Project: The Clustered Team Orienteering Problem with Services Sequence

Prof. Renata Mansini, Roberto Zanotti, Alessandro Gobbi

Algoritmi di Ottimizzazione: a.a. 2016/2017

1 The Problem

The name Orienteering Problem (OP) originates from the sport game of orienteering. In this game, individual competitors start at a specified control point, try to visit as many checkpoints (nodes) as possible and return to the control point within a given time frame. Each checkpoint has a certain score and the objective is to maximize the total collected score. The Team Orienteering Problem (TOP) is an OP where the goal is to determine several paths, each limited by a total traveling time T_{max}, that maximize the total collected score. The TOP corresponds to playing the game of orienteering by teams of several persons, each collecting scores during the same time span. We consider a slightly modified version of the TOP in which checkpoints are grouped together in clusters.

Let us consider a set $N = \{1, ..., n_{max}\}$ of nodes and a collection $\mathcal{C} = \{C_1, ..., C_{c_{max}}\}$ of ordered sets of nodes, called clusters. Given two nodes i and j, with $i \neq j$, we define t_{ij} as the time needed to travel between node i and node j. Note that, in general, $t_{ij} \neq t_{ji}$. Each cluster $C_c \subseteq N, c = 1, \ldots, c_{max}$, is a subset of the set of nodes N, with $C_c \cap C_l = \emptyset$, $\forall c, l = 1, \ldots, c_{max}, c \neq l$. A profit $p_c, c = 1, \ldots, c_{max}$, is associated with each cluster C_c.

We introduce a set $V = \{1, \ldots, v_{max}\}$ of vehicles. In order to collect the profit p_c associated with cluster c, each node $i \in C_c$ has to be visited by exactly one vehicle. It is not required that all the nodes in a cluster are visited by the same vehicle.

Let us also consider a set $S = \{1, ..., s_{max}\}$ of service types. Exactly one service type is associated with each node (except for nodes 0 and $n_{max}+1$, which are not associated with any service type). We define constant q_{is}, which is equal to 1 if service type s is associated with node i, 0 otherwise. In order to consider a node i as visited, the associated service, that has a duration $d_i, i \in N$, has to be provided by a vehicle.

As we already mentioned, each cluster $C_c, c = 1, \ldots, c_{max}$, is an ordered set of nodes. As a consequence, the order in which the nodes in the cluster has to be visited is determined by the position that each node holds in the cluster. For cluster C_c, constant w_{ij} indicates whether node i must be visited before node j, $i \neq j, i, j \in C_c$. It is equal to 1 if i must be visited before j, 0 otherwise.

Furthermore, we define $S^V_v \subseteq S, v \in V$, as the subset of service types that a vehicle is allowed to provide. A service type s can then be provided by vehicle v if and only if $s \in S^V_v$. For the sake of simplicity, we define constant a_{iv} that is equal to 1 if vehicle v can provide the service required by node i, 0 otherwise. A starting node 0 and an ending node $n_{max}+1$ are defined, and they are the same for each vehicle. Each vehicle $v \in V$ starts from node 0 at time 0 and must arrive in node $n_{max}+1$ before time T_{max}. In order to simplify the notation, we define set $N^+ = N \cup \{0\}$ as a set of nodes that contains all the nodes in N plus the starting node 0. The goal is to the determine a path (i.e. an ordered set of nodes that starts at 0 and ends at $n_{max}+1$) for each vehicle, such that the total collected profit is maximized. Each path is subject to specific time restrictions, which means that the sum of the travel times between the nodes belonging to the path and the service times of the visited nodes in the path has to be lower than T_{max}.

2 Notation

$C = \{C_1, \dots, C_{c_{max}}\}$	Set of clusters
$S = \{1, \dots, s_{max}\}$	Set of services
$N = \{1, \dots, n_{max}\}$	Set of nodes
$N^+ = N \cup \{0\}$	Set of nodes that includes the starting and ending node
$V = \{1, \dots, v_{max}\}$	Set of vehicles
T_{max}	Maximum time available for each vehicle
a_{iv}	$\begin{cases} 1 & \text{if node } i \text{ requires a service that can be performed by vehicle } v \\ 0 & \text{otherwise} \end{cases}$
p_c	Profit associated with cluster c
w_{ij}	$\begin{cases} 1 & \text{if node } i \text{ must be visited before node } j \\ 0 & \text{otherwise} \end{cases}$
t_{ij}	Travel time between node i and node j
d_i	Duration of the service that has to be performed in node i
$\delta^+(i)$	Set of arcs that have node i as the starting node
$\delta^-(i)$	Set of arcs that have node i as the arrival node

3 Variables

$$x_{ij}^v = \begin{cases} 1 & \text{if arc } (i,j) \text{ is traversed by vehicle v} \\ 0 & \text{otherwise} \end{cases}$$

$$z_{ij} \in \mathbb{R}_0^+ = \text{time of arrival in node } j \text{ from node } i$$

$$y_c = \begin{cases} 1 & \text{if all nodes in cluster } c \text{ have been visited} \\ 0 & \text{otherwise} \end{cases}$$

4 Model

$$max \sum_{c \in C} p_c y_c \qquad (1)$$

$$\sum_{(0,i)\in\delta^+(0)} x^v_{0i} = \sum_{(i,n_{max}+1)\in\delta^-(n_{max}+1)} x^v_{i,n_{max}+1} = 1 \qquad \forall v \in V \qquad (2)$$

$$\sum_{v\in V} \sum_{(i,j)\in\delta^+(i)} x^v_{ij} = \sum_{v\in V} \sum_{(j,i)\in\delta^-(i)} x^v_{ji} = y_c \qquad \forall c \in \{1,\dots,c_{max}\}, i \in C_c \qquad (3)$$

$$\sum_{(i,j)\in\delta^+(i)} x^v_{ij} = \sum_{(j,i)\in\delta^-(i)} x^v_{ji} \qquad \forall v \in V, i \in N \qquad (4)$$

$$\sum_{(i,j)\in\delta^+(i)} z_{ij} - \sum_{(j,i)\in\delta^-(i)} z_{ji} = \sum_{v\in V} \sum_{(i,j)\in\delta^+(i)} (t_{ij} + d_i) x^v_{ij} \qquad \forall i \in N \qquad (5)$$

$$z_{0i} = t_{0i} \sum_{v\in V} x^v_{0i} \qquad \forall (0,i) \in \delta^+(0) \qquad (6)$$

$$\sum_{(i,j)\in\delta^+(i)} x^v_{ij} + \sum_{(j,i)\in\delta^-(i)} x^v_{ji} \le 2a_{iv} \qquad \forall v \in V, i \in N \qquad (7)$$

$$z_{ij} \le T_{max} \sum_{v\in V} x^v_{ij} \qquad \forall i \in N^+, (i,j) \in \delta^+(i) \qquad (8)$$

$$w_{ij} \left(\sum_{(k,i)\in\delta^-(i)} z_{ki} + d_i \sum_{v\in V} \sum_{(k,i)\in\delta^-(i)} x^v_{ki} \right) \le \sum_{(k,j)\in\delta^-(j)} z_{kj} \qquad \forall c \in \{1,\dots,c_{max}\}, i,j \in C_c \qquad (9)$$

$$z_{ij} \ge 0 \qquad \forall i \in N^+, (i,j) \in \delta^+(i) \qquad (10)$$

$$x^v_{ij} \in \{0,1\} \qquad \forall i \in N^+, (i,j) \in \delta^+(i), v \in V \qquad (11)$$

$$y_c \in \{0,1\} \qquad \forall c \in \{1,\dots,c_{max}\} \qquad (12)$$

The objective function (1) maximizes the total collected profit. Constraints (2) ensure that exactly v_{max} vehicles exit node 0 and v_{max} vehicles arrive at node $n_{max}+1$. Constraints (3) ensure that node $i \in C_c$ can be visited by a single vehicle and they also impose that, if a node in a cluster is visited by a vehicle, every other node in the same cluster has to be visited by a vehicle. If every node in a cluster c is visited, variable y_c takes value 1. Constraints (4) are introduced to ensure that if a vehicle v arrives in node $i \in N$, the same vehicle leaves node i. Constraints (5) ensure that, if a vehicle $v \in V$ visits node $j \in N^+ \cup \{n_{max}+1\}$ immediately after node $i \in N$, the time elapsed between the arrival in node i and the arrival in node j is

equal to the time t_i needed to provide the service required by node i plus the travel time t_{ij} between node i and node j. Constraints (6) serve the same purpose of constraints (5), but deal with the special case of the starting node. In this case, the time z_{ij} required to reach any node $i \in N$ from the starting node 0 is equal to the travel time t_{0i} between node 0 and node i. Constraints (7) ensure that vehicle $v \in V$ can visit node $i \in N$ if and only if it is able to provide the service required by the node. Constraints (8) ensure that, given a pair of nodes (i, j), variable z_{ij} can take a value greater than 0 if and only if there is a vehicle that travels from node i to node j. Constraints (9) are introduced to ensure that, given a cluster C_c, if a precedence relationship exists between node $i \in C_c$ and $j \in C_c$, then node j can be visited only after node i has been provided with the service it requires.
Finally, constraints (10) impose non negative conditions on the z variables, while constraints (11) and (12) impose binary conditions on the x and y variables.

5 Requirements

You have to achieve the following project requirements:

1. Solution methods:

   a) introduce and implement a meta-heuristic method choosing between a VNS (variants are also allowed), a Tabu Search, a Kernel Search or an Adaptive Large Neighborhood Search, but combinations of the approaches are more than welcome. The developed approach can make use of Gurobi for solving subproblems.

   b) Alternatively to requirement a), you can implement an exact algorithm using Gurobi, possibly identifying valid inequalities that can help the problem solution. Moreover, you can also propose your own mathematical formulation for the problem not necessarily equal to the one that we proposed.

2. Test the proposed algorithm on benchmark instances that will be provided shortly (along with a benchmark value for each instance).

We will provide you with:

- A java class that contains all the information needed to define an instance of the problem.
- A java class that reads a benchmark instance file and creates an instance of the problem.

At the end of the project, the group should provide an executable jar file that takes a time limit (expressed in seconds) (*-t "timeLimit"*) and the name of the instance file to solve (*-f "fileName"*) as arguments. Once the time limit is reached, the executable has to write to disk the best solution available in *sol* format (*model.write("fileName.sol")*) and terminate. We will test the executable on all the provided benchmark instances and on a set of new benchmark instances that has not been made available.

6 Project evaluation

The proposed meta-heuristic (or exact algorithm) will be compared with the benchmark values and, if, on average, the gap between the benchmark values and the objective function values it obtains is lower than a predetermined percentage, the group will be granted access to the oral exam. The oral exam consists in a presentation made be the group followed (or interrupted) by some questions.

At least a week before the presentation, the group should provide a report containing a precise description of each of the previous points (the implemented algorithm should be described step by step or using pseudo code). Moreover, the report should contain a detailed analysis of the computational results obtained. Since a major goal of your research is to study the behavior of the proposed meta-heuristic, a special care has to be devoted to the analysis of the results. Finally, the identification of particular problem properties will be considered as a plus in the project evaluation.

You can freely use any information you might obtain from the web and that can help you in achieving the project requirements.

N.B. Being admitted to the oral exam is not a guarantee that a grade ≥ 18 will be obtained.