Progetto dei Corsi di

Complex Systems and Network Science

Corso di Laurea Magistrale in Informatica

A.A. 2021-2022

Dipartimento di Informatica — Scienza e Ingegneria
Università di Bologna

Ozalp Babaoglu   Angelo Trotta

November 23, 2021

1 General Information about the Project

As part of your course requirement, you are to complete the project described below, which must be carried out individually. Submission of the project for evaluation must be done via email to the address: `angelo.trotta5@unibo.it`.

The deadline for submission is **23:59:59 hours on 04 January, 2022**. The email must have the subject field as *CSNS Project 2022* and must be sent from your University address (`name.surname@studio.unibo.it`).
You will receive a confirmation message within a few days of your submission. The email should contain an archive (in .zip or .tar.gz format) containing the following:

1. The source code that was developed (either in NetLogo or PeerSim);
2. A short paper, in PDF format, describing the model that was implemented, the experiments that were carried out using it, and a discussion explaining the results that were obtained.

Your full name, email address and student ID number (matricola) must be included in all of the source files, in the paper, and in the submission email that you send. The source code should be well documented and formatted, following good programming practices. The paper can be written in Italian or in English, and should be structured like a technical paper, thus containing a title, abstract and bibliography. It is strongly suggested that you limit the length to 16 pages and that you follow the Springer format for *Lecture Notes in Computer Science* (LNCS). Templates are available for both Word[1] and LaTeX[2]. You can use any text processing system you prefer (even though LaTeX is suggested) to write the paper as long as you submit the result as a PDF file.

The project must be done *individually*: no sharing of papers or source code is permitted. You are, of course, encouraged to discuss issues and solutions with fellow students or with the instructors.

2 Grading Policy

For your project to be satisfactory, it must satisfy the following requirements:

- The project must implement the specifications that follow. You are allowed (and encouraged) to apply modifications and extensions to the project, but they must be proposed to the instructors beforehand and approved by them.
- The model's implementation, and all of the related simulations and experiments, must be carried out using either the NetLogo or PeerSim software systems. If PeerSim is chosen, the cycle-driven simulation engine should

[1] Link to .doc template
[2] Link to .tex template

be used, and the simulator must be configurable by means of the standard PeerSim configuration file.

- Your paper must thoroughly describe the model that was implemented and justify all significant design decisions and extensions that were applied to it. You should also discuss the expected behavior of the model, by making previsions. Most importantly, you have to explain the experiments that you performed in terms of methodology and the results that you obtained. Significant implementation details can be inserted, if important in the context of the model, but should otherwise be kept as comments in the code itself.

You are encouraged to focus on a simple model and to apply extensions to it only if you completely understand the behavior of the base model. This can be achieved by working in modular fashion, thus incrementally (and carefully) adding new features, enriching your model. Ending up with a complex, unpredictable and difficult to understand model is very easy. On the contrary, you should prove through your experiments that you fully understand the behavior of your model and that you can interpret the results you obtained, and are able to relate them with real-world phenomena. Finally, you should try to find tipping points or interesting equilibrium states in your model.

If you are interested in these topics (e.g. you want to build better models or study other systems of this kind), do not hesitate to contact us when looking for a thesis topic.

3 Introduction to the Project

The purpose of this year's project is to study, implement and analyze evolutionary models of social interactions in finite populations [1] using the *Evolutionary Game Theory (EGT)*. EGT is a branch of a more general discipline called *Game Theory*.

3.1 Game Theory

Game theory is a discipline devoted to studying social interactions where individuals' decisions are interdependent, i.e. situations where the outcome of the interaction for any individual generally depends not only on her own choices, but also on the choices made by every other individual. Thus, several scholars have pointed out that game theory could well be defined as 'interactive decision theory'. Interactive social interactions are modeled in game theory as *games*. A game is an abstract representation of a social interaction which is meant to capture its most basic properties. In particular, a game typically comprises:

- the set of individuals who interact (called players)
- the different choices available to each of the individuals when they are called upon to act (named actions or pure strategies)

- the information individuals have at the time of making their decisions
- a payoff function that assigns a value to each individual for each possible combination of choices made by every individual. In most cases, payoffs represent the preferences of each individual over each possible outcome of the social interaction, though there are some evolutionary models where payoffs represent Darwinian fitness.

Game theory has nowadays various branches. Historically, the first branch to be developed was *Traditional Game Theory (TGT)*. TGT is also the branch where most of the work has been focused. In TGT, payoffs reflect preferences and players are assumed to be rational, meaning that they act as if they have consistent preferences and unlimited computational capacity to achieve their well-defined objectives. The aim of the discipline is to study how these instrumentally rational players would behave in order to obtain the maximum possible payoff in the formal game. A key problem in TGT is that, in general, assuming rational behavior for any one player rules out very few actions –and consequently very few outcomes– in the absence of strong assumptions about what players know about others' rationality, knowledge and actions. Hence, in order to derive specific predictions about how rational players would behave, it is often necessary to make very stringent assumptions about everyone's beliefs and their reciprocal consistency. If one assumes common knowledge of rationality and consistency of beliefs, then the outcome of the game is a Nash equilibrium, which is a set of strategies, one for each player, such that no player, knowing the other players' strategies in that set, could improve her expected payoff by unilaterally changing her own strategy. An equivalent definition is the following: A Nash equilibrium is a strategy profile (i.e. one strategy for each player in the game) where every player is best responding to the strategies of the others. Oftentimes, games have several Nash equilibria. These aquilibria can be defined by *pure strategies* or *mixed strategies*, which means that players choose each action with a certain probability.

3.2 Evolutionary Game Theory

After the emergence of traditional game theory, biologists realized the potential of game theory to formally study adaptation and coevolution of biological populations, particularly in contexts where the fitness of a phenotype depends on the composition of the population. The main elements of the game became:

- **Players** are assumed to be pre-programmed to play one given strategy, i.e. players are mere carriers of a particular fixed strategy that had been genetically endowed to them and could not be changed during the course of the player's lifetime.
- **Strategies** are not assumed to be selected by players, but rather hardwired in the agents' genetic make-up. Strategies were, basically, phenotypes.

- Since strategies are not consciously chosen by players, but they are simply hardwired, **information** at the time of making the decision plays no significant role.
- **Payoffs** did not represent any order of preference, but Darwinian fitness, i.e. the expected reproductive contribution to future generations.

The main assumption underlying evolutionary thinking was that strategies with greater payoffs at a particular time would tend to spread more and thus have better chances of being present in the future.
The key insight that game theory contributed to evolutionary biology is that, once the strategy distribution changes as a result of the evolutionary process, the relative fitness of the remaining strategies may also change, so previously unsuccessful strategies may turn out to be successful in the new environment, and thus increase their prevalence. In other words, the fitness landscape is not static, but it also evolves as the distribution of strategies changes.

Evolutionary ideas proved very useful to understand several phenomena in many disciplines, but it became increasingly clear that a direct application of the principles of *Darwinian* natural selection was not always appropriate for the study of (*non-Darwinian*) social evolution. In many contexts, it seems more natural to assume that players are capable of adapting their behavior within their lifetime, occasionally revising their strategy in a way that tends to favor strategies leading to higher payoffs over strategies leading to lower payoffs. Hence, in non-Darwinian systems, the canonical evolutionary model typically comprises the following elements:

- a population of agents,
- a game that is recurrently played by the agents,
- a procedure by which revision opportunities are assigned to agents,
- a revision protocol, which dictates how individual agents choose their strategy when they are given the opportunity to revise. As *revision protocol* we introduce the **imitate the better realization** that reads as follow: look at another (randomly selected) agent and adopt her strategy if and only if her payoff was greater than yours. Other revision rules can be found in literature [1]

Note that this approach to EGT can formally encompass the biological interpretation, since one can always interpret the revision of a strategy as a death and birth event, rather than as a conscious decision. Having said that, it is clear that different interpretations may seem more natural in different contexts. The important point is that the framework behind the two interpretations is the same.

3.2.1 Noise and initial conditions

The previous model can be improved by adding two features that can be crucial in the execution of a EGT game:

- The possibility of setting initial conditions explicitly. This is an important feature because initial conditions can be very relevant for the evolution of a system.
- The possibility that revising agents select a strategy at random with a small probability. This type of noise in the revision process may account for experimentation or errors in economic settings, or for mutations in biological contexts. The inclusion of noise in a model can sometimes change its dynamical behavior dramatically. This is important because dynamic characteristics of a model that are not robust to the inclusion of small noise may not correspond to relevant properties of the real-world system that we aim to understand. Besides, as a positive side-effect, adding small amounts of noise to a model often makes the analysis of its dynamics easier to undertake.

3.2.2 Spatial interactions

In models with *spatial structure*, agents do not interact with all other agents with the same probability, but they interact preferentially with those who are nearby. More generally, populations where some pairs of agents are more likely to interact with each other than with others are called *structured* populations. This contrasts with the random matching models, where all members of the population were equally likely to interact with each other. The dynamics of an evolutionary process under random matching can be very different from the dynamics of the same process in a structured population.

4 The Project

Given that models in Evolutionary Game Theory comprise many individuals who repeatedly interact among themselves and occasionally revise their individually-owned strategies, agent-based modeling is an appropriate methodology for building EGT models. Agent-based modeling allows us to build models that are closer to the real-world systems that we want to study, because in an agent-based model we are free to choose the sort of assumptions that we deem appropriate in purely scientific terms. We may not be able to fully analyze all aspects of the resulting agent-based model mathematically, but we will be able to explore it through computer simulation.

The purpose of the project is to study the evolutionary behavior of a system composed of multiple agents using the EGT approach. In your report you need to provide the reader with all necessary background on the problem/phenomenon you have chosen, and the motivations for that choice (e.g. why this is an interesting problem, what applications does it have, etc). The simulation/modeling of the phenomenon has to be justified as clearly as possible. For instance, you should give the full details of your encoding/simulation of the phenomenon as well as argue why this encoding/simulation is a good one. During your study of the system, you need to analyze its evolution by studying all of the used parameters (such as the thresholds and the learning factors, and their impact on the system) and the behavior under “stress” conditions.

The specific application to be modeled will be chosen by the student. The main building blocks of the project can be summarized in three steps:

1. model the application scenario using the agent-based approach and define the agents' interactions

2. define the game that the agents are playing and the corresponding payoff matrix

3. analyze the evolution of the agents' strategy through the EGT approach.

1. Agent-Based modeling *Agent-Based Modeling (ABM)* is a methodology used to build formal models of real-world systems that are made up by individual units which repeatedly interact among themselves and/or with their environment. In this phase, you should implement the mobility of the agent (if any), the agents' connections/interactions and the dynamics of them (if any).

2. Game definition The game is defined by the payoffs matrix:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots \\ \vdots & \ddots & \\ a_{N1} & & a_{NN} \end{bmatrix}$$

where N is the number of possible actions that an agent can perform and a_{ij} is the payoff that agent i obtains when meeting an agent playing strategy j, with $i, j \in 1, \ldots, N$.
The modeling of the payoff matrix definition depends on the specific system that you are going to study. Depending on the inclusion of any spatial interaction (see Section 3.2.2) the game will be played among any pair of agents in the system or only among neighbors agents.

3. Evolution analysis Each agent has a predefined strategy (defined randomly at the beginning or defined by the user). As the execution of the game goes through time, the strategy of each agent will be updated following the *revision protocol*. You will analyze the evolution of the strategies and the total payoffs gathered by the agents during their lifetime. The noise during the revision protocol (as defined in Section 3.2.1) must be introduced to analyze the stability of the system.

4.1 Hints for agent-based modeling for EGT

The agent refers to a distinct part of our (computational) model that is meant to represent a decision-maker. Agents could represent human beings, non-human animals, institutions, firms, etc. Agents have individually-owned variables, which describe their internal state (e.g. a strategy), and are able to conduct certain computations or tasks, i.e. they are able to run instructions (e.g. to update their strategy). These instructions are sometimes called decision rules, or rules of behavior, and most often imply some kind of interaction with other agents or with the environment. The following are some of the individually-owned variables that the agents are going to implement:

- strategy
- payoff
- my-coplayers (the set of agents with whom this agent plays the game)

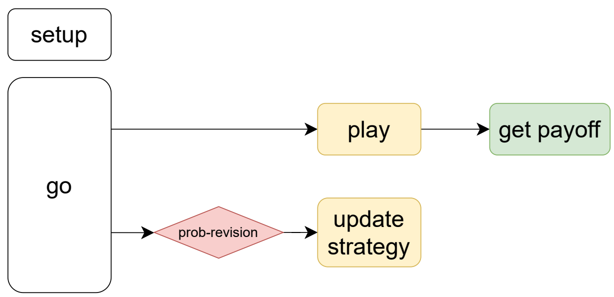

Figure 1: Skeleton of the code.

And the following are examples of instructions that the agents should be able to run (Figure 1 shows a possible skeleton of the code to implement):

- to play (play a certain game with my-coplayers and set the payoff appropriately)
- to update-strategy (revise strategy according to a certain revision protocol)

At high level, the following sequence of events is repeatedly executed:

- every agent obtains a payoff by selecting another agent at random and playing the game.
- with a certain probability *prob-revision*, individual agents are given the opportunity to revise their strategies activating their revision protocol.

4.2 Project example

The model we are going to develop in this section will allow us to explore games with any number of strategies. Thus, we will be able to model games like the classical *Hawk-Dove-Retaliator* [2], which is an extension of the Hawk-Dove game, with the additional strategy Retaliator.

Before we look at the game, the meaning of the term strategy in this context must be clear. A behavioral strategy is simply a fixed and predictable way of behaving in a contest. It does not imply that contesting animals make conscious decisions. Although contests involve two contestants, the purpose of a game-theory model is to compare alternate strategies with each other to see if one is better. In this case, we compare the contest strategies of Hawk, Dove and Retaliators.

A Dove (i.e., an animal that always plays the dove strategy) uses threat display in a contest but never fights. If the opponent also displays, then the Dove continues to display as well, but if the opponent attacks, the Dove retreats immediately, losing the contest but avoiding injury. Thus, a contest between two Doves is protracted and wastes a lot of time for both contestants, although neither contestant is injured. In contrast, a Hawk (an animal that always plays the Hawk strategy) attacks immediately. If a Hawk plays against a Dove, the Hawk always wins and the dove always loses because the dove retreats immediately. On the other hand, if a Hawk plays another Hawk, a fight ensues and both contestants risk injury as a result. Retaliators are just like Doves, except in contests against Hawks. When playing against Hawks, Retaliators behave like Hawks. A possible payoff matrix for this game is the following:

	Hawk(H)	Dove(D)	Retaliator(R)
Hawk(H)	-1	2	-1
Dove(D)	0	1	1
Retaliator(R)	-1	1	1

Let us consider the population game where agents are matched to play the normal form game with payoffs as above. Note that Retaliators are weakly dominated by Doves: they get a strictly lower expected payoff than Doves in any situation, except in those population states with no Hawks whatsoever (at which Retaliators get exactly the same payoff as Doves).

Figure 2 shows the best response correspondence of this game. Population states are represented in a simplex, and the color at any population state indicates the strategy that provides the highest expected payoff at that state: orange for Hawk, green for Dove, and blue for Retaliator. As an example, the population state where the three strategies are equally present, i.e. ($\frac{1}{3}$H + $\frac{1}{3}$D +$\frac{1}{3}$R), which lies at the barycenter of the simplex (in the middle of the triangle), is colored in green, denoting that the strategy that provides the highest expected payoff at that state is Dove.

We would like to study the dynamic stability of the unique stable strategy ($\frac{1}{2}$H + $\frac{1}{2}$D) both without and with spatial contexts.

Having seen all this, it may come as no surprise that if we simulate this game with the random-matching model we implemented in the previous chapter, retaliators tend to disappear from any interior population state. Figure 3 shows an illustrative simulation starting from a situation where all agents are Retaliators (and including some noise to allow for the entry of any strategy).

Now let's explore the dynamics of the spatial Hawk-Dove-Retaliator game. Will Retaliators survive in a spatial context? Figure 4 shows that Retaliators do

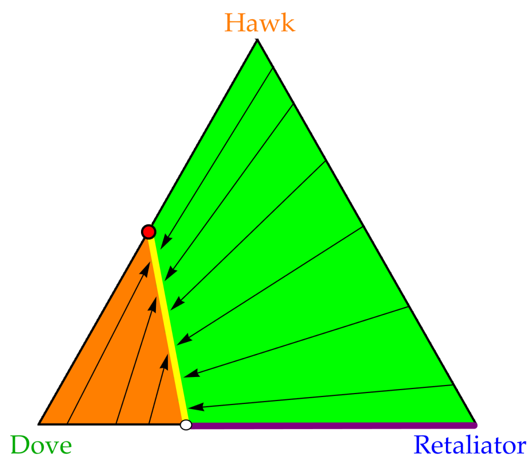

Figure 2: Best response correspondence for the Hawk-Dove-Retaliator game. Color indicates the strategy with the highest expected payoff at each population state. Arrows are just a visual aid that indicate the direction of the best response. The yellow line indicates that both Dove and Hawk are best response. The purple line indicates that both Dove and Retaliator are best response. All three strategies are best response at the white circle at ($\frac{2}{3}$D $+\frac{1}{3}$R). Finally, the unique stable strategy ($\frac{1}{2}$H + $\frac{1}{2}$D) is indicated with a red circle.

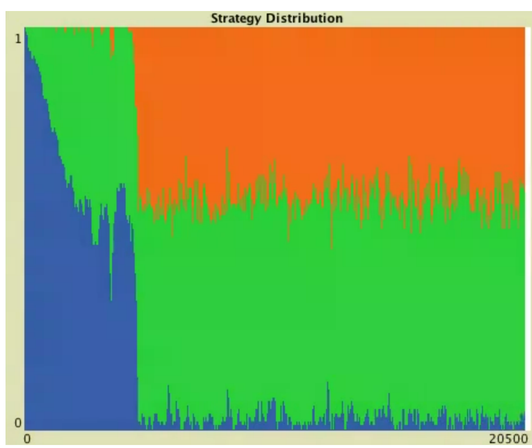

Figure 3: Simulation without spatial context. The system reaches the expected unique stable strategy ($\frac{1}{2}$H + $\frac{1}{2}$D)

not only survive, but they are capable of taking over about half the population. The greater level of noise means that more Hawks appear by chance. This harms Retaliators more than it harms Doves, but Retaliators still manage to stay the most prevalent strategy in the population. How can this be? We can

Figure 4: Simulation with spatial context.

notice that in spatial contexts neighbors face similar situations when playing the game (since their neighborhoods overlap). Because of this, it is often the case that neighbors choose the same strategy, and therefore clusters of agents using the same strategy are common. In the Hawk-Dove-Retaliator game, clusters of Retaliators are more stable than clusters of Doves (which are easily invadable by Hawks) and also more stable than clusters of Hawks (which are easily invadable by Doves). This partially explains the amazing success of Retaliators in spatial contexts, even though they are weakly dominated by Doves.

References

[1] Sandholm W.H. (2012) *Evolutionary Game Theory*. In: Meyers R. (eds) Computational Complexity. Springer, New York, NY.

[2] Ken Yasukawa, (2019) *Game Theory*, In: Encyclopedia of Animal Behavior (Second Edition), Academic Press, Pages 45-50,