FEDERAL STATE AUTONOMOUS EDUCATIONAL

INSTITUTION OF HIGHER EDUCATION

ITMO UNIVERSITY

Report

Lab No. 1

“One-criteria: exact and approximate methods”

Performed by

Daan Rodríguez

Academic group

J4132c

St. Petersburg

2024

1. Problem Selection

The selected problem is the Traveling Salesman Problem (TSP), a fundamental and well-known problem in combinatorial optimization. The problem can be stated as follows: Given a set of cities and the distances between each pair of cities, the goal is to find the shortest possible route that visits each city exactly once and returns to the origin city.

The TSP is classified as an NP-hard problem, which means that there is no known algorithm that can solve all instances of this problem efficiently (i.e., in polynomial time). As the number of cities increases, the number of possible routes grows factorially, making it impractical to solve by brute force for large instances. This inherent complexity makes the TSP an ideal candidate for testing various optimization techniques.

2. Algorithms Selected

To address the TSP, two types of algorithms were chosen:

- **Exact Algorithm**: Branch and Bound
- **Approximate Algorithm**: Genetic Algorithm

3. Generating Initial Data

To evaluate the performance of the selected algorithms, 15 instances of distance matrices were generated. Each matrix represents the distances between 10 cities. The distances were generated using random integers between 1 and 100 to simulate real-world variability. The diagonal entries of the matrices were set to 0, representing that the distance from a city to itself is zero.

4. Exact Algorithm: Branch and Bound

Explanation

The Branch and Bound algorithm is a methodical approach for solving combinatorial optimization problems. It works by dividing the problem into smaller subproblems (branching) and using a bounding function to prune subproblems that cannot yield a better solution than the current best-known solution.

Steps:

- **Initialization**: Start with an initial route and calculate its total cost.

- **Branching**: Generate all possible extensions of the current partial route by adding one more city at a time.
- **Bounding**: For each extension, compute a lower bound on the total route cost. This lower bound represents the best possible cost for completing the route from the current partial route.
- **Pruning**: If the lower bound of a subproblem exceeds the cost of the best-known complete route, discard the subproblem.
- **Recursion**: Recursively apply the branching and bounding process to the remaining routes until all possible routes have been explored or pruned.

This method guarantees finding the optimal solution but can be computationally intensive, especially for larger instances.

5. Approximate Algorithm: Genetic Algorithm

Explanation

The Genetic Algorithm (GA) is a heuristic search technique inspired by the principles of natural evolution and genetics. It is used to find approximate solutions to optimization and search problems through techniques such as selection, crossover, and mutation.

Steps:

- **Population Initialization**: Create an initial population of random routes (individuals).
- **Fitness Evaluation**: Evaluate each route's total travel cost (fitness). Routes with lower costs are considered more "fit."
- **Selection**: Select pairs of routes to be parents based on their fitness, with fitter routes having a higher chance of being selected.
- **Crossover**: Combine parts of two parent routes to create new routes (offspring). This mimics the process of genetic recombination.
- **Mutation**: Introduce small random changes to some routes to maintain genetic diversity and explore new parts of the solution space.
- **Replacement**: Form a new generation by replacing some or all of the old population with new routes.

The process is repeated for many generations, gradually improving the population's overall quality. While GAs do not guarantee finding the optimal solution, they often produce good solutions within a reasonable timeframe.

6. Evaluation of the Genetic Algorithm with Different Stopping Conditions

To assess the performance of the Genetic Algorithm, three different stopping conditions were tested:

Fixed Generations

The algorithm runs for a predefined number of generations. This condition ensures a fixed execution time but may not always yield the best solution.

Convergence Threshold

The algorithm stops if the improvement in solution quality falls below a certain threshold for a set number of consecutive generations. This method aims to halt the algorithm when it ceases to make significant progress.

Time Limit

The algorithm runs for a predefined amount of time. This condition is useful for real-time applications where a solution is needed within a specific timeframe. However, the quality of the solution can vary depending on the time limit set.

7. Results

Branch and Bound

- **Best Cost**: 116
- **Best Route**: [0, 5, 1, 4, 7, 9, 3, 6, 2, 8]
- **Execution time:** 1.11 seconds
- **Memory used:** 0.0 MB

The Branch and Bound algorithm found the optimal solution with a cost of 116. This result demonstrates the algorithm's ability to guarantee the best solution, albeit at the cost of significant computational effort.

Genetic Algorithm

Evaluation of Genetic Algorithm with Different Stopping Conditions

1. **Fixed Generations**
    - **Best Cost**: 224
    - **Best Route**: [4, 8, 0, 7, 9, 5, 6, 1, 3, 2]
    - **Execution Time**: 0.75 seconds

The fixed generations condition produced a better solution compared to other stopping conditions, demonstrating the algorithm's potential to improve given more generations.

2. **Convergence Threshold**
    - **Best Cost**: 384
    - **Best Route**: [7, 1, 3, 0, 6, 5, 9, 8, 4, 2]
    - **Generations Until Convergence**: 9999

The convergence threshold condition did not yield an optimal solution within the allowed generations, indicating that the chosen threshold might have been too high, or the algorithm got stuck in a local optimum.

3. **Time Limit**
    - **Best Cost**: 323
    - **Best Route**: [1, 5, 7, 2, 9, 3, 0, 6, 8, 4]
    - **Execution Time**: 5.00 seconds

The time limit condition produced a solution quickly but with a higher cost than the fixed generations condition. This result underscores the trade-off between execution time and solution quality.

8. Conclusion

The results obtained show that the exact algorithm Branch and Bound was able to find the optimal solution for the Traveling Salesman Problem in a specific test instance, with a route cost of 116. This demonstrates the algorithm's ability to guarantee the best possible solution, albeit at a significant computational effort, with an execution time of 1.11 seconds.

On the other hand, the Genetic Algorithm produced approximate solutions under different stopping conditions. The fixed generations condition resulted in a route cost of 224 with an execution time of 0.75 seconds, indicating that in this case, allowing for more generations could further improve the solution. The convergence threshold condition did not yield an optimal solution within the allowed generations, suggesting that the chosen threshold might have been too high or that the algorithm got stuck in a local optimum. The time limit condition produced a quick solution but with a higher route cost, highlighting the trade-off between execution time and solution quality.

In conclusion, Branch and Bound is suitable for smaller instances where finding the optimal solution is critical, while the Genetic Algorithm is more suitable for larger instances where a good solution is needed quickly. The choice of stopping conditions for the Genetic Algorithm significantly influences its performance and the quality of the solution, underscoring the need for careful selection based on the specific requirements of the problem.

9. Code.

https://colab.research.google.com/drive/12GFLKuhxTXZa6JAQJE4heEei_jga6zHg?usp=sharing