•••*BUGS*•••

Bugs may not be distributed for profit and any distribution must include the full documentation.

This is as close to a hypertext document as I could make it with MS Word 4.0. The entire document is color coded. Anything in color represent a specific point like a variable or an option. Several color-coded words can be looked up elsewhere in the document. The color codings are as follows:

Blue**:** **large sections of the document**

Red**:** **Menu bar names**

Green**:** **Menu choices**

Cyan**:** **Program variables**

Note that many of the menu choices ARE various program variables as well, but are

separated so you know what you can access from the menubar and what you cannot.

A Brief History:

This not only my first attempt at an artificial life simulation but is also my second serious C program, so please be kind. :) This project began when I became addicted to artificial life in the dusty shelves of a University bookstore. I have since written a couple of variants of John Conway's Game of Life and Stuart Kauffman's Boolean Networks to which I gave a unique twist. I read about David Ackley's AL and decided that it was a fairly simple idea that I could probably implement. Bugs is much different from AL however, mainly in that the bugs have no neural network brains. In a later version I tried to give them neural network analysis of their environment and it proved to be above beyond my ability at the time. I am now working on other things and doubt I return to Bugs to finish the neural networks.

What Bugs does:

Bugs creates a rather small square grid world of forty by twenty-eight cells. The world has three things in it. Rocks, plants, and bugs, not to mention abundant open space. The bugs have limited visual input dependent on the direction they are facing and what rocks they can see around. Based on this input, they react with very little sense of randomness at all. Instead they implement the use of a complex algorithm I have designed for optimal input/output reactions. This is where I was going to put a learning neural network, but as it is now they use my design which is detailed below. A bug can do but a few things in any one turn. It may move forward, it may rotate one eighth of a turn either direction, it may eat assuming it is on top of a plant, it may reproduce if it is able to, or it will die if circumstances deem it so. In addition it may be bumped to an adjacent cell if it is on top of another bug. This will happen extremely rarely as is detailed below.

The Controls:

Bugs gives you a whole lot of control over the environment and the simulation. It may take a while to get used to what each of the variables corresponds to and what exactly it changes. Note that none of the **Gene** menu dialogs or the **Statistics** dialog will come up until a simulation is begun. Here are the commands organized by menu:

Under the Apple Menu:

The typical **About Bugs** box will show a pretty picture. :)

Under the File Menu:

New Map will regenerate a completely new map. This map will have a different array of randomly distributed rocks and plants at its onset and will correspond to the program defaults of the Settings menu, unless they have been altered by you prior to executing the **New Map** command.

Pause will halt all execution of the program until it is selected again.

Step will step forward one timestep at a time so you can see exactly how an individual bug is reacting to its environment. In a single timestep each bug interprets its inputs and executes and output exactly once. Also, plants will be grown on a rate per timestep value. This option is dimmed if Pause is not turned on.

Quit quits Bugs.

Under the Genes menu **you can** pull up a graph of the complete gene pool for any particular gene. The taller bars represent more individuals with that value for the given gene. The blue-marked graphs show percentage genes. The red mark designates the point between fifty and fifty-one percent. The purple-marked graphs shows energy levels from 0 to 1600. Each pixel-wide mark designates four energy levels. The green-marked graphs show energy levels from 0 to 50. Each mark is a separate energy level.

Under the View menu:

Fast Calculation turns continuous window updating on and off. The simulation runs MUCH **faster with this option turned on.** All of the genepool graphs, **Statistics** dialog, and **Environment** menu options will operate normally with **Fast Calculation** on. You will notice that the window doesn't clean up after closing these dialogs because the main window isn't updating. With or without any dialogs having been pulled up, when you turn **Fast Calculation** off the entire window will update to the present environmental configuration.

Show Statistics pulls up a dialog box with four basic statistics about the present simulation. These statistics are self-evident. Note that **Total Births** includes the initial bugs.

Under the Settings menu:

Initial Population sets the number of bugs at the beginning of the beginning of the simulation.

Maximum Population set s the maximum number of bugs that may ever be alive at one time.

Initial Energy sets the energy amount the initial bugs start with.

Energy to Reproduce sets the energy level the initial bugs must have in order to reproduce.

Reproduction Energy Cost sets the amount of energy lost by reproducing.

Forward Energy Cost sets the amount of energy lost by moving forward into a new cell.

Rotation **Energy Cost** sets the amount of energy lost by rotating an eighth of a turn.

Food/Rock Dispersal sets the amount of rock and plants that are strewn onto the map at the beginning of a simulation. The ratios in the hierarchical menu are a representation of the the fractions of the

entire map covered with each of the three possible cell states at the beginning of a simulation.
Note that any changes you make in the **Settings** menu will NOT take effect until you generate a **New Map** from the File menu.

Under the Environment menu:

Food Growth Rate sets the percentage possibility that each empty cell on the map will grow a plant each timestep.

Food Value sets the amount of energy rewarded to bugs when they eat a plant.

Note that any changes you make in the **Environment** menu will take place immediately thus effecting the present simulation.

Here are the genes that are displayed from the **Genes** menu. Each bug has an array of these genes which essentially define their phenotype and genotype:

ForwardAfterEat : percentage of the time that a bug will go forward immediately after eating if no plant is in sight.

ForwardPersistance : percentage of the time that a bug will persist in moving forward provided no plant is in sight.

RotatePersistance : percentage of the time that a bug will rotate one eighth of a turn if no plant is in sight.

FoodPersistance : since bugs never walk into a cell that other bugs are facing, bugs will switch back and forth from a plant without entering it if another bug is doing the same thing. This gene determines about how long a bug will "argue" with another bug over one plant before moving on elsewhere.

EnergyToReproduce : the level of energy that is necessary to asexually produce one offspring in an adjacent cell.

ReproductionCost : the energy cost of asexually producing one offspring.

IntitialEnergy : the energy level that a "newborn" bug is equipped with.

ForwardCost : the energy cost of moving forward one cell.

RotateCost : the energy cost of rotating one eighth of a turn

NoFoodPersist : the level of energy below which a bug will not "argue" over a plant with another bug at all and move onto other plants immediately.

How it works:

Here is exactly how each bug works. It can see a ninety-degree view up to three squares distant. However, its view can be blocked by rocks which get in its way. In such an event, it cannot see what is behind the rock. After assessing its environment it heads directly towards the closest plant it can see. However, of higher precedence than food is avoiding other bugs. If it sees a bug directly ahead OR if it sees a bug in a cell adjacent to the cell directly ahead AND such a bug is facing the square adjacent (in other words, if a bug already occupies its forward cell OR if a bug MIGHT enter the forward cell in the next timestep) then it will refuse to go forward and

rotate towards the next nearest plant instead. In the event that it is on a plant, it spends that turn eating. In the event that it cannot see any plants he randomly determines a course of action. The possible actions are as follows:

(only happens if on a plant)
reproduce (automatically happens only when energy is above the **ReproductionEnergyCost** gene.

Note that turning takes a timestep. If it is facing north and wants to go north-east, it must first rotate and then go forward.

For the sake of clarification, not only can bugs not see over or around rocks, they also cannot, under any circumstances, go over a rock.

Energy and Reproduction:

The bugs have an energy level which represents their health. This decreases when they rotate, move forward, or reproduce and increases when they eat a plant. If the energy level drops to zero or below they die and a plant grows in their spot. Think of it as fertilizing the soil or think of it as a nutritious corpse, I really don't care which one. If their energy goes over a certain limit they will reproduce. If there is a blank space directly adjacent to them, a new bug will spontaneously appear in one of the available spaces. If no room is immediately available, a bug will appear in such an adjacent cell as soon as possible.

Plant Growth:

Plants grow on their own, randomly appearing in blank spaces every once in a while, the rate of which is directly dependent on the Food Growth Rate variable under the Environment menu.

The following diagrams show how a bug can see its environment. The numbers correspond to the position in its inputarray which will either receive no information if the cell is empty or out of view (out of view would constitute that the bug's vision is partially obscured by a rock), or else, information on whether the cell is a rock, a plant, or a bug. These inputs will also showed the exact direction any bugs in sight are facing. In all cases the bug in question occupies cell 15.

facing a cardinal direction (N, E, S, W):

example facing north

```
0    1    2    3    4    5    6
     7    8    9   10   11
          12   13   14
               15
```

facing a diagonal direction: direction(NE, SE, SW, NW):

example facing north-east

```
     0    1    2    3
     7    8    9    4
     12   13   10   5
          15   14   11   6
```

Further Notes:

You will inevitably notice that certain bugs get caught in endless loops of turning towards and away from a food source if another bug is doing the same thing. Very quickly here is what is happening: Either bug is facing one off from the food so he rotates towards it. It wants to go forward but sees the other bug facing into that square so he rotates away instead, at which point the cycle restarts. This would prove to be a fatal to circumstance to the bugs. They would rotate busily until on of them died. Therefore the genes FoodPersistance and NoFoodPersist deal with this nasty little problem.

The bugs are inset with the natural insistence on avoiding each other as was previously stated. There is one possible case where two bugs might enter the same cell on the same turn. It is the following. Xs are blank, the R is a rock, and the numbers are two
bugs:

```
X   X   X   X   X
X   X   X   X   X
X   X   X   X   X
X   1   R   2   X
```

If bug #1 is facing north-east and bug #2 is facing northwest, they can't see each other. This is the only such case where neither bug can see an oncoming bug. If they both decide to move forward, one of the bugs upon entering the space will be bumped to a randomly chosen adjacent blank space. If there is not adjacent blank space because there are rocks everywhere and BOTH of the spaces they entered from grew plants during the turn (This won't happen in a billion million years unless you set the FoodGrowthRate to some astronomical number) then the bug that would have been bumped will die. Think of it as losing a battle for space, or perhaps just getting crushed under the other one's weight. (I realize this is completely impossible for Earth-bugs, but it happens to my bugs, so there).

Known "bugs":

It doesn't always right-justify the Statistics dialog perfectly, especially the total deaths figure I've noticed. Beats the hell outta me what's wrong with my algorithm but it's perfectly legible as it is so live with it.

Keith Wiley Feb. 7, 1995
keithw@wam.umd.edu

102 Keith Rd.
Carrboro NC, 27510
USA