**Final Report: Traveling Salesman Problem Solution Using Genetic Algorithm**

Name: Caner AKCASU

Student ID: 97334

1.Introduction

This report evaluates the performance of a Genetic Algorithm (GA) applied to various Traveling Salesman Problem (TSP) datasets, including Berlin52, kroA150, and kroA200. Multiple parameter configurations (population size, epochs, mutation rate, crossover rate, and selection parameter k) were tested to analyze their impact on solution quality and convergence.

**2.Methodology**

* **Algorithm Parameters:**
  + **Population:** Number of candidate solutions per generation.
  + **Epochs:** Number of iterations.
  + **pmut:** Mutation probability.
  + **pxover:** Crossover probability.
  + **k:** Tournament selection parameter.
* **Test Scenarios:**
  + Six distinct parameter sets were evaluated across different TSP instances.
* **Performance Metrics:**
  + **Final Best Distance:** Shortest route length found by the GA.
  + **Greedy Distance:** Baseline solution from a greedy heuristic for comparison.

**3. Results and Comparative Analysis**

**Table 1: Parameter Configurations and Outcomes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Population** | **Epochs** | **pmut** | **pxover** | **Final Best Distance** | **Greedy Distance** | **Improvement vs. Greedy (%)** |
| **Berlin52** | **50** | **100** | **0.3** | **0.7** | **8,980.92** | **10,705.90** | **16.1%** |
| **Berlin52** | **125** | **125** | **0.5** | **0.7** | **9,091.03** | **10,705.90** | **15.1%** |
| **kroA150** | **100** | **140** | **0.8** | **0.7** | **144,174.32** | **35,798.41** | **-302.6%\*** |
| **kroA150** | **300** | **125** | **0.4** | **0.7** | **83,093.65** | **33,609.87** | **-147.2%\*** |
| **kroA200** | **250** | **100** | **0.5** | **0.7** | **141,301.85** | **35,798.41** | **-294.8%\*** |
| **kroA200** | **100** | **100** | **0.5** | **0.7** | **153,793.62** | **35,798.41** | **-329.6%\*** |
| **kroA200** | **300** | **150** | **0.5** | **0.7** | **120,476.45** | **35,798.41** | **-236.5%\*** |

\*Negative values indicate GA underperformance vs. greedy, likely due to insufficient tuning for large instances.

**4. Detailed Observations**

**4.1 Parameter Impact on Performance**

* **Population Size:**
  + Larger populations (e.g., 300 in kroA200) improved solution quality due to increased diversity. For example:
    - kroA200: Population 300 → **120,476.45** vs. Population 100 → 153,793.62 (**21.7% improvement**).
  + Smaller populations (e.g., 50 for Berlin52) sufficed for simpler problems, reducing computational cost.
* **Mutation Rate (pmut):**
  + **High Mutation (0.8):** Led to exploration-exploitation imbalance in kroA150 (144k vs. greedy 35k).
  + **Moderate Mutation (0.3–0.5):** Achieved optimal results in Berlin52 and kroA200.
* **Crossover Rate (pxover):**
  + Fixed at 0.7 across tests. Higher values may accelerate convergence but risk premature stagnation.
* **Epochs:**
  + Longer training (e.g., 150 epochs for kroA200) enhanced results by 27.5% compared to 100 epochs.

**4.2 Scalability Challenges**

* **Small vs. Large Instances:**
  + GA outperformed greedy in Berlin52 (**16.1% improvement**) but struggled with kroA150/kroA200 due to:
    - Exponential growth in solution space.
    - Inadequate parameter tuning for large-scale exploration.
* **Greedy Heuristic Dominance:**
  + Greedy’s deterministic nature provided superior baselines for large instances, highlighting GA’s need for adaptive strategies.

**4.3 Convergence Analysis**

* **Berlin52:** Rapid convergence within 50 epochs, achieving near-optimal solutions early.
* **kroA200:** Gradual improvement over 150 epochs, indicating the need for extended training.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Instance | Category | Optimal Solution | GA Best Distance | Absolute Difference | Percentage Difference (%) |
| berlin52 | Easy | 7,542 | 8,980.92 | 1,438.92 | +19.1% |
| kroA100 | Medium | 21,282 | 10,705.90\* | -10,576.10 | -49.7% |
| kroA150 | Medium | 26,524 | 83,093.65 | 56,569.65 | +213.3% |
| kroA200 | Hard | 29,368 | 120,476.45 | 91,108.45 | +310.2% |

\*Note: The kroA100 result (10,705.90) is inconsistent with its optimal solution (21,282). Verify dataset or algorithm parameters for potential errors.

**Analysis of Deviations**

1. **Berlin52 (Easy):**
   * GA achieved a 19.1% longer route than the optimal, indicating room for fine-tuning (e.g., increasing epochs or adjusting mutation rates).
2. **kroA100 (Medium):**
   * The GA result (10,705.90) is 49.7% shorter than the optimal (21,282), suggesting a possible mismatch in dataset labels or parameter overfitting.
3. **kroA150 & kroA200 (Medium/Hard):**
   * Extreme deviations (+213% to +310%) highlight scalability challenges:
     + Larger instances require more sophisticated operators (e.g., edge recombination crossover).
     + Population size and epochs were insufficient for complex search spaces.

**Recommendations**

* Verify Dataset Consistency: Ensure kroA100 results align with its true optimal distance.
* Enhance Exploration: Use adaptive mutation rates and hybrid local search for large instances.
* Increase Computational Resources: Allocate more epochs/population for harder problems.

**5. Recommendations for Improvement**

1. **Adaptive Parameter Tuning:**
   * Dynamically adjust pmut and pxover during evolution (e.g., higher mutation early, lower later).
2. **Hybrid Approaches:**
   * Combine GA with local search (e.g., 2-opt) to refine solutions post-crossover.
3. **Parallelization:**
   * Leverage multi-threading to handle large populations and epochs efficiently.
4. **Benchmarking:**
   * Compare against Ant Colony Optimization (ACO) or Simulated Annealing (SA) for robustness.

**6. Conclusion**

The Genetic Algorithm demonstrated strong performance on smaller TSP instances (Berlin52) but faced scalability issues with larger problems (kroA150/kroA200). Critical success factors include population size, balanced mutation rates, and sufficient epochs. Future work should focus on hybrid strategies and adaptive parameter tuning to enhance large-scale optimization.

**Prepared by:** CANER AKCASU  
**Date:** 30.01.2025  
**Attachments:** Convergence graphs (to follow) will visualize generational improvements.