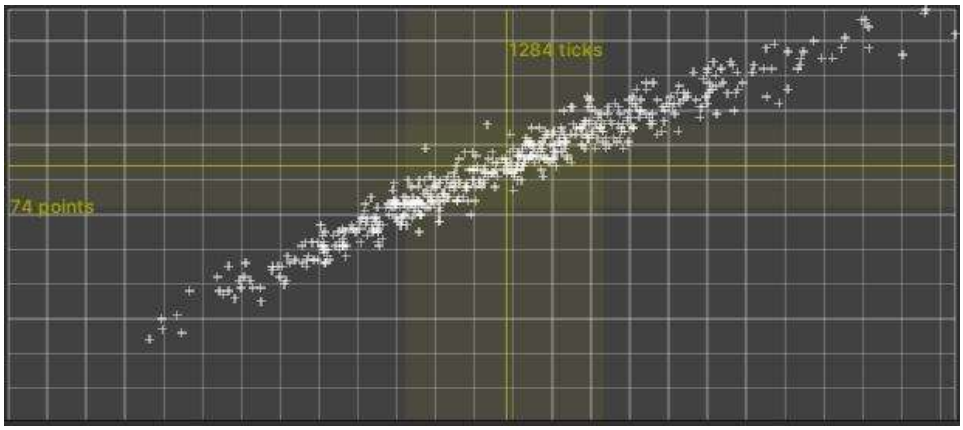
From this assignment I have learned that with a knowledge of the world and the rules, pathfinding (BFS) is not necessarily the best method for solving a problem. If it were a larger, more complex game with more variables (known, random or unknown) and options, then pathfinding would be much more effective.

In (high) contrast to the initial assignment, I also can't really figure out the point of this assignment. I never felt that I had a clear goal or could show what I'd learned. If I had adapted my initial code (slightly) and resubmitted that my snake would have survived much longer and got more food, and although maybe that was what you wanted to see (high scores, ignoring the rules, like the Hamiltonion cycle last assignment) this wasn't obvious. I'd also like to mention that although me writing this will not do me any favours, it's important to analyse the task as well as the results, and as I did with the last exercise, I have done so here.

Overall I went through several attempts to try stick to the specs and none could work as well as me writing 3 statements. Considering the amount of work I put in this feels very disappointing.

Scatter Plot and screenshot

A scatter plot for my results



This is the winding behaviour when food is not reachable