Tymoteusz Chatys

Simulation 1 – Simplest simulation

1.1 – No weighting

Assumptions:

- 1 Hub
- Hub must be in a city or town
- Truck travels to every city/town and back

The program uses brute force to calculate the best placement of the hub, to reduce the distance travelled.

1.2 – Weighted by population

Assumptions:

- 1 Hub
- Hub can be in a city or town
- Truck travels to every city/town and back
- Cities/Towns with a larger population are weighted more than the others. Therefore, resulting in the hub placement being closer to larger cities and towns.

The program uses brute force to calculate the best placement of the hub, to reduce the distance travelled.

Simulation 2 – Random location

2.1– No weighting

Assumptions:

- 1 Hub
- Hub can be anywhere in the UK
- Truck travels to every city/town and back

The program gets random points, calculates the distance to travel to every city and back to the random point. Then picks the random point with the lowest total distance and begins to hill climb to the only and therefore global maxima.

2.2– Weighted by population

Assumptions:

- 1 Hub
- Hub can be anywhere in the UK
- Truck travels to every city/town and back
- Cities/Towns with a larger population are weighted more than the others. Therefore, resulting in the hub placement being closer to larger cities and towns.

Same as 2.1, however it is weighted by population.

Simulation 3 – Nearest Neighbour

3.1– Nearest Neighbour

An attempt to get as close to the answer for Travelling Salesman as possible, which is impossible to brute force with 10^{150} possible solutions.

Assumptions:

- Hubs can be in a random location
- Truck must travel from the hub, to every city, not returning to the hub until all cities and towns have been visited

3.2– Improvement

Nearest neighbour provides at best a mediocre approximation to the travelling salesman problem. Therefore, the answer from 3.1 is taken and is attempted to be improved upon by applying random changes in the order of movement of the route.
If the new route is shorter than the nearest neighbour route, it is then made the optimal route and random changes keep being applied.

After certain amount of random changes, if the program is unable to find an improved route, the program knows it has arrived at a dead end and begins to look down another route/tree.
This results in the program to go down many different options as there is several ways to improve the shortest route by applying random changes and therefore this algorithm is repeated many times to find the best route.

Found that this technique shortens the answer by a further 5%.

This could be used in the event where each town and city need something distributed by the government and is limited to only one truck.