



# Report

GitHub Repository: <https://github.com/taraliu23/CSE6140-Final-Project>

Group Member: Tingyu Liu (tliu479@gatech.edu)

## Description of Algorithms

### 1. Brute Force

The Brute Force algorithm calculates all the city permutations and returns the optimal tour and cost.

Time complexity:  $O(n!)$

### 2. Approximate with MST

The Approximate MST algorithm is a 2-approximation algorithm. It constructs a Minimum Spanning Tree (MST) and performs a depth-first search.

Time complexity:

$O(n^2 \log n)$

Algorithm:

1. compute distance matrix
2. construct MST with Kruskal's algorithm
3. List visited cities in the preorder walk of the MST and add one at the end

### 3. Local search with simulated annealing

Simulated Annealing is a local search algorithm and can achieve near-optimal results.

Algorithm:

1. init with a random travel, temperature, and cooling rate.
2. while the temperature is higher than the limit:
  - swap two cities and generate travel, calculate the cost
  - accept new state
  - update the local optimal travel
  - decrease the temperature with the cooling rate
3. return the local optimum

## Performance Table

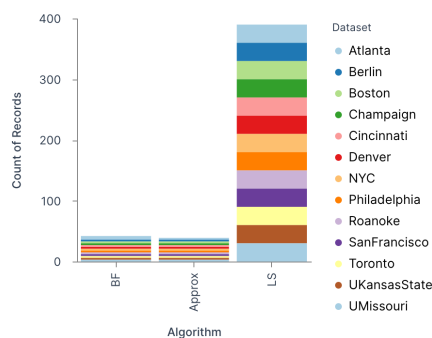
	avg time			avg quality			std quality			full tour			avg rel error		
Algorithm	Approx	BF	LS	Approx	BF	LS	Approx	BF	LS	Approx	BF	LS	Approx	BF	LS
Dataset															
Atlanta	0.0	200.0	0.03	305974.47	3375249.93	2369227.38	0.0	48775.82	170037.52	FALSE	TRUE	TRUE	0.0	0.0	0.0
Berlin	0.0	200.0	0.04	412.33	19248.66	11433.69	0.0	0.0	479.73	FALSE	TRUE	TRUE	0.0	0.0	0.0
Boston	0.0	200.0	0.04	13156.06	2220895.38	1162328.23	0.0	0.0	33545.13	FALSE	TRUE	TRUE	0.0	0.0	0.0
Champaign	0.0	200.0	0.05	21160.95	209944.26	82558.91	0.0	0.0	6698.8	FALSE	TRUE	TRUE	0.0	0.0	0.0
Cincinnati	0.0	8.92	0.02	143267.02	277952.59	280632.94	0.0	0.0	3566.86	FALSE	TRUE	TRUE	0.0	0.0	0.0
Denver	0.0	200.0	0.05	8986.27	546265.49	205102.82	0.0	758.57	8341.24	FALSE	TRUE	TRUE	0.0	0.0	0.0
NYC	0.0	200.0	0.04	67375.81	7166313.08	2769304.64	0.0	0.0	154538.18	FALSE	TRUE	TRUE	0.0	0.0	0.0
Philadelphia	0.0	200.0	0.02	477466.98	3624920.35	1796664.96	0.0	0.0	152281.21	FALSE	TRUE	TRUE	0.0	0.0	0.0
Roanoke	0.03	200.0	0.11	43218.51	6784252.11	3303619.1	0.0	2851.5	114315.09	FALSE	TRUE	TRUE	0.0	0.0	0.0
SanFrancisco	0.01	200.0	0.05	10839.73	5591803.8	2130136.44	0.0	0.0	75508.4	FALSE	TRUE	TRUE	0.0	0.0	0.0
Toronto	0.01	200.0	0.06	15422.86	9116968.19	3721400.1	0.0	0.0	318525.21	FALSE	TRUE	TRUE	0.0	0.0	0.0
UKansasState	0.0	6.16	0.02	52292.14	62962.31	62962.31	0.0	0.0	0.0	FALSE	TRUE	TRUE	0.0	0.0	0.0
UMissouri	0.01	200.0	0.06	7645.63	662933.16	294287.12	0.0	0.0	12453.2	FALSE	TRUE	TRUE	0.0	0.0	0.0

A total of 472 TSP instances were computed, and the table shows the aggregated average time, solution quality, and relative error for each algorithm and dataset.

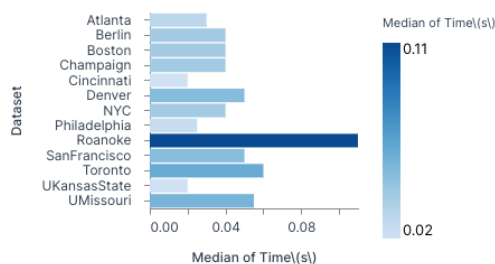
Additionally: The **full\_tour** column is averaged over all runs. If the value is greater than 0.5, the algorithm consistently produces valid tours for the dataset and is marked as **TRUE** in this table. Otherwise, it is marked as **FALSE**.

## Analysis and Visualization

Count of Records by Dataset



Median of Time(seconds) By Dataset



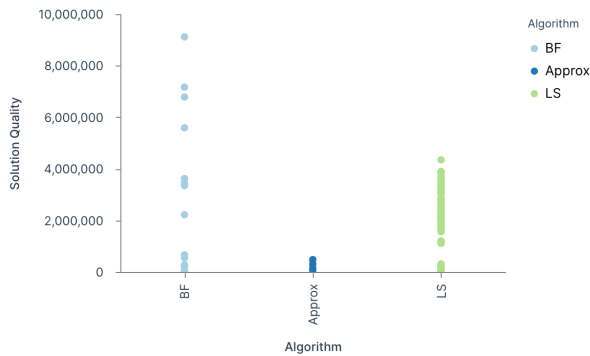
### Insights:

- MST Approximation and Local Search with simulated annealing shows efficiency across datasets.
- Brute Force algorithm dominates the runtime for larger datasets.

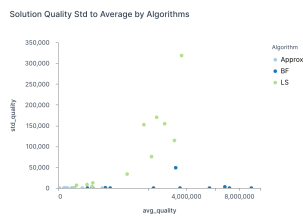
### Observation:

- Most datasets have median execution times under 0.05 seconds.
- Roanoke, the dataset with the largest number of points, have highest median times.

Solution Quality by Algorithms



Solution Quality By Algorithm



Solution Quality Variability

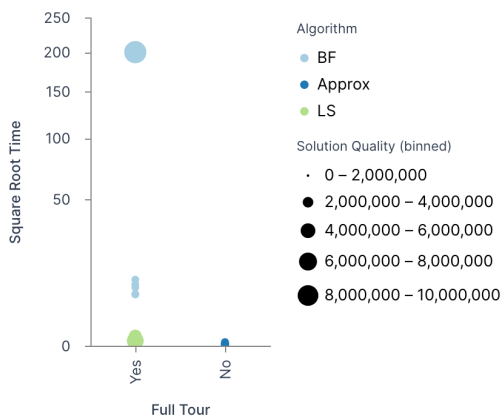
• **Observation:**

- Most datasets have median execution times under 0.05 seconds.
- Roanoke, the dataset with the largest number of points, have highest median times.

• **Insights:**

- MST Approximation and Local Search with simulated annealing shows efficiency across datasets.
- Brute Force algorithm dominates the runtime for larger datasets.

Full Tour Boolean to Time By Algorithms and Quality



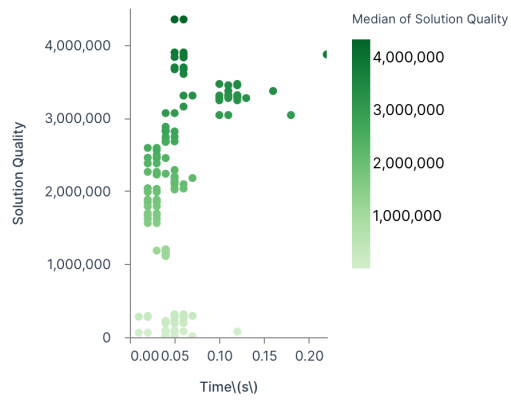
• **Observation:**

- Brute Force has the best quality across datasets, and MST Approximation has the lowest quality.
- Local Search has intermediate-quality solutions with some variability

• **Insights:**

- MST Approximation sacrifices solution quality for speed.
- Local Search returns near-optimal solutions, and it balances speed and quality.

Local Search Quality to Time



## Reference

- [1]GeeksforGeeks, “Approximate solution for Travelling Salesman Problem using MST,” GeeksforGeeks, Nov. 04, 2013. Accessed: Dec. 02, 2024. [Online]. Available: <https://www.geeksforgeeks.org/approximate-solution-for-travelling-salesman-problem-using-mst/>
- [2]“minimum\_spanning\_tree — SciPy v1.14.1 Manual.” Accessed: Dec. 02, 2024. [Online]. Available: [https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.csgraph.minimum\\_spanning\\_tree.html](https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.csgraph.minimum_spanning_tree.html)