Community detection algorithms

for real and artificial networks

Project Report - Group 3

Fábio Cruz

Nr. 96957

fabio.rocha.cruz

@tecnico.ulisboa.pt

Luís Fernandes

Nr. 97018

luis.martins.fernandes

@tecnico.ulisboa.pt

**ABSTRACT**

In this paper, we analyze different community detection algorithms along with real and artificial benchmark networks. We also compare and evaluate those algorithms by analyzing their internal and external measures, ranking them using some of those criteria and checking the accuracy of the methods by comparing the resulting partitions with the expected ones for the different networks.

# INTRODUCTION

Pacman is one of the most iconic games of all time, where the player controls Pac-Man, a yellow round face with an open mouth, who must eat all the dots inside an enclosed maze while avoiding four colored ghosts. The main goal is to eat as many dots as possible, thus advancing through levels and increasing the game score.

In the original game, each of the ghosts is programmed with an individual "personality", a different algorithm it uses to determine its method of moving through the maze. This is where we will apply our twist to the game. Instead of the ghosts having this individual AI behavior, we want to implement a multi-agent system that allows the ghosts to communicate and coordinate between them in order to trap the Pac-Man. This would be like a reversed Pacman game, where our multi-agent system goal would be to finish the game as soon as possible.

We also implemented the addition of special food, which makes the Pac-Man to be able to temporarily kill a ghost. During this period the ghosts have something like a fear emotion, where they try to run away from the Pac-Man.

Regarding the Pac-Man in our version of the game, it can’t be controlled by a player during the development since that would influence the results of our experiments because we want the Pac-Man to have the same behavior throughout the experimental tests. The solution that we implemented was that the Pac-Man was also an agent. Initially it had a reactive behavior, and later on we upgraded it to use a simple algorithm to escape the ghosts. We also implemented to option of the Pac-Man being controlled by a player so we could try and beat the final version of our coordinated ghosts.

# COMMUNITY DETECTION ALGORITHMS

The multi-agent system that we suggested was a typical Pacman game structure regarding the characters, with one Pac-Man and four ghosts. These ghosts would have to cooperate and communicate between them in order to catch the Pac-Man as soon as possible, thus forming a small homogeneous society.

Initially the idea was that the ghosts would learn with experience trough reinforcement learning, using a reward function based on the time since the beginning of the game and the points of the Pac-Man. Unfortunately, due to our current situation and lack of personal time, we were not able to implement this part of the project, so we decided to take a slightly different approach to the project, which we will explain later in this document, in order to have some experimental results based on the implementation of the ghosts.

Since it would probably be too easy for the ghosts to succeed if they had knowledge of the whole map and positions of the other ghosts and the Pac-Man, we decided that each ghost only knew the positions of the agents (ghosts and Pac-Man) within a certain area around them. For example, if Pac-Man is within a ghost’s area that ghost “sees” Pac-Man and can try to catch him, and if he “sees” another ghost, they can communicate by exchanging information that might be useful, such as the position of the Pac-Man, and then coordinate to catch him.

We implemented two main types of behavior for the ghosts during the game: an active behavior, where the ghosts actively try to catch the Pac-Man by communicating information about the position of the Pac-Man between each other and coordinating to catch him, and also a different behavior when the Pac-Man eats a special type of food that gives him the ability to temporarily kill the ghosts. In this period the ghosts behavior is to run away from Pac-Man. Taking this into account, we can consider that the ghosts are some sort of emotional agents that can be aggressive (active phase) and scared (running phase).

Regarding the Pac-Man agent, for the active phase of the ghosts we first implemented a reactive behavior to facilitate the observation of the ghosts learning improvement. With the progression of the implementation of the project we later gave this agent a more complex behavior, where the Pac-Man tries to choose the paths that lead to the better utility, which in this case is surviving for as long as possible by always running in the opposite direction of the ghosts when he sees them in his area of vision. We also wanted to improve on this behavior and make this agent follow the paths with more food when not in danger of being eaten by the ghosts, but again due to lack of time we couldn’t implement this since it is also not a crucial part of the project. Regarding the running phase of the ghosts, when the Pac-Man eats the special food and has the ability to temporarily eat them, he reverses his behavior and tries to catch the ghosts while that period is active.

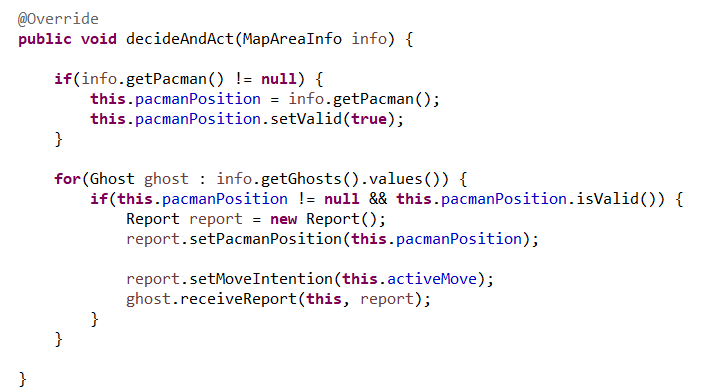
Summing all up, our project has a social architecture where the ghosts have a limited vision of the game map and communicate and coordinate when they can in order to achieve their goals. Adding to this, the project also has an adaptative behavior for a single agent, the Pac-Man, where he also acts in order to maximize its life time by trying to escape the ghosts, which it can detect also with a limited vision. Both these architectures have a responsive behavior that allows the agents to react in time when there are changes in the environment. The control structure is decentralized and goal-directed for the ghosts, since there isn’t a single agent that coordinates everything, and the agents always behave in order to improve their results and achieve their goals.

# NETWORK ANALYSIS

For the implementation of the project we decided to save some time in the graphic component of the project by using an already existing original Pac-Man project. We then proceeded to change it first in order to have both the possibility of a player using the Pac-Man, and it using a reactive behavior to facilitate the testing and implementation of the ghosts against this AI agent. After that we changed the decision behavior of the original agents of the game until we reached the current state of the game. The implementation of these agents will be explained next.



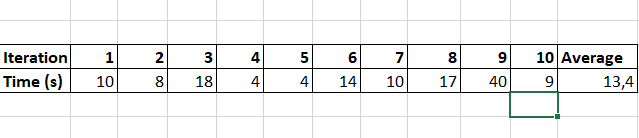
approach to implement this change on the ghost’s movement. We ended up implementing a randomness on half of the ghosts when choosing a path when they are in a place on the map where they can choose more than 2 possible directions (generally on junctions). So this means that when all the ghosts are chasing Pac-Man 2 of them go directly after him using the shortest path possible, and the other 2 do the same but with a probability of 40% of making a different turn on junctions. Our hope with this was that this made them take different routes to intersect the Pac-Man and catch him quicker consequently.



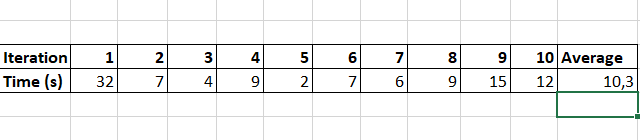
# RESULTS

Regarding the results, the main results that we planned to have in the proposal of this project were related to the reinforcement learning that we wanted to implement, and were basically based on comparisons of the time the ghosts take to catch the Pac-Man throughout the hundreds of iterations that were going to be executed while the ghosts were learning. Unfortunately, since this was not implement like we mentioned before, we decided that we would measure the time that the ghosts took to catch Pac-Man of the different phases of the project that we went through and compare them: reactive Pac-Man vs original ghosts; reactive Pac-Man vs improved ghosts (with communication and coordination); improved Pac-Man vs original ghosts and improved Pac-Man vs improved ghosts.

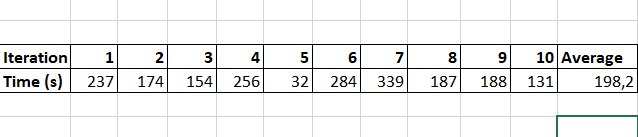
Reactive Pac-Man vs Original Ghosts:



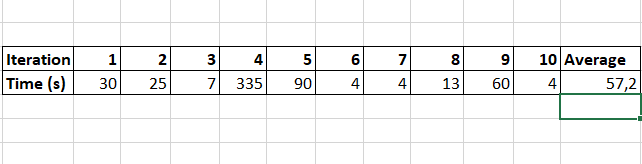
Reactive Pac-Man vs Improved Ghosts:



Improved Pac-Man vs Original Ghosts:



Improved Pac-Man vs Improved Ghosts:



# CONCLUSIONS

First, regarding the results observed above we can take some conclusions.

The most obvious thing that comes up is that the average time that it takes the ghosts to catch Pac-Man when using the original ghosts against the improved Pac-Man is much bigger than any other average. This happens because the only ghost that originally actually actively chases Pac-Man is the red ghost, but this improved Pac-Man is implemented to run from the ghosts when he sees them so he instantly runs away from red ghost when he is nearby, and since Pac-Man’s speed is bigger that the red ghost speed it is very difficult to catch him, unless Pac-Man gets trapped by another ghosts when moving randomly. So we can observe that if the ghosts don’t communicate and coordinate and attack together it is very difficult to beat the improved Pac-Man. The only reason he does not actually win is because he doesn’t follow the food, but instead moves randomly when he doesn’t detect ghosts.

All of these reasons also explain the results with improved Pac-Man and improved ghosts. These results are clearly lower than the ones mentioned before, but are still higher than when testing the reactive Pac-Man because in this case the Pac-Man doesn’t run away from the ghosts and is almost instantly caught, as we can see by those results.

Now if we observe the results when the Pac-Man only moves randomly, we can see that the average time when facing the improved ghosts is slightly lower than when facing the original ghosts. These averages are very close because since the Pac-Man moves randomly he will probably stay in the same zone for a long time, moving forwards and backwards repeatedly. In the original ghosts, this means that the red ghost that immediately aims for the Pac-Man catches him very quickly as it would be expected. In the improved ghosts the times are similar but depends a little on the initial position of the ghosts, because as soon as one of the ghosts is near enough to see Pac-Man he goes after him and catches him very quickly as well, so if the initial positions of these ghosts were placed as far away as possible these results would also obviously be higher.

So the main conclusions about the results are that the communication and coordination between the ghosts is very useful, in particular when they are facing a smart Pac-Man, and that a reactive Pac-Man does very poorly against either ghost’s behavior because the original AI for the ghosts is already very decent and moving randomly makes a very easy target.

To conclude this report, we just want to say again that with more time to implement this project we would like to improve some important things: when Pac-Man doesn’t see a ghost, implementing an algorithm for him to chase the food on the map instead of moving randomly, which would make it much easier for him to win the game in some situations; implement new and different paths for each ghost when the ghosts are chasing to trap the Pac-Man quicker and more efficiently; and finally, make the ghosts (and maybe even the Pac-Man) to learn the best routes and behaviors by using deep reinforcement learning and using a reward function based on the points gained by Pac-Man and the time to catch him.

# ACKNOWLEDGMENTS

Our thanks to all the professors of the course for the help and material provided, to Armin Kazemi for the base Pacman project provided in the Github platform and to all the creators of the information that we used to implement this project.

# REFERENCES

1. Course materials regarding agents and multi-agent systems
2. <https://www.computerhope.com/jargon/p/pacman.htm>
3. <https://en.wikipedia.org/wiki/Pac-Man>
4. <https://www.youtube.com/watch?v=qwhXIzNrb9w>
5. <https://github.com/arminkz/Pacman>