Assignment – 3

Foundation of Artificial Intelligence – CS2701

Name: Khushboo Thaker

ID: 4147510

Email: [kmt81@pitt.edu](mailto:kmt81@pitt.edu)

## The choice you’ve made to get the learning framework going. Have you tried different state representations? Have you tried different exploration approaches? Have you tried different learning rates? How do they impact the learning process?

**State Space**

I tried 3 different ways for my state space representations

1. Height and width between the bird and the pipe

S(X,Y)

1. Height and width between the bird and the pipe with pixelate subspace
   * S(XP,YP)
   * I.e. XP = X / n and YP = Y / n( this reduces the search space and combines different nearby pixels to represent search space)
   * I tried **n** ranging from 4,8,16,32 but found out that 4 is better as 8-32 are faster initially due to less number of states but are not able to go long way
   * Why these values! Because bird height and width of bird is 32 so it is relative also each pipe module is 32 in height so it is in proportion.
2. Height and width between the bird and momentum with which the bird flies
   * S(X,Y, M) but this reduces the search space enormously and its hard to manage for my desktop. But I found this state space very interesting though I don’t have good results to show on this space. But this space can be modified in the pixelate and momentum subspace to provide an efficient bird.

**My final state space is 2) S(XP,YP) with n = 4**

**Exploration Techniques:**

1. First always go in one direction in default works as being discussed on the website shared with the assignment
2. Created a visitedlist and first explore any of the unvisited paths rather than exploring already evaluated path
3. Random

My exploration technique can be found in rL.py python file in program **getNextAction()**

1. First Priority is for not done (visited) actions
2. Now if the both the actions are done and still the q values are 0 then a random step is taken
3. Otherwise the normal step of taking the path with high q value

**Conclusion: Although exploring this way is tedious and slow it comes with fast learning at a later stage**

**Learning Rates:**

* Learning rates and rewards play a crucial game. Initially my rewards were very low and I learned that they were not able to propogate till the end stage
* I chose alpha = 0.7 but cannot try **dynamic alpha based on visitation of nodes** though I am sure this would have increased the learning process
* I chose gamma = 0.8 as it gives importance to immediate rewards.

## Overall, how long did it take you to train the bird? What do you think you’d need to do to make the training more efficient? (Asked in another way, If you had more time to work on this project, what would you do next?)

1. For fixed pipe length it took me 2 hours to learn to cross 16 pipes
2. For random pipe length it took me 3 hours to learn only to reach to pipe 5

Due to last minute learning I cannot give more time for learning my flappy bird. If possible I would like to know if I can submit my more trained bird by tomorrow morning.

As discussed in the first question the the better statespace representation can reduce learning time specially S(XP, YP, MP) with momentum adding an extra crucial information. I would like to train my bird with this setup if I get more time on the project

## A brief description of how to load your trained bird into the program. (The grader won’t be able to wait for your bird to train from scratch.)

## Flappy bird with fixed pipe which reaches atleast 16 pipes

## **Python flappybirdRandom.py**

1. Flappy bird with random pipe which reaches atleast 5 pipes

**Python flappybirdtrained.py**

1. Flappy bird with random pipes without any training

**Python flappybird.py**

I haven’t used any additional tools but the I have done discussion with Professor Rebecca which helped me the problem of unlearning the learned states in further runs

## 

## What different heuristic functions have you tried before settling on your final choice? What ideas worked? What didn't? Be sure to describe your final heuristic function in sufficient details.

The different heuristics I tried are as below:

1. Number of blocks reachable by the opposite queens
2. 1/ Number of blocks reachable by my queen
3. Number of adjacent blocks open for my queens
4. Number of adjacent blocks open for opposite player queen
5. 1/ (Number of blocks I am reachable – Number of black opposite player is reachable)
6. Random

I tried playing all the heuristics against random player for 100 runs and I found that the No of adjacent blocks heuristics wins most of the time although sometimes with large number of moves and very few blocks left. So I selected it to be the final heuristic

My final heuristic is 4) No of adjacent blacks open for opposite player queen. As my code is based on prime locations this heuristic makes more sense

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Below is the example: Table 2   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  | |  |  | Q |  |  |  |  |  |  |  | |  |  | Q |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  | q |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | | Intial Board condition given to Player with white queen -Q |
| **Possible Move1**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  | |  |  | q |  |  |  |  |  |  |  | |  |  | x |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  | Q |  |  |  |  |  |  | |  |  |  | q |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |   **Possible Move 2**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  | |  |  | q |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | x |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  | Q |  |  |  |  |  |  | |  |  |  | q |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  | | So as the move 1 helps in blocking 2 queens this would be given priority over the move 2 |

**References:**

Lecture Notes

Class Notes

<https://en.wikipedia.org/wiki/Game_of_the_Amazons>

Jens Lieberum, An evaluation function for the game of amazons, Theoretical Computer Science, Volume 349, Issue 2, 2005, Pages 230-244, ISSN 0304-3975, http://dx.doi.org/10.1016/j.tcs.2005.09.048.