6CCS3PRJ 3rd Year

Predator/Prey behaviours in Pacman - Cooperative ghost behaviour using auction mechanism

Final Project Report

Author: Kiem Mark Duc Nguyen

Student ID: 1710305

Supervisor: Jeffery Raphael

Programme: Computer Science BSc

XX/XX/20

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Kiem Mark Duc Nguyen
XX/XX/20

Abstract

This project covers one of the possible uses of Market-based multi-agent communication in a predator / prey scenario. More specifically, this method of multi-agent coordination will be used and tested in the popular video game Pacman, where the ghosts are the multiple predator agents that will communicate and coordinate with each other to capture Pacman acting as the prey. The goal of this project is to simulate predator / prey behaviour using a specific method of market-based coordination, called the Auction method. This method will employ a specific auction type called the English auction in which participants all place an anonymous bid at the same time and the highest bid wins the auction. To be implemented into pacman, each auction participant will be a single ghost, and each bid will be based on each ghosts state. The winning ghost will be the one with the highest possible performance measure - in this case, the ghost with the highest probability of catching pacman - which will hopefully simulate coordination between ghosts and maximise the performance measure of winning the game by catching pacman. This will in turn hopefully pave the way to inspiring the use of market-based approaches in different domains of Artificial Intelligence.

Acknowledgements

I would firstly like to start by thanking

Contents

Chapter 1

Introduction

Multi-Agent Systems (MAS), Multi-Robot Systems (MRS) and Distributed Artificial Intelligence (DAI) is a key area of research in Artificial Intelligence and Robotics industry in recent years. This method of coordinating more than one entity to work together as a team to solve a given problem has many implications that can be vastly superior than utilising a singular entity. This project focuses more on the Multi-Agent Systems side, as Multi-Robot Systems are more focused on real world physical robots navigating a real world environment. Distributed Artificial Intelligence is a more general term which describes a subfield of Artificial Intelligence, whereas Multi-Agent Systems emphasise the use of software Agents, where an agent is described by Jaques Ferber as “a physical or virtual entity that can act, perceive its environment (in a partial way) and communicate with others, is autonomous and has skills to achieve its goals” [1].

1.1 Motivation

Multi-agent coordination is a method of allowing multiple intelligent entities to communicate and coordinate a set of tasks to complete an overall objective as a team. These entities can range from basic factory line robots, to robots with built-in artificial intelligence, to a software agent in a video game. Utilising coordination techniques to allow a team of agents to simultaneously complete a task is, in theory, vastly better than using only a single agent. If the coordination is done right, objectives can be completed many times faster than normal with just a single agent. However there are many challenges to this multi-agent approach. One example is creating an algorithm that is scalable enough that more agents could be added to aid the task, without negatively affecting the performance of the team as a whole. Another challenge is robustness. These algorithms may sometimes need to be designed in such a way that if one or more agents fail to complete a task, the group is not hindered in making progress. The final challenge is that the coordination techniques need to be efficient enough to warrant the use of multiple agents, instead of just one efficient agent.

1.2 Scope

This project aims to explore a specific technique used to coordinate multiple agents. This technique is called Market-Based Multi-Agent Coordination (MBMAC), where more than one agent can communicate with each other to coordinate an ‘Auction’ to distribute a set of tasks. More specifically, an auction method known as the English Auction will be used to coordinate ghost behaviour in a game of Pacman. The objective is to control a team of ghosts to communicate with each other through auctions, which will in turn coordinate predator behaviours to catch Pacman as efficiently as possible. This method will be compared to existing traditional methods of predator behaviour in Pacman, such as the original decision making process of the team of 4 ghosts in the original arcade game. The aim will be to create a technique that is more efficient than this behaviour.

1.3 Project Structure

This report will be separated into clear sections outlining the whole process of the project.

In chapter 2, the background of this topic will be reviewed, such as the motivation of this technique in other pieces of literature. Existing techniques of Market-Based Multi-Agent Communication will be evaluated for their current efficiency, scalability and robustness. Real world applications of will also be reviewed, where current techniques can be used to inform the design of the algorithm used in this project.

Chapter 3 will explore possible ways of implementing Multi-Agent Coordination in the pacman package provided by UC Berkeley. An algorithm for the auction method will be designed formally at first, creating the most efficient method of carrying out Market-Based strategies. Finally, an algorithmic design of the auction method will be integrated into the Multi-Agent Communication technique created in the existing pacman package.

Chapter 4 is where the implementation will be tested and evaluated against existing methods of Multi-Agent Coordination. This Auction Mechanism will be compared against other potential coordination techniques.

In chapter 5 any final conclusions about the project will be discussed and a summary of the whole process will be reviewed. Any future work for the implementation of Market-Based Multi-Agent Coordination techniques will be stated, as well as other potential Multi-Agent solutions.

Chapter 2

Background

2.1 Multi-Agent Coordination

“A team of mobile robots that work in parallel has the potential to finish a given task faster than a single robot.” (Kai M. Wurm [2])

This quote is quite an obvious observation when thought of purely intuitively. This subfield of Artificial Intelligence has been on the forefront of research for a long time. However there are many challenges involved in achieving this seemingly simple task.

Firstly there is the problem of splitting a very general objective into many achievable subtasks that individual entities can easily complete using a divide and conquer technique. The tasks themselves have to be assigned so that they can be achieved concurrently by each entity, as well as the performance of each measured in order to optimise future deployments.

Tasks can be assigned in one of two ways: A centralised approach allows one central entity to assess the state of the whole environment and assign tasks to each sub-entity accordingly, and a distributed approach where all entities have the same priorities, assess their own surroundings try to maximise their own performance measures based on their local knowledge.

The former strategy has the advantage of coming up with a more optimal global solution. This is due to the fact that the central agent will have access to information about the whole state of the environment, then creating a plan based on that information and finally distributing roles. A drawback to this is that the central agent will need to have access to information that is accurate and up to date, which may not always be possible in certain conditions (e.g. Dynamic and/or Stochastic environments). Another downside is that this approach will inherently have a single point of failure: the central agent. This massively reduces the robustness of the system as a whole, compared to the distributed approach.

The latter strategy will be much more robust than the former, as each entity will be able to make decisions independently in real time, without the need for a task allocator. If one agent fails a task, another agent can step in and complete the remaining task, therefore removing the disadvantage of having a single point of failure. These agents will carry out tasks with the objective of maximising their own performance measures with a local solution. However this does not always mean that the sum of the tasks will add up to an optimal global solution, which is one of the downsides of a distributed approach.

The second challenge of multi-agent coordination is making sure that utilising more than one entity actually improves the overall performance of the system. In a situation where it is still possible to use a single agent, if using more than one agent does not provide any additional benefits to performance, there will be a greater cost of running the system, as more entities consequently will require more resources. In the case of multi-robot coordination, 2 robots will require double the amount of resources than 1. If using 2 robots does not increase the performance of completing tasks twice as fast or efficiently, the viability of multi-robot coordination may not be justified. Thus, in the case of multiple software agents, there may not be enough computing power to allow multi-agents to operate at maximum capacity.

The final challenge is that in a multi-agent system, in theory, if 1 agent crashes, the other agents should be able to remedy the situation and finish the task of the crashed agent. However, it is still possible in this case that a failed agent can negatively impact the performance of the remaining agents, therefore negatively impacting the performance of the system as a whole. This performance impact can happen due to a variety of issues. One example is that the crashed agent could be the bridge of communication between two groups of agents, and by crashing, a vital connection point and means of communication has been cut off, severing a path with which to transfer vital data. Another example that could happen with real world multi-robot coordination, is that a crashed agent may physically obstruct other agents wanting to complete their tasks, thus reducing the efficiency of the whole system.

2.2 Auction Method

The auction method is a way of implementing task planning in multi-agent coordination that utilises the premise of real world auctions. There are many different types of auctions available:

- **English Auction** (*Open ascending price auction*): The most common auction type used today. The auctioneer announces a base price and bidders place their bids in ascending price one after the other, where the final highest bidder takes the prize.
- **Dutch Auction** (*Open descending price auction*): The auctioneer announces an initial high price for the items and keeps lowering the price until a bidder claims the items at that price.
- **Blind Auction** (*Sealed First Price Auction*): All bidders simultaneously submit sealed bids so that all other participants do not know each others bids. The winner is the one with the highest sealed bid.

An auction is used specifically for task allocation because in a multi-agent system, each agent takes the role of the bidder, whereas the different tasks are the bidding prizes. The bid of each agent can be measured in many ways. A typical method of measuring the bid sizes of each agent is the performance measure of that agent if that task is allocated to it. This ensures that the agent that will perform the best at that specific task is selected accordingly, therefore maximising the efficiency of the system.

Chapter 3

Market-Based Multi-Agent Communication

3.1 UC Berkeley Pacman Implementation

This project will utilise the pacman package created by UC Berkeley [3], which is a custom remake of the original pacman game developed and released by Namco. The package provides some basic implementation mimicking the behaviours in the original pacman game, however there are a few other additional quality of life extensions included to allow for ease of expansion.

3.1.1 Prey (Pacman)

There are extremely basic implementations of pacman that are included in the package, namely RandomAgent, RandomishAgent and SensingAgent. These are the pacman 'agents' that control pacman differently to each other. As stated in the names, RandomAgent and RandomishAgent makes pacman move in a randomised manner, with no goals whatsoever. SensingAgent is an agent that purely retrieves and prints all of the state information in the running pacman game. Therefore, these are not enough to test the implementations of the predator behaviours later on.

There are two options to go from here:

- Manually play as pacman, which will save time as a pacman agent will not have to be made, however testing will not be as thorough.
- Design a basic pacman agent, which will take up time from designing the ghost agents, however testing will be much more thorough and extensive.

3.1.1 Predator (Ghost)

The ghosts have a much more fleshed out implementation included in the package, namely RandomGhost and DirectionalGhost. RandomGhost moves each ghost agent randomly around the map, and DirectionalGhost allows each ghost to either chase pacman or run away when scared. The latter implementation will be used and adapted for the design of the market-based multi-agent coordination methods.

3.2 Formal Auction Algorithm

The auction algorithm will be split into multiple major parts:

- Frequency of Auctions
- Bid Measurement
- Auctioneer Design
- Task Allocation

3.2.1 Frequency of Auctions

Since decisions in the pacman game are made every game tick, the frequency of auctions can have an impact on the speed of the game itself, as well as the performance of the agents. If an auction is called every game tick, this can massively slow down the running of each game, which will hinder progression in designing and testing the multi-agent system. However, if the auction is not called frequently enough, the ghosts may not be as effective at adapting to its environment and communicating with other ghosts in order to complete their objectives of capturing their prey. Therefore a decision has to be made in order to find the right balance between the two in order to maximise efficiency.

3.2.2 Bid Measurement

Each ghost agent will perform a bid at every auction. These bids have to be based on a metric that is useful and obtainable in the game package. For example, at the most basic level, each bid can be based purely on the distance of each ghost to its prey. The closest ghost should have the highest bid, as it has the highest chance of capturing pacman. However this does not take into consideration many other factors, such as obstacles or whether or not it is in a 'scared' state.

3.2.3 Auctioneer Design

The auctioneer will be created in a similar manner to the GhostAgent abstract class. Here, global variables such as **isAuction** (whether or not an auction is taking place) and **bids** can be defined, which all ghost agents will have access to. This will essentially be the communication aspect of the multi-agent system.

3.2.4 Task Allocation

Finally, once all bids have been placed, each ghost will have to be assigned a task by the auctioneer. These tasks can be as simple as **chase** or **runAway**, however there can be additional tasks that can be implemented which can enhance the performance of the multi-agent system. One of the tasks that can be added is **patrol**, which allows a ghost agent to patrol an area of interest if it is not chasing pacman. This is because it would not be very efficient if the only ghost that gets allocated a task is the winning bidder.

Chapter 8

References

[1] J. Ferber, *Multi-Agent System: An Introduction to Distributed Artificial Intelligence*. Harlow: Addison Wesley Longman, 1999.

[2] K. M. Wurm, *Techniques for Multi-Robot Coordination and Navigation*. Albert-Ludwigs-Universität Freiburg im Breisgau, 2012.

[3] UC Berkeley. (2014, 10. 22). *UC Berkeley CS188 Intro to AI -- Course Materials* [Online]. Available: http://ai.berkeley.edu/multiagent.html

[4]