eman ta zabal zazu

Universidad del País Vasco / Euskal Herriko Unibertsitatea

EAP MASTER

FACULTY OF ECONOMICS

Pedro Iraburu Muñoz

EVOLUTIONARY GAME THEORY:

altruistic behavior

22th of February, 2020

Bilbo

Are altruistic behaviors dominant strategies?

Pedro Iraburu Muñoz

22/02/2020

Abstract

This short project will analyze the importance and the consequences of altruistic behaviors within species, from a game theory perspective. For this, I will find the Nash Equilibria of a very simple game where two traits (cooperation and aggressiveness) compete with each other.

Keywords: Evolution, game theory, MESA.

Introduction

What is Evolutionary Game Theory? EGT is just the field of Game Theory that studies biological changes in populations. The main objective of this is discovering the behavior of a biological or cultural trait: how does the state variable evolve over time (Newton, 2018). This is one of the main differences between the classical game theory and EGT, the focus is not on the NE but on the behavior of this key state variable, changed by survival of the fittest, imitation or optimization.

The definition was born in 1973, by the hands of John Maynard Smith, with his famous paper, ”The Logic of Animal Conflict”, where he and other authors analyzed conflicts between members of the same species, concluding that they avoided ”serious injuries”, explained as due to group selection for behavior benefiting the species, rather than individuals (Maynard and Price, 1973).

Objective: The objective of this short article is creating a model suited for simulation that would get similar conclusions to what states the literature on the subject (altruistic behavior within species). This model albeit simple in nature, tries to emulate the behavior of two distinctive traits within a specie: aggression and altruism; and the evolution of the population of each one.

Process: In order to do this an ABM (Agent Based Model) is needed. An Agent Based Model is one where the individuals play an important part in its characterization. Above the numerous types of ABM, this one is a modular, python based model in which you need to program each component, including the analysis and visualization (using a server with a javascript interface). As it is quite a lot of work to just simulate the model, I will outline a ”very buggy” code that shows how the agents interact with each other. This can be visualized with the tools Mesa offers us, like a local server that shows dynamically the evolution of each population. All this information can be checked in the official webpage: `https://mesa.readthedocs.io/en/master/overview.html`

Theoretical solution

A very simple Game The model is based on a very famous game: the hawk-dove conflict (Maynard and Price, 1973), where animals fight for survival in a series of rounds. In this game, there are two different traits within a species: aggression, portrayed by ”Hawks”, and altruism, portrayed by the ”Doves”. The first ones will fight for food, taking 75% of it, while the non aggressive ones will share one half. We will assume, for the sake of simplicity[1], that the food is valued in 2 ”energy units”. At the end of a round, everyone will loose 1 energy point. If any individual is left with 0 energy, it will die in the next round. However, if any animal had excess energy after the end of a round, it could reproduce, transmitting the same trait to its youngling. Again, for easing the model, I will assume they grow instantly from newborn to adult.

I will proceed to describe the interactions between the two groups when they get to a piece of food, that I will later implement in the ABM model:

- If a dove encounters another dove, they will share the food (1 unit each).
- If a hawk encounters a dove, it will fight, getting 1.5 units of the food and leaving the dove with just 0.5 units
- If two hawks encounter, they will waste all the energy by confronting each other.

Analyzing from a game theory perspective, let’s find the Nash Equilibria, where each strategies is a best response to all of the others:

Table 1: Representation of the hawk/dove game in normal form

	D	H
D	1.0, 1.0	0.5, 1.5
H	1.5, 0.5	0.0, 0.0

[1]Also, this is the way it is coded in MESA.

Assuming we know the strategy of the second player, hawk, which alternative would maximize our chances of surviving? The dove strategy. Consequently, if the other player were to play as a dove, the best response answer would be behaving like a hawk. So, there are two Nash Equilibria, each one where the individuals play the opposite of the other. Let's suppose there is a proportion of doves of 70%, on average, the payoff would be:

$$D = d * 1 + (1 - d) * 0.5, \tag{1}$$

$$H = d * 1.5 + (1 - d) * 0, \tag{2}$$

with $d = 0.7$ being the probability of encountering a dove. Solving this simple equation, we get that each strategy yields, on average: $D = 0.85$ and $H = 1.05$. Playing Hawk is the best response to this situation, as there are more doves, and the probability of encountering another hawk, is low. Following this process, and solving to find the equilibrium $D = H$, the proportion of doves to hawks is 50%[2]. Now I will proceed to explain how I outlined the simulation of the model. The results we expect is that in the long run, the proportion of both traits will be equal.

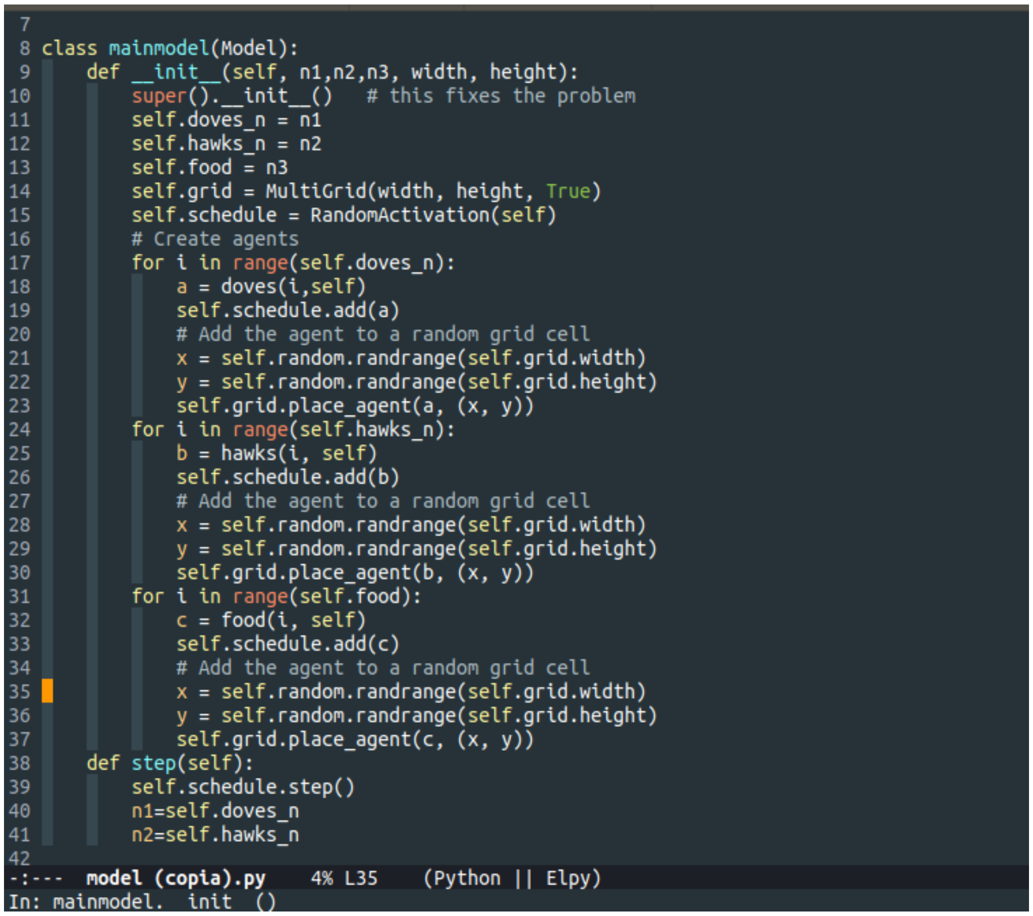
[2] If the outcomes are changed, so will this number.

Simulation

For the simulation, I will use MESA, a python based ABM, where individuals play an important role on its characterization. It is modular, so you can change each part of your model at will: agents, tools, etc. In this assignment, I will use the ”server” tool, where we can see in real time how the individuals behave.

First of all, we need to define the model. In the next piece of code, I will initialize the model, defining all the inputs it will use. Beginning at line 16 until the 37th, the agents are individually created and added to a random grid cell of the representation we saw in the first figure. At the end, the step function is defined, it sets the schedule, searching for each step function the agents use.

Figure 1: Simulation modules, the model

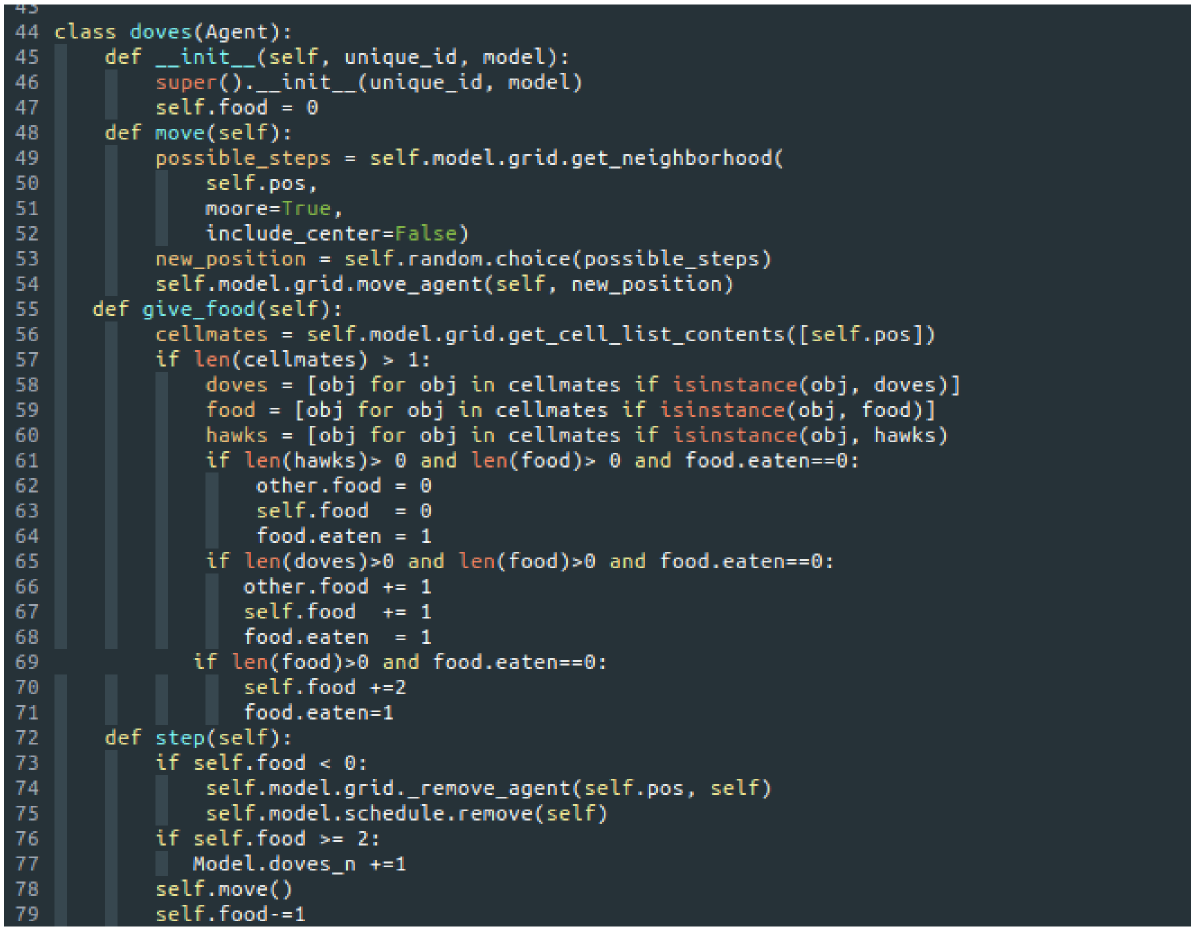


```
 7
 8 class mainmodel(Model):
 9     def __init__(self, n1,n2,n3, width, height):
10         super().__init__()   # this fixes the problem
11         self.doves_n = n1
12         self.hawks_n = n2
13         self.food = n3
14         self.grid = MultiGrid(width, height, True)
15         self.schedule = RandomActivation(self)
16         # Create agents
17         for i in range(self.doves_n):
18             a = doves(i,self)
19             self.schedule.add(a)
20             # Add the agent to a random grid cell
21             x = self.random.randrange(self.grid.width)
22             y = self.random.randrange(self.grid.height)
23             self.grid.place_agent(a, (x, y))
24         for i in range(self.hawks_n):
25             b = hawks(i, self)
26             self.schedule.add(b)
27             # Add the agent to a random grid cell
28             x = self.random.randrange(self.grid.width)
29             y = self.random.randrange(self.grid.height)
30             self.grid.place_agent(b, (x, y))
31         for i in range(self.food):
32             c = food(i, self)
33             self.schedule.add(c)
34             # Add the agent to a random grid cell
35             x = self.random.randrange(self.grid.width)
36             y = self.random.randrange(self.grid.height)
37             self.grid.place_agent(c, (x, y))
38     def step(self):
39         self.schedule.step()
40         n1=self.doves_n
41         n2=self.hawks_n
42
-:---  model (copia).py    4% L35    (Python || Elpy)
In: mainmodel.__init__()
```

Source: python code, MESA.

Here we have an example of an agent, the doves. In the first lines I initialize the agent module and set the function that describes how it moves, in this case, randomly. After this, we can find two other functions, one that describes the beahavior of the agent regarding food ($give\_food$), and the step function, which sets the behavior of the agent inside the model.

Figure 2: Simulation modules, the first agent

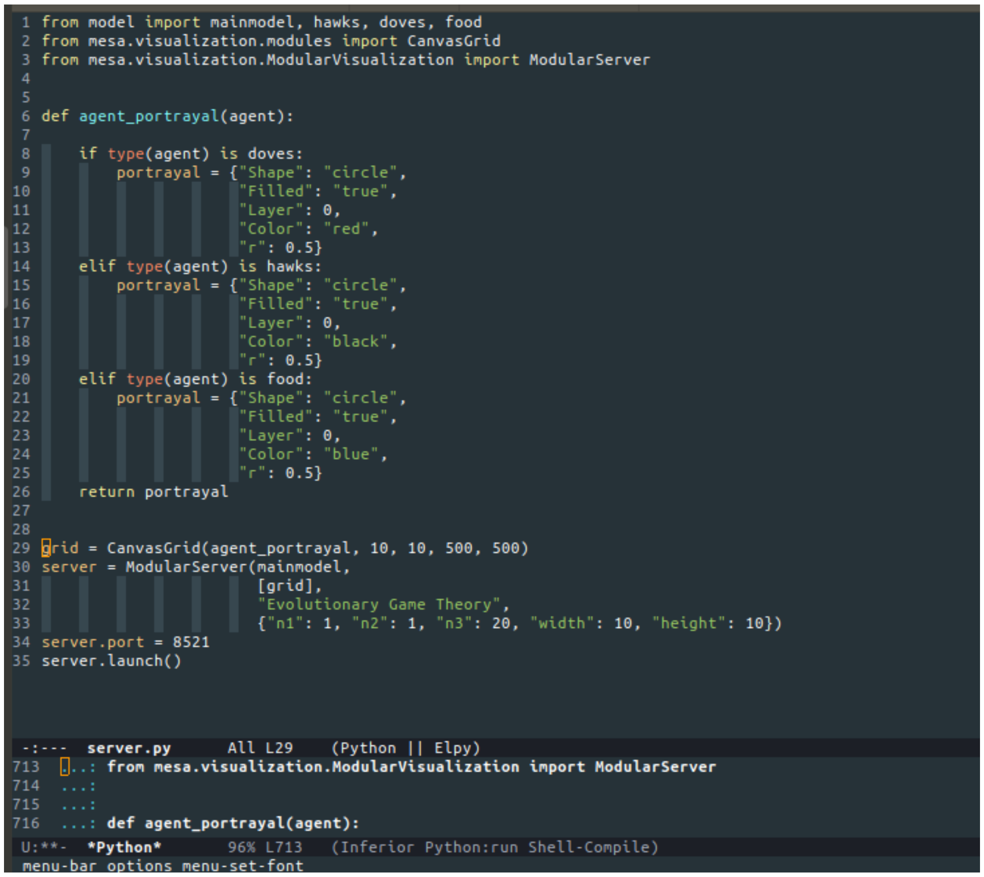


```python
class doves(Agent):
    def __init__(self, unique_id, model):
        super().__init__(unique_id, model)
        self.food = 0
    def move(self):
        possible_steps = self.model.grid.get_neighborhood(
            self.pos,
            moore=True,
            include_center=False)
        new_position = self.random.choice(possible_steps)
        self.model.grid.move_agent(self, new_position)
   def give_food(self):
        cellmates = self.model.grid.get_cell_list_contents([self.pos])
        if len(cellmates) > 1:
            doves = [obj for obj in cellmates if isinstance(obj, doves)]
            food = [obj for obj in cellmates if isinstance(obj, food)]
            hawks = [obj for obj in cellmates if isinstance(obj, hawks)
            if len(hawks)> 0 and len(food)> 0 and food.eaten==0:
                other.food = 0
                self.food  = 0
                food.eaten = 1
            if len(doves)>0 and len(food)>0 and food.eaten==0:
               other.food += 1
               self.food  += 1
               food.eaten  = 1
           if len(food)>0 and food.eaten==0:
               self.food +=2
               food.eaten=1
    def step(self):
        if self.food < 0:
            self.model.grid._remove_agent(self.pos, self)
            self.model.schedule.remove(self)
        if self.food >= 2:
           Model.doves_n +=1
        self.move()
        self.food-=1
```

Source: python code, MESA.

For the final part of the analysis, I will describe the server code. It is a separate python file where, after importing the libraries, we set the portrayal of the agents in the real time visualization: hawks are black, doves are red, food is blue. In the last lines of code, I create the grid and choose the port I want to run the server in.

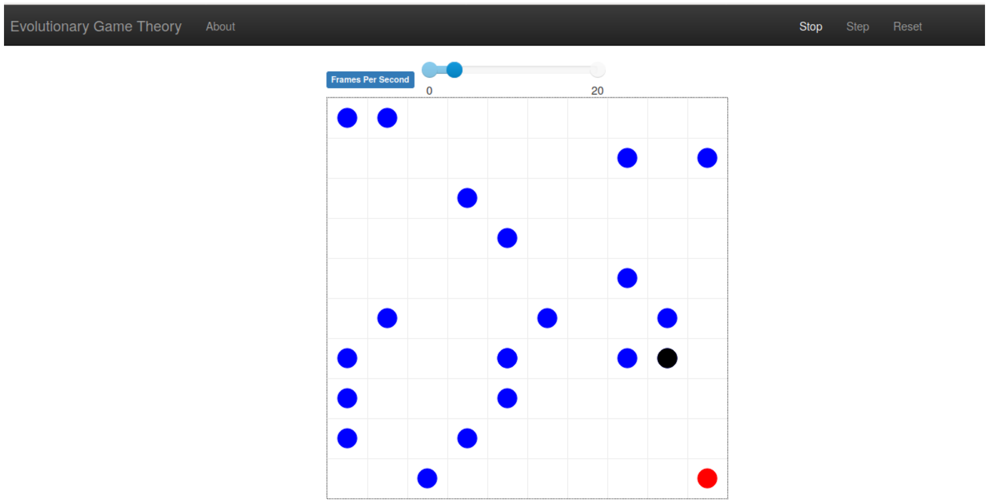
Figure 3: Simulation modules, the server



```
from model import mainmodel, hawks, doves, food
from mesa.visualization.modules import CanvasGrid
from mesa.visualization.ModularVisualization import ModularServer


def agent_portrayal(agent):

    if type(agent) is doves:
        portrayal = {"Shape": "circle",
                     "Filled": "true",
                     "Layer": 0,
                     "Color": "red",
                     "r": 0.5}
    elif type(agent) is hawks:
        portrayal = {"Shape": "circle",
                     "Filled": "true",
                     "Layer": 0,
                     "Color": "black",
                     "r": 0.5}
    elif type(agent) is food:
        portrayal = {"Shape": "circle",
                     "Filled": "true",
                     "Layer": 0,
                     "Color": "blue",
                     "r": 0.5}
    return portrayal


grid = CanvasGrid(agent_portrayal, 10, 10, 500, 500)
server = ModularServer(mainmodel,
                       [grid],
                       "Evolutionary Game Theory",
                       {"n1": 1, "n2": 1, "n3": 20, "width": 10, "height": 10})
server.port = 8521
server.launch()



-:---  server.py       All L29    (Python || Elpy)
713  ...: from mesa.visualization.ModularVisualization import ModularServer
714  ...:
715  ...:
716  ...: def agent_portrayal(agent):
U:**-  *Python*        96% L713   (Inferior Python:run Shell-Compile)
menu-bar options menu-set-font
```

Source: python code, MESA.

Figure 4: Representation of the species in real time



Source: python code, MESA.

Conclusion

Are altruistic behaviors really dominant strategies? Well, it depends on the proportion of aggressive and unselfish individuals, as we have proven in the theoretical analysis. On the long run, there will be a half of each trait inside the population of this synthetic species. The model simulation would help us analyzing the variable (even though we know what the proportion on the long run is): how it changes over time, how elastic ist is, what would happen if we had only doves in the first period, and we introduced a hawk after a few rounds, etc.

Studying the existence of both traits, aggression and altruism, is key to understand why are they present in many species, including us, and what function do they carry in a social/evolutionary perspective. For example, from an evolutionary biological point of view, altruism hinders one individual expected number of offspring, but it enhances the number other organisms within the species is likely to produce (Okasha, 2013).

References

[1] Newton, J (2018): *"Evolutionary game theory: A renaissance"*, retrieved from https://www.econstor.eu/bitstream/10419/179191/1/games-09-00031-v2.pdf.

[2] Maynard Smith, J & Price, G.R (1973): *"The logic of Animal Conflict"*, retrieved from https://www.semanticscholar.org/paper/The-Logic-of-Animal-Conflict-Smith-Price/65f00fc243eb98d83c2e2a76f867bb4d1d3be9d5.

[3] MESA. (2016): *"MESA Overview"*, retrieved from https://mesa.readthedocs.io/en/master/overview.html.

[4] Okasha, S. (2013): *"Biological Altruism"*, retrieved from https://plato.stanford.edu/entries/altruism-biological/.