Type I Project Proposal for CSI 4105 DESIGN AND ANALYSIS OF ALGORITHMS II

Project Title: Investigate Different Algorithms for the Traveling Salesman Problem

Tao Feng - 7603033

Rodolfo Pichardo - 7133899

Presented to Professor Sylvia Boyd

February 14, 2017

1 Problem Statement

The traveling salesman problem (TSP) is a problem in graph theory, which requires a salesman to take the shortest possible route to visit each city exactly once and returns to the origin city. In the theory of CSI 4105 Design and Analysis of Algorithms II, This problem is known to be NP-hard, and cannot be solved exactly in polynomial time. Many exact and heuristic algorithms have been developed to solve this problem. Our project is type I, and our team will use two different algorithmic strategies to solve the problem, then compare them under a particular workload.

2 Objective of the Research

The main goal of this project is to implement and compare two different heuristic algorithms for solving the Travelling Salesman Problem, then determine which one yields better results from two aspects:

- Performance: We are interested in seeing whether an algorithm performs better when evaluating different datasets.
- Quality: The total distance is another important feature to evaluate which algorithm will result the most desirable result with different datasets as input.

3 Approach / Methodology

We will implement two different heuristic algorithms for solving the Travelling Salesman Problem:

- Lin-Kernighan algorithm.
- Genetic algorithm.

The algorithms will be implemented using Java. and will be evaluated by running the algorithm several times for each test dataset.

3.1 Brief description of Lin-Kernighan Algorithm and Genetic Algorithm

The Lin-Kernighan algorithm is an heuristic algorithm for the Travelling Salesman Problem. The algorithm starts by creating a hamilton cycle (this is trivial for a complete graph). Then, the algorithms removes k edges, which will -effectively- create k distinct paths, finally the paths will be re-connected by adding k edges such that the resulting route is the optimal with respect to those edges. This operation is repeated until all edges are considered. Note that the values k are defined by the algorithm during each iteration.

Genetic algorithms use the principles of selection and evolution to produce several solutions to a given problem.The most common type of genetic algorithm works like this: a route is created randomly. The cities in the route are then evaluated. Two cities are then selected based on their fitness, the higher the fitness, the higher the chance of being selected. These cities then

"reproduce" to create one or more cities after which the cities are mutated randomly. This continues until a suitable solution has been found or a certain number of generations have passed.

3.2 Implementations and evaluation

Both algorithms will be written in Java, this process should be controlled by GIT. We will design the user interface to select the different algorithm, show the connections between each cities, and the statistic results. In the back end, both algorithms will be implemented in different Java classes, and use a log file or database to keep the record.
For our evaluation, we will execute both algorithms more than 5 times with different size of source data, and record the results (the length of the path) as well as the time it took the algorithm to get the results. In the end, the results of each algorithm will be averaged and compared.

4 Requirements

IDEs	Eclipse
Version Control	GIT
Programming Language	JAVA 1.8
Data Source	National TSP Collection (http://www.math.uwaterloo.ca/tsp/world/countries.html)
Size of Dataset	(1) Less than 100 cities. (2) Greater than or equal to 100 cities, but less than 1000 cities. (3) Greater than or equal to 1000 cities.
Sample Data	1 41800.0000 82650.0000

The data from National TSP Collection is open source and will be downloaded freely.