Modeling Natural Selection using NetLogo

Daniel Cohen-Cobos

Department of Computer Science, The College of Wooster

May 6, 2022

Abstract

In this paper we looked into simulating natural selection using an agent-based computer simulation, monitoring their traits and their evolution across generations. The simulation was written in NetLogo, an agent-based programming language, to model an environment in which the simulated species sought food to survive each iteration. If the agent collected enough food, it reproduced having a child that could mutate its set of traits. It could change its average velocity, awareness of its surroundings, orientation, or violence towards other agents, potentially gaining the ability to steal the food of its kind for survival. We ran the simulation 8 times to analyze the results to find the initial behavior of the species to those traits. We then ran a longer simulation to examine how the analysis changed over a longer time. The findings include: a dominant trait, an unsuccessful trait, overpopulation and population equilibrium at different times, and the creation of two breeds coexisting with each other having different sets of traits.

Contents

1 Introduction

1.1 What is Natural Selection?

Charles Darwin introduced the concept of natural selection in his book titled *On the Origin of Species* in 1859. This revolutionary book contradicted what he denoted as *artificial selection* which was the common belief back then, where each species had been specifically designed and had always remained constant in their respective characteristics. He defined natural selection as the ability of species to adapt to their environment [1]. Species change to adjust to their surroundings and improve their chance of survival.

1.2 How does it work?

In a nutshell, species across generations mutate their abilities and customs to adapt to their surroundings. *"Distinct species of organisms apply themselves to different ecological tasks using their appropriate sets of tools"* [2]. They mutate slowly and those who can manage to adapt better will survive, and then reproduce repeating the cycle. Those who have unsuccessful mutations will be more likely to die and therefore will not spread the wrong mutation to the next generations.

In real life, noticeable traits can be seen by comparing species thousands of years apart. After various successful mutations, different variations of the same initial species can coexist together. *While many of these forms differ in subtle ways, most can be readily recognized and categorized as types or species quite distinct from others* [2]. This idea was tested in this study, were in the first iteration, all individuals had the same traits, and by the end of the simulation, there were almost no two identical individuals from the species.

2 Procedure

In this study, we used the concept of natural selection to model trait evolution on a species. The species and its individuals will be referred to as **cronies** throughout the rest of this article.

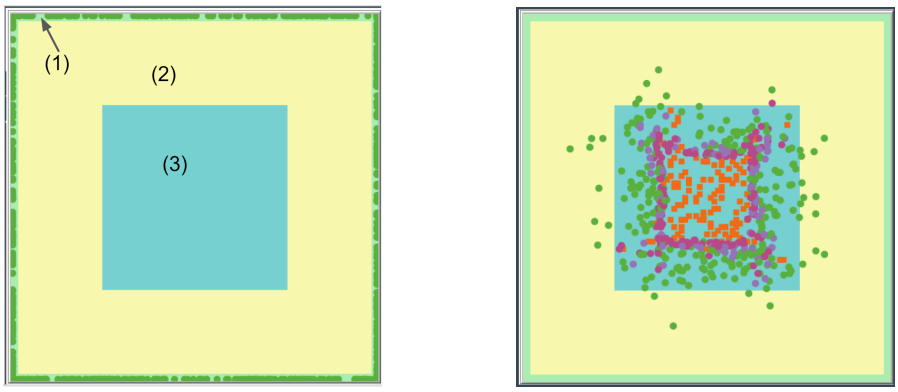
2.1 Tools

We used agent-based programming to model natural selection. This type of programming allows simulating various instances of an agent and its interaction with its environment and other agents. In our case, these agents were the cronies, and the environment is the map on which the agents will be simulated in. Agent-based programming allows agent-agent and agent-environment interaction using externally specified objects, which was the main reason why this type of simulation was chosen. [3]

The agent-based programming language used in this study was NetLogo [4], because of its easy use, efficiency, and its versatility [3]. NetLogo creates a 2-dimensional output to view the environment, the agents, and every interaction that occurs in each iteration. It can also output graphs to monitor parameters within the simulation, or create variables such as mean, the highest number of certain variables, or other parameters to use for a better analysis of the evolution.

2.2 Model Design

The model was based on [5, 6], and consists of a 2D plane on which the cronies lie. The plane has the shape of a square and is composed of three zones, as seen in Fig 1. The **home zone** is where the cronies start and end after each iteration; the **empty zone** is where the cronies start moving through when the iteration starts, on their way to the **food zone**, where food is randomly placed. This design was inspired by *Primer*'s design [5] where the exterior of



(a) (b)

Figure 1: Model's Environment divided into three zones. In (a) we can see the zones, which are the home zone (1), empty zone (2), and food zone (3). In (b) we can see the cronies in the environment, where the cronies with no food is color coded green; if they collected one food unit they are coded purple, and if they collected two food units they are coded magenta. The food units are represented as orange squares.

the map is the "safe zone" and the board is the "fight zone" for the cronies to survive. In the case of this study, there is a space separating these two zones where food is now generated to allow the cronies to leave the food zone once they have their food item(s) leaving those who didn't collect it some more space to do so.

2.2.1 Food Mechanics

Food served as a way of survival and reproduction. If a crony collected one unit of food and managed to bring it back home before the iteration finished, It would survive that iteration. If it managed to collect two units of food and bring both of them back home, it would not only survive but also reproduce, creating a new instance of itself with a chance of a slight modification to its parent´s traits.

2.2.2 Species & Iterations

Each iteration started with the cronies in the home zone facing the middle of the map. Then, they moved in the function of their traits. They each had an orientation parameter that was used to determine how they moved towards their next objective. The cronies first went

through a seeking phase where they went towards the middle of the map to collect their item of food which to allow them to survive that iteration. Once a crony came close enough to a food unit, it collected that item and carried it as long as it had room to store it (they only had space to carry two). Once they collected their food, they started their second phase where they moved in the opposite direction to the middle, so they could return to the home zone. On their way back, they could have collected their second food item which allowed them to reproduce, but they could not have collect a third food item. If the crony lost their food item (explained in the trait section), they would return to their first phase and switch directions towards the middle. The cronies were constantly moving until they reached the home zone after collecting their item(s).

When they reproduced, the child received the same trait parameters as its parent. The simulation had an evolution parameter that determined the chance of the child to have a mild modification to those traits. If there was a modification, those modifications were random and distributed across the different trait parameters.

2.3 Traits

Traits are the essence of each crony. It determined how its breed was evolving and how it interacted in the environment. Each trait had an advantage and a disadvantage like they do in real life [7]. Traits are created to adapt to different environments and may not work in different habitats. When the cronies were born, it started off as an identical copy of its parent, and then it went through the following equation

$$t_a = t_{a_0} + \mu_1\delta_a - \mu_1\delta_a$$

for each of the traits to determine if they would increase or recess. Here t_a is the evaluated trait, t_{a_0} is the initial state of the trait, μ_i is a random number between 0 or 1 (in this case μ is used twice to get two random numbers between 0 and 1) and δ_a is the increment of

the trait which remained constant throughout the simulation. The reason for this equation was for the trait of the father to be duplicated and then have a 50% chance of increasing or decreasing δ_a. If both random numbers happen to be the same, the trait will remain the same since it will either not change, or the increments will cancel out. This caused each trait to only have a 25% chance of increasing, 25% chance of decreasing, and a 50% chance of remaining the same. This made half of the cronies of the simulation to be identical to their parents and the other half to be mutations.

The following traits were chosen to explore the different interactions the cronies can have with the environment and each other.

2.3.1 Speed

The speed trait t_s made the cronies run faster in one of the phases, but as a consequence to conserve energy, they went slower in their other phase. This should be very beneficial to ensure they either collect food faster before it gets collected or to make sure the food gets home safer. This trait has those two possibilities, in which the speed trait parameter can be positive for a faster first phase, or negative for a faster second phase. The equations to move the cronies every iteration are

$$\Delta s_+ = 1 + t_s$$

$$\Delta s_- = (1 - t_s)^2 \quad ,$$

where Δs_+ is the spatial displacement in the faster phase, and Δs_- in the slower phase. The slower phase was motivated by the concept of *energy* from Newtonian mechanics, where $E_k = (m/2)v^2$, being the total "movement" energy dependent on the velocity squared. A way to visualize this is that the extra energy spent to increase the velocity in the faster phase is notably subtracted from the other phase since it then squares the number making the displacement smaller.

2.3.2 Awareness

The awareness trait t_a made the required distance to collect food wider, which allowed the cronies to find food faster and return home sooner. This can be seen as a "vision radius". The downside of this trait is that their orientation worsened with this trait. The metaphor behind this idea is that a high awareness implies that the crony moves around more and gets distracted easier by what they see around them. When looking for food the radius r, it became $r = r_0(1 + t_a)$, being r_0 the standard radius of the simulation; incrementing the distance needed to find the food units. Note that this new radius used for cronies with awareness trait was also used in the violence trait, which is explained below. The way we implemented the disorientation was

$$\theta = \theta_0 + (\mu_5 - 2.5) * t_a \quad ,$$

where θ was the angle at which the cronies were heading and θ_0 the initial angle. This caused the cronies to always have some kind of path deviation.

2.3.3 Orientation

Opposite of awareness, the orientation trait t_o ensures an easier collection of food. In a radius $r_0 * (1 + t_o)$, any food within that region will be set as a target, and the crony will be set to be facing directly at that target. As a consequence of this, the target may be in the region but not be the closest food item to the crony. It will go towards a food item that may not be the most convenient, and it may not reach the food item in time before its taken. Another possibility of this trait is that the target may switch from one iteration to another, causing the crony to switch paths toward another food item before collecting it.

2.3.4 Violence

As the last trait, the violence trait t_v enabled agent-agent interaction by allowing agents to steal food from one another. When two cronies were within a radius r of each other (this radius could be affected by the awareness trait as explained previously), they interacted with each other. Each crony gained a probability of stealing a food unit from the other crony (if they have at least one), based on its violence parameter (as long as $t_v > 0$). The probability of a successful food unit being stolen was

$$P(t_v) = \frac{t_v}{t_v + \delta_v} \quad ,$$

where the higher the violence trait, the higher the probability, but never actually achieving 100% chance of a successful steal since this function approaches 1 as the violence trait goes up. If they both had a violence trait parameter, there was a chance they fought for food where they would exchange the unit until one of them failed to steal from the other. As a consequence, they only stole for survival, which means that they couldn't steal food if they already had had a food item. This helped their survival but limited them to reproduce by stealing.

3 Results and Analysis

In the constructed simulations, we used all trait increments $\delta_i = 0.01$, initial trait values $t_i = 0$ for each trait, radius $r = 1$, and θ of each crony (at the start of every iteration) to be facing the center of the model. The total time per iteration in NetLogo was set to 125 ticks, which was enough time to get at least 95% of the cronies back to the home zone in a normal path before the traits were implemented. We run the simulation starting with 500 cronies, and 500 food units were generated per iteration.

3.1 100 Iterations

We ran the simulation 8 times, where the results can be seen in Fig. 2. In all of the runs, we saw that each trait was present in approximately 40 % of the cronies after the first 5 iterations. The total population decreased over time in all 8 runs, giving insight that the total population was not in equilibrium with the food units, and there was **overpopulation**. The population appeared to continue to decrease at the end of the runs, which meant that a longer number of iterations was required to achieve population equilibrium, where the total amount of cronies should remain within a small range.

We saw that the most successful trait was speed, where all cronies evolved to have this trait before the 50th iteration in every run. This was likely caused by the faster cronies collecting food before those who did not move as fast. Any crony that did not possess the speed trait died at the first half of the simulation runs. We can see how successful this trait was by the trait mean values, wherein every run, speed ended up being the highest mean value, indicating that cronies evolved this trait significantly more than the other traits. We can conclude by this that the initial **dominant trait** was speed.

Another trait that appeared to be very successful was violence, wherein every run except (1), most of the cronies evolved to have this trait before the simulations were finished. None of the 8 runs achieved a full population with a violence trait, but most of them appeared to head in that direction. Perhaps with longer simulations, we could see the entire crony population becoming violent and stealing food from each other. In run (1) the violence trait appears to have spread slower. The main trait value for violence in that run is lower due to less population with this trait, but it followed the same growth pattern as the other graphs, which leads us to believe that after more iterations, the violence trait would grow in this run.

The orientation trait appeared to have **failed**, since in all runs except (2) and (4), the mean value for the trait was negative. In runs (1), (5), (6), and (8), not even half of the population evolved to a positive trait value to try orientation at any point of those runs. It

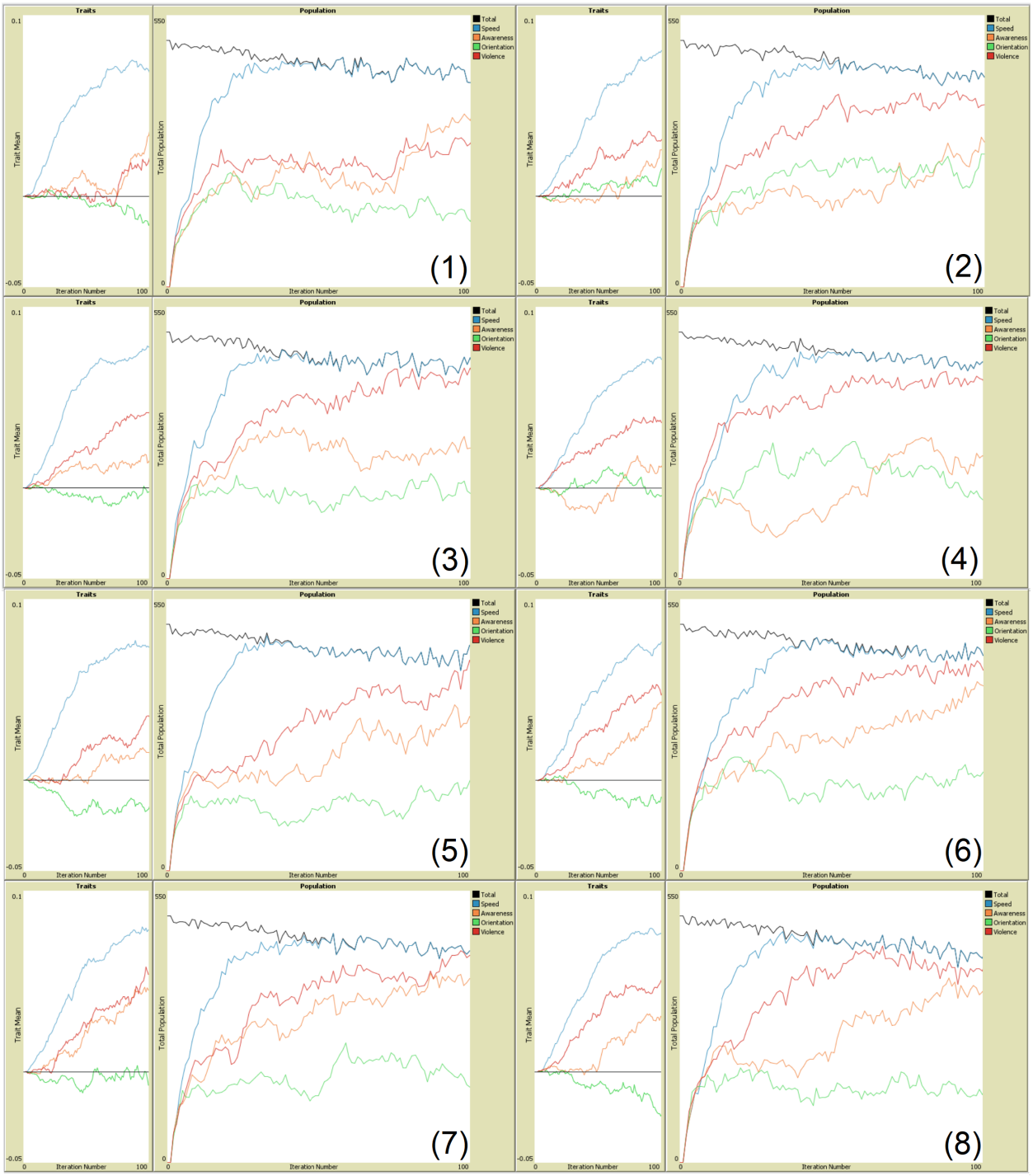


Figure 2: Output of 8 different simulation results after 100 iterations, labeled 1 through 8. Each output consists of two graphs, where the graph to the left corresponds to the mean trait parameter of each crony; and the one on the right corresponds to the number of cronies with said parameter. The color coding is blue for the speed trait, orange for awareness, green for orientation, and red for violence. The black lines determine the value 0 for the mean graph and the total number of cronies in the second graph.

appears as if the cronies did not need orientation since they started facing the center at the start of the iterations and they did not tend to vary their path enough for them to need the trait.

Finally, the awareness mean value was positive at the end of all 8 runs, indicating that the trait was useful to adapt to the cronies' environment. This trait was present in the majority of the cronies at the end of the runs, and except for run (4), it looked as if all the cronies would eventually inherit this trait. We see this since the number of cronies increases at the end of those runs. In run (4), we see that the cronies have a much higher orientation trait than awareness, but then switch, indicating that orientation was not needed for survival, and since the loss of this was the downside of awareness, those who had good orientation died to allow those with better awareness to expand within the total crony population.

We conclude by this analysis of the traits that speed was **successful**, violence and awareness were slightly less successful but still necessary for their survival, and that orientation was not a good trait for the cronies to evolve with to survive and reproduce. We also learn that longer simulations are required to acquire better data and see if the traits eventually expand to every crony or if it disappears.

3.2 Longer Simulation

To look deeper into the traits and how the total number of cronies evolved, we simulated a run for 1000 iterations, which can be seen in Fig. 3. The first thing we saw was that the population count **stabilized** after approximately 200 iterations. The total amount of cronies oscillated about (450 ± 25) cronies once it was in equilibrium, which is smaller than the number of food units spawned per iteration. If all the food units were collected by the end of each iteration, there would be one crony spawned for every other that died, keeping the total population number constant. This tells us that the cronies did not evolve to look for any food they could find, but rather evolved to fight the other cronies ignoring food units that were not found after an iteration finished.

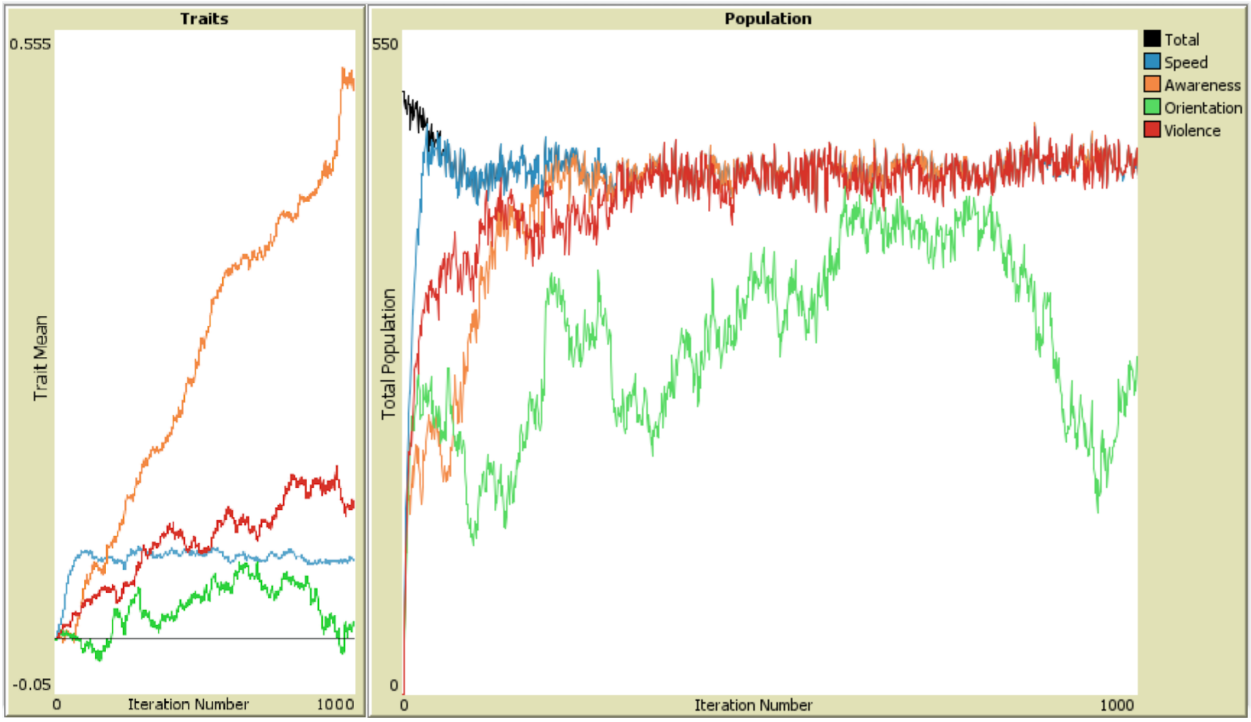


Figure 3: Output of the simulation results after 1000 iterations. It consists of two graphs, where one to the left corresponds to the mean trait parameter for every crony; and the one on the right corresponds to the number of cronies with said parameter. The color coding is blue for the speed trait, orange for awareness, green for orientation, and red for violence. The black lines determine the value 0 for the mean graph and the total number of cronies in the second graph.

Looking into the traits, we found that the speed trait is constantly increasing during the first 100 iterations, but it later **stabilizes** to a value as we can see in the mean value graph. This is perhaps because they found the limit at which they can run fast ensuring that they get to the home zone before the iteration finishes. This is a perfect example of a trait reaching equilibrium, where the trait expands to the whole population (in this case at the very beginning of the run), and reaches the perfect point to allow the advantage of the trait to be the most beneficial against its disadvantage without pivoting up or down.

The orientation parameter among the crony population appears to oscillate, while the mean trait value is positive for most of the run. By the main value almost achieving the same value as the speed equilibrium value we can assume that **two breeds** was created. One with an orientation trait and a high awareness value (enough for the mean of all cronies

to be almost as big as the speed main value); and one without the trait. These two breeds appear to fight each other and those with orientation traits dominate in number throughout the run. Nevertheless, both of them coexist and none of them die after 1000 iterations.

The violence trait takes a longer time to spread among the population than the speed and awareness trait, indicating that the trait was not as useful as the others at the beginning of the run. This could be caused by the low probability of the trait acting when the trait value is low. Once the probability was considerable, all the cronies started inheriting violence and interacting with each other. The cronies used this trait as a new way to collect food, contributing to their survival.

Finally, looking at the awareness trait, we find that it took a long time to spread among the cronies than the speed trait, but the mean trait increased significantly, achieving a higher value than any other trait had achieved in any previous run. This constant increase indicates that the trait's disadvantage is not an obstacle to the development of this trait. This trait became more important than speed since it allows the cronies to collect food from a further distance. We can conclude that this is the **dominating trait** since it has a steady increase over the other traits, even if it spread slower than speed.

4 Conclusion

In this paper, we created a simulation to model natural selection and trait evolution using NetLogo, creating four possible traits that the species referred to as cronies, could develop. Multiple behaviors were found by conducting multiple runs. The population number showed signs of overpopulation and equilibrium at different points of the simulations. The first trait, speed, showed to stabilize after a certain time, reaching a trait amount equilibrium. The next trait was awareness, which demonstrated dominance over the other traits. On the contrary, the third trait orientation showed to have failed to be completely useful in that specific environment, since a considerable amount of the crony population could survive without

it. This trait created two breeds that coexisted with each other oscillating in population number. Finally, the last trait was violence which allowed the cronies to interact with each other by giving the option of stealing food. This trait was successful giving the cronies a new way to acquire food besides collecting it.

This showed us multiple behaviors that can be found in different species in the real world. Whether it is different breeds of the same trait, one trait that dominates and overgrows those who do not possess it, or a trait that does not work for a specific environment, species have shown these behaviors throughout history. We can conclude by these behavioral examples that the simulation succeeded at simulating natural selection of a single species and trait evolution, and how these traits affect the model over generations.

5 Acknowledgement

I'd like to thank The College of Wooster Computer Science Department for the tools needed for this report, Including Dr. Guarnera, and the Teacher Assistants Pavithra Reddy, and Brandon Charles for helping me with my setup and procedure.

References

[1] N. Geographic, "Natural selection," 2019. Accessed on April 4th 2022, https://www.nationalgeographic.org/encyclopedia/natural-selection/#:~:text=English%20naturalist%20Charles%20Darwin%20developed,On%20the%20Origin%20of%20Species.

[2] T. L. Vincent and J. S. Brown, *Evolutionary game theory, natural selection, and Darwinian dynamics*. Cambridge University Press, 2005.

[3] U. Wilensky and W. Rand, *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. Mit Press, 2015.

[4] "Netlogo." http://www.netlogoweb.org/. Accessed: May 5th 2022.

[5] Primer, "Simulating natural selection," 2018. Accessed on February 24th 2022, https://www.youtube.com/watch?v=0ZGbIKd0XrM.

[6] B. Glymour, "Wayward modeling: population genetics and natural selection," *Philosophy of Science*, vol. 73, no. 4, pp. 369–389, 2006.

[7] A. Kleinman, "The basic science and mathematics of random mutation and natural selection," *Statistics in Medicine*, vol. 33, no. 29, pp. 5074–5080, 2014.

Appendix

This study was inspired by Primer's Youtube series on natural selection and trait evolution [5]