CS 461 Program report.

Timothy Hoang.

Generic Algorithm Implementation using Python. I chose List as my data structure for this program. List structure in Python provides me various tools that I can easily use to manipulate my bit strings as well as my data read from the Input file. It also provided me a way to control my max fitness as well as my average fitness scores. I can optimize my program more if I can parallel my functions. I actually got stuck at trying to splice up the DNA and making new children. The problem wasn’t trying to splice up the DNA but rather indicate which parent am I splicing up on. I got confused and it is still not resolve, so I went randomly choose a parent to splice. My Average fitness decrease dramatically after generation 900. It seems like after generation 1200th, my algorithm does not have fitness anymore, or rather set to 1. It does not seem efficient, especially since I did not get the right number report as the sample. So about 1200 generations were needed before we can realize if the program was efficient or running right. Speculate 500 generations before realizing something was off. The program ran slow because of many iterations it had to go through, not a very enjoyable testing experience. I do feel like if I interpret Dictonary (Map) structure into this program, it might help me see clearer on where I messed up. Overall, this program is an extremely helpful program. It helped me get used to large iterations that I never dealt with before and also helped me on keeping track of each variables and structure I used. I spent too much time getting confused on trying to differentiate the generation from the population.

My program did not run the way I wanted it to. However, please provide mt feed backs of what/where I went wrong and what exactly was off from my program that made it no sufficient. Very helpful program on getting used to generic algorithm and how to implement it.