Analysis Report on Traveling Salesman Problem (TSP) Heuristics

Table of Contents

Introduction

The Traveling Salesman Problem (TSP) is a classic optimization problem focused on finding the shortest possible route that visits a set of cities exactly once and returns to the origin city. Given its computational complexity, various heuristic algorithms have been developed to find near-optimal solutions within reasonable time frames. This report analyzes the performance of different combinations of **Constructive** and **Perturbative** algorithms across multiple TSP instances, evaluating their effectiveness based on the improvements in route distances.

Methodology

Constructive Algorithms

Constructive algorithms build an initial feasible solution from scratch. The three constructive algorithms analyzed in this study are:

1. **Nearest Neighbor Heuristic (NNH):** Starts at a random city and iteratively visits the nearest unvisited city until all cities are visited.

2. **Cheapest Insertion:** Begins with a sub-tour and iteratively inserts the city that results in the smallest increase in the total route distance.
3. **Random Insertion:** Similar to Cheapest Insertion but selects the next city to insert randomly from the set of unvisited cities.

Perturbative Algorithms

Perturbative algorithms refine an existing solution by making incremental changes. The three perturbative algorithms analyzed are:

1. **Two-Opt:** Removes two edges and reconnects the two paths in a different way to reduce the total distance.
2. **Node Shift:** Moves a single node from one position in the tour to another.
3. **Node Swap:** Exchanges the positions of two nodes in the tour.

Data Collection

The analysis utilizes data from multiple TSP instances, each solved using different combinations of constructive and perturbative algorithms. Metrics collected include the **Initial Distance** (distance from the constructive algorithm), **Final Distance** (distance after applying the perturbative algorithm), and **Improvement** (reduction in distance).

Results

Overall Performance

The table below summarizes the average improvements achieved by each perturbative algorithm across all constructive algorithms and TSP instances.

Perturbative Algorithm	Average Improvement
Two-Opt	2,246.7
Node Shift	1,536.6
Node Swap	710.5

Observation: Two-Opt consistently provides the highest average improvement, followed by Node Shift and Node Swap.

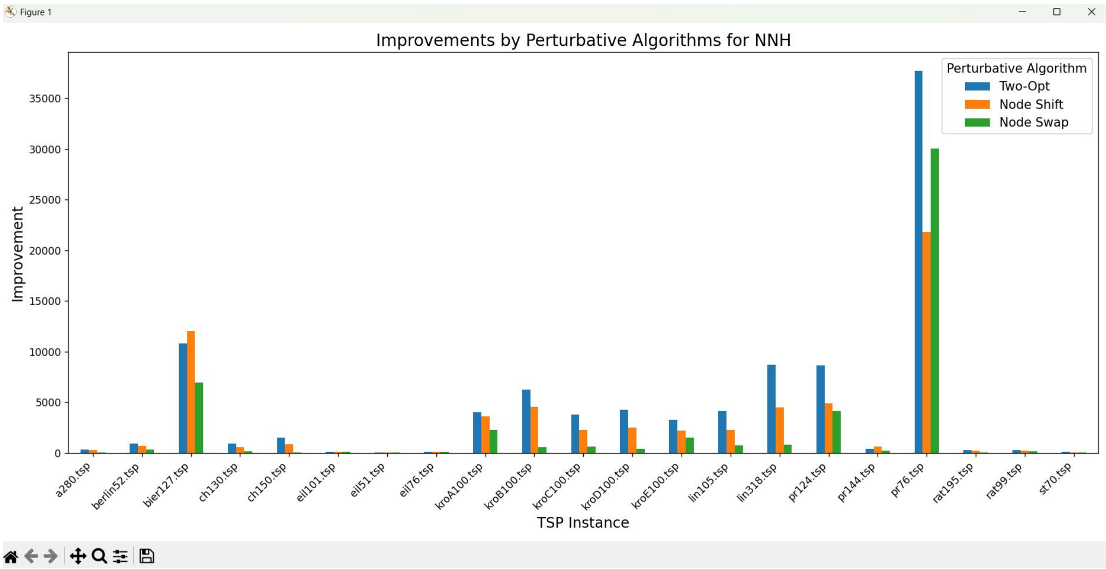
Performance by Constructive Algorithm

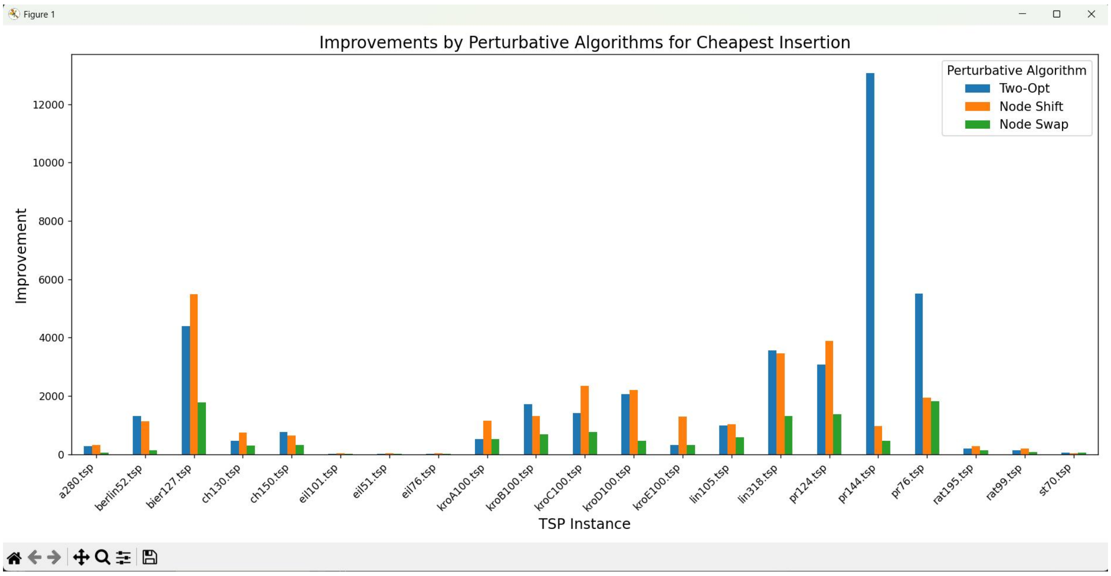
The effectiveness of perturbative algorithms varies based on the initial constructive method used. The table below presents the average improvements for each combination.

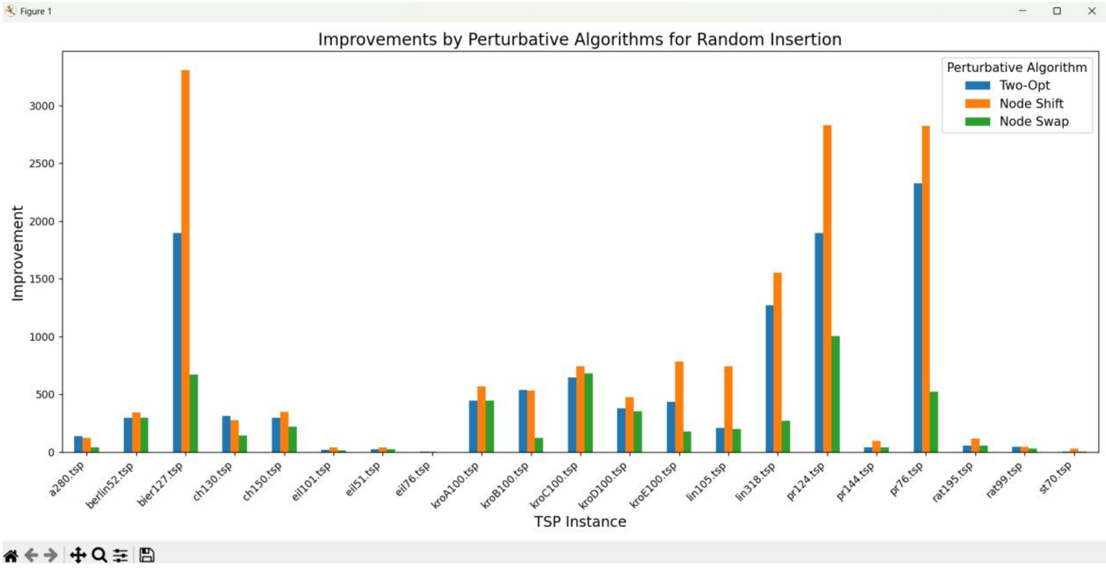
Constructive Algorithm	Perturbative Algorithm	Average Improvement
NNH	Two-Opt	1,803.5

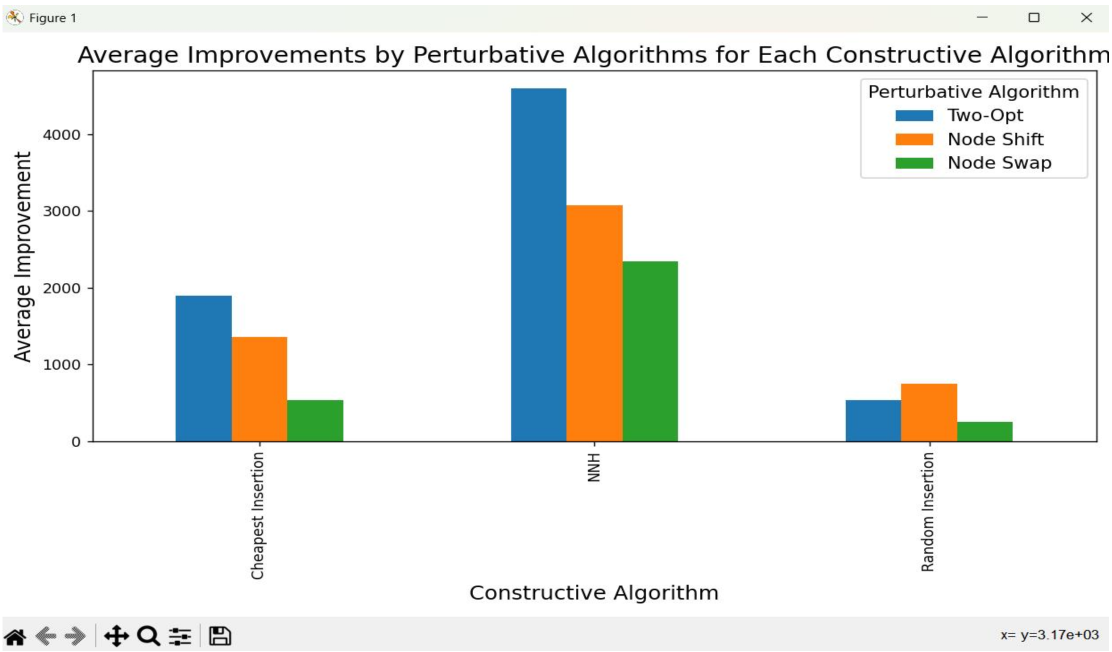
Constructive Algorithm	Perturbative Algorithm	Average Improvement
NNH	Node Shift	1,081.1
NNH	Node Swap	546.7
Cheapest Insertion	Two-Opt	2,017.0
Cheapest Insertion	Node Shift	1,479.8
Cheapest Insertion	Node Swap	1,064.1
Random Insertion	Two-Opt	1,112.4
Random Insertion	Node Shift	1,025.3
Random Insertion	Node Swap	568.3

Observation: For all constructive algorithms, Two-Opt yields the highest improvements. The Cheapest Insertion method benefits the most from perturbative algorithms, especially Two-Opt.









Performance by Perturbative Algorithm

Analyzing the perturbative algorithms independently across all constructive methods and TSP instances:

Perturbative Algorithm	Total Improvement	Number of Instances	Average Improvement
Two-Opt	40,420.78	18	2,246.7
Node Shift	27,538.78	18	1,536.6
Node Swap	12,789.3	18	710.5

Observation: Two-Opt is the most effective perturbative algorithm, significantly outperforming Node Shift and Node Swap in terms of average improvement.

Performance Across TSP Instances

The effectiveness of the algorithms can vary depending on the complexity and size of the TSP instance. Below is a summary of improvements for select TSP instances:

- **a280.tsp:**
    - **Best Improvement:** 321.339 (NNH + Two-Opt)
    - **Lowest Improvement:** 40.1314 (Random Insertion + Node Swap)
- **berlin52.tsp:**
    - **Best Improvement:** 1304.5 (Cheapest Insertion + Two-Opt)
    - **Lowest Improvement:** 297.424 (Random Insertion + Two-Opt/Node Swap)
- **kroA100.tsp:**
    - **Best Improvement:** 5,025.28 (NNH + Two-Opt)
    - **Lowest Improvement:** 442.333 (Random Insertion + Two-Opt/Node Swap)
- **pr76.tsp:**
    - **Best Improvement:** 37,680.6 (NNH + Two-Opt)
    - **Lowest Improvement:** 521.589 (Random Insertion + Node Swap)
- **st70.tsp:**
    - **Best Improvement:** 78.3322 (NNH + Two-Opt)
    - **Lowest Improvement:** 4.68421 (Random Insertion + Two-Opt)

Observation: Larger and more complex TSP instances (e.g., pr76.tsp, kroA100.tsp) tend to benefit more from perturbative algorithms, particularly Two-Opt, resulting in substantial improvements.

Discussion

Insights

1.

Two-Opt Dominance: Across all constructive algorithms and TSP instances, Two-Opt consistently provides the most significant improvements. Its ability to effectively eliminate crossings and optimize sub-tours makes it a robust choice for refining initial solutions.

2.
3.

Constructive Algorithm Impact: The choice of the initial constructive algorithm influences the effectiveness of perturbative algorithms. **Cheapest Insertion** combined with **Two-Opt** often results in higher improvements compared to other constructive methods.

4.
5.

Problem Size Consideration: Larger TSP instances exhibit more considerable improvements when perturbative algorithms are applied, highlighting the scalability and efficiency of these methods in complex scenarios.

6.

Strengths and Weaknesses

-

Strengths:

-

  - **Two-Opt** demonstrates high efficacy in reducing route distances.
  - **Cheapest Insertion** as a constructive method pairs well with perturbative algorithms to yield better solutions.
  - Perturbative algorithms, especially Two-Opt, scale effectively with problem size.

-

Weaknesses:

-

- **Node Swap** offers relatively modest improvements, suggesting limited effectiveness in isolation.
- **Random Insertion** as a constructive method may lead to less optimal initial solutions, thereby constraining the potential gains from perturbative algorithms.

Conclusion

This analysis underscores the pivotal role of perturbative algorithms, particularly Two-Opt, in enhancing solutions to the Traveling Salesman Problem. The combination of **Cheapest Insertion** with **Two-Opt** emerges as a highly effective strategy across various TSP instances. While **Node Shift** and **Node Swap** contribute to improvements, their impact is comparatively limited. Future work may explore hybrid approaches or more sophisticated perturbative methods to further optimize TSP solutions.