TJHSST Computer Systems Project Proposal Using Genetic Algorithms to Optimize the Traveling Salesman Problem 2007-2008

Ryan Honig

November 2, 2007

1 Problem Statement

My goal is to create a program that can solve the Traveling Salesman Problem, finding near-optimal solutions for any set of points. I will use genetic algorithms to try to find the optimal paths between the points. In the end, after I create a working alorithm that will find near optimal paths, I hope to create a graphic interface that will display the chosen points and the paths through those points as the algorithm runs.

2 Purpose

The main purpose of my project is to develop my own genetic algorithm that can hopefully find close to optimal solutions for the Traveling Salesman Problem. Once this is done I hope to modify the program to work for asymmetric problems and create a user interface that will graphically display the current problem and run the algorithm to find a solution. Since the main focus of my project is on the Traveling Salesman Problem, I will elaborate on that. The Traveling Salesman Problem is a problem in which a set of points is given and you want to find the shortest path that travels between each point once and then returns to the starting point. A symmetric problem is one in which the distance between towns A and B is the same as the distance

between towns B and A. An Asymmetric problem is one in which the distance between towns A and B is different from the distance between towns B and A. This is a good problem to tackle for the computer systems lab because it is fairly complex and deals both with some complex algorithms and with some higher level math. By finding an efficient and optimal solution to the traveling salesman problem, it can be applied to the larger NP-complete field of optimization problems which can contribute to many fields of study.

3 Scope of Study

With my project, I would like to develop an efficient algorithm that can find near-optimal solutions for both symmetric and asymmetric traveling salesman problems and then incorporate it into a user interface that will run the algorithm and display the paths that the algorithm comes up with. My algorithm will be a mix of basic heuristics and the more complex genetic algorithms. In order to do this, I would need to research more about algorithms that have been used in the past and how the differences between symmetric and asymmetric problems would affect these algorithms.

4 Background

The Traveling Salesman Problem has been around for a long time, but more efficient programs for solving the Traveling Salesman Problems are still being created. Many different algorithms have been used to attempt to solve Traveling Salesman Problems, including heuristics, genetic algorithms, and brute force. Heuristics are the best for finding 'good', but not optimal, paths fairly quickly, while genetic algorithms take longer but find more optimal paths. The paper A Genetic Local Search Algorithm for Solving Symmetric and Asymmetric Traveling Salesman Problems by Bernd Freisleben and Peter Merz details how a good way to create an algorithm for the Traveling Salesman Problem is to use a basic heuristic to find the initial pool of paths and then use the genetic algorithm on this pool of paths to find a near-optimal solution. I hope to build off of this approach by creating an algorithm that will work for both symmetric and asymmetric Traveling Salesman Problems.

5 Procedure and Methodology

In order to meet my goals for this project, I hope to finish working out the bugs with my initial version of my traveling salesman program within the next week or two so that I can begin implementing the heuristic to create my initial pool of paths and incorporate a mutation system within the genetic algorithm so that the pool doesn't get 'stuck' on one single path. By the end of second quarter I want to have my algorithm nearly completed so that I can begin optimizing it and adding in functionality that will allow it to complete asymmetric traveling salesman programs. I would then begin making a user interface to house this algorithm and make it easier to both perform the algorithm and visualize the path. I am currently using C to program my algorithm, but I would like to use C++ to create the User Interface in the end. I have found a great site, which can be found here: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/, which has a database with many traveling salesman problem data sets and their best known solutions. I can use these data sets to test my algorithm and measure the percent error of the paths my program generates from the best known paths to determine how well my program is performing.

6 Expexted Results and Value to Others

I hope that by the end of the project, my algorithm will produce near-optimal solutions for both the symmetric and asymmetric problems in an efficient run time. I will run the algorithm with different sizes of the initial pool and different amounts of times for the genetic crossover to occur. I will organize this data into tables by displaying the percent error of the best path that my program produces for a given problem and the percent error of the average path that my program produces, for a given number of runs, and the average time that it took for the program to run that particular set of data. These results will also be organized by the size of the initial pool and the amount of times the crossover is performed.