MAS Mini Project Documentation

[1]Arjun Verma : IMT2017008
[1]Kaustubh Nair : IMT2017025
Multi-Agent Systems
MAS_MiniProject

Abstract. *The following documentation captures our attempt to solve the MAS Mini Project and provides a reasoning for the solution proposed for each of the tasks. The implementation can be found on Github.*

1. Introduction

While modelling a solution for the problem proposed, we solely worked towards adhering to the conditions stated in the specification:

- No imperative statements are given to any of the turtles.
- Given a set of choices, the agents compute a payoff for each of the choices and make a choice based on a strategy.
- The program should not contain any leader agent.

Keeping the above conditions in mind, the solution we propose relies heavily on the concept of **terrain modelling**. Although tampering with the environment seems like playing God, but what better way than introducing incentives in your environment to instill reasoning into your agents to act a particular way. By bringing in incentives into the environment, each turtle acts autonomously by applying a strategy to obtain the incentive. In our model, each patch has a tuple of scores associated with it, (a tuple because we score the x-coordinate and the y-coordinate separately) and the strategy that the turtles apply in our solution is of maximizing it's payoff.

We begin by describing a few of the terms and variables used in the program and then go on to describe the solution for each of the task individually and providing a real life scenario that explains the solution proposed.

2. Terms and Variables

- We've created 2 named breeds, **redturtles** and **blueturtles**, and an individual agent from the breed is referred to by, **redturtle** and **blueturtle**, respectively.

- <u>**Turtle variables**</u>

  These account for the autonomic behaviour of the turtles.

  - **self_interest:** As the term states, it captures the overall net self-benefit of the turtle. It is this variable that the turtle aims to maximize.

- **potential_gain:** This is the benefit that a turtle gets, on reaching a certain patch.

- **potential_gain_clustering:** This is the benefit that a turtle gets from being as close as possible to a point on the grid which is randomly selected (This was introduced to give the turtles an incentive to form a compact block).

- **potential_gain_grouping:** This is the benefit that a turtle gets, from having turtles in it's neighbourhood. The potential_gain_grouping is actually set as (**potential_gain_samegroup - potential_gain_diffgroup**), where **potential_gain_samegroup** is the potential gain from having turtles of the same breed in it's neighborhood and **potential_gain_diffgroup** is the potential gain from having turtles of a different breed in it's neighborhood. This will be later elaborated upon in the section explaining Task 3.

• **<u>Patch variables</u>**

These account for the incentives to be generated in the terrain.

- **cost_function_x:** This is the function that allots a score to the patch based on it's x-coordinate.

- **cost_function_y:** This is the function that allots a score to the patch based on it's y-coordinate.

- **cost_function_clustering:** This is the function to assign a score to the patch to account for clustering among agents. (Generates incentives in such a way so as to bring the turtles close to a random point.)

3. Tasks

We now provide images of the outputs of our model, for each of the task specified, along with a real life scenario that could depict each of them.

3.1. Task 1

Problem statement: For the first task, get the turtles to form two lines– one horizontal and one vertical, anywhere in the space

Output Image:

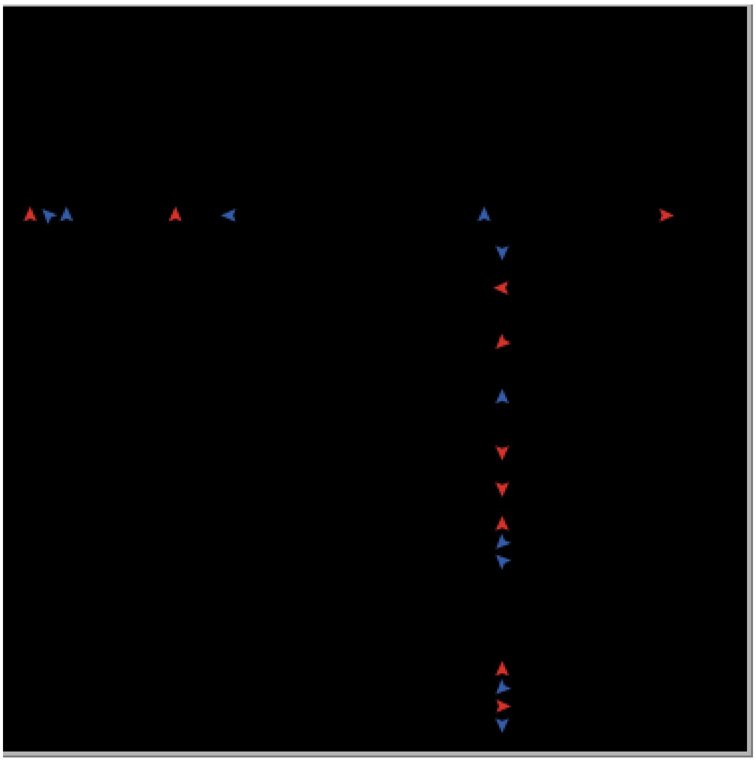

Figure 1. Task 1

Explanation : For Task 1, what our model does is that it randomly chooses a horizontal and a vertical line in the grid space and assigns all the patches on these lines a very high score (equal for all the patches on these lines) and the rest of the patches on the grid are assigned scores based on their proximity to these high score patches. The turtles roam around the grid picking up scores (maximizing their self_interest) and eventually end up on the high score patch.

Real life scenario: The above solution can be analogous to the situation where, let's say we have a bunch of "hungry" agents, and to get them to form a line, all we do is set up "plates of food" on a random horizontal and vertical line on the grid.

3.2. Task 2

Problem statement: For the second task, get the turtles to form either two vertical lines or two horizontal lines, where each line comprises of agents of the same colour

Output Image:

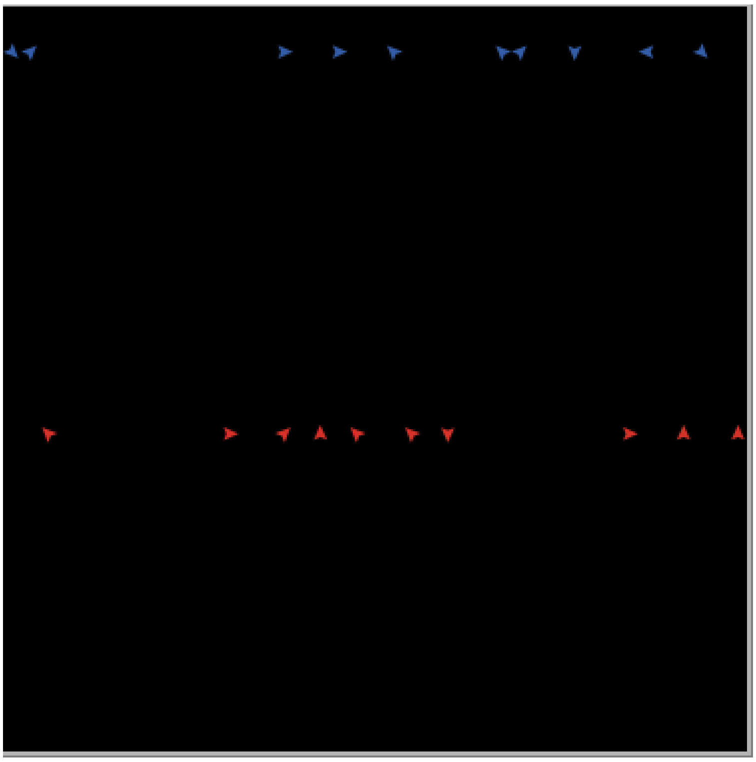

Figure 2. Task 2

Explanation : For Task 2, what our model does is essentially the same thing that was done for Task 1, with the only change that it assigns separate high scores for each breed, i.e , it randomly chooses two horizontal or vertical lines in the grid space and assigns all the patches on these lines a very high scores (which are different for different breeds) and the rest of the patches on the grid are assigned scores based on their proximity to these high score patches. The turtles roam around the grid picking up scores (maximizing their self_interest) and eventually end up on the high scoring patches.

Real life scenario: The above solution basically introduces the concept of preferences among breeds and we draw the same analogy as in Task 1, only this time the ”hungry” agents have an innate set of preferences, and to get them to form a line, all we do is set up their preferred ”food plate” on the random horizontal or vertical line on the grid. Such preferences among a group are quite prevalent in nature, and can be seen through a very simple example of animals categorized as ”herbivores” and ”carnivores”, who prefer a certain set of choices as the food they consume.

3.3. Task 3

Problem statement: For the third task, get the turtles to form a compact block, such that each turtle has at least three other turtles of the same colour as them, in their immediate neighbourhood

Output Image:

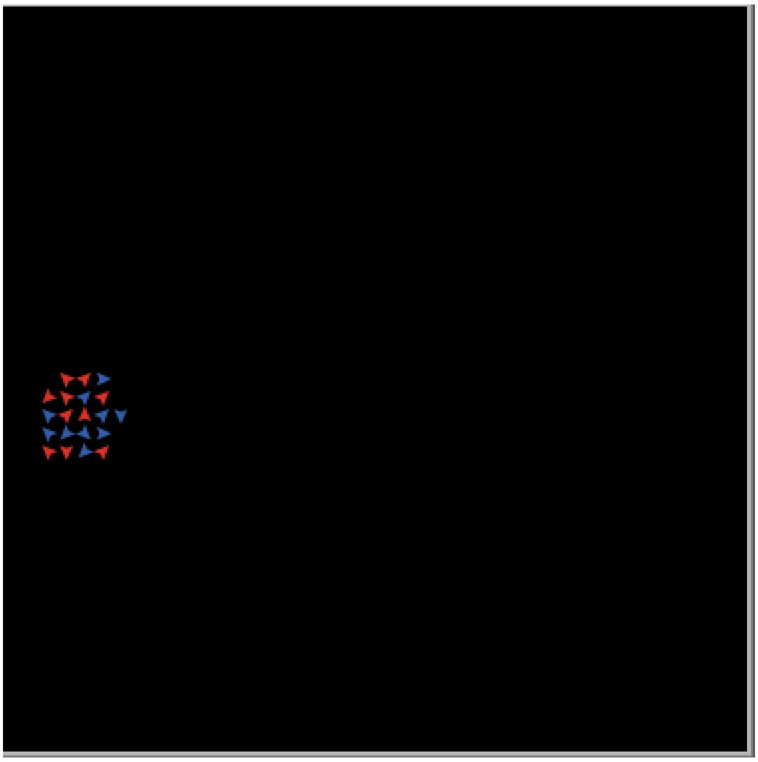

Figure 3. Task 3

Explanation : For Task 3, what our model does is that it randomly chooses a single patch and assigns it a very high score value. All other patches are assigned scores based on their proximity to this patch. This is an incentive for the turtles to come together and form a compact block. This is captured by the variable **potential_gain_clustering**. In addition to this, we have used the variable **potential_gain_grouping** to capture the benefit from having at least three other turtles in their neighborhood. So what we have done is that at each movement, the turtle looks for other turtles in their immediate neighborhood (the 8 adjacent patches), and assigns a score accordingly. It does not matter in which of the 8 available patches the turtles are as all the patches are scored equally. The only thing that matters is the breed of the turtles. We add up the scores/potential_gain due to each of the breeds in the immediate 8 space neighborhood and the turtles of the same breed are summed up under **potential_gain_samegroup** and turtles of the other breed are summed up under **potential_gain_diffgroup**. We now calculate the net potential_gain due to grouping/neighborhood as :

$$\textbf{potential\_gain\_grouping = potential\_gain\_samegroup - potential\_gain\_diffgroup} \tag{1}$$

As we can see, we now have two quantities: **potential_gain_clustering** and **potential_gain_grouping** to maximize for each of the sub-task, i.e, grouping and clustering. (Note, the scores have such been assigned such that the two scores for the sub-tasks are comparable.) The net potential_gain is then taken as a scaled sum of the two scores in a 40:60 ratio. (different ratios were experimented with and these seem to be giving the best possible solution.)

4. Conclusion and Remarks

In this section, we would first like to begin with by providing two general justifications, which hold true for all the three tasks above:

- *Justification for the movement of the turtles:* The concept of modelling the terrain so as to score patches to tempt the turtles to form a line was to build upon a scenario, where let's say that there are certain artefacts spread across the grid and each turtle can obtain at most one artefact. Thus the turtles move around by picking up scores/artefacts from the grid and when they come across a patch which holds a artefact of higher value, they drop the current score/artefact and replace it with the higher score, eventually moving towards the artefact of highest value.

- *Justification for the stationary rotation of the turtles:* Although it might be not that clearly evident from the images above, the turtles upon reaching the patch with the highest score (their position in line), do not become completely stationary but are rotating around in their respective patches. This was left to be so as to not provide any imperative stop statements but also to show that the turtles are on the constant lookout for a patch of a higher value to move on to but choose not to leave the current patch as it is the patch with the highest value and keep on rotating there as they are unaware of the global knowledge of the scores of all the patches which would otherwise tell them that the patch they are on is indeed the patch with the highest possible score in their vicinity.

Personal remarks about our model:

- *Positive Remark:* Using the concept of terrain modelling, we were more or less able to complete all the three tasks without violating any of the conditions mentioned in the problem statement.

- *Negative Remark:* Using terrain modelling, seems too restrictive an approach as it is essentially playing God and introducing too many modifications to the environment.

With this, we would like to conclude our work. Future modifications to the model involve working upon lowering the restrictions imposed as mentioned above in the Negative Remarks' section.