COS710: Assignment 3

Myron Guanhao Ouyang
Department of Computer Science
University of Pretoria
South Africa
u16008368@tuks.co.za

***Abstract*—Genetic algorithms are a form of optimization algorithms. They are based off of the process of natural selection. There are two approaches to implementing a genetic algorithm; direct and indirect, this paper will empirically compare both approaches by evaluating their performance on several Travelling Salesman Problems.**

I. Introduction

Genetic algorithms are a form of optimization algorithms based off of Darwin's theory of evolution, namely natural selection and survival of the fittest. There are two primary representations of a genetic algorithm, namely; direct and indirect.

The primary focus of this paper is to perform a comparative study between an indirect implementation of the a genetic algorithm (IGA) and direct implementation of a genetic algorithm (DGA).

This will be accomplished by implementing both algorithms and solving several Travelling Salesman Problems (TSP). The Traveling Salesman is a common optimization problem in which the 'salesman' needs to visit all the cities once in a list of cities and return to the starting city, finding the shortest route (also known as the tour) between all cities.

The remainder of the paper follows the following format; Section II will be a high level discussion on the approaches used for both implementations. Section III will be a more in depth view of the algorithms, Section IV will show the results of the implementation of both algorithms and finally Section V will conclude the results of this paper.

II. Background

The background section will give a brief summary of both approaches to implementing a genetic algorithm.

As stated earlier in the introduction, Genetic algorithms are optimization algorithms based off of natural selection [1]. This is accomplished by taking a population of individuals (representing possible solutions to a problem space) and running them through genetic operators such as; mutations and cross-over between the best individuals. Resulting in offspring that are more likely to have a better fitness value than their parents in the next generation.

The TSP used were taken from TSPLIB, in particular only the problems that made use of 2D Euclidean distance was used.

A. Direct Genetic Algorithm

In this representation of the algorithm, individuals (also known as chromosomes) are direct representations of solutions to the problem space. In the case of solving the Travelling Salesman Problem (TSP), each chromosome is made up of a list of cities (genes) and the order of these cities represent the order in which the salesman needs to visit each city.

B. Indirect Genetic Algorithm

In this representation of the algorithm, individuals are composed of a set of instructions to build a solution to the problem space. Each gene in the chromosome is an instruction that can be interpreted to either add, remove or mutate the list of cities the salesman needs to travel to.

III. Implementation

This section describes, in detail, the steps and thought processes followed to implement both approaches to solve TSP using a genetic algorithm.

The general implementation of a genetic algorithm is composed of a series of relatively simple steps:

Genetic Algorithm:

1) Generate an initial population
2) Evaluate population, get a fitness value for each individual
3) Apply genetic operators on population
   - Selection
   - Cross-over
   - Mutation
4) Take results from previous step and create a new population for the next generation
5) Repeat Step 2-4 until population has converged or a limit has been reached

In this implementation, each population is run for 10000 generations at 5 epochs.

A. Direct Genetic Algorithm

In the DGA approach, each gene in a chromosome represents a unique city and the order of the genes represent the order in which the cities need to be traversed.

The initial population is composed of a collection of randomly generated lists of cities (individual). Each list does not have duplicates and the order of each city is randomized. The

fitness for each individual is calculated by calculating the total distance accumulated if the cities are traversed in order.

The selection method used is a combination of Elitism [1] and Tournament selection [2]. The population is ordered by fitness value, with the lowest fitness value at the top. The top 10% of individuals are directly copied over to the next generation. The remainder of the new generation's population is populated by children of parents produced through Tournament selection. It was found, through experimentation, that this combination of selection operators produced better results than using either one of the operators on their own.

The cross over method implemented was a simple one-point cross over method. A slight modification was made to make the cross over method more dynamic, the point of cross over is a randomly chosen point that is at most half of the length of the chromosome. It was found that this produced slightly better results as having a set point would tend to replace sets of cities that had a good fitness value.

The mutation operator used is a simple uniform swap method. The probability for the chromosome to mutate remains a constant 20% throughout the generations. The number of genes that will be swapped is randomly selected from a ranch of 1-N.

B. Indirect Genetic Algorithm

In the IGA approach, each gene represents an instruction used to construct a list of cities to traverse. The length of each chromosome is randomly selected from a range of $N - 2N$ where N is the number of cities. The instructions are represented by a single letter. Refer to table I.

A	Add a random city to the list of cities from available list of cities.
B	Add next best city to the list of cities from available list of cities.
M	Mutate list of cities.
D	Remove random city from list of cities.
L	Remove the pair of cities that has the longest distance in the list of cities.

Table I
LIST OF INSTRUCTIONS

The "A" instruction will remove a random city from the list of available cities and add it to the list of cities to traverse. If the available cities list is depleted, the instruction is ignored. "B" will take the last city in the list of cities and remove a city from the list of available cities that has the shortest distance with the last city and add it to the list of cities. If either lists of cities are empty, this instruction is ignored. "M" will take the list of cities to traverse and perform the mutation operation described in the previous sub-section. "D" will remove a random city in the list of cities to traverse and add it back to the available list of cities. "L" will remove the pair of cities that have the longest distance in the list of cities to traverse and add them back to the list of available cities.

The fitness for each chromosome is calculated by decoding the instruction set and calculating the total distance of the resulting list of cities.

The initial population is a randomly generated string that is composed of the instructions from the instruction set. The genetic operators (selection, crossover and mutation) used in the IGA is the same as the ones implemented in the DGA approach.

IV. RESEARCH RESULTS

This section describes the results obtained from implementation of both approaches of the algorithm.

Table II and table III show the best, average and standard deviation of fitness values for each TSP. It is clear from the results that DGA is performing better than IGA.

File Name	Best	Mean	σ
berlin52	11380	12081.8	394.5166
kroA150	86836	95855	4969.8181
kroA100	53552	58215	2909.8855
kroB100	57486	59293.4	1452.9818
kroC100	57900	60678.6	2668.1490
kroD100	55535	58674	2965.0279
kroE100	52697	57901.8	3162.9194
pr107	171653	180403	9861.3956
ra99	2901	3011	95.2071
st70	1243	1340.6	76.0015

Table II
DIRECT DATA

File Name	Best	Mean	σ
berlin52	13888	15108.4	1525.9513
kroA150	147960	156095.6	6515.9995
kroA100	95586	99343.6	3408.8663
kroB100	87549	94439.6	6040.4435
kroC100	97665	101676.8	2375.3232
kroD100	87721	94057.8	5449.9712
kroE100	94433	98270.6	3557.0067
pr107	275992	294484.6	15086.4561
ra99	4626	4885.2	202.1913
st70	1921	2068	127.3153

Table III
INDIRECT DATA

A Mann-Whitney U test was conducted on each TSP's results and the results are shown in table IV. The H_0 would be rejected if the P-value was lower than P. It can be confirmed that there is a significant difference between the results obtained from the two approaches to solve the various TSP's.

P	p-value	Z-value	U-value
0.05	0.00793651	2.654759	25.00

Table IV
MANN-WHITNEY U TEST RESULTS

The results were expected as the indirect approach would result in faster convergence on a local minima, and it would take longer to reach a point where its chromosomes would result in a complete list of cities.

V. CONCLUSION

This paper conducted a comparative study between the indirect approach to genetic algorithms and the direct approach. From the results obtained from the several TSP test cases, it can be concluded that IGA does not perform nearly as well as

DGA. Both algorithm's results where compared making use of the Mann-Whitney U test and the results were statistically significant. It would seem that the abstraction of solutions provided by the indirect approach is a hindrance on the overall performance of the algorithm.

REFERENCES

[1] Haldurai Lingaraj. A study on genetic algorithm and its applications. *International Journal of Computer Sciences and Engineering*, 4:139–143, 10 2016.

[2] Pillay Raghavjee. A comparative study of genetic algorithms using a direct and indirect representation in solving the south african school timetabling problem. *ORSSA Proceedings*, pages 31–39, 2013.

APPENDIX