2-Agent Cooperative Snake Game

David Antunes
MEIC
Instituto Superior Técnico
Lisbon Portugal
david.f.l.antunes@tecnico.ulisboa.pt

Hugo Ribeiro
MEIC
Instituto Superior Técnico
Lisbon Portugal
hugofribeiro@tecnico.ulisboa.pt

Mariana Serrão
MEIC
Instituto Superior Técnico
Lisbon Portugal
mariana.serrao@tecnico.ulisboa.pt

ABSTRACT

The following paper proposes the development of a multi-agent system based on the popular action game Snake. The game will be adapted to have two intelligent agents control two separate snakes, each with its own respective cherry to pick up on the map, while avoiding collision with either snake's body (which grows with every fruit picked up). The goal of the agents is to collectively pick up as many points (cherries) as possible while also "surviving" for as long as they can. To study this situation, several different agent types and strategies will be employed and empirical results, in the form of total points obtained and number of steps survived, will be collected for each of them. These will then be analyzed and compared, which will allow us to draw conclusions about the adequacy and efficiency of these architectures in the face of this problem/environment. We expect to be able to develop a strategy where the agents communicate the path they intend to take, are able to negotiate if their trajectories intersect and achieve a common ground where both reach their individual goals and avoid collisions.

INTRODUCTION

We all remember Snake, the game that captivated us during our childhood and left a mark on us. For this project we wanted to elevate this game and take it to the next level. We have made the decision to transform the single-player Snake into a multiplayer game. This change not only amplifies the game's difficulty but also adds a twist, as now the agents must communicate and coordinate with each other to evade collisions and achieve the highest possible score. While this may seem like a futile game, coordination between two agents is a wide and very important field of study that provides insights on rational behavior in societies.

During our research for this project we came across 2 articles that caught our attention.The first article explored the use of [deep reinforcement learning in playing Snake](), where the agent continually learns through its interactions with the environment. The author defined the actions and state space, as well as the rewards, for the agent to learn from. After testing, the author concluded that the performance of the agent is influenced by how the state space and rewards are defined. Furthermore, the results demonstrated that the agent always achieved a higher score than a random agent. Although there are similarities, there are also notable differences in our project. In our case, we will be working with 2 agents that need to learn to not only avoid the walls and themselves but also avoid each other. Another huge difference is that our agents will have full observability of the entire environment. These distinctions introduce unique challenges and opportunities for our project, setting it apart from this article.

The second article is about [Multi-agent cooperative pathfinding]() where they propose a new algorithm to avoid collisions.Their algorithm consist in "building a tree that is rooted at the start state and spans towards randomly sampled states from some given state space", after reaching the goal region, the algorithm can follow its edges backwards to obtain the first feasible path. Overcoming this particular challenge will be one of our main objectives, as we aim to enable our agents to achieve the highest score possible. We will explore various approaches to tackle this obstacle in order to find the most effective one.

Now to introduce you to our idea and the problems that arise with it. In our design, we have 2 snakes, each with their own unique "cherry" that will be randomly placed on the map. Apart from these modifications, the rest will work as a normal snake game, both snakes will start small and progressively grow as they eat the cherries. The ultimate objective remains the same: to achieve the highest possible score (total number of cherries captured by both agents). In order to achieve the best performance possible, we need to ensure 2 things: that the agents can communicate with each other so they can find a path that doesn't cause a collision and that both agents are aware of their size so they don't collide with themselves.

Our final goal is to test and compare several different agent types, alongside decision making and conflict resolution strategies, to determine which architecture is more capable of facing the problem at hand and achieving the best results.

APPROACH

An environment is everything in the world that is external to the agent and changes with the agent's actions. It is the agent's source of information both in terms of sensory input and rewards gained.

Since the focus of this project is to develop several different agent architectures and test them against each other, we started off by picking an already made environment that emulates the problem at hand. This consists of a python based 2-player Snake game which can be found in a public repository and was developed by Divya Kustagi. All credits for making this system go to its author(s) as we only intend to change the game logic very slightly in order to make it cooperative instead of adversarial (points from both snakes will be added up) and to put two cherries on the map instead of only one.

Thus the environment we will use consists of an episodic game (when any of the agents collides with a wall or a snake the environment resets) with a 60x60 grid shaped map with two cherries randomly appearing in any of the cells and the two snakes starting with minimal size in a fixed place. The two agents will explore the map and pass over the cherries' locations to collect them, which makes them instantly reappear in a new cell. We can therefore define the environment as static (since it only changes through the actions of the agents), discrete - with a limited number of actions and possible states that are fully determined by the previous state and new agent actions (deterministic). At each time step, the agent "sees" the environment, obtaining information about both snakes' positions and movement direction as well as the cherries' positions. Alongside having full knowledge of the map's boundaries, we can say this is an accessible environment.

The agents that will traverse this environment will be taking actions in the direction of their goal making them proactive. Reactiveness is not an important feature in our problem since we are not working with a dynamic system, but, in some cases, we will want our agents to display social abilities. The agent's possible actions are to change direction to the Left, Upwards, Right or Downwards or to do Noop and keep going forward. The decision making will be an autonomous process that secures the agent's mobility in the map.

Different agent architectures will be tested to determine which is more adequate and overall better at performing the task at hand. We will start with simple random agents, then move on to greedy agents that incorporate some rationality in the way they move in the map as they take the fastest route to reach their fruit. More complex agents that communicate their intentions to others in order to coordinate and negotiate with their partner and follow a common-goal oriented behavior will be the next studied and we intend to try out different conflict resolution mechanisms for when their desired paths intersect and risk collision. Finally, the learning component of agents will be experimented with in a fully reinforcement learning based architecture where the snakes will learn how to move on the map simply through trial and error

EMPIRICAL EVALUATION

To compare the several agent architectures that we will implement we need to have a fixed set of metrics to measure performance in terms of the desired behavior and objective we have for the system. The most effective way to see which architecture is better at any game is to compare the total amount of points obtained, or in this case, the total amount of cherries collected.

It's also important to see how agents react when they are unable to come up with a "winning" situation. For this purpose we want to look at how many time steps the agents were able to survive no matter how many cherries they grabbed. Of course, we also care about movement efficiency so it's very relevant to evaluate how many steps on average it takes for the agents to pick up a cherry. This derived measure can be computed from the two last metrics mentioned.

On a different note, we want to analyze the cause of death/loss for each of the agents' runs. Since we expect rational agents (for example) to die very little from collisions with walls and social agents to be able to avoid collision with another snake more than other architectures. The end states will therefore be recorded and classified as "collision with wall", "collision with the other snake" and "collision with self".

All of these 4 metrics will be captured for a sufficiently large sample of runs in the environment and then averaged out to obtain the expected result for each agent type