Linköping Institute of Technology
Department of Mathematics/Division of Optimization
Kaj Holmberg

2017–03–17

Project information

Optimization of realistic complex systems

Project 4: Solving the snow removal problem

1 Information

The task in project 4 is to solve (a simplified version of) the snow removal problem. You will use Vineopt for solving the Chinese postman problem and rural postman problem. The division of work between vehicles should be done by hand and/or with the help of heuristics, implemented in the python code SNOWPLAN.

Fictive scenario: You are hired by the municipality of Linköping for planning the snow removal. A consulting firm have delivered a program, SNOWPLAN, which is claimed to be efficient for planning. Your task is to evaluate SNOWPLAN and at the same time to produce good plans for snow removal. One may allow the programs to compete with each other or cooperate, i.e. to allow one program to improve the solution obtained by the other program, in both directions. The municipality is interested in the following:

- Good snow removal plans and comparison with regard to different time weights.
- A comparison between SNOWPLAN and the result obtainable by Vineopt.
- Discussion of shortcomings/potential of SNOWPLAN.
- Discussion of possibilities of combining Vineopt and SNOWPLAN.
- An offer for the snow removal in Vadstena.

2 Preparations

1. Study the description of the snow removal problem.
2. Study the description of SNOWPLAN.
3. Study relevant parts of the information about Vineopt.
4. Read about local search methodologies and metaheuristics, e.g. simulated annealing.
5. Solve the problem as will be described below.
6. Demonstrate your results by writing a report which can be read both by a person who is not familiar with optimization and a person who knows optimization and interested how you solved the problem. Include illustrations of good divisions of work between vehicles.
7. Optionally, prepare a presentation of the result/report.

3 Postman problems

The Chinese postman problem is described in Holmberg (2010), chapter 10.6. There you find a mathematical model and an optimization method for the problem. (This method is implemented in Vineopt.) The problem is simply to finding the cheapest tour using in each arc in the graph at least once.

The rural postman problem is also described briefly in the same chapter. In this problem not all the arcs need to be used. Instead there is a given subset of arcs that should be used at least once. If this set of " necessary " arcs is connected, the rural postman problem can be solved to optimality in the same way as the Chinese postman problem. (You simply duplicate arcs the cheapest way, so that all nodes get even degree.)

If the " necessary " arcs do not form a connected set the problem is difficult to solve. However, there are relatively effective heuristics that usually find pretty good solutions. (One of them is implemented in Vineopt.)

If there are multiple postmen who share the work, the problem is difficult. You must first decide which arcs each postman will use. When you've done that, you can solve a rural postman problem for each postman. The main question is how to determine the allocation of arc to postmen.

One possibility is to do this by hand in Vineopt. For a real problem the image of the network contains much information. It is often quite easy to make a sensible division of the graph into the right number of parts by viewing the network. (This is how snow removal is planned today.)

Another option is to use local search or a metaheuristic, such as simulated annealing. One thus determines an allocation of the arcs to the postmen, and evaluates the solution by solving a rural postman problem for each postman and summing up costs.

Important for a successful outcome is to define a suitable environment for heuristic. The simplest way is to move a single arc from one postman to another. One may also consider switching arcs, one does not want to change the number arcs per postman. One can also envisage more advanced replacements, such as moving whole the structures. A good subtour can be moved in its entirety between postmen. The difficult thing then is to find these structures.

One may also consider starting with constructive heuristics trying to build up an assignment in a smart way.

4 Snow removal in cities

A general description of the problem is given separately. In this project, we deal with a simplified problem, namely, we do not consider that it takes longer time to turn rather than to drive straight. Furthermore, we consider only normal roads, not pedestrian roads and cycling roads. Finally, we do not consider cleaning of turning places and crossings.

We consider the average speed to be 7.2 km/h. The time to clean a road is estimated in

average to be twice the length of the road divided by the speed. This time includes two or three lines (e.g., middle, right side, left side) as well as cleaning of places for turning and/or crossings. All of this can be seen as one operation that starts in one end of the street and finishes in the other end. With these numbers, this operations takes exactly one second per meter, i.e., it takes l_j seconds to clean street j if it has length l_j.

Under these assumptions, the problem is reduced to a Chinese postman problem, if there is only one vehicle to do all snow removal.

The task is to compare the result of using various number of (identical) vehicles. Clearly, it is faster to let several vehicles to work simultaneously, but one has to pay to use these vehicles. Each used vehicle has a fixed cost (which however does not depend on how much the vehicle is used, only on the fact that it is used). We do not specify a starting point, but assume that vehicles can be transported to any place before our planning starts.

We solve a rural postman problem for each vehicle. The total cost is the sum of the costs for the tours for each vehicle plus a fixed cost for each vehicle. We assume that the fixed cost is 200 for each vehicle and each second of driving costs 1. Therefore the total cost for a solution is the sum of the costs obtained by Vineopt for each vehicle plus number of vehicles times 200.

The solution time is equal to the maximal time (the cost) for a vehicle. To find the best solution one needs to compare the time with the cost. For many vehicles, the cost is high but the time is short, while for few vehicles, the cost is lower but the time is longer. In this project, we consider two scenarios. In one scenario, cost and time are equally important, and in the other one the time is twice as important as the cost, i.e., the objective is the total cost plus two times the times.

4.1 Input data

The street network is available in Vineopt format (as a rural postman problem, undirected graph with arc costs and possibility to define "required" paths). The arc costs are equal to the length of street segments measured in meters.

There is a small network, *colonia*, (student accomodations near märkesbacken) with 24 nodes and 36 links, two slightly larger networks, *skanninge-n* (Skänninge norra), with 62 nodes and 80 links and *atvid-s* (Åtvidaberg södra), with 68 nodes and 95 links, as well as two larger, *studentryd* (part of Ryd where students live), with 387 nodes and 519 links and *vadstena* (a small town with a large castle near lake Vättern), with 581 node and 778 links.

4.2 Computational details

Start Vineopt by `dine 5`, which gives an extended version. (§ toggles the map as background.)

If only one vehicle is used, one can directly use Vineopt to find an optimal postman tour. If the work is divided between several vehicles, one may decide (by hand or using some other way) which streets should each vehicle clean. There are tools in Vineopt to define

node and arc sets. In the menu `Select`, there are several methods to choose sets, e.g., with a rectangular form. The alternative `toggle` means that all marked entries are changed from being selected to non-selected and from non-selected to selected.

Then the selected arcs are used to define a set of arcs, "linkset". Repeating this process, one generates a set of arcs for each vehicle. (Be careful not to forget any arc, because each arc should belong to exactly one set).

When this is done, one can either see all the arc sets (in different colors) or select them one at a time. Then the marked arcs can be made into "required" arcs in a rural postman problem. One can solve such a problem for each vehicle. This approach is however quite time demanding for a large amount of vehicles.

Notice that Vineopt can read and write arc sets in a file. Hence one can create a good division and save it in order to use as a starting solution for SNOWPLAN. One can also read a division generated by SNOWPLAN to look at its representation and maybe even improve it manually.

There is an implementation of SNOWPLAN available in python, with proposed values of the parameters. It might be necessary to adjust them slightly for some of the problems. Some important parameters:
1. How the starting solution is generated?
2. How order and allocation change between the iterations?
3. The cooling parameters. How fast should the temperature decrease?
4. What kind of improvements of the solutions one should do?

All details are described in the documentation about SNOWPLAN.

It is recommended to create a new folder and run both Vineopt and SNOWPLAN there, and keep all files in this folder. SNOWPLAN requires the file `vno_dir.ini` in the folder when running matlab. It is created automatically by Vineopt.

Your task is make SNOWPLAN work as good as possible, which can need adjustment of some parameters. It can be interesting to make tests with a bad starting solution to see how fast the solutions are improved, and tests with a good starting solution to see if there is any improvements at all.

Notice that a method that uses randomization (which SNOWPLAN does) usually give different solutions if run twice on the same problem. Also note that SNOWPLAN must not be run too few iterations, if the idea of simulated annealing should work.

5 Assignments

1. Consider the small test problem *colonia*.

   a) Solve the problem for one vehicle. Give the solution, the total cost and the time required for cleaning.

   b) Find a solution for two vehicles by selecting manually in Vineopt which links should each vehicle use. Try to find a good division. Give the best solution, the total cost and the time required for cleaning.

c) Find a solution for three vehicles the same way as above.

d) Suppose that both time and cost are of the same importance. What solution is the best?

e) Suppose that the time is twice as important as the cost, i.e., the time has weight two and the cost has weight one. What solution is then the best?

2. Consider the problem *colonia*.

   a) Solve the problem for two vehicles using the heuristic SNOWPLAN. Compare the result with 1b.

   b) Solve the problem for two vehicles using the heuristic SNOWPLAN. Compare the result with 1c.

   c) Answer questions 1d and 1e for the new solutions.

3. Solve the slightly larger problem *skanninge-n* (Skänninge norra). Do the same things as in assignments 1 and 2.

4. Solve the problem *atvid-s* (Åtvidaberg södra). Find solutions for 1, 2, 3 and 4 vehicles using SNOWPLAN. Indicate objective function value, total cost and time, as well as the solution time for Snowplan. Find the best number of vehicles when time and cost have the same weight, and when time is twice as important as the cost. Study the best solution/division in Vineopt.

5. Consider the largest problem *vadstena*. The aim is to find a very good solution for 6 vehicles. Use the same weight for cost and time. Apply first SNOWPLAN. Note the time needed for each vehicle, especially note which vehicle takes the longest time. Then start Vineopt with this solution, and try to improve it by moving arcs from one set to another. (Observe that one can illustrate all arc sets simultaneously using different colors for arcs in the graph.) Make several promising improvements and save the corresponding arc sets in a file.

   The read them into SNOWPLAN. Only doing one iteration, yields the objective function value, and shows whether the solution was improved. One can run several iterations in SNOWPLAN to try to improve the solution further.

   Choose the best among the obtained solutions and try further to improve it in Vineopt and then in SNOWPLAN.

6. Voluntary assignment: Do something cool with the problem *studentryd*.