A BRANCH-AND-CUT ALGORITHM FOR THE AMBULANCE LOCATION PROBLEM

1. The ambulance location problem

The input to our problem is:

(1) A set V of locations.
(2) A weight function $w\colon V \to \mathbb{R}_+$ assigning weights to the locations; the weight of a location is a measure of its population.
(3) A distance function $d\colon V \times V \to \mathbb{R}_+$ satisfying $d(v, v) = 0$ for all $v \in V$. Notice d neither has to be symmetric nor satisfy the triangular inequality.
(4) The maximum number $p > 0$ of locations that may be assigned an ambulance.

The goal is to assign ambulances to at most p of the locations. A location is then served by the nearest ambulance, and the objective is to minimize the weighted average response time. More precisely, we want to find a set $A \subseteq V$ with $|A| \leq p$ that minimizes

$$\sum_{u \in V} w(u) \min\{\, d(u, v) : v \in A \,\}.$$

2. Tasks

In this project we study the ambulance location problem. The aim is to study linear programming relaxations and heuristics for this problem and to combine them in a branch-and-cut algorithm. The project consists of the following tasks (all tasks should be delivered through Brightspace):

(1) Formulate the problem as an integer programming problem. Formulations for the problem will then be discussed during class, giving you the possibility to change your formulation before continuing. You should deliver a PDF file with your formulation showing that it is indeed a formulation. *Due date:* 24/9.
(2) Write a program that uses an integer programming solver like CPLEX or Gurobi to solve small instances of the problem and perform an empirical test of the quality of your formulation by comparing the integer optimal with the optimal value of the linear programming relaxation.

    Your program should, given an instance of the ambulance location problem, use CPLEX or Gurobi to find the integer optimal. It then should find the optimal value of the linear programming relaxation and show both optima for comparison.

    You should submit your complete code with instructions on how it should be used. These instructions should include the programming language used. *Due date:* 11/10.
(3) Improve the bound given by the linear programming relaxation from item (1) either by using alternative relaxations or adding cutting planes.

    You should write a program that, given an instance of the problem, solves the linear programming relaxation in (1) and the improved relaxation you devised for this task. Your program should show the optimal values of both the original and improved relaxations.

You should submit a PDF file with the extra inequalities you used or a description of the other relaxations used, together with the code of your program. *Due date:* 25/10.

(4) Develop one or more heuristics to find primal solutions, that is, feasible assignments of ambulances.

You should write a program that, given an instance of the problem, runs one or more heuristics to find a feasible primal solution. Your program should then show the solution and its objective value.

You should submit the code of your heuristics together with instructions on how to use it. *Due date:* 26/11.

(5) Combine the dual bounds (linear programming relaxation) you developed together with the primal heuristics in a branch-and-cut algorithm for the ambulance location problem. As part of this final task, you should write a report with a maximum of 3 pages discussing the five tasks above; your report should include all the references you used.

Your program should get as input an instance of the ambulance location problem and solve it using a branch-and-cut approach, showing at the end the optimal integer solution and its objective value.

You should submit the complete code of the project, together with the report. *Due date:* 20/12.

3. Practical matters

Here are some other points to keep in mind:

- You should work in pairs.
- All the code you produce for any of the tasks should be handed in. You can write code in C, C++, Python, Matlab, or AIMMS.
- You will need to use some integer/linear programming solver. You should pick one from either CPLEX or Gurobi.

Test instances for the problem can be found in Brightspace. Each instance is given in a text file in the following format, and your programs should accept instances in this file format:

- The first line contains 2 space-separated numbers: the number of locations n and the maximum number p of locations that can be assigned an ambulance. Then the set of locations is $V = \{1, \ldots, n\}$.
- The second line contains n space-separated numbers giving the weight of each location.
- The following n lines (number them 1, …, n) each contains n space-separated numbers. The numbers on line $i \in \{1, \ldots, n\}$ give the distances $d(i, 1)$, …, $d(i, n)$ from i to each location.