In this experiment, we will be looking at the effects of different GA parameters on how well a GA can converge. In order to do this, I ran the same GA with different values for the parameters and recorded the best fitness score on each run. The parameters that I changed in this experiment are population size, mutation rate, crossover rate and elitism.

As of writing now, I haven’t managed to achieve optimal results, my algorithm produces results but it tends to only slightly converge or just remain at the same value, I’m suspecting it might be due to the crossover method not working as intended possibly causing no new genes to be introduced to the population

The first experiment was run using a population size of 100 and a mutation rate of 0.4%. The second experiment was run with a population size of 500 and a mutation rate of 0.01%. The third experiment was run with a population size of 1000 and a mutation rate of 0.5%.

Over 10 generations , I found that for population size of 100 and mutation rate of 0.4, the accuracy was 93%. For population size of 500 and mutation rate of 0.01, the accuracy was 87%. For population size of 1000 and mutation rate of 0.5, the accuracy was 83%. In my case, a smaller population size might with a higher be more beneficial as there is a higher chance of unique individuals being carried over to the next generation.

The first GA parameter that I will describe is the population size. The population size is how many individuals are in the population at one time. The larger this number, the more likely it is for a good solution to be found. The population size can be set between 1 and 100,000. I tried different values of population size with different runs and found that a good value for me was around 100 individuals.

The second GA parameter that I will talk about is the crossover probability which controls how often two parents are selected to produce a child individual in each generation. This value can range from 0% (no crossovers) to 100% (all crossovers). For my experiment, I tried different values with different runs and found that a good value was around 30%.

I found that if you want your GA to converge faster you should decrease your population size. Increasing the mutation rate also helps to make a GA more convergent but it is not as effective as decreasing population size. Crossover rates have no effect on convergence time while elitism has an insignificant effect on convergence time as well.

I introduced 2 new method called compare and predict, compare is a relatively simple function that takes a Boolean rule[] and a Boolean instance[] and compares the contents inside them with one another using a for loop .If the contents are identical, it returns true else it returns false. Predict uses the same input parameters as compare, it uses the compare method to check if the rules and instances are the same, if so it returns the final Boolean in instance being equal to the final Boolean in the rule. If the compare value was false, you negate the final value of rule and equal that to final value of the instance.

If I had to redo the assignment I would use a stricter fitness function, it is possible that my fitness function potentially converges too quickly because of not enough rules were generated. I would also amend by adding more input parameters, it is possible that I may have too few parameters to use effectively. Ideally ,If given enough time I would use around 10 input parameters.