COMSM0302 - GA COURSEWORK

JAMES MARSHALL
OCTOBER 16, 2008

Submission deadline: Midnight Friday 5th December 2008
Late submission deadline: Midnight Monday 8th December 2008
Weighting: 30% of total course marks

1. Problem Description

The problem you will be developing a GA to solve is a Capacitated Vehicle Routing Problem (CVRP). The CVRP is defined by a graph, one vertex of which is a depot; the remainder of the vertices each have a demand level. The CVRP is to find a set of vehicle routes originating from and returning to this depot (i.e *circuits*), which completely services the demand at the other vertices. Each vehicle *must*, on arriving at a vertex, *completely* load all of the demand from that vertex. The capacity of each vehicle must never be exceeded.

The graph on which the CVRP must be solved is a complete, undirected graph, which can be represented by a symmetric weighted adjacency matrix as below. The weights of each edge are the cost for traversing that edge. The objective function for the CVRP is then to minimise the total cost of all the vehicle routes used to satisfy the demand. If any demand is left unsatisfied, or if any vehicle exceeds its capacity, the objective function is undefined. The number of vehicles is a free parameter in the optimisation.

The definition of the problem you will be working on can be downloaded from the unit homepage, along with a description of the datafile format. The problem is based on a complete 76 vertex undirected graph, described as a set of points in 2-dimensional Euclidean space. One of these vertices is designated as the depot, and for all the other vertices an integer demand is specified. An integer capacity for the vehicles is also specified.

Any GA technique covered in the lectures, or that you have read about, can be used in your project. However the use of other search techniques, even if used in addition to the GA, is not permitted.

2. Submission

You should submit a report, and your code, via the online submission system.

The report should be a PDF of no more than 2 A4 sides, with a minimum font size of 10 point. Additional pages over the page limit will not be read, and submissions with a font size smaller than 10 point will not be read at all. The report should include at the top the overall cost of your best solution, a description of the circuit each vehicle follows, and the number of individuals generated to find it (one individual represents a complete candidate solution to the problem, whether valid or invalid). The remainder of the report should briefly explain your choices on population initialisation and size, representation and encoding, selection operators, genetic operators, termination condition, and anything else you consider relevant.

The code can be submitted in a language of your choice, as long as a compiler is provided for it as standard on the Department's Linux lab machines. You must provide a Makefile so that I can build and run the program with the commands

'make' and 'make run'. When run your program should run without further input from the user, and should terminate by printing out the number of individuals (candidate solutions) generated, the best cost solution found so far, and the circuit followed by each vehicle in the form `0->1->2->0` (assuming vertex 0 is the depot). These circuits should also be printed in a file called 'solution.txt', with each circuit on its own line in the format just described.

Your program should also produce a file called 'log.txt' and add a line to this every time a selection is performed, or a genetic operator applied, of the typical form ⟨operator id letter⟩:⟨parent's fitness⟩:⟨offspring fitness⟩. When selection is performed a line should be added beginning with the identifier 'S' and indicating the selected parent's fitness, the population total fitness, and the population size. For the genetic operators 'X' should be used for a crossover operator, and 'M' for a mutation operator. If you have other genetic operators you should assign them a unique identifying letter and explain this in your report. Crossover should include the fitness of both parents with the fitness of one offspring... if crossover generates multiple offspring then additional lines for the same application of the operator should be generated. So, example lines in the log.txt file representing selection of two parents, crossover producing two offspring, and mutation performed on one of the offspring might be

S:7:571:100
S:9:571:100
X:7:9:8
X:7:9:7
M:7:11

The correct formatting of the files 'log.txt' and 'solution.txt' is very important, as most of the marks for this coursework will be based on automated evaluation of its contents. Submissions without these files, with incorrect files, or without functioning Makefiles will be penalised.

3. Marking

Marks will be allocated as follows

Overall report quality	5%
Choice and justification of selection and genetic operators	20%
Empirical performance	75%