Research Paper Logistics

# Testing The Quality of The Proposed Solution to The TSP Problem using Genetic Algorithms

**Ing. Jakub Brodský** CTU Faculty of Transportation Sciences, Department of Logistics and Management of Transport, Konviktská 20, 110 00 Praha 1, Czech Republic

**Ing. Denisa Mocková, Ph.D.** CTU Faculty of Transportation Sciences, Department of Logistics and Management of Transport, Konviktská 20, 110 00 Praha 1, Czech Republic

**ABSTRACT**

The article concerns the testing of genetic operators of agenetic algorithm for calculation of a transportation task in the travelling salesman problem (TSP). Software was developed for testing the solution of the given problem, using a genetic algorithm with proposed variant operators of selection, crossover and mutation, along with other parameters of the problem.

**KEYWORDS** TSP problem, genetic algorithm, testing, software solution

## INTRODUCTION

The travelling salesman problem (TSP) is an NP-hard problem. In these problems the set of all the admissible solutions is so extensive that it is impossible, using the exact methods, to find an optimal solution within an acceptable time horizon. For this reason methods, known as meta-heuristics, have been developed that operate on the principles of combinatorial optimization and seek solutions in the limited area of all admissible solutions.The most frequently used and examinedmeta-heuristicsare genetic algorithms (GA) [1],due to their simplicityand easy implementation to the own test software. The quality of the proposed solution, which is described in [2],will be tested on a smaller (13 vertices) and larger (72 vertices) data sample.

### Software for solving TSP problem using a genetic algorithm

In order to be able to test individual parameters of the genetic operators (GO) it was necessary to develop an application with a graphic interface for the Windows environment. The application is programmed in the C# language and enables the user to fully use the above-proposed solution to the calculation for the travelling salesman problem by way of a genetic algorithm. The application can calculate also the exact solution (permutation solution); this option is though limited to twenty vertices.

### Testing the quality of the proposed solution

### Input data for testing operators

In order to verify data with an exact calculation and at the same time to test that the algorithm can calculate extensive samples within a reasonable time, we used two sets of data that were transferred from the GPS coordinates to the Cartesian coordinate system with the aid of Mercator projection. A smaller set of data represents regional towns of the Czech Republic. The smaller set of data was dealt with in both exact calculation and GA. The number of vertices in this set is 13. A larger set of data contains district towns of the Czech Republic. The larger set of data was processed by the GA and the exact calculation was used for comparison, using an IP solver tool [3]. The set comprises 72 vertices.

### Exact solution of smaller set of data

With the help of the programmed application the exact solution of the problem was calculated for the given set of data. The calculation, using this approach, took 3 hours, 24 minutes and 46 seconds, with 479,001,600 routes having been calculated. The shortest distance found was 851 kilometers.

The subsequent tests of various combinations of operations and parameters will thus be compared with this result. The aim of the testing is to determine at what combination the algorithm is able to attain, as quickly as possible, the optimal solution.

### Testing of operators on a smaller set of data – GA

**As stated in [2], the following options of GO were tested:**

- 2 variants of the selection (tournament binary, tournament tertiary),
- 3 crossover variants (one-point – variant 1, one-point – variant 2, two-point),
- 2 mutation variants (swapping one gene in the case of one individual, swapping two genes),
- 2 variants of the number of individuals within a generation (10, 20),
- 2 variants of the number of iterations (200, 10 000),
- crossover probability – 0.7, 0.8, 0.85, 0.9,
- mutation probability – 0.001, 0.01, 0.1, 0.2,
- elitism – 0, 1, 2.

During testing with a smaller data sample we could compare the result of each test with the exact solution. By comparing the results we reached the following conclusions:

- For a data sample of this size (13 vertices) the genetic algorithm, with the number of 10 000 iterations, is able to find an optimal solution to the problem with any combination of other operators and parameters. While at the same time the average calculation time stood at about 20 seconds, which is, against the time needed for the exact calculation (3 hours, 24 minutes and 46 seconds), an increase in speed by 61 400%, i.e. the calculation is 614 times faster.
- In the case of the calculation of 200 iterations the results

no longer attained every time the optimal solution. However, from these results it can be inferred which operators and parameters have a positive or negative impact on the calculation results. From the analysis of the results it follows that a result is affected most by the type of chosen crossover, the degree of elitism and the number of individuals in a generation. The percentage rate of success of finding an optimal solution with these operators and parameters is shown in table 1.

**Table - 1 Comparison of the results of testing operators with a smaller data sample for 200 iterations**

| Elitism | Crossover type | Number of individuals | Total number of tests | Of which optimal solutions | Success rate of finding optimal solution [%] |
|---|---|---|---|---|---|
| 0 | One-pointVariant 1 | 10 | 64 | 5 | 7.81 |
| | | 20 | 64 | 21 | 32.81 |
| | One-pointVariant 2 | 10 | 64 | 21 | 32.81 |
| | | 20 | 64 | 37 | 57.81 |
| | Two-point | 10 | 64 | 13 | 20.31 |
| | | 20 | 64 | 24 | 37.50 |
| 1 | One-pointVariant 1 | 10 | 64 | 21 | 32.81 |
| | | 20 | 64 | 39 | 60.94 |
| | One-pointVariant 2 | 10 | 64 | 39 | 60.94 |
| | | 20 | 64 | 55 | 85.94 |
| | Two-point | 10 | 64 | 42 | 65.63 |
| | | 20 | 64 | 48 | 75.00 |
| 2 | One-pointVariant 1 | 10 | 64 | 16 | 25.00 |
| | | 20 | 64 | 37 | 57.81 |
| | One-pointVariant 2 | 10 | 64 | 38 | 59.38 |
| | | 20 | 64 | 58 | 90.63 |
| | Two-point | 10 | 64 | 49 | 76.56 |
| | | 20 | 64 | 52 | 81.25 |

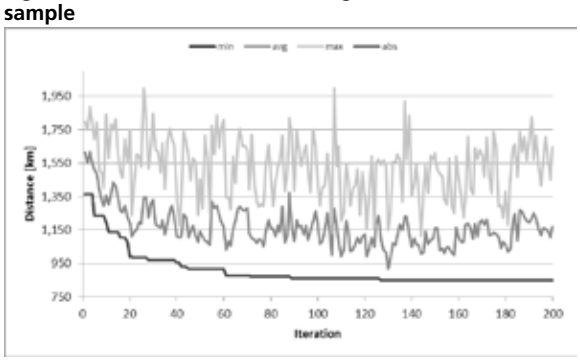
**Source: author**

**Best solution found for the smaller data set**
**The best result was achieved by the genetic algorithm by using the following parameters:**

- Selection type – tournament – binary
- Crossover type – two-point
- Mutation type – swapping one gene
- Number of individuals – 10
- Number of iterations – 200
- Crossover probability – 0.8
- Mutation probability – 0.2
- Elitism – 1st degree

The algorithm reached the lowest value found (851 km) at the 127th iteration. The time needed for the GA calculation, using the settings described above, was 0.205 seconds.

**Figure 1: Iterations for best settings found with a smaller sample**




**Source: author**

**Testing of operators in the larger set of data – GA**
For the purpose of making table 2and some captions of the following tablesclearer, the names of the types of selection, crossover and mutation are replaced by indices.

**Selection type**

- 1 – tournament binary
- 2 – tournament tertiary

**Crossover type**

- 1 – one-point – variant 1
- 2 – one-point – variant 2
- 3 – two-point

**Mutation type**

- 1 – swapping one gene
- 2 – swapping two genes

**The parameters of the settings are as follows:**

- Number of individuals within a generation 20, 40, 60
- Number of iterations 10,000
- Crossover probabilities– 0.7, 0.8, 0.85, 0.9
- Mutation probabilities – 0.001, 0.01, 0.1, 0.2
- Elitism – 0, 1

A summary of the testing results with a larger data sample is shown in table 2. Only the best solution is given for the columns for the number of individuals, crossover probability, mutationprobability, due to the extensive nature of the table. The table 2 shows the best result found in bold.

**Table - 2 Comparison of the results of testing operators with a larger data sample**

| Elitism | Selection type | Crossover type | Mutation type | Number of iterations | Number of individuals | Crossover probability | Mutation probability | Best result [km] | Iteration at which best result was achieved | Time [m:s] |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 10,000 | 60 | 0.7 | 0.01 | 2,748 | 7,578 | 03:45 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.01 | 2,775 | 6,579 | 03:43 |
| | | 2 | 1 | 10,000 | 60 | 0.7 | 0.001 | 1,856 | 7,917 | 03:15 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.01 | 1,949 | 6,559 | 03:00 |
| | | 3 | 1 | 10,000 | 20 | 0.7 | 0.01 | 4,033 | 3,533 | 01:06 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.001 | 4,023 | 8,835 | 02:59 |
| | 2 | 1 | 1 | 10,000 | 60 | 0.8 | 0.001 | 2,014 | 9,318 | 04:15 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.1 | 1,994 | 8,651 | 04:08 |
| | | 2 | 1 | 10,000 | 60 | 0.85 | 0.001 | 1,734 | 8,353 | 03:39 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.1 | 1,713 | 9,671 | 03:35 |
| | | 3 | 1 | 10,000 | 60 | 0.7 | 0.001 | 1,771 | 5,472 | 03:42 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.001 | 1,854 | 8,482 | 03:45 |
| 1 | 1 | 1 | 1 | 10,000 | 60 | 0.9 | 0.01 | 1,851 | 9,873 | 04:12 |
| | | | 2 | 10,000 | 60 | 0.8 | 0.1 | 1,940 | 9,923 | 04:06 |
| | | 2 | 1 | 10,000 | 60 | 0,7 | 0.001 | 1,756 | 9,648 | 03:21 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.01 | 1,722 | 8,141 | 03:22 |
| | | 3 | 1 | 10,000 | 40 | 0.85 | 0.1 | 1,873 | 9,223 | 02:17 |
| | | | 2 | 10,000 | 20 | 0.7 | 0.01 | 1,868 | 9,303 | 01:14 |
| | 2 | 1 | 1 | 10,000 | 60 | 0.8 | 0.01 | 1,810 | 8,260 | 04:49 |
| | | | 2 | 10,000 | 60 | 0.8 | 0.001 | 1,803 | 7,448 | 04:43 |
| | | 2 | 1 | 10,000 | 60 | 0.8 | 0.1 | 1,712 | 9,501 | 04:02 |
| | | | 2 | 10,000 | 60 | 0.85 | 0.1 | 1,721 | 9,423 | 04:04 |
| | | 3 | 1 | 10,000 | 40 | 0.8 | 0.01 | 1,711 | 8,719 | 03:08 |
| | | | 2 | 10,000 | 60 | 0.7 | 0.2 | 1,722 | 8,380 | 03:56 |

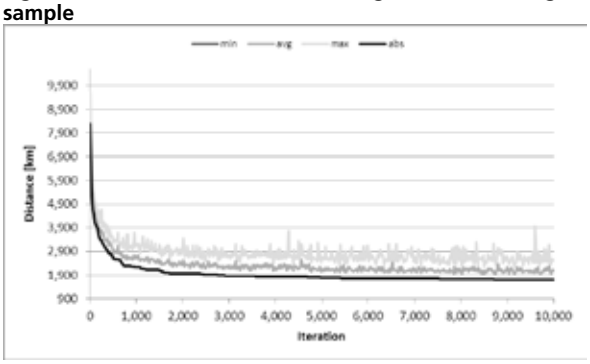
**Source: author**

**Best solution found for the larger data set**
**The best result was achieved by a genetic algorithm using the following parameters:**

- Selection type – tournament tertiary
- Crossover type – two-point
- Mutation – swapping one gene
- Number of individuals – 40
- Number of iterations – 10,000
- Crossover probability – 0.8
- Mutation probability – 0.01
- Elitism – 1st degree

The lowest value (1,711 km) was found by the genetic algorithm at the 8,719th iteration. The time needed for completing the calculation by GA was 3 minutes and 8.265 seconds.

**Figure 2:Iterations for the best settings found on a larger sample**




**Source: author**

**CONCLUSIONS**

Testing of operators was carried out on two data samples – small (13 vertices) and large (72 vertices).By comparing the test results for the small sample we reached the conclusion that the algorithm works and can attain an optimal solution to the problem in a fraction of the time needed for calculating the problem using an exact approach.This though applies only in the case where we choose a sufficiently high value of the number of iterations (during testing this value was set at 10,000 iterations). If a small data sample was tested with a low value of the number of iterations (during testing it was 200 iterations), the algorithm then failed in every combination to find an optimal solution. From these data we however can deduce which operators are more beneficial for obtaining an optimal solution. The most important in this case is the degree of elitism. If it is set at the degree 1 and more, the algorithm immediately records a better result over a minor number of iterations, because thanks to elitism the individuals do not degenerate, i.e. the best individual within a generation is always passed on into the next generation.

When tested on a large data sample, the exact calculation was performed with the aid of Concorde [4]. This software uses the linear programming methods, more specifically integer programming – abbreviated to IP. With the help of this method the exact solution was found in 1.22 seconds and the resulting value is 1,703 km. The best result found by our genetic algorithm was 1,711, i.e. a deviation of 0.47%.

In conclusion it can be said that the algorithm is prepared for the TSP calculation. It can be further enhanced by modifying operators of one-point crossover, which could acquire another degree of freedom by way of an algorithm randomly determining which side of the parents, from the crossover point, is to be copied into the offspring and from which side the offspring is to be supplemented to form a whole individual.

**Acknowledgements**
This research was supported "Application of Exact and Metaheuristic Methods to VRP in Transportation", (FIS:161 - 1611404F000).

**REFERENCES**

[1]Holland JH. (1975), "Adaptations in Natural and Artificial Systems." Ann Arbor, MI: University of Michigan Press | | [2]Brodský J. (2013), "Testování operátorů genetických algoritmů", master thesis. Praha: ČVUT v Praze | | [3]http://lpsolve.sourceforge.net/5.5/ | | [4]http://www.math.uwaterloo.ca/tsp/concorde/ |