Graph optimization project

Luca Ferraro (10748116), Bernardo Camajori Tedeschini (10584438),
William Colombo (10537311)
https://github.com/LucaFerraro/Graph_optimization_project

A.Y. 2020 - 2021

Contents

List of Figures

1 Introduction

The aim of this project is solving an integer linear programming (ILP) problem by using different strategies.
The problem that we have to so solve is the one described in Figure 1.

Consider a directed graph $G = (N, A)$, where each node can host a facility. A set of requests for treatment K is given: each request k originates in a node $o_k \in N$ and has a demand for treatment d_k. The whole demand of k must be treated by one and only one facility, and a single path must be selected from the origin node o_k to the facility to which k is assigned.

Opening a facility in node $i \in N$ costs c_i and a facility opened in i can treat an overall demand cap_i.

Each arc of the graph has a limited capacity uu. Arc capacity is the same for all the arcs. For each demand that uses arc (i, j) a cost g_{ij} must be paid.

Many requests can originate in the same node but they can be assigned to different facilities and can be routed on different paths even if they are assigned to the same facility.

The problem consists in deciding where to open the facilities, to which facility each request is assigned, and the routing on a single path of each request, with the aim of minimizing the sum of the costs.

Figure 1: Problem description

In the following section, we describe all the methods that we have implemented to solve this problem.

2 Problem formulation and continuous relaxation

In this section, we provide the ILP formulation of the problem and its continuous relaxation.

2.1 Variables for ILP formulation

The first thing we have to decide is which variables we need.
Since we have to take three decisions, we need 3 variables.

1. $y_i \in \{0,1\}$: this is the location variable, which assumes value 1 if a facility is installed in node $i \in N$ and 0 otherwise.

2. $z_{k,i} \in \{0,1\}$: this is the assignment variable, and is $= 1$ if the demand $k \in K$ is served by node $i \in N$ and 0 otherwise.

3. $x^k_{i,j} \in \{0,1\}$: this is the routing variable, and it assumes value $= 1$ if the demand $k \in K$ uses the arc *(i,j)* $\in A$ and 0 otherwise.

2.2 ILP formulation

Here, we provide the ILP formulation of the problem.
In the formulation, the flow conservation constraint (the third one) has been written as the difference between the flow entering a node and the one exiting the same node so that at the second member of the constraint we can use directly the variable $z_{k,i}$ to distinguish between the cases in which the difference has to be 0 or 1. Indeed, if $z_{k,i}$=0, then node i is not serving demand k, that is the node is a transit node for the demand k and therefore the difference has to be 0; vide versa, if $z_{k,i}$=0, then node i is serving demand k, that is the node is the destination node for the demand k and therefore the difference has to be 1.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i \in N} c_i y_i + \sum_{k \in K} \sum_{(i,j) \in A} g_{i,j} x^k_{i,j} \\
\text{subject to} \quad & \sum_{k \in K} d_k x^k_{i,j} \le uu & \forall (i,j) \in A, \\
& \sum_{k \in K} d_k z_{k,i} \le y_i cap_i & \forall i \in N, \\
& \sum_{(j,i) \in A} x^k_{j,i} + \sum_{(i,j) \in A} x^k_{i,j} = \begin{cases} -1, & i = o_k \\ z_{k,i}, & otherwise \end{cases} & \forall i \in N, k \in K, \\
& y_i \in \{0,1\} & \forall i \in N, \\
& z_{k,i} \in \{0,1\} & \forall k \in K, i \in N, \\
& x^k_{i,j} \in \{0,1\} & \forall (i,j) \in A, k \in K
\end{aligned}
$$

2.3 Continuous relaxation

The formulation for the CR of the problem is the same as for the ILP. The only difference is in the domain of the variables that now are not binary but continuous and can assume all values in [0,1]:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i \in N} c_i y_i + \sum_{k \in K} \sum_{(i,j) \in A} g_{i,j} x_{i,j}^k \\
\text{subject to} \quad & \sum_{k \in K} d_k x_{i,j}^k \leq uu & \forall (i,j) \in A, \\
& \sum_{k \in K} d_k z_{k,i} \leq y_i cap_i & \forall i \in N, \\
& \sum_{(j,i) \in A} x_{j,i}^k + \sum_{(i,j) \in A} x_{i,j}^k = \begin{cases} -1, & i = o_k \\ z_{k,i}, & otherwise \end{cases} & \forall i \in N, k \in K, \\
& 0 \leq y_i \leq 1 & \forall i \in N, \\
& 0 \leq z_{k,i} \leq 1 & \forall k \in K, i \in N, \\
& 0 \leq x_{i,j}^k \leq 1 & \forall (i,j) \in A, k \in K
\end{aligned}
$$

3 Lagrangian relaxation

We can use the Lagrangian relaxation (LR) since we have agreed that the integrality property does not hold, so we can in theory obtain a lower bound that is better than the one found with the continuous relaxation.
The constraint we have chosen to relax is the node capacity constraint (the second one in formulation in Section 2): since we have one of those constraint for each $i \in N$, the constraint is inserted in the objective function though multipliers λ_i.
The new formulation then is:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i \in N} c_i y_i + \sum_{k \in K} \sum_{(i,j) \in A} g_{i,j} x^k_{i,j} - \sum_{i \in N} \lambda_i (y_i cap_i - \sum_{k \in K} d_k z_{k,i}) \\
\text{subject to} \quad & \sum_{k \in K} d_k x^k_{i,j} \leq uu && \forall (i,j) \in A, \\
& \sum_{(j,i) \in A} x^k_{j,i} + \sum_{(i,j) \in A} x^k_{i,j} = \begin{cases} -1, & i = o_k \\ z_{k,i}, & otherwise \end{cases} && \forall i \in N, k \in K, \\
& y_i \in \{0,1\} && \forall i \in N, \\
& z_{k,i} \in \{0,1\} && \forall k \in K, i \in N, \\
& x^k_{i,j} \in \{0,1\} && \forall (i,j) \in A, k \in K
\end{aligned}
$$

4 Valid inequalities and cutting planes

5 Randomized rounding

For applying the randomized rounding approach we consider the continuous relaxation of the problem as described in Section 2 but with the addition of a more explicit consistency constraint relating variables $z_{k,i}$ and y_i:

$$z_{k,i} \leq y_i, \forall k \in K, i \in N \tag{1}$$

By adding this constraint, we can apply the randomized rounding procedure only two times, one to the $z_{k,i}$ variables and then to the $x^k_{i,j}$ variables. In fact, the the solver will automatically fix the value for the variables y_i in dependance of the values assigned to the $z_{k,i}$ by the randomized rounding.

6 Greedy

The idea of our greedy is looking at one demand at a time and assign try to assign it to a facility that is already opened and that can serve the demand.
Of course, at the beginning there is no opened facility, so we have to decide which one to open; similarly, once a facility Our decision is to open the facilities in non-decreasing order of opening cost.

The scheme for the greedy then is:

1. For each demand $k \in K$, check if there is a facility that is opened and that can serve the current demand:

    - If yes, compute the shortest path to the first found facility by keeping into account the arc capacity (that is added to the shortest path problem as a constraint) and assign the demand to that facility by routing it on the path found.
    - If no, open as new facility the one whose opening cost is minimum, assign the demand to this facility and route the demand to it on the shortest path (again, keeping into account the arc capacity).

Note that with this algorithm we are able to find a feasible solution in a number of iterations that is at most equal to the number of the demands

7 Local search - K-opt neighborhood

We have decided to implement a K-optimum neighborhood.
The initial solution for the algorithm is the one computed by our greedy algorithm.
All what we have to do is writing the ILP formulation of the problem with the addition of the K-opt constraint

$$\sum_{i \in N-Y} y_i + \sum_{i \in Y} (1-y_i) + \sum_{(i,j,k) \in (A \times D)-X} x_{i,j,k} + \sum_{(i,j,k) \in X} (1-x_{i,j,k}) + \sum_{(k,i) \in (D \times N)-Z} z_{k,i} + \sum_{(k,i) \in Z} (1-z_{k,i}) \leq K \tag{2}$$

Note that here, the ensembles X, Y and Z are the ones containing respectively the variables $x_{i,j,k}$, y_i and $z_{k,i}$ that are $= 1$ from the greedy.
For the value of K, we have made many attempts and we have set it equal to the number of variables $/50$.