Optimal charging station location in a linear cycle path with deviations

Luca Pirolo[1][0009−0007−8861−3897], Pietro Belotti[1][0000−0001−6591−6886], Federico Malucelli[1][0000−0003−0034−1721], Rossella Moscarelli[2][0000−0001−6967−1513], and Paolo Pileri[2][0000−0002−4354−9349]

[1] Dip. di Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Via Ponzio 34, 20133 Milano, Italy
{luca.pirolo,pietro.belotti,federico.malucelli}@polimi.it
[2] Dip. di Architettura e Studi Urbani, Politecnico di Milano
Via Bonardi 3, 20133 Milano, Italy
{rossella.moscarelli,paolo.pileri}@polimi.it

Abstract. Bicycle tourism is on the rise thanks to assisted-pedaling bikes, also known as e-bikes. While pedalling is still required on these bikes, they allow for longer rides through a battery-powered motor that has an autonomy of a few tens of kilometers. Batteries can then be recharged in one to two hours at recharging stations. Due to the waiting time, these stations should be installed at points of interest such as town centers or monuments for the cyclist to explore during recharge.
We consider the problem of installing charging stations (CSs) on a road or trail network in order to minimize the maximum distance between two CSs, subject to a budget constraint. Optimal placing of CSs for bike trail networks constitutes a known class of location problems; we focus on a special case where the graph representing the trail/road network is a *caterpillar* graph whose spine is a cycle path while the leaves are points of interest, connected to the trail via side roads. For this case, we show that the optimization problem can be solved to optimality by a binary search algorithm where a shortest path problem is solved at each iteration.
We apply our approach to find the CS locations on a 210km-long section of the VENTO bike trail in northern Italy.

Keywords: E-bike · Binary search · Shortest path

1 Introduction

Bicycles are the protagonist of a wave of change in mobility that is taking place in several cities across the globe, both big and small. The advent of assisted-pedaling bicycles, or e-bikes [6], which sport a battery-powered electric motor to provide extra riding power, has made this sustainable means of transportation even more enticing for both everyday commuters and tourists, the latter now capable of riding for long hours and enjoying the countryside or even mountain trails without a thorough training.

Leisure riding is the focus of this paper. We consider a long linear cycle path such as VENTO[3] or the Danube cycle path[4]. Several cycle paths like these two usually run along the banks of a river, where stopping is of scarce touristic interest. However, from the main course of the cycle path, one has several possibilities to reach places of interest by making a small detour.

With the rapid growth of e-bike ridership, urban developers and city administrations face the problem of deploying a suitable charging infrastructure. The e-bike charging stations (CSs) should be placed in strategic positions so as to guarantee a coverage of the whole cycle path. However, since recharging a full e-bike requires a non-negligible time, the CSs should be positioned in places where alternative activities are possible, i.e., places with amenities such as restaurants, museums, swimming pools, and so forth. Moreover, the presence of a CS could induce e-cyclists to discover new places and generate positive externalities. One therefore faces the problem of finding the optimal locations of a set of e-bike CSs to optimize a given objective subject to several constraints arising from, e.g., budget and e-bike battery autonomy.

The problem of locating chargers is perhaps of much bigger impact for electric vehicles (EVs), because EVs need to recharge in order to move; several works on locating EV charging stations exist [5,12,19]. Note that the problem of locating e-bike chargers in a touristic setting cannot be compared to that for cars. First, the demand is not fixed as it is moving in the network. Moreover, the fact that riders tend to select their trajectories according to their specific interests generates a variety of different possible itineraries. Finally, while recharging a car requires a large amount of energy given its mass and required autonomy, the amount of energy for an e-bike is comparably negligible and thus does not require powerful grid outlets.

2 Optimally locating charging stations on cycle paths

Consider a problem where a network of bike trails is given as a graph $G = (N, E)$, together with a path $p = (i_1, i_2, \ldots, i_m)$ with $i_k \in N \forall k = 1, 2, \ldots, m$. A distance d_{ij} is associated with every edge $\{i, j\}$ of E. The cost of installing a CS at node i, denoted as c_i, and a budget b are also given.

Given a set S of nodes for installing a CS, we define a *stretch* as a pair $(i_{k'}, i_{k''})$ of nodes from p such that $1 \leq k' < k'' \leq m$, $i_{k'} \in S$, $i_{k''} \in S$, and $i_k \notin S, \forall k : k' < k < k''$. For simplicity, we assume that $i_1, i_m \in S$. The *stretch length* of $(i_{k'}, i_{k''})$ is defined as the total distance on p between $i_{k'}$ and $i_{k''}$, i.e., $\sum_{k=k'}^{k''-1} d_{i_k, i_{k+1}}$. Finally, the *longest stretch* on path p is the stretch whose length is maximum.

The problem we study in this paper consists of finding the location of CSs on the nodes of the network that minimizes the maximum stretch length while ensuring that the total CS installation cost is at most b. We call this problem

[3] https://www.cicloviavento.it
[4] https://www.danube-cycle-path.com

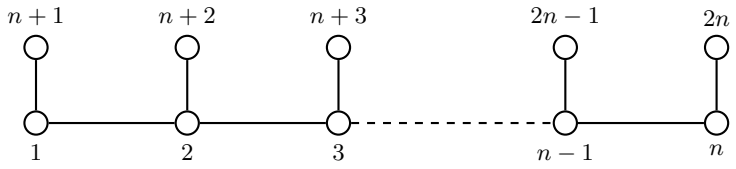


Fig. 1. Caterpillar graph structure: $1, 2, \ldots, n$ are the nodes forming the cycle path, whereas $n+1, n+2 \ldots, 2n$ are areas of interest, where CSs can be installed.

Capacitated MinMax Stretch Problem (CMMSP). The question posed by this problem is the following: in what nodes of the path p should a CS be installed so that (i) installation cost is within budget and (ii) the maximum distance that a cyclist may have to cover, without running out of battery, while entering and exiting the graph at any point (intermediate or not) of the path p, is minimum?

Our contribution is an algorithm for solving a special case of CMMSP: the trail network has the structure of a *caterpillar* graph (see Fig. 1), i.e., a graph that resembles the structure of a cycle path whose each node is connected to one or more nearby points of interest (POIs); second, we assume that (i) none of the POIs are on the cycle path and that (ii) to facilitate leisure activities while recharging the e-bike, a CS should only be installed at a POI, therefore only nodes that do not belong to the caterpillar spine can host a CS. Finally, in order to include the extreme case where a cyclist visits *all* POIs between her entrance and exit nodes, we consider a path p that begins at the first node of the caterpillar graph's spine and ends at the last one, and it visits all intermediate ones. The path p is therefore non-simple.

Because we aim at installing facilities in a subset of nodes and due to the *minmax* objective function, the problem can be classified as a facility location problem where a set of *centers* are sought on a path [3,9,17,18]. In the case where the CS installation cost is equal at every POI, the budget constraint reduces to a cardinality constraint and the problem falls into the class of k-center location [10,13]. The facility location problem literature is vast and k-center location problems on path graphs have been investigated [1,2,11]. Besides similar applications such as the installation of rechargers for EVs, rather than for e-bikes, some literature exists for path graphs with other applications [4]. To the best of our knowledge, our proposed algorithm is new and unlike any approach found in the literature for similar problems.

We formalize the problem in Section 3, where we provide a Mixed Integer Linear Optimization (MILO) model, then provide a reduction to a polynomially solvable problem in Section 4. In Section 5 we use real-world data from the VENTO cycle path and describe how we have created an instance of the CMMSP problem from a general Geographical Information System (GIS) database, which contains geographical data about the whole region rather than just the area surrounding the cycle path. Then in Section 6 we report on computational tests for finding an optimal solution on the same VENTO instance.

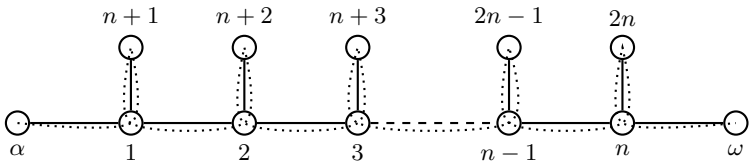


Fig. 2. Secondary caterpillar graph with fictitious source and destination nodes α and ω. The dotted curve is the path p through all POIs.

3 An optimization model

As mentioned above, specializing the CMMSP to a cycle path that is connected to a set of points of interest implies that $G = (N, E)$ is structured as a *caterpillar* graph, as shown in Fig. 1. Let $N = L \cup H$ where $L = \{1, 2, \ldots, n\}$ is a set of nodes that correspond to the locations along the cycle path from which we can deviate, and $H = \{n + 1, n + 2, \ldots, 2n\}$ is a set of nodes that correspond each to the tourist sites that may host a CS. The set of edges E is the union of two subsets: $E' = \{\{i, i + 1\}, i = 1, 2, \ldots, n - 1\}$, i.e., the edges forming the cycle path proper; and $E'' = \{\{i, n + i\}, i = 1, 2, \ldots, n\}$, which form the connection between each point i on the cycle path and a nearby POI.

The distances between consecutive nodes in L (i.e., $d_{i,i+1}, i = 1, \ldots, n - 1$) and the lengths of the deviations ($d_{i,n+i}, i = 1, \ldots, n$) are given. Due to the practical considerations outlined in the previous section, we also assume that the path p that is the basis for stretches is the path between the first and the last cycle path exits 1 and n, touching all POIs in between.

An optimization model can be developed by considering a *secondary* caterpillar graph with two extra nodes called α and ω and with edges $\{\alpha, 1\}$ and $\{n, \omega\}$, both having null distance, as shown in Fig. 2. Path $p = (\alpha, 1, n + 1, 1, 2, n + 2, 2, 3, \ldots, n - 1, n, 2n, n, \omega)$ is indicated with a dotted curve through all nodes from α to ω, traversing each node in $\{1, 2, \ldots, n\}$ twice.

Optimization model: the augmented graph. Before we formulate an optimization model, we introduce an *augmented graph* $G^* = (N^*, A^*)$ as the directed acyclic graph defined with $N^* = \{\alpha, 1, 2, \ldots, n, \omega\}$ and $A^* = \{(i, j) : i, j \in N^*, i < j\}$, where, by a little abuse of notation, we assume $\alpha < i < \omega$ for all $i = 1, 2, \ldots, n$. Define now the *travelling length* l_{ij} on all $(i, j) \in A^*$ as the total distance of an $i - j$ path that traverses once every POI node in between:

$$l_{ij} = \begin{cases} \sum_{h=1}^{j-1} d_{h,h+1} + 2\sum_{h=1}^{j-1} d_{h,n+h} \quad + d_{j,n+j} & \text{if } i = \alpha \\ d_{i,n+i} + \sum_{h=i}^{n-1} d_{h,h+1} + 2\sum_{h=i+1}^{n} d_{h,n+h} & \text{if } j = \omega \\ d_{i,n+i} + \sum_{h=i}^{j-1} d_{h,h+1} + 2\sum_{h=i+1}^{j-1} d_{h,n+h} \; + d_{j,n+j} & \text{otherwise.} \end{cases}$$

Every arc $(i, j) \in A^*$ represents a stretch on G with length l_{ij}. Arc (i, j) is associated a label pair (l_{ij}, f_{ij}), where l_{ij} is defined above while the *arc cost*

f_{ij} is the cost c_{n+j} of installing a CS at the head node j of the arc (i,j), with $f_{i,\omega} = 0 \forall i \in N^*$. An example for a cycle path with $n = 4$ and for edge distances $d_{i,i+1} = 5 \, \forall i = 1, 2, \ldots, n-1$ and $d_{i,n+i} = 2 \, \forall i = 1, 2, \ldots, n$ is provided in Fig. 3, where each arc (i,j) has label (l_{ij}, f_{ij}). Note that in general, for a cycle path with n exit points and consequently n points of interest, the augmented graph contains $O(n^2)$ arcs.

We reduce the CMMSP to the problem of finding an $\alpha - \omega$ path p^* whose total cost, described with arc costs f_{ij}, is below the given budget b and such that $\max_{(i,j) \in p^*} l_{ij}$ is minimum. An optimal solution to CMMSP is then given by all nodes $n + j \in N$ such that $\exists i \in N^* : (i,j) \in p^*$. Let us consider two sample solutions on the graph in Fig. 3, where for simplicity we use c_i in place of c_{n+i}. The first one is defined by the $\alpha - \omega$ path $(\alpha, 2, \omega)$, equivalent to installing a single CS at node 2; its two stretches $\alpha - 2$ and $2 - \omega$ have length $2+2+5+2 = 11$ and $2 + 5 + 2 + 2 + 5 + 2 + 2 = 20$, hence the maximum stretch length is 20 while its CS installation cost is the total travelling cost $c_2 + 0 = c_2$. The second example is path $(\alpha, 1, 2, 4, \omega)$, equivalent to installing a CS at nodes 1, 2, and 4. The stretch lengths are therefore 2, $2+5+2 = 9$, $2+5+2+2+5+2 = 18$, and 2, their maximum being 18, while the total installation cost is $c_1 + c_2 + c_4$. Clearly the second solution has better objective function (maximum stretch length), but it has a larger installation cost which may be infeasible if $c_1 + c_2 + c_4 > b$.

An optimization model can be written with the following variables:

- $\delta \geq 0$: the maximum stretch length, to be minimized;
- $x_{ij} \in \{0, 1\}$: 1 if arc $(i,j) \in A^*$ is used, 0 otherwise.

An optimal solution $(\bar{\delta}, \bar{x})$ to this path problem is then translated to an optimal solution to the CMMSP as follows: install a CS at every node $n + j \in N$ of the original graph if there exists $(i,j) \in A^*$ such that $\bar{x}_{ij} = 1$. Below is a model for the CMMSP:

$$\min \quad \delta \tag{1}$$

$$\text{s.t.} \quad l_{ij} x_{ij} \leq \delta \qquad \forall (i,j) \in A^* \tag{2}$$

$$\sum_{(i,j) \in A^*} f_{ij} x_{ij} \leq b \tag{3}$$

$$\sum_{(i,j) \in \mathrm{FS}(i)} x_{ij} - \sum_{(j,i) \in \mathrm{BS}(i)} x_{ji} = \begin{cases} 1 & \text{if } i = \alpha \\ -1 & \text{if } i = \omega \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in N^* \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i,j) \in A^*. \tag{5}$$

The objective function indicates the *minmax* nature of our problem. Constraints (2) define δ as the maximum stretch length. Constraint (3) requires that the total cost of installing CSs be within budget. Finally, constraints (4) are flow conservation constraints that yield an $\alpha - \omega$ path; we use $\mathrm{FS}(i)$ and $\mathrm{BS}(i)$ for *forward star* and *backward star*, respectively, of a node $i \in N^*$.

The above MILO problem can be solved in polynomial time by another reduction procedure explained in the next section.

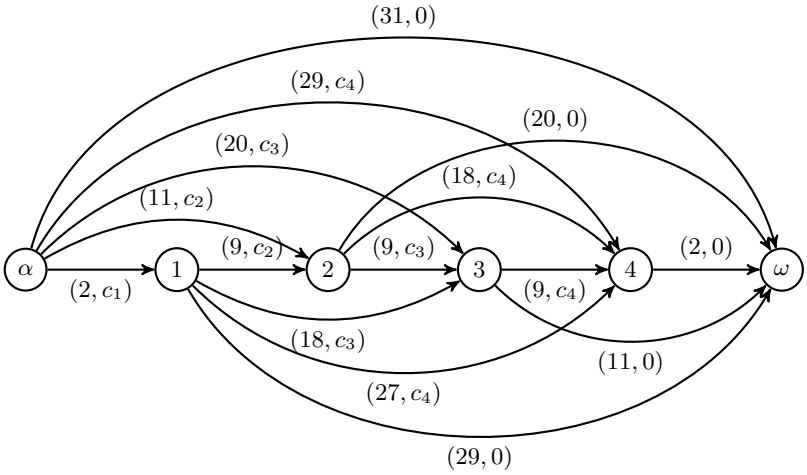


Fig. 3. Augmented graph for a sample caterpillar graph with $n = 4$ and edge distances $d_{i,n+i} = 2$ for $i = 1, 2, \ldots, 4$, $d_{1,2} = d_{2,3} = d_{3,4} = 5$, and $d_{\alpha,1} = d_{4,\omega} = 0$. Each arc has label (l_{ij}, f_{ij}) where l_{ij} is the stretch length and f_{ij} is the cost.

4 Reduction to binary search on shortest paths

Rather than solving the MILO problem (1)-(5) with an off-the-shelf solver, a simple observation allows for solving the problem in polynomial time: for a fixed value $\tilde{\delta}$ of δ, constraints (2) are equivalent to removing all arcs $(i, j) \in A^*$ such that $l_{ij} > \tilde{\delta}$, and the problem becomes equivalent to determining, via a shortest-path algorithm, whether or not there exists an $\alpha - \omega$ path whose total cost, in terms of the f_{ij}, is at most b. A binary search then suffices to find the optimal value of δ.

Denote as $F(G^*(\tilde{\delta}))$ the total cost of the shortest $\alpha - \omega$ path on the graph $G^*(\tilde{\delta}) := (N^*, A^*(\tilde{\delta}))$, where $A^*(\tilde{\delta}) := \{(i, j) \in A^* : l_{ij} \leq \tilde{\delta}\}$ and where the path length is based on arc costs f_{ij} as mentioned above. The case where α and ω are not connected on $G^*(\tilde{\delta})$ is defined with $F(G^*(\tilde{\delta})) = +\infty$. Note that if $b = 0 < \min_{j \in N} c_{n+j}$, then the only feasible solution is $x_{\alpha,\omega} = 1, x_{ij} = 0 \forall (i, j) \in A^* : i \neq \alpha \vee j \neq \omega$, with a stretch length of $\delta = l_{\alpha,\omega}$.

Algorithm 1 describes the binary search for solving CMMSP. Note that there are at most $|A^*|$ suitable values of $\tilde{\delta}$, since δ itself has value in the discrete set of l_{ij} values for $(i, j) \in A^*$. For this reason, the binary search loop requires at most $\log_2 |A^*| \in O(\log(n^2)) = O(2 \log n) = O(\log n)$ iterations, each with a complexity that is at most that of the shortest-path algorithm used.

Algorithm 1 Binary search for solving the CMMSP.

procedure BinSearchCMMSP(N^*, A^*, l, f, b)
  $\delta_{\min} \leftarrow \min_{(i,j)\in A^*} l_{ij}$
  $\delta_{\max} \leftarrow \max_{(i,j)\in A^*} l_{ij}$
  loop
    $\tilde{\delta} \leftarrow \max\{l_{ij} : (i,j) \in A^*, l_{ij} \leq \frac{\delta_{\min}+\delta_{\max}}{2}\}$
    $A^*(\tilde{\delta}) \leftarrow \{(i,j) \in A^* : l_{ij} \leq \tilde{\delta}\}$
    Compute shortest path p on $G^*(\tilde{\delta})$
    if $F(G^*(\tilde{\delta})) > b$ **then**
      $\delta_{\min} \leftarrow \min\{l_{ij} : (i,j) \in A^*, l_{ij} \geq \tilde{\delta}\}$ ▷ Solution over budget
    else
      $\delta_{\max} \leftarrow \max\{l_{ij} : (i,j) \in A^*, l_{ij} \leq \tilde{\delta}\}$ ▷ Seek shorter stretch
    end if
    if $\delta_{\min} = \delta_{\max}$ **then** ▷ Current path defines optimal solution
      $S \leftarrow \{j \in N^* \setminus \{\omega\} : (i,j) \in p\}$
      return $(\delta_{\min}, S)$
    end if
  end loop
end procedure

5 Application to the VENTO cycle path

The VENTO (from VENezia–Venice and TOrino–Turin) is a cycle path about 700 kilometers long that runs along the Po River, between Turin and Venice via Milan: it crosses four regions (Piedmont, Lombardy, Emilia Romagna, and Veneto), 13 districts/provinces and 118 municipalities. In 2016, it was included in the first Italian strategy to realize ten touristic cycle routes, the so-called "Ciclovie Turistiche Nazionali" [14]. Fig. 4 shows the complete map of the cycle path. Note that the structure does not follow that of a single path but (i) it contains a long connecting trail from Pavia to Milan; and, most importantly, (ii) it splits into two parallel trails between Piacenza and Cremona.

For this article, we apply the algorithm described in the previous section to a portion of VENTO ranging from Piacenza to San Benedetto Po (province of Mantua), with a total length of about 210 km. Fig. 5 shows in detail this portion, together with the selected POIs (which are selected based on well-defined criteria set forth within an EU project for this purpose). In order to apply our algorithm, only the southern of the parallel trails between Piacenza and Cremona has been considered, which makes the portion under analysis a path.

We have implemented and interfaced our algorithm to a Geographical Information System (GIS). The data used as input is in the form of three files in the GIS *shapefile layer* format. These files contain:

1. the cycle path: a *linestring* layer (i.e. a line formed as a sequence of segments) that contains the information on the path from the city of Piacenza to San Benedetto Po, but does not contain any node;

Fig. 4. The overall VENTO cycle route, from Turin to Venice.

2. the road network of the region containing the cycle path, defined as a directed graph with nodes and arcs;
3. the eligible destination points, selected with criteria such as artistic/cultural value, presence of bars/restaurants, etc.

All data and software used in this project are Open Source. In particular, data comes from OpenStreetMap [15] (OSM), which is a free and editable web map, built by volunteers and released under an open-content license. Therefore the process of creating the network is replicable and scalable, according to the size of the dataset and the location of the case study.

To deal with the creation of the cycle path graph, we have used QGIS v3.32.3. QGIS [16] is a free and open-source cross-platform desktop GIS application that supports viewing, editing, printing, and analyzing geospatial data using a user-friendly interface. QGIS is an official project of the Open Source Geospatial Foundation (OSGeo).

In order to create a suitable instance graph on which to apply our algorithm, the layers at hand must be manipulated with several algorithms, possibly computationally expensive, which we detail below.

Identify cycleway/network intersections. Given that the cycle path is a linestring with no explicit connection to the road network, we must first transform the cycle path into a path graph that is embedded in the road graph. To this purpose, a *buffer* of 10 meters around the cycleway layer is created. Then the terminal nodes within this buffer are extracted from the road network vector, and these nodes are adjoined to the cycle path, which then becomes a path connected to the road network.

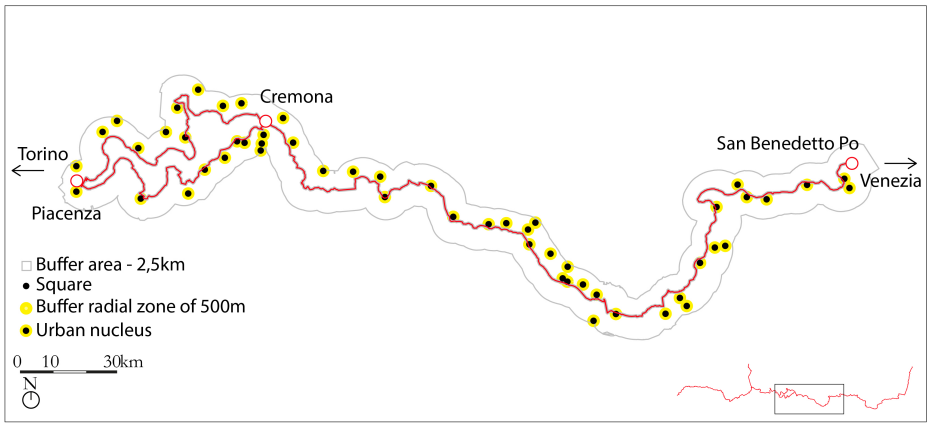


Fig. 5. Portion of cycle path under study with POIs within distance.

Creation of a 2500m buffer area. The application at hand determines through touristic-based criteria that POIs are those that are at most 2500m from the main cycle path. Hence we now restrict our attention to the subgraph induced by all nodes that can be reached via a 2500m-long path from of the cycle path. This is done through the "`service area`" function in QGIS's Network Analysis library, which uses Dijkstra's shortest path algorithm to define the roads reachable with a total weight of 2500.

Identification of nearby POI*s and connection to the cycle path.* A circular buffer with a 500m radius is created around the urban nuclei within the 2500m buffer. Potential POIs, ranked on a set of criteria, are then selected and connected to the main cycle path, using functionality "`v.net.distance`" of QGIS's GRASS system library [7], which uses Dijkstra's shortest-path algorithm.

The resulting graph, depicted in Fig. 6, contains $n = 44$ nodes on the cycle path and about as many nearby POIs. Note that some of the latter is connected to the cycle path through the same exit point: the instance can be adapted by creating as many separate exit points, which have a distance of 0 from one another, and each exit point is connected independently to the corresponding POI.

6 Computational tests

The experimental tests were conducted on a Lenovo ThinkPad with CPU Intel i5-1135G7 clocked at 2.40GHz, with 16GB of RAM and Windows 11 as operating system, equipped with a QGIS installation. All algorithms used for this work were either coded in Python 3.11.5 (binary search) or used Python modules such as `networkx` [8]. In particular, the shortest path algorithm used in each binary

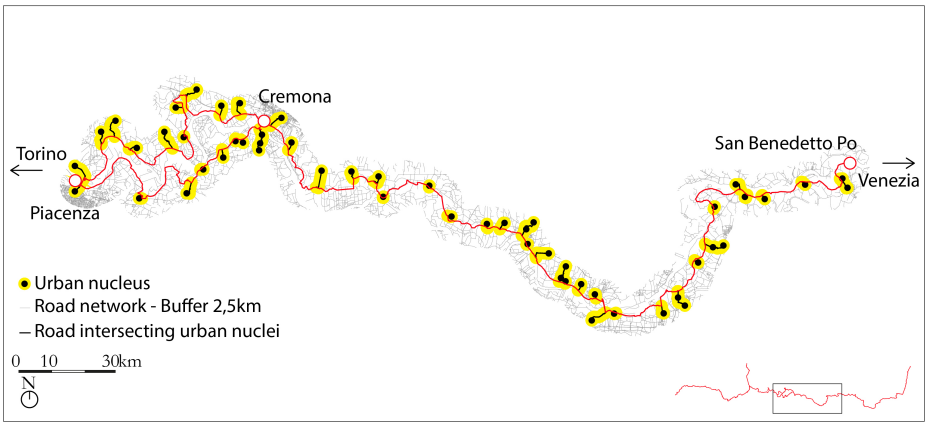


Fig. 6. VENTO cycle path portion with POIs and their connection (only the southern branch of the Piacenza-Cremona portion is considered).

search iteration is an implementation of Dijkstra's algorithm in `networkx`. While CSs can be of several types, for simplicity here we have decided to assign an equal price c_i for all POIs. This translates into a simplification of the budget constraint (3) into a cardinality constraint, and therefore the shortest-path step in our algorithm consists, for these tests, in finding an $\alpha - \omega$ path with the smallest number of arcs.

We now assess the performance of the algorithm on the VENTO instance for budget ranging in $B = \{2000, 4000, 6000, 8000, 16000, 24000, 32000, 64000\}$. For each $b \in B$ we have run the binary search algorithm presented in Section 4, and we report in Table 1 the run time for the binary search, the optimal value of δ and the corresponding optimal set of CSs.

Figs. 7 and 8 report, for all values of $b \in B$, the shortest path (with thick red arrows) and the graph (with thin black arrows) containing all arcs (i, j) with $l_{ij} \leq \delta$ for the optimal value of δ found at the last iteration of the binary search, where $A^*(\delta)$ only contains arcs (i, j) with $l_{ij} \leq \delta$. For $b = 8000$, for instance, the optimal solution $\{7, 16, 37, 36\}$ consists of the POIs Castelvetro Piacentino, Motta Baluffi, Viadana, and Suzzara.

The decreasing run times observed in Table 1 for increasing b could be explained as follows: for large values of b, the binary search mainly updates (i.e., decreases) $\delta_{\max}$, at least at the initial iterations, due to the shortest path algorithm finding a solution with CS installation cost below b. This leads to sparser graphs, as all $(i, j) \in A^*$ with $l_{ij} > \tilde{\delta}$ are removed, and consequently faster runs of the chosen shortest-path algorithm. Small values of b instead yield, at least in the initial iterations of the binary search, an increase of $\delta_{\min}$ and hence comparably slower runs of the shortest-path algorithm.

The run times in the order of a fraction of a second are acceptable for the instance at hand, especially because they solve a long-term decision problem,

Budget	Time [s]	δ_{opt} [km]	Optimal set of CSs
2000	0.31	133.160	$\{22\}$
4000	0.26	89.475	$\{13, 29\}$
6000	0.22	68.395	$\{10, 20, 33\}$
8000	0.20	54.784	$\{7, 16, 27, 36\}$
16000	0.14	33.280	$\{2, 10, 13, 18, 24, 30, 35, 38\}$
24000	0.14	24.102	$\{1, 3, 9, 12, 15, 18, 23, 27, 30, 34, 37, 38\}$
32000	0.12	19.025	$\{1, 3, 7, 11, 12, 14, 17, 20, 23, 27, 29, 31, 34, 37, 38, 40\}$
64000	0.12	18.581	$\{1, 3, 7, 11, 12, 13, 16, 18, 22, 25, 28, 30, 32, 35, 37, 38, 40\}$

Table 1. Run times and optimal value of δ for different values of the budget b.

i.e., that of choosing installation locations for CSs. Nevertheless, the algorithm presented here could be made more efficient especially for small values of b by observing that

- the augmented graph is acyclic, hence a more efficient shortest-path algorithm such as `SPT_Acyclic` could replace Dijkstra's; to the best of our knowledge, `networkx` implements only Dijkstra's and Bellman-Ford algorithms for finding the shortest-path tree;
- we need a single $\alpha - \omega$ path rather than the shortest-path tree, therefore one could use algorithms that stop as soon as ω's label becomes permanent; it might even be interesting to check whether the network simplex algorithm, when minimizing the installation cost $\sum_{(i,j)\in A^*} f_{ij}x_{ij}$ subject to the flow-conservation constraints (4), finds an optimal basis more efficiently.

Hence, for large CMMSP instances on caterpillar graphs an implementation that takes advantage of these observations could further improve on performance and perhaps even reverse the run time trend w.r.t. the values of b. However, the worst-case complexity is $O(\log_2 |A^*|) = O(\log n)$ iterations of the binary search, each solving a shortest-path problem on a directed acyclic graph with complexity $O(|N^*| + |A^*|) = O(n^2)$, for an overall polynomial-time complexity of $O(n^2 \log n)$, which makes our approach scalable to very large instances.

7 Conclusions

We have presented a minmax location problem on caterpillar graphs that finds application in determining the location of e-bike charging stations in the area surrounding a cycle path. The problem admits a Mixed Integer Linear Optimization model via a graph transformation, but we show that it can be solved in polynomial time thanks to a reduction step that yields a solution algorithm consisting in a binary search.

Aside from the theoretical result of finding a polynomial-time algorithm, in practice instances of the CMMSP can be large due to the length of cycle paths, some of which can span hundreds of kilometers and hence have potentially

hundreds of points of interest in their surrounding areas. We apply our algorithm to a real-world case of the VENTO cycle path in Northern Italy and detail the procedure for solving it when starting from a GIS database for the area that contains the cycle path.

Possible research directions include the generalization of CMMSP to the case where only a subset of POIs are visited, and especially the case in which this subset is uncertain. Also, while the results in this paper hold for caterpillar graphs, the problem at hand has a parallel portion that has been only partially considered in our experiments, and it would be interesting to extend our results, if possible, to the case where the cycle path contains loops as is the case for VENTO. In addition, both stretch length and total installation costs are conflicting objective functions and it would be of interest to study a bi-objective optimization version of the problem.

Finally, the general CMMSP is an ongoing research direction. While the MILO model (1)-(5) can be easily adapted to a general bicycle trail network, i.e., on a general non-caterpillar graph G, and to more than one path p, in practice the problem becomes more difficult and we could not find a similar polynomial-time reduction scheme as the one presented here.

Contributions

FM, LP, and PB developed the optimization model and binary search algorithm; RM, PP, and LP identified the criteria for definition of POIs and collected data for the cycle path and surrounding areas; LP implemented the binary search algorithm and performed computational experiments.

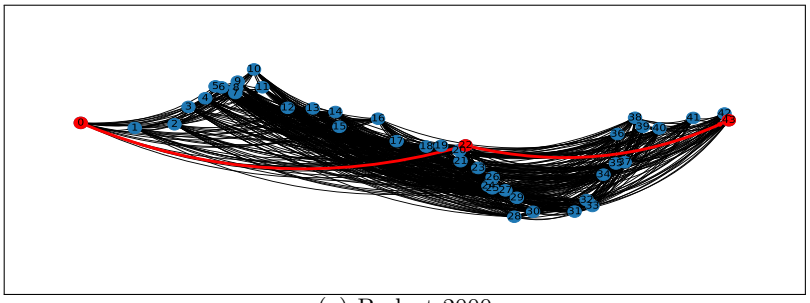
Acknowledgment

This study was carried out within the MOST – Sustainable Mobility National Research Center and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1033 17/06/2022, CN00000023). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

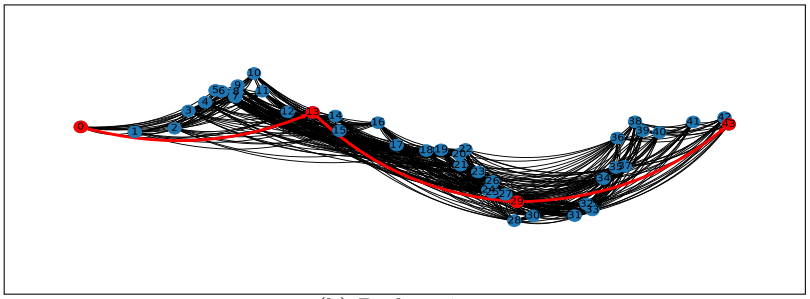
References

1. Ageev, A., Gimadi, E., Shtepa, A.: How fast can the uniform capacitated facility location problem be solved on path graphs. In: International Conference on Analysis of Images, Social Networks and Texts. pp. 303–314. Springer (2021)
2. Bhattacharya, B., De, M., Kameda, T., Roy, S., Sokol, V., Song, Z.: Back-up 2-center on a path/tree/cycle/unicycle. In: Computing and Combinatorics: 20th International Conference, COCOON 2014, Atlanta, GA, USA, August 4-6, 2014. Proceedings 20. pp. 417–428. Springer (2014)
3. Current, J., Daskin, M., Schilling, D.: Discrete network location models. Facility Location pp. 81–118 (2002)

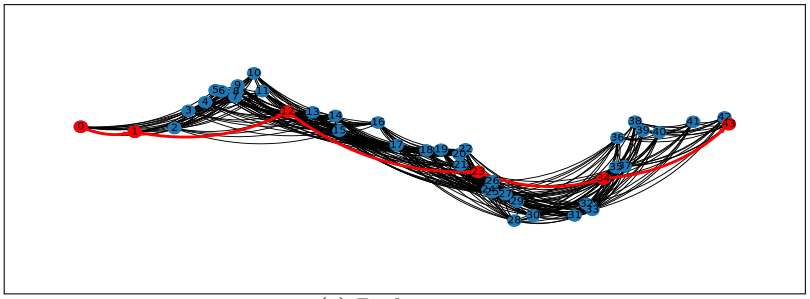
4. Demange, M., Gabrel, V., Haddad, M., Murat, C.: A robust p-center problem under pressure to locate shelters in wildfire context. EURO Journal on Computational Optimization **8**(2), 103–139 (2020)
5. Dong, G., Ma, J., Wei, R., Haycox, J.: Electric vehicle charging point placement optimisation by exploiting spatial statistics and maximal coverage location models. Transportation Research Part D: Transport and Environment **67**, 77–88 (2019)
6. European Union: Regulation (EU) No 168/2013 of the European Parliament and of the Council of 15 January 2013 on the approval and market surveillance of two- or three-wheel vehicles and quadricycles. `https://eur-lex.europa.eu/eli/reg/2013/168/2020-11-14` (2013), accessed on 28/10/2022
7. GRASS Development Team: Geographic Resources Analysis Support System (GRASS GIS) Software, Version 8.2. Open Source Geospatial Foundation (2022). `https://doi.org/10.5281/zenodo.5176030`
8. Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using NetworkX. In: Varoquaux, G., Vaught, T., Millman, J. (eds.) Proceedings of the 7th Python in Science Conference (SciPy2008). pp. 11–15 (2008)
9. Hakimi, S.L.: Optimum locations of switching centers and the absolute centers and medians of a graph. Operations Research **12**(3), 450–459 (1964), `http://www.jstor.org/stable/168125`
10. Hakimi, S.L., Schmeichel, E.F., Pierce, J.G.: On p-centers in networks. Transportation Science **12**(1), 1–15 (1978)
11. Huang, R.: A short note on locating facilities on a path to minimize load range equity measure. Annals of Operations Research **246**(1), 363–369 (2016)
12. Lam, A.Y., Leung, Y.W., Chu, X.: Electric vehicle charging station placement: Formulation, complexity, and solutions. IEEE Transactions on Smart Grid **5**(6), 2846–2856 (2014)
13. Minieka, E.: The m-center problem. SIAM Review **12**(1) (1970)
14. Ministero delle Infrastrutture e dei Trasporti: Decreto ministeriale numero 517 (29/11/2018), `https://www.gazzettaufficiale.it/eli/id/2019/01/22/19A00326/sg`
15. OpenStreetMap contributors: Planet dump retrieved from `https://planet.osm.org` (2023), `https://www.openstreetmap.org`
16. QGIS Development Team: QGIS Geographic Information System. QGIS Association (2024), `https://www.qgis.org`
17. Slater, P.J.: Centrality of paths and vertices in a graph: cores and pits. Tech. rep., Sandia National Labs., Albuquerque, NM (USA) (1980)
18. Tansel, B.C., Francis, R.L., Lowe, T.J.: Location on networks: A survey. part I: The p-center and p-median problems. Management Science pp. 482–497 (1983)
19. Zhu, Z.H., Gao, Z.Y., Zheng, J.F., Du, H.M.: Charging station location problem of plug-in electric vehicles. Journal of Transport Geography **52**, 11–22 (2016)



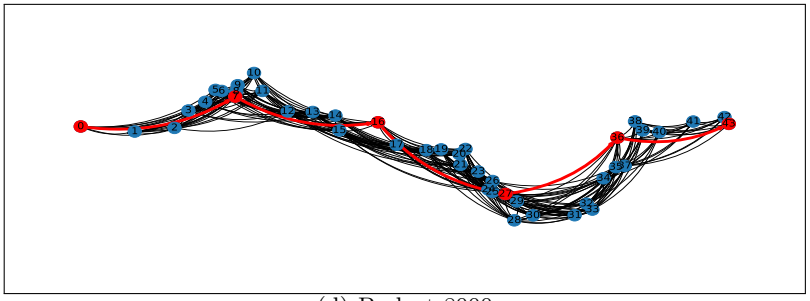
(a) Budget 2000



(b) Budget 4000

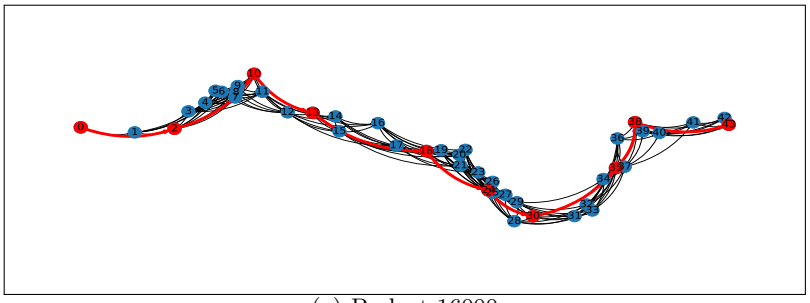


(c) Budget 6000

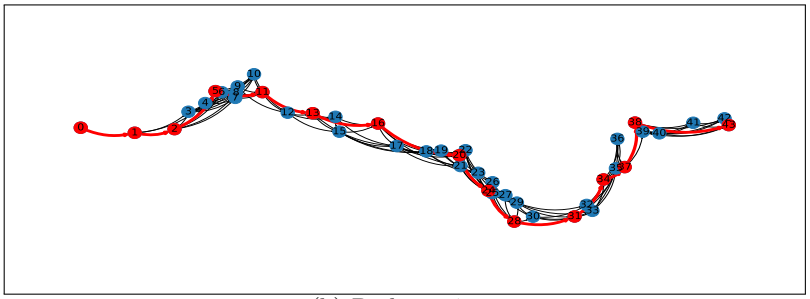


(d) Budget 8000

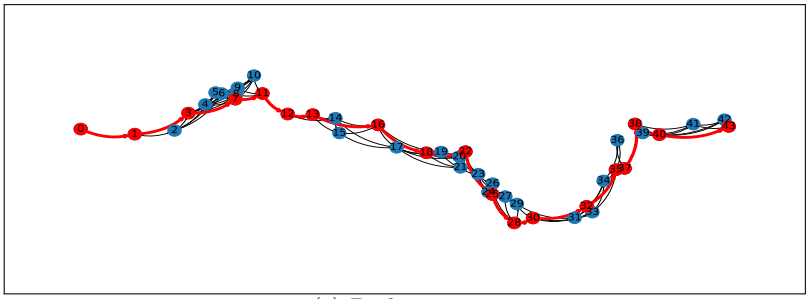
Fig. 7. Graph $G^*(\delta)$ (black arcs) and solution (red) for the optimal value of δ with the given budget for $b \in \{2000, 4000, 6000, 8000\}$.



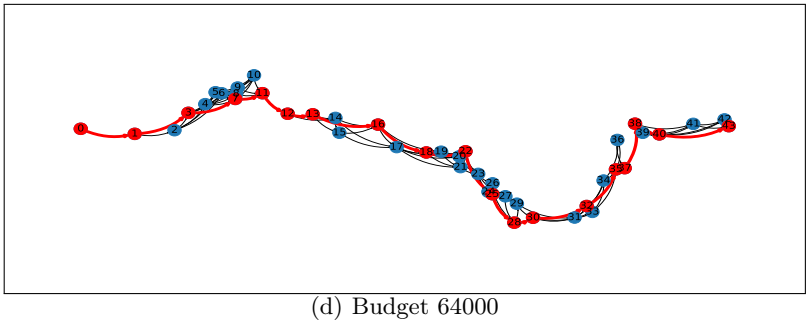
(a) Budget 16000



(b) Budget 24000



(c) Budget 32000



(d) Budget 64000

Fig. 8. Graph $G^*(\delta)$ (black arcs) and solution (red) for the optimal value of δ with the given budget for $b \in \{16000, 24000, 32000, 64000\}$.