Timeline to Final Review

December 16, 2020

Following the recommendations of the panel, and in discussion with my supervisors, I have streamlined my research to focus on 3 research questions. The timeline below (Section 6) consists of 4 groups of tasks, one for each research question and one for the literature review, which is ongoing parallel to the other 3 groups. The timeline goes from the last 2 work weeks of 2020 through to the reconvene seminar in April. Work after the seminar will be primarily thesis writeup. Below is a clarification of some concepts from my seminar and an explanation of each group of tasks.

1 Background

1.1 The Learning Process

Each agent is controlled by its own neural network. The network encodes the agent's behaviour i.e. it encodes a function that maps from local observations to actions. The neural network architecture is fixed, but the weights are changed throughout the learning process.

Centralised

In the centralised setup, a learning algorithm creates a population of solutions. A solution is a sequence of neural network weights for the network of each agent on the team. For example, if a network has 50 weights and there are 4 agents, the solution has a length of 200. Each part of the solution is assigned to a random agent. The agents then carry out several episodes, starting from random positions at the nest each episode and acting for a fixed number of time steps. The score is calculated at the end of each episode and it is the sum of rewards for all resources gathered by the team minus the costs paid by all agents. The average score over all episodes is calculated and used to create a new population of solutions. This repeats until the algorithm terminates.

Decentralised

The decentralised setup has changed since the seminar and now uses a technique called cooperative coevolution. In cooperative coevolution, rather than one evolutionary process learning a combined solution for the entire team, each agent on the team has its own evolutionary process learning a solution for only itself. For example, if there are 4 agents, there are 4 evolutionary processes running concurrently. At generation i, each of the 4 processes creates a population of solutions for its agent, where a solution is the weights of a single agent's neural network. The score of a solution at generation i is calculated by pairing it with the best solutions from each of the other 3 agents' evolutionary processes at generation i-1. The score is for that single agent's performance; it is the team reward (total resources collected) divided by the number of agents (regardless of their contribution) minus the cost incurred by that particular agent.

Difference

The difference between centralised and decentralised is that a centralised algorithm uses one learning process to learn a solution for the entire team whereas the decentralised algorithm uses multiple concurrent learning processes to learn solutions for each agent independently that are then combined. Centralised learning rewards the team while decentralised learning rewards the individual.

2 Group 1: Literature Review

I have created a spreadsheet to systematically keep track of the literature and ensure my research is supported by a firm knowledge of the field. Throughout the next few months, I will add relevant papers to the spreadsheet as I delve deeper into each research question, doing brief first passes of each to get an overview and deeper second and third passes for any that are especially relevant.

3 Group 2: Landscape Analysis

3.1 Creating a Fitness Gradient

Bootstrapping and deception are inherent to the evolutionary approach [?, ?]

"The more complex the task, the more susceptible evolution is to bootstrapping issues and being trapped in local optima" [?]

Bootstrapping is "when the task is too demanding for the fitness function to apply any meaningful selection pressure on a randomly generated population of initial candidate solutions" [?]

"the bootstrapping problem is often caused by the gap between the design objective and primi- tive capabilities of the controller, which makes it a challenging task to devise fitness functions applying selective pressure towards better solutions in early generations, and therefore, prevents the evolutionary process from starting" [?]

Solutions are to insert human knowledge or to promote diversity. Insert human knowledge by using incremental evolution (solve steps towards solution), decompose behaviours (evolve sub-controllers then evolve combined controller) or

Inserting human knowledge reduces potential for automation [?]

Danger of the experimenter introducing "negative biases" [?]

The lack of gradient might be inherent to specialisation problems and complex problems in general where there is a high-level goal and agents can only do low-level behaviours. Creating a gradient means either creating more sophisticated component behaviours that can be combined to solve the problem (e.g. 'go to source' and 'go to nest', rather than 'move forward 1 step' and 'move backward 1 step') or identifying what I believe are milestones towards solving the problem and rewarding them, which biases the learning to what I think is a good solution. Overall, adding a gradient is 'cheating', in a sense, and my findings may not generalise to other specialisation problems if I do this.

3.2 Analysing the Landscape

A shortcoming of my landscape analysis that was identified at the seminar is the lack of insight into the degree of specialisation of the discovered solutions. To include this information in the plots, I will first need to identify a suitable metric that quantifies specialisation and implement it in the code. Once this is done, I can reproduce the previously done landscape analysis and then repeat it with different slope angles. This will tell us how the landscape changes when there is increased/reduced pressure for specialisation to emerge.

Additionally, there are two interesting sub-questions:

- What are the features of the landscape for an individual agent in the decentralised setup?
- What are the features of the landscape when the slope foraging environment is made more difficult (e.g. by adding obstacles)?

These are not strictly necessary to answer the question, however, so will only be investigated if time permits.

4 Group 3: Centralised vs Decentralised

For this question, I am focusing on the scalability of each setup with the team size and the robustness of each setup to changes in the team members (i.e. what happens to performance and specialisation when some of the evolved team members are removed or replaced). This will use the new metric for degree of specialisation developed for the previous question and will also require some fixes to the code. The key fixes to the code are:

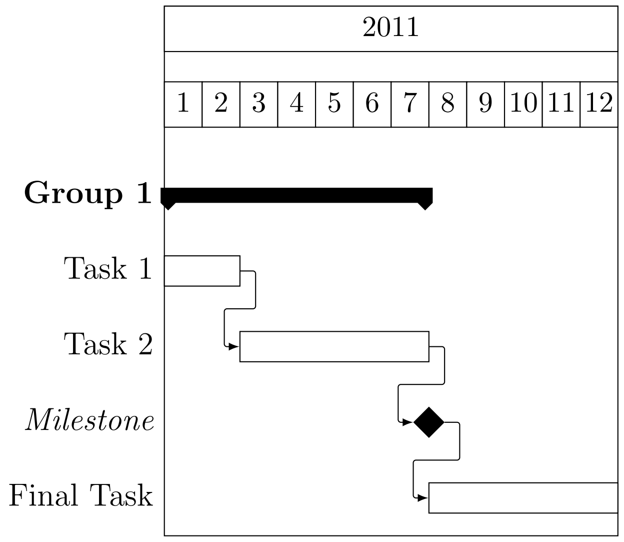
- Seeding each evolutionary run from a unique starting point (to remove bias)
- Adding multi-threading to further speed up experiments (if it is straightforward to do so)
- Implementing cooperative coevolution to more faithfully represent the idea of decentralisation

5 Group 4: Price of Anarchy

The purpose of this task is to complement the experimental results with a theoretical model that can explain them. This game-theoretic model will help interpret the differences in performance and specialisation between the centralised and decentralised setups.

Completing this group of tasks involves formulating a game to model the slope foraging problem, studying the dynamics and using the findings to interpret the problem. The toolset for completing these tasks is available and only needs to be applied.

6 Timeline


Group 1
Task 1
Task 2
Milestone
Final Task