Simulation Project Guideline

Group :Boissier, Caetano, Couput, Séailles

March 2, 2023

1. Project overview

We'll simulate the game Hens Fox Viper. Our goal is to test various sets of strategies and behaviors for each species so that we can have an overview of how to survive in the simulation.

1.1 Report modifications

05/04/2023 Initial version

21/04/2023 First iteration

26/05/2023 Second and last iteration

1.2 Context

The environment for this project is a virtual static environment, represented by a 2D grid. The agents (hen, fox, and viper) will move around on this grid and interact with each other based on their behaviors and attributes. The environment is a forest ecosystem (a limited area of the world) with different regions representing the habitat of each agent. When an agent is in its habitat, its properties (speed, lifespan) are enhanced. The environment is dynamic, as it changes based on the agents' actions and interactions, but it is closed, as agents might not enter or leave the system at any time. The environment can also contain randomly generated food items for the animals to consume.

What are the hypotheses and the constraints that our solution has to deal with?

One hypothesis is that the agents will adapt their strategies based on the actions of other agents in the environment. Constraints include limited knowledge of the environment, finite resources for each agent, and the requirement of decentralized decision-making.

- The agents will behave based on predefined behaviors and attributes, and will not have any learning or decision-making capabilities beyond this.
- The environment will not change or evolve over time, and will remain static.
- The simulation will only include three types of animals (hen, fox, and viper) and will include other elements such as food appearing on the grid.

Why is a multiagent solution adequate to solve this problem?

A multi-agent solution is adequate because it allows the agents to operate independently and make decisions based on their own goals and attributes, while still interacting with other agents in the environment. This decentralized approach supports scalability and adaptability to various situations.

What are our indicators? What will we observe?

The indicators to measure the success of a strategy include the population balance of each agent, the time it takes for one species to disappear, or to become dominant. Also, the death toll by hunger and by predators is important to consider whether the environment is relevant and well-defined. Observations will focus on agent interactions, resource usage, and environmental changes. Our goal is to see how a change in agents' behaviors affects their individual and global displacements.

1.3 Multiagent solution overview

Agent: The agents in the system are the hen, fox, and viper. Each agent has its own set of attributes, such as speed, hunger level, sight range, and energy level. The agents interact with each other based on their behaviors and attributes.

Environment: The environment is a 2D grid where the agents move around and interact with each other. As we mentioned before, the environment can contain randomly generated food items for the animals to consume. Each animal has a detection range that allows it to detect other animals within a certain distance.

Interaction: The agents interact with each other by detecting other agents within their detection range. When an animal detects another animal within its range, it may move towards it if it is prey or move away from it if it is a predator. If a predator catches its prey, it consumes it and gains some energy.

Organization: The agents act independently based on their attributes and behaviors.

There is no central decision-making process, and each agent must rely on its own abilities to survive in the environment.

Indicators: The simulation will produce indicators such as the number of animals alive at the end of the simulation, the time it takes for one species to become dominant over the others, and the average number of each animal type consumed by the predators. These indicators will provide insight into which animal is dominant in the simulation and how the different behaviors and attributes of the agents affect the outcome.

Overall, the multiagent model for the hen-fox-viper simulation is based on the interactions between the agents in the environment. Each agent acts independently based on its own attributes and behaviors, and the outcome of the simulation is determined by the interactions between the agents.

2. Conceptual model

2.1 Agent

Overall, the agents in the simulation perceive the presence of other agents and food within their detection range and act based on their own attributes and behaviors. The agents' decision processes are more or less simple and the agents' action models include moving, consuming food, and getting consumed by predators.

Our goal is to change the agents' behaviors, strategies and attributes and observe the consequences on the evolution of the system. Here is an example set of behaviors we could test :

Hen: The hen perceives the presence of other animals within its detection range, food items within its detection range, and whether it is hungry or not. Its knowledge includes its own attributes (speed, hunger level, range) and its behaviors, depending on the strategy tested (for instance move straight towards food if hungry) and the other agents and food in its range. The hen's decision process is simple and consists of moving towards food if it is hungry, moving randomly if it is not, and avoiding predators if they are within its detection range. The hen's action model includes moving in the direction of its chosen behavior, consuming food if it reaches it, and getting consumed if it is caught by a predator.

Fox: The fox perceives the presence of other animals within its detection range, food items within its detection range, and whether it is hungry or not. Its knowledge includes its own attributes and behaviors. The fox's decision process is more complex than the hen's and includes hunting prey in priority if it is within his range and that its hunger level is good enough , consuming food in priority if its hunger level is too low, and avoiding predators in all cases. The fox's action model includes moving towards prey or food, consuming food if it reaches it, and getting consumed if it is caught by a viper.

Viper: The viper perceives the presence of other animals within its detection range and whether it is hungry or not. Its knowledge includes its own attributes and behaviors. The viper's decision process is similar to the fox's and includes hunting prey, consuming food, and avoiding predators. The viper's action model includes moving towards prey or food, consuming food if it reaches it, and catching and consuming other animals if they are within its detection range.

2.2 Environment

For the Hen-Fox-Viper simulation project, the environment model would be as follows:

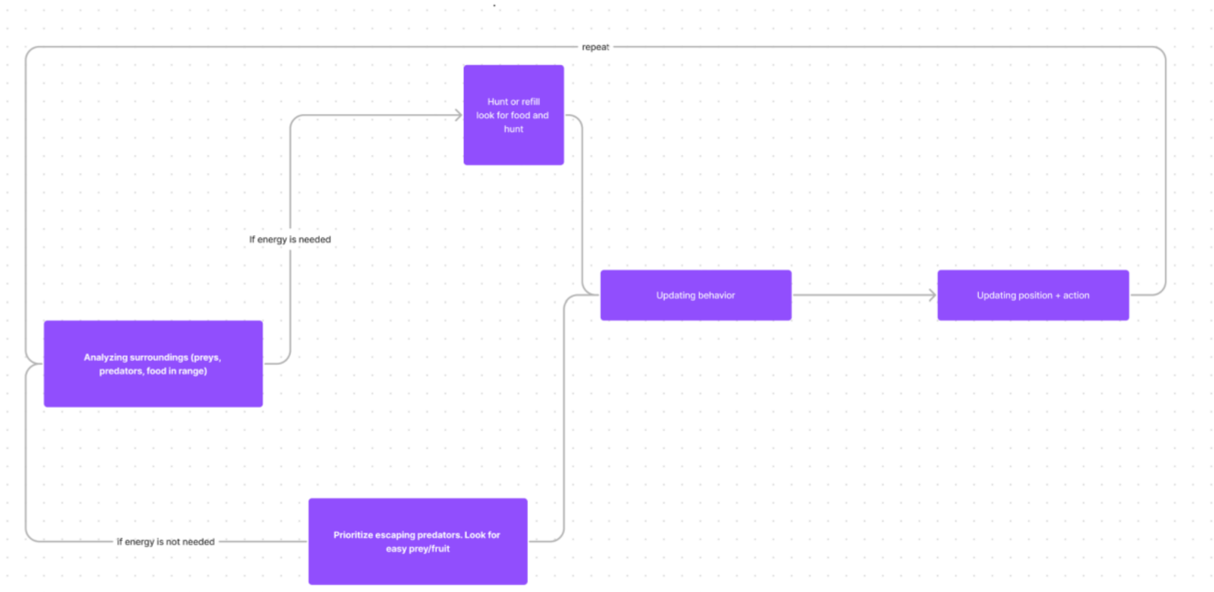
The environment:

- Is represented as a 2D grid with cells
- Contains the agents: hen, fox, and viper
- Contains food items
- Agents can perceive their immediate surroundings within certain range of detection
- Agents can move around the environment, consume food items, and attack/avoid other agents based on their Individual behaviors and attributes
- The environment will be dynamic, with food items appearing and disappearing randomly over time

The perception of the agents will be limited to their immediate surroundings, within a certain range of detection. The range of detection will differ from agent to agent, with the viper having the largest range, followed by the fox, and then the hen for instance. The agents will be able to perceive the presence of other agents and food items within their range of detection.

The actions of the agents on the environment will include movement, attacking, consuming food items, and reproducing (if applicable). The movement of the agents will be governed by their individual speed attribute. The attacking behavior of the agents will be based on their prey/predator relationships, with the fox attacking the hen, the hen attacking the viper and the viper attacking the fox. The consumption of food items will provide agents with sustenance and affect their hunger level attribute.

2.3 interaction



Automate of agent behaviour

Updating behavior corresponds to the definition of a new objective for an agent that detect a modification in its surrounding environment.

Hunting competition: when a prey appears on the grid, all agents in the detection range of the prey may try to catch it. The first agent to reach the prey will eat it and gain energy points.

Predator-prey relationship: if a predator detects a prey, it will try to catch it. If the prey is caught, it will be eaten by the predator and the predator will gain energy points. If the prey manages to escape, it will continue to roam the grid.

Predators may also compete with each other: for example, if two foxes detect the same prey, they may try to catch it at the same time. The fastest fox will catch the prey, leaving the other one hungry.

These interactions will influence the agents' decisions, as they will need to constantly adapt their behavior to the presence of other agents and the availability of food.

2.4 Organization

In this system, the organization is simple as each animal acts independently without any coordination. Each animal follows its own set of rules and has a specific behavior that determines how it will interact with the environment and other agents. The rules can be defined based on the attributes of each animal such as speed, hunger level, and detection range.

The fox, for example, will try to hunt the hen or viper if it is hungry and within its detection range. The hen will try to avoid the fox and viper and search for food within its range. The viper will stay still and wait for its prey to come within its striking distance.

The organizational relations between the agents are competitive as they are all competing for survival and resources. This means that agents can be both predators and prey depending on their situation and attributes.

The consequences of this organization are that there is no centralized control or coordination, which can lead to emergent behaviors and outcomes. For example, the dominance of a particular animal species may emerge based on its attributes and interactions with other agents in the environment. It also means that there is no guarantee of a stable equilibrium, as the environment and agents may constantly change and interact in unpredictable ways.

3. Implementation

In this section, we will discuss the implementation of our conceptual model for the Hen-Fox-Viper simulation project. The goal is to explain the necessary adaptations and decisions made while developing the simulation model from the conceptual model.

3.1 From the conceptual to the simulation model

We implemented our conceptual model by defining agents, their attributes, and behaviors, as well as their interactions within the environment. Speaking of the environment, we also had to define it in the first place ! The following steps were taken to adapt the conceptual model into the simulation model:

- Creation of agent classes: We created separate classes for each agent type (hen, fox, and viper) and defined their attributes : speed (velocity), detection range (vision) and energy rate (so that an agent would die if it hasn't eaten in a while).

- Environment setup: We set up the environment as a 2D grid and implemented functions to randomly generate food items on the grid. We ensured that the environment would be dynamic, allowing for food items to appear and disappear over time.

- Implementing agent perception: We programmed each agent's perception abilities, allowing them to detect other agents and food items within their detection range.

- Implementing agent behaviors and decision-making processes: We implemented the decision-making processes and behaviors for each agent as described in the conceptual model, including movement, food consumption, predator-prey interactions. This is done through an `update()` function, which takes as input an agent, the environment data, the food it eats, and the prey type it eats. Then, this function checks if it is time for the agent to die (due to its energy level being too low), defines the new target of the agent (being food or a prey) if it is still alive, it updates its position consequently and update its new energy level.

- Implementing agent "interactions": Speaking of "interactions is not really correct, since our agents don't share information with each other. However, to test some group strategies, we built a logic in our code that enables agents to take their allies' positions to make a decision (for example if they have to attack a prey or if they prefer to get closer to an ally).

- Simulation loop: We set up a main simulation loop that would iterate through each agent's actions, updating the environment and agent states accordingly. The loop would continue for a specified number of iterations or until specific conditions were met (e.g., one species becoming dominant, all agents are dead).

3.2 Indicators' implementation

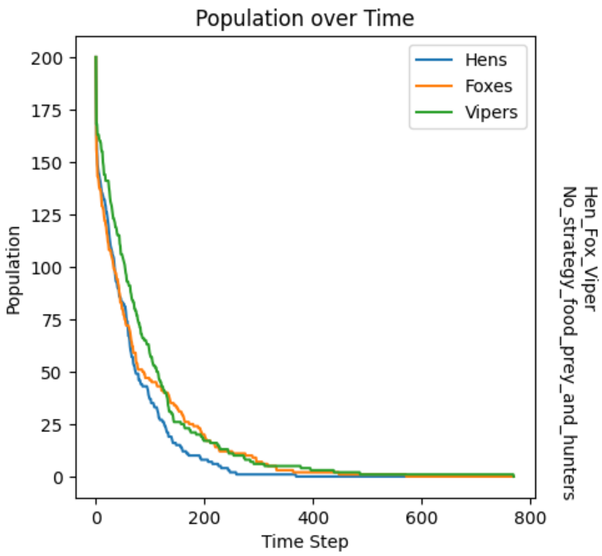
To monitor the simulation's performance, we implemented the following indicators in greater detail:

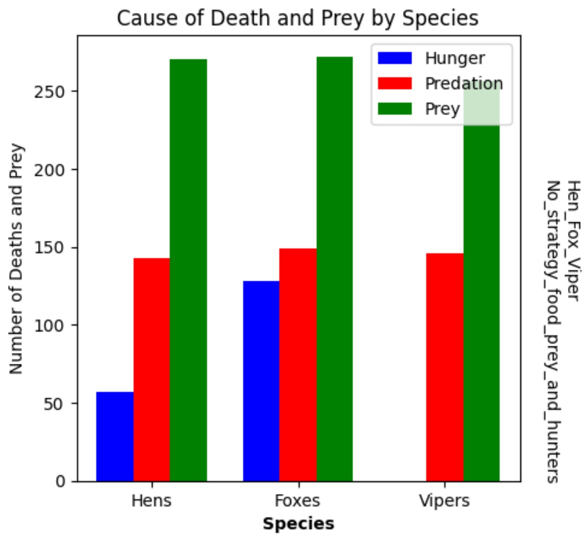
1. **Population balance**:
   a. The population count of hens, foxes, and vipers is counted at each time step by tracking the number of living agents of each type.
   b. The counts are stored in separate lists (`hen_population`, `fox_population`, `viper_population`) that record the population count of each agent type at each time step.
2. **Time to species dominance or disappearance**:

    a. The disappearance times for each species (hen, fox, viper) are tracked by checking if the population count of a species becomes zero for the first time. The corresponding disappearance time is recorded as the current time step.
    b. The domination times and durations are determined by comparing the population proportions of each species to a dominance threshold (55%). If the proportion of a species exceeds the threshold, the domination time is recorded as the current time step. The duration of domination is incremented for each time step where the dominance condition is met.
3. **Death toll**:
    a. Deaths due to hunger and predation are tracked separately for each species.
    b. For hunger-related deaths, the count is incremented for agents that have the `died_of_starvation` attribute set to `True`.
    c. For predation-related deaths, the count is incremented for agents that have consumed food (`food_eaten` attribute > 0).

These indicators provide insights into the population dynamics and performance of the simulation, allowing for analysis and comparison of different runs or scenarios.

Then, we plotted some graphs to display the evolution of those indicators over time during our different simulations :



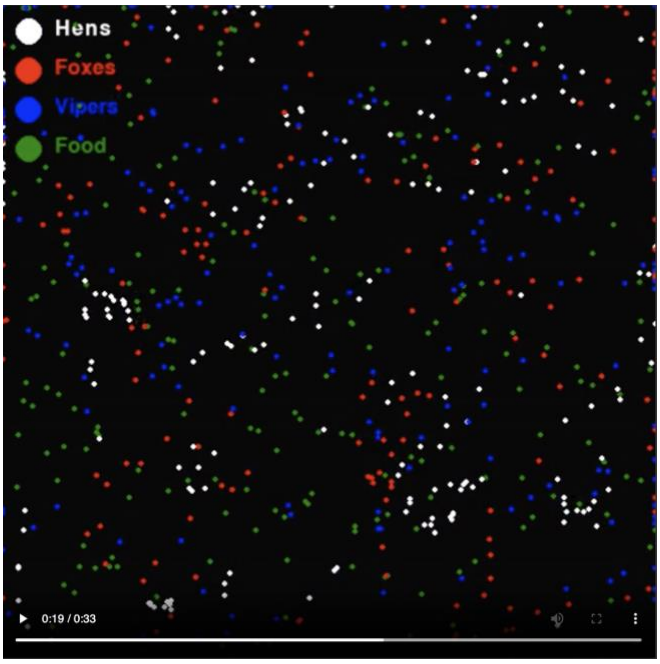


Eventually, we wanted to give a visual dimension to our project, so we used the `pygame` library to visualize the evolution of agents during the simulation. This was really important to iterate over our simulations and to realize what problems we were facing with the agents' behavior.

Alors, avec l'aide de cette librairie, nous avons pu afficher nos agents sur un écran à chaque étape de nos simulations, enregistrer chacune des images d'écran, pour finalement obtenir une vidéo qui retrace toutes les images contenant les déplacements de nos agents et leurs interactions avec l'environnement.

Voici à quoi ressemble une “frame” de vidéo :

Nous enverrons différentes vidéos issues de nos simulations en même temps que notre compte-rendu !



3.3 Scenarios

Now that we have defined our environment, our characteristics of interest for every agents and now that we can monitor their activity and the status of the simulation at each of its step, we are ready to define our two scenarios and our witness experiment. .

We will develop on the scenarios just after this part, but you can already note that for every scenario that we implement, the hens, the vipers and the foxes will keep the same attributes. This enables us to really test the scenarios and strategies instead of just analyzing the shift in attributes and capabilities.

<u>Here is an overview of the simulation parameters used in the code:</u>

The screen dimensions are set to a width of 500 pixels and a height of 500 pixels. We tested different dimensions and this is just pretty random, but it works.

The simulation involves three types of agents: hens, foxes, vipers and food items. Each agent is represented by a little disk on the screen, with a different color for each class.

To define the **class attributes**, we chose to construct balanced agents, but with distinct characteristics. Therefore, we built it this way :

Foxes will be the worst about their hunger followed by the hens and then the vipers that have the lowest energy loss rate.

For the speed, the hens would be the slowest, followed by the viper and the foxes will be able to run faster than the others.

Concerning the vision range, viper would have a narrow vision of the environment while the foxes would have better sight and the hens would be able to see enemies from far away.

<u>More precisely</u> :

The hens have a maximum velocity (vmax) of 0.2, a vision radius of 70 pixels, and an energy decrease rate of 0.002.

The foxes have a maximum velocity of 0.3, a vision radius of 50 pixels, and an energy decrease rate of 0.003.

The vipers have a maximum velocity of 0.25, a vision radius of 30 pixels, and an energy decrease rate of 0.001.

<u>Details of the scenarios following our conceptual model</u>

The idea is to try out three different scenarios :

- One in which all our agents interact without considering a specific strategy : they just aim at the closest prey. This will act as a simulation of reference, and will enable us to test if the attributes we have defined for the different classes don't make a class too strong or too weak in comparison with the two other classes. Then we'll be sure that our classes are balanced in terms of life expectancy.

- Then, we'll test a group strategy : A predator would attack its prey only if it has more allies around him than its prey has. Also, if a predator is not in a position to attack (because he doesn't have enough allies in his surroundings), it will rush towards the closest ally. The goal here is to mimic the behavior of wild animals, and to see if some patterns appear in the simulation. Typically, we would expect that allies reunite and create mini communities naturally in order to survive and be able to eat smaller preys' communities.

- Eventually, we want to test an individual strategy, which is a bit bolder. In the game Hens Foxes Vipers, your prey is the predator of your predator, and the predator of your predator is your prey. Then, a strategy to prevent a predator from pursuing you is to rush to a prey, so that this prey can target your predator. Also, if you rush toward the direction of your predator, you might encounter his predator, which will be your prey. We'll try this strategy to see if some interesting survival patterns and behaviors appear.

To go further on our experiment, we could enable enemy agents to collaborate against a target, or test other strategies.

<u>How we perform the experiments</u>

The simulation runs until the user chooses to quit or until 500 frames are generated, whichever comes first. We chose this time because it enables us to generate videos of 50 seconds at a rate of 10 fps.

The agents have interactions with each other based on their vision radius. For example, they can chase and eat food items or other agents within their vision range.

The agents have energy levels that decrease over time. If an agent's energy reaches 0, it dies. The cause of death can be either starvation or hunting, depending on the agent's behavior and interactions.

At the end of each experiment, some plots are generated, describing how the population of each class had evolved, how many agents of each class have died of starvation, of hunting, etc.

3.4 Results

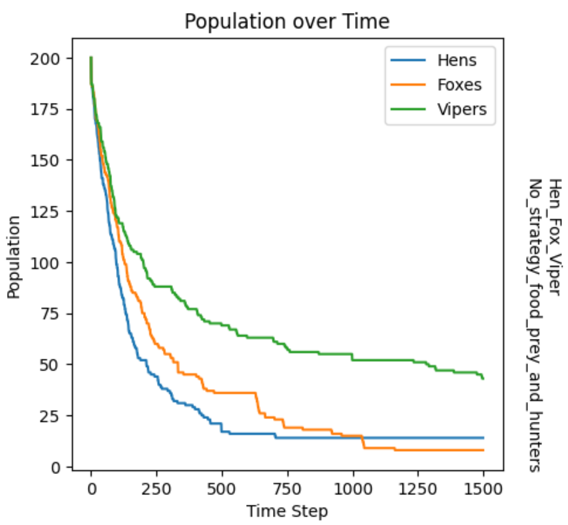
Know that you have access to the videos of the simulations we did for our project.

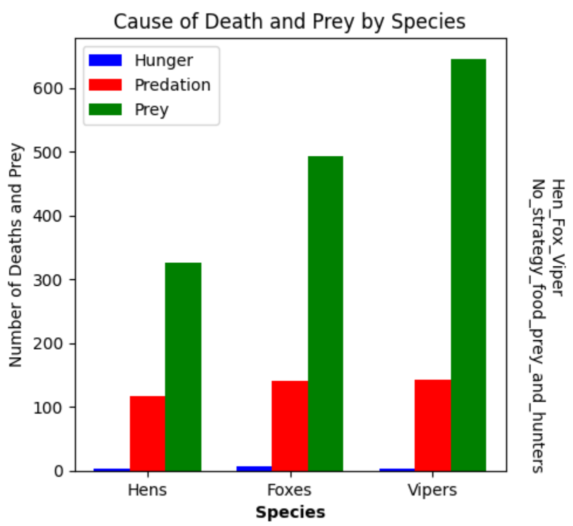
<u>Scenario 1</u> : Witness simulation

This witness scenario plays its part. We can see clearly the food being generated. Also, the evolution of species' populations is not too abrupt : The classes are balanced. However, vipers take the lead and become dominant at iteration 472. It doesn't come too quickly though, so it's not shocking.

Here is the console output to prove it :

Hens disappeared at: None
Foxes disappeared at: None
Vipers disappeared at: None
Hens dominated from: None to None
Foxes dominated from: None to None
Vipers dominated from: 472 to 1467





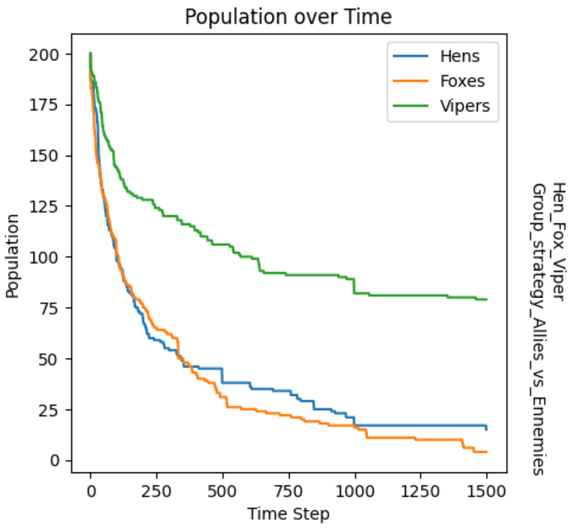
Finally, the evolution of population size through time follows more or less the same tendency for both species, and the ratio between the proportions between the number of death by hunger, death by predation and number of eaten preys is similar from one class to another.

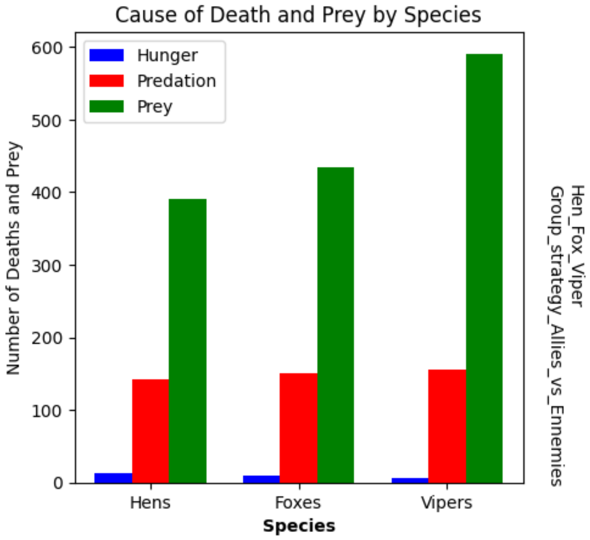
Even if we can't really observe anything interesting here in terms of strategy / behavior,, we made sure that our class attributes made them balanced, and that our simulation worked well ! This is essential to be able to continue on different scenarios.

<u>Scenario 2</u> : Group strategy

As a reminder, this strategy states that a predator would attack its prey only if it has more allies around him than its prey has. Also, if a predator is not in a position to attack (because he doesn't have enough allies in his surroundings), it will rush towards the closest ally. Typically, we expected that allies reunite and create mini communities naturally in order to survive and be able to eat smaller preys' communities.

Let's see what happened from the plots :





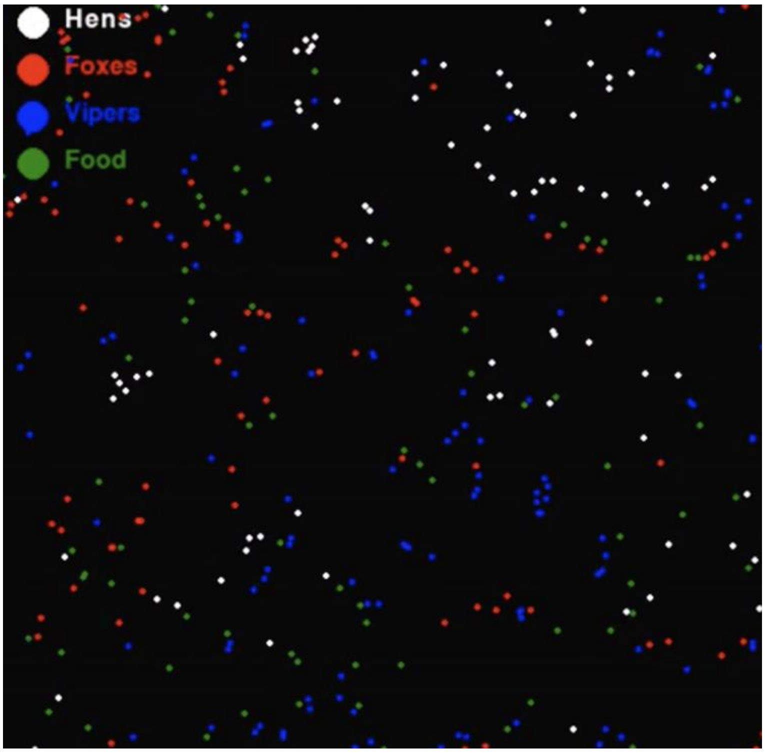
Well, it seems that the vipers were advantaged by this group strategy. This strategy relies very much on the vision range of each class. Indeed, the decision making process is made by evaluating the number of **preys** and **allies** within the agent's **detection range** !

However, this strategy can seem a bit odd. Indeed, in our experiment, preys, whatever their number might be, can't do anything against their predators. Then, the idea to avoid hunting preys when they are too numerous doesn't make sense.

Then, if all agents play their part and avoid their preys equally, it wouldn't advantage any species. However, **the vipers** have the **lowest detection range**. Then, they tend to go more toward their allies than the other species, and consequently they feel more numerous and can attack their preys more easily (because they don't see their preys when they are not close enough).

This simulation felt super interesting, since it enabled us to understand that some strategies, even if they are applied to all animals (i.e agents), can grant a decisive advantage to one particular species (i.e class).

We also clearly see some patterns that weren't appearing in the witness simulation.
Indeed, one can see that many groups of individuals are naturally created since the early stage of the simulation :



Frame of the simulation 2

Scenario 3 : Bold strategy

Eventually, we wanted to test a bolder and more ambitious strategy. As a reminder, in the game Hens Foxes Vipers, your prey is the predator of your predator, and your predator is the prey of your prey. Then, a strategy to bait a prey is to rush toward a predator to try and eat its own predator. We'll try this strategy to see if some interesting survival patterns and behaviors appear.

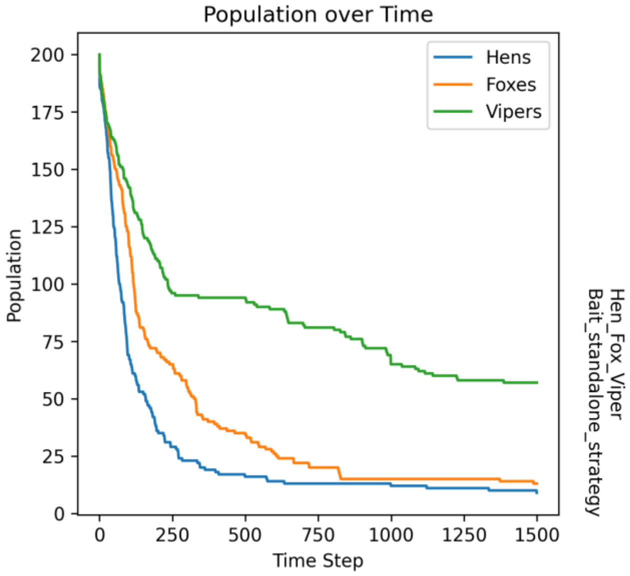
Here is the reasoning of an agent adopting this strategy :
"If I have one of my predators' prey within my field of vision, I can run in its direction. If I run towards my ally who is being eaten by the predator, I am heading in the predator's direction without running into it (so avoiding to be eaten) and hope to catch its own predator, which is my prey."

To implement this strategy very concretely, here is the relevant piece of code :

```python
# agent doesn't have a target or haven't reevaluated since 11 updates, find a new one
if not self.target or self.age % 11 == 0:
    min_dist = 9999999
    min_agent = None
    # for food, prey, and allies, only consider those agents which are of different classes
    for a in (food + prey + allies):
        if a is not self and a.is_alive and type(self) != type(a):
            sq_dist = (self.x - a.x) ** 2 + (self.y - a.y) ** 2
            if sq_dist < min_dist and sq_dist < self.vision ** 2:  # Consider only those within vision.
                min_dist = sq_dist
                min_agent = a
    if not min_agent:
        # Check if a predator is near with a target, if so, target the predator's target instead
        for p in predators:
            if p.is_alive and p.target and ((self.x - p.x) ** 2 + (self.y - p.y) ** 2) < self.vision ** 2:
                min_agent = p.target
    if min_dist < self.vision ** 2:  # Set target only if it's within vision.
        self.target = min_agent
```

I think that the code is pretty explicit by itself.

Then, how about the results of this last simulation ?



Well, vipers won this last simulation. When investigating over the video, we can make two observations :

- The strategy takes place only if the agent has no prey to target, which is pretty rare at the beginning. Then, there are plenty of agents which just die even before experimenting the strategy…

- When the simulation goes on, more and more foxes tend to adopt the strategy. However, for most of them, it is too late. Indeed, the predators of their predators, which are the hens, have taken great damage and are not that available for hunting. As a consequence, agents tend to die with their allies, or end up eating the generated food infinitely, without dying.

This strategy shows that to be efficient, some behaviors must be implemented from the beginning of a simulation. Otherwise, it can be too late…

Conclusion

In conclusion, our work consisted of simulating different scenarios in the game "Hens Foxes Vipers" to explore various strategies and their impact on the population dynamics of the species.

In the witness simulation scenario, we observed a balanced evolution of species populations, with vipers gradually becoming dominant without causing any abrupt changes. The console output confirmed this trend. Although we didn't observe any particularly interesting strategies or behaviors, we ensured that the simulation was well-balanced and functioning correctly, laying the foundation for further exploration in different scenarios.

In the group strategy scenario, we implemented a strategy where predators would attack their prey only if they had more allies present than the prey. This strategy favored the vipers due to their lower detection range, leading them to gravitate more towards their allies and perceive themselves as more numerous. This advantage resulted in easier attacks on their prey. While this strategy seemed peculiar since the prey had no means of defending themselves regardless of their numbers, it highlighted the influence of species-specific attributes on the effectiveness of a strategy.

The group strategy simulation revealed patterns that were not evident in the witness simulation, with the formation of natural groups early on in the simulation. This observation further emphasized the impact of strategies on species dynamics and the emergence of distinct patterns in the population.

In the bold strategy scenario, we tested a more ambitious approach where agents would rush towards a predator to try and eat its own predator, following the concept that "your prey is the predator of your predator." Vipers emerged as the winners in this simulation, but it was observed that many agents died before being able to employ the strategy. Additionally, as the simulation progressed, more foxes attempted to adopt the strategy, but it was often too late as the hens (predators of their predators) had already suffered significant damage and were not readily available as prey. This scenario highlighted the importance of implementing certain behaviors from the beginning of a simulation for them to be effective.

Overall, our work shed light on the influence of different strategies on population dynamics and the advantages conferred to specific species. It also emphasized the significance of early implementation and balanced attributes in studying the survival patterns and behaviors of the species. Further exploration and analysis of these strategies can provide valuable insights into ecological interactions and dynamics within the game.

Short opening

One thing that would be really interesting is to train the agents with a machine learning approach from the results of many simulations to adapt their attributes and their strategies to see where it would bring us. To do so, we would have to define some new metrics and for new parameters / attributes on which to play.

4. References

[1] Différentes stratégies de jeu imaginées par un établissement scolaire

https://saintbaudille.fr/wp-content/uploads/2021/09/Poules_renards_viperes.pdf

[2] Sur les stratégies de défense

https://link-springer-com.em-lyon.idm.oclc.org/article/10.1006/bulm.1999.0114

[3] Sur les stratégies d'attaque des animaux

https://www.ouest-france.fr/leditiondusoir/2020-09-11/cinq-strategies-gagnantes-chez-les-animaux-pour-survivre-a-une-apocalypse-88f6d99a-1267-4ae0-99b3-436be02f3f13