Introduction

The purpose of this assignment is to explore the randomized optimization algorithms.

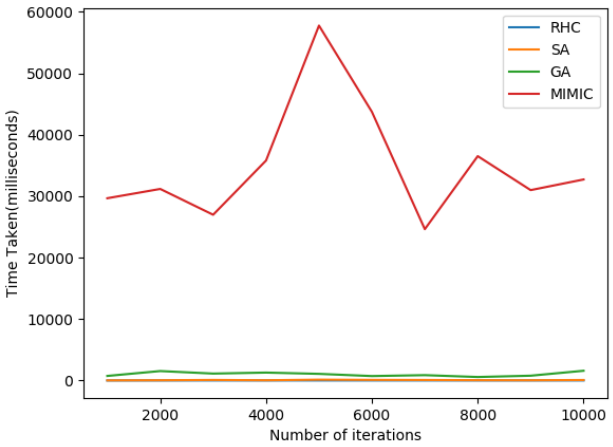
In the first part of this assignment I will be exhibiting the strengths of Simulated Annealing, Genetic Algorithm and MIMIC over three optimization problems.
I will first exhibit the strengths of Simulated Annealing algorithm over the continuous peaks problem Simulated Annealing performs the best in cases where there are large basins of attraction so that it does not get trapped in locally optimal solutions.
Next , I will exhibit the strength of MIMIC over the four peaks problem. It can be seen that MIMIC and Genetic algorithm both have comparable results for this problem but MIMIC outperforms Genetic Algorithm in terms of consistency and it's performance over longer strings. The third problem that I will be using to exhibit the strengths of Genetic Algorithm is the knapsack problem. Comparison is done by comparing the fitness score with the size of the input string. Bit strings are used for the Continuous peaks and four peaks problem of sixes varying from 10 to 100. Next, the number of iterations are varied from 1000 to 10,000.

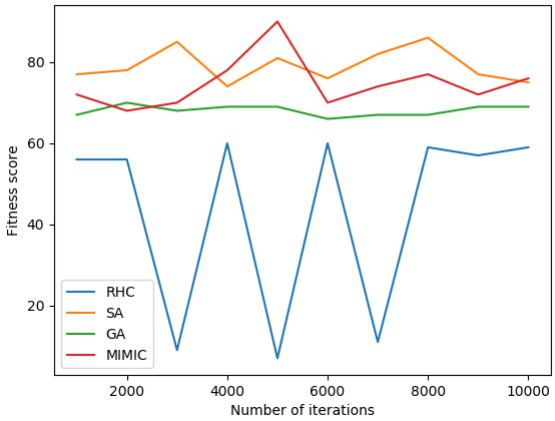
The string size used for different iteration runs was 50.

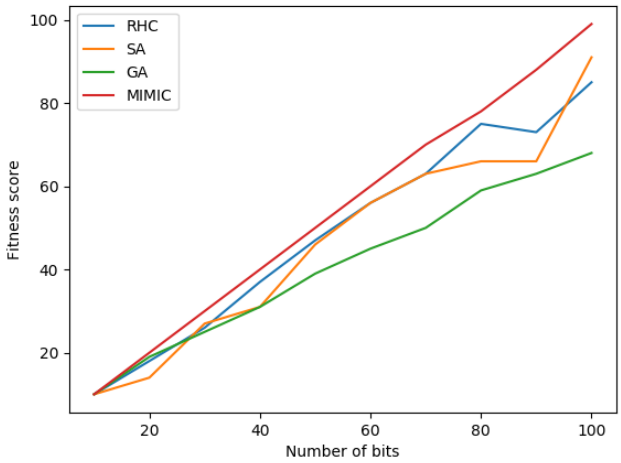
In the second part of this assignment I will be using the voice dataset from my assignment 1 for calculating the weights for the neural network. I used WEKA for my assignment 1 to calculate the weights using back propagation. It will be seen that back propagation produces weights which produce the maximum testing and training accuracy.
The testing and training dataset used is in the google drive link.

I used the python library mlrose to perform this entire analysis all the required scripts are in the google drive link shared in the README file.

Continuous Peaks







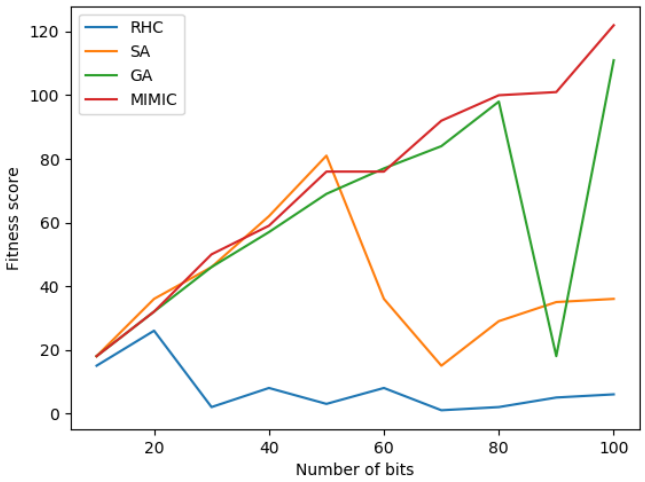
In Continuous Peaks problem the number of continuous ones and number of continuous zeros are counted and if the number of continuous ones is greater than a threshold T and the number of continuous zeros is also greater than the threshold T then a value equal to the size of the input array is added to the Max of the number of continuous zeros and number of continuous ones(Max((0,x) and (1,x)) to calculate the fitness score.
mlrose uses a default Threshold T = 0.1 times the size of the input array.

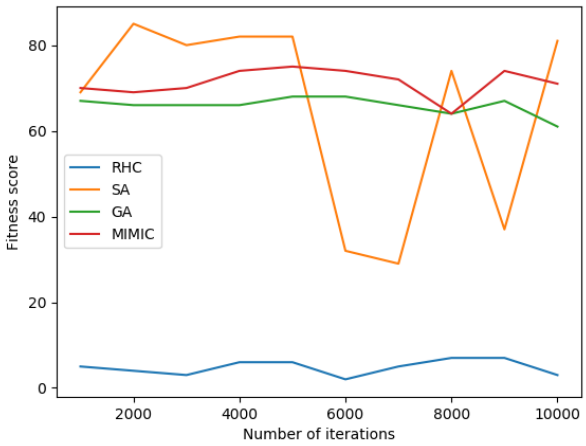
As we can see from the results that for the continuous peaks optimization problem both Simulated Annealing and MIMIC have comparable results for fitness score v/s number of iterations but we can see that the time taken by MIMIC to compute the fitness score comparable to Simulated Annealing is 50 times greater than Simulated Annealing to calculate the fitness score.
Simulated annealing is an algorithm which as Charles mentioned in the class analogous to Metropolis Hastings i.e. we keep on decreasing and increasing the temperature of the metal so that all the molecules can find an optimum state of arrangement where they could find lowest energy. So, in contrast to Randomized Hill Climbing which just ‘exploits’ and not ‘explores’ which in other words mean that it can get stuck at a local maxima . However, SA in contrast as when at a local maxima can increase the temperature to infinity which can cause it explore

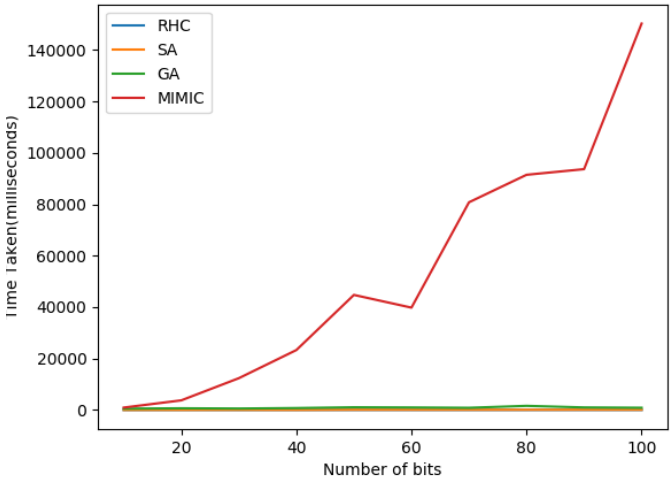
other basins of attraction as well and when temperature is less then it just exploits the current basin of attraction. So, as in continuous peak problem there is a high chance of being stuck at local maxima Simulated annealing proves to perform the best.

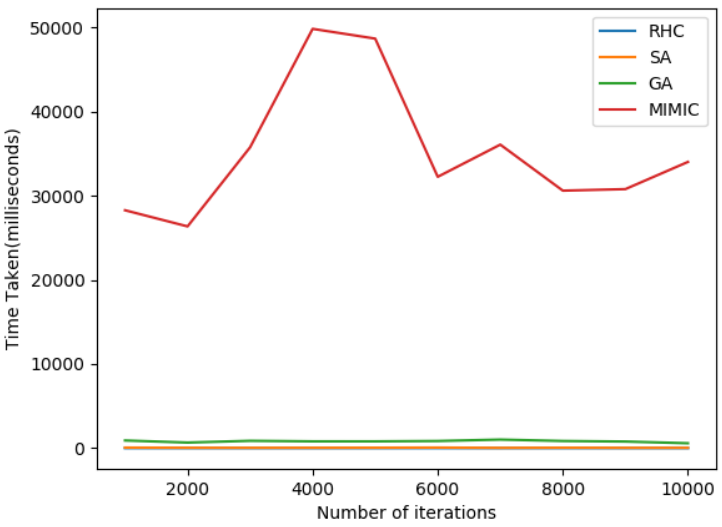
We can see from the graph of fitness scores v/s number of bits that as we increase the number of bits , Genetic algorithm performs poorly as compared to the Simulated annealing and MIMIC which is because in genetic algorithm mutation is performed and if a crossover is done on a string which is at its local optima then it would lead the crossed over string to go down to the lowest point in the curve (valley) which is not good as it would take introduce a string in the population with a low fitness value. So GA , RHC are not good choices for this problem as with GA a string of low fitness value might be introduced to the population and with RHC there is a high probability of being stuck at local optima since continuous peaks problem require lot many iterations to find the optimum fitness string and value. Considering, MIMIC's exceptionally high computational time we can say that SA is the optimum algorithm to solve continuous peaks problem.

Four Peaks





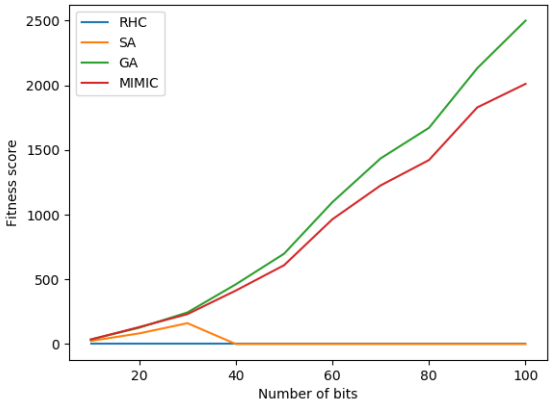


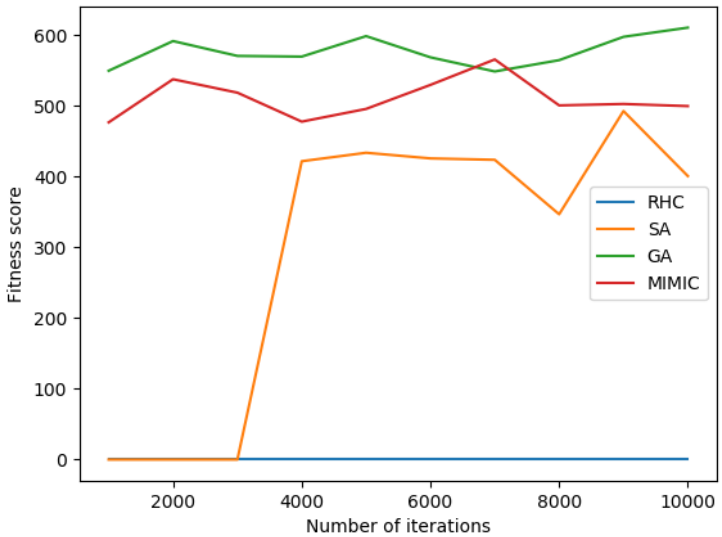


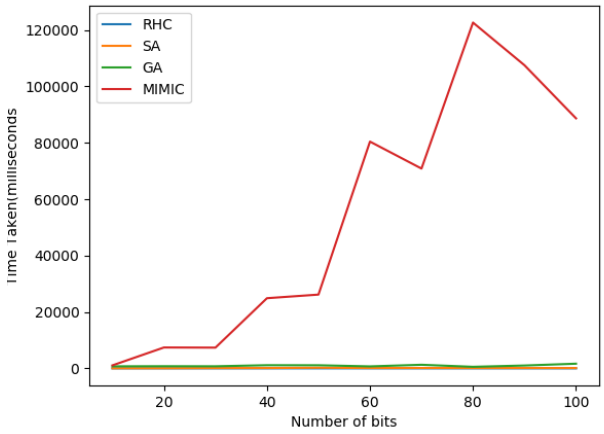
In contrast to continuous peaks problem where the continuous zeros and ones are allowed to form anywhere, in four peaks problem number of continuous 1's at the starting of the string and number of continuous zeros at the ending of the string are counted . If the number of continuous ones at the starting of the string and the number of continuous zeros at the end of the string are greater than a threshold T then a value equal to the size of the string is added to the max(tail(0,x) and head(1,x)) where x is the input string. There can be two global maxima of this problem when the number of leading 1's equal to T+1 are followed by all 0's or number of trailing 0's equal to T+1 preceded by all 1's. However, there can be two local maxima when there is a string of all zeros or a string of all 1's.
The basin of attraction of local maxima becomes large as we increase the value of threshold.

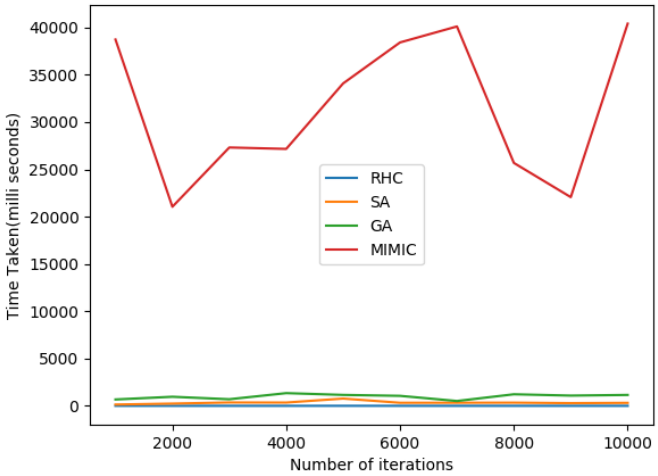
I am using this problem to depict the strength of the MIMIC algorithm . As we can see from the results and as explained above as there is a possibility of being stuck at two local maxima , RHC performs very badly as it does not explore. However, SA was performing well unto a bit some of 50 , it's performance considerable dropped as increased the bit size to100. S, it is not appropriate to use SA for this problem for longer bits.This is because SA is more likely to get trapped in the local optimal solutions. However, MIMIC performed consistently well even with longer bits. Performance of GA was unreliable since it dropped considerably for a bit size of 90. This can be because as GA performs crossover, performing a crossover at a point of local maxima will drop a string to a valley and introduce a low value string to the population. Also, GA was never able to achieve a fitness score equivalent to MIMIC even after increasing the number of iterations to 10000. Also, even though SA has a higher fitness score than MIMIC at lower iteration values but SA is not a reliable choice for this problem as its performance drops considerable for longer bits. As MIMIC can learn from it's previous searches so that can be the reason why it performs the best as it eliminates the possibility of searching at local maxima regions and searches only in regions which have global maxima in future iterations. Performance of MIMIC can be further improved by changing the population size.

Knapsack









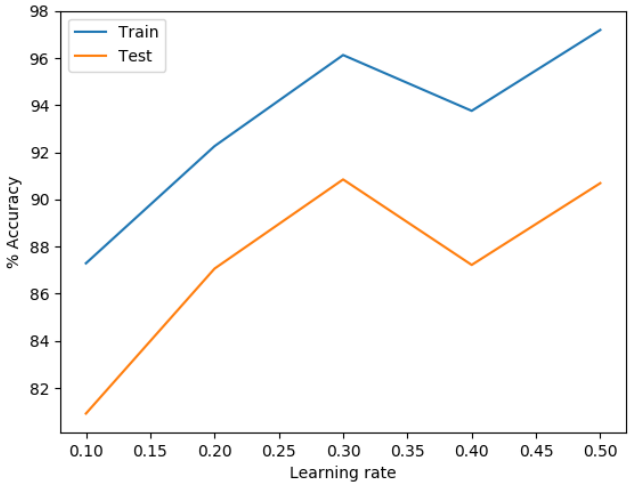
The knapsack problem is based on the analogy of having a knapsack which can only holds weights upto a maximal weight. Eg if we have a knapsack which can only hold weight unto 10 kg and we have 3 items of weight 2kg, 8kg and 4 kg. Then what should be the values of these weights so that we can fit them to the knapsack. One of the solutions would be to choose only those items which weigh 8kg and 2kg and other solution could be to chose half of the item which weighs 8 kg and chose 2kg and 4kg item along with. So, the knapsack problem can help to find the values of each of the items whose prior weight is known.

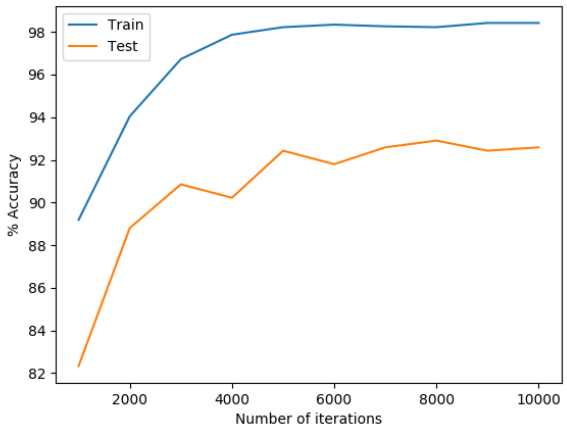
As we can see from the results that both GA and MIMIC have comparable performance but GA outperforms MIMIC for longer strings . Also, GA gets the highest fitness value at fewest number of iterations. GA is also considerable fast from MIMIC as we can see from the time graphs .Since the solution space of the knapsack problem can have many suboptimal solutions so RHC and SA performs considerably poor since they have the tendency to get trapped at local optimal values. However, as we increase the number of iterations then SA's performance improves because of its tendency to explore for other optimal solutions and not just exploit the current local optimal value. As genetic algorithms is considerably fast I.e. unto 120 times faster than mimic for longer strings. As MIMIC's fitness value at 7000 iterations is comparable to GA's fitness value at 2000 iterations, we can say that GA gives a highest fitness value at 5000 iterations lesser than MIMIC so clearly GA is a better choice for the knapsack problem

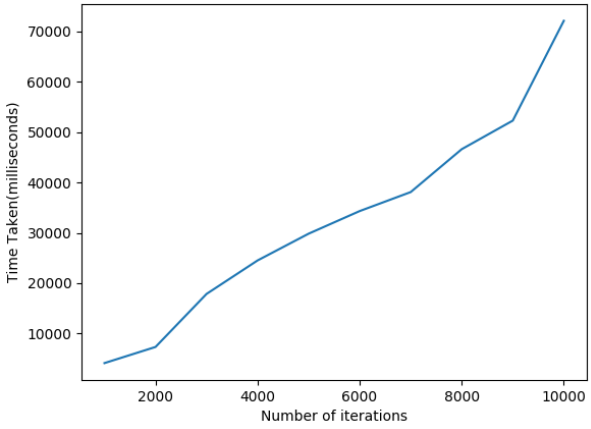
Neural Networks

I am using the voice data from my assignment 1 to find the weights for the neural network using the below three optimization algorithms. The number of nodes were similar to what I used last time with weka which was 11 (attributes+classes)/2 and since weka uses the sigmoid function so the function used was also sigmoid

RHC







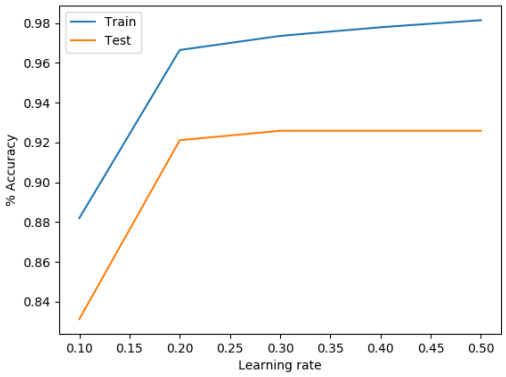
I first used the training and testing data to find the optimal learning rate for the algorithm which was found to be 0.3 . From this, we can say that at a learning rate below this the algorithm was taking too slow steps to reach the optimal value whereas at a learning rate higher than this the algorithm takes too big steps such that it misses the optimal value.
Next, I used this learning rate and varied the number of iterations to find the optimal number of iterations at which the testing and training accuracy is maximum . As we can see from the results that after an iteration of 8000 , there is no significant improvement in the training and testing accuracy and the total time taken to achieve this highest accuracy was 40 seconds.

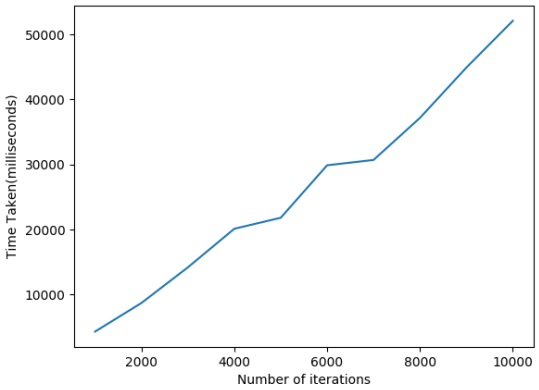
The maximum training accuracy obtained at 8000 iterations and a learning rate of 0.3 was 98% and maximum testing accuracy obtained at 8000 iterations and a learning rate of 0.3 was 93%.

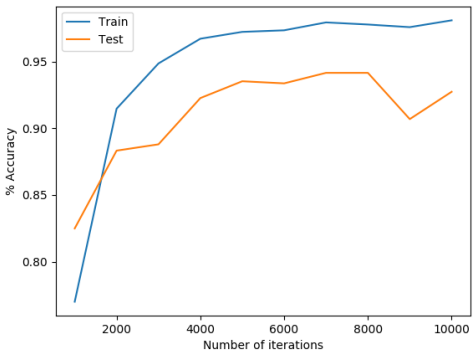
On the contrary using back propagation I achieved a maximum training accuracy of 99.6% and maximum testing Accuracy was 96.25% with a learning rate 0f 0.4 in a time of 4 seconds.

By this we can say that using back propagation to find the weights is a more optimal way than using Randomized Hill Climbing.

Simulated Annealing



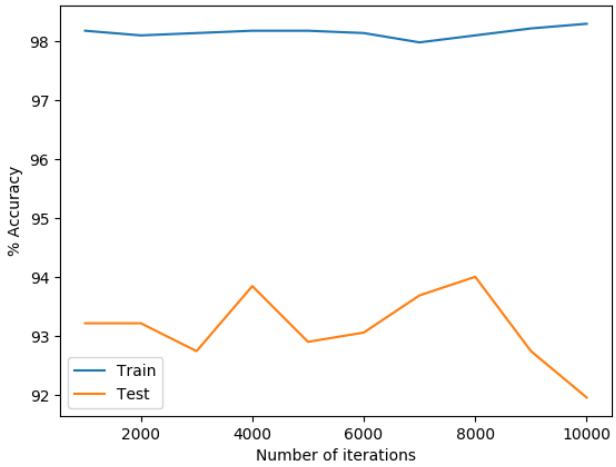


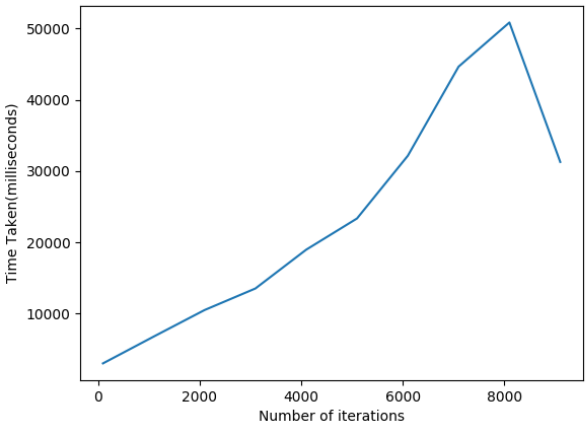


The optimal learning rate for both the testing and the training data was found to be 0.20. As we can see from the results that even for simulated annealing the maximum accuracy for both the training and testing data can be found at 8000 iterations and the maximum training accuracy obtained was 98% and maximum testing accuracy was found to 93%. The time taken to perform 8000 iterations which was the number of iterations at which we obtained the maximum testing and training accuracy was again 40 seconds.

As compared to the ANN algorithm which used back propagation to find weights which was done in the previous assignment which took 4 seconds to achieve a maximum training accuracy of 99.6% and the maximum testing accuracy obtained was 96.25% we can say that SA doesn't perform with same efficiency as compared to the back propagation method.

Genetic Algorithm





As we can see from the results that the maximum accuracy that can be achieved was of 98.5% on training data and on testing data was of 92% at 4000 iterations after which the accuracy does not change . The time for 4000 iterations was 20 seconds which is still higher to the 4 seconds time taken using back propagation. Also, the accuracy for both testing (96.25) and training (98) is still higher with back propagation. So, we can say that back propagation is the most optimal method to calculate the weights for a neural network.