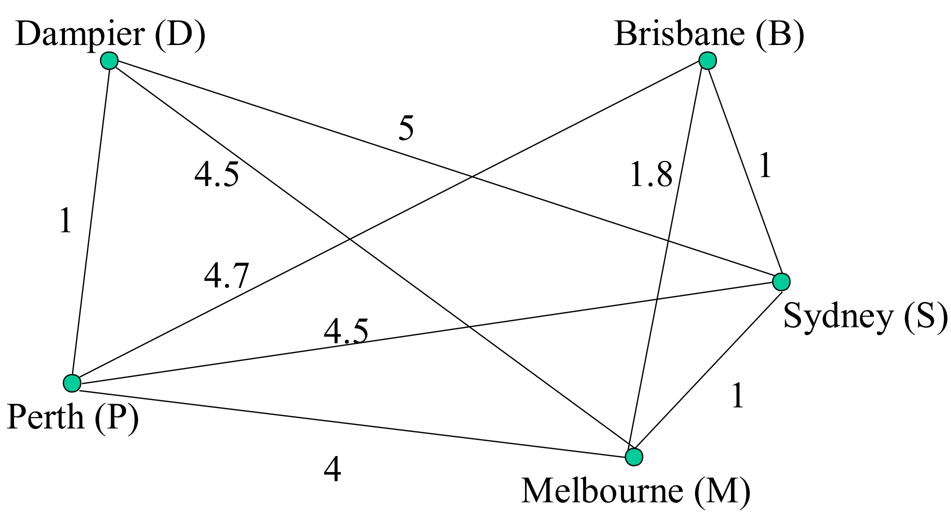
CSG2341 Intelligent Systems

Travelling Salesperson Problem

This is an example of a problem that needs special purpose genetic operators. Solutions to this problem are permutations of a list of cities. You have probably heard of the Travelling Salesman Problem. This is a harder version called the Blind Travelling Salesman Problem! The material in this section is based on (Goldberg, 1989, pp 170-175).

In the Travelling Salesman Problem, you are to imagine a salesman who has a number of cities to visit, and wants to visit each of them once in such an order that his round trip is as short as possible. An ordering of cities to visit is called a tour. Many real scheduling problems, such as scheduling of deliveries or crew rostering, turn out to be equivalent to the Travelling Salesman Problem (TSP). Here is an example instance of the problem:



In this example, there are 5 cities (towns in this case) and we know the distances (in 1000's of km) between each pair of cities (except Dampier and Brisbane, where there is no direct route).

One possible tour is Perth to Melbourne to Brisbane to Sydney to Dampier, then back to Perth. The length of that tour is 4+1.8+1+5+1=12.8. Can you find a shorter one?

In the Blind Travelling Salesman Problem, the task is to find the shortest tour that you can, where your only feedback is the total length of any proposed tour. I.e. you tell me the tour you want and I'll tell you its length, but I won't tell you the distances between the cities.

To tackle this problem with a GA, we need to design an encoding of tours as a chromosome. The first one that you might think of is to number the cities, say 1 to 8 for an 8-city problem, and simply write down the order to visit them. So one chromosome might be 2 3 5 6 1 4 8 7, meaning start at 2, then go to 3, then 5 etc. But how would you perform crossover and mutation on such chromosomes?

Mutation:

There are many possibilities for mutation. We have chosen one that is sometimes called 2-opt. It chooses a pair of cities a pair of cities and reverses the part of the chromosome between them. (Draw a picture to see what effect this has on a tour. Why might this be useful?) For example,

A = 2 3 **5** **6** **1** **4** 8 7
x x

becomes

A = 2 3 **4** **1** **6** **5** 8 7

Crossover:

Here is an example of two-point crossover on two valid chromosomes:

A = 2 3 **5** **6** **1** 4 8 7
x x
B = 7 1 **2** **3** **8** 6 5 4

Gives children

2 3 **2** **3** **8** 4 8 7

7 1 **5** **6** **1** 6 5 4

So the first child would visit cities 2, 3 and 8 twice, and never visit 5, 6 or 1, and the second child vice-versa. This is no good! One solution is to use a special purpose operator, the PMX operator (devised by Goldberg and Lingle), which works like this: The matching segments between the crossover points are used to specify a re-ordering in each parent. In this case, 5 and 2 are swapped in each parent, and the same with 3 and 6, and with 1 and 8, giving:

A’ = 5 6 **2 3 8** 4 1 7

B’ = 7 8 **5 6 1** 3 2 4

The children here have each inherited part of their ordering information from each parent. Other permutation crossover operators have been designed for other problems.

Reference

Goldberg, D. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, 1989.