Approaches to Solving the MinMax Single-Depot Multiple Traveling Salesman Problem: Genetic Algorithm and Ant Colony Optimization

Adriana Vlad
Faculty of Computer Science
Computational Optimization and Artificial Intelligence
Iasi, Romania
adriana.vlad157@gmail.com

Andreea-Daniela Lupu
Faculty of Computer Science
Computational Optimization and Artificial Intelligence
Iasi, Romania
dandreaalupu@yahoo.com

***Abstract*—This paper presents two hybrid approaches for solving the Min-Max Single-Depot Multiple Traveling Salesman Problem (TSP), leveraging Ant Colony Optimization (ACO) and Genetic Algorithm with Hill Climbing (GAHC). We explore the methods using a set of instances, including `eil51`, `berlin52`, `eil76`, and `rat99`, exploring multiple configurations with different numbers of salesmen (2, 3, 5, and 7). In the GAHC method, we utilize tournament selection, Similar Tour Crossover (STX), and dynamic mutation rate adjustments and 4 types of mutations, while ACO incorporates genetic algorithm elements and two-opt local search before pheromone updates. Our results demonstrate significant performance improvements, especially in the `EIL76 M3` instance, where the hybridized ACO and GAHC approach achieved a tour length of 195.7, outperforming all found benchmarks. The integration of local search techniques and the hybridization of ACO with GA elements prove to be highly effective in tackling the Min-Max TSP, offering a promising approach to combinatorial optimization problems.**

***Index Terms*—ACO, Ant Colony Optimization, Hill Climbing, Hybrid, Optimization, Min-Max TSP, TSP, GA, Genetic Algorithm, Traveling Salesman Problem**

I. Introduction

In this article, we present two solutions for the Min-Max Single-Depot Multiple Traveling Salesmen Problem (mTSP) and test them using several instances. Specifically, we focus on the following instances: eil51, berlin52, eil76, and rat99. For each of these, we explore four different values for the number of salesmen (i.e., 2, 3, 5, and 7), resulting in a total of 16 distinct multiple-TSP instances. In each case, the depot city is considered to be the first city from the list of cities.

To solve this problem, we employ Ant Colony Optimization (ACO) and Genetic Algorithm with Hill Climbing (GAHC). Both methods are enhanced using parallelization via threads to improve efficiency.

In GAHC, the first 1% of the population is copied to the next generation in an elitist manner and tournament selection is performed to obtain the remaining 99% of the population. While processing the population in a parallelized manner, a novel crossover technique called Similar Tour Crossover (STX) is employed, as well as 4 types of mutation. The top 1% of the population is updated only if results improve, and the remaining 99% is replaced entirely no matter the results. Every 10-th generation, we adjust the mutation rate based on the population's diversity (as measured by changes in standard deviation). When the change is positive, indicating that the population diversity is increasing, Hill Climbing (HC) is applied to further refine the solutions and mutation probability is decreased, otherwise this probability increases. HC is applied on the top 1% of the population, as well as another random 1%.

In ACO, the algorithm begins by processing the ant population in a tipical manner. More specifically each unassigned city is given a selection probability based on pheromone values, a city is selected using roulette wheel selection and then is added to the curent shortest tour/route with 99% probability, otherwise being added to a random tour. Before the pheromone update, however, GA 2O and HC are used. Every 5th iteration for 50 generations, GA is applied, but with an elitism of 2%. To refine the solutions, we employ two-opt(2O), a local search optimization technique, which iteratively swaps pairs of cities in a tour to reduce the overall tour length. For further refinement, every 10th iteration HC is used on a random 1% of the population as well as the best solution so far.

II. Problem Definition

The Min-Max Single-Depot Multiple Traveling Salesmen Problem (Min-Max multiple TSP) is a variant of the multiple-TSP, where the goal is to equally distribute the workload among the salesmen. Specifically, it aims to minimize the maximum tour length of any salesman. In other words, the problem seeks to find the set of tours for each salesman such that the longest tour among all salesmen is as short as possible, with the idea of minimizing the overall cost of visiting all cities.

In this study, we work with four different TSP instances: eil51, berlin52, eil76, and rat99, each representing a set of cities that need to be visited by salesmen. For each instance, we experiment with varying the number of salesmen (2, 3, 5, 7). The distribution of cities for each instance is shown in the following figures. These visualizations provide insight into how the cities are arranged spatially and how the problem becomes more complex as the number of salesmen increases.

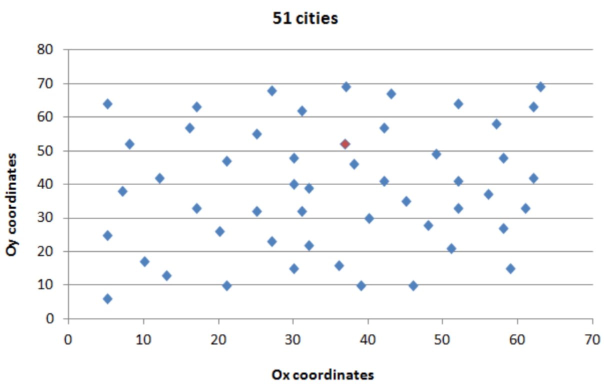

Fig. 1. City distribution for the eil51 instance with 2, 3, 5, and 7 salesmen.

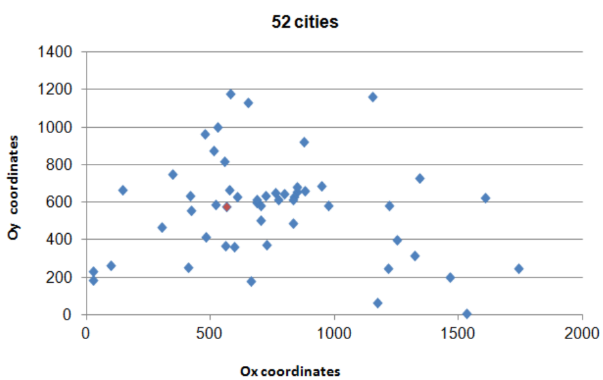

Fig. 2. City distribution for the berlin52 instance with 2, 3, 5, and 7 salesmen.

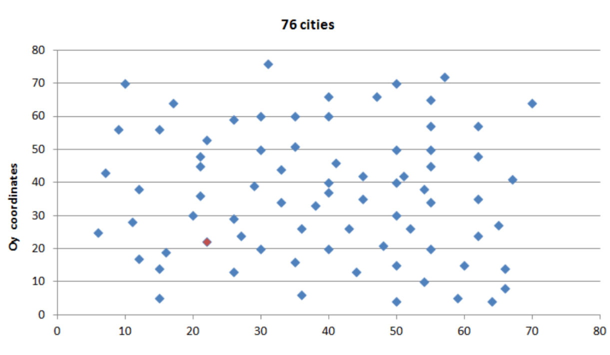

Fig. 3. City distribution for the eil76 instance with 2, 3, 5, and 7 salesmen.

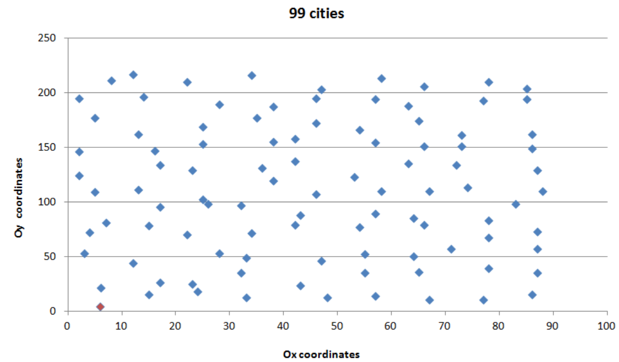

Fig. 4. City distribution for the rat99 instance with 2, 3, 5, and 7 salesmen.

In our approach, we assign a number starting from 0 to each city from the well-known instances.

III. Solution Representation

The solution is represented using a chromosome structure. Each chromosome consists of multiple routes/tours, one for each salesman, where a route is an ordered sequence of cities assigned to a specific salesman. Cities are represented in numerical, integer, form. Each route starts and ends at the depot (city ‘0‘). The structure also contains key attributes such as the total cost, fitness (inverse of the longest route), and amplitude (difference between the longest and shortest routes), ensuring workload balance among salesmen. The solution representation allows efficient manipulation through genetic operators such as crossover, mutation, and hill climbing, as well as pheromone-based optimization for Ant Colony Optimization (ACO).

IV. Hill Climbing

Hill Climbing (HC) is used in both the Genetic Algorithm (GAHC) and Ant Colony Optimization (ACO) approaches to refine the best solution found so far. In this implementation, best improvement is used, where the algorithm iteratively explores neighboring solutions and selects the best improvement based on the objective function. Specifically, the algorithm generates several neighboring solutions by performing 4 types of small changes to the current solution: swapping cities within a tour, reversing subsections of tours, or moving cities between tours and moving a city within a tour. Each of these modified solutions is evaluated for its fitness, which corresponds to 1 / length of longest tour. The solution that results in the best fitness value is accepted and replaces the old one.

The process continues until no further improvements can be found or a predefined maximum number of iterations is reached, namely 5 for GA and 10 for ACO. This ensures that the best solution is progressively refined, leading to a more optimal configuration.

- **Reverse subsection within Tours:** Performs a reverse operation between positions i and j within the tour. If this swap results in a better solution (lower cost), the solution is updated.
  Example of the swap operation:

- **Initial Route:** `[0, 1, 2, 3, 4, 5]`
- **After Swap (start = 1 and end = 4):** `[[0, 4, 3, 2, 1, 5]]`

- **City Movement within Routes:** Moves a city from one position to another within the same route and evaluates the fitness of the new solution. Again, if an improvement is found, the solution is updated. This step helps explore alternative city arrangements within routes.
  - **Initial Route:** `[0, 1, 2, 3, 4, 5]`
  - **After Move (City 2 moved to position 4):** `[0, 1, 3, 4, 2, 5]`

  In this case, city 2 is moved from its initial position and placed at position 4. The other cities between positions 1 and 4 are shifted accordingly to accommodate the new position.
- **Simple City Swaps:** Iterates over all pairs of cities i and j and swaps them. If the resulting solution improves (lower cost), the solution is updated.
  - **Initial Route:** `[0, 1, 2, 3, 4, 5]`
  - **After Swap (City 1 and City 4 swapped):** `[0, 4, 2, 3, 1, 5]`

  The cities are simply swapped without any additional shifts or reverse operations. This step helps to further refine the solution by considering simple swaps.
- **City Transfer between Routes:** The final loop performs *inter-route city transfers*. It moves a city from one route to another, shifting other cities as necessary. This operation aims to balance the workload among routes by redistributing cities. After each transfer, the fitness of the solution is evaluated, and improvements are applied.

V. Similar Tour Crossover

The Similar Tour Crossover (STX) is a crossover method used for a different representation of the problem in a different article ([1]) to replicate mTSP solutions from two parents. This crossover technique is designed to combine the features of both parents by selecting the most similar tours/routes.

- **Step 1: Tour Selection:** Tours are chosen in order from the first parent, and then a tour is selected from the second parent that has the maximum number of common cities with the selected tour from the first parent.
- **Step 2: Two-Point Crossover:** A two-point crossover is applied between the selected tours, in a typical manner.
- **Step 3: Repeating for all Tours:** This process is repeated until the child has a tour for each salesman. Along the way, cities are only assigned to new tours if they weren't already assigned to that child to avoid repeating cities. This leaves the problem of missing cities from the child solution.
- **Step 5: Assigning Remaining Cities:** The remaining cities are added using a greedy approach. Each missing city is inserted into a route by evaluating all possible positions, and the city is placed in the position that causes the least increase in the total cost.

The method is called Similar Tour Crossover (STX) because it emphasizes combining the most similar tours from both parents, ensuring that the child inherits beneficial patterns from both parent solutions. This process improves the solution by transferring good characteristics between parents.

Other crossover techniques, such as PMX (partically mapped crossover), were initially tested, but results seemed much better and convergence much quicker with STX.

VI. Mutation

Mutation is applied to the population in order to introduce diversity and explore the solution space. A mutation probability pm is used to decide whether a given individual will undergo mutation. In the case of GA this pm is variable, increasing and decreasing over generations, starting at a value of 0.05. For ACO, the value is fixed at 0.001. "Chance" is a natural number generated randomly. The mutation process is divided into four distinct types, much like HC:

- **Reverse Subtour (Chance $\in [0, 0.25)$):** A random route is selected, and a subsequence of cities is reversed to create a new solution.
- **Simple Swap (Chance $\in [0.25, 0.50)$):** A simple swap of two cities within the same route is performed.
- **City Movement (Chance $\in [0.50, 0.75)$):** A city is moved from one position to another within the same route, adjusting the order of the cities.
- **City Transfer $\in [0.75, 1)$:** A city is moved from one route to another, helping to balance the workload between routes.

Mutation is attempted V/2 times, where V is the number of cities in an instance, per child, with each attempt having a chance pm to lead to a mutation. The process allows for continuous refinement of the population, ensuring that a wide range of potential solutions is explored.

VII. Genetic Algorithm

A. Approach

In the Genetic Algorithm with Hill Climbing (GAHC) approach for solving the Min-Max Multiple Traveling Salesman Problem (Min-Max mTSP), the solution process begins with a population of randomly initialized chromosomes. Each chromosome represents a potential solution where cities are divided among salesmen in a balanced manner, aiming to minimize the maximum tour length across all salesmen.

- **Initialization:** The first step in the GAHC algorithm is to initialize the population of chromosomes. Each chromosome has a set of routes that represent the tours taken by the salesmen. The cities are randomly shuffled to introduce variability, ensuring that the tours are well-distributed among the salesmen. The population is initialized such that the total number of cities is evenly divided among the salesmen, with any extra cities being distributed across the routes. This initialization is based on the blind assumption that routes are likely to contain similar numbers of cities.

- **Selection:** The first 1% of the population is directly copied to the new population. For the remaining 99%, tournament selection with a size of 5 is applied to choose which chromosomes are copied to the new population. The tournament selection process selects parents based on their fitness, favoring chromosomes with better fitness values.
- **Parallelized Processing:** The population is processed in parallel using multiple threads, optimizing the efficiency of the algorithm. Random pairs of chromosomes in the new population are paired and sent to their own thread where crossover and mutation are applied to introduce diversity into the population.
- **Updating The Population:** After all threads are finished, the first 1% of the population is updated if the resulted chromosomes are better, while the remaining 99% of the population is replaced entirely, regardless of results. This strategy ensures that the best individuals are always preserved while maintaining diversity in the population.
- **Mutation Rate Adjustment:** The mutation rate in the Genetic Algorithm with Hill Climbing (GAHC) is dynamically adjusted every 10th generation based on the diversity of the population. The mutation rate is adjusted within a defined range, 0.00001 and 0.1, ensuring that it remains neither too high nor too low. The mutation is also set to decrease more than it increases over time, by applying a small decrease to it regardless of population diversity.
- **Hill Climbing:** If pm decreases, HC is applied to refine the solutions on a very small part of the population. This process is also optimized using threads. This minimal application of HC ensures that local optima are explored, but that not all chromosomes get stuck in their local optima, avoiding premature convergence.
- **Stopping Criteria:** The algorithm continues for a set number of generations, in our case 1000, or until there is no improvement in the best solution for a certain number of generations (controlled by a `PATIENCE` parameter). This ensures that the algorithm terminates when further improvements are unlikely, preventing unnecessary computations. In our implementation, `PATIENCE` = 50, but this can vary greatly depending on the observed tendencies of the algorithm.

B. General experiment observations

Various initial mutation probabilities (pm) were tested to assess their impact on performance. The starting value $pm = 0.05$ generally provided the best results, though values of $pm = 0.1$ and $pm = 0.02$ performed better for the rat99 problem. Also worth noting is that the intensity of the adjustments to pm was typically reduced for larger population sizes.

A crossover probability of 0.8 was used.

The problem instances with 2 or 3 salesmen worked better with larger population sizes, while instances with M = 5 and M = 7 were more permissive.

HC proved most helpful for rat99, but we suspect its impact on smaller datasets is lesser.

Larger population size, such as 500000, yielded more consistent results, converging in fewer generations but requiring more time per generation. The benefits outweighed the detriments primarily for instances with 2 or 3 salesmen. Otherwise the experiments used a population of 100000.

C. Results

Legend:

- **Best:** Best MinMax value
- **Worst:** Worst MinMax value
- **Mean:** Mean of the MinMax values
- **STD:** Standard Deviation of the MinMax values
- **Cost:** Total cost of all tours in the best solution
- **Ampl:** Amplitude of tour costs in the best solution

TABLE I
RESULTS FOR EIL51 GAHC

M	Best	Worst	Mean	STD	Cost	Ampl
2	222.7334	226.1078	223.1338	0.6714	444.3314	0.3434
3	159.5715	160.8840	159.7119	0.3569	473.6398	0.5779
5	118.1338	127.5044	119.9097	1.6171	577.9204	0.9149
7	112.0714	112.0714	112.0714	0.0000	711.7313	5.6139

TABLE II
RESULTS FOR BERLIN-52 GAHC

M	Best	Worst	Mean	STD	Cost	Ampl
2	4110.2130	4133.2437	4114.1367	7.2104	8207.1066	2.4858
3	3069.5860	3184.1840	3084.9340	30.7019	9135.6780	5.4236
5	2440.9220	2441.3930	2440.9850	0.1627	11660.2000	28.5375
7	2440.9220	2440.9220	2440.9220	9.25E-13	14380.4800	216.5716

TABLE III
RESULTS FOR EIL76 GAHC

M	Best	Worst	Mean	STD	Cost	Ampl
2	280.8539	284.5992	282.9902	1.5006	561.0754	0.1397
3	195.7222	199.7368	196.9699	0.7115	583.6683	0.4127
5	143.8255	151.0303	146.1351	1.6285	713.1630	0.8589
7	127.5618	131.9257	128.5270	0.9819	855.5257	0.4158

TABLE IV
RESULTS FOR RAT99 GAHC

M	Best	Worst	Mean	STD	Cost	Ampl
2	665.9909	679.6157	670.2171	5.1341	1331.7559	0.1595
3	517.7230	523.0676	518.8815	1.8277	1546.7431	0.1847
5	451.5578	460.1203	458.5802	2.3297	2251.2721	1.2706
7	437.6538	442.5022	440.5418	1.6941	2959.6470	1.0685

- For EIL-51 the average time is 33.9071 seconds. For 3 salesmen, the average time is 25.8541 seconds. For 5 salesmen, the average time is 49.8885 seconds. For 7 salesmen, the average time is 49.3779 seconds.

- For BERLIN-52, with 2 salesmen, the average time is 49.2367 seconds. For 3 salesmen, the average time is 74.4544 seconds. For 5 salesmen, the average time is 51.9905 seconds. For 7 salesmen, the average time is 41.5773 seconds.
- For EIL-76, with 2 salesmen, the average time is 123.7606 seconds. For 3 salesmen, the average time is 69.7660 seconds. For 5 salesmen, the average time is 78.9204 seconds. For 7 salesmen, the average time is 95.6196 seconds.
- For RAT-99, with 2 salesmen, the average time is 110.8684 seconds. For 3 salesmen, the average time is 120.5096 seconds. For 5 salesmen, the average time is 115.7153 seconds. For 7 salesmen, the average time is 115.9439 seconds.

D. GAHC Routes

1) GAHC EIL-51:

- **M = 2:**
  - Salesman 1: 0 → 21 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 2 → 27 → 30 → 7 → 25 → 6 → 22 → 42 → 23 → 13 → 5 → 47 → 0
  - Salesman 2: 0 → 26 → 50 → 45 → 11 → 46 → 3 → 17 → 24 → 12 → 40 → 39 → 18 → 41 → 43 → 16 → 36 → 14 → 44 → 32 → 38 → 9 → 48 → 4 → 37 → 10 → 31 → 0
- **M = 3:**
  - Salesman 1: 0 → 31 → 10 → 37 → 4 → 36 → 14 → 44 → 32 → 38 → 9 → 48 → 8 → 29 → 33 → 49 → 20 → 15 → 1 → 0
  - Salesman 2: 0 → 50 → 45 → 11 → 46 → 17 → 3 → 16 → 43 → 41 → 18 → 39 → 40 → 12 → 24 → 13 → 5 → 26 → 0
  - Salesman 3: 0 → 47 → 22 → 23 → 42 → 6 → 25 → 7 → 30 → 27 → 2 → 35 → 34 → 19 → 28 → 21 → 0
- **M = 5:**
  - Salesman 1: 0 → 21 → 2 → 27 → 30 → 7 → 25 → 6 → 42 → 23 → 22 → 47 → 0
  - Salesman 2: 0 → 31 → 10 → 37 → 4 → 48 → 9 → 38 → 32 → 44 → 14 → 36 → 11 → 45 → 0
  - Salesman 3: 0 → 5 → 13 → 24 → 12 → 41 → 43 → 16 → 0
  - Salesman 4: 0 → 26 → 50 → 46 → 3 → 18 → 39 → 40 → 17 → 0
  - Salesman 5: 0 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 0
- **M = 7:**
  - Salesman 1: 0 → 21 → 30 → 27 → 2 → 19 → 35 → 34 → 28 → 1 → 10 → 0
  - Salesman 2: 0 → 15 → 8 → 49 → 20 → 33 → 29 → 38 → 9 → 0
  - Salesman 3: 0 → 7 → 25 → 6 → 22 → 42 → 23 → 0
  - Salesman 4: 0 → 26 → 47 → 13 → 24 → 12 → 40 → 17 → 45 → 0
  - Salesman 5: 0 → 50 → 46 → 3 → 18 → 41 → 16 → 5 → 0
  - Salesman 6: 0 → 39 → 0
  - Salesman 7: 0 → 31 → 11 → 36 → 43 → 14 → 44 → 32 → 48 → 4 → 37 → 0

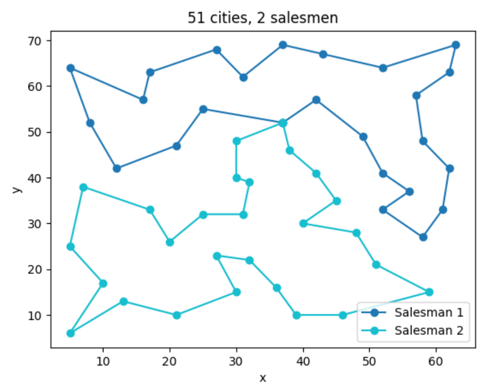

Fig. 5. Route for EIL-51 with M=2

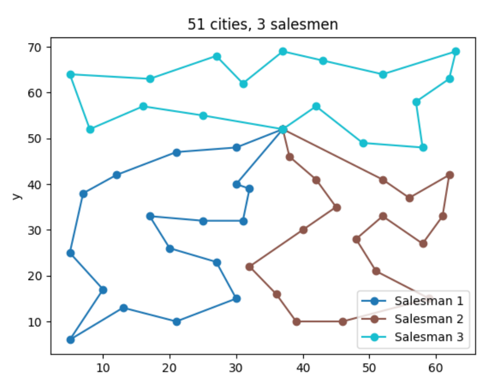

Fig. 6. Route for EIL-51 with M=3

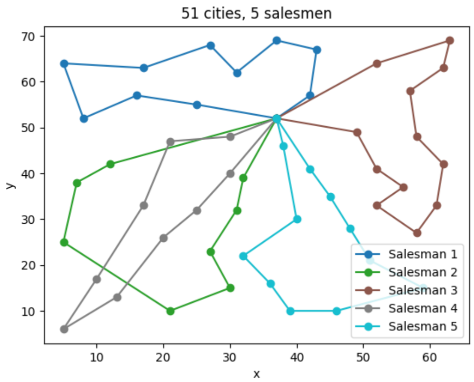

Fig. 7. Route for EIL-51 with M=5

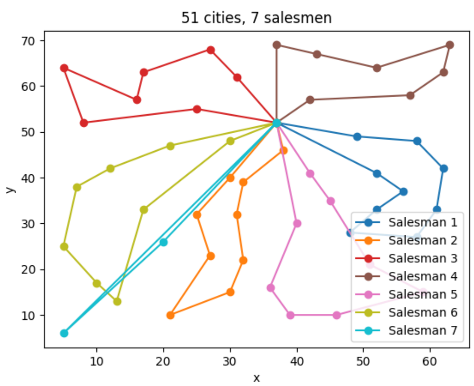

Fig. 8. Route for EIL-51 with M=7

2) GAHC BERLIN-52:

- **M=2:**
  - Salesman 1: [0, 21, 17, 30, 20, 29, 41, 1, 6, 16, 2, 44, 18, 40, 7, 8, 9, 32, 42, 14, 4, 37, 36, 39, 38, 33, 34, 35, 48, 31, 0]
  - Salesman 2: [0, 43, 45, 47, 23, 5, 3, 24, 11, 50, 10, 51, 13, 12, 26, 27, 25, 46, 28, 15, 49, 19, 22, 0]
- **M=3:**
  - Salesman 1: [0, 21, 17, 16, 2, 44, 18, 40, 7, 8, 9, 32, 42, 3, 5, 14, 4, 23, 37, 39, 38, 35, 34, 48, 31, 0]
  - Salesman 2: [0, 30, 20, 41, 6, 1, 29, 28, 46, 25, 15, 49, 19, 22, 0]
  - Salesman 3: [0, 43, 45, 27, 26, 12, 13, 51, 10, 50, 11, 24, 47, 36, 33, 0]
- **M=5:**
  - Salesman 1: [0, 48, 31, 44, 18, 40, 7, 8, 9, 32, 42, 3, 5, 14, 4, 37, 39, 38, 35, 0]
  - Salesman 2: [0, 34, 33, 36, 47, 23, 24, 27, 26, 10, 50, 11, 0]
  - Salesman 3: [0, 43, 15, 25, 12, 13, 46, 45, 0]
  - Salesman 4: [0, 51, 0]
  - Salesman 5: [0, 21, 30, 17, 2, 16, 20, 41, 6, 1, 29, 28, 49, 19, 22, 0]

- **M=7:**
  - Salesman 1: [0, 22, 30, 20, 29, 41, 1, 6, 16, 2, 17, 31, 48, 35, 0]
  - Salesman 2: [0, 38, 37, 4, 42, 32, 9, 8, 7, 40, 44, 18, 0]
  - Salesman 3: [0, 34, 33, 36, 14, 3, 11, 50, 10, 27, 45, 43, 0]
  - Salesman 4: [0, 39, 47, 24, 46, 28, 49, 19, 21, 0]
  - Salesman 5: [0, 23, 5, 0]
  - Salesman 6: [0, 15, 25, 26, 12, 13, 0]
  - Salesman 7: [0, 51, 0]

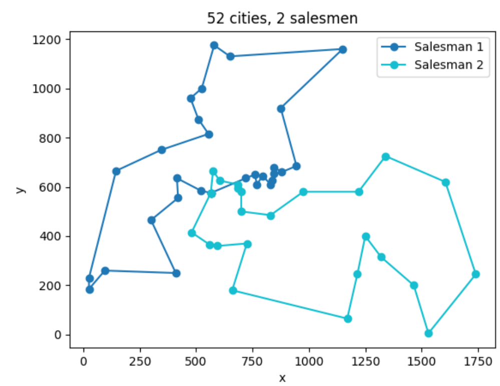

Fig. 9. Route for BERLIN-52 with M=2

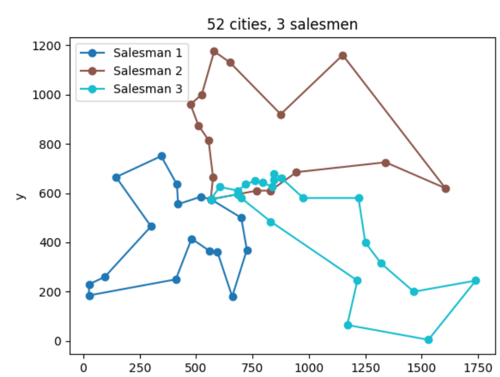

Fig. 10. Route for BERLIN-52 with M=3

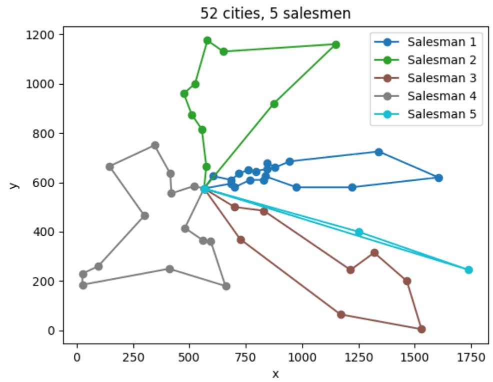

Fig. 11. Route for BERLIN-52 with M=5

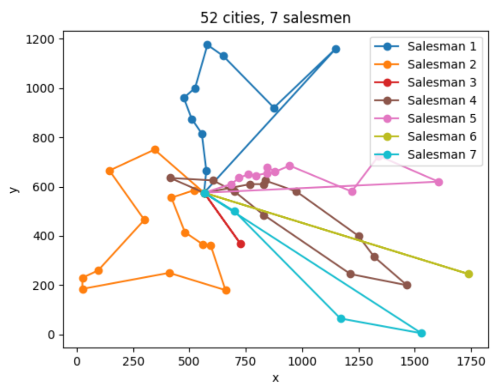

Fig. 12. Route for BERLIN-52 with M=7

3) GAHC EIL-76:

- **M = 2:**
  - Route 1: [0, 21, 63, 41, 42, 40, 55, 22, 48, 23, 17, 49, 24, 54, 30, 9, 57, 71, 38, 8, 31, 43, 2, 39, 11, 16, 25, 66, 3, 75, 74, 67, 5, 50, 15, 62, 32, 0]
  - Route 2: [0, 61, 27, 60, 20, 46, 35, 68, 70, 59, 69, 19, 36, 4, 14, 56, 12, 53, 18, 13, 58, 65, 64, 37, 10, 52, 6, 34, 7, 45, 33, 51, 26, 44, 28, 47, 29, 73, 1, 72, 0]
- **M = 3:**
  - Route 1: [0, 72, 1, 67, 3, 74, 75, 25, 66, 45, 7, 18, 53, 12, 56, 14, 4, 36, 19, 69, 59, 70, 68, 35, 46, 20, 60, 21, 0]
  - Route 2: [0, 32, 5, 50, 16, 39, 11, 57, 37, 64, 65, 10, 58, 13, 52, 34, 6, 33, 51, 26, 44, 28, 47, 29, 73, 27, 61, 0]
  - Route 3: [0, 42, 41, 63, 40, 55, 22, 48, 23, 17, 49, 24, 54, 30, 9, 71, 38, 8, 31, 43, 2, 15, 62, 0]
- **M = 5:**
  - Route 1: [0, 72, 67, 74, 3, 44, 26, 51, 33, 45, 7, 18, 53, 12, 56, 14, 28, 47, 29, 1, 0]
  - Route 2: [0, 5, 75, 66, 34, 13, 58, 65, 10, 52, 6, 25, 0]
  - Route 3: [0, 61, 27, 73, 20, 46, 4, 36, 19, 69, 59, 70, 35, 68, 60, 21, 63, 0]
  - Route 4: [0, 62, 15, 2, 43, 31, 8, 24, 54, 49, 17, 23, 48, 22, 55, 40, 41, 42, 0]
  - Route 5: [0, 32, 50, 16, 39, 11, 71, 57, 9, 37, 64, 30, 38, 0]
- **M = 7:**
  - Route 1: [0, 32, 50, 16, 11, 57, 37, 64, 65, 10, 52, 6, 75, 5, 0]
  - Route 2: [0, 61, 27, 60, 20, 46, 35, 68, 70, 59, 69, 19, 36, 4, 47, 73, 0]
  - Route 3: [0, 67, 74, 71, 9, 30, 38, 8, 31, 43, 0]
  - Route 4: [0, 42, 41, 63, 21, 3, 33, 45, 66, 25, 39, 0]
  - Route 5: [0, 72, 1, 29, 44, 28, 14, 56, 12, 53, 18, 7, 51, 26, 0]
  - Route 6: [0, 40, 55, 22, 48, 23, 17, 54, 24, 49, 2, 15, 62, 0]
  - Route 7: [0, 34, 58, 13, 0]

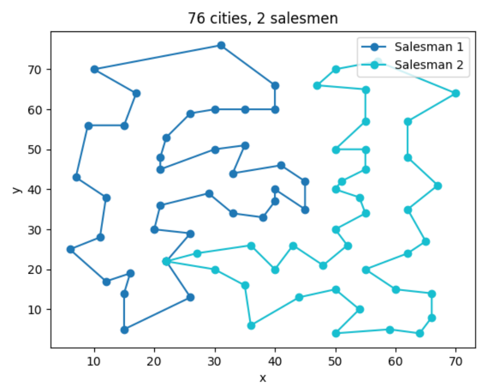

Fig. 13. Route for EIL-76 with M=2

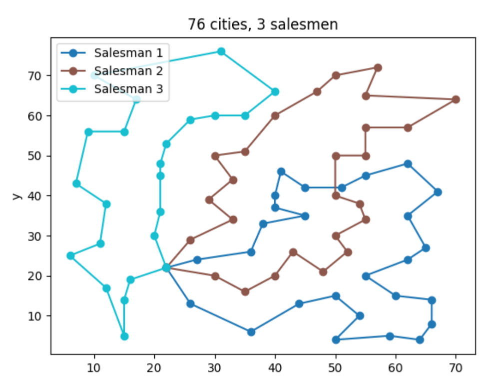

Fig. 14. Route for EIL-76 with M=3

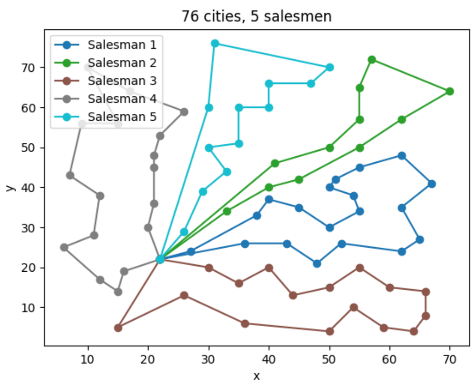

Fig. 15. Route for EIL-76 with M=5

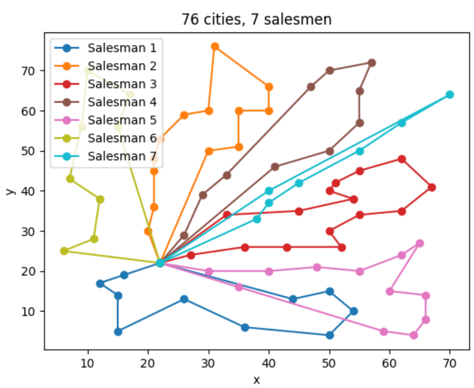

Fig. 16. Route for EIL-76 with M=7

4) GAHC RAT-99:

- **M = 2:**
  - Route 1: 0 → 10 → 11 → 12 → 21 → 20 → 29 → 30 → 31 → 40 → 39 → 38 → 47 → 56 → 55 → 64 → 65 → 74 → 75 → 83 → 84 → 85 → 86 → 87 → 88 → 89 → 98 → 97 → 96 → 95 → 94 → 93 → 92 → 91 → 90 → 82 → 81 → 72 → 73 → 63 → 54 → 45 → 46 → 37 → 28 → 36 → 27 → 18 → 19 → 9 → 0
  - Route 2: 0 → 1 → 2 → 13 → 14 → 22 → 23 → 32 → 33 → 42 → 41 → 49 → 48 → 57 → 58 → 66 → 67 → 76 → 78 → 77 → 68 → 69 → 70 → 79 → 80 → 71 → 62 → 61 → 60 → 59 → 50 → 51 → 52 → 53 → 44 → 43 → 35 → 34 → 26 → 25 → 24 → 15 → 16 → 17 → 8 → 7 → 6 → 5 → 4 → 3 → 0
- **M = 3:**
  - Route 1: 0 → 2 → 11 → 12 → 21 → 20 → 30 → 31 → 40 → 49 → 59 → 60 → 69 → 68 → 77 → 78 → 87 → 88 → 89 → 98 → 97 → 96 → 95 → 86 → 76 → 67 → 66 → 58 → 57 → 48 → 47 → 38 → 39 → 0
  - Route 2: 0 → 3 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 15 → 24 → 25 → 26 → 34 → 35 → 43 → 44 → 53 → 62 → 71 → 80 → 79 → 70 → 61 → 52 → 51 → 50 → 41 → 42 → 33 → 32 → 23 → 22 → 14 → 13 → 0
  - Route 3: 0 → 9 → 18 → 27 → 36 → 45 → 54 → 63 → 73 → 72 → 81 → 82 → 90 → 91 → 92 → 93 → 94 → 85 → 84 → 83 → 75 → 74 → 65 → 64 → 55 → 56 → 46 → 37 → 28 → 29 → 19 → 10 → 1 → 0
- **M = 5:**
  - Route 1: 0 → 9 → 18 → 27 → 36 → 45 → 54 → 63 → 72 → 81 → 90 → 91 → 92 → 82 → 73 → 64 → 55 → 46 → 37 → 20 → 12 → 0
  - Route 2: 0 → 10 → 29 → 39 → 48 → 57 → 58 → 66 → 67 → 75 → 84 → 85 → 94 → 93 → 83 → 74 → 65 → 56 → 47 → 38 → 28 → 19 → 0
  - Route 3: 0 → 1 → 11 → 21 → 31 → 40 → 59 → 69 → 78 → 87 → 96 → 95 → 86 → 76 → 77 → 68 → 49 → 30 → 0
  - Route 4: 0 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 15 → 24 → 25 → 26 → 34 → 35 → 43 → 44 → 53 → 62 → 52 → 51 → 50 → 41 → 42 → 33 → 32 → 23 → 22 → 14 → 13 → 0
  - Route 5: 0 → 60 → 70 → 79 → 88 → 97 → 98 → 89 → 80 → 71 → 61 → 0
- **M = 7:**
  - Route 1: 0 → 9 → 18 → 27 → 45 → 54 → 63 → 72 → 81 → 90 → 91 → 92 → 82 → 73 → 64 → 55 → 46 → 36 → 0
  - Route 2: 0 → 3 → 13 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 15 → 14 → 24 → 25 → 26 → 34 → 35 → 43 → 44 → 53 → 42 → 33 → 23 → 22 → 12 → 11 → 2 → 1 → 0
  - Route 3: 0 → 10 → 21 → 20 → 31 → 40 → 50 → 60 → 61 → 68 → 69 → 70 → 80 → 71 → 62 → 52 → 51 → 41 → 32 → 0
  - Route 4: 0 → 19 → 28 → 37 → 56 → 65 → 74 → 83 → 93 → 94 → 85 → 84 → 75 → 0
  - Route 5: 0 → 47 → 66 → 76 → 86 → 95 → 96 → 0
  - Route 6: 0 → 29 → 39 → 48 → 58 → 77 → 87 → 97 → 78 → 0
  - Route 7: 0 → 30 → 49 → 59 → 79 → 89 → 98 → 88 → 67 → 57 → 38 → 0

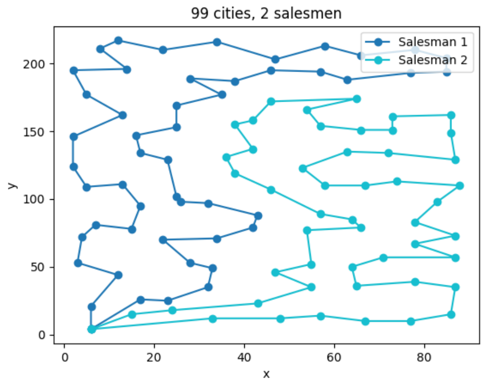

Fig. 17. Route for RAT-99 with M=2

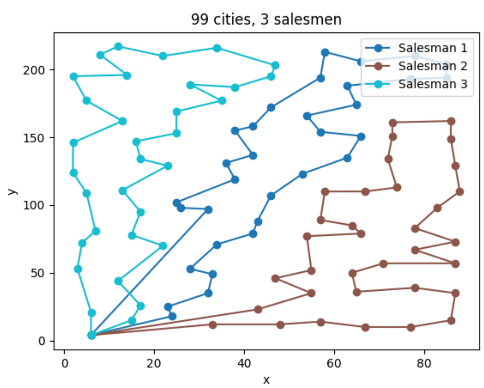

Fig. 18. Route for RAT-99 with M=3

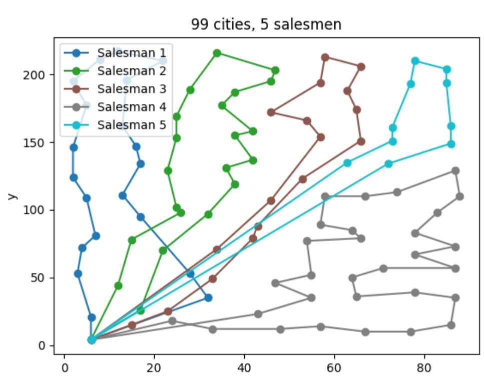

Fig. 19. Route for RAT-99 with M=5

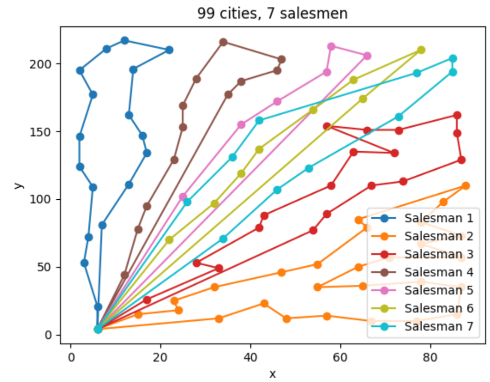

Fig. 20. Route for RAT-99 with M=7

VIII. Ant Colony Optimization

A. Approach

In this implementation of Ant Colony Optimization (ACO), the algorithm operates in a parallelized manner to process the ant population and incorporates elements of the Genetic Algorithm (GA) to enhance its performance. The process begins with the initialization of the ant population and pheromone trails, followed by the iterative update of the pheromone matrix and population of ants through various selection, crossover, and local search strategies.

1) Simple ACO: Each iteration of the ACO algorithm begins by processing the population of ants, where a set of probabilities is calculated for each yet unassigned city based on pheromone levels and distance matrices. These probabilities are then used to guide the ants in constructing their tours, with the help of roulette wheel selection, selecting cities to visit based on the amount of pheromone and the inverse of the distance between cities. The ants' routes are dynamically constructed based on these probabilities, and once all cities are visited, the fitness of each ant is calculated. Of note here is that generally the selected city is added to the current shortest route of the chromosome, with a probability of 99%, otherwise being added to a random route.

2) Ant processing before pheromone update: Every 5th iteration of ACO, 50 generations of GA are applied, but with an elitism of 2%. To refine the solutions further, we employ two-opt(2O), a local search optimization technique often associated with TSP problems. It aims to improve a given solution by iteratively reversing segments of the tour to reduce the overall route length. The basic idea is to remove two edges from the tour, reverse the order of the cities between the two edges, and then reconnect the segments in a different way. Much like in the case of HC, we also imposed a limit to this search, namely a maximum of 10 improvements per call. HC is also applied, every 10 iterations, used only on a random 1% of the population as well as the best solution so far.

3) Pheromone Update: Once the population has been refined, the pheromone matrix is updated based on the performance of the ants. The pheromones are evaporated by a factor of 0.05 to simulate the diminishing effect of past solutions.

Then, pheromone levels are reinforced along the paths taken by the ants. The amount of pheromone deposited on each edge is inversely proportional to the cost of the solution (i.e., the better the solution, the more pheromone is deposited). This reinforces successful paths, encouraging future ants to follow them, while less optimal paths receive less pheromone and are less likely to be chosen in subsequent iterations.

4) ALPHA and BETA: We experimented with many values for these parameters. The best performing idea was to use 0.75 and 2.25 as initial values, and then linearly modify them to 2.25 and 0.75 by the time of the last iteration.

B. General experiment observation

Crossover probability is 0.8 and mutation probability a fixed 0.001.

Unlike the previous method, this one allows for much smaller population sizes, but leads to comparable results for most of the tested problems, often faster. All these tests were done on a population of 5000.

For instances with a known minimum the tests were stopped as soon as they reached that value.

C. Results

TABLE V
RESULTS FOR EIL51 ACO

M	Best	Worst	Mean	STD	Cost	Ampl
2	222.7334	226.1078	222.9942	0.6725	444.3314	0.8014
3	159.5715	165.8248	160.7578	1.9482	472.6555	0.4096
5	118.1338	120.0175	118.4787	0.6712	572.9232	0.9910
7	112.0714	112.0714	112.0714	1.45E-14	688.8383	6.7994

TABLE VI
RESULTS FOR BERLIN-52 ACO

M	Best	Worst	Mean	STD	Cost	Ampl
2	4110.2131	4138.4278	4130.8201	27.4365	8207.1067	1.0719
3	3153.7654	3191.1539	3183.4348	0.0000	8991.4673	3.1755
5	2440.9219	2441.3926	2440.9657	0.1293	11538.3889	69.9270
7	2440.9219	2440.9219	2440.9219	9.25E-13	14538.6214	284.9332

TABLE VII
RESULTS FOR EIL76 ACO

M	Best	Worst	Mean	STD	Cost	Ampl
2	280.8539	285.5191	281.9780	1.0948	561.0754	0.0299
3	195.7222	197.6797	196.4495	0.6345	584.6461	0.1691
5	143.1915	152.8294	146.8785	2.3982	705.8647	0.5528
7	127.5617	128.8025	127.9532	0.3617	852.0726	2.8656

TABLE VIII
RESULTS FOR RAT99 ACO

M	Best	Worst	Mean	STD	Cost	Ampl
2	665.9909	674.0551	667.4624	2.6950	1331.6562	0.2258
3	517.7230	523.6336	519.6527	1.8712	1543.8334	1.2831
5	459.9276	466.6939	463.1918	1.9683	33.0642	2.1487
7	438.5731	442.9396	441.7296	1.1056	2895.9954	3.7024

- For EIL51, with 2 salesmen, the average time obtained is 8.011 seconds, while for 3 salesmen, it is 21.0431 seconds. The times for 5 and 7 salesmen are 31.4197 and 2.8556 seconds, respectively.
- In the case of BERLIN52, with 2 salesmen, the average time is 37.9887 seconds, 38.66846667 seconds for 3 salesmen, 23.49946667 seconds for 5 salesmen, and 0.9545 seconds for 7 salesmen.
- For EIL76, the average time for 2 salesmen is 54.3081 seconds, for 3 salesmen it is 45.4969 seconds, for 5 salesmen it increases to 82.5606 seconds, and for 7 salesmen, the average time is 57.9750 seconds.
- For RAT99, the average time for 2 salesmen is 107.7210 seconds, for 3 salesmen it is 112.5585 seconds, and for 5 and 7 salesmen, the times are 115.7153 and 115.9439 seconds, respectively.

D. Routes

1) ACO EIL-51:

- **M = 2:**
  - Route 1: 0 → 21 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 2 → 27 → 30 → 7 → 25 → 6 → 22 → 42 → 23 → 12 → 13 → 5 → 47 → 0
  - Route 2: 0 → 31 → 10 → 37 → 4 → 48 → 9 → 38 → 32 → 44 → 14 → 36 → 16 → 43 → 41 → 18 → 39 → 40 → 24 → 17 → 3 → 46 → 11 → 45 → 50 → 26 → 0
- **M = 3:**
  - Route 1: 0 → 50 → 45 → 11 → 46 → 17 → 3 → 16 → 43 → 41 → 18 → 39 → 40 → 12 → 24 → 13 → 5 → 26 → 0
  - Route 2: 0 → 31 → 10 → 37 → 4 → 36 → 14 → 44 → 32 → 38 → 9 → 48 → 8 → 29 → 33 → 20 → 49 → 15 → 0
  - Route 3: 0 → 21 → 1 → 28 → 19 → 34 → 35 → 2 → 27 → 30 → 7 → 25 → 6 → 42 → 23 → 22 → 47 → 0
- **M = 5:**
  - Route 1: 0 → 31 → 4 → 36 → 43 → 14 → 44 → 32 → 38 → 9 → 48 → 37 → 10 → 0
  - Route 2: 0 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 2 → 0
  - Route 3: 0 → 21 → 27 → 30 → 7 → 25 → 6 → 42 → 23 → 22 → 47 → 0
  - Route 4: 0 → 26 → 5 → 13 → 24 → 12 → 40 → 3 → 17 → 46 → 11 → 45 → 0

- Route 5: 0 → 50 → 18 → 39 → 41 → 16 → 0

- **M = 7:**
  - Route 1: 0 → 21 → 30 → 27 → 2 → 35 → 34 → 19 → 28 → 20 → 0
  - Route 2: 0 → 7 → 25 → 6 → 22 → 42 → 23 → 17 → 50 → 0
  - Route 3: 0 → 26 → 3 → 40 → 12 → 24 → 13 → 5 → 47 → 0
  - Route 4: 0 → 39 → 0
  - Route 5: 0 → 4 → 14 → 8 → 29 → 33 → 49 → 15 → 1 → 0
  - Route 6: 0 → 11 → 16 → 36 → 43 → 41 → 18 → 46 → 0
  - Route 7: 0 → 31 → 10 → 37 → 48 → 9 → 38 → 32 → 44 → 45 → 0

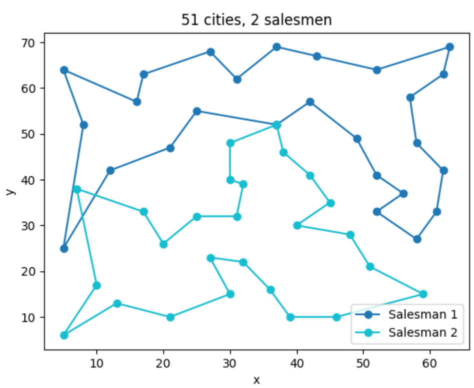

Fig. 21. Route for EIL-51 with M=2

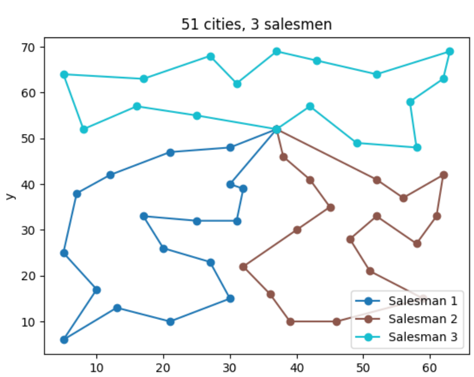

Fig. 22. Route for EIL-51 with M=3

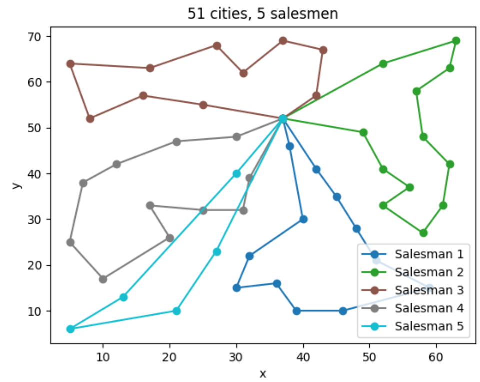

Fig. 23. Route for EIL-51 with M=5

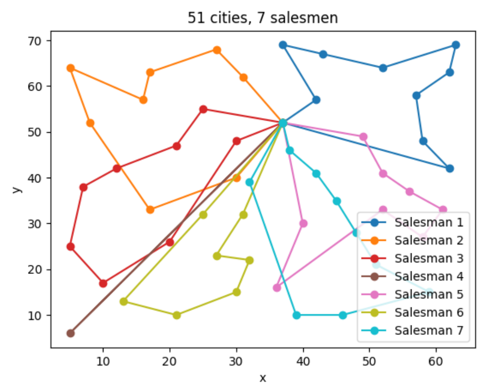

Fig. 24. Route for EIL-51 with M=7

2) ACO BERLIN-52:

- **M = 2:**
  - Route 1: 0 → 31 → 48 → 35 → 34 → 33 → 38 → 39 → 36 → 37 → 47 → 23 → 4 → 14 → 5 → 3 → 42 → 32 → 9 → 8 → 7 → 40 → 18 → 44 → 2 → 16 → 41 → 6 → 1 → 29 → 20 → 0
  - Route 2: 0 → 21 → 17 → 30 → 22 → 19 → 49 → 15 → 28 → 46 → 25 → 27 → 26 → 12 → 13 → 51 → 10 → 50 → 11 → 24 → 45 → 43 → 0
- **M = 3:**
  - Route 1: 0 → 48 → 35 → 34 → 33 → 36 → 24 → 11 → 50 → 10 → 51 → 13 → 12 → 26 → 27 → 45 → 43 → 0
  - Route 2: 0 → 31 → 38 → 39 → 37 → 47 → 23 → 4 → 14 → 5 → 3 → 42 → 32 → 9 → 8 → 7 → 40 → 18 → 44 → 2 → 16 → 0
  - Route 3: 0 → 21 → 17 → 30 → 20 → 41 → 6 → 1 → 29 → 28 → 46 → 25 → 15 → 49 → 19 → 22 → 0
- **M = 5:**
  - Route 1: 0 → 21 → 30 → 17 → 2 → 16 → 20 → 41 → 6 → 1 → 29 → 28 → 49 → 19 → 22 → 0
  - Route 2: 0 → 51 → 45 → 0

- Route 3: 0 → 48 → 38 → 37 → 4 → 24 → 27 → 26 → 12 → 13 → 46 → 15 → 0
- Route 4: 0 → 34 → 33 → 36 → 47 → 23 → 5 → 42 → 32 → 9 → 8 → 7 → 40 → 18 → 44 → 31 → 0
- Route 5: 0 → 35 → 39 → 14 → 3 → 11 → 50 → 10 → 25 → 43 → 0

- **M = 7:**
  - Route 1: 0 → 15 → 49 → 19 → 29 → 22 → 20 → 1 → 0
  - Route 2: 0 → 21 → 30 → 17 → 2 → 16 → 41 → 6 → 28 → 0
  - Route 3: 0 → 33 → 35 → 39 → 4 → 3 → 11 → 50 → 10 → 0
  - Route 4: 0 → 51 → 0
  - Route 5: 0 → 48 → 34 → 38 → 14 → 5 → 24 → 27 → 26 → 12 → 13 → 0
  - Route 6: 0 → 31 → 44 → 18 → 40 → 7 → 8 → 9 → 42 → 32 → 0
  - Route 7: 0 → 36 → 37 → 23 → 47 → 43 → 45 → 25 → 46 → 0

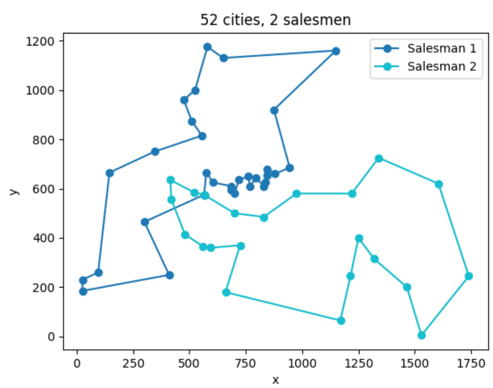

Fig. 25. Route for BERLIN-52 with M=2

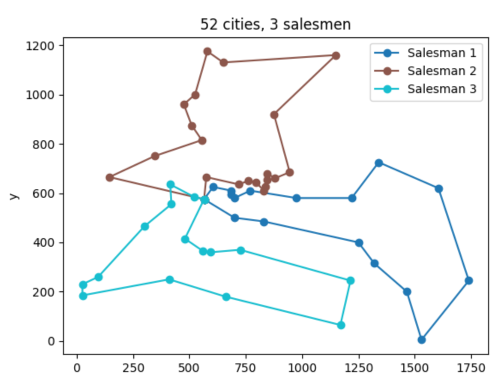

Fig. 26. Route for BERLIN-52 with M=3

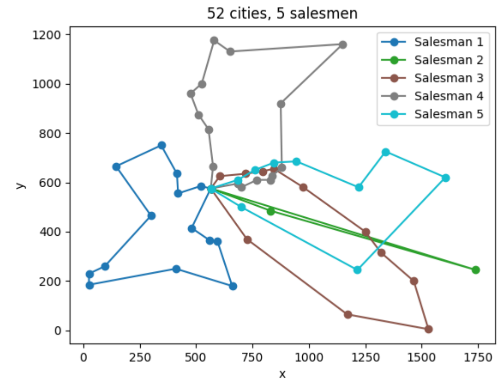

Fig. 27. Route for BERLIN-52 with M=5

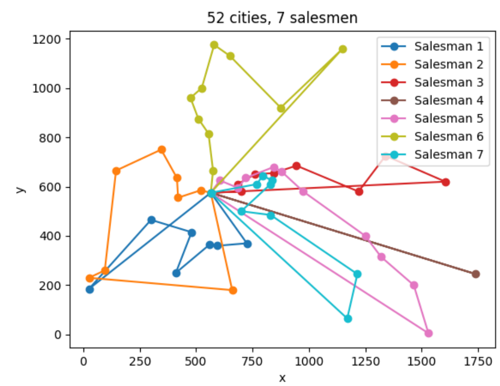

Fig. 28. Route for BERLIN-52 with M=7

3) ACO EIL-76:

- **M = 2:**
  - Route 1: 0 → 21 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 2 → 27 → 30 → 7 → 25 → 6 → 22 → 42 → 23 → 12 → 13 → 5 → 47 → 0
  - Route 2: 0 → 31 → 10 → 37 → 4 → 48 → 9 → 38 → 32 → 44 → 14 → 36 → 16 → 43 → 41 → 18 → 39 → 40 → 24 → 17 → 3 → 46 → 11 → 45 → 50 → 26 → 0
- **M = 3:**
  - Route 1: 0 → 50 → 45 → 11 → 46 → 17 → 3 → 16 → 43 → 41 → 18 → 39 → 40 → 12 → 24 → 13 → 5 → 26 → 0
  - Route 2: 0 → 31 → 10 → 37 → 4 → 36 → 14 → 44 → 32 → 38 → 9 → 48 → 8 → 29 → 33 → 20 → 49 → 15 → 0
  - Route 3: 0 → 21 → 1 → 28 → 19 → 34 → 35 → 2 → 27 → 30 → 7 → 25 → 6 → 42 → 23 → 22 → 47 → 0
- **M = 5:**
  - Route 1: 0 → 31 → 4 → 36 → 43 → 14 → 44 → 32 → 38 → 9 → 48 → 37 → 10 → 0

- Route 2: 0 → 1 → 15 → 49 → 8 → 29 → 33 → 20 → 28 → 19 → 34 → 35 → 2 → 0
- Route 3: 0 → 21 → 27 → 30 → 7 → 25 → 6 → 42 → 23 → 22 → 47 → 0
- Route 4: 0 → 26 → 5 → 13 → 24 → 12 → 40 → 3 → 17 → 46 → 11 → 45 → 0
- Route 5: 0 → 50 → 18 → 39 → 41 → 16 → 0

- **M = 7:**
  - Route 1: 0 → 21 → 30 → 27 → 2 → 35 → 34 → 19 → 28 → 20 → 0
  - Route 2: 0 → 7 → 25 → 6 → 22 → 42 → 23 → 17 → 50 → 0
  - Route 3: 0 → 26 → 3 → 40 → 12 → 24 → 13 → 5 → 47 → 0
  - Route 4: 0 → 39 → 0
  - Route 5: 0 → 4 → 14 → 8 → 29 → 33 → 49 → 15 → 1 → 0
  - Route 6: 0 → 11 → 16 → 36 → 43 → 41 → 18 → 46 → 0
  - Route 7: 0 → 31 → 10 → 37 → 48 → 9 → 38 → 32 → 44 → 45 → 0

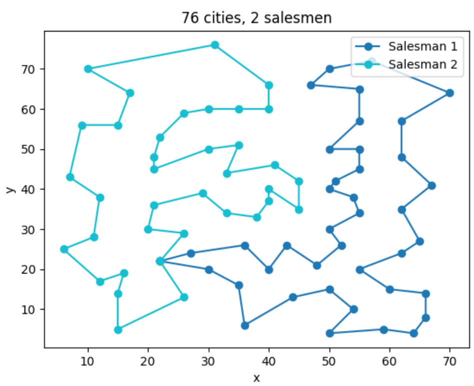

Fig. 29. Route for EIL-76 with M=2

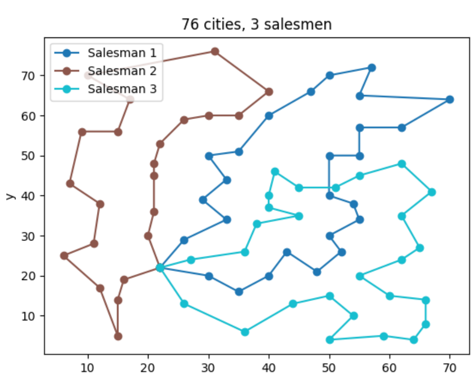

Fig. 30. Route for EIL-76 with M=3

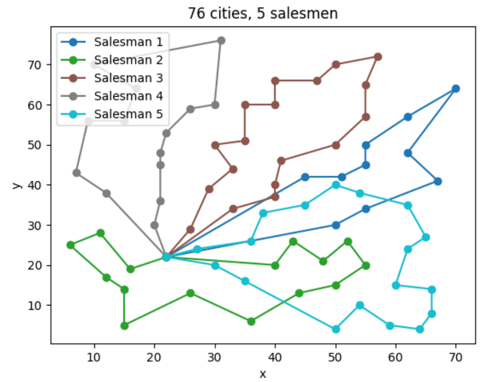

Fig. 31. Route for EIL-76 with M=5

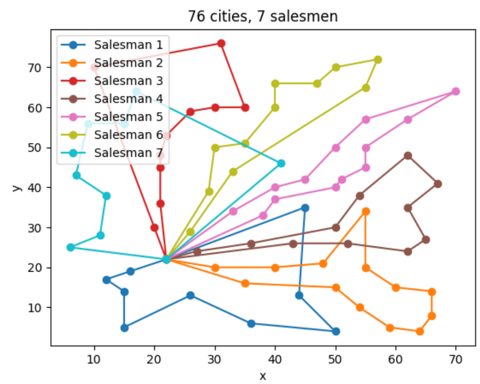

Fig. 32. Route for EIL-76 with M=7

4) ACO RAT-99:

- **M = 2:**
  - Route 1: 0 → 1 → 2 → 13 → 14 → 22 → 23 → 32 → 33 → 42 → 41 → 49 → 48 → 57 → 58 → 66 → 67 → 76 → 78 → 77 → 68 → 69 → 70 → 79 → 80 → 71 → 62 → 61 → 60 → 59 → 50 → 51 → 52 → 53 → 44 → 43 → 35 → 34 → 26 → 25 → 24 → 15 → 16 → 17 → 8 → 7 → 6 → 5 → 4 → 3 → 0
  - Route 2: 0 → 10 → 11 → 12 → 21 → 20 → 29 → 30 → 31 → 40 → 39 → 38 → 47 → 56 → 55 → 64 → 65 → 74 → 75 → 83 → 84 → 85 → 86 → 87 → 88 → 89 → 98 → 97 → 96 → 95 → 94 → 93 → 92 → 91 → 90 → 82 → 81 → 72 → 73 → 63 → 54 → 45 → 46 → 37 → 28 → 36 → 27 → 18 → 19 → 9 → 0
- **M = 3:**
  - Route 1: 0 → 9 → 18 → 27 → 36 → 45 → 54 → 63 → 73 → 72 → 81 → 82 → 90 → 91 → 92 → 93 → 94 → 85 → 84 → 83 → 75 → 74 → 65 → 64 → 55 → 56 → 46 → 37 → 28 → 29 → 19 → 0
  - Route 2: 0 → 1 → 10 → 20 → 39 → 38 → 47 → 48 → 57 → 58 → 66 → 67 → 76 → 86 → 95 →

96 → 97 → 98 → 89 → 88 → 87 → 78 → 77 → 68 → 69 → 60 → 59 → 49 → 40 → 31 → 30 → 21 → 12 → 11 → 2 → 0

- Route 3: 0 → 3 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 15 → 24 → 25 → 26 → 34 → 35 → 43 → 44 → 53 → 62 → 71 → 80 → 79 → 70 → 61 → 52 → 51 → 50 → 41 → 42 → 33 → 32 → 23 → 22 → 14 → 13 → 0

- **M = 5:**
  - Route 1: 0 → 9 → 36 → 73 → 82 → 92 → 93 → 91 → 90 → 81 → 72 → 63 → 54 → 45 → 27 → 18 → 0
  - Route 2: 0 → 3 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 26 → 35 → 33 → 42 → 50 → 58 → 66 → 75 → 74 → 65 → 56 → 47 → 38 → 20 → 0
  - Route 3: 0 → 1 → 11 → 12 → 22 → 23 → 32 → 41 → 51 → 52 → 61 → 70 → 79 → 98 → 97 → 87 → 77 → 0
  - Route 4: 0 → 19 → 28 → 37 → 46 → 55 → 64 → 83 → 84 → 85 → 94 → 95 → 96 → 86 → 76 → 67 → 57 → 48 → 39 → 29 → 10 → 0
  - Route 5: 0 → 2 → 13 → 14 → 15 → 24 → 25 → 34 → 43 → 44 → 53 → 62 → 71 → 80 → 89 → 88 → 78 → 68 → 69 → 60 → 59 → 49 → 40 → 31 → 30 → 21 → 0

- **M = 7:**
  - Route 1: 0 → 18 → 27 → 36 → 45 → 54 → 63 → 72 → 81 → 90 → 91 → 92 → 82 → 73 → 64 → 0
  - Route 2: 0 → 2 → 12 → 22 → 23 → 24 → 25 → 34 → 43 → 44 → 53 → 52 → 62 → 71 → 80 → 79 → 77 → 76 → 67 → 66 → 55 → 46 → 37 → 28 → 0
  - Route 3: 0 → 1 → 10 → 20 → 30 → 49 → 59 → 68 → 87 → 96 → 95 → 86 → 58 → 48 → 39 → 0
  - Route 4: 0 → 3 → 13 → 4 → 5 → 6 → 7 → 8 → 17 → 16 → 15 → 14 → 26 → 35 → 33 → 42 → 50 → 0
  - Route 5: 0 → 9 → 19 → 56 → 65 → 74 → 83 → 93 → 94 → 85 → 84 → 75 → 47 → 0
  - Route 6: 0 → 11 → 21 → 32 → 41 → 51 → 61 → 70 → 89 → 98 → 88 → 78 → 57 → 38 → 0
  - Route 7: 0 → 31 → 40 → 60 → 69 → 97 → 29 → 0

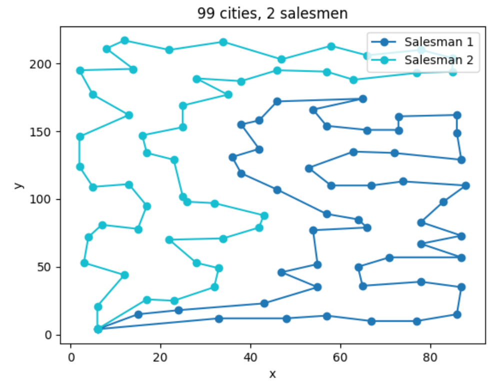

Fig. 33. Route for RAT-99 with M=2

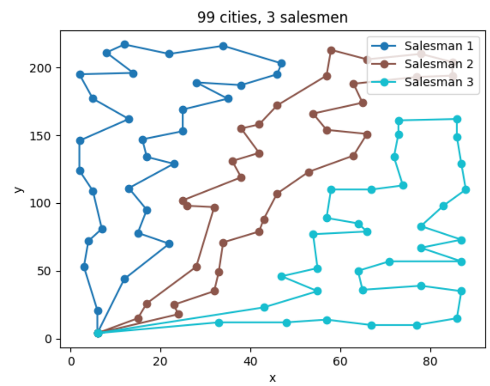

Fig. 34. Route for RAT-99 with M=3

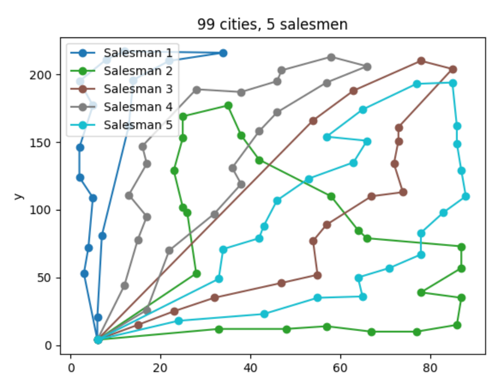

Fig. 35. Route for RAT-99 with M=5

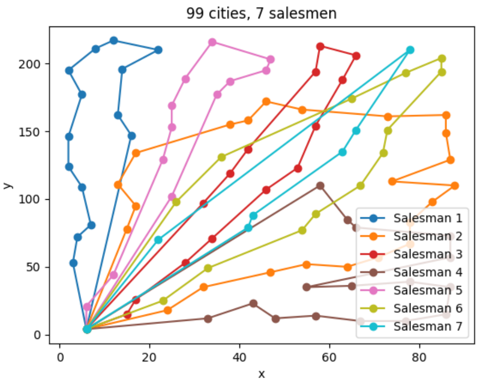

Fig. 36. Route for RAT-99 with M=7

IX. Benchmark comparison

Below is a table comparing our results to the popular CPLEX benchmark, as well as HGA, from article [1], which provided the best results we were able to find.

TABLE IX
Comparison of Results for Different Instances and Algorithms

Instance	eil51				berlin52			
m	2	3	5	7	2	3	5	7
CPLEX	222.7	159.6	124.0	112.1	4110.2	3244.4	2441.4	2440.9
HGA-best	222.7	159.6	118.1	112.1	4110.2	3069.6	2440.9	2440.9
GA-best	222.73	159.57	118.13	112.07	4110.21	3069.59	2440.92	2440.92
GA-avg	223.13	159.71	119.91	112.07	4114.14	3084.93	2440.99	2440.92
ACO-best	222.73	159.57	118.13	112.07	4110.21	3153.77	2440.92	2440.92
ACO-avg	222.99	160.76	118.48	112.07	4130.82	3183.43	2440.97	2440.92

TABLE X
Comparison of Results for Different Instances and Algorithms

Instance	eil76				rat99			
m	2	3	5	7	2	3	5	7
CPLEX	280.9	197.3	150.3	139.6	728.8	587.2	469.3	443.9
HGA-best	280.9	196.7	142.9	127.6	666.0	517.7	450.3	436.7
GA-best	280.85	195.72	143.83	127.56	665.99	517.72	451.56	437.65
GA-avg	282.99	196.97	146.14	128.53	670.22	518.88	458.58	440.54
ACO-best	280.85	195.72	143.19	127.56	665.99	517.72	459.93	438.57
ACO-avg	281.98	196.45	146.88	127.95	667.46	519.65	463.19	441.73

X. Conclusions

In this study, we applied two enhanced optimization methods to solve the Min-Max Single-Depot Traveling Salesman Problem (TSP): **Ant Colony Optimization (ACO)** and **Genetic Algorithm with Hill Climbing (GAHC)**. Both algorithms were adapted and hybridized to improve solution quality and computational efficiency.

By incorporating elements from multiple strategies, such as Genetic Algorithms, with **tournament selection**, many types of mutations and **Similar Tour Crossover (STX)**, as well as **local search techniques** like Two-Opt and Hill Climbing, we were able to improve the convergence speed and solution accuracy. The hybridization, particularly in ACO, provided quick and accurate results in many cases.

The most impresive result of this work was the performance on the **EIL76 M3** problem, where the optimized solution achieved a **tour length of 195.7**, outperforming any benchmark we were able to find, where the best previously reported solution was **196.7**. This represents a significant improvement, showcasing the effectiveness of our proposed methods.

Overall, both approaches proved to be powerful hybrid strategies, yielding impressive results for the Min-Max TSP. This work highlights the potential of hybrid algorithms to improve performance in combinatorial optimization problems and opens avenues for further improvements in both ACO and GA implementations.

References

[1] Mahmoudinazlou, S., Kwon, C. (2023). A Hybrid Genetic Algorithm for the Min-Max Multiple Traveling Salesman Problem. arXiv:2307.07120v3 [cs.NE]. Department of Industrial and Management Systems Engineering, University of South Florida, Tampa, FL 33620, U.S.A., sasanm@usf.edu; Department of Industrial and Systems Engineering, KAIST, Daejeon, 34141, Republic of Korea, chkwon@kaist.ac.kr. Available at: https://arxiv.org/abs/2307.07120

[2] Pereira, S. de C., Pires, E. J. S., Oliveira, P. B. de M. (2023). Ant-Balanced Multiple Traveling Salesmen: ACO-BmTSP. Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300–253 Bragança, Portugal; Departamento de Engenharias, Universidade de Trás-os-Montes e Alto Douro, 5000–811 Vila Real, Portugal; INESC TEC—INESCTechnology and Science, Rua Dr. Roberto Frias, 4200–465 Porto, Portugal. Correspondence: epires@utad.pt.

[3] Harrath, Y., Salman, A. F., Alqaddoumi, A., Hasan, H., Radhi, A. (2019). A novel hybrid approach for solving the multiple traveling salesmen problem. Pages 103-112. Received 21 Jun 2018, Accepted 21 Dec 2018, Published online: 12 Feb 2019. *Journal of Optimization*, 25765299. https://doi.org/10.1080/25765299.2019.1565193

[4] Bryant, Kylie, "Genetic Algorithms and the Travelling Salesman Problem" (2000). HMC Senior Theses. 126. https://scholarship.claremont.edu/hmc_theses/126

[5] Fronita, M., Gernowo, R., Gunawan, V. (2020). Comparison of Genetic Algorithm and Hill Climbing for Shortest Path Optimization Mapping. Master of Information System, School of Postgraduate Studies, Diponegoro University, Semarang – Indonesia 50242; Department of Physics, Faculty of Science and Mathematics, Diponegoro University, Semarang – Indonesia 50275. Corresponding author: mona_fron@yahoo.co.id. Published by EDP Sciences under the terms of the Creative Commons Attribution License 4.0. Available at: http://creativecommons.org/licenses/by/4.0/.

[6] de Melo Menezes, B.A., Herrmann, N., Kuchen, H. et al. High-Level Parallel Ant Colony Optimization with Algorithmic Skeletons. Int J Parallel Prog 49, 776–801 (2021). https://doi.org/10.1007/s10766-021-00714-1

[7] Yuan, S., B. Skinner, S. Huang, D. Liu. 2013. A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms. European Journal of Operational Research 228(1) 72–82.

[8] Singh, A., A. S. Baghel. 2009. A new grouping genetic algorithm approach to the multiple traveling salesperson problem. Soft Computing 13 95–101.

[9] Lu, L.-C., T.-W. Yue. 2019. Mission-oriented ant-team aco for min–max mtsp. Applied Soft Computing 76 436–444.

[10] Liu, W., S. Li, F. Zhao, A. Zheng. 2009. An ant colony optimization algorithm for the multiple traveling salesmen problem. 2009 4th IEEE Conference on Industrial Electronics and Applications. IEEE, 1533–1537.

[11] Venkatesh, P., A. Singh. 2015. Two metaheuristic approaches for the multiple traveling salesperson problem. Applied Soft Computing 26 74–89.