Ecology

I needed a topic for my senior project when I was an undergrad. I decided to reinforce my knowledge of C++, while working with genetic algorithms. Normally, when you are developing a genetic solution to a problem, you are concerned with the evaluation or fitness function. As I was driving on my long commute to night classes, I thought how best to determine fitness if not survival. Then a multi-level ecosystem grew in my mind. I had studied the Lotka – Volterra predator prey relation, differential equations determining the two population levels by their interactions. Well, why not expand on that idea without using any differential equations? Why not breed herbivores and carnivores using the genetic operators: recombination and point mutation? Since I had herbivores on my mind I figured they needed something to eat, so I added a plant layer too. Then those plants needed to be supplied with nutrients, so a fourth layer was added. When any organism dies, its energy flows to the nutrient layer. During eating of plants, or of herbivores, there is waste, as there is during the birthing process. Those also supply energy to the nutrient layer.

Since three of the trophic layers are represented by individuals, an object oriented design was a great fit. I was ready to start writing down all of my ideas by the time I arrived at school. The inheritance path was roughed out in a few minutes, with some notion of the methods each organism would require. Organism was my base class from which Plants inherit directly. I created an Animal class which inherits from Organism, since Herbivores and Carnivores have much in common. At each level I added chromosomes, so by the time you got to Herbivore or Carnivore there were three chromosomes in each. Carnivores and Herbivores have different behaviors, due to the various genes on their various chromosomes, determining how they react to certain situations. Plants need their own means of defending themselves, so their chromosomes have genes which determine poison levels and spine length. The Animal chromosome is mostly filled with genes used to determined movement characteristics. The Organism chromosome has genes which determine energy uptake, and how much to return to the nutrient layer. I had roughed out my Senior project program before that evening’s class began. The next few weeks were used creating those objects, and adding methods to the various classes.

The senior project was spread over three quarters, but I had chosen a complex program to write. It went through many revisions as I reworked my ideas, and added Windows resources. When I turned in the working copy it still had a few glitches but it was good enough to gain an A. It demonstrated my mastery of most of the tools of C++. It also showed me a glimpse into how an ecology can fail, as the various trophic levels’ population counts gained dynamic stability, then something would upset the balance and the system would crash. As I reworked the application I found a memory leak I could not tame. I knew exactly what the problem was and where it was, but could not get C++ to perform the complex delete needed, due to the multilayered inheritance tree.

So I took another tack and crafted my own memory management scheme. I studied my population logs and found the largest populations of each trophic level. Then I allocated space from heap memory for the maximum number of each organism. Once all that memory was preallocated, I would reuse it after an organism died. The memory for all of the objects is allocated for the life of the simulation. The only thing that changes is whether it is linked into the living population, or linked into the free pool. When creating a new organism, the links transfer from the free pool into the living population and each of its variables get reinitialized. The system will crash if too many objects of any given type are used, due the fact that a fixed number are allocated, and there is no provision to add more. I've adjusted the initial numbers higher and it works just fine.

https://en.wikipedia.org/wiki/Lotka–Volterra_equations

https://en.wikipedia.org/wiki/Genetic_algorithm

http://www.obitko.com/tutorials/genetic-algorithms/

https://www2.palomar.edu/anthro/mendel/mendel_1.htm

https://en.wikipedia.org/wiki/Mendelian_inheritance