Name: Manva Patel

Email address: m329pate@uwaterloo.ca

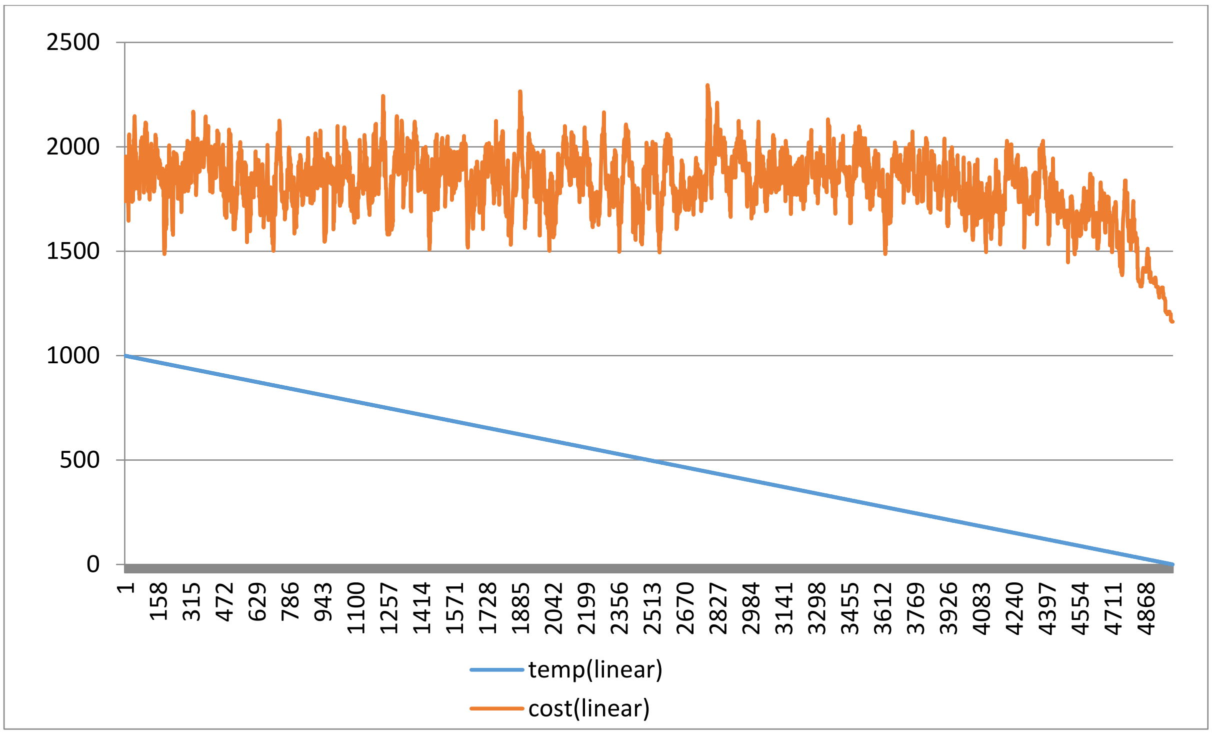
Student number: 20815074

Assignment 2 Question 1

a. The moveset is generated by randomly selecting any two cities from the city_list and swapping them. I have used the random.sample() function to select any two random indexes from the list of cities. These two indexes are then swapped to generate the neighbours of the current city_list.

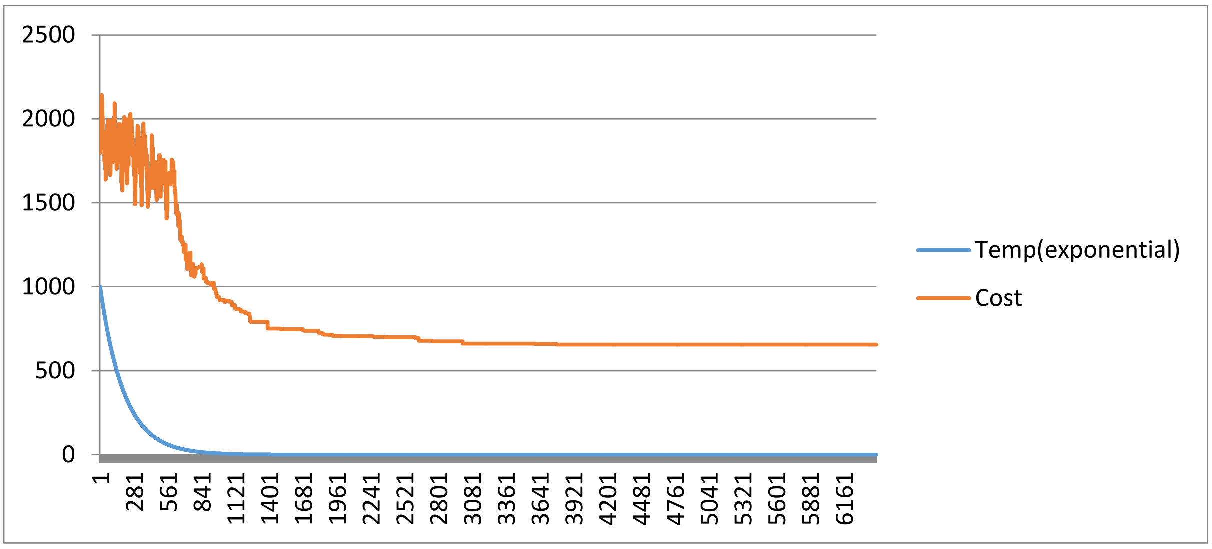
b. I checked the outputs for three annealing schedules. I checked them with the 36 city problem and plotted their cost and temperature variation during one run of the algorithm as follows:

- **Linear:** I reduced the temperature linearly by 0.2 in every iteration. The algorithm iterated 5000 times. The final cost obtained was 1162.726.



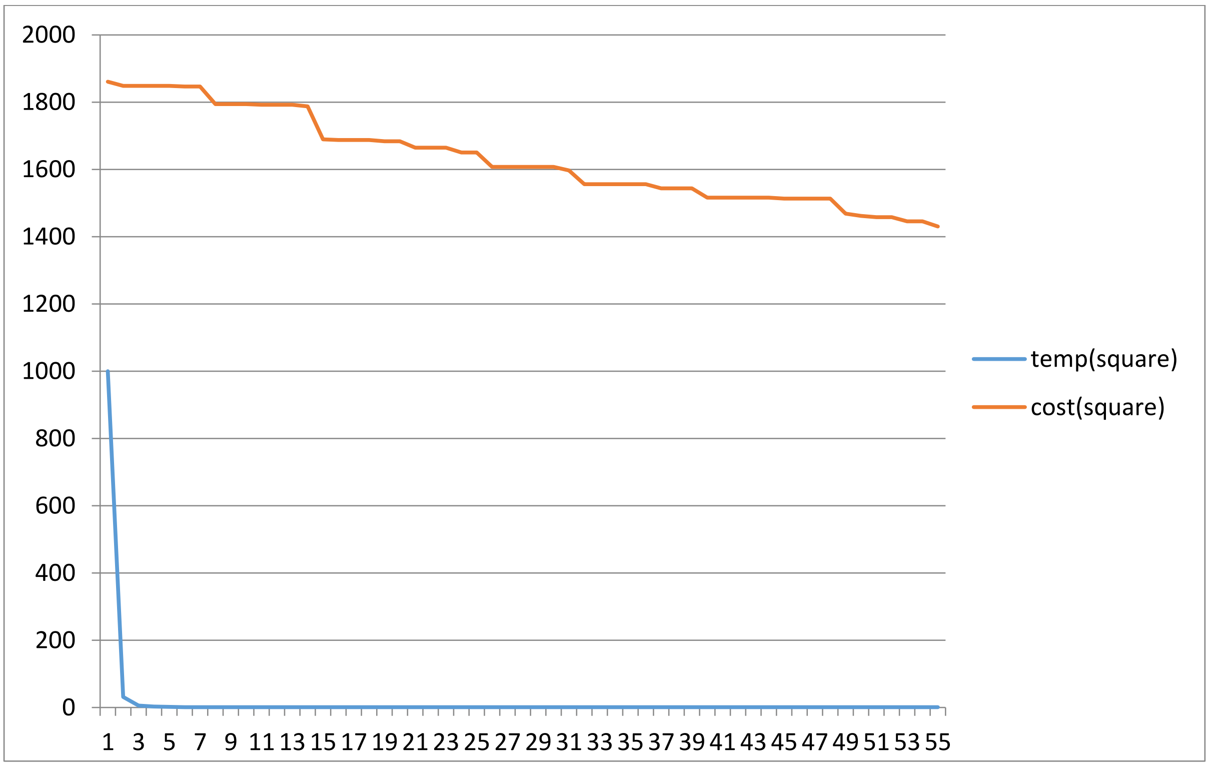
Graph 1: For linear reduction in temperature. X-axis : The number of iterations. Y-axis: Cost and temperature in each iteration

- **Exponential:** Here, I reduced the temperature exponentially by 0.995. The algorithm iterated 6432 times and the best cost obtained was 655.865.



Graph 2: For exponential reduction in temperature. X-axis : The number of iterations. Y-axis: Cost and temperature in each iteration

- **Square:** Here, I reduced the temperature by square root. This reduced the number of iterations drastically. The program only took 51 iterations. And the best cost obtained was 1430.251.

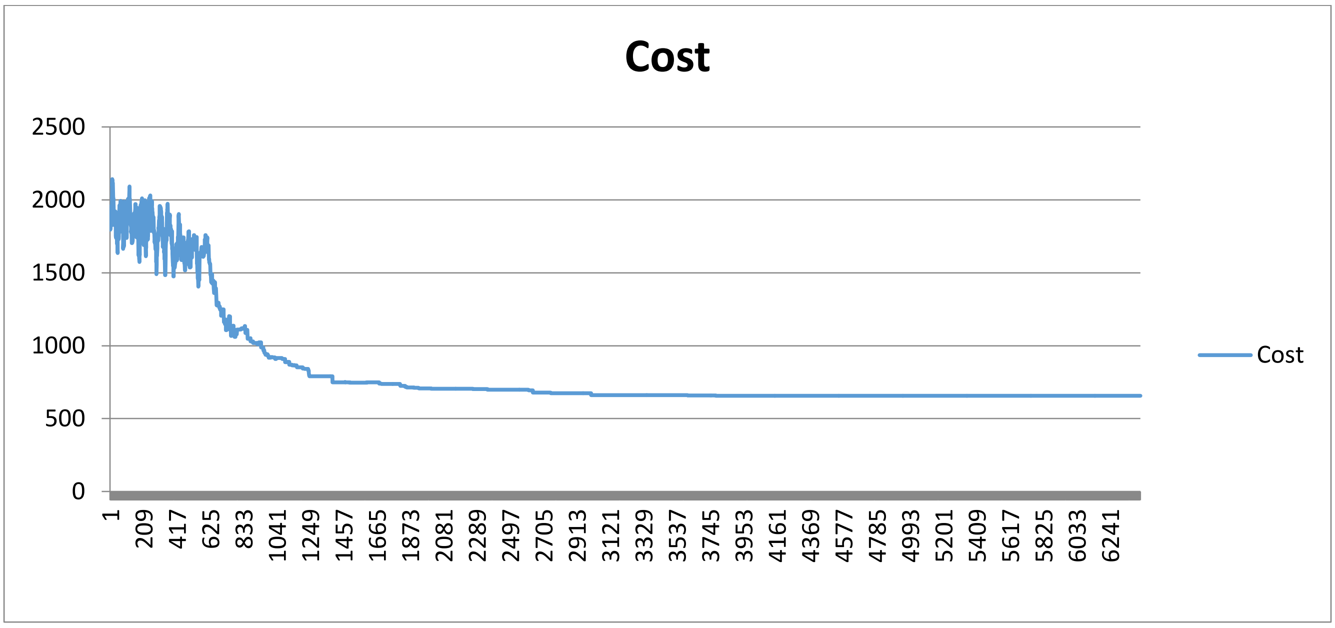


Graph 3: Temperature reduced by square-root. X-axis : The number of iterations. Y-axis: Cost and temperature in each iteration

By looking at the graph, it can be conferred that the best results are obtained if we reduce the temperature exponentially. The tradeoff between number of iterations and the result obtained is pretty good. Further, the cost changes in the beginning, but then it becomes steady at lower temperature. This is according to the property of simulated annealing. Reducing the temperature by square root is a bad option, as it causes the program to terminate quickly. The temperature reduces drastically in the beginning which does not let the algorithm explore its domain. So, the effect of simulated annealing is not seen. Also, reducing the temperature linearly does not give a good answer. The best cost obtained is quite higher than the one obtained by reducing temperature exponentially.

So I believe that the best annealing schedule is to reduce the temperature exponentially.

c. As shown in the graph, the cost of the best solution which is obtained when running the 36 city problem is 655.8648389. The cost of the best solution varies with each run.



Graph 4: x-axis shows the number of iterations and y-axis shows the cost of the solution during each iteration

d. I believe that the version of simulated annealing which I have implemented is complete. An algorithm is said to be complete if it gives a solution of a problem. It is not necessary that the solution has to be the best one. From this definition of completeness, we can say that the above algorithm is complete as it gives as output the order of cities that should be travelled. Though it does not give the best solution, it does give an output. So the algorithm is complete

e. The simulated annealing algorithm is not always optimal. As seen in the output, it might happen that when the temperature decreases, the algorithm gets stuck in a local minima and it cannot overcome it at low temperature. Further, the algorithm selects the moveset by randomly swapping cities. So, it might be possible that the algorithm will never generate the best path and so it will not get optimal solution. So, even if there is a better solution, the algorithm will not detect it.

Assignment 2 Question 2:

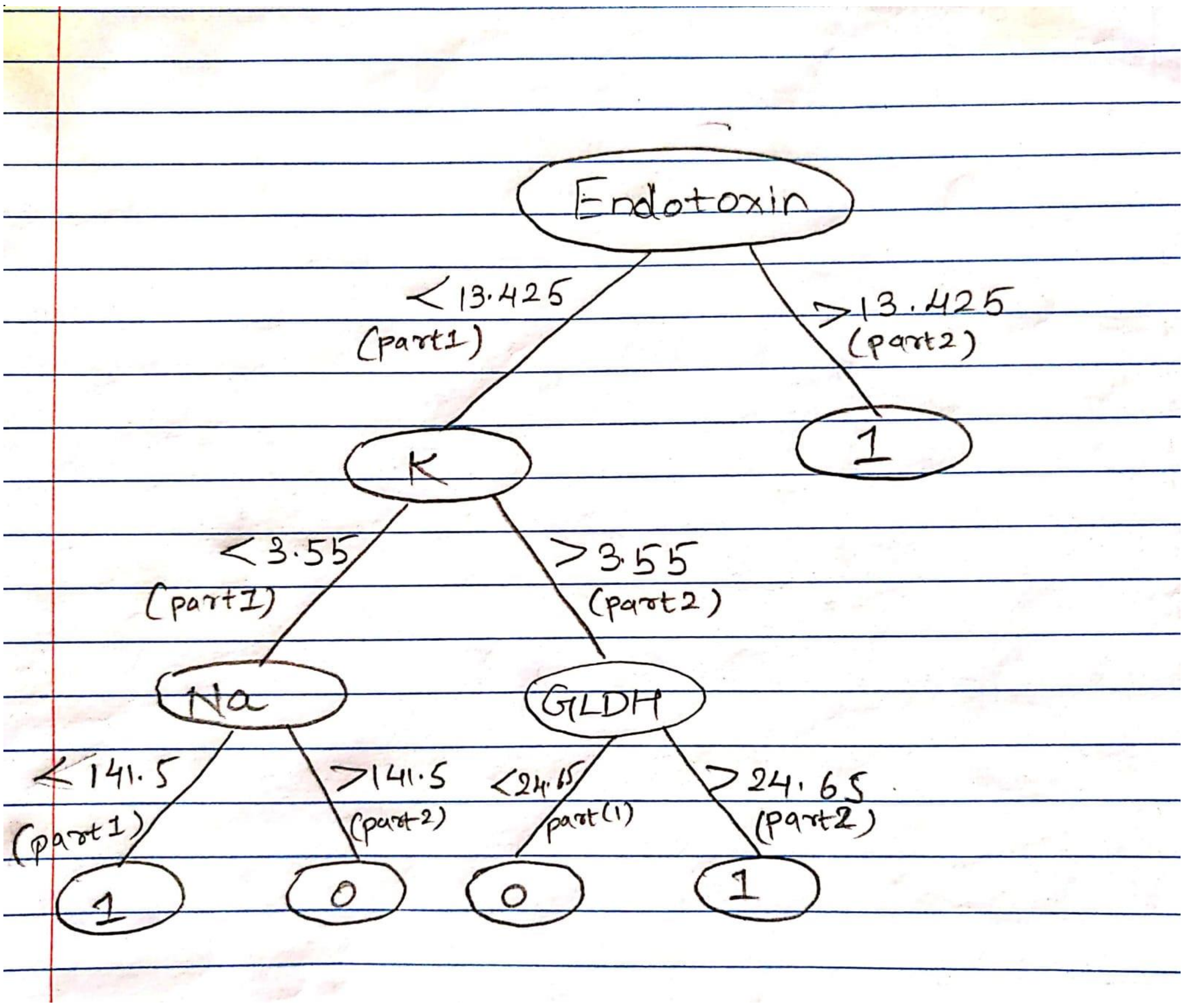
The better evaluation function that I have used looks for the number of series of 1's, 2's and 3's the current player has. It also looks for the series of the opposition player. Weights are assigned to the series of different numbers. The series of 3's will have a higher weight than series of 2's, which will in turn have higher weight than the series of 1's. So, a weighted sum of the series is taken for the current player and the opposite player. The weighted sum of the opposite player is subtracted from the weighted sum of the current player.

From this, we can say that if the current player has higher number of series, then the score will be higher. Also, if the current player has more number of 3's, then it has more chances of winning ans so it will return a better score. But if the current player does not have any series of 2's and 3's or has a less number of series, then it's chances of winning are low and so the score will be lower. At the same time if the opposition player has higher number of series of 3's and 2's, then it will reduce the score of the current player, as that would mean that the chances of the opposition player winning are high.

The basic evaluate function only checked for the longest chain, whereas this function checks for the number of possibilities the current player has for winning. If it has more series of 3's, then it will have many options for winning and so it increases the chances of winning. It also takes into account the opposition player and reduces the score if the opponent is close to winning.

Assignment 2 Question 3

2) The decision tree obtained is as follows:



3) The tree correctly classifies all the instances of the training data. So the prediction accuracy on training data is 100%.

4) The tree correctly classifies all the instances of test data as well. The accuracy on test data is 100%.