Rahma Ahmed 11/30/17

Genetic Algorithm for Stem Cells

**Background/Introduction:**

Cancer cells that are good at finding sources of nutrients are more likely to spread and increase in number over time. In this lab a StemCell is an extension of the CellWalker class. These are two kinds of cancerous cells that metastasize differently. A Tumor class is used to create an arrayList of cellWalkers and allow them to run and reproduce. This lab tests how a cell’s future generations can be more fit to continue their cell line if they are able to find nutrients. All cells have a likelihood of mutating which affects their lifespan and likelihood of reproducing. This lab, helps us further understand the genetic algorithm and the different aspects that affect a cell line’s future.

Methods/Design:

*Changes made to the CellWalker class:*

* Added a float called recordDist to keep track of how close a cell got to a source of nutrients and a float called fitness to keep track of how good a certain cell in a cell line is. The fitness effects where or not a cell line continues (very similar to lifespan).
* Boolean to keep track of whether or not a target has been hit. An integer for finishTime, how long it took a cell to hit a target. And color c to set the color of the cells’ nuclei.
* Fitness is set to lifespan in the constructor and color c is set to random red, green, and blue values between 0 and 255. FinishTime is set to zero and recordDist is set to 1000;
* The fitness function is essentially a cell’s likelihood of being in the next generation. If the recordDistance is less than one, it gets set to one. If the cell reached a source (hitTarget is true), the lifespan of the cell increases, it gets to live longer and it’s fitness value also increases.
* The mutate function applies a random force to the cell. It also decreases a cell’s lifespan and telomeres (capability of reproducing) and updates the position of the cell.
* checkTarget checks if a cell made it to any source. It initializes closestTarget as a float 700. It ten loops through the array of nutrients (source) to find which source is the closest to the cell. It then checks if the distance between the cell and the closest source is less than the recorded distance and sets recordDist to closestTarget. If closestTarget is less than zero and hitTarget is false, it sets hitTarget to true. Otherwise, if hittarget is false, finishTime is incremented by 1.
* Display is the same as before with a small change. The nucleus of the cell is now set with a variable, c, mentioned above.
* Run now takes in a float for the mutation rate. As long as the cell has not reached a source, step is called. Then an if statement is used to set the probability of a mutation occurring using the mutationRate that is inputted into run.
* getFitness returns the value of fitness.

*Tumor class*

* Added a float for mutationRate and an integer to keep track of the number of generations.
* Tumor takes in a value for the mutationRate. It then sets all the values of the parameters with generation equal to zero.
* Live loops through the arraylist of cells and checks if they have reached the target. It then calls the run function to make them move and display.
* TargetReached loops through the cell arraylist and checks to see if a cell had hit a target. If it has, it increases it calls the fitness function so that this cell has a higher likelihood of living longer and continuing its line. Otherwise it returns false.
* One small change made in run is that when a new cell is reproduced, it has the same color, c, value as the cell that produced it. Also, the generations variable increments whenever a new generation of cells are added.
* Get generations returns the value of generations
* Fitness calculates the fitness of every cell.

*Main Sketch*

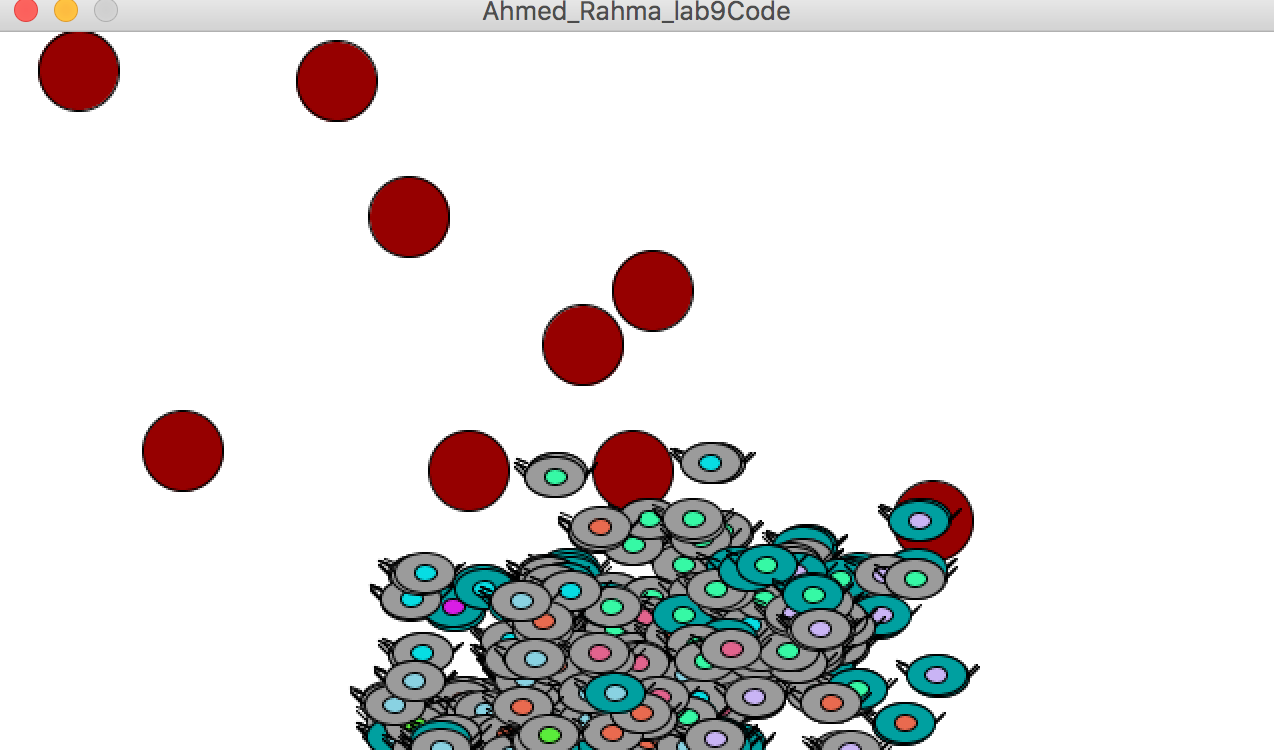
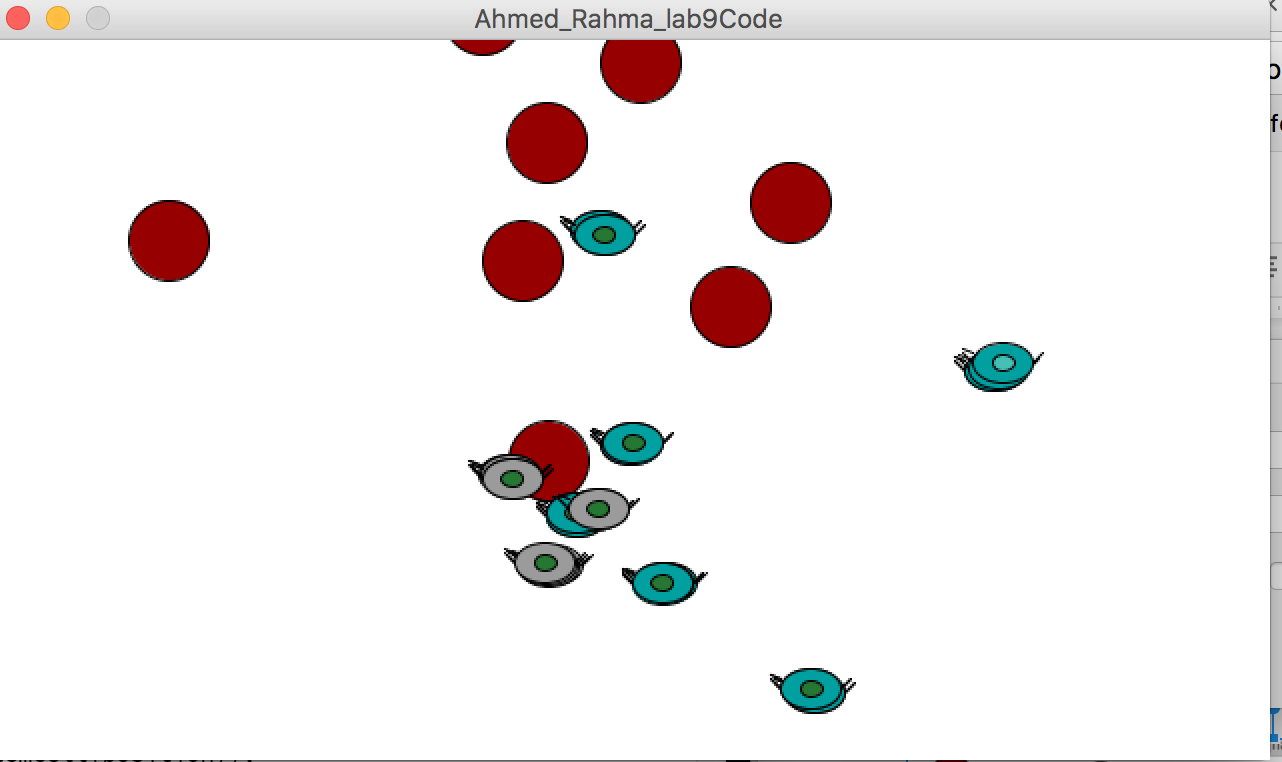
* Three new integers have been added: lifetime (how long each generation lives), lifecycle (timer for cycle of generation), and recordtime (fastest time any cell has reached a nutrient source). Also, source is now an array of PVectors.
* In setup, mutationRate, lifecycle, lifetime, and recordtime are set to 0.01, 0, 300, and lifetime respectively. tumorPop (the population of cells AKA the tumor) is initialized and the source array is initialized and filled with random PVectors with x values between 40 and width - 30 and y values between 40 and height - 100).
* In draw, a loop loops through the source array and draws all the sources of nutrients as red circles. It then displays all the cells in tumor using the function live. An if statement then sets the recordtime to the value of lifecycle if the tumor reaches a source and lifecycle is smaller than recordtime. Another if statement increments lifecycle so long as it has not yet reached the value of lifetime. Otherwise, a new generation is started by setting lifecycle to zero. Fitness is then called to give credit to any cells that have reached the target by increasing their lifespan and fitness. Run is the called on tumor to allow the next generation to be made by reproduction.

Results:

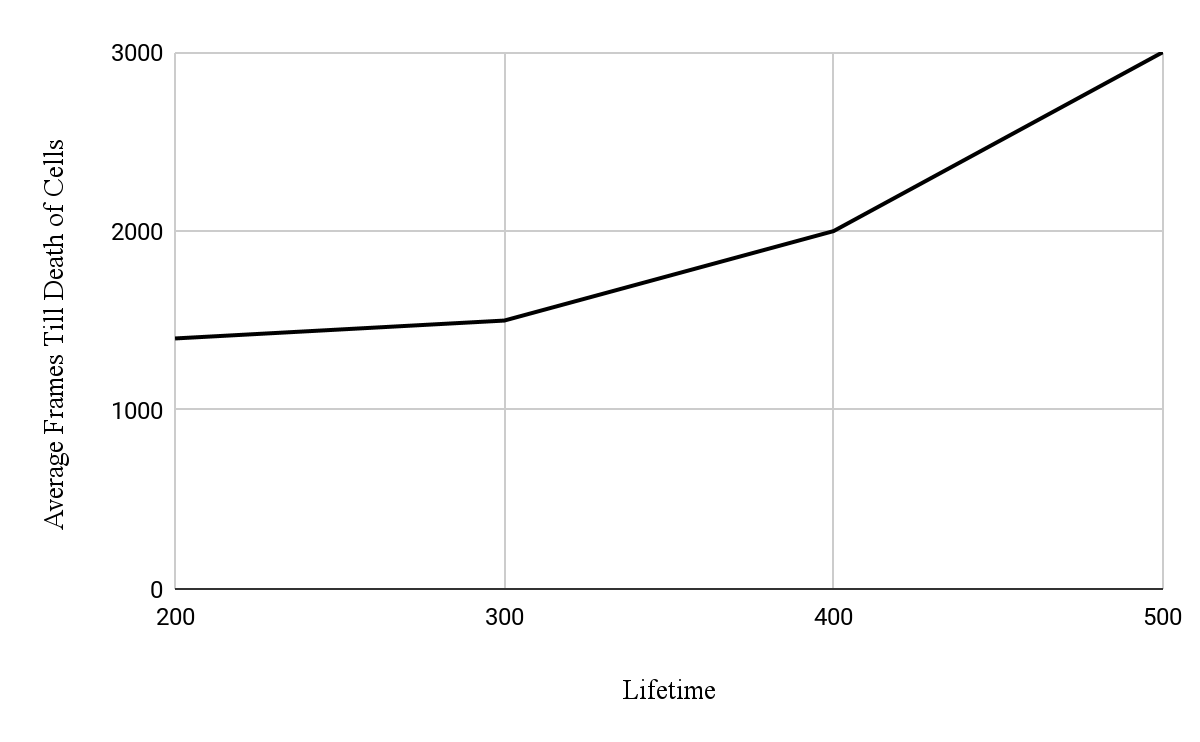
My code makes ten different cell lines that all come from 10 unique stem cells. Each cell line has its own nucleus color and symprob (probability of a stem cell in the line producing another stem cell). These cells move towards a source and have some random movement incorporated as well. Each cell has a 0.21 probability of being mutated. MutationRate and lifetime were varied to test the effects of both on the amount of time a tumor is alive for (average number of frames). Graph 1 shows that lifetime and average number of frames the tumor is able to stay alive have positive correlation. This is expected since an increase in lifetime gives a cell more time meaning it is more likely to reproduce more cells which will then reproduce even more cells.

Graph 2 shows that mutation rate has a negative correlation with the average frames the cells are alive. This makes sense for a similar reason that the results in graph 1 do. An increase in mutation rate means a decrease in lifespan. This in turn means less time for a cell to increase its lineage and therefore the tumor with the ten cell lines is likely to die off much quicker as the mutation rate increases.

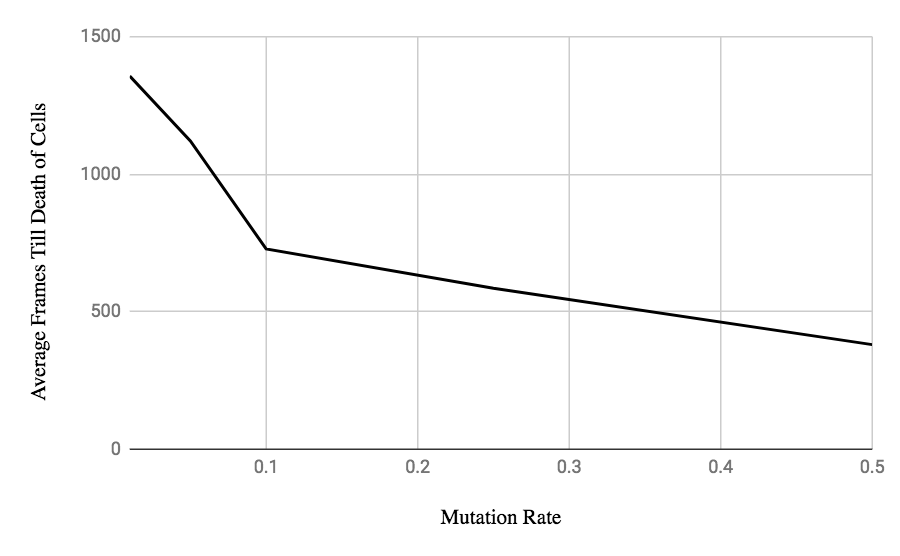
Lastly, preliminary tests show that a very small change in proliferation (+/-0.005 to 0.01) had a dramatic effect on the tumor’s overall time alive. Though I was not focused on this factor, and do not therefore have enough data to show something of that much value, these preliminary tests that I quickly ran, surprised me because I did not expect the effect of proliferation to be much more dramatic than that of mutationRate changes since mutationRate changes the lifespan of cells. I expected that if a cell has a shorter lifespan then it also has a shorter amount of time to proliferate (reproduce).

**Figure 1:** The cells reproduce and move towards the sources of nutrients shown in red. Image on the left is the simulation after about 500 frames. Image on the right is the simulation at around 1200 frames. The right image shows that at the end, there are only to cell lines left which eventually also die off.



**Graph 1:** Shows the average number of frames the tumor was able to stay alive for three trials at lifetimes ranging from 200 - 500 with mutation rate consistently at 0.01. This graph shows a positive correlation between lifetime and average frames alive of the cells.



**Graph 2:**This table shows that mutation rate has negative correlation with the average frames the cells are alive. The lifetime for these simulations was consistently 300.

Conclusion:

This model shows how different variables affect the possibility that a cell’s lineage continues for many generations or dies off quickly.

Next steps:

In the future, I would incorporate DNA to make the ocde a lot clearer than it currently is. A very interesting and helpful addition could be to make the fitness be more complex so that it is not only dependent on whether it hits target but also if it gets close to the target. I would also run extensive tests since most of these tests are preliminary and not really scientifically significant. Additionally, I would make my graphs show percent change so that it is a lot clearer that lifetime has a more significant effect than mutation rate on the cell lines.

Credit/Acknowledgements:

I used our textbook, Nature of Code, and I also worked with Lucy and we talked through the various parts of the project.

Citation:

Shiffman, D. (2012). Nature of Code.

Slides from class

Genetic Algorithm code posted on moodle