Authors:

Edgar Ernesto Rodriguez Pablo

Ryan Vu Hang

Saishnu Ramesh Kumar

CSCI 191T - Bio-Inspired Machine Learning

Project 1 - Ant Colony Optimization & Heuristic Approach For The Traveling Salesman Problem, A Comparison

Abstract:

The Traveling Salesman Problem (TSP) poses a challenge in finding the shortest route among cities while visiting each exactly once. Various algorithms like brute force, greedy, and approximate methods have been used to tackle it. This report explores the Ant Colony Optimization (ACO) algorithm, particularly the Ant System (AS), and compares it with the Nearest Neighbor (NN) approach. The comparison involves simulating 50 cities and computing their shortest routes. For NN, the algorithm iteratively selects the nearest unvisited city, while AS employs multiple ants to explore paths based on pheromone levels and distances. Experimentally, AS generally outperforms NN in total distance covered due to its adaptability and global perspective. NN, though computationally less intensive, suffers from its greedy nature, potentially missing optimal routes. AS's iterative nature, however, may be sensitive to parameter choices, affecting its performance. Overall, the study sheds light on the strengths and weaknesses of NN and AS, suggesting NN for basic optimization with limited resources and AS for complex problems requiring global optimization.

Introduction:

The Traveling Salesman Problem (TSP) is a dilemma that Computer Scientists have faced due to how simple it is to solve but how difficult it is to optimize. While difficult, finding an optimal solution is necessary due to its real-world use cases such as logistics and finding the shortest path. The problem can be stated as finding the shortest possible route with the constraint of only visiting a city (node) exactly once and returning to the original city with a given list of cities and distances (edges) between each pair of cities. As this problem is widespread, there have been many implementations to solve this issue effectively. Some implementations include, brute force, greedy, and approximate. Brute force has limitations due to its inefficiency in generating every possible solution to obtain the shortest path. Certain greedy algorithms like nearest neighbors have limitations due to greedily searching the shortest path of the current city the salesman is at. As for the approximate algorithm, such as 2-opt, is used to check for routes that have been crossed heuristically and to reorder it so that it does not. Aside from those algorithms, this report will cover the Ant System (AS) implementation of the Ant Colony Optimization algorithm and compare its performance with Nearest Neighbor (NN).

Approach:

The approach used to compare AS with NN is through creating a simulation of 50 cities and allowing both algorithms to go through and compute their shortest route. The simulation and environment created were through a graph Abstract Data Type (ADT). To represent the cities within the graph by representing them as nodes and their respective paths as edges with a weight based on the distance between the two cities. To obtain the distance between each node for the algorithm's path, each node will have a randomly generated coordinate in a 500 by 500 graph.

For the implementation of Nearest Neighbors, the algorithm steps are as follows:

1. Start at the initial node of 0 and mark the node as visited.
2. Search through all cities that have yet to be visited.
    a. Compute the cities' distances from the current city.
3. Select the node that has the shortest distance.
    a. Connect the current node with the selected node.
    b. Move to a new node.
    c. Mark the new node as visited.
4. Repeat Step 2 until all cities have been visited.
5. Compute the total weight by summing all weighted edges.

For the implementation of the Ant System, the algorithm steps are to:

1. Initialize N ants and let each ant start at a random node.

2. For each ant, select a new node based on the pheromone levels and distance.
3. Repeat step 2 until all cities have been visited.
4. Update Pheromone Trail:
    a. $p^{k}_{xy} = \frac{(\tau^{\alpha}xy)(\eta^{\beta}xy)}{\sum_{z \in allowedy} (\tau^{\alpha}xy)(\eta^{\beta}xy)}$

Get the distance between two nodes η^{β} and obtain the current pheromone trail from the two nodes τ^{α}, multiply them, divide them by the sum of all pheromone trails, and the distance from the current point to every other node.

5. Update and keep track of the best solution found.
6. If max iterations of the algorithm have been reached, stop the algorithm. Otherwise, proceed again to Step 2.
7. Compute the total weight from the best solution found.

Experiment:

The configuration used for both algorithms is to generate the weights of the edges based on the distance of the city's randomly generated coordinates where the coordinate range is (0,500) for both axes. But for the Ant System algorithm, the number of ants used is 50 and the iteration is 500. The pheromone evaporation rate used is 0.2, and the influence factors used are alpha (α) = 1 and beta (β) = 2. The tech stack used for this experiment is the following, Python 3.11 to write the implementation, and the libraries used are NetworkX to instantiate the graph data structure and Matplotlib to visualize the graph itself.

Result & Discussion:

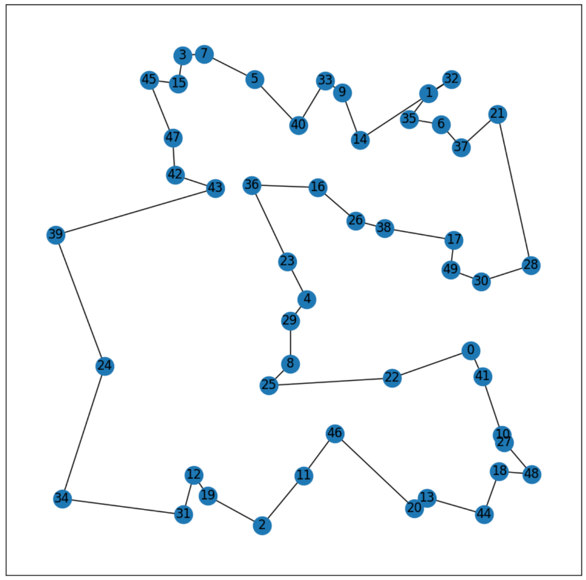

Figure 1. 50 ants traveling through 50 cities

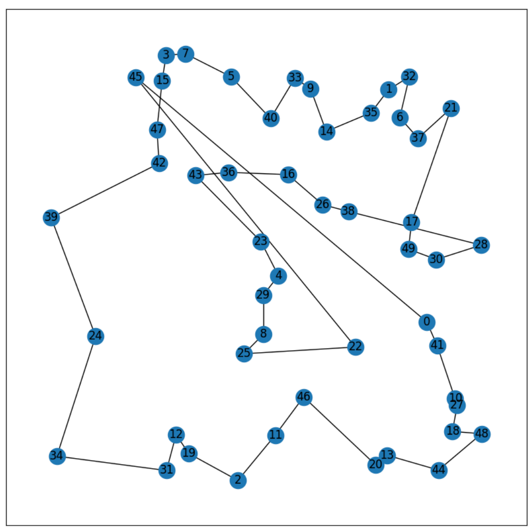

Figure 2. Nearest Neighbor greedy algorithm traveling through 50 cities

The result of the experiment is that AS generally outperforms NN in terms of total distance. AS's global perspective and adaptability enable it to explore diverse solutions more effectively compared to NN. While NN is computationally less intensive compared to AS due to its simple iterative nature, AS exhibits greater robustness to changes in problem instances and parameters compared to NN. Its ability to dynamically adjust to varying conditions allows it to handle uncertainties and fluctuations in the search space more efficiently. The limitations of NN are mainly due to its greedy nature and by only looking at the shortest edge from the current city, one city may be missed which can cause overlap when having to go back to the specific city at the end of the path. As for AS, its limitation is its iterative nature of computing the optimal solution. One factor that may have influenced the results is the parameter choice for AS. The sensitivity of the variables may improve or worsen the algorithm due to the choice of prioritizing exploitation or exploration.

Conclusion:

Overall, this study contributes to the understanding of the strengths and weaknesses of Nearest Neighbor and Ant Colony Optimization algorithms in solving combinatorial optimization problems such as the TSP. NN is suitable for solving basic optimization problems where computational resources are limited or real-time performance is critical. On the other hand, AS is more suited for complex combinatorial optimization problems where finding globally optimal solutions is paramount. The findings provide valuable insights for researchers and practitioners in selecting suitable algorithms for similar optimization tasks based on specific requirements and constraints.

Reflection:

Overall, while everything went well with the presentation, Two things that I specifically could have improved upon would be discussing my project with my colleague. There was a limited amount of engagement after going through all the slides, I found myself lacking topics or concepts to discuss. One way I could have improved this is to talk to the audience about my project or give them questions to see if they understood my project. Furthermore, I do believe that my content was quite difficult to understand due to unclear explanations and the way that I speak. I was not verbal enough to explain the algorithms along with the results. To improve this issue, I now updated the slides to contain concise bullet points and visually appealing slides. Without a doubt, when preparing slides to present my ideas and findings, I am committed to implementing these actions to overall improve engagement and clarity within my presentation.

External Links

Referenced Scholarpedia and Wikipedia for pseudo code of algorithm.

Referenced the ACO Book for different types of parameters to use within the AS algorithm.