Analysis of Genetic Algorithm based solutions for the Traveling Salesman Problem

Kacper Smyczyk, 236732

June, 2020

1. Introduction

The Travelling Salesman Problem (TSP) is about finding the most optimal tour of given number of cities with the criteria of visiting each city only once and returning to the starting city at the end. The solution is said to be the most optimal if the traveled distance is minimized. While the exact origins of the traveling salesman problem are not specified, a handbook for traveling salesmen from 1832 mentions it without giving any mathematical treatment [1]. The problem itself is an NP-hard problem in combinatorial optimization thus no optimal polynomial time solutions are known [2].
The TSP has several applications, such as planning, logistics, and the manufacture of microchips. Slightly modified, it appears as a sub-problem in many areas, such as DNA sequencing. In these applications, the concepts of city and distance change accordingly to fit into the problem.

2. TSP as a graph problem

The problem can be modeled as an undirected weighted graph in which cities are vertices, paths are edges and the distance between cities is represented as the weight of an edge. It is a minimization problem starting and finishing at the same selected vertex while visiting every other vertex exactly once. Often the graph is complete if every city is connected to another. The Miller–Tucker–Zemlin formula for the problem is given [3]:

$$Min \sum_{i=1}^{n} \sum_{i \neq j}^{n} \sum_{j=1} c_{ij} x_{ij}$$

$$x_{ij} \in \{0,1\}$$

$$\sum_{i=1,i\neq j}^{n} x_{ij} = 1 \qquad j = 1,\dots,n$$

$$\sum_{j=1,i\neq j}^{n} x_{ij} = 1 \qquad i = 1,\dots,n$$

$$u_i - u_j + nx_{ij} \leq n-1, \qquad 2 \leq i \neq j \leq n,$$

$$0 \leq u_i \leq n-1 \qquad 2 \leq i \leq n$$

where c_{ij} is the cost of traveling from city i to city j, n is the number of cities, $x_{ij} = \begin{cases} 1 & the\ path\ goes\ from\ city\ i\ to\ city\ j \\ 0 & otherwise \end{cases}$.

The Traveling Salesman Problem can be formulated as not symmetric which means that paths may not exist from one city to another or the distances might be different resulting in a directed graph.

3. Genetic Algorithms

Genetic Algorithms are adaptive search algorithms based on the evolutionary ideas of biology, namely natural selection. This metaheuristic was first introduced by John Holland in the 1960s and has been since widely studied and applied in many fields[4].

Genetic Algorithm (GA) typically starts with a base population of candidate solutions (called chromosomes or individuals) and artificially evolves these solutions during runtime. Solutions are encoded to translate them into a string of genes that make up the chromosome just as in genetics. Different problems may require encoding other than the standard binary encoding and problem may have more than one possible encoding. After establishing the base population (commonly at random) the GA then evaluates the fitness of every individual and selects which of them will move to the next generation. Selected Individuals undergo additional operations such as crossover and mutation. The selection process yields individuals chosen for the mating (reproduction) process based on their

fitness. Generally the selection process is meant to reduce the search area while the additional operations of mutation, crossover and reproduction are extending it. A good GA must balance these two processes to achieve the global optimum (individual with low fitness may still carry important genetical information)[5]. Reproduction is about preserving the best solutions to the next generation as-is. Crossover takes two randomly chosen solutions now called "parents" and produces offsprings containing some combination of genes from the parents. Mutation, which changes some genes randomly, aims to simulate the concept of random evolutionary genetic mutations. Such mutations can of course produce superior or inferior offsprings but the inferior ones are less likely to survive until the next generation.

4. Methods

Two solutions were chosen: Entropy-based GA for TSP [6] and GA with mixed region search for Asymmetric TSP [7].

4.1 Entropy-based GA

Information entropy is introduced to diversify populations in order to improve the search of global optimum. TSP is a combinatorial optimization problem, thus while applying GA one may encounter a trap of falling into a local optimal solution. This happens when populations become less and less diverse over time.

4.1.1 Representation

This approach takes the standard representation in which each chromosome represents a path consisting of visited cities in order [8]. The starting population is initialized randomly.

4.1.2 Fitness

Solutions are evaluated in this approach by given function:

$$eval(t_k) = \frac{1}{\sum_{i=1}^{n} d(c_i, c_{i+1}) + d(c_n, c_1)}$$

where t_k is a chromosome and $d(c_i, c_{i+1})$ is the distance between cities c_i and c_{i+1}.

4.1.3 Genetic Operators

Selection is controlled under the roulette wheel selection mechanism[8]. EBGA also uses cycle crossover (CX)[9] and swap mutation[8].

4.1.4 Algorithm

In the EBGA, diversity of chromosome population is measured at every generation in order to improve populations lacking diversity. Diversity of a population is calculated by comparing chromosomes genes at given position (ith visited city).

The diversity H_i of the ith position (i = 1,2,...,n) is defined as

$$H_i = -\sum_{c \epsilon C} pr_{ic} \ln pr_{ic}$$

where

$pr_{ic} = \frac{na_{ic}}{population\_size}$

C : set of cities

na_{ic} : number of appearances of city c at the ith potision

Then to improve the diversity, m chromosomes are randomly chosen from the population and their genes change places on the positions with diversity below the threshold (genes change places in each chromosome itself, not swap with genes from other chromosomes).

4.1.5 Parameters

Experiments concluded for the EBGA had population size set to 20, maximum generation limit set to 20000, crossover probability set to 0.8, mutation probability set to 0.2, a equal to 3 and the number of chromosomes to diversify (m) was being randomly chosen between $\frac{population\_size}{a}$ and population_size-1.

4.2 GA with mixed region search

This Genetic Algorithm was implemented for the asymmetric TSP. This GA extends search space by generating and including infeasible solutions in the population. The algorithm generates good quality infeasible solutions with subtours through genetic operators, so that the solutions can be utilized as seeds for the patching algorithm.

4.2.1 Representation

This approach uses the adjacency representation because in the standard path representation the subtours cannot be easily distinguished [10]. In the adjacency list the ith entry is the information about where the subroutine goes from the ith city.

4.2.2 Fitness

Solutions are evaluated by the sum of arc costs in the tours represented by the chromosome. Fitness is not lowered because of the need to recover feasibility since that would defeat the purpose of expanding the search space.

4.2.3 Genetic Operators

Two crossover operations are used: partially matched crossover (PMX)[10] and modified tie break crossover (TBX). Modified TBX takes a pair of parents and randomly divides the chromosomes into three parts. Then the middle sub-strings are exchanged between them to create offsprings. In these offsprings random numbers between 0 and 1 are then added to the missing/repeated genes. Now these genes are ordered and can be replaced by the missing/repeated values accordingly. Three way swapping is used as a mutation. Mutations only occur when there are no changes in fitness for the best chromosomes in two consecutive generations.
The algorithm also uses modified elitism to preserve good solutions while maintaining diversity in order to not fall into the local optimum.

4.2.4 Repair Algorithm

The GA presented expands the search space thus it must repair the

infeasible solutions from time to time. This approach used the Karp's patching algorithm for this purpose[11]. It is also stated that randomly chosen repair cycle of generations produced the best results.

4.2.5 Parameters

Experiments concluded for this GA had population sizes varying between 17 and 443. Auction algorithm by Bertsekas [12] was used to solve the assignment relaxation problem. The number of chromosomes was set to 200, maximum generation limit set to 1000, repair cycle set to a random number from [1, 5] generations, three elite groups consisting of 60%, 30% and 10% of the best 40% and the rest 40% and 20% in combined pool. To show effect of the expanded search space the program was also run with repair cycle set to [1, 1] which results in the feasibility being maintained throughout the GA.

5. Results

Both approaches proved to be useful by finding more optimal solutions than the compared solutions. These two solutions shed insight on what is important while tackling different TSA problems with Genetic Algorithms and how to interpret test results.

5.1 Entropy-based GA

EBGA provided better results than the traditional GA and had faster convergence [Figure 1][6]. This solution showed that escaping the local solution is a crucial problem that needs to be tackled while developing GA based solutions for optimization problems. Solving this problem helps get good results in faster time.

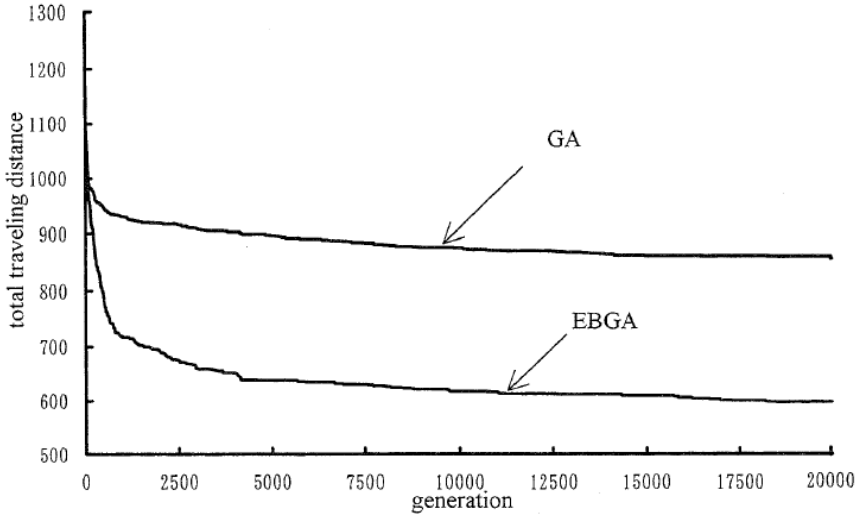

Figure 1: EBGA vs. GA [6]

5.2 GA with a mixed region search

This implementation was tested on 27 tests taken from TSPLib [13] alongside the Branch and Bound algorithm [28] taken from NETLIB [14]. The GA proved to be useful by getting solutions of the same and in some cases even better quality than the branch and bound algorithm. For four of the tests the GA was strongly favored over b&b resulting in much faster and better solutions. The GA solutions were also less dependent on the initial random values. Expanding the search space also proved to be fruitful outperforming the all feasible version 25 times out of 69 comparisons and yielding worse solution only 5 times. Another conclusion is to take notice of the size of given problem and change parameters accordingly. Another important factor is to analyze whether the best solutions are coming from sooner or later generations as it can be a useful hint towards parameter manipulation.

5.3 Parameters manipulation

One of the main difficulties when designing and testing a Genetic Algorithm lays in the choice of parameters. The parameters are subject to the problem being tackled but a useful technique described by De Jong is to start with a high crossover probability, low mutation probability and a

population of a moderate size [15]. The importance of the starting population should also be monitored whether it has high impact on the results.

6. Conclusion

Both of the presented solutions to the problems from the TSP family resulted in useful and correct solutions. Each of them show techniques on how to design a Genetic Algorithm, whether it be the choice of parameters or different genetic operators. These solutions present the issues one can encounter while solving a problem with a Genetic Algorithm. There are many factors and methods to be considered based solely on the nature of the problem itself. These two solutions also highlight the need to adapt based on tested results. The GA with a mixed region search shows that the implemented Genetic Algorithm approach may not be the best for every problem there is while getting better efficient solutions for some.

References

[1] "Der Handlungsreisende – wie er sein soll und was er zu tun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiß zu sein – von einem alten Commis-Voyageur" (The travelling salesman — how he must be and what he should do in order to get commissions and be sure of the happy success in his business — by an old *commis-voyageur*) (June, 2020)

[2] C.H. Papadimitriou, Euclidean traveling salesman problem is NP-complete, Theoretical Computer Science 4 (1978) 237–244

[3] Miller, C.E., Tucker, A.W., Zemlin, R.A.: integer programming formulation of traveling salesman problems. J. ACM 7(4), 326-329 (1960)

[4] Goldberg, D. E.: Genetic algorithms and their applications. WNT, Warsaw, 1995.

[5] D. Beasley, D. Bull, and R. Martin, An Overview of genetic algorithms: Part 1, Fundamentals, University Computing, vol. 2, pp. 58-69, 1993

[6] Y. Tsujimura, M. Gen, Entropy-based genetic algorithm for solving TSP, 1998

[7] Choi, In Chan, Kim, Seong In, Kim, Hak Soo, A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem. In: Computers and Operations Research. 2003 ; Vol. 30, No. 5. pp. 773-786.

[8] Gen, 31. and R. Cheng: Genetic Algorithms and Engineering Design. John ST-iely & Sons, 1997.

[9] Michalewicz. Z.: Genetic A1gorithms + Data Structures = Evolution Programs, 2nd eds., Springer-Verlag, 1994

[10] Poon P, Carter J. Genetic algorithm crossover operations for ordering applications. Computers and Operations Research 1995;22:135–47.

[11] Karp R. A patching algorithm for the nonsymmetric traveling-salesman problem. SIAM Journal Computing 1979;8:561–73.

[12] Bertsekas D. Linear network optimization: algorithms and codes. Cambridge: The MIT Press, 1991.

[13] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/ (June, 2020)

[14] NETLIB, http://www.netlib.org/toms/750 (June, 2020)

[15] K. De Jong, W. M. Spears, “Using Genetic Algorithms to Solve NP Complete Problems” Proceedings of the Third International Conference on Genetic Algorithm, Morgan Kaufman, Los Altos, CA, 124 – 132, 1989