About the Genetic Algorithms Demo

The program GA.html is a simple demonstration of the genetic algorithm, written in JavaScript.

Brief Instructions: The red things are Eaters, which eat the plants (the green things). A generation or "year" lasts 250 "days," after which a new generation of eaters is created by "breeding" the previous generation. As generation follows generation, the Eaters might evolve to become better eaters. Try putting things into fast forward by setting the Run Speed to "Yearly Stats Only." After a while, reduce the speed and see how the Eaters' behavior has changed. Use the "World Design" menus to give the eaters different environments, and see how the environment affects the behaviors that they evolve.

More About the Genetic Algorithm

The program demonstrates the *genetic algorithm*, in which methods analogous to the process of natural evolution are applied to solve problems on a computer. This "artificial evolution" uses reproduction, mutation, and genetic recombination to "evolve" a solution to a problem.

The genetic algorithm can be applied to many different types of problems, but this demo uses it to evolve simulated "organisms" called Eaters in a simulated world that contains simulated plants for the Eaters to eat. I stress the word "simulated", but even that word really goes too far. The program is really just a kind of metaphor for natural evolution in the real world. You can judge for yourself the extent to which that metaphor might be valid.

When you first load the web page, you will see a world containing 250 plants and 50 Eaters. The plants tend to grow in clumps. The Eaters will display random-looking, undirected behavior, but sometimes an Eater will eat a plant. (It does this merely my moving to the position occupied by the plant.) At the end of each year, a new world is produced, containing a new generation of Eaters. The new Eaters are produced as copies, with modification, of Eaters from the previous generation. The more plants that an Eater has eaten, the better its chances of producing "offspring" in the new generation. Differential reproduction, based on "fitness" defined as the number of plants eaten, would by itself tend to increase the average fitness of the population. The random modifications introduced during reproduction allow new behaviors to be tested and, if they turn out to be advantageous, to spread through the population in later generations. There are two types of modification: "mutation", in which random changes are introduced as an Eater is copied, and "crossover", in which a new Eater is made by combining parts of the genetic makeup of two Eaters from the previous generation.

This is all you really need to know to start playing with the program. The rest of this page contains the details you need in order to understand what is going on.

More About the Program

The Eaters' world is divided into little squares. Each square can hold an Eater or a plant, or it can be blank. A square cannot be occupied by two things at once. An Eater can "see" the single square just in front of it. (The front is the pointy end of the T-shaped Eater.) It can see one of four different things: another Eater, a plant, a blank space, or a wall. In addition to this external information, the Eater has an internal memory that contains a number between 0 and 15. This number is called the "state" of the Eater. At each time step, the Eater can perform one of the actions:

```
1. Move forward one square
2. Move backwards one square
3. Turn in place 90 degrees to the left
4. Turn in place 90 degrees to the right.
```

It can also change its state, by changing the number in its memory. If it tries to move into a wall or onto a square that already contains another Eater, it will not be allowed to move; however, it can still change its state. If it moves onto a square containing a plant, it "eats" the plant and scores a point. Depending on menu settings, the plant might immediately grow back somewhere else. At the end of a year, the "fitness" of the Eater depends the number of plants it has eaten, plus one. (The 1 is added to avoid having a fitness of zero.) The more plants eaten, the higher the fitness.

Now, how does an Eater decide what to do on each turn? It bases its decision on two things, its current state and the item that it sees in front of it. Its behavior is completely determined by a set of rules that tell it what to do for each possible combination of current state and item-in-view. All the Eater does is follow its rules. The rules differ from one Eater to another. In fact, the Eater's rules, which completely determine the behavior of the Eater, make up the Eater's genetic endowment, what we will call its "chromosome." The chromosome consists of 64 rules (one for each combination of one of 16 states and one of 4 items-in-view). Each rule specifies two things: an action and a new state. The chromosome can be considered to be just a list of 128 numbers.

Here is what happens at the end of a year: The average fitness of the current population is computed. A new population is created by making copies of the Eaters in the current population. An Eater's chance of being copied depends on its fitness. Eaters that have fitness much below average are likely not to be reproduced at all (although they always have some chance of reproduction.) Eaters with high fitness are likely to be copied several times. Then a mutation operation is applied: Each of the 128 numbers in each chromosome has a chance of being randomly changed. This chance is 0.1% by default and is controlled by the Mutation Probability menu. Finally, a crossover operation is performed: Pairs of Eaters in the new population are chosen at random and have a certain probability of being "crossed over." This probability is 80% by default and is controlled by the Crossover Probability menu. Crossover here means that the chromosomes exchange some genetic material; a random position between 1 and 128 is chosen and all data in the chromosomes after that position are swapped between the two chromosomes.

Note that the default mutation rate, 0.1%, seems very low, but since it is appied to each of the 128 numbers in each chromosome, it means that there is about a 1 in 8 chance that a given Eater will be mutated in some way. This low rate of mutation seems to give better long-term evolution of the Eaters, but they might evolve faster with a higher mutation rate, at least

at first. You might try hitting an evolved population with a burst of high mutation for one or two generations to see what that will do.

After the reproduction step, the new Eaters are placed in a new world with a new set of plants, and a new year begins. The Eaters are initially in state number zero and are facing in random directions.

The average fitness and the highest individual fitness for the previous year are displayed below the "World." Below that is a table that shows these statistics for certain previous years. Information is displayed in this table every year for the first 10 years, then every tenth year up to the 1000-th year, and every 100-th year after that. Also, if a new high average score is achieved in a given year, data for that year is added to the table, even if it would be skipped otherwise. New high average scores are marked with an asterix. After the first 100 years, this window also shows the "average average score" for the previous 100 years.

Various aspects of the world are controlled using the various menus, which appear to the right of the world. Mutation Probability and Crossover Probability were mentioned above. Most of the other menus are either self-explanatory or best left to discovery through exploration. Whenever you make a change in this menu, it becomes operative at first opportunity. Changes that affect the initial state of the world, such as the "Number of Plants", will be applied immediately if the world is on Day 1 of Year 1; otherwise, they will be applied at the end of the current year. Some settings, such as the "When a plant is eaten" menu, can take effect even in the middle of a year.

The "Fitness score is" menu needs some explanation. It is designed to test the idea that the genetic algorithm works better if it doesn't try to work *too* quickly. For example, if you set this menu to "Cube of Number of Plants Eaten," then the fitness score of an eater is n^3, where n is the number of plants that the eater has eater in a year. This greatly increases the relative fitness of eaters that have eaten a lot plants, compared to eaters that have eaten only a few, and that means that the more successful eater will tend to have a lot more offspring. The better eaters tend to quickly take over the population with copies of themselves, which might seem like a good thing—but it means that genetic diversity can be lost too quickly. The genetic algorithm might work better if the better eaters have less of an advantage.

The "Start from Scratch" button will start over with a new, randomly generated population of Eaters, using the current settings of the menu. All of the collected statistics from the previous run are discarded, and the world starts over from year one.

The speed settings in the Run Speed menu should be fairly obvious. Note that the "Yearly Stats Only" speed is designed to make the years go by as quickly as possible. When this speed is selected, the contents of the world are not shown. A possible technique is to let the program run at this maximum speed until it looks from the average scores like something interesting has happened. You can then switch back to one of the other speed settings and try to see what new behavior the Eaters have "learned."

David Eck