Pursuit Evasion on Graphs Using Java

Abstract:

The real world problem I explored in this project was simulating pursuit evasion on graphs when there were 2 players, with one trying to get to the other, and the other trying to get away. To better understand the problem in this project, I conceptualized the players and their strategies and tested each file. The core CS concepts I used were algorithm design, data structures, and time and space complexity. My key findings were understanding game dynamics, and implementing different player visualization techniques.

Results:

The experiments I ran in this program were RandomPlayer, move towards the player, and move away from the player. The metrics I reported for for the outcomes of the experiments were the game outcome, the total moves, and the execution time. The outcome of the results ended up being that the algorithms that moved towards the player and away from the player ended up doing better than the random player algorithm. The move away from player algorithm was the most successful of the three algorithms. These results suggest that strategic and targeted algorithms are more effective than random ones when it comes to pursuit and evasion.

Extensions:

I didn't do extensions for this project.

Reflection:

1. To implement the Graph, vertices and edges are stored in ArrayLists and use a HashMap. Adding vertices and edges and looking for adjacent ones have shorter run times. Methods with longer run times are the ones removing elements in the list and traversing the list to check through it.
2. For the Voronoi game, I changed parts of the Greedy approach because it seemed to be the most efficient approach to solving the Voronoi game. It seems reasonable because prioritized the vertices with the highest values and most coverage over the graph.

Acknowledgements:

I worked with Kama on this project and I also went to TA hours.