UPPSALA UNIVERSITY
DEPARTMENT OF MATHEMATICS
Fiona Skerman

Assignment 7
Spring 2020

Lab 6: Genetic Algorithms

The deadline for this sheet is midnight Wednesday 25th of May.

Please submit hand-ins on studium. All code should be included. Please feel free to submit videos illustrating your results where appropriate, via studium or uploaded elsewhere such as vimeo or youtube. You may work in groups of size 1-5, and only one group member needs to submit the assignment. State clearly the members of the group. This exercise will be covered in lab session on Thursday 12th of May.

6. Genetic Algorithms

For this problem we will imagine we have a painter robot similar to the robot which picked up cans in the lectures. We will use this robot to paint the floor of a room. To make it interesting, the painter starts at a random place in the room, and paints continuously. We will also imagine that there is exactly enough paint to cover the floor. This means that it is wasteful to visit the same spot more than once or to stay in the same place.

To see if there is a optimal set of rules for the painter to follow, you will create a genetic algorithm. You may write your own code from scratch or use `painter_play.m` or `painter_play.py` as starting points.

As inputs, this function receives

1 . A chromosome: A 1x54 array of numbers between 0 and 3 that shows how to respond (0: no turn, 1:turn left, 2:turn right, 3: random turn left/right) in each of the 54 possible states. The state is the state of the squares forward/left/right and the current square. Let $[c, f, \ell, r]$ denote states of the current square, forward square, left square and right square respectively. Write 0 for empy, 1 for wall/obstruction and 2 for painted. Note that $c \in \{0, 2\}$ and $f, \ell, r \in \{0, 1, 2\}$ so there are $2 \times 3^3 = 54$ possible states.

2. An environment: A 2D array representing a rectangular room. Empty (paintable) space is represented by a zero, while furniture or extra walls are represented by ones. Outside walls are automatically created by `painter_play()`.

The function `painter_play()` then uses the rule set to guide a painter, initially placed in the room with a random position and direction, until the paint can is empty. Note that the painter does not move when it tries to walk into a wall or furniture. The efficiency (total fraction of paintable space covered) is then given as an output, as well as the X-Y trajectory (i.e. the positions of the painter at each time step) of the painter. To see that the painter works, you can try passing it an empty room for an environment and a trivial chromosome. For example, a chromosome consisting of all 3s produces a kind of random walk. Now do the following:

1. Think of a simple strategy for the painter to cover a lot of space in an empty room. Describe this strategy in a few words or sketch it, but do not try to encode it in the chromosome. **(1 points)**

2. Create 50 random chromosomes in a 50x54 matrix, as well as a 20x40 empty room. Create a genetic algorithm to evolve this population over 200 generations, playing each chromosome several times and storing the chromosomes average efficiency as the fitness.

   You may choose any rule for picking the next generation from the previous one so long as it includes crossovers and mutation and that individuals with higher fitness are more likely to have offspring in next generation. (An example is to use single-point crossover with a mutation rate of 0.002 per locus per generation.) Plot the final set of chromosomes. Plot an example trajectory of one of the more successful chromosomes (or make a video). Is this what you expected? **(4 points)**.

3. Plot the average fitness in the population vs generation. You will likely see large sudden *jumps* in fitness, corresponding to strategic innovations. In your own words, write down two possible examples of an innovation that would increase fitness. **(2 points)**

4. Add some furniture to the empty room (about 100 square metres in total) and use one of your highly evolved chromosomes, and plot the trajectory (or make a video). How does the efficiency compare to that in an empty room? If the strategy fails, how does it fail? Now try running the genetic algorithm with your new furnished room from the start. How does the strategy compare to the empty room strategy? **(3 points)**