Assignment #HW2

CSSE490: Bio-Inspired Artificial Intelligence

Prepared and submitted by:

Bruno Christensen

Collaboration and resources:

I worked with another student: Jordan Ansari

Resources I used to complete this assignment (websites, textbook, friends, etc.):

Homework 1

Stackoverflow for JFrame

When complete, please remember to commit a PDF to your repo!

(File-> Save As->PDF)

a. An explanation of the experiment (including parameters used)

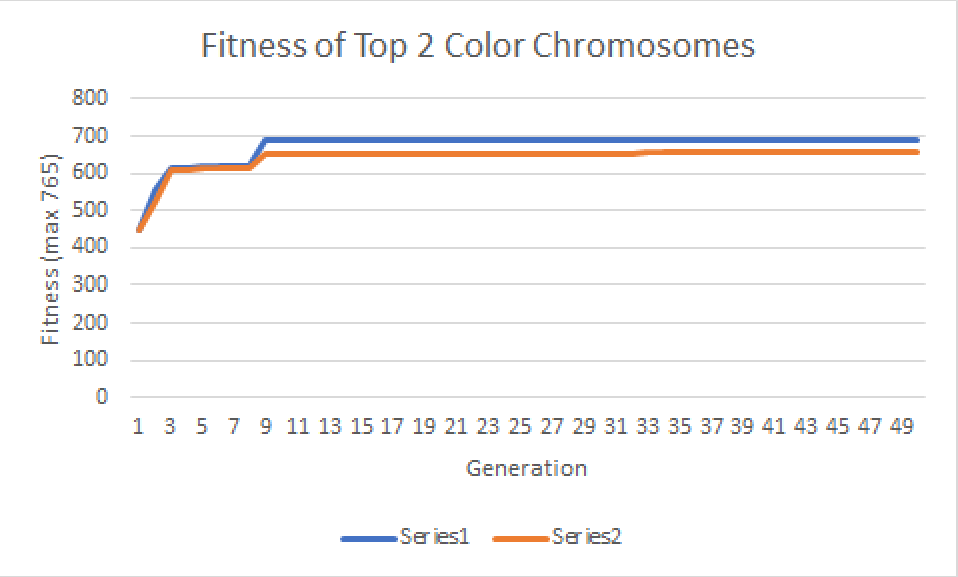
This program was designed to evolve a series of 24 bits to reach an RGB value of a color. The methodology for evolution was as follows: The top 2 colors were considered elite and unchanged. The bottom 2 colors were culled and replaced with entirely new random colors. The remaining 10 were mutated from chromosomes 2-7. The mutation rate increased as the significance of the bit decreased. That is, the first bit in the R value is much less likely to mutate than the last bit. This created a population which experiences fewer substantial changes, but still has 2 completely new members every generation.

Population size:	*14*
mutation rate:	*Depends (See above)*
crossover:	*FALSE*
elitism rate:	*2/14*
selection method:	*TRUNCATION*
max generations:	*User Defined (50 for our experiment)*
target expression	*“000000001111111111111111”*

b. A hypothesis (might just be a guess)

We will be able to evolve to a color that is very similar to the color specified. While the chromosome may not exactly reach the desired value due to the way mutation occurs, the visual appearance of the color will likely be similar.

c. A plot of fitness over time and the best final expression and any observations



Best final expression: "000101111111010011010110”

d. What if anything you can conclude and anything you learned or affirmed by doing so?

We were able to get close to the desired color, although there are still some imperfections. Conversion between hex, decimal and binary made some of the checking difficult, although it is obvious running this program in manual mode that the algorithm does pick up a pattern of what colors to trend towards. This method of different chances for mutations in each bit shows how species can trend towards a similar appearance- nearly all colors end up looking the same since their more significant bits are less likely to change. This fact is also true in evolution, where major features about a species rarely change, but smaller features do change.

Proposal:
Favorite Color Program:
1. Our program would be made to find a user's favorite color using evolution and input.
2. The program would be a .jar file that displays an interactive interface, allowing the user to find their favorite color.
3. This would help us further our HW1 task, as well as give us a new experience for different mutation.
4. Automation: A random color would be generated as the target, and 12 randomly generated colors would be created as the species. The fitness would be how close in value the colors are to each other, and the goal would be to mutate towards the target color.

Human-interaction: We'd first need to make a UI that displays 12 randomly generated colors in boxes with the hexadecimal values beneath them. Then, we'd need to add mouse compatibility so the colors could be clicked on, and display numbers 1-12 depending on the order they were clicked. We would be able to test this by printing out the order of selected colors to see if it matches up.

The bottom of the screen would display a "next" button if all 12 have been clicked, and create a new generation of colors. We'd remove the rankings and colors from the screen and display the new generation, and the cycle would repeat.

5. Bruno and I would like to work together on this.
Summary of the program:
Start with ~12 randomly generated genotypes, which get converted into a hexadecimal representation of a color. Display those colors on a screen (phenotype), and ask the user to rank them. The next generation could be made a few ways, but our thinking is to have the top 2 be considered the elite, the bottom 5 be culled, and the remaining 5 are used for mutations. Continue this for however many generations the user wants. The chromosomes would be split into the R, G, and B values, and the mutation chance for each bit would increase from left to right (to make significant changes less likely).
Extra features: see if the user favors a particular R, G, or B value (i.e. high red and low blue) and mutate based off of that.
Allow the user to choose how they'd like mutation (elitism or not, mutation value, etc.)