Solving the Vehicle Routing Problem for Multiple Multi-Capacity Vehicles

Michael Sanders
TJHSST Computer Systems Lab 2008-2009

Abstract

The Vehicle Routing Problem (VRP) has existed as long as a distributor has needed to deliver items. As such, the VRP has been solved with many different methods, including agent architecture and ant colony modeling. However, these methods have generally been set up for an established organization that has a specific number of vehicles with only a few unique capacities. This project aims to create a program that will identify a very good solution for the VRP, but in a case where all the vehicles could have different capacities. This is the situation faced by some volunteer groups that do not have established vehicle fleets and rely on people volunteering vehicles when something needs to be distributed.

Background

A great deal of research has been done into the VRP and its variants, such as the VRP with Time Windows (VRPTW) and Multi-Depot VRP (MDVRP). Of particular interest to this project are those that deal with solutions making use of genetic algorithms. Several projects have made use of genetic vehicle representation, where each solution has genetic material that represents the solution's routes.

Genetic Algorithms

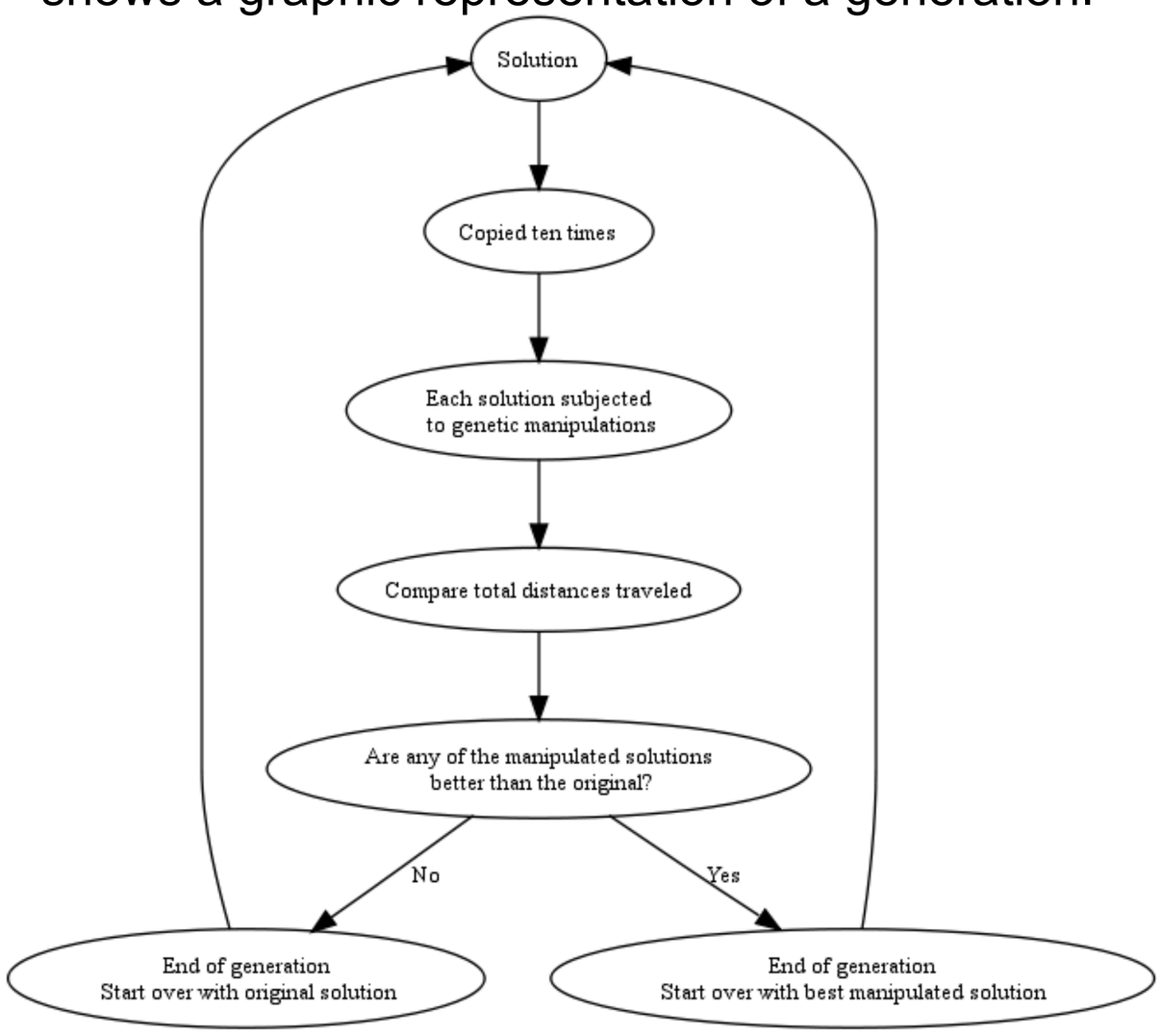
The solution finder implements a genetic algorithm to manipulate Solution objects. Operations will involve removing customers from one route and placing them in another (insertion), switching two customers between two routes (swap), and randomizing routes. Combinations of these operations can be combined to optimize the final solution.

Conclusions and Extensions

A method of using genetic algorithms to solve the VRPMMCV was found. The program can be implemented by adding solutions to the breeding pool and assigning them different combinations of genetic manipulations. For this project, the final version of the program used a breeding pool with two solutions that were each assigned to the insertion operation, as testing had shown this to have the best result of the combinations tested. Further projects can determine methods to optimize the breeding pool's design. Other extensions include optimizing the mutation rate and implementing more genetic operations.

Procedures

The program first reads in all customer and vehicle information. A solution is created by randomly assigning customers to vehicles and routes. The program then proceeds a set number of generations. During each generation, a set of solutions are manipulated by different genetic operators. After all the solutions have been manipulated, their distances are compared with the original distance. The solution with the shortest distance becomes the original solution for the next generation. The flowchart below shows a graphic representation of a generation.



Testing

Each of the different genetic operations were tested singly, in pairs, and then all together. The resulting number is a ratio, found by dividing the total distance traveled on the final solution by the total distance traveled on the original solution; thus, the smaller the ratio, the better solution was found. The results indicated that using only the insertion operation resulted in the best average results.

Run #/ Type of operation	Swap	Insert	Rndm.	Swap+ Insert	Swap+ Rndm.	Insert+ Rndm.	All
1	.921	.833	.930	.890	.933	.884	.901
2	.920	.875	.920	.915	.901	.894	.904
3	.905	.863	.938	.892	.937	.924	.913
4	.935	.862	.945	.850	.890	.944	.929
Avg.	.920	.858	.933	.887	.915	.912	.912