

CS2004

Algorithms and their Applications

Worksheet 15: The Travelling Salesman Problem²⁰¹⁸



Index

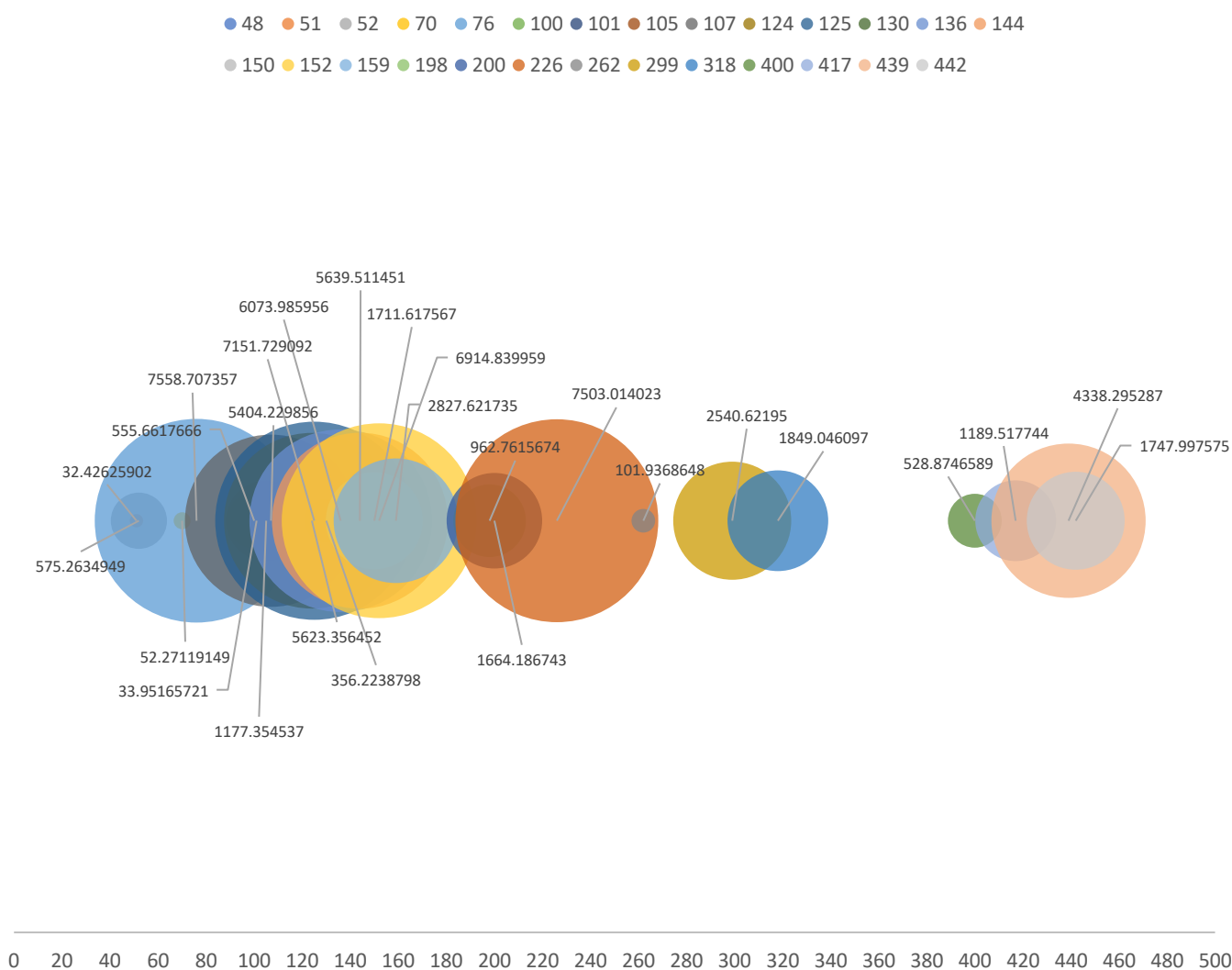
<i>Introduction.....</i>	<i>2</i>
The UI.....	3
Setting Iterations	3
<i>Random Mutation Hill Climber.....</i>	<i>6</i>
<i>Random Restart Hill Climber</i>	<i>7</i>
<i>Stochastic Hill Climber</i>	<i>8</i>
<i>Simulated Annealing.....</i>	<i>9</i>
<i>Conclusion</i>	<i>10</i>

Introduction

This report gathers the analysis of twenty-seven datasets using four different algorithms. These are:

- Random Mutation Hill Climber
- Random Restart Hill Climber
- Stochastic Hill Climber
- Simulated Annealing

The data sets vary in size and average distance as displayed in the bubble graph below. While the sizes of the datasets go from 48 to 442, the average distance differ from one data set to another.



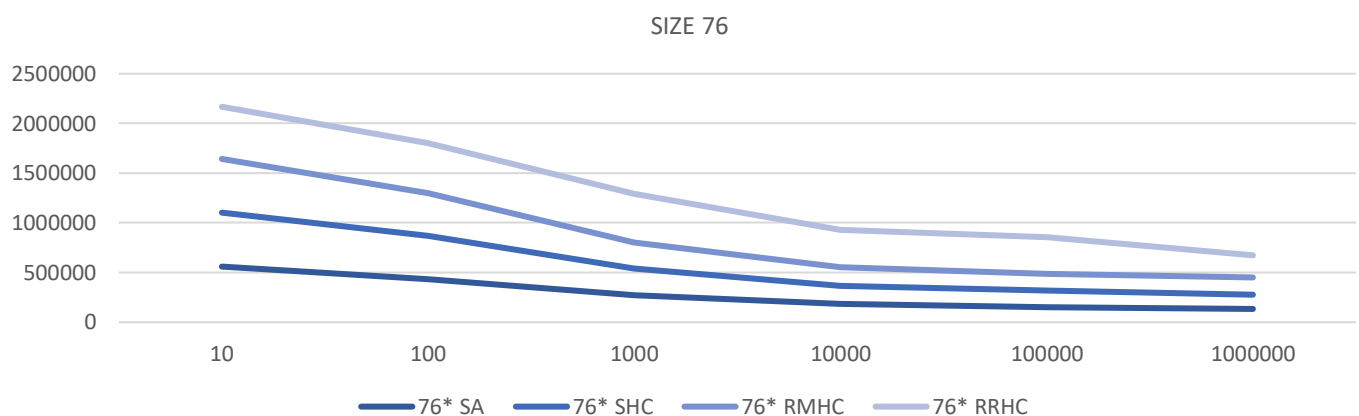
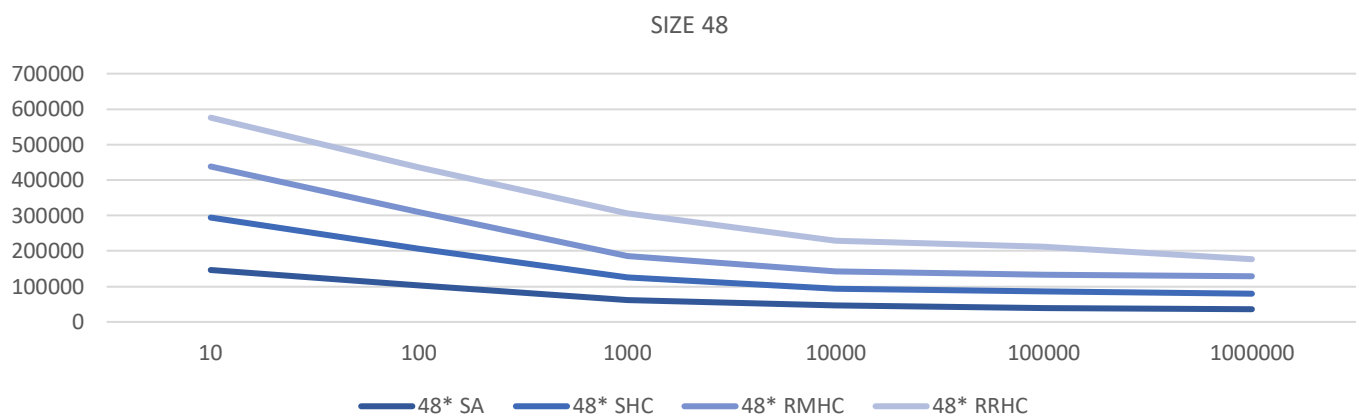
User Interface

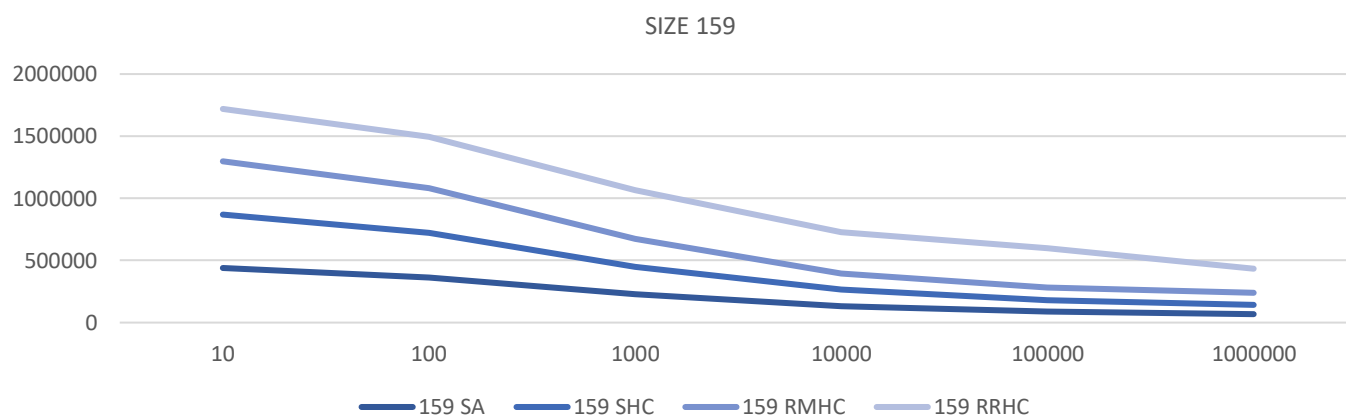
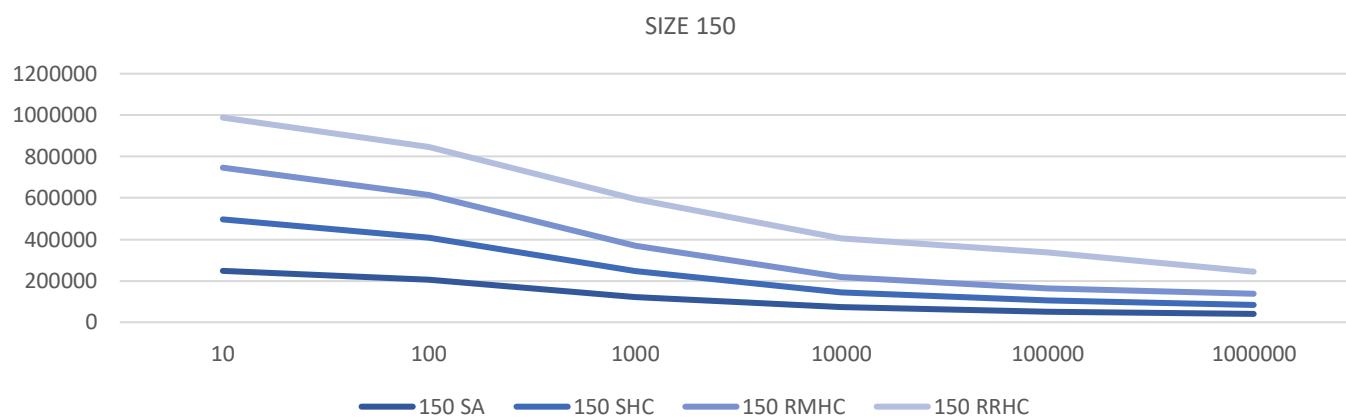
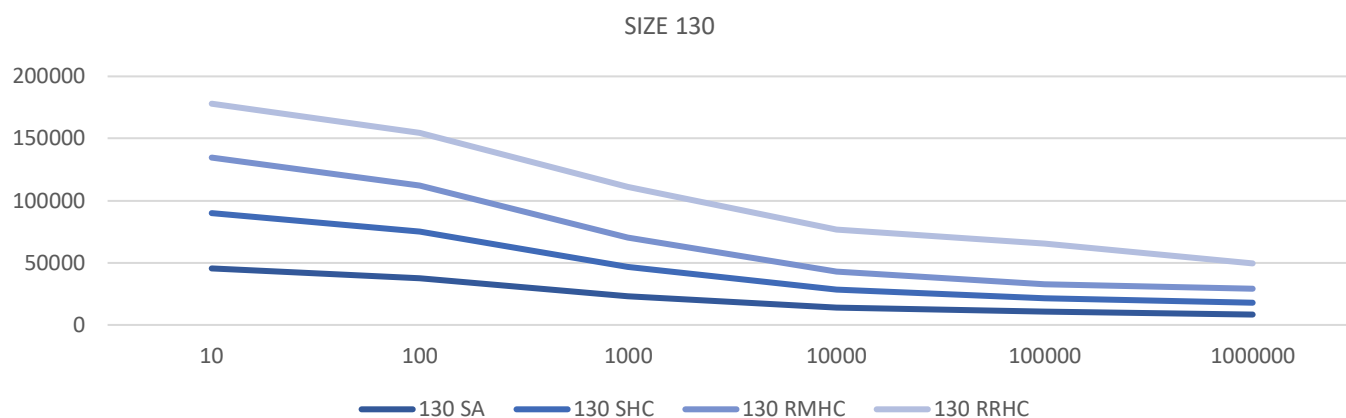
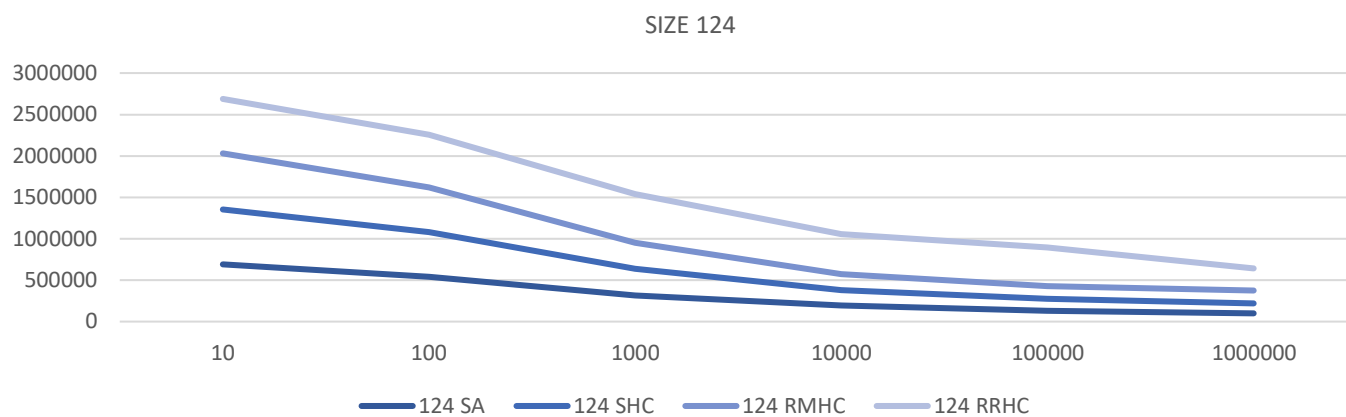
Once the user select a data size from the drop down menu, the app calculates the best temperatures.

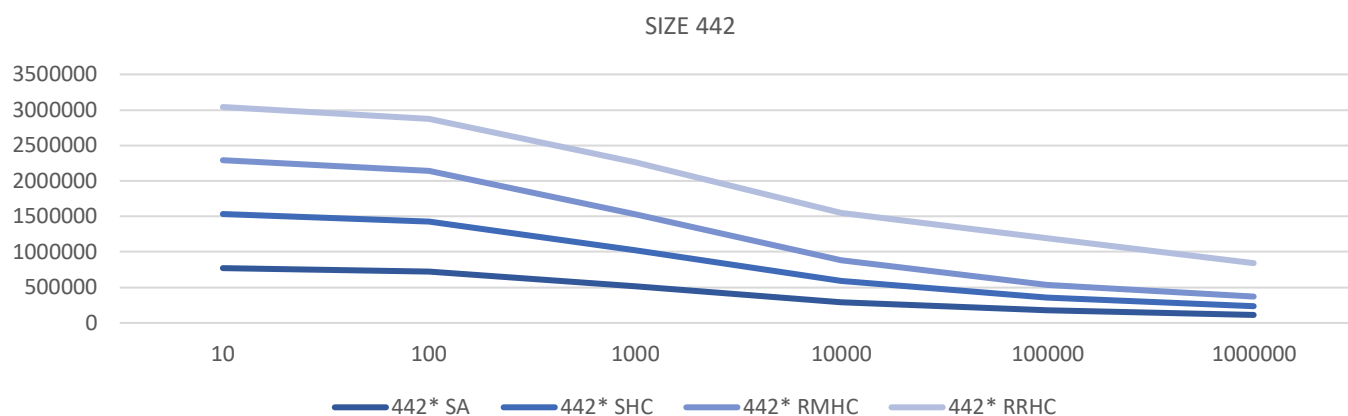
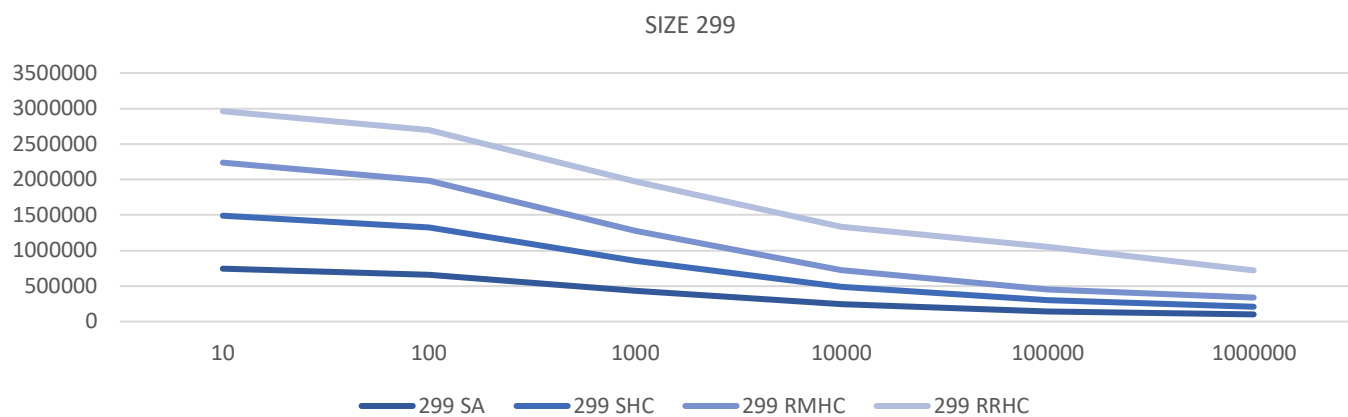


Setting Iterations

To select the number of iterations I will use throughout the whole project, I ran several datasets with different sizes from 10 to 1,000,000 iterations.





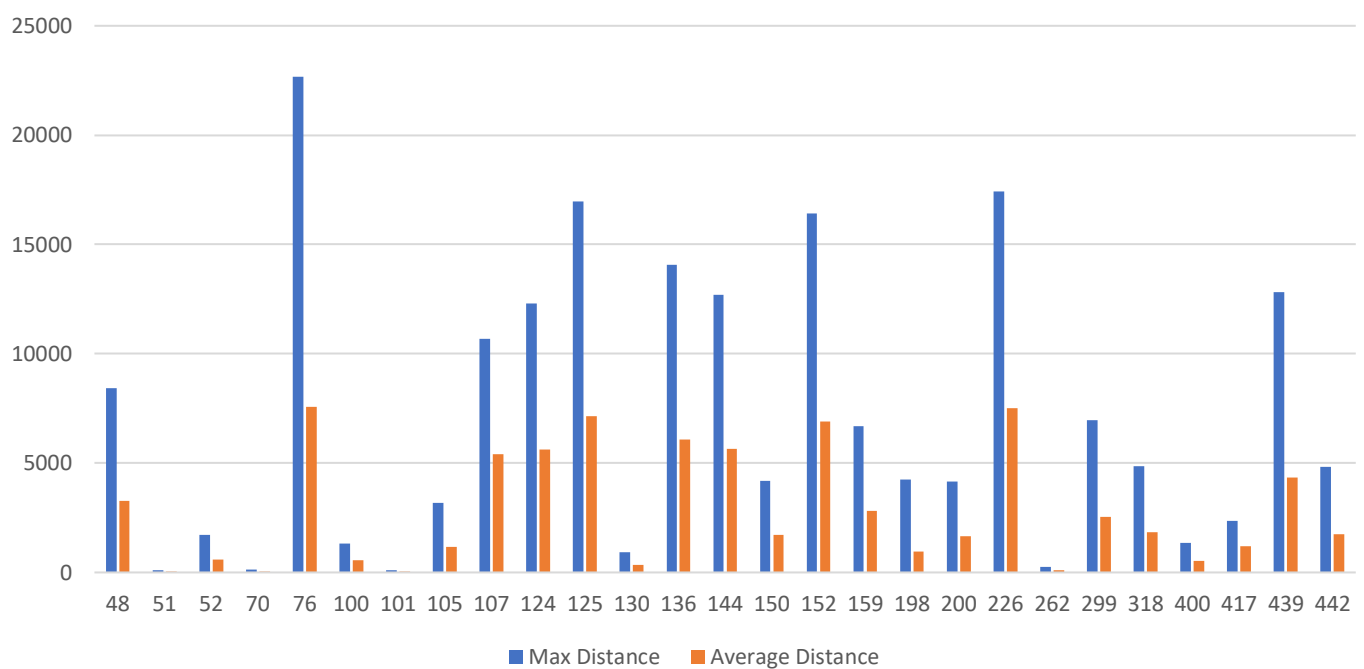
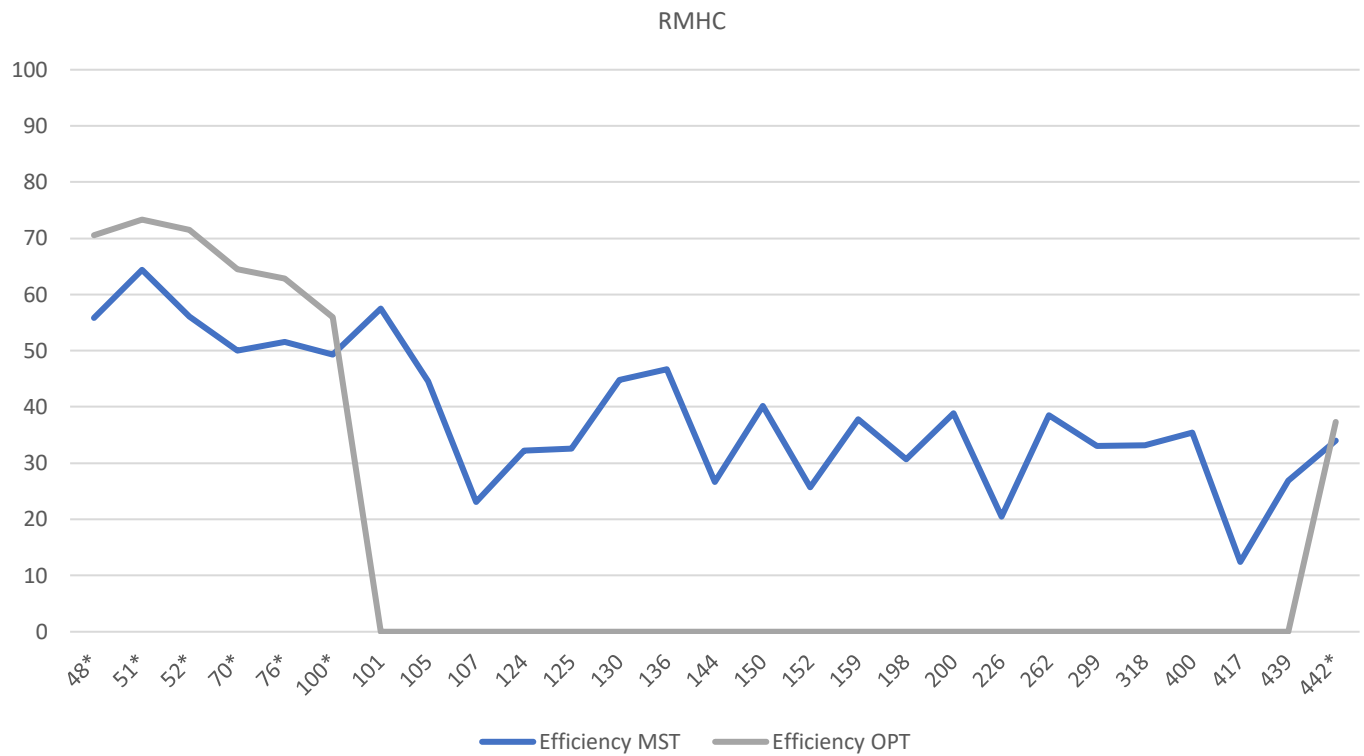


Random Mutation Hill Climber

This method consists in creating a random generated path and then apply a small change several times.

1. Generate random path
2. Apply small change
3. Compare fitness
4. Store path with lowest fitness
5. Repeat “1” times

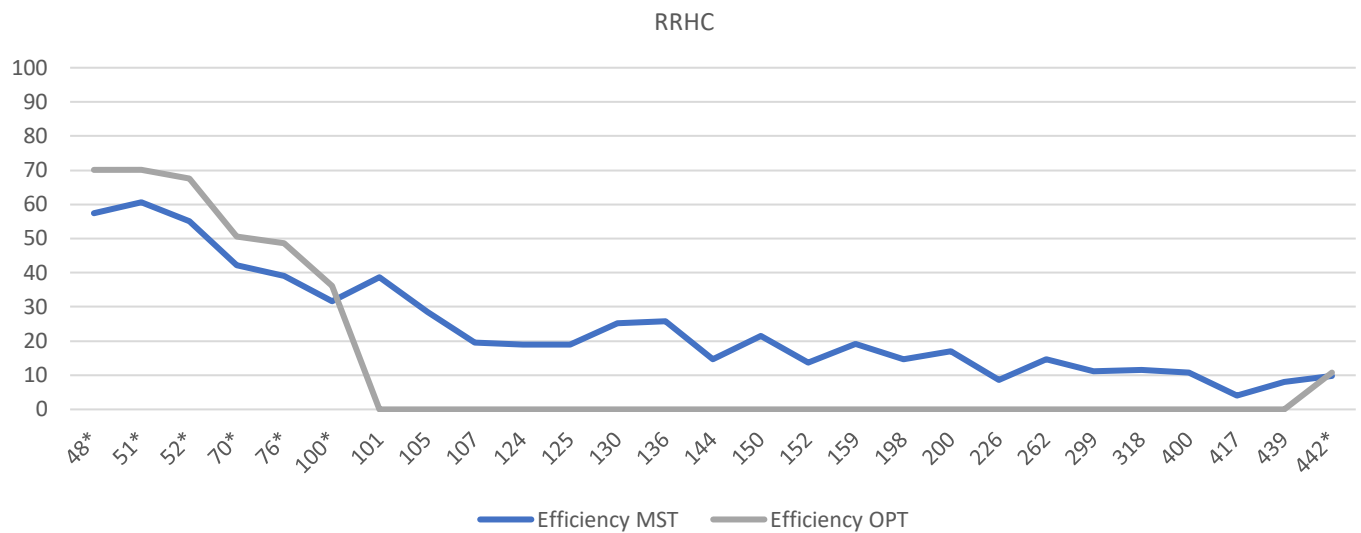
Comparing the efficiency with the maximum distance, we can see how the maximum distance affect the efficiency.



Random Restart Hill Climber

The random restart hill climber is basically an extension of the random mutation. This method run the Random Mutation X times and the Random Mutation produces Y changes to the path. The multiplication of X and Y should be equal to the total number of iterations.

1. Generate random path
2. Run RMHC Y times
3. Choose the most efficient path
4. Repeat X times.



Stochastic Hill Climber

This method works similar the Random Mutation Hill Climber. After applying the small change, it calculates the probability using the new and old fitness in this formula

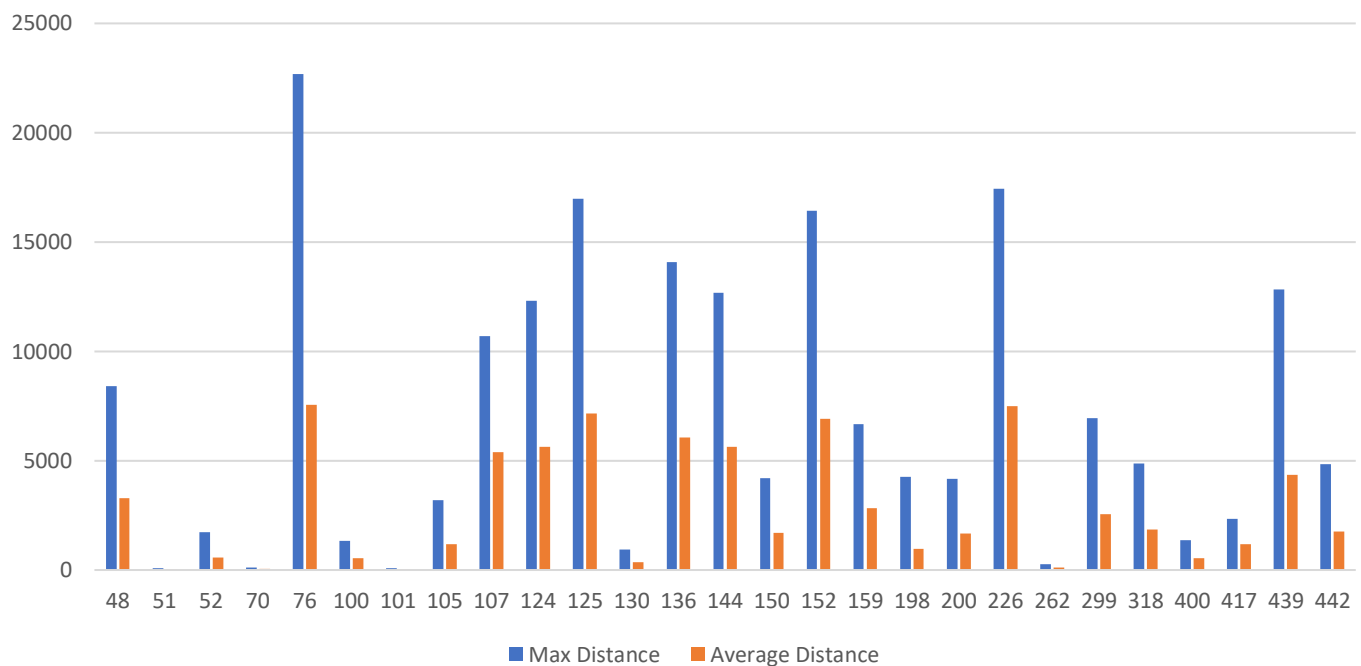
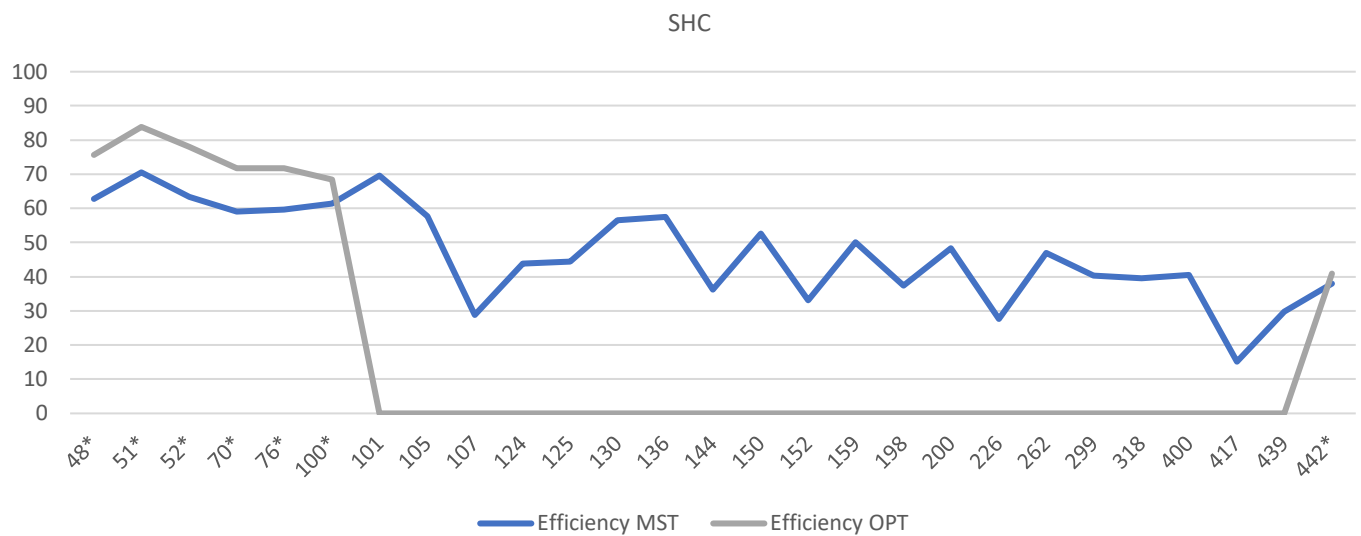
$$\text{Pr}(\text{accept}) = \frac{1}{1 + e^{(f' - f)/T}}$$

Then compares the probability with a number randomly generated between 0 and 1. If the probability is greater than the randomly generated number, the new path is admitted.

To calculate the temperature, I have used this formula:

Size * K = T;

I was given the optimal temperature for the data set 48 which is 45 Then I just calculated the constant which is 1.63×10^{-3} .

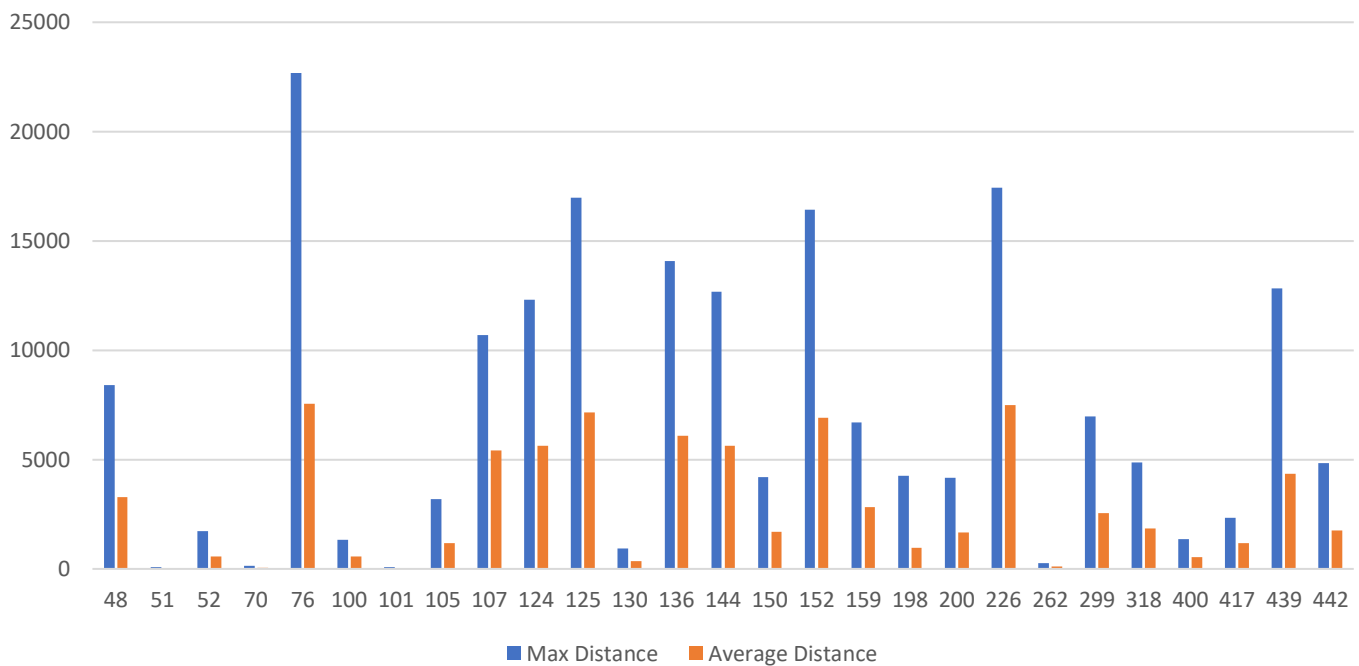
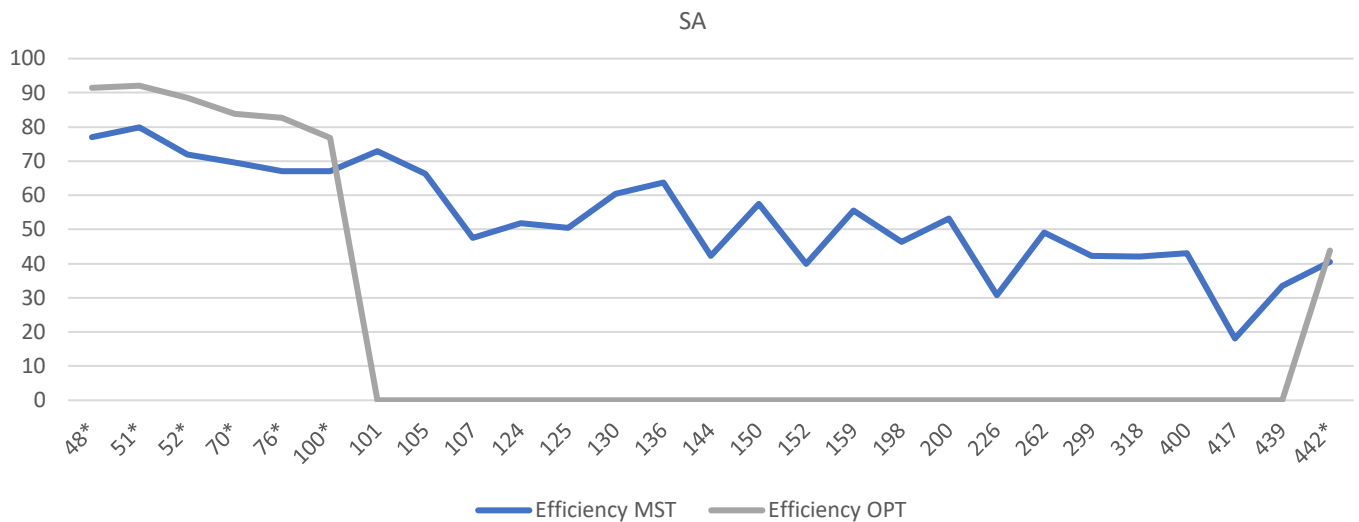


Simulated Annealing

This method allows a worse solution to be accepted so that local maximums can be circumnavigated.

It contains 3 main parameter that has to be calculated for this method to be more efficient.

- Temperature: it is the initial temperature. It has to be high as it will be reduced throughout the process. I have used this formula to calculate it: $MST - (Size * Max\ distance) / Any\ number\ to\ scale\ down$
- Minimum Temperature: it has to be a really small number which will help to bring the initial temperature down to 0
- Cooling Rate. Which is calculated using the min and the intitule temperature and the number of iterations.



Conclusion

Throughout the various test performed, the Simulated Annealing algorithm has featured the best efficiency hence it has been selected to calculate the following graph. This contains the efficiency using the Simulated Annealing Algorithm and the MST which is displayed on the Y axis against the size of the dataset and the average distance. This test was run with these parameters:

- Iterations = 1,000,000
- Temperature = depending on the dataset (The code calculates it automatically)
- Minimum Temperature = 0.00001.

As seen in this graph the efficiency improves slightly when the average distance is smaller.

