Project Proposal Team 4

Aron Granberg (arong@kth.se)
Jón Freysteinn Jónsson (jfjon@kth.se)
Mårten Wiman (mwiman@kth.se)
Vidar Palmér (vidarp@kth.se)

September 29, 2017

Description

We have decided to tackle an interesting problem from a previous round of the competition Hash Code created by Google. Hash Code problems are designed to have a very high real world applicability as well as being very hard to solve optimally. The description of the problem follows below:

> The Street View imagery available in Google Maps is captured using specialized vehicles called Street View cars. These cars carry multiple cameras capturing pictures as the car moves around a city. Capturing the imagery of a city poses an optimization problem: the fleet of cars is available for a limited amount of time and we want to cover as much of the city streets as possible.

So in this problem we have a graph representing a city with streets and junctions and a number of cars. We want to cover each street at least once, but it's not beneficial to traverse a street more than once.

We are considering some modifications to the original problem. For example it is more realistic with an infinite time limit and scoring based on how long it takes to cover the entire city. Alternatively we could try to optimize the number of cars used or add other costs: for example having different traversal costs for different streets depending on a simulated time of day (it's probably not optimal to be stuck in queues during rush hour).

Case Study

The input used in the Hash Code competition was a large map of the streets of Paris. We intend to evaluate our algorithms on this test case in order to be able to compare them to solutions written by other people. We also plan to design two more test cases. One of those test cases we are going to make small enough that we can solve it exactly using other techniques. We can then evaluate if our algorithm produces the same result (or at least one with the same score).

Resources

The problem we will study is a generalization of the route inspection problem (also known as the Chinese postman problem), in which there is only one agent and each edge is undirected.

It is known that the route inspection problem is solvable in polynomial time, and we will begin by taking a look at how by reading the article *"Matching Euler tours and the Chinese postman problem"* by *Edmonds, J.* and *Johnson, E.L.*

However, if both undirected and directed edges are allowed then the problem becomes NP-complete. Still, in the article *"A 3/2-approximation algorithm for the mixed postman problem"* the authors *Raghavachari, B* and *Veerasamy, J* show that there exists a polynomial-time algorithm that finds a path through all edges, and which has length at most $\frac{3}{2}$ times the optimum.

One approach that is often used for these kinds of problems is some kind of heuristic based local optimization. The Traveling Salesman Problem (TSP) has some similarities to our problem and since that problem has been extensively studied we are going to try to look at some popular heuristics used for TSP. In particular we are going to look at the article *An Efficient Heuristic Algorithm for the Traveling Salesman Problem* by *Parham Azimi* and *Peyman Daneshvar*.

Planning

We aim for 3 in Im, 2 in Dp, 2 in Br and 1 in An.

At 50 % completion of the project, we aim for the following:

- Implemented a simple visualizer to simplify debugging.
- Constructed two test cases, one of which is very small.
- Implemented a brute-force algorithm to solve small problem instances exactly.
- Studied relevant literature to determine what approaches might be suitable.
- Implemented at least one approximate polynomial-time solution.

At 100 % completion of the project, we aim for the following:

- Implemented a more advanced exact solution, perhaps based on a PDDL solver.
- Implemented at least two different approximate solutions.
- Investigated if local optimizations can be used to improve solutions.
- Analysed the results and written the report.

Division of work:

Reading papers	10ph
Analysis of gathered information	18ph
Implementation	40ph
Analysis results	16ph
Writing report	16ph
Σ	100ph